Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Control Engineering

Advanced Algorithms for Vehicle Dynamics Control

Dissertation thesis

Ing. David Vošahlík

Ph.D. programme: Cybernetics and Robotics
Supervisor: Doc. Ing. Tomáš Haniš, Ph.D.
Supervisor specialist: Ing. Jaroslav Pekař, Ph.D.

Prague, September 2024

Thesis Supervisor:
Doc. Ing. Tomáš Haniš, Ph.D.
Department of Control Engineering
Faculty of Electrical Engineering
Czech Technical University in Prague
Karlovo náměstí 13
121 35 Prague 2
Czech Republic
tomas.hanis@fel.cvut.cz

Thesis Supervisor Specialist:
Ing. Jaroslav Pekař, Ph.D.
Garrett Motion Czech Republic s.r.o.
V Parku 2326/18
148 00 Praha – Chodov
Czech Republic
Jaroslav.Pekar@garrettmotion.com

Declaration

I hereby declare I have written this doctoral thesis independently and quoted all the sources of information used in accordance with methodological instructions on ethical principles for writing an academic thesis. Moreover, I state that this thesis has neither been submitted nor accepted for any other degree.

In Prague, September 2024

.....
Ing. David Vošahlík

Abstract

This thesis investigates advancements in the control and safety of modern vehicles, focusing on traction control allocation, tire-to-road interface estimation, and vehicle trajectory planning, particularly within the context of electric and self-driving vehicles. First part delves into traction control allocation in over-actuated systems where each wheel is independently powered. Traditional methods for traction allocation often rely on direct force or slip ratio allocation via optimization. This thesis introduces a novel vehicle motion feedback allocation method, which uses vehicle motion references at a vehicle center point (CP) to control wheel pivot points, offering a refined and responsive control mechanism. This method addresses the complexities of traction control by transforming vehicle motion references at the CP to wheel pivot points, enhancing both performance and efficiency. Moreover, it brings many benefits compared to the state-of-the-art methods, such as improved robustness, adaptability, low complexity, etc. To further improve the allocation algorithm performance the challenge of estimating the tire-to-road interface, is addressed in this thesis. Traditional control systems often compromise performance for robustness due to the nonlinear and uncertain nature of tire-to-road interactions. This thesis presents two innovative methods for real-time estimation of the optimal slip ratio (λ_{opt}), which corresponds to the maximum available traction force. The proposed methods include an Unscented Kalman Filter (UKF)-based estimator and a Recursive Least Squares (RLS)-based estimator. These estimators are validated through simulations and real-world experiments. Additionaly, the UKF-based algorithm is used as a labeling tool for a self-supervised neural network training for surface slipperiness predictions use case. Qualitative and quantitative results of the slipperiness estimator based on camera images are shown. Final part of the thesis explores vehicle trajectory planning, a critical component not only for autonomous and self-driving vehicles. It compares two advanced algorithms: Model Predictive Control (MPC) and Minimum Violation Planning (MVP). MPC is known for its application in process control but faces challenges in non-convex environments typical of vehicle navigation. MVP, on the other hand, handles constraints as logical statements transformed into cost functions, ensuring a planned trajectory even in complex non-convex scenarios. This thesis also discusses modifications to MVP to reduce calculation time and compares its performance with MPC in various test scenarios. The results demonstrate MVP's superiority in handling complex planning problems, making it a promising approach for real-time autonomous vehicle operation. The thesis proposes novel methods for traction control allocation, tire-to-road interface estimation, and trajectory planning, addressing current limitations and enhancing the safety, efficiency, and functionality of modern vehicles. These advancements pave the way for future developments in drive-by-wire technology and autonomous driving systems.

Abstrakt

Tato disertační práce se zabývá pokroky v oblasti řízení a bezpečnosti vozidel, zaměřuje se na rozdělování momentu trakce, odhadování parametrů rozhraní mezi pneumatikou a vozovkou a plánování trajektorie vozidel, zejména v kontextu elektrických a autonomních vozidel. První část se věnuje rozdělování momentu trakce v přeaktuovaných systémech, kde je každé kolo poháněno nezávisle. Tradiční metody rozdělování často spoléhají na přímé rozdělování trakční síly nebo podélného skluzu pomocí optimalizace. Tato disertační práce představuje novou metodu rozdělování na základě zpětné vazby z pohybu vozidla, která využívá referencí pohybu vozidla v centrálním bodě vozidla (CP) k řízení bodů zavěšení kol. Tato metoda nabízí jemnější a rychlejší mechanismus řízení. Řeší problém rozdělování trakce transformací referencí pohybu vozidla v CP na body zavěšení kol, což zvyšuje jak výkon, tak i účinnost. Kromě toho přináší mnoho výhod ve srovnání se současnými metodami, jako je zlepšená robustnost, přizpůsobivost, jednoduchost systému, atd. Další část práce se zabývá odhadem parametrů rozhraní mezi pneumatikou a vozovkou, které je klíčové pro řízení dynamiky vozidla. Tradiční řídicí systémy často snižují výkon ve prospěch robustnosti, zejména kvůli nelinearitám a nejistotám v interakcích mezi pneumatikou a vozovkou. Tato kapitola představuje dvě inovativní metody pro odhad optimálního podélného skluzu (λ_{opt}), který odpovídá maximální dostupné trakční síle, v reálném čase. Navrhované metody zahrnují odhadovač založený na Unscented Kalmanově filtru (UKF) a odhadovač založený na metodě nejmenších čtverců (RLS). Tyto odhadovače byly ověřeny prostřednictvím simulací a reálných experimentů. Dále je odhadovačc založený na UKF použit pro anotace při tréninku a validaci samo-učící se neuronové sítě pro predikci kluzkosti povrchu. Jsou zde uvedeny kvalitativní i kvantitativní výsledky prediktoru kluzkosti na základě obrazů z kamery. Závěrečná část disertační práce se zabývá plánováním trajektorie vozidla, což je klíčová součást nejen autonomních vozidel. Porovnává dva algoritmy: modelově prediktivní řízení (MPC) a plánování s minimálním porušením (MVP). MPC je známé pro své využití v řízení procesů, ale je pouze s obtížemi použitelné pro nekonvexní problémy, typické pro navigaci vozidla. MVP naopak zpracovává omezení jako logické výrazy transformované do účelových funkcí, což zajišťuje plánovanou trajektorii i ve složitých nekonvexních scénářích. Tato práce také diskutuje modifikace MVP za účelem snížení doby výpočtu a porovnává jeho výkon s MPC v různých testovacích scénářích. Výsledky ukazují, že MVP je vhodnější pro řešení dynamických a složitých plánovacích problémů, což z něj činí slibný přístup pro provoz autonomních vozidel v reálném čase. Celkově tato disertační práce přináší významné příspěvky do oblasti systémů řízení vozidel. Navrhuje nové metody pro rozdělování momentů trakce, odhad parametrů rozhraní mezi pneumatikou a vozovkou a plánování trajektorie, které řeší současné nedostatky a zvyšují bezpečnost, efektivitu a funkčnost moderních vozidel. Tyto pokroky připravují půdu pro budoucí rozvoj technologií drive-by-wire a autonomních systémů řízení.

Keywords: Vehicle dynamics control, traction control allocation, tire-to-road interface estimation, vehicle trajectory planning, autonomous vehicles, self-driving vehicles, drive-by-wire, control systems, Model Predictive Control, Minimum Violation Planning

Klíčová slova: Dynamika vozidla, alokace trakčního momentu, odhad vlastností rozhraní mezi vozovkou a pneumatikou, plánování trajektorie vozidla, autonomní vozidla, drive-by-wire technologie, řídicí systémy, Model Predictive Control, Minimum Violation Planning

Acknowledgements

I would like to express my gratitude to my supervisor, doc. Ing. Tomáš Haniš, Ph.D., for his guidance, support, and encouragement throughout my research. His expertise and insights have been invaluable in shaping this thesis. I would also like to thank my colleagues and collaborators, especially my supervisor specialist Ing. Jaroslav Pekař, Ph.D., from Garrett motion for their contributions and feedback, which have enriched my professional skills. I would also like to thank my friends and colleagues from Czech Technical University for their support and encouragement. Their feedback and insights have been helpfull in creting this thesis.

Nonetheless, I would also like to thank my family, especially to my wife, for their unwavering support and encouragement. Their support has been a constant source of motivation. Finally, I would like to acknowledge the Czech Technical University in Prague for providing me with the resources and opportunities to pursue my research. This thesis would not have been possible without their support.

List of Tables

List of Figures

Contents

Chapter 1

Introduction

The introduction of this thesis delves into the extensive use of land vehicles such as passenger cars, utility vehicles, trucks, and buses in modern society worldwide. In the United States, where passenger vehicles are particularly prevalent, approximately 87% of the passenger miles traveled in 2020 were in cars and trucks [4]. With the global number of cars increasing alongside population growth, it becomes imperative to explore new methods to enhance the convenience and safety of car usage.

1.1 Safety of the Vehicles

Despite more than a century of automobile history, vehicle safety remains a paramount concern. For instance, in 2022, over 20,000 people lost their lives in road accidents within the European Union (EU) [5]. Nearly half of these fatalities involved drivers or passengers in passenger vehicles [5]. This persistent issue underscores the need for continual improvements in vehicle safety technologies and regulations.

Over the years, authorities have introduced numerous measures to reduce road fatalities, which can be broadly categorized into passive and active safety measures. Passive safety systems are designed to mitigate the adverse effects of collisions, while active safety systems aim to prevent collisions from occurring in the first place. Examples of passive safety systems include seat belts, vehicle designs with deformation zones, and many others. These measures are mandated by regulations and implemented by car manufacturers, requiring drivers only to utilize them, such as fastening seat belts.

Active safety systems, on the other hand, encompass well-established technologies like the Anti-lock Braking System (ABS) and Electronic Stability Control (ESC), as well as newer innovations such as head-up displays, lane departure warnings, and lane-keeping assistants. The contributions presented in this thesis might be categorized as active safety systems. The thesis focuses on exploring the state-of-the art systems and proposes

algorithms having potential to significantly improve road safety and economy, especially for electric vehicles with recently emerging new powertrain architectures.

1.2 Electric Vehicles

In recent years, the shift towards e-mobility has gained momentum due to environmental concerns and efforts to reduce transportation generated emissions. Between 2016 and 2022, the estimated number of electric vehicles (EVs) in use worldwide exceeded 25 900 000 units [6].

Electric vehicles (EVs) are characterized by various powertrain architectures, all sharing the common component of an electric traction motor. The primary architectures of EVs include:

- Battery Electric Vehicle (BEV) – powered solely by an electric motor with energy stored in a battery.
- Hybrid Electric Vehicle (HEV) – powered by both an electric motor and another power source, with energy stored in a battery. HEVs can be further classified into:
  - Mild Hybrid Electric Vehicle (MHEV) – primarily powered by an internal combustion engine, with an electric motor assisting and energy stored in a battery.
  - Full Hybrid Electric Vehicle (FHEV) – powered by both an electric motor and an internal combustion engine, with energy stored in a battery.
  - Plug-in Hybrid Electric Vehicle (PHEV) – similar to FHEV but can be charged from the grid.
  - Extended Range Electric Vehicle (EREV) – an electric motor with an internal combustion engine for extended range, rechargeable from the grid.
  - Fuel Cell Electric Vehicle (FCEV) – powered by an electric motor with energy stored in a hydrogen tank.

According to [6], there has been a notable shift towards Battery Electric Vehicles (BEVs) compared to other types such as plug-in hybrid vehicles. Most modern EV cars are equipped with a single electric motor, but new architectures featuring multiple traction motors (e.g., one per axle or even per wheel) are emerging. These advanced configurations enhance vehicle performance and offer new opportunities for control and optimization. The Chapters 2 and 3 primarily address the control of vehicles with each wheel powered by its own motor, with Chapter 2 focusing on this topic specifically.

The increasing demand for higher efficiency, safety, and advanced functionality in modern vehicle traction systems necessitates complex solutions from both mechatronics

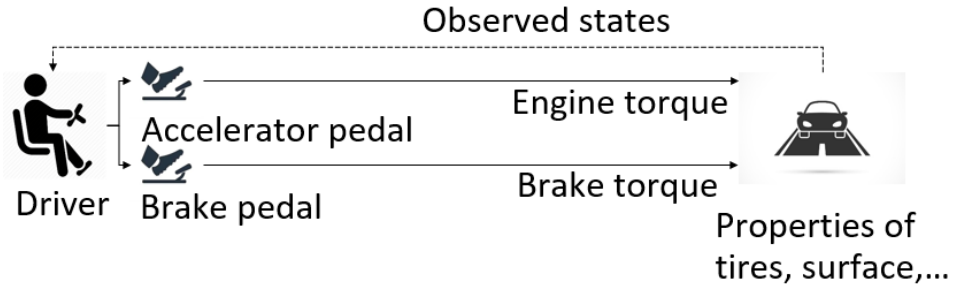


Figure 1.1: Traditional vehicle control method.

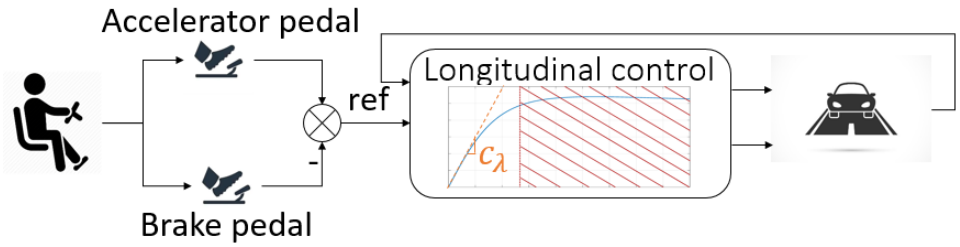


Figure 1.2: Emerging drive-by-wire technology control system architecture.

and software engineering perspectives. The latter is becoming increasingly crucial as vehicles incorporate more sophisticated and complex systems.

Vehicles have evolved significantly since their inception, but the fundamental control mechanisms have largely remained the same: a steering wheel for directional control, an accelerator pedal for torque actuation, and a brake pedal for braking torque (see Fig. 1.1). However, this traditional control concept is being redefined with the advent of drive-by-wire technology [7, 8, 9]. In drive-by-wire systems, the driver is removed from the direct vehicle dynamics feedback loop, instead the driver is setting references for vehicle movement rather than directly controlling the actuators (see Fig. 1.2). The growing complexity and power of modern vehicles, especially electric vehicles (EVs) with four-wheel independent drive, further necessitate the development of new control systems.

Drive-by-wire chassis offer numerous benefits over traditional systems. These systems are over-actuated, meaning they can achieve the same system behavior through different input trajectories. This flexibility is beneficial especially for energy-optimizing control strategies and enhancing vehicle stability and maneuverability. However, the over-actuation also introduces design challenges that must be addressed.

1.3 Self-driving Vehicles

The introduction of new control systems is further driven by the emerging technology of self-driving vehicles. Self-driving technology enables vehicles to operate without human intervention, relying on integrated sensors, actuators, and control systems. Sensors detect

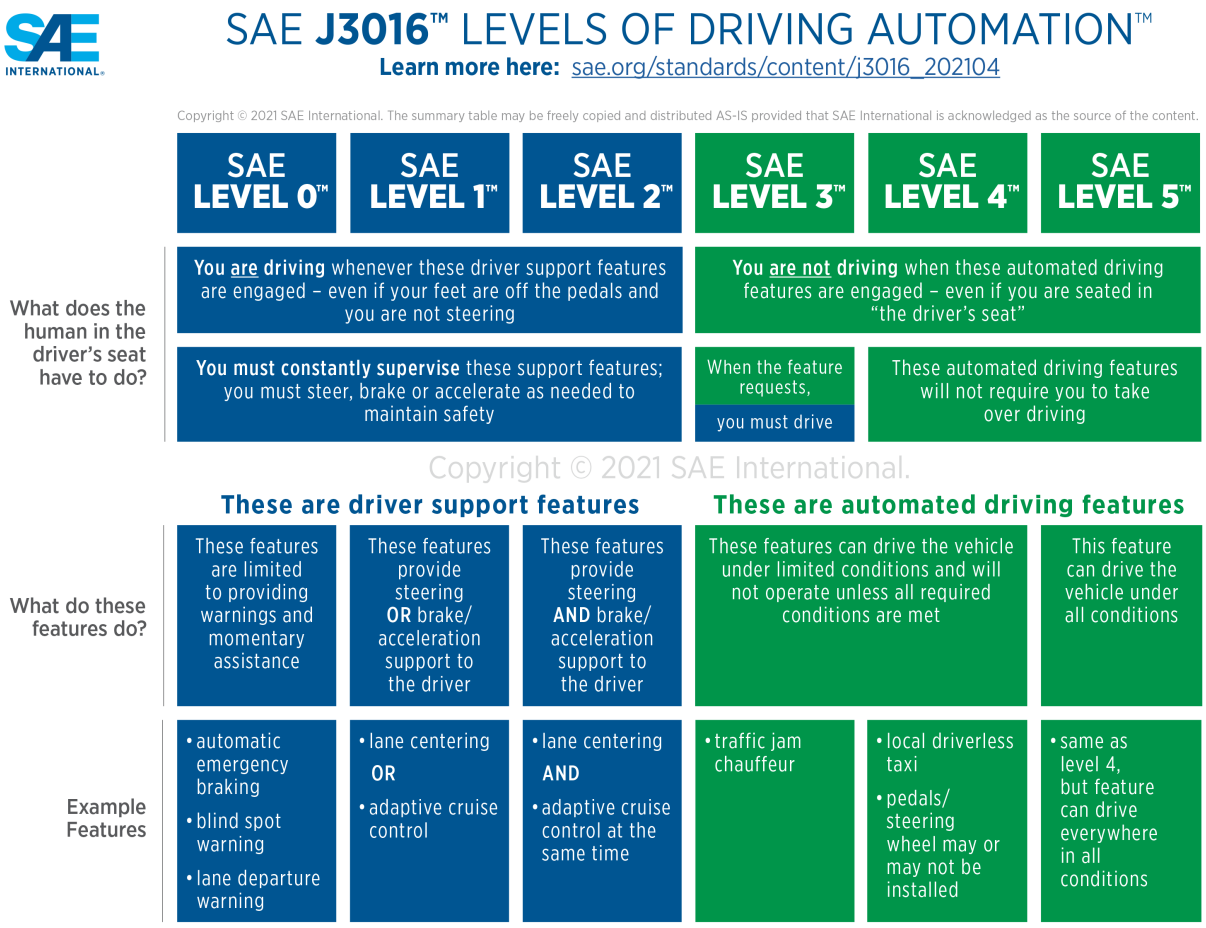


Figure 1.3: Self-driving vehicle autonomy levels. Adopted from [1].

the environment, actuators control vehicle movement, and control systems process sensor data to make movement decisions. Although still in its early stages, self-driving technology holds the potential to revolutionize transportation.

Self-driving technology has been a focus of research and development for some time. In 2014, the Society of Automotive Engineers (SAE) defined six levels of autonomy [1], which are summarized in Fig. 1.3. Further, emerging open-source autonomous driving stacks, such as Autoware [10] and Apollo [11], are promissing platforms for further development and innovation.

Currently, most vehicles in traffic are at best approaching level 3 autonomy under certain conditions. Level 3 autonomy is defined as the vehicle being capable of driving itself, but the driver must be ready to take control at any time. Tesla's autopilot system is one of the most well-known examples of this technology [12].

However, there remains significant room for improvement in self-driving technology. One of the primary challenges lies in the decision-making and trajectory planning modules of the self-driving stack. These components are critical for ensuring the safety and

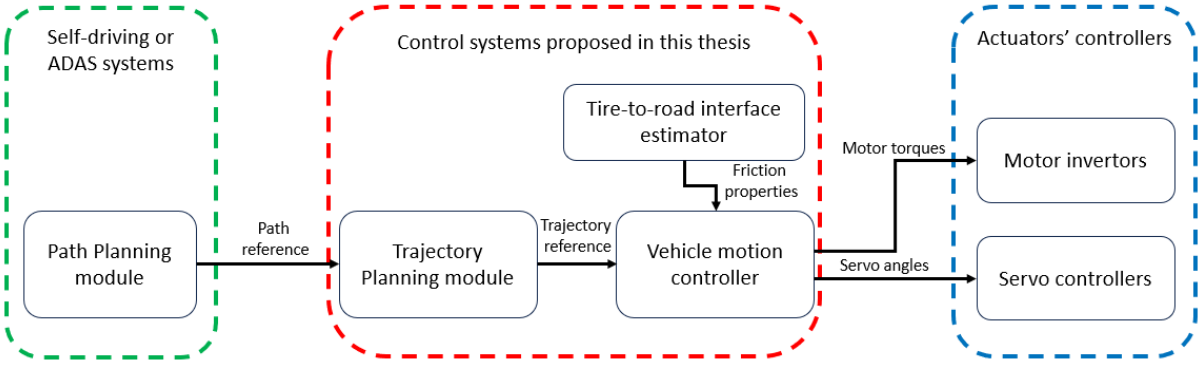


Figure 1.4: Interaction and integration of the proposed components into the vehicle software stack. The green dashed line represents higher-level systems like self-driving or ADAS, the blue dashed line represents lower-level systems like motor and steering servos, and the red dashed line represents the components proposed in this thesis.

efficiency of autonomous vehicles. Chapter 4 is focusing on this topic, comparing two algorithms for trajectory planning and proposing modifications to improve efficiency.

1.4 Dissertation Contribution and Outline

This dissertation aims to contribute to the field of drive-by-wire technology, focusing on three main topics:

- Motion control of vehicles with each wheel powered by its own electric motor, specifically addressing the traction control allocation task with a novel approach.
- Tire-to-road estimation, which is crucial for vehicle control and involves estimating the friction properties between the tire and the road.
- Vehicle trajectory planning, essential for the implementation of self-driving technology.

These topics are interrelated, forming the control layer of modern vehicles. While each component can be implemented independently, the greatest benefits are achieved by integrating all components. Fig. 1.4 illustrates the interaction between the proposed components and their integration into the vehicle software stack. The green dashed line encapsulates higher-level systems like self-driving or Advanced Driver Assistance Systems (ADAS), the blue dashed line encapsulates lower-level systems like motor and steering servos, and the red dashed line encapsulates the components proposed in this dissertation.

Each topic is detailed in a separate chapter of this dissertation.

Traction Control Allocation

The traction control allocation related content in this thesis was published in [13, 14]. The over-actuated, independently driven vehicle faces the traction allocation problem, which involves safely and economically distributing torque to individual wheels. This task is complex due to the ambiguity in allocation, often addressed through direct optimization-based methods like those in [3, 15, 16]. This dissertation proposes a novel vehicle motion feedback controller that addresses the traction allocation task, offering significant advantages over state-of-the-art methods. This controller is discussed in detail in Chapter 2.

Such a controller is essential for implementing self-driving systems and beneficial for human-driven vehicles, helping drivers operate their vehicles more safely and economically.

Tire-to-road Interface Estimation

The vehicle motion feedback controller introduced in Chapter 2 leverages knowledge of the optimal slip ratio for better performance and increased safety. Chapter 3 presents two novel architectures for estimating the optimal slip ratio: an Unscented Kalman Filter (UKF) based estimator and a Recursive Least Squares (RLS) based estimator. The tire-to-road interface estimation related content in this thesis was published in [17, 18].

The UKF-based estimator is inherently capable of estimating the peak of the slip curve (detailed in Section 2.1.4), which is also useful for training a Convolutional Neural Network (CNN) for surface friction prediction. Both estimators and the CNN are validated on a real-world subscale platform, with results presented in Chapter 3.

Vehicle Trajectory Planning

The final chapter, Chapter 4, focuses on self-driving technology and the trajectory planning module. The vehicle trajectory planning related content in this thesis was published in [19, 20]. This chapter describes the interface between the path planning module, which uses a simple model to plan the vehicle's path. However, such a model often neglects many vehicle dynamics, necessitating short-term vehicle state trajectory planning to accurately track the planned path. The chapter compares Model Predictive Control (MPC) and Minimum Violation Planning (MVP) algorithms for this purpose.

Additionally, Chapter 4 discusses modifications to the MVP algorithm that reduce calculation time, demonstrating significant reductions without notable degradation in the solution quality. This work contributes to the development of more efficient and reliable self-driving systems, addressing current limitations in decision-making and trajectory planning.

Overall, this dissertation provides significant contributions to the field of vehicle con-

trol systems, particularly in the context of drive-by-wire technology and self-driving vehicles. By addressing key challenges in motion control, tire-to-road estimation, and trajectory planning, this work aims to enhance the safety, efficiency, and functionality of modern vehicles, paving the way for future advancements in automotive technology.

Chapter 2

Traction Control Allocation

The task of vehicle traction control is inherently complex due to the absence of directly measured or detected wheel traction forces and the challenge of traction force allocation in an over-actuated system, where up to four wheels are driven independently and typically each is individually braked. The control allocation problem is often addressed by optimally distributing traction force, torque, or slip ratio across the wheels, as detailed in the traction control survey paper [21]. Various methods have been proposed to solve this problem, each with its own optimization criteria and approach.

For instance, [15] suggests solving the control allocation through constrained minimization of traction force. Similarly, [3] proposes a solution using a minimum least-squares formulation to minimize the slip ratio for each wheel. Another approach, described in [16], focuses on optimal traction force allocation with different optimization criteria. In [22], the authors extend the method of [3] by incorporating lateral dynamics and lateral force distribution, thus minimizing a combination of longitudinal and lateral forces. [23] offers a different method where the reference traction force for each wheel is computed based on vehicle and wheel parameters, commanded yaw rate, and longitudinal acceleration.

Other optimization-based approaches include [24], which uses a neural network and sliding mode controller for control allocation, and [25], which neglects the tire model and assumes wheel torque will be converted into traction force without loss. This assumption is valid for low slip ratios but falls short during dynamic maneuvers with higher slip ratios where the tire model's influence is significant. Similar assumptions are made in [26] and [27], where the ideal force is computed based on vehicle motion parameters.

The traction force of a particular wheel can be considered a monotonic function of the wheel slip ratio within the stable operating range of the slip curve, given its typical shape (see Fig. 2.3a). Therefore, both traction force and slip ratio allocation essentially follow the same principle of traction force distribution. In contrast to these "direct" allocation methods, a novel motion feedback-based allocation method was introduced in

[13], which will be described in this chapter. Environmental parameters such as wheel normal force/load, tire-to-road interface friction coefficient, and the reduction of longitudinal wheel capacity due to lateral forces can be represented as variations in the slip curve scaling and its parameters (see [28]).

Novel Vehicle Motion Feedback Allocation Method

An innovative alternative to traditional control allocation methods is proposed in [13]. The proposed method relies on the common assumption that either a self-driving algorithm commands vehicle speed and yaw rate, or a driver commands vehicle acceleration and yaw rate at the vehicle's Center Point (CP). The CP is an arbitrarily chosen location where vehicle motion references (e.g., velocity, acceleration, yaw rate) are tracked and respective signals are measured.

The general motion of a rigid body in a plane is fully parameterized by its longitudinal, lateral, and angular speed/acceleration at any arbitrary point on the rigid body. Therefore, using CP variables for the driver-to-vehicle interface is general and lossless. Although vehicle measurement points can be distributed across the vehicle, it is assumed that all vehicle states are acquired (measured or estimated) at the CP. Vehicle velocity and acceleration state variables, references, and measurements can be directly and unambiguously transformed from the vehicle CP to wheel pivot points (for details, see eq. (2.25)).

The wheel-level control system can be designed in a centralized or distributed manner. While this section primarily focuses on traction control allocation, the same control system framework can also be applied to lateral dynamics. Unlike direct allocation schemes, this approach controls the velocity at the wheel pivot point instead of the vehicle force.

This control structure offers several advantages over traditional traction systems based on direct force or slip ratio allocation. These benefits are discussed in detail in Section 2.7.1.

Assumptions

The proposed system assumes the availability of necessary signals, either measured or estimated, generally at any arbitrary point. Specifically, the selected required signals are:

- **At the vehicle center:** Vehicle longitudinal velocity and acceleration, and vehicle angular rates (e.g., yaw rate).
- **At each wheel:** Wheel RPM, wheel traction torque, brake torque, and wheel steering angle.

This chapter is structured as follows: First, a high-fidelity nonlinear vehicle twin-track model implemented in Simulink is presented in Section 2.1. This model is transparent, easy to analyze, open to modifications, and beneficial for demonstrating and verifying the properties and functionalities of the proposed controller. Next, the control system is described, including its application to braking systems with both friction and recuperative brakes. Following this, an analysis and simulations demonstrating the performance of the proposed control system using the Matlab & Simulink nonlinear model are provided. Further, the IPG CarMaker high-fidelity EFORCE formula validation model is introduced to validate the control system and compare it to the state-of-the-art approach described in [3]. The CarMaker model is parameterized using real CTU student EFORCE formula measurements. Finally, the Hardware in the Loop (HiL) setup used for validating the brake torque blending mechanism is introduced, and the HiL-CarMaker-Simulink co-simulation test scenarios and results are presented.

This chapter including the figures is based on my previously published work in [13, 14].

In summary, this chapter will elaborate on a novel vehicle motion feedback allocation method that promises improved performance and efficiency over traditional traction control methods. This approach is based on transforming vehicle motion references at the CP to wheel pivot points, offering a more refined and responsive control mechanism. The chapter will detail the theoretical foundations, practical implementation, and validation of this method, providing a comprehensive overview of its advantages and potential applications in modern vehicle control systems.

2.1 High Fidelity Mathematical Twin-track Model

The nonlinear twin-track model has been adopted from [2, Chapter 11] and [29, Chapter 2] for the purpose of validating and verifying the functionality of the proposed controller. An implementation of this model is available in the Git repository [30]. The model uses multiple coordinate systems (CS), which are indicated by superscripts in the expressions. These coordinate systems, illustrated in Fig. 2.1, are described as follows:

- **Vehicle body-fixed**: Originates at the vehicle's center point (CP). All vehicle measurements and driver commands are assumed to be located at the CP without loss of generality. Superscript v is used for this CS.
- **Wheel pivot-fixed**: Originates at the wheel pivot point and is oriented the same as the vehicle body. Superscript b_i is used, where i denotes the i-th wheel. This CS is a translation of the vehicle body-fixed CS along vector $\mathbf{r}_i$.

- **Wheel-fixed**: Originates at the center of the wheel and is bound to the wheel, including its orientation. Superscript w_i is used, where i denotes the i-th wheel. This CS is rotated from the wheel pivot-fixed CS by the wheel steering angle δ_i.

The mathematical model consists of four fundamental parts:

- Vehicle nonlinear rigid body dynamics (Section 2.1.1)
- Suspension model for load transfer (Section 2.1.2)
- Drivetrain and brake model, including wheel dynamics (Section 2.1.3)
- Tire-to-road interface model (Section 2.1.4)

These parts will be briefly described here for clarity of the control systems derivation, with further details available in [2, 30].

The model inputs are

$$\mathbf{u} = [\tau^e_{\text{ref},1}, \dots, \tau^e_{\text{ref},4}, \tau^b_{\text{ref},1}, \dots, \tau^b_{\text{ref},4}, \delta_1, \delta_2], \tag{2.1}$$

where $\tau^e_{\text{ref},i}$ is the commanded e-motor torque for the i-th wheel ($i \in \{1, 2, 3, 4\}$), $\tau^b_{\text{ref},i}$ is the commanded friction brake torque for the i-th wheel, and δ_1, δ_2 are the steering angles of the front wheels.

The steering angles δ_i are constrained as follows:

$$\delta_1 = \delta_2 \stackrel{!}{=} \delta, \tag{2.2}$$

The Ackerman steering is neglected for the sake of mathematical clarity but is implemented in the high fidelity CarMaker model for final validation of the control system.

The 12 twin-track model state variables are

$$\mathbf{x} = [\mathbf{v}^v, \boldsymbol{\Omega}^v, \phi, \theta, \psi, \omega_1, \dots, \omega_4], \tag{2.3}$$

where $\mathbf{v}^v$ is the vehicle velocity vector at CG (vehicle body-fixed CS), $\boldsymbol{\Omega}^v$ is the vehicle body angular rates vector (roll rate, pitch rate, and yaw rate) at CG (vehicle body-fixed CS), ϕ is the vehicle body roll angle, θ is the vehicle body pitch angle, ψ is the vehicle body yaw angle, and ω_i ($i \in \{1, 2, 3, 4\}$) are the angular velocities of the wheels (wheel-fixed CS).

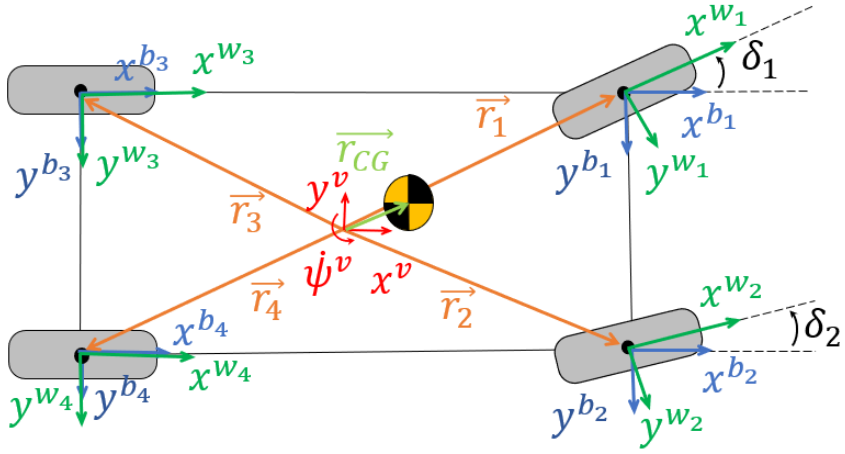


Figure 2.1: Coordinate systems considered in the model – xy plane view.

2.1.1 Vehicle Body Nonlinear Dynamics Model

The CP is set to be at the vehicle's center of gravity (CG) to simplify the model equations' derivation, where the CG is rearward of the vehicle's geometric center.

The dynamics of the twin-track rigid body is based on Newton-Euler equations:

$$m_v \left(\dot{\mathbf{v}}^v + \boldsymbol{\Omega}^v \times \mathbf{v}^v\right) = \mathbf{F}^v, \tag{2.4a}$$

$$\boldsymbol{\Theta}_v \cdot \dot{\boldsymbol{\Omega}}^v + \boldsymbol{\Omega}^v \times \left(\boldsymbol{\Theta}_v \cdot \boldsymbol{\Omega}^v\right) = \mathbf{T}^v, \tag{2.4b}$$

where $\mathbf{F}^v$ is the resulting force at CG, $\mathbf{T}^v$ is the resulting torque at CG, m_v is vehicle mass, and $\boldsymbol{\Theta}_v$ is the tensor of vehicle moment of inertia at CG.

The resulting force $\mathbf{F}^v$ and torque $\mathbf{T}^v$ are given by:

$$\mathbf{F}^v = \sum_{i=1}^{4} \mathbf{T}_v^{b_i} \cdot \mathbf{F}^{b_i} - \mathbf{F}_{res} + \mathbf{F}_g^v, \tag{2.5}$$

$$\mathbf{T}^v = \sum_{i=1}^{4} \mathbf{r}_i^v \times \mathbf{F}^{b_i} + \mathbf{r}_{res}^v \times \mathbf{F}_{res}, \tag{2.6}$$

$$\mathbf{F}_{res} = \frac{1}{2} c_{res} \rho A \sqrt{\left(v_x^v\right)^2 + \left(v_y^v\right)^2} \begin{bmatrix} v_x^v \\ v_y^v \\ 0 \end{bmatrix}, \tag{2.7}$$

where $\mathbf{F}^{b_i}$ is the force vector generated by the i-th wheel (wheel pivot-fixed CS), $\mathbf{F}_{res}$ is the combination of all resistant and aerodynamic forces acting on the vehicle at the center of pressure (CPr), $\mathbf{F}_g^v$ is gravitational force (vehicle body-fixed CS), $\mathbf{r}_i^v$ is the position of the i-th wheel, $\mathbf{r}_{res}^v$ is the position vector of the CPr (vehicle body-fixed CS), c_{res} is the aerodynamic drag constant, ρ is air density, and A is the equivalent frontal area.

2.1.2 Suspension Model

The suspension model is represented by a spring-damper system at each wheel, with dynamics described as:

$$F_{\Delta zi}^{b_i} = \left(-c_i \Delta l_i + d_i \frac{d\Delta l_i}{dt} \right), \tag{2.8}$$

$$\Delta l_i = \mathbf{r}_i^v \times \mathbf{\Omega}^v, \tag{2.9}$$

where d_i is damping rate, c_i is spring stiffness, $\mathbf{\Omega}^v$ are Euler angles (pitch, roll, yaw), Δl_i is spring compression, and $F_{\Delta zi}^{b_i}$ is the normal force from suspension. The suspension model is used to model the load transfer and the effect of the suspension on the wheel normal force. The suspension model does not have any state and connected dynamics. The needed state variables (Δl_i and $\frac{d\Delta l_i}{dt}$) are derived from the vehicle body state variables (see Section 2.1.1)

2.1.3 Drivetrain and Wheel Model

The drivetrain and wheel model dynamics are characterized as:

$$J_i \cdot \dot{\omega}_i = \tau_i + F_x^{w_i} \cdot r_{w_i} + \tau_{res,i}\left(\omega_i\right), \tag{2.10}$$

$$\tau_i = \tau_i^e + \tau_i^b, \tag{2.11}$$

where J_i is the wheel moment of inertia along the wheel shaft (y) axis, ω_i is the wheel angular speed, τ_i is the wheel drive and brake torque, $F_x^{w_i}$ is the longitudinal traction force generated by the wheel, $\tau_{res,i}\left(\omega_i\right)$ are all-wheel and e-motor losses, and r_{w_i} is the wheel effective radius.

The actuator model addresses the delays in the powertrain system, such as the CAN bus communication delay and overlooked dynamics. The first-order actuator model defines the actual e-motor torque τ_i^e utilized in wheel dynamics as:

$$\dot{\tau}_i^e = \frac{1}{T_e}\left(\tau_{\mathrm{ref},i}^e - \tau_i^e\right). \tag{2.12}$$

The system delays generally reach up to ten milliseconds [31], which is considered the worst-case scenario time constant ($T_e = 10\mathrm{ms}$) for this study and is integrated into the e-motor model.

The electro-mechanical friction brake system undergoes various torque-tracking delays due to multiple factors. Firstly, the delay stemming from the electric motor and inverter

used to generate hydraulic pressure is comparable to the delays mentioned in Equation (2.12). Secondly, friction brakes need to close the gap between the brake disc and pads and build the required pressure. Lastly, the hydraulic circuit introduces transportation delay and nonlinearities. The combined effect of these factors results in a delay in torque delivery ranging from 150 to 500 milliseconds according to [32]. The e-motor dynamics are significantly faster, with the torque dynamics time constant being less than 20 milliseconds, as per [33]. The friction brake dynamics are typically at least 2-5 times slower than the e-motors. The dynamics are expressed as:

$$\dot{\tau}_i^b = \frac{1}{T_b}\left(\tau_{\mathrm{ref},i}^b - \tau_i^b\right), \tag{2.13}$$

where the symbols carry the same meaning as in equation (2.12), with the distinction that the superscript b denotes friction brakes. The worst-case time constant $T_b = 100$ ms is employed in the mathematical model.

The reduction of the wheel longitudinal traction force $F_x^{w_i}$ is modeled by the wheel longitudinal slip curve (Fig. 3.2) and traction ellipse shape (Fig. 2.4a). The dependency is sketched as:

$$F_x^{w_i} = f\left(\lambda_i, F_z^{w_i}, \mu_i, F_y^{w_i}\right). \tag{2.14}$$

2.1.4 Tire-to-Road Interface

The forces at the tire-to-road interface are typically described using slip variables, specifically the longitudinal slip ratio λ and the slip angle α (see eq. (2.15) and (2.18)). For modeling tire force, the Pacejka magic formula is utilized for both the longitudinal and lateral directions (refer to [2]). To combine the longitudinal and lateral traction properties of the wheel, the traction ellipse is employed (see [28, 34] for more details).

The wheel's longitudinal traction force is determined by the slip ratio λ_i, as defined in eq. (2.15).

$$\lambda_i = \frac{\omega_i \cdot r_{w_i} - v_x^{w_i}}{\max\left(|\omega_i| \cdot r_{w_i}, |v_x^{w_i}|\right)}, \tag{2.15}$$

In this equation, $v_x^{w_i}$ is the longitudinal speed of the i-th wheel hub (in the wheel-fixed coordinate system), ω_i is the angular speed of the i-th wheel, and r_{w_i} is the effective radius of the i-th wheel. The relationship between the longitudinal force generated by a specific wheel $F_x^{w_i}$ and the wheel slip ratio and tire-interface variable is represented by the slip curve and the simplified Pacejka magic formula (see [28]), given by:

$$\overline{F_x^{w_i}} = \mu_i F_z^{w_i} D \sin\left(C \arctan\left(B\lambda - E\left(B\lambda - \arctan\left(B\lambda\right)\right)\right)\right), \tag{2.16}$$

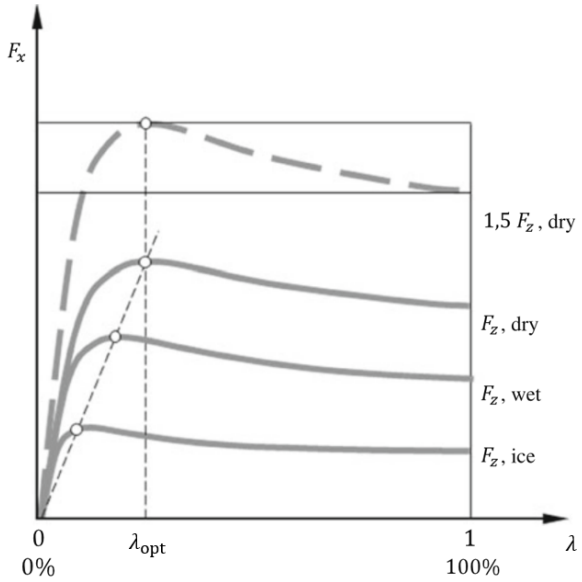


Figure 2.2: Longitudinal slip curve. Adapted from [2]. The same shape applies to negative slip ratios generating negative/braking traction force.

Here, μ_i is the road friction coefficient, $F_z^{w_i}$ is the normal force on the wheel, and B, C, D, E are Pacejka's shaping coefficients. $\overline{F_x^{w_i}}$ is the traction force generated by the wheel in the x direction (see Fig. 3.2).

The slip curve can be divided into two regions based on the slope: stable with a positive slope and unstable with a negative slope. Linearizing the wheel dynamics around a stable Operating Point (OP) results in a stable system, and vice versa. Examples of various OP linearizations are shown in Fig. 2.3a. The root locus of the minimum realization of the entire vehicle model linearization at these points is depicted in Fig. 2.3b.

It is important to recognize the dependency of the resulting traction force on the normal load $F_z^{w_i}$, Pacejka coefficient D, and the road friction coefficient μ_i. A new artificial tire interface variable was introduced in [13] to unify these dependencies into a single, convenient-to-use variable. This simplification neglects the shift in the slip curve maximum due to changes in the tire-to-road interface, combined loading changes, or lateral slip effects (see Fig. 3.2).

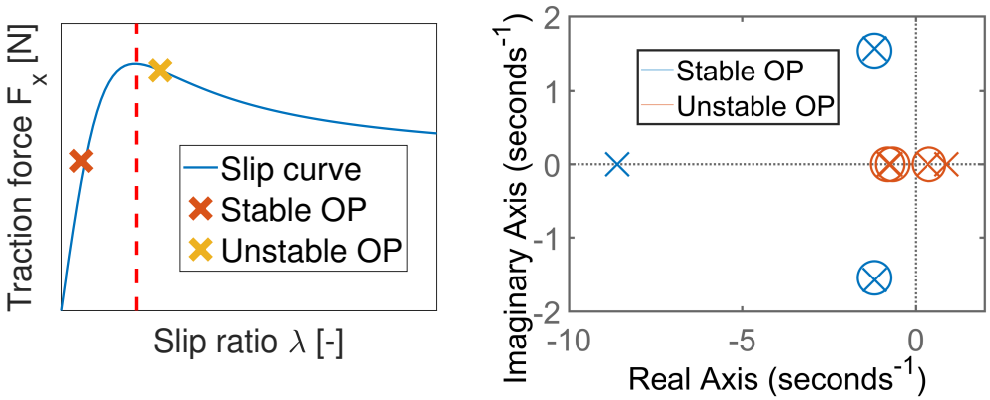
The tire-interface variable ϵ_i for an i-th wheel is defined in [13] as:

$$F_x^{w_i} = \epsilon_i \cdot \overline{F_x^{w_i}}, \quad \epsilon_i \in \mathbb{R}^+. \tag{2.17}$$

Remark. *If $\epsilon_i = 1$ then equation (2.16) becomes the standard Pacejka magic formula [28].*

The wheel's lateral traction force is determined by the slip angle α_i, defined as:

$$\alpha_i = -\arctan\left(\frac{v_y^{w_i}}{|v_x^{w_i}|}\right), \tag{2.18}$$



(a) Stable and unstable Operating Point (OP).

(b) Root locus of minimum realization stable and unstable OP linearized model.

Figure 2.3: The stable and unstable slip curve OPs.

where $v_x^{w_i}$ is the longitudinal speed of the i-th wheel hub and $v_y^{w_i}$ is the lateral speed of the i-th wheel hub (in the wheel-fixed coordinate system).

The relationship between the lateral force generated by a specific wheel $F_y^{w_i}$ and the wheel slip angle is represented by the slip curve and the Pacejka magic formula with the same formulation as in eq. (2.16), but with the slip variable α_i instead of λ_i, and a different set of parameters representing lateral tire-to-road interface properties [35].

Finally, the traction ellipse is introduced to combine the longitudinal and lateral wheel traction capacities. The traction ellipse represents the bounded friction force generated by the tire-to-road contact patch during combined longitudinal and lateral motion. Combined slip occurs when the vehicle accelerates or brakes while cornering. A tire cannot generate a combined traction force (comprising lateral and longitudinal components) greater than $F_z^{w_i} \cdot \mu_i$, where $F_z^{w_i}$ is the normal force on the wheel. This restriction is expressed by the friction ellipse (also known as Kamm's circle):

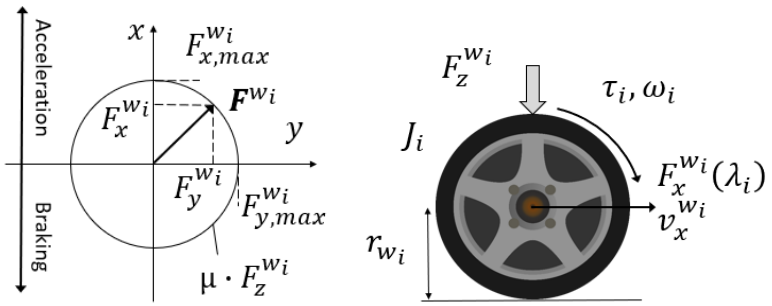
$$F_{combined}^{w_i} = \sqrt{\frac{\left(F_x^{w_i}\right)^2}{\left(D_x\right)^2} + \frac{\left(F_y^{w_i}\right)^2}{\left(D_y\right)^2}} \leq \mu_i F_z^{w_i}, \tag{2.19}$$

where D_x and D_y are the longitudinal and lateral Pacejka magic formula D parameters.

Let us denote the forces calculated using the Pacejka Magic formula as $F_{x,max}$ and $F_{y,max}$. Then, the following algorithm (Eq. (2.20) - (2.24)) is applied to scale (if needed) the resulting force:

$$\beta = \arccos\left(\frac{|\lambda|}{\sqrt{\lambda^2 + \sin^2(\alpha)}}\right), \tag{2.20}$$

$$\mu_{x,act} = \frac{F_{x,max}}{F_z}, \qquad \mu_{y,act} = \frac{F_{y,max}}{F_z}, \tag{2.21}$$



(a) Wheel traction ellipse. (b) Wheel model

Figure 2.4: Wheel model and traction ellipse.

$$\mu_{x,max} = D_x, \qquad \mu_{y,max} = D_y, \tag{2.22}$$

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

$$\mu_y = \frac{\tan(\beta)}{\sqrt{\left(\frac{1}{\mu_{x,max}}\right)^2 + \left(\frac{\tan(\beta)}{\mu_{y,act}}\right)^2}}, \qquad F_y = \left|\frac{\mu_y}{\mu_{y,act}}\right| F_{y,max}. \tag{2.24}$$

The model implementation in MATLAB & Simulink involves detailed steps to ensure the accuracy and reliability of simulations. Additionally, the vehicle model's parameters are meticulously tuned to reflect realistic conditions, ensuring the model's applicability in various scenarios. These steps underscore the robustness of the twin-track model in capturing the dynamics of real-world vehicle behavior, making it an invaluable tool for controller design and testing.

In conclusion, the twin-track model, with its comprehensive approach to vehicle dynamics, provides a detailed framework for understanding and simulating vehicle behavior under various conditions. Its integration of multiple coordinate systems, suspension dynamics, drivetrain, and tire-road interface interactions offers a holistic view of vehicle behavior. This model is not only essential for the vehicle dynamics control systems development but also serves as a foundation for controller validation and verification.

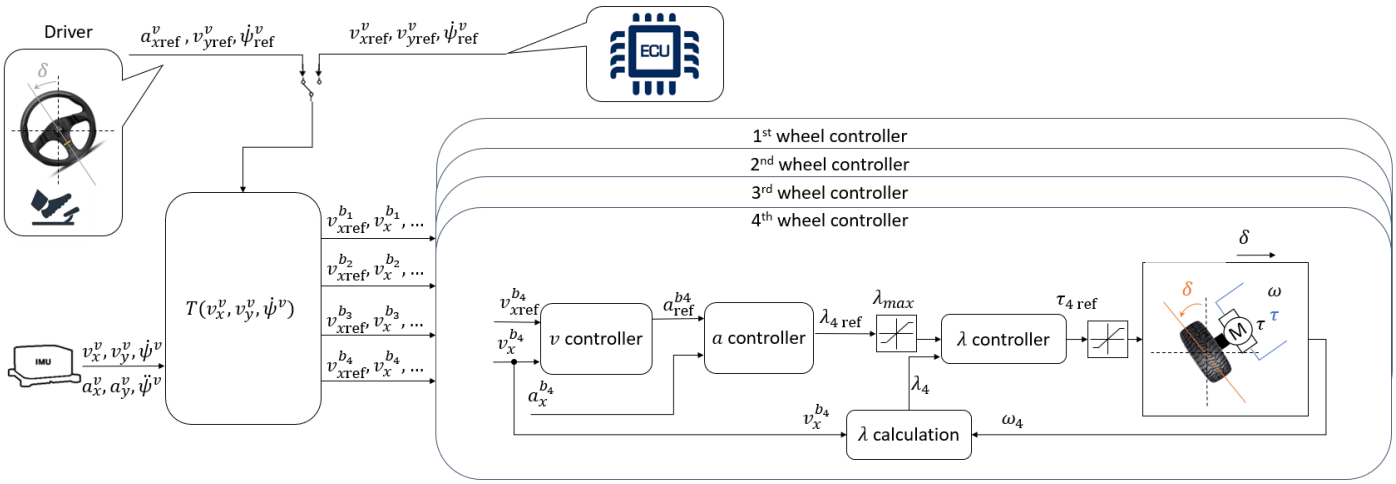


Figure 2.5: Proposed traction allocation control system architecture.

2.2 Vehicle Motion Feedback Control Allocation Architecture

The primary contribution of this section (as published in [13]) is the proposed control architecture, which offers a system-level design approach to controlling vehicle longitudinal dynamics. The main concept involves using transformation (2.25) to map vehicle states (speed or acceleration), including both references and measurements, to individual wheels. Subsequently, vehicle-level state variables are managed at the wheel level using any preferred control strategy. In contrast, the direct optimization-based methods utilized in [3, 15, 16] determine the individual wheel slip ratio or traction force by solving the control allocation task ambiguities via optimization.

The transformation of vehicle-level signals to the wheel level is defined as:

$$\mathbf{v}^{b_i}{}_{\text{ref}} = T\left(\mathbf{v}^{v}{}_{\text{ref}}, \boldsymbol{\Omega}^{v}{}_{\text{ref}}\right) = \mathbf{v}^{v}{}_{\text{ref}} + \boldsymbol{\Omega}^{v}{}_{\text{ref}} \times \mathbf{r}^{v}{}_{i} + \boldsymbol{\gamma}\left(\boldsymbol{\Omega}^{v}{}_{\text{ref}}\right), \tag{2.25}$$

where $\mathbf{v}^{v}{}_{\text{ref}}$ is the commanded velocity vector at the vehicle center point (CP), $\boldsymbol{\Omega}^{v}_{\text{ref}}$ represents the desired roll, pitch, and yaw rates at the CP, $\mathbf{r}^{v}{}_{i}$ denotes the position of the i-th wheel, and $\mathbf{v}^{b_i}{}_{\text{ref}}$ is the velocity reference signal for the i-th wheel. Since this work primarily focuses on wheel longitudinal dynamics, only the $v^{b_i}_{x\,\text{ref}}\left(v^{v}_{x}, \dot{\psi}^{v}\right)$, as a function of vehicle velocity v^{v}_{x} and yaw rate $\dot{\psi}^{v}$, is extracted from the transformation results for further control. However, extending this to lateral speed $v^{b_i}_{y\,\text{ref}}$ tracking is possible following the same principles.

The $\boldsymbol{\gamma}$ function augments the pure physical transformation (2.25) by introducing a lateral vs. longitudinal dynamics preference feature. The $\boldsymbol{\gamma}$ function is derived from the vector cross-product definition and is defined as:

$$\boldsymbol{\gamma}\left(\boldsymbol{\Omega}^{v}\right)=\left(\Gamma\cdot\left(-\dot{\psi}^{v}\cdot r_{y\ i}^{v}\right),\ 0,\ 0\right), \tag{2.26}$$

where $\Gamma \in \mathbb{R}_0^+$ is the tuning parameter. When $\Gamma = 0$, the transformation (2.25) retains its original physics-based meaning. The value of Γ amplifies the effect of yaw rate on the resulting $v_x^{b_i}$ in (2.25), thereby preferring yaw rate $\dot{\psi}^v$ over longitudinal velocity v_x^v tracking. The designer can tune Γ to match specific vehicle lateral to longitudinal dynamics control preferences. Physically, Γ can be interpreted as a parameter artificially increasing the vehicle width.

The control laws at the wheel level – specifically, velocity, acceleration, and slip ratio tracking – are presented here for completeness. However, it is important to note that these can be replaced by any other suitable control mechanism to provide the desired functionality.

2.2.1 Drive-by-human and self-driven controller hierarchy

The hierarchical structure of the wheel level control layer is depicted in Fig. 2.5. The transformation (2.25) is general and applicable for both self-driving (automated) and human-driven vehicles. Trajectory planning algorithms typically generate vehicle velocity and yaw rate profiles to be tracked, which aligns perfectly with the presented transformation (2.25). In such cases, the transformation (2.25) is directly employed.

However, the proposed control system can also be used in human-driven vehicles, where vehicle acceleration and yaw rate are more suitable variables for control. In this scenario, the transformation (2.25) must be modified as:

$$\mathbf{v}^{b_i}{}_{\text{ref}}=T\left(0,\boldsymbol{\Omega}^{v}{}_{\text{ref}}\right)=0+\boldsymbol{\Omega}^{v}{}_{\text{ref}}\times\mathbf{r}^{v}{}_{i}+\boldsymbol{\gamma}\left(\boldsymbol{\Omega}^{v}{}_{\text{ref}}\right). \tag{2.27}$$

The reference for the wheel acceleration controller $a_{x\ \text{ref}}^{b_i}$ is then determined as:

$$a_{x\ \text{ref}}^{b_i}=a_{x\ \text{ref}}^{v}+a_{\text{lat}}^{b_i}, \tag{2.28}$$

where $a_{\text{lat}}^{b_i}$ is the acceleration commanded by the wheel pivot point velocity controller, representing the corrective action for $\boldsymbol{\Omega}^{v}{}_{\text{ref}}$ tracking, and $a_{x\ \text{ref}}^{v}$ is the driver-generated CP acceleration command.

2.2.2 Wheel controller hierarchy

The control layer at the wheel pivot point level can be replaced by any appropriate control system and is presented here for completeness. The controller can be designed either centrally or in a distributed manner.

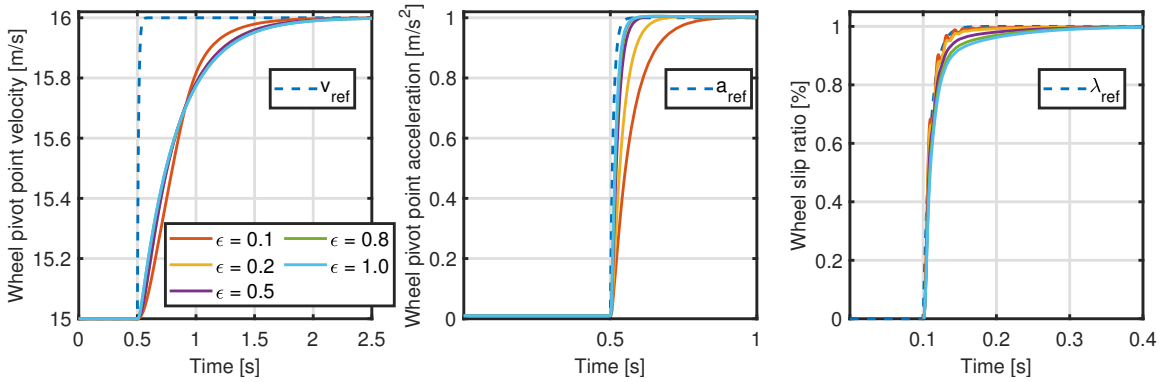


Figure 2.6: Wheel level individual control loops step responses. The step responses are from left to right: Wheel pivot point velocity tracking, wheel pivot point acceleration tracking, and slip ratio tracking.

The wheel level part of the control system consists of three hierarchically connected controllers: the wheel longitudinal slip ratio λ_i, wheel pivot point acceleration $a_x^{b_i}$, and wheel pivot point velocity $v_x^{b_i}$ controllers. The wheel pivot point velocity $v_x^{b_i}$ is commanded from the transformation (2.25) and uses the wheel pivot point acceleration $a_{x\ \text{ref}}^{b_i}$ command as the manipulated variable. Then, the wheel pivot point acceleration $a_x^{b_i}$ is controlled via the $\lambda_{i\ \text{ref}}$ command. Finally, the wheel slip ratio λ_i is controlled by manipulating the wheel torque $\tau_{i\ \text{ref}}$. All controllers were designed using continuous-time design techniques and then discretized with a frequency $100Hz$. The frequency was chosen to ensure that the proposed control system is easily deployable on the embedded hardware used in vehicles.

Measuring wheel-level acceleration and velocities is challenging and costly. Therefore, only CP measurements (vehicle body state measurements) are used. The wheel-level velocities and accelerations are computed via the transformation (2.25) (as illustrated in Fig. 2.5). To summarize, the building blocks, from the inner to the outer loop, are:

- λ_i **tracking** - inner/core layer providing wheel slip ratio tracking functionality via torque $\tau_{i\ \text{ref}}$ manipulation.
- $a_x^{b_i}$ **tracking** - middle layer providing wheel pivot point acceleration $a_x^{b_i}$ tracking functionality via $\lambda_{i\ \text{ref}}$ manipulation.
- $v_x^{b_i}$ **tracking** - outermost layer controlling wheel pivot point velocity via $a_{x\ \text{ref}}^{b_i}$ manipulation.

Step responses of the individual loops are shown in Fig. 2.6.

λ tracking

The wheel slip ratio (assuming a stable OP) can be physically understood as a proportional part of the available traction force. This is more evident when using the bilinear slip curve

approximation and tire stiffness c_λ as:

$$F_x = F_z c_\lambda \lambda. \tag{2.29}$$

The control of the slip ratio and wheel angular speed has been addressed in numerous studies. For instance, [36] proposes a piece-wise linear feedback controller that manages the slip ratio λ and traction force for each wheel. A similar problem is tackled in [37]. The slip ratio is a prime candidate for the controlled variable in any traction system. The concept of λ-control is not new and is common in railroad vehicles (see [38]) and current automotive R&D projects such as [39]. Moreover, the proposed controller retains all the necessary safety and economy functionality of the wheel-level controllers.

The $\tau_{\text{ref},i}$ value is generated based on the demanded $\lambda_{i\ \text{ref}}$ value, wheel speed ω_i, and wheel hub longitudinal speed $v_x^{w_i}$. Initially, the measured values at the vehicle CP are transformed using eq. (2.25) to the wheel pivot point. Then, the measured slip ratio is computed using eq. (2.15). The slip ratio controller was designed using linear techniques, specifically root locus, and fine-tuned on the nonlinear model presented in Section 2.1. A full PID controller was designed on a linearized vehicle model with a nonzero slip ratio stable OP ($\lambda_{OP} = 0.05$).

The linear stability analysis of the stable and unstable slip curve OPs (see Fig. 2.3a) is shown in Fig. 2.7. The two OPs minimum realizations of controlled vehicle dynamics linearization are depicted. The controller was designed for the stable OP (positive slip curve slope). The unstable OP (negative slip curve slope) is stabilized with the very same discrete-time ($T_s = 0.01s$) PID controller designed for the stable OP as shown in Fig. 2.7.

It is important to note that slip ratio λ control is possible only when the wheel pivot point has a nonzero velocity $v_x^{b_i} > \xi; \xi > 0$ to enable slip ratio calculation (see eq. (2.15)). Therefore, the control policy presented is applicable only for vehicle velocities exceeding a certain threshold, e.g., $|\mathbf{v}^{\mathbf{v}}| > 3m/s$.

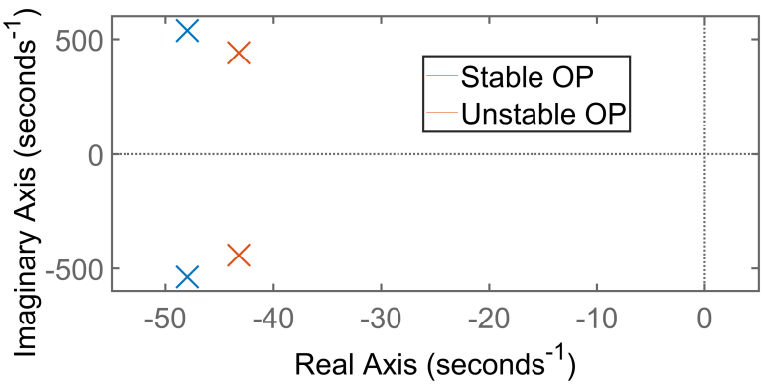


Figure 2.7: Compensated stable and unstable OP linearized system root locus.

$a_x^{b_i}$ tracking - "λ-reference generation"

This layer is responsible for tracking the $a_{x,\text{ref}}^{b_i}$ setpoint (acceleration tracking at the i-th wheel pivot point). Input-output linearization of the nonlinear model, including the λ controller, was used for the continuous-time PI controller design. The PI controller was discretized with $100Hz$ frequency. The $\lambda_{\max}$ limit is an algorithm parameter and can be selected based on slip curve estimation as the slip ratio value for maximal wheel traction λ_{opt} (see [40, 17] and Chapter 3 for details). However, in this section, it is implemented as a constant.

Traction limits

When a vehicle approaches its traction limits, ensuring precise yaw rate tracking becomes more critical than focusing on acceleration or deceleration. This principle is commonly embedded in Electronic Stability Program (ESP) controllers. If the commanded slip ratio $\lambda_{i,\ \text{ref}}$ nears its maximum threshold $\lambda_{\max}$, the reference for the wheel acceleration controller $a_{x\ \text{ref}}^{b_i}$ is recalculated as follows:

$$a_{x\ \text{ref}}^{b_i} = (a_{x\ \text{ref}}^{v} - a_{\text{lat,max}}) + a_{\text{lat}}^{b_i}, \tag{2.30}$$

$$a_{\text{lat,max}} = \max\left(a_{\text{lat}}^{b_i}\right),\ i \in (1,2,3,4)\,. \tag{2.31}$$

This method ensures that the acceleration needed for accurate yaw rate tracking is prioritized over longitudinal acceleration in conditions where traction is limited, such as on icy roads. This strategy is applied similarly in both self-driven and human-driven vehicle controllers.

$v_x^{b_i}$ tracking - "$a_x^{b_i}$-reference generation"

The velocity at the wheel pivot point $v_x^{b_i}$, which represents the vehicle's state, is derived from the vehicle's longitudinal speed v_x^v, lateral speed v_y^v, and yaw rate $\dot{\psi}^v$ measurements at the vehicle's CP, transforming these into the wheel pivot point coordinate system (see eq. (2.25)). The reference speed for each wheel, derived from this transformation, is tracked using a proportional (P) controller that generates the required wheel pivot point acceleration $a_{x,\text{ref}}^{b_i}$. The P controller design involves input-output linearization of the nonlinear model, incorporating the acceleration controller.

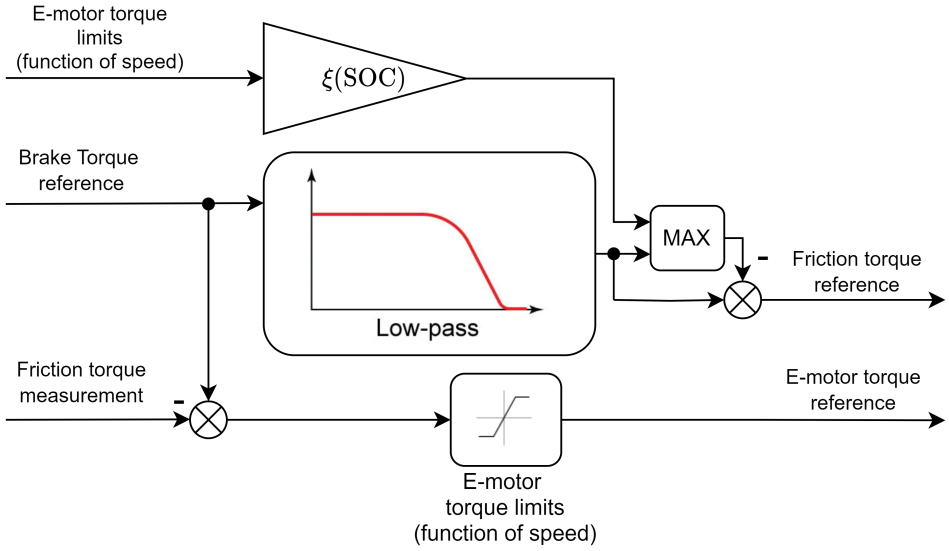


Figure 2.8: Proposed brake blending scheme.

2.3 Brake Torque Blending

Brake torque blending is a critical issue in contemporary vehicle braking systems, particularly in hybrid and electric vehicles. While regenerative brakes offer advantages such as energy recovery, simpler control, and quicker torque build-up compared to traditional friction-based brakes, they also have certain limitations. The primary constraints include lower brake torque magnitudes compared to friction brakes, the motor speed-torque characteristics, and the brake torque limitation by the state of charge (SOC) of the vehicle battery. To ensure effective brake performance, a combination of regenerative and friction-based brake systems is necessary, leading to the development of a brake torque blending algorithm as detailed in [14].

The primary goal of the proposed blending algorithm is to allocate the brake torque references from the superior control system (see Section 2.2) between the regenerative and friction-based brake systems. The algorithm aims to maximize the utilization of regenerative braking without compromising overall braking performance. The schematic representation of the proposed brake torque combination is shown in Fig. 2.8. The solution involves using friction brakes to generate torques for steady-state and slowly varying reference signals. Conversely, the torques for dynamically and rapidly changing reference signals are produced by the regenerative brakes (using the electric motor). To maintain optimal brake performance and system response time, only a portion of the currently available regenerative brake torque is used for low-frequency torque generation to prevent actuator saturation. The proportion of electric motor torque allocated for low-frequency tasks is a design parameter ξ defined following equation (2.32). The remaining portion is then available to handle rapid changes as needed. The amount of regenerative

torque allocated for rapid changes can vary based on the battery SOC, i.e., $\xi\,(\text{SOC})$. When the SOC is low, the allocation is limited only by the e-motor saturation limits and the performance requirements mentioned earlier. For instance, it is advisable to use 80% of the available regenerative torque for low-frequency torque ($\xi = 0.8$). However, when the SOC is high, such as above 80%, a smaller portion of the available regenerative torque should be used for low-frequency changes to ensure some battery storage capacity is available for storing regenerated energy from braking, e.g., $\xi = 0.2$.

Thus, the proposed blending mechanism can automatically adjust the brake blending strategy to achieve optimal performance at various levels of battery SOC. Along with SOC-dependent electric motor torque limits, this approach maximizes both brake system performance and energy efficiency. The friction torque reference is calculated using the equation:

$$\tau^{b}_{\text{ref},i} = LP\left(\tau_{\text{ref},i}\right) - \max\left(LP\left(\tau_{\text{ref},i}\right)\ ,\ \ \xi \cdot \tau_{m,lim}\right), \tag{2.32}$$

where $\tau^{b}_{\text{ref},i}$ is the friction brake torque reference[1], $\tau_{\text{ref},i}$ is the overall brake torque reference[1], τ^{e}_{lim} is the e-motor torque limit[1], and LP denotes the low-pass filter.

The electric motor brake reference is computed using the equation:

$$\tau^{e}_{\text{ref},i} = \max\left(\tau_{\text{ref},i} - \tau^{b}_{i}\ ,\ \ \tau_{m,lim}\right), \tag{2.33}$$

where $\tau_{\text{ref},i}$ is the brake torque reference, and τ^{b}_{i} is the measured brake torque. This ensures that the commanded brake torque reference is always tracked unless the actuators reach their saturation limits.

2.4 Simulation Results of the Motion Feedback Controller

The simulation-based experiments were conducted to demonstrate the system's performance. First, Matlab & Simulink simulations utilizing the nonlinear model described in Section 2.1 are presented to provide a detailed understanding of the controller features. Subsequently, simulations and analyses using a high-fidelity validation student formula model from the IPG CarMaker environment co-simulation with Simulink are shown.

[1]Note: The brake torques are negative during braking.

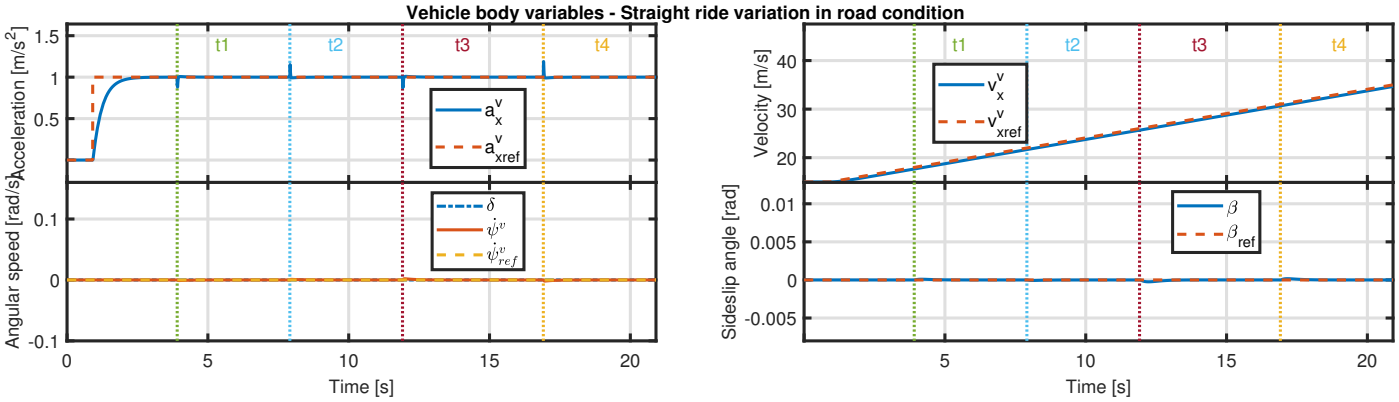


Figure 2.9: Vehicle body-fixed variables (v_x^v, $\dot{\psi}^v$, β and a_x^v) are shown here for the straight ride with ϵ_i variation (described in section Section 2.4.1). Simulation experiment is evaluated using the Matlab & Simulink model (see Section 2.1).

2.4.1 Description of MATLAB & Simulink Based Experiment

This section details the experiment performed using the model introduced in Section 2.1. The effect of different surface types on the slip curve shape is approximated by scaling the slip curve for the Matlab & Simulink experiments. The slip curve shape is preserved but scaled down by an appropriate factor. This simplification does not introduce significant error, as the slip curve is relatively flat around λ_{opt}, which is often assumed constant in related works [3, 41, 36]. Moreover, the control strategy presented is robust concerning the position of λ_{opt}. The estimation of the optimal slip ratio is discussed in [17] and described in Chapter 3. Once the λ_{opt} estimate is obtained, its integration into the proposed controller is straightforward, although the integration evaluation is not discussed in this work and will be conducted in future research. The self-driven vehicle controller hierarchy (see Section 2.2.1) was utilized in this experiment.

The results are presented in graphs comprising:

- **Vehicle body-fixed variables** - the response of vehicle-fixed body variables such as v_x^v, $\dot{\psi}^v$, β, and a_x^v (acceleration of the vehicle in the x direction in the vehicle body-fixed coordinate system).
- **Wheel variables** - variables for each wheel such as τ_i, λ_i, ϵ_i, and $a_x^{w_i}$ (acceleration of the wheel in the x direction in the wheel coordinate system).

The following experiment was carried out.

Straight Ride - Road Condition Variation

An artificial variation of the tire-to-road interface ϵ_i with constant vehicle acceleration while driving straight is simulated in this experiment.

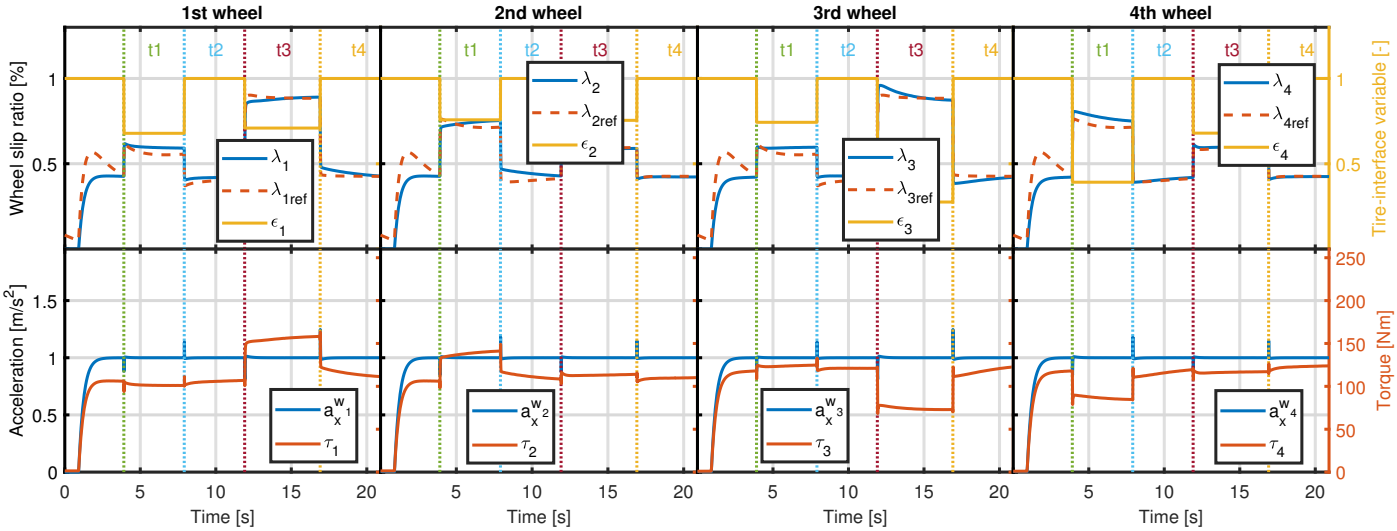


Figure 2.10: Wheels variables ($a_x^{w_i}$, τ_i, λ_i and ϵ_i for each wheel) are shown here for the straight ride with ϵ_i variation (described in Section 2.4.1). Simulation experiment is evaluated using the Matlab & Simulink model (see Section 2.1).

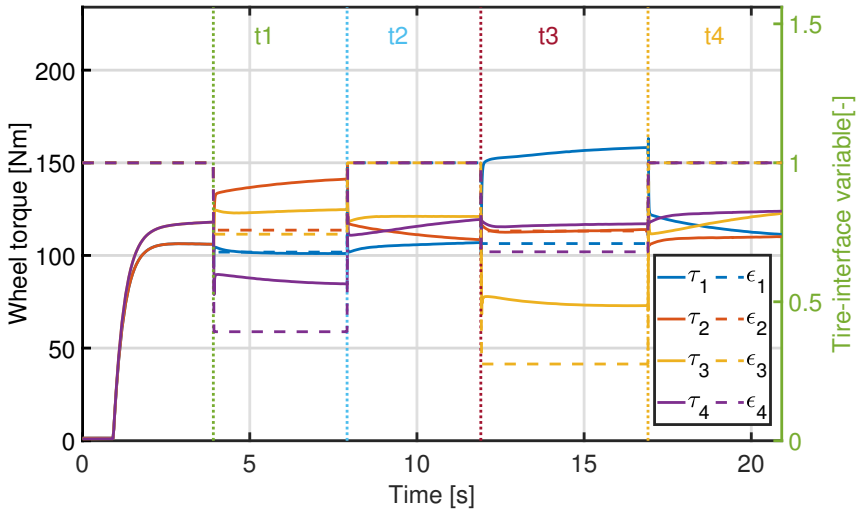


Figure 2.11: Detail of wheel torques τ_i and tire-interface variables ϵ_i for each wheel are shown here for the Simulink model (see Section 2.1) simulation experiment (see Section 2.4.1). The torque allocation based on μ_i, $F_z w_i$, and ϵ_i can be seen.

The variations in ϵ_i are different for each wheel throughout the experiment. The conditions during this experiment are:

- A steering angle $\delta = 0\,rad$ is maintained throughout the experiment.
- ϵ_i remains piecewise constant and varies during the simulation as follows:
  - $\epsilon_i = 1$ for times $t < t_1$, $t_2 < t < t_3$, and $t > t_4$
  - ϵ_i is constant but different for each wheel otherwise ($\epsilon_i \neq 1$ and $\epsilon_i \neq \epsilon_j, i \neq j$)
- The velocity reference is:
  - $v^v_{x,\mathrm{ref}}\,(t < 1) = 15\,m/s$
  - $\frac{d}{dt} v^v_{x,\mathrm{ref}}\,(t > 1) = 1\,m/s^2$

The vehicle body-fixed variables are shown in Fig. 2.9. The wheel variables are presented in Fig. 2.10. The wheel torques τ_i and tire-to-road interface variables ϵ_i for each wheel are detailed in Fig. 2.11. The results demonstrate that the vehicle tracks the velocity reference well, despite the challenging conditions. The different ϵ_i for each wheel simulate different surfaces encountered by each wheel. Although this scenario is somewhat artificial and unrealistic, it validates the proposed controller's functionality. The torque distribution for each wheel is clearly observable in Fig. 2.11. The experiment's summary and results are further discussed in detail in Section 2.7.1, along with the CarMaker experiment outcomes.

2.4.2 EFORCE Formula CarMaker High Fidelity Validation Experiment

The effectiveness and functionality of the proposed hierarchical control system were validated using the IPG CarMaker environment with a high-fidelity model from FEE CTU in Prague, representing the EFORCE student formula. This model incorporates various dynamic phenomena such as tire models, Ackermann steering, and load transfer effects, and can be accessed at [42]. The formula model used is depicted in Fig. 2.12. For validation, the CarMaker formula model was calibrated with real parameters. The drive-by-human controller hierarchy was utilized in the CarMaker experiments (see Section 2.2.1).

The measured values of the Pacejka magic formula, which are employed in the CarMaker model, are shown in Fig. 2.16.

Two experiments were carried out to evaluate the controller's performance. The first experiment involved acceleration combined with an ISO 3888-1 double lane change maneuver, while the second focused on torque vectoring. Detailed descriptions of both experiments are provided below.

Figure 2.12: EFORCE student formula used for the CarMaker model.

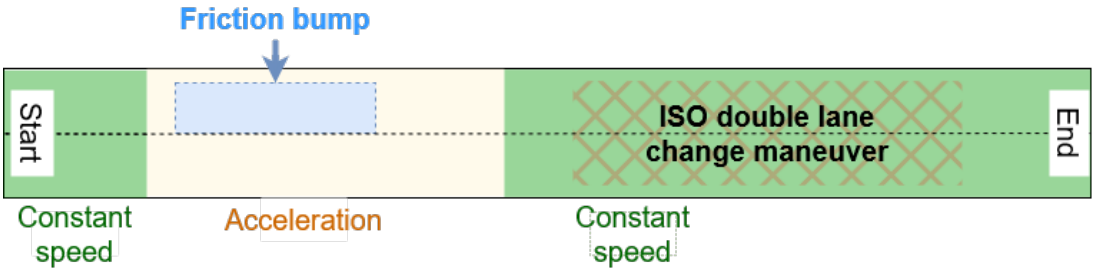


Figure 2.13: Acceleration and double lane change maneuver including the friction bump used in Section 2.4.2.

The IPG driver model generates control system references using the accelerator and brake pedals. These pedal positions were converted into a vehicle acceleration reference $a_{x\ \mathrm{ref}}^{v}$. The driver also controlled the steering wheel angle δ, which was translated into a vehicle yaw rate reference $\dot{\psi}_{\mathrm{ref}}^{v}$. The experiment consisted of two parts: acceleration and an ISO 3888-1 double lane change maneuver illustrated in Fig. 2.13. The experiment began with the vehicle maintaining a speed of 20 km/h. At approximately $t \approx 2s$, the vehicle began to accelerate. During acceleration, the vehicle encountered a friction bump with the left side wheels (approximately $t \in (4, 6)$). Finally, upon reaching 95 km/h, an ISO double lane change maneuver was executed (approximately $t \in (15, 20)$).

The comparison between $\Gamma = 0$ and $\Gamma = 10$ is shown in Fig. 2.14. It is evident that the $\Gamma = 0$ setting was unable to complete the double lane change at high speed (vehicle velocity $v_x^v = 95km/h$) and experienced greater yaw rate tracking disturbance due to the friction bump compared to the $\Gamma = 10$ setting. A comparison of the proposed control

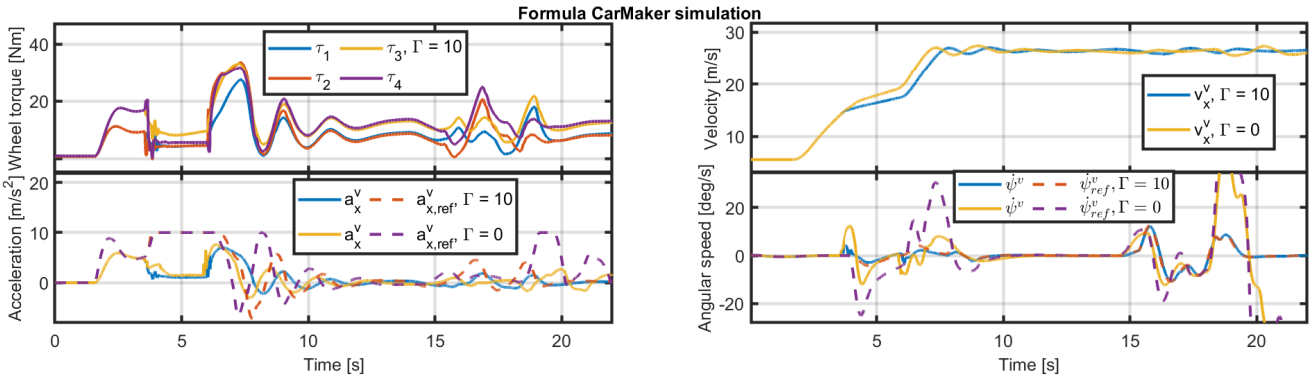


Figure 2.14: Comparison of two lateral preference Γ values for the CarMaker experiment described in Section 2.4.2. The figures are from left to right in the first row: wheel torques and vehicle velocity. Next, in the second row: vehicle acceleration with its reference and vehicle yaw rate with its reference.

system with $\Gamma = 10$ tuning and the implementation of [3] is shown in Fig. 2.15 for the wheel-level variables. It is clear that [3] could not handle the friction bump, primarily due to a lack of traction limit handling compared to the proposed solution (see Section 2.2.2). Additionally, the solution in [3] requires numerous additional signals to be estimated or measured (e.g., wheel traction force, wheel normal force, etc.).

An analysis of the experiment described above under various condition variations is presented:

- Comparison of lateral preference Γ variables
- Inaccurate λ_{opt} estimation
- Different friction bump values

Various Friction Bump Friction Coefficient Values

This section discusses varying the friction coefficient values $\mu_i \in \{0.15, 0.25, 0.4, 0.8\}$ of the friction bump encountered during acceleration. The experiment includes both the acceleration and the ISO 3888-1 double lane change maneuver, as depicted in Fig. 2.13.

First, the optimization-based solution [3] is compared for all friction coefficient variations in Fig. 2.17. With the optimization-based solution, the driver could not handle the lowest friction surface $\mu_1 = 0.15$. Several factors could explain this behavior. The IPG driver attempts to counteract the yaw rate disturbance caused by different surfaces on the left side wheels, potentially leading to driver-induced oscillations. Additionally, the left-side wheels may fail to provide the required force due to force saturation, which are not addressed in the optimization-based solution. Lastly, the optimization-based solution tracks the yaw rate reference $\dot{\psi}^v_{\text{ref}}$ in an open-loop manner without feedback. The yaw

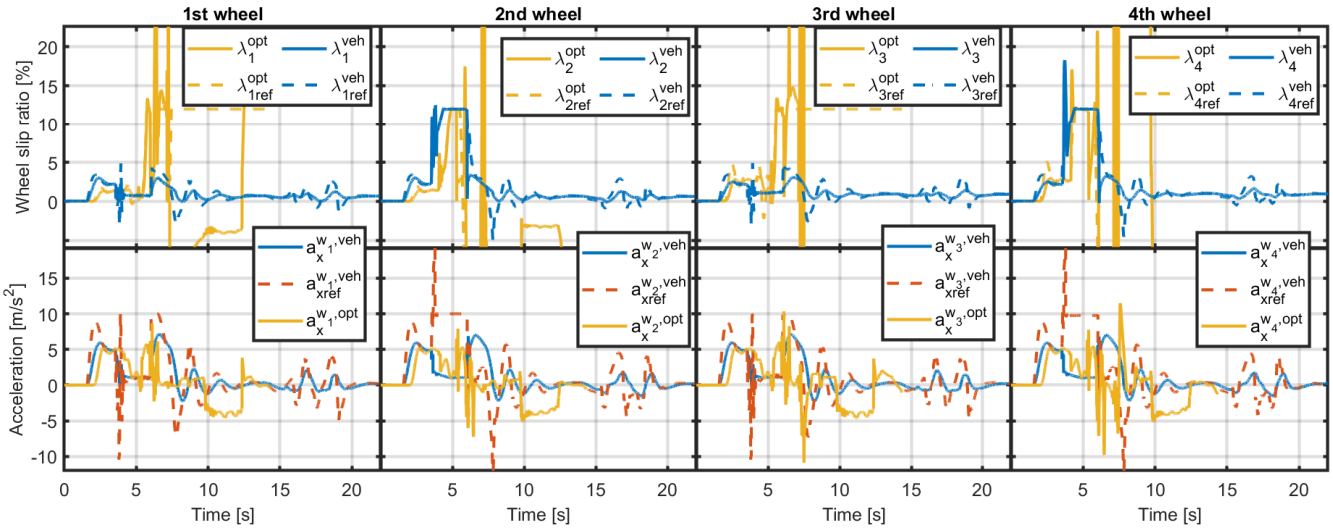


Figure 2.15: Comparison of wheel acceleration $a_x^{w_i}$ and slip ratio λ_i tracking between the proposed allocation scheme (veh superscript) and the optimization-based allocation scheme from [3] (opt superscript). Details are described in Section 2.4.2.

rate reference is converted into the required vehicle body torque M_z, which is then used to determine individual wheel force references as described in [3].

Next, the same comparison for all surface types employing the proposed control system is shown in Fig. 2.18. The proposed control system enables the driver to handle all friction coefficient values, unlike the optimization-based solution [3] shown in Fig. 2.17. The proposed control system provides a vehicle-level motion feedback control law, ensuring accurate vehicle reference tracking even under extreme conditions.

The comparison for the lowest friction coefficient μ_1 of both the optimization-based and the proposed solution at the wheel level is shown in Fig. 2.15. Detailed moments when the vehicle enters the friction bump are presented in Figures 2.19 and 2.20. The details of the control allocation are shown in Fig. 2.21. It can be seen that the optimization-based system from [3] performs well on nominal surfaces. However, the disturbance in yaw rate tracking is significant when the vehicle encounters the low friction surface with the left wheels. This issue is not present in the proposed system, which respects traction limits. The vehicle cannot track the acceleration reference due to physical constraints on the friction bump. In this case, the second and fourth wheels apply a maximum slip ratio, which is still insufficient to meet the requested vehicle acceleration (see Fig. 2.19). Therefore, the proposed system maintains the highest possible acceleration while tracking the driver's yaw rate reference given through the steering wheel. This behavior can be clearly seen in Fig. 2.21. The disturbance rejection of the optimization-based solution of [3] is having much worse performance than the proposed system. Experiment video can be found at [43].

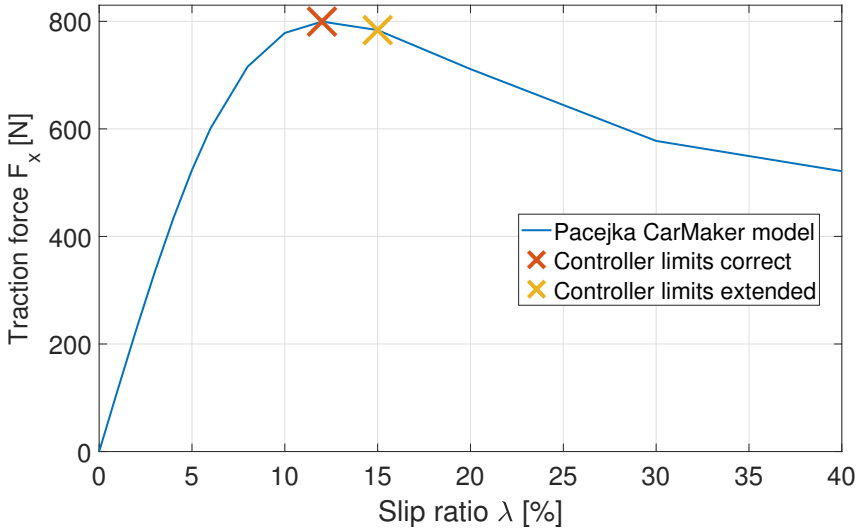


Figure 2.16: Pacejka magic formula measured values used in the CarMaker model. The limit value used as $\lambda_{\max}$ in the control system is selected correctly once and extended beyond the true λ_{opt} to simulate an incorrect λ_{opt} estimation.

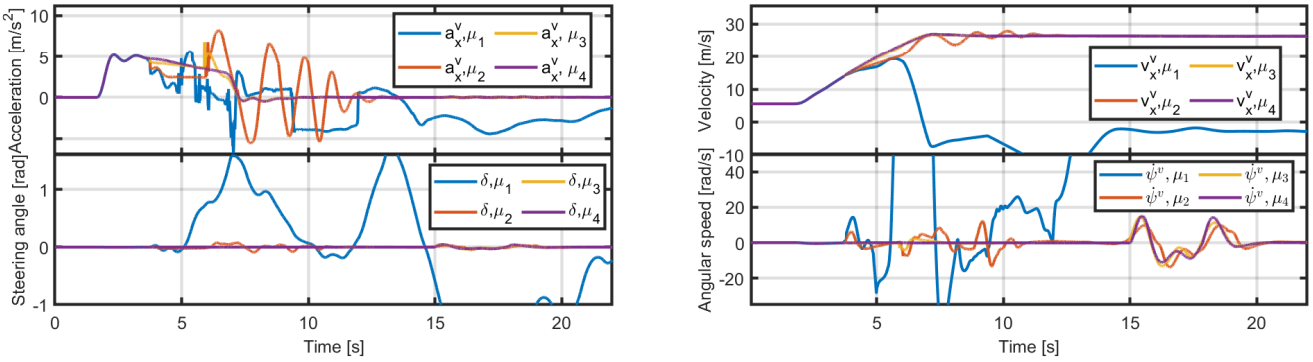


Figure 2.17: Comparison of the optimization-based solution [3] for various friction coefficients μ_i of the friction bump, driven over with the left side wheels during acceleration. Details are explained in Section 2.4.2.

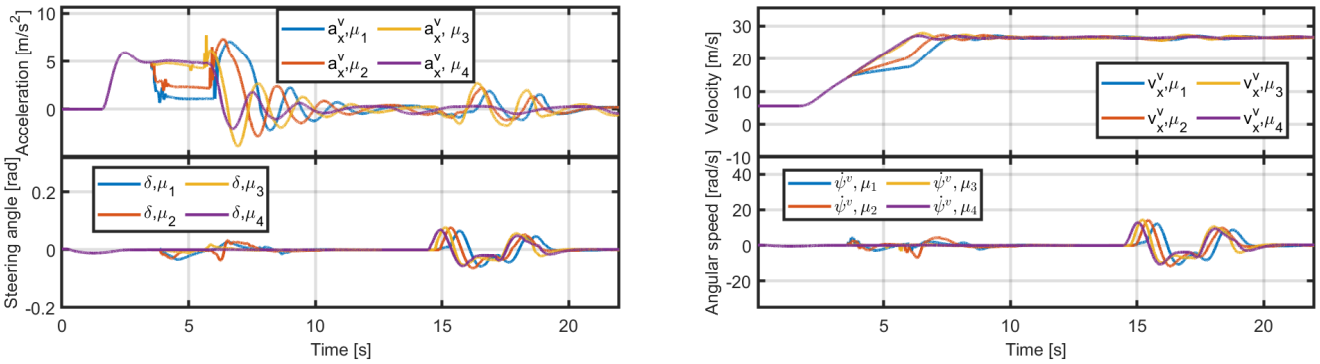


Figure 2.18: Comparison of the proposed control system for various friction coefficients μ_i of the friction bump, driven over with the left side wheels during acceleration. Details are explained in Section 2.4.2.

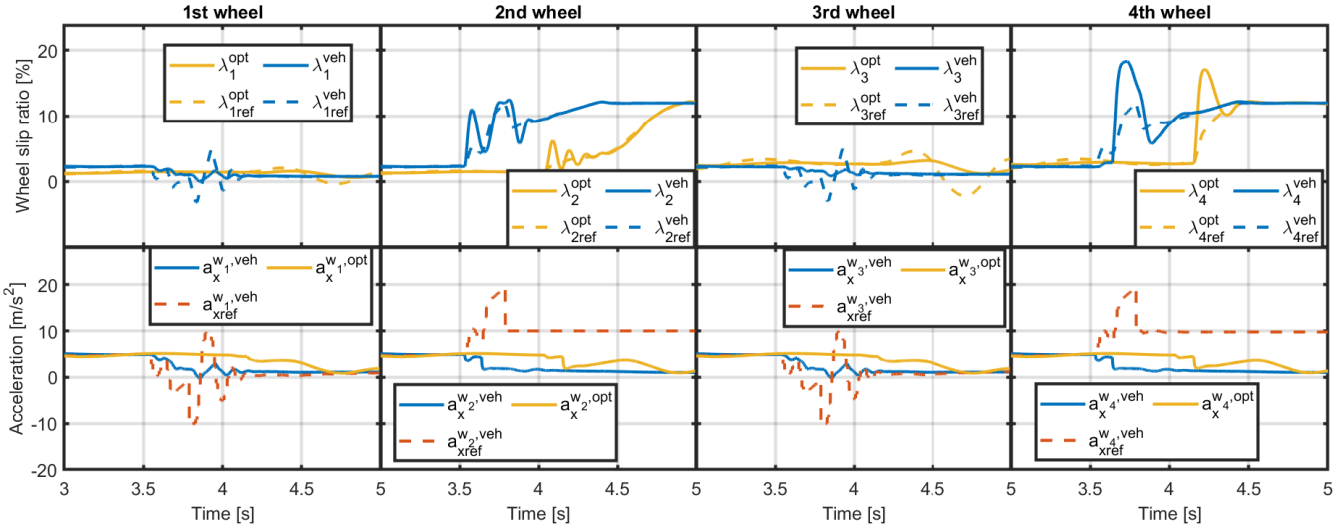


Figure 2.19: Detailed comparison of the proposed allocation scheme (veh superscript) and the optimization-based allocation scheme from [3] (opt superscript) for the lowest friction coefficient $\mu_1 = 0.15$ of the friction bump. Wheel variables ($a_x^{w_i}$ and λ_i for each wheel) are shown. Details are described in Section 2.4.2.

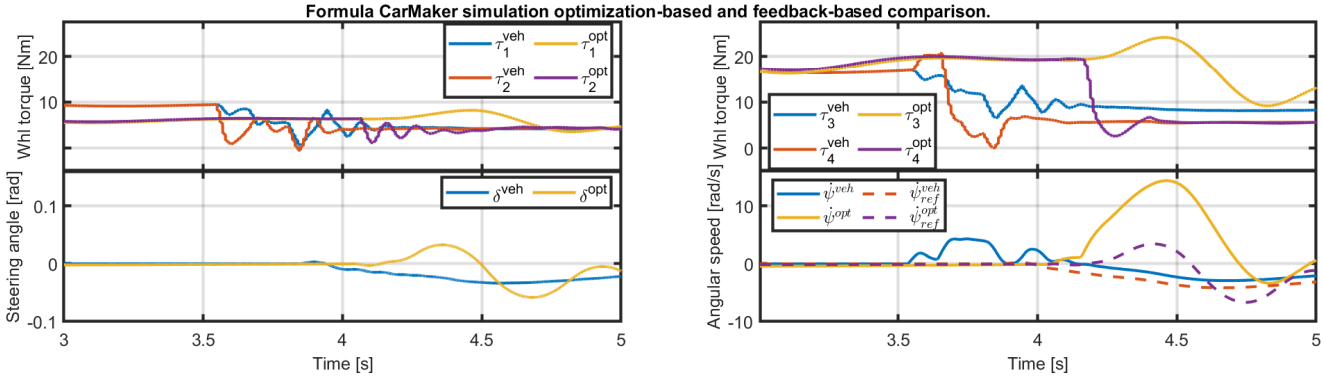


Figure 2.20: Detailed comparison of the proposed allocation scheme (veh superscript) and the optimization-based allocation scheme from [3] (opt superscript) for the lowest friction coefficient $\mu_1 = 0.15$ of the friction bump. Wheel torques τ_i, driver's steering wheel command δ, and vehicle yaw rate $\dot{\psi}^v$ are shown. Details are described in Section 2.4.2.

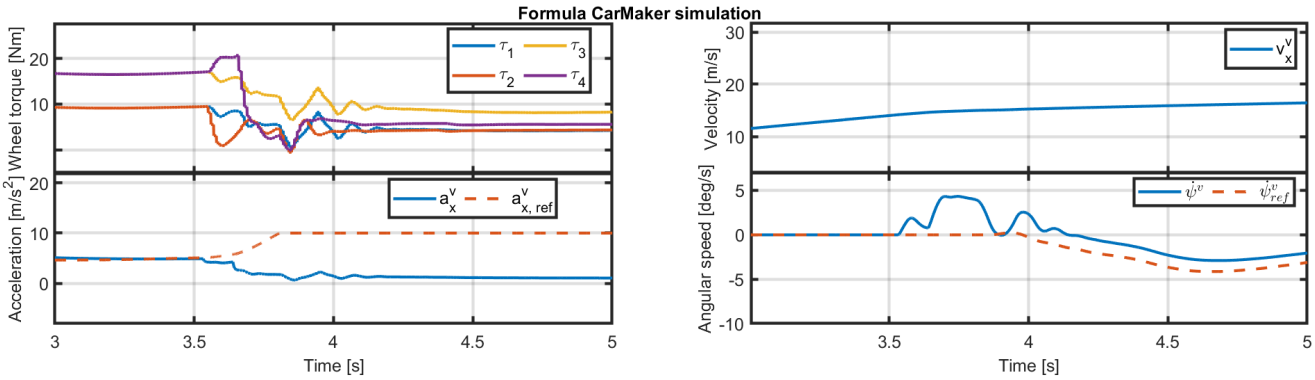


Figure 2.21: Details of the proposed allocation scheme are shown. Wheel torques τ_i, vehicle yaw rate $\dot{\psi}^v$, vehicle acceleration a_x^v, and vehicle velocity v_x^v are depicted for the friction bump's lowest friction coefficient $\mu_1 = 0.15$. Details are described in Section 2.4.2.

Inaccurate λ_{opt} Estimate

The Pacejka magic formula slip curve utilized in the IPG CarMaker model is displayed in Fig. 2.16. The proposed control system was calibrated for the stable portion of the slip curve, where $|\lambda| < \lambda_{\text{opt}} = 12\%$ (see Fig. 2.16). However, determining the λ_{opt} parameter accurately can be difficult, resulting in potential misestimations. The second experiment emulates an incorrect λ_{opt} estimate. In this experiment, the controller limit λ_{max} was set to the erroneous value of $\lambda_{\text{opt}} = 15\%$, which exceeds the slip curve maximum and falls into the unstable region (see Fig. 2.16). According to local linearization analysis at this operating point, the controlled system maintains stability (refer to Section 2.2.2). The performance of the system with the inaccurately estimated λ_{opt} is illustrated in Fig. 2.22. It is evident that the system continues to track effectively even when slip ratio values extend into the unstable slip curve region.

Lateral Stability Functionality

The final experiment showcases the vehicle's ability to track yaw rate using the proposed control system's wheel torques, essentially demonstrating a torque vectoring functionality. The wheels are not steered ($\delta_1 = \delta_2 = 0$) to highlight the steering capability of the proposed solution. The performance of the proposed system, based on the vehicle motion feedback controller, is presented in Fig. 2.23. For comparison, the torque vectoring performance of the optimization-based solution is shown in Fig. 2.24. Both systems are capable of tracking the requested yaw rate references effectively on a nominal surface (the surface for which the control is designed).

2.4.3 Vehicle Models' Parameters

The specific parameters for both the CarMaker model and the Matlab & Simulink-based model are detailed in this section. The model parameters are provided in Table Table 2.1. Additionally, the parameters for the simplified Pacejka magic formula are shown in Table Table 2.2.

Table 2.1: Selected parameters of the models.

Parameter	CarMaker	Simulink
Nominal wheel radius $[m]$	0.205	0.33
Vehicle mass $[kg]$	155.5	1300
Distance from CG to rear wheels $[m]$	0.83	1.63
Max Front Motors power $[kW]$	8	150
Max Rear Motors power $[kW]$	35	200
Distance from CG to front wheels $[m]$	1.18	1.74

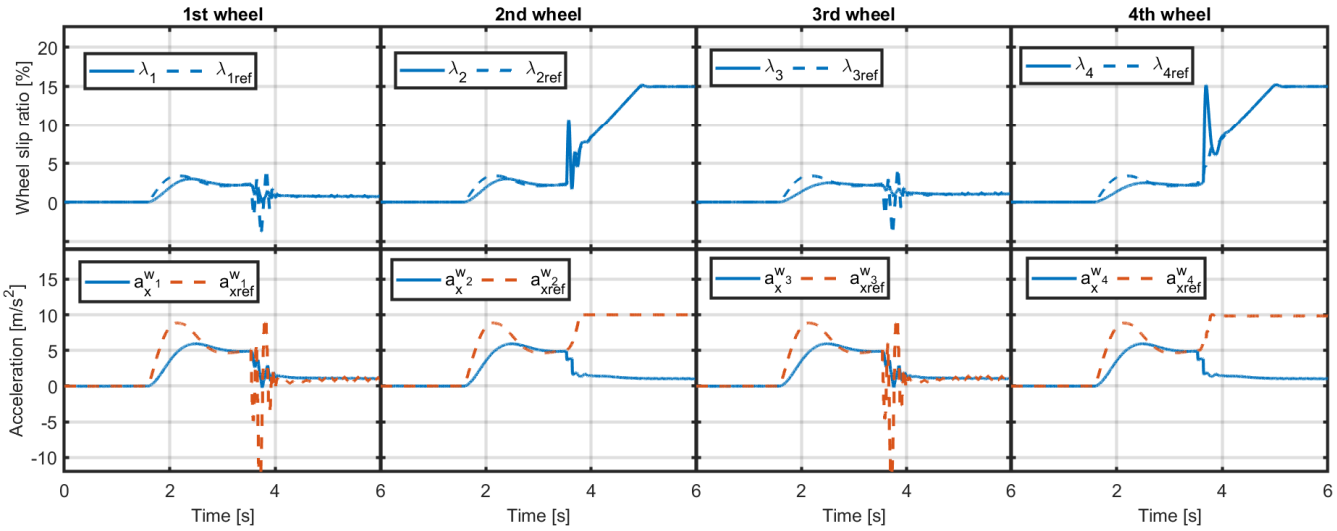


Figure 2.22: Details of the proposed allocation scheme are shown. Wheel variables ($a_x^{w_i}$ and λ_i for each wheel) are depicted for the friction bump's lowest friction coefficient $\mu_1 = 0.15$. The maximum slip ratio $\lambda_{\max}$ was set to 15%, simulating an incorrect λ_{opt} parameter estimation. Details are described in Section 2.4.2.

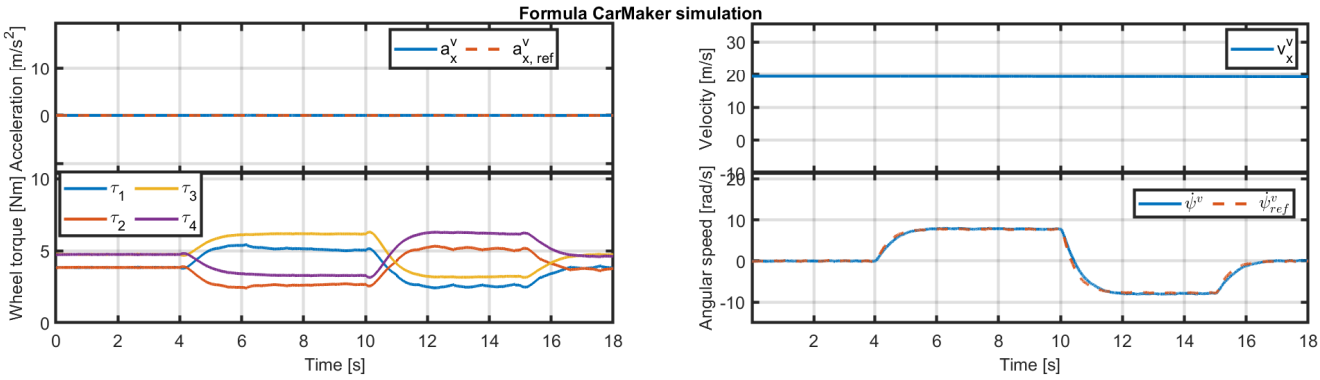


Figure 2.23: Torque vectoring experiment described in Section 2.4.2. The proposed control allocation performance is shown. Vehicle acceleration a_x^v, velocity v_x^v, yaw rate $\dot{\psi}^v$, and wheel torques τ_i; $i \in 1, 2, 3, 4$.

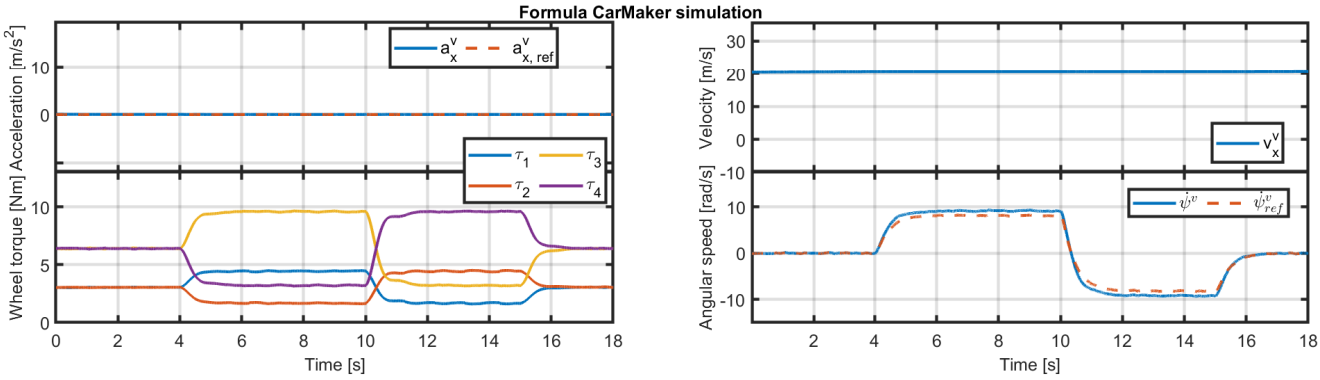


Figure 2.24: Torque vectoring experiment described in Section 2.4.2. The optimization-based control allocation proposed in [3] is shown. Vehicle acceleration a_x^v, velocity v_x^v, yaw rate $\dot{\psi}^v$, and wheel torques τ_i; $i \in 1, 2, 3, 4$.

Table 2.2: Pacejka parameters of the models.

Model	B	C	D	E
Simulink model	7	1.6	1	-0.5
CarMaker model	0.1	1.8	1	-0.95

2.5 HiL Validation of Brake Blending

A friction brake Hardware-in-the-Loop (HiL) stand was developed to evaluate the proposed approach in a real world. The HiL stand is depicted in Fig. 2.25. It consists of a brake-by-wire unit, a PC, and a brake caliper.

The PC runs the simulation environment (Simulink and CarMaker) and connects to the brake-by-wire unit via a USB to CAN PEAK device converter. The brake-by-wire unit is a mechatronic device that regulates mechanical brake pressure using digital communication over the CAN bus. The brake-by-wire unit includes actuators, sensors, and a control unit that communicates with the PC over CAN. It controls the motor controllers responsible for generating brake pressure and measures the resulting brake pressure. The hydraulic system comprises a master cylinder actuated by the brake-by-wire unit and a brake caliper. For a more detailed description of the brake-by-wire unit and HiL test stand, refer to [44]. The mechanical torque acting on the wheels is calculated from the generated pressure using a measured static look-up table.

2.6 Proposed Brake Torque Blending Validation

The proposed control system, incorporating CP point motion feedback and a brake torque blending mechanism as detailed in Section 2.2 and Section 2.3, was validated through selected test scenarios. Additionally, the friction brake HiL setup (refer to Section 2.5) was integrated with both simulation environments (Simulink and CarMaker) to assess the approach using actual hardware. In these experiments, it was assumed that the battery's SOC exceeded 80%, leading to a setting of $\xi = 0.2$.

2.6.1 Mathematical Model Experiment

Initially, the non-linear mathematical model described in section Section 2.1 was employed to validate the control system in a scenario involving ϵ transition, as shown in Fig. 2.26. The variable ϵ represents the tire-to-road interface, accounting for various factors such as changes in friction coefficient μ, tire wear, dynamic load transfer, shifts in the center of gravity, and alterations in vehicle weight.

Understanding how these changes impact the tire-to-road interface model, specifically

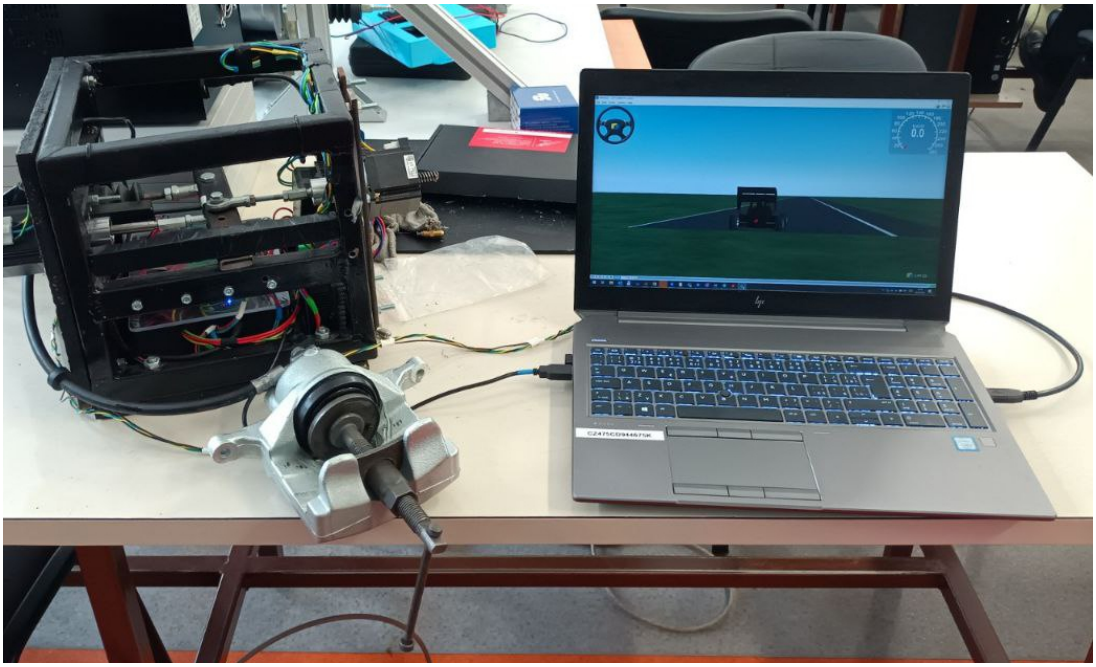


Figure 2.25: Hardware-in-the-loop setup for validating friction brakes.

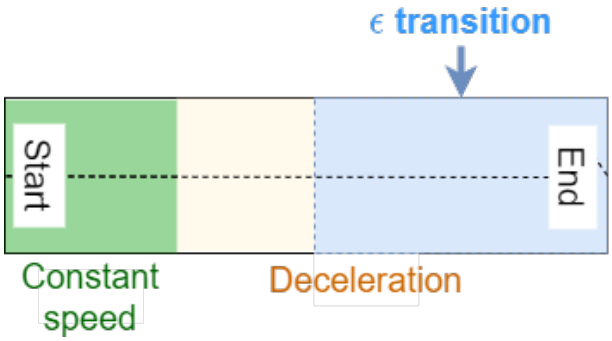


Figure 2.26: Validation scenario with ϵ transition used in the mathematical model (refer to Section 2.1).

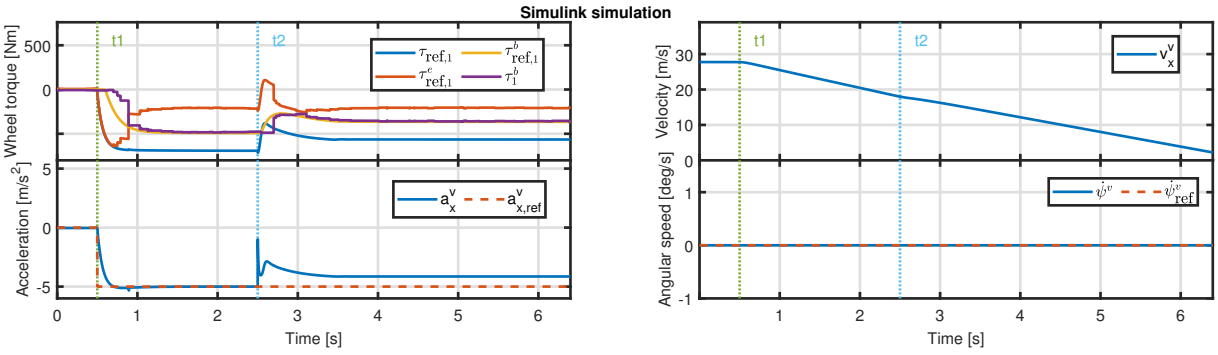


Figure 2.27: Selected variables for the mathematical model experiment described in Section 2.6.1. The top left figure shows the brake torque blending result employing the HiL unit. The other figures depict the vehicle-level variables.

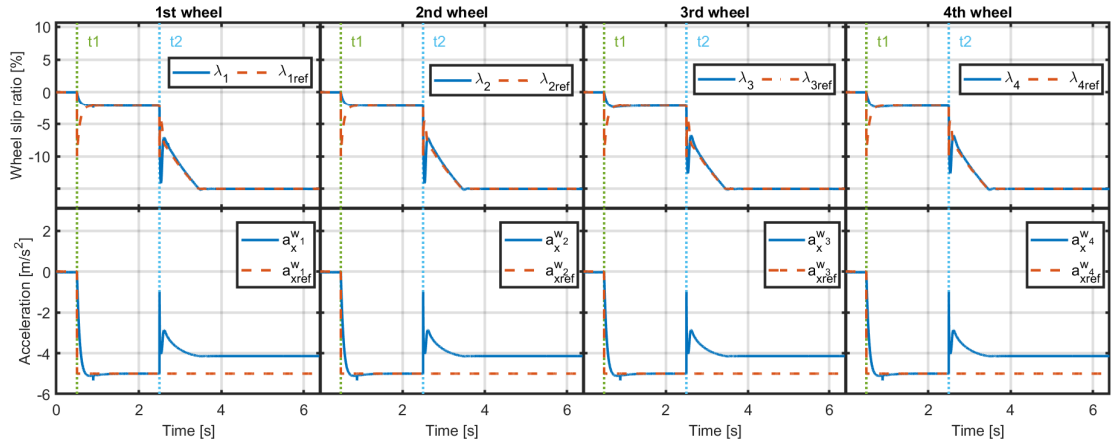


Figure 2.28: Wheels' acceleration $a_x^{w_i}$ and slip ratio λ_i tracking using the motion feedback controller in the Simulink validation environment. Experiment details are provided in Section 2.6.1.

the Pacejka magic formula (2.16), is crucial. Surface changes are modeled by variations in the friction coefficient μ, represented by changes in the tire-to-road interface variable ϵ. Changes in tire properties are reflected by the parameters A, B, C, D, with ϵ approximating the change in parameter D. Additionally, changes in ϵ can also represent variations in the normal force $F_z^{b_i}$, encompassing changes in the vehicle's center of gravity, dynamic load transfer or different loading conditions.

The figures Fig. 2.27 and Fig. 2.28 illustrate this experiment. Initially, the vehicle travels at approximately 100 km/h until time t_1, where it begins braking. At time t_2, the ϵ drops to 0.3, simulating the surface change. The experiment demonstrates the vehicle's ability to stop despite a significant change at the tire-to-road interface (a 70% change in ϵ).

The CP motion feedback controller generates reference brake torques for each wheel, tracked by the brake blending mechanism described in Section 2.3. Wheel slips are controlled and maintained below predefined limits (15%), ensuring safety and maximum brake torque derived from the Pacejka slip curve (see Fig. 2.16). The tracking and blending of brake torques are depicted in Fig. 2.27. The friction brakes deliver the low-frequency portion of the brake torque, while the electronic motor handles the dynamic portion. This arrangement ensures proper execution of dynamic maneuvers with the e-motor's assistance, while providing high-power braking with friction brakes. The dynamic changes are managed by the e-motor, highlighting the importance of reserving some e-motor torque for fast dynamic changes. The tuning parameter ξ determines how much torque should be reserved, balancing performance (low ξ) and economy (high ξ). The ξ value can also be adjusted based on the battery's SOC.

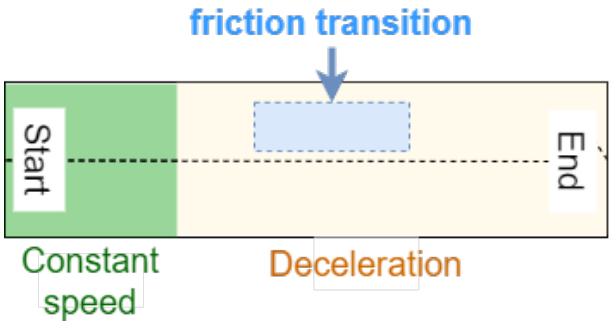


Figure 2.29: Validation scenario with friction coefficient μ split used with the CarMaker model.

2.6.2 CarMaker Model Experiment

The CarMaker experiment was conducted using the CarMaker validation environment and the CarMaker formula model described in Section 2.4.2. The CarMaker driver controlled the vehicle, operating the steering wheel, accelerator, and brake pedal. The experiment scenario is outlined in Fig. 2.29, with corresponding signals shown in Fig. 2.30 and Fig. 2.31. The vehicle maintains a constant speed of 100 km/h until time t_1, then starts braking while passing a friction transition on one side of the vehicle (the μ split scenario with $\mu = 0.15$ representing an icy road). The vehicle encounters the icy patch with its first wheel at approximately time t_2 and exits the patch at time t_3.

The significant reduction in traction force and brake torque on one side, due to the slip ratio controller, is compensated by the CP motion feedback controller by reducing the torque on the other side to prevent the vehicle from rotating. This reduction needs to be rapid to counteract the vehicle yaw rate disturbance, with the e-motor torque managing the dynamic change and the friction torque handling steady-state brake torques. This experiment confirms the CP motion control system's ability to track the vehicle yaw rate reference (as shown in Fig. 2.30). Slip ratios λ_i and wheel pivot point accelerations $a_x^{b_i}$ are not tracked due to the limited brake power of the CarMaker formula model.

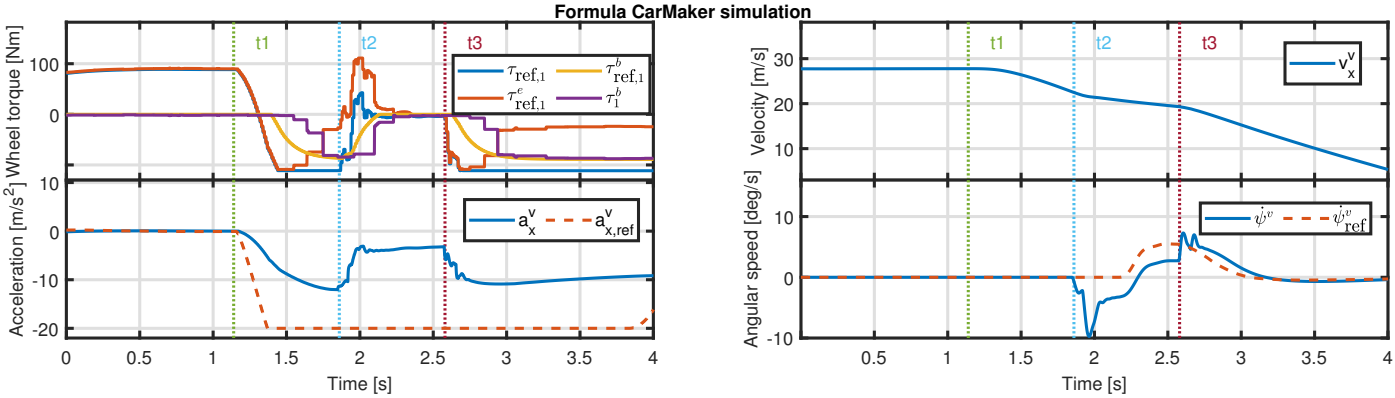


Figure 2.30: Selected variables for the CarMaker experiment described in Section 2.6.2. The top left figure shows the brake torque blending result using the HiL unit. The other figures display vehicle-level variables.

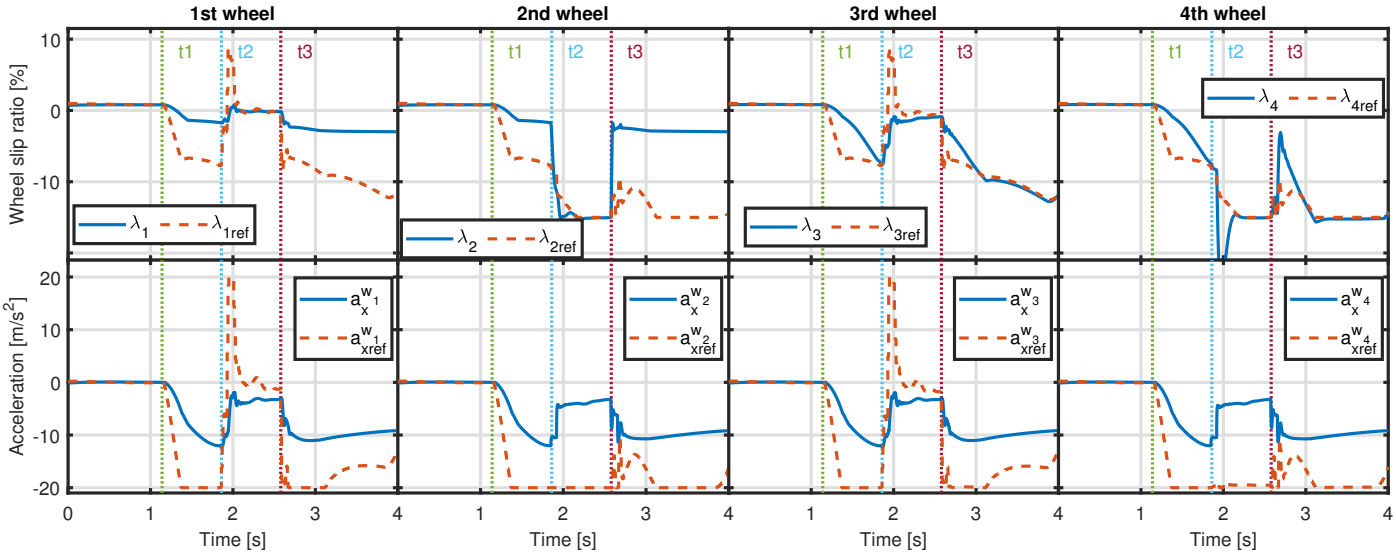


Figure 2.31: Wheels' acceleration $a_x^{w_i}$ and slip ratio λ_i tracking using the motion feedback controller in the CarMaker validation environment. Experiment details are described in Section 2.6.2.

2.7 Summary

An innovative alternative to traditional control allocation methods is proposed in [13, 14], and is elaborated upon in this section. The suggested control system fundamentally changes the existing control paradigm, which typically allocates torques (slip ratios, or forces) directly to the wheels in an open-loop manner. Instead, the proposed system translates the vehicle's velocity or acceleration tracking problem to the wheel pivot points, inherently solving the control allocation task. This approach is detailed in Section 2.2. The validation of this system using the Matlab & Simulink model (Section 2.1) is presented in Section 2.4.1. Further validation and in-depth analysis using the CarMaker model are discussed in Section 2.4.2. The application of this system to the brake system is described in Section 2.3, and the HiL stand used for validating the brake torque blending mechanism is introduced in Section 2.5. The results of the HiL-CarMaker-Simulink co-simulation test scenarios are presented in Section 2.6.

The proposed control system offers numerous advantages over traditional traction systems based on direct force/slip ratio allocation. These benefits are explored in the following section.

2.7.1 Benefits of the Proposed Control System

Among the various benefits of the proposed control system are:

Inherent Control Allocation

The transformation from the vehicle Center Point (CP) to any wheel pivot point is unambiguous for vehicle state variables. This ensures that the set of wheel-level reference signals is unique for any vehicle motion. The combination of this transformation with feedback control of the vehicle/wheel state variables inherently resolves the control allocation problem, unlike systems based on traction force control [3, 15, 22, 24, 26, 45, 41], where control allocation must be explicitly managed. The inherent adjustment of applied traction torque for each wheel is demonstrated in the experiment with variations in ϵ_i values (see Section 2.4.1) in Fig. 2.10 and Fig. 2.11 (at times t_1 and t_3). The control law automatically adjusts the torque to track the velocity reference signal and reflect changes in road conditions and traction capabilities. This inherent allocation is also shown in the CarMaker experiment (see Fig. 2.14).

Robustness to Center of Gravity (CG) Location and/or Wheels' Steady-State Normal Loading

The velocity control is robust to variations in wheel normal force/load. Moreover, the proposed method can inherently provide the distribution ratio of the wheel's normal force (assuming uniform road friction properties). The impact of CG position and overall vehicle mass on traction torque allocation is illustrated in Fig. 2.11. Initially, the vehicle accelerates with the CG positioned rearwards of the vehicle CP, resulting in higher torques on the rear axle (τ_4 and τ_3) compared to the front axle (τ_2 and τ_1). Changes in ϵ_i at time stamps t_1 to t_4 can represent sudden shifts in CG location and overall vehicle mass. The value of ϵ_i scales the $\mu_i F_z^{w_i} D$ term (see eq. (2.16)), where the normal force $F_z^{w_i}$ depends on CG location and vehicle mass during steady-state driving.

Robustness to Dynamic Load Transfer

During maneuvers, the normal force on the wheels dynamically changes. The control system's invariance to CG location implies robustness to dynamic load transfer. Road grade and vehicle roll changes, interpreted as variations in normal force, are handled similarly to dynamic load transfer. Moreover, the nonlinear simulation model includes load transfer effects via the suspension model (see Section 2.1 and [30]).

Robustness to Tire-to-Road Interface Variation

The feedback control of vehicle velocity and wheel angular speed ensures robustness against road condition uncertainty. The ϵ_i value scales the $\mu_i F_z^{w_i} D$ term, modeling road condition variations. Figures mentioned earlier demonstrate this point, particularly Fig. 2.11.

Standard Instrumentation

The proposed system does not require advanced instrumentation beyond what is standard in modern vehicles. The instrumentation level is comparable to that of vehicles equipped with four e-motors, ABS, and ESP systems.

Inherent Wheel Safety Limits Preservation

Wheel traction is considered lost when the slip ratio exceeds the range where the slip curve derivative is positive ($|\lambda| \geq \lambda_{\text{opt}}$ in Fig. 3.2, eq. (2.16)). The λ_{opt} parameter is a direct limitation in the wheel-level control law. It is assumed that the λ_{opt} parameter is known or estimated, a challenging problem discussed in [40, 17] and Chapter 3. The proposed control system maintains the slip ratio λ within prescribed boundaries (defined by the

λ_{opt} parameter), ensuring maximum brake torque delivery, derived from the Pacejka slip curve. This is achieved by bounding the maximum $\lambda_{i,\text{ref}}$ value in the $a_{x,\text{ref}}^{b_i}$ acceleration controller to λ_{opt}.

Unifying Traction System Design

The control strategy integrates the functionality of all conventional wheel-level ADAS traction components, such as ABS, ESC, and ASR.

System/Vehicle-Level Design

The proposed traction control strategy is designed at the vehicle system level, providing higher integration of traction functionality, improved performance, and robustness. The algorithm's low complexity enhances functionality while reducing development and deployment time and costs.

Yaw Rate Tracking and Preference Over Longitudinal Dynamics Tracking

The proposed transformation introduces a tunable parameter that influences the preference between yaw rate and longitudinal dynamics tracking. This functionality is demonstrated in the EFORCE formula CarMaker experiment (see Fig. 2.14 and Fig. 2.15). The anti-symmetric action of the slip ratio generating yaw rate $\dot{\psi}^v$ is evident between the 15th and 20th seconds in Fig. 2.14, inherently implementing Torque Vectoring functionality. The preference for yaw rate over longitudinal dynamics was tested in the double lane change maneuver, with the difference between $\Gamma = 10$ and $\Gamma = 0$ clearly visible in Fig. 2.14 (yaw rate tracking).

Enhanced Integration and Coordination

The proposed control system enhances the integration and coordination of various vehicle control systems. By unifying the control of traction, braking, and stability functions, the system ensures seamless operation across different driving scenarios. This holistic approach not only improves performance but also simplifies the overall vehicle control architecture, making it easier to implement and maintain.

Scalability and Adaptability

The control system is designed to be scalable and adaptable to different vehicle types and configurations. Whether it is a passenger car, a sports vehicle, or a heavy-duty truck, the system can be tuned to meet specific performance requirements. This adaptability ensures that the benefits of the proposed control strategy can be realized across a wide range of

vehicles, providing manufacturers with a flexible solution to enhance vehicle safety and performance.

Cost-Effectiveness

The low complexity of the proposed control algorithm translates to reduced computational requirements and lower costs associated with hardware and software development. Additionally, the system's ability to integrate with existing vehicle control systems without requiring significant modifications further enhances its cost-effectiveness. This makes it an attractive solution for manufacturers looking to improve vehicle performance without incurring high costs.

Chapter 3

Tire-to-road Interface Estimation

Vehicle dynamics control and assistance systems such as anti-lock brake system (ABS), traction control (TC), electronic stability program (ESP), and many other systems are defined by the tire-to-road interface properties. Unfortunately, the variability in these interface properties is described by a highly nonlinear and uncertain model. Therefore, conventional control solutions (such as the ones mentioned) are usually designed to be robust to ensure vehicle safety and stability even in such varying environment. However, this robustness often comes at the cost of vehicle performance.

A very illustrative example of this robustness-for-performance trade-off is the on-off behavior of the ABS algorithm. Braking performance could be significantly increased if the maximum tire traction force was utilized for the entire braking period, without the relaxation phase typical of ABS. A sketch illustrating how ABS works is shown in Fig. 3.1. The actual ABS torque pattern is depicted in red, while the optimal brake torque pattern is shown in dashed blue. The ABS algorithm reduces the braking torque to prevent wheel lock-up, but this relaxation phase, although necessary to prevent lock-up, reduces overall braking performance. The figure also shows the ideal braking torque in the dashed blue line.

The vehicle traction force F_x is derived from the tires and their model. There are numerous tire models, such as the Burckhardt model, Brush model, and Pacejka magic formula model, among others. The wheel traction force (in the x direction) in these models is defined similarly to the dry friction force as:

$$F_x = F_z \, \mu(\lambda), \tag{3.1}$$

where F_z represents the wheel's normal loading, and μ the friction coefficient. The friction coefficient μ is usually modeled as a function of the wheel longitudinal slip ratio λ in a tire model. The topic of slip ratio λ estimation is well-covered in studies such as [46, 47, 48]. This chapter, however, focuses on the estimation of the optimal slip ratio λ_{opt}, which

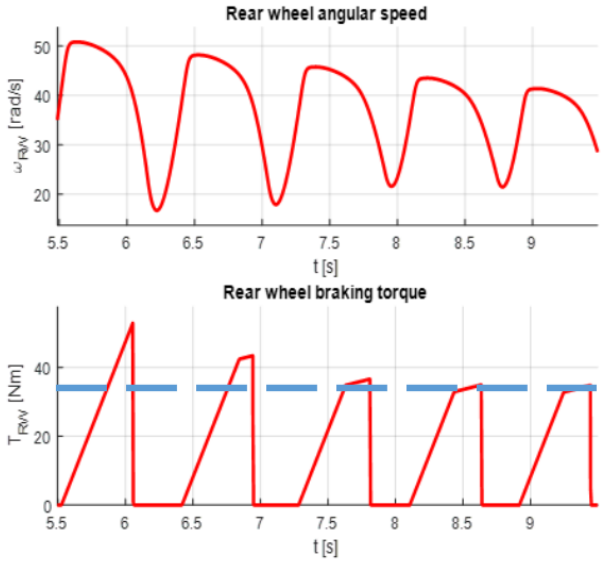


Figure 3.1: Anti-lock brake system (ABS) operation. Red lines represent the ABS operation, the blue line represents the ideal braking torque.

provides the maximum traction force in contrast to the slip ratio estimation, which is not covered here.

This chapter including the figures is based on my previously published work in [17, 18].

Most tire models have a unique maximum friction coefficient $\mu_{\max}$, achieved at a unique optimal slip ratio λ_{opt}. This relationship is sketched as:

$$\mu_{\max} = \mu\left(\lambda_{\text{opt}}\right), \tag{3.2}$$

where $\mu(\lambda)$ represents the tire model (in this chapter, the Pacejka magic formula is used – details are described in Section 3.2).

Knowledge of $\mu_{\max}$ is usually crucial for vehicle dynamics control and is commonly utilized in many advanced driver assistance systems (ADAS), particularly in vehicle traction systems such as anti-skid regulation (ASR), ABS, ESP, and TC [49, 50]. Estimating the maximum friction coefficient $\mu_{\max}$ is a widely studied problem [51, 52, 53, 54, 55]. However, environmental changes (wet concrete, snow, ice, etc.) or variations in tire properties (wear, different types, etc.) cause the maximum friction coefficient $\mu_{\max}$ to be obtained at slightly different λ_{opt} values. The shift in λ_{opt} is often neglected in many studies because its estimation is considered very challenging. As a result, many studies assume λ_{opt} to be constant [3, 36, 41, 56]. However, assuming a constant λ_{opt} may lead to performance losses of several percent in the worst-case scenarios of λ_{opt} shift (see Fig. 3.2). The effect of estimation errors in λ_{opt} is analyzed in Section 3.5.2. It is important to note that for given conditions (weather, tires, etc.), λ_{opt} is a constant that physically corresponds to

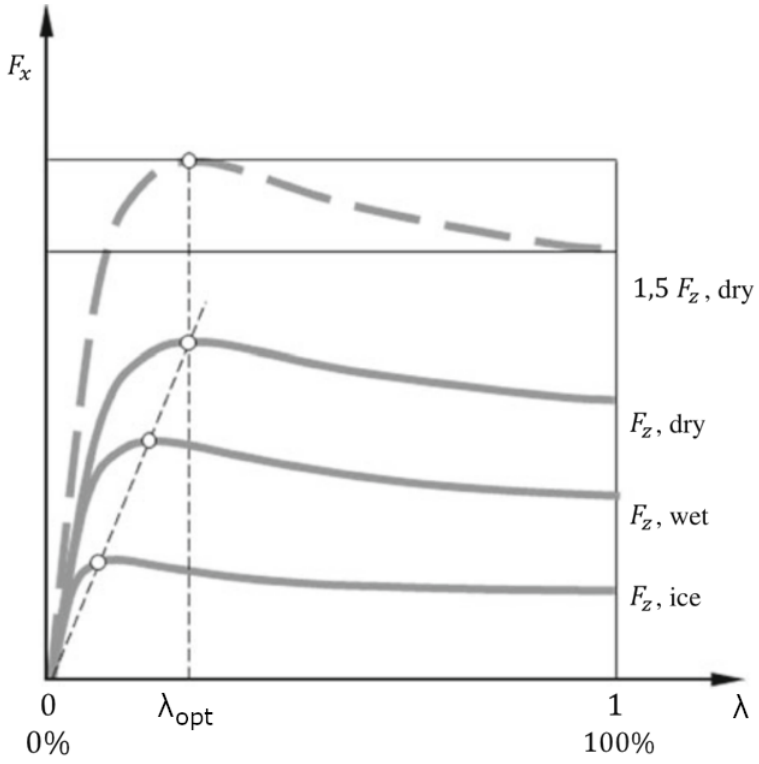


Figure 3.2: Longitudinal slip curve. Adopted from [2].

the highest possible traction force of the wheel.

3.1 Optimal Slip Ratio Estimation

This chapter employs the well-known Pacejka magic formula as defined in [35]. While it may seem insufficient to estimate only λ_{opt} without knowing the friction coefficient $\mu_{\max}$, this assumption is not valid for many systems.

Knowing just the value of λ_{opt} is sufficient and critically important not only in conventional systems like ABS or TC but also in cutting-edge works such as [3, 13]. Studies like [3, 13, 36, 41] require only the slip ratio λ_{opt} that corresponds to the maximum available friction $\mu_{\max}$, not $\mu_{\max}$ itself. In many control systems, the estimated value of λ_{opt} is not even safety-critical as it is used merely as a limit or reference (in a slip ratio controller [13, 57, 58] and Chapter 2) or as an initial value (for ABS systems [50]). Improved knowledge of λ_{opt} can significantly enhance these systems' performance. Furthermore, knowing λ_{opt} is not only valuable for cars but also for other land vehicles such as trains [59] or planes [47]. In [59], the authors set λ_{opt} to a constant value for a railway vehicle, where performance can be improved in the same manner as discussed in this chapter. Additionally, the $\mu_{\max}$ can be derived as a by-product of the proposed UKF-based estimator. The estimated $\mu_{\max}$ from the UKF-based estimator was used to automatically label camera images in [18]. This use case, where the slipperiness of the road is estimated from camera

images, is presented in Section 3.6 and previously published in [18]. The work presented in [18] is multidisciplinary. My specific contribution was preparing the dataset used for neural network training and validation. The actual training and validation of the neural network were conducted by the other authors of the paper.

3.1.1 Main Contributions

The main contributions of this chapter are as follows:

- A novel real-time estimation method of λ_{opt} addressing the simplification of λ_{opt} shifts. This chapter does not focus on slip ratio λ estimation (published in [17]).
- A novel architecture for a UKF-based λ_{opt} estimator using traction force estimation (published in [17]).
- A novel RLS-based algorithm for estimating λ_{opt} using force estimation (published in [17]).
- Comparison of the proposed methods in real-world experiments (published in [17]).
- The UKF-based maximum friction coefficient estimator for automatic labeling of camera images used in neural network-based predictor of surface slipperiness (published in [18]).

3.1.2 Related Research

Estimating the properties of the road surface is challenging due to the nonlinear nature of tire-to-road interactions and the significant levels of uncertainty and variability. Numerous papers have tackled this challenge, and estimation algorithms can generally be categorized into two main branches:

- Estimation based on direct surface measurements.
- Estimation based on responses of wheel and vehicle dynamics.

Estimation Based on Surface Measurements

Various measurement methods can be used to infer the friction properties of the road surface. Common methods include optical cameras, infrared thermometers, and temperature sensors. In [60], F. Holzmann proposed a predictive method for road friction estimation using measurements from cameras and microphones. This method involves matching the measurements to pre-stored reference samples. Another example of surface measurement

is presented in [61], where a thermometer detects the heat energy emitted during freezing. Ultrasonic measurements are used in [62], while optical position detection is employed in [63].

However, these methods have several disadvantages, such as challenges in sensor placement and cost. Additionally, they only capture certain aspects of the complex force calculation chain (refer to Section 3.2.2 for details on force calculations). Most studies focus on measuring road surface conditions, neglecting important tire properties such as newness, wear, and aging. In contrast, dynamics-based estimation naturally considers all these effects.

Estimation Based on Wheel and Vehicle Dynamics

The majority of state-of-the-art studies primarily focus on estimating the maximum friction coefficient $\mu_{\max}$ [51, 52, 53, 54, 55], often neglecting the shift in λ_{opt}. However, this chapter emphasizes estimating the optimal slip ratio λ_{opt} that corresponds to the maximum traction force (or friction coefficient $\mu_{\max}$). Some existing studies specifically focus on estimating λ_{opt}. In [40], the authors use the sign of the slip curve slope to estimate λ_{opt}. Another method is presented in [64], where an adaptive neuro-fuzzy inference system is used for estimating λ_{opt}.

In [65], the slope of the slip curve is estimated based on vehicle dynamics response, and λ_{opt} is determined from the change in the slope's sign. A similar idea is presented in [66], where the slip curve slope at the origin (zero slip ratios) is used for estimating $\mu_{\max}$. In [67], the authors propose a peak slip line for estimating the lateral optimal slip angle α_{peak}, relying on the brush tire model properties. Generally, the majority of the previous works techniques for λ_{opt} estimation are rule-based. For instance, in [58], the slip curve slope is observed, and the optimal slip ratio λ_{opt} estimate is updated by a step δ (a parameter of the estimation algorithm) based on the curve's sign. Notably, this study focuses on railway vehicles rather than road vehicles. Another rule-based estimation is presented in [57], where the slip curve is divided into regions based on the slope value, and the update step for λ_{opt} estimation is selected accordingly. The estimated λ_{opt} is then used as a reference for the slip ratio controller.

In general, estimators based on vehicle dynamics response can estimate not only the net force but also its maximum and the corresponding slip ratio λ_{opt}. Both proposed methods for λ_{opt} estimation in this chapter fall into this category. An advantage of both proposed estimators is their decentralized nature. They are based on the dynamics of a single wheel and require no additional parameters beyond those in the wheel dynamics equation (details in Section 3.2). This provides an advantage over estimators based on the entire vehicle system [68], which may require additional parameters such as vehicle mass.

The proposed estimators are also less complex than the one in [68] and easier to adapt to various vehicles and powertrain architectures. Additionally, the UKF-based estimator can provide the maximum friction coefficient $\mu_{\max}$ as a by-product. Extending the RLS-based estimator to provide $\mu_{\max}$ is also possible, though not shown here.

3.1.3 Validation of the Estimators

The proposed estimators were validated through simulations using the model from Section 3.2 and real-world experiments using a subscale radio-controlled validation platform (refer to Section 3.4). For simulation validation, a nonlinear high-fidelity twin-track model with an implemented traction ellipse (refer to [2]) was used. The model implementation can be found in [30]. Moreover, the estimators were used to train and validate the surface slipperiness predictor (refer to Section 3.6 and [18]) using the platform described in Section 3.4.

3.1.4 Chapter Structure

The structure of this chapter is organized as follows. First, the models employed for the development and validation of the estimators are presented in Section 3.2. The two proposed estimators are described in Section 3.3. Next, the remotely controlled subscale platform is introduced in Section 3.4. Further, the estimators are validated through simulations and experiments, and the results are presented in Section 3.5. Finally, a neural network-based predictor of surface slipperiness is presented in Section 3.6. The UKF-based maximum friction coefficient estimator is used for automatic labeling of the camera images to train and validate the predictor.

3.2 Mathematical Simulation Models

As the derivation of the estimators depends solely on the wheel dynamics equation, the complete twin-track model will not be reiterated here. However, the full twin-track model is employed for validating the estimators. For comprehensive information, please refer to [2] and Section 2.1.

Twin Track Model

The full twin-track model is implemented in MATLAB & Simulink (see [30]). This model is later used to validate the proposed estimators through simulations. To provide more realistic data, measurement noise is added to the model outputs.

The twin-track model includes:

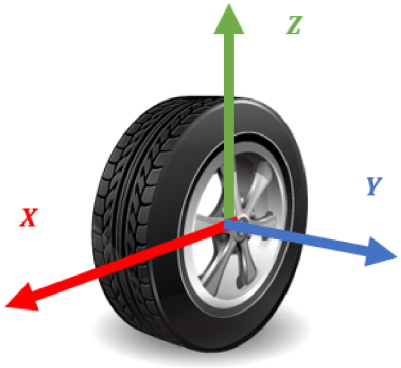


Figure 3.3: Wheel Coordinate system.

- Nonlinear vehicle dynamics with suspension inspired by [2] and implemented in [30].
- The model consists of four wheels, where each wheel encompasses:
  - Wheel dynamics.
  - Pacejka magic formula tire model.
  - Friction ellipse/circle, also known as Kamm's circle.

The mathematical model is designed for an all-wheel-independent-drive vehicle, meaning a vehicle with an e-motor for each wheel. However, the estimators are general and can be applied to other architectures as long as the underlying assumptions are met (see Section 3.3).

3.2.1 Wheel Nonlinear Dynamics Model

This section revisits the wheel-level model (see Section 2.1.3). The wheel model is associated with the wheel coordinate system presented in Fig. 3.3. Wheel dynamics are characterized as follows:

$$J_i \cdot \dot{\omega}_i = \tau_i^t - F_x^i \cdot r_{w_i} - \tau_i^{res}, \tag{3.3}$$

where J_i $[kg \cdot m^2]$ is the wheel's moment of inertia along the wheel shaft (y) axis (including driveline rotational inertia), ω_i $[rad/s]$ is the wheel's angular speed along the wheel shaft axis, τ_i^t $[Nm]$ is the wheel's traction torque, F_x^i $[N]$ is the longitudinal traction force generated by the particular wheel, τ_i^{res} $[Nm]$ represents torques combining wheel and e-motor resistance effects, and r_{w_i} $[m]$ is the wheel's effective radius. The subscript $i = 1, 2, 3, 4$ denotes the i-th wheel.

The specific variables and parameters of the wheel model are illustrated in Fig. 2.4b. The road condition, wheel normal force/load, and the effect of the wheel's lateral force

are modeled by modifying the wheel's longitudinal traction force F_x^i based on the wheel's longitudinal slip curve (see Fig. 3.2) and traction ellipse (see Fig. 2.4a), as described in Section 3.2.2.

3.2.2 Tire-to-Road Interface

The tire-to-road interface model is implemented similarly to the tire-to-road interface model presented in Section 2.1.4. Traction forces on the wheel result from the interaction between the tire and the road surface, commonly described using slip variables and slip curves for both longitudinal and lateral directions. The widely used Pacejka magic formula is employed for tire modeling in both longitudinal and lateral directions (refer to [35, 2]).

Additionally, the traction ellipse is used to capture the relationship between the longitudinal and lateral traction properties of the wheels (for more details, see [35, 34] and Section 2.1.4). This ellipse represents the maximum friction force generated by the tire-to-road contact patch under combined longitudinal and lateral loading, ensuring that the combined force from acceleration or braking during cornering maneuvers does not exceed the vertical force $\mu_i F_z^i$ applied to the wheel by the vehicle (see Fig. 2.4a). The friction ellipse is also known as Kamm's circle. For more details, refer to Section 2.1.

The parameters of the Pacejka magic formula can be measured, e.g., on a chassis dynamometer, or estimated using different approaches as stated in Section 3.1. The Pacejka magic formula parameters for the longitudinal direction used for the simulation are shown in Table 3.1.

3.3 Estimators

The objective of the estimator is to characterize the tire-to-road traction properties of the wheel in the longitudinal direction, with a focus on estimating λ_{opt}. A moderate error in the estimate of $\hat{\lambda}_{\text{opt}}$ around the maximum does not result in significant deviations in the maximum traction force, as illustrated in Fig. 3.5. This is one of the main reasons why the λ_{opt} shift is often disregarded in many studies (see Section 3.1 for details). However, the consequences of neglecting the λ_{opt} shift are explored in Section 3.5.2.

The estimators described in this section are designed for individual wheels rather than the entire vehicle, reducing the complexity of the estimation problem. Each wheel of the vehicle has then an estimator implemented as shown in Fig. 3.4.

With this architecture, the estimators can be easily adapted to any type of wheeled vehicle, such as cars, trucks, or even trains, as long as the following assumptions are met:

- Known wheel inertia J and radius r_w.

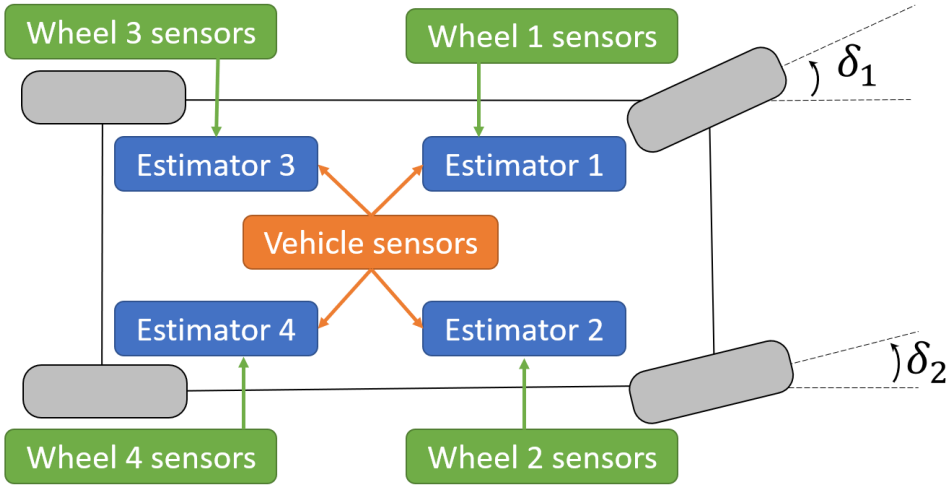


Figure 3.4: Each wheel is equipped with its own estimator.

- Measured or estimated wheel torque τ_i, wheel speed ω_i, and wheel pivot point longitudinal speed v_{ix}.
- Optionally, for improved performance, measured or estimated wheel traction force F_x^i and wheel normal force F_z^i.

Two estimator structures are presented and compared. Both algorithms rely on the measurement or estimation of traction force F_x, which is also detailed here. The components presented include:

- Traction force F_x estimation algorithm.
- Recursive Least Squares (RLS) based algorithm for λ_{opt} estimation.
- Unscented Kalman Filter (UKF) based algorithm for λ_{opt} estimation.

All components are based on wheel dynamics (see eq. (2.10)) and are individually implemented for each wheel. Both estimation algorithms (UKF- and RLS-based) use the wheel longitudinal traction force and wheel normal loading/force as inputs. The longitudinal force F_x estimation serves as the input to the UKF- and RLS-based estimators. Force estimation is discussed in Section 3.3.1. If a force sensor is available in the application, the force estimator can be omitted or replaced with direct measurements.

Additionally, knowledge of the wheel normal force F_z is assumed to improve estimation results. However, if F_z is not accessible (neither measured nor estimated), it can be set to one as $F_z = 1$. This approach is applied to the validation platform described in Section 3.4, and results indicate that this approximation performs well (see Section 3.5.2). This approximation introduces another limitation: F_z should remain approximately constant, meaning that either braking or acceleration conditions must be selected to ignore dynamic load transfer effects.

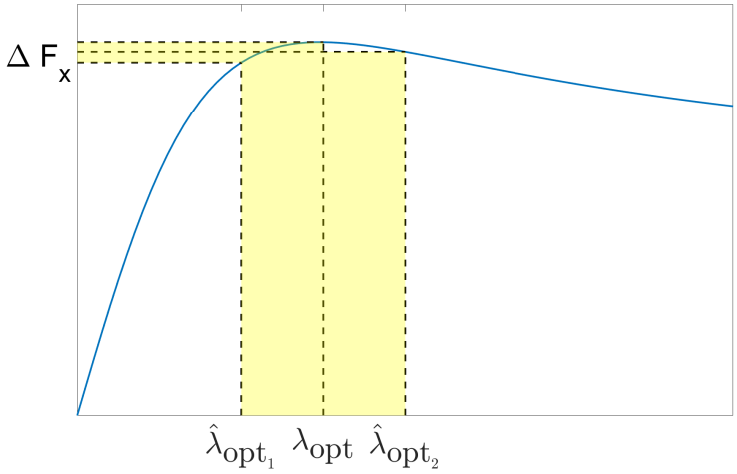


Figure 3.5: Illustration of traction force error caused by an imperfect maximum slip ratio estimate.

All blocks are represented using discrete time. The sampling time should be less than or equal to $10ms$, as this period is generally sufficient for the discretization of the wheel dynamics equation (3.3). For the validation platform (see Section 3.4) and simulations, the sampling time $T_s = 10ms$ was chosen. The validation platform (see Section 3.4) is also running with a sampling time of $10ms$.

3.3.1 Force Estimation

The force estimation is based on wheel dynamics (see eq. (2.10)) and can be implemented individually for each wheel. Euler's method is used to discretize the dynamics to align with the discrete nature of the UKF and RLS. The time-discrete system model used in the estimation algorithm is:

$$\omega[t+1] = \omega[t] + \frac{T_s \cdot r_w}{J} F_x[t] + \frac{T_s}{J}\tau[t]. \tag{3.4}$$

The traction force $F_x[t]$ is then defined as:

$$F_x[t] = \frac{J}{T_s \cdot r_w}\kappa[t] - \frac{1}{r_w}\tau[t], \tag{3.5}$$

$$\kappa[t+1] = a\ \kappa[t] + (1-a)\left(\omega[t] - \omega[t-1]\right) \approx \frac{d\omega}{dt}, \tag{3.6}$$

$$a = e^{-\omega_c T_s}, \tag{3.7}$$

where T_s is the sampling time and ω_c is the filter crossover frequency. The variable κ represents the filtered (first-order low pass) difference in wheel angular velocity ω. To

compute the force from this equation, the required parameters (J, r_w) and the applied traction torque $\tau = \tau^t - \tau^{\text{res}}$ are assumed to be known. The resistance torque τ^{res} effects (rolling resistance, friction, other losses) are assumed to be included in the torque τ. A similar force estimation algorithm (in continuous time) was previously introduced and validated in [3].

3.3.2 RLS-based Estimation Algorithm

The core idea of the RLS-based estimator is to approximate the derivative of the Pacejka slip curve (see eq. (2.16)) with a first-order polynomial (Taylor expansion) near the current slip ratio value. The slope of this polynomial is then interpreted as the derivative of the slip curve. Next, the slip curve is approximated by a quadratic function, where the coefficients of the quadratic polynomial are obtained through constrained least squares minimization. The maximum of this quadratic approximation is used to estimate the optimal slip ratio λ_{opt}. The slip curve slope is typically high in the linear region, decreases to zero in the region of maximum traction slip ratio λ_{opt}, and becomes negative beyond λ_{opt}.

The RLS algorithm computes the parameters of the affine function approximation (first-order polynomial), which is useful for further processing. A similar technical approach was used in [65], where a linear function (without a constant term) was used in the small slip ratio region (linear region) to estimate the maximum friction coefficient.

The first-order polynomial approximation parameters are estimated using the RLS algorithm following equation

$$\mu(\lambda) = \frac{F_x}{F_z}(\lambda(t), \dots) \approx a_1(t) \cdot \lambda(t) + a_0(t) + e(t), \tag{3.8}$$

where F_x is the traction force, F_z is the wheel normal loading, λ is the slip ratio, e is measurement noise, and $\mathbf{a} = [a_1, a_0]^T$ are parameters to be estimated by the RLS algorithm. The wheel slip ratio λ and the longitudinal traction force F_x are assumed to be estimated or measured (see Section 3.3.1). Optionally, F_z^i – i-th wheel normal force/loading – can be estimated or measured (see Section 3.3.1). If it is not available, set it to 1, as mentioned in Section 3.3 (as was done for the validation platform – see Section 3.4).

For the RLS parameter estimation, eq. (3.8) can be rewritten as:

$$\mu(t) = \mathbf{z}^T(t)\,\mathbf{a}(t) + e(t), \tag{3.9}$$

where $\mathbf{z}^T(t) = [\lambda(t), 1]$ is the measurement data (regression vector). The algorithm is then executed iteratively, where each loop consists of:

a. **Regression vector** – Measure and build regression vector $\mathbf{z}(t)$ for the current time step.

b. **Prediction error** – Compute the prediction error $e(t)$ from the previous time step parameter estimates as follows:

$$e(t) = \frac{F_x}{F_z}(t) - \mathbf{z}^T(t)\,\mathbf{a}(t-1). \tag{3.10}$$

c. **Update gain** – Calculate the update gain as follows:

$$\mathbf{K}(t) = \frac{\mathbf{P}(t-1)\,\mathbf{z}(t)}{1+\mathbf{z}^T(t)\,\mathbf{P}(t-1)\,\mathbf{z}(t)}, \tag{3.11}$$

and reduce the covariance matrix according to:

$$\hat{\mathbf{P}}(t) = \mathbf{P}(t-1) - \mathbf{K}(t)\,\mathbf{z}^T(t)\,\mathbf{P}(t-1). \tag{3.12}$$

d. **Forgetting** – Exponential forgetting was selected for use in this thesis, but it can be easily extended to other forms of forgetting (e.g., restricted exponential forgetting):

$$\mathbf{P}(t) = \frac{1}{\varphi}\hat{\mathbf{P}}(t), \varphi \in (0,1). \tag{3.13}$$

e. **Parameter update** – Update the parameter estimate as follows:

$$\mathbf{a}(t) = \mathbf{a}(t-1) + \mathbf{K}(t)\,e(t). \tag{3.14}$$

The RLS algorithm itself is computationally efficient and suitable for embedded applications. Numerically stable methods are recommended for embedded targets. For instance, the lower diagonal (LD) factorization with LD factor updates using dyadic reduction for restricted exponential forgetting (see [69]) is a stable and numerically efficient algorithm with high potential for embedded applications.

RLS-based λ_{opt} Estimation

The objective of the λ_{opt} estimator is to determine the current $\hat{\lambda}_{\text{opt}}(t)$ estimate based on the parameters $\mathbf{a}(t)$. By understanding the slip curve slope a_1 for a given λ, one can estimate $\hat{\lambda}_{\text{opt}}$ as described in Algorithm 1.

Updates to the $\hat{\lambda}_{\text{opt}}$ estimate occur only when the absolute slip ratio is non-zero (no updates are made while the vehicle is coasting) and when the vehicle is moving in a

straight line or nearly so (minimizing the influence of the wheel slip angle α on the slip curve). This threshold is labeled as λ_{coast} in Algorithm 1.

Because the slip curve is symmetric for both positive and negative slip ratios, we describe only the positive slip ratio region for simplicity. Extending to the negative slip ratio region is straightforward. The slip curve is approximated with a quadratic function $q(\lambda)$:

$$q(\lambda) = b_2\lambda^2 + b_1\lambda + b_0. \tag{3.15}$$

The function q is concave (not convex) if and only if b_2 is negative. Thus, b_2 must be negative to reflect the concave nature of the slip curve (see Fig. 3.2). The maximum of this quadratic approximation q is used as the current time step's $\bar{\lambda}_{\text{opt}}$ estimate, determined by setting the derivative to zero:

$$\frac{\partial \mu_i}{\partial \lambda} = \frac{\partial q}{\partial \lambda} = 2b_2\lambda + b_1 \stackrel{!}{=} 0, \tag{3.16}$$

$$\bar{\lambda}_{\text{opt}} = -\frac{b_1}{2b_2} \tag{3.17}$$

The coefficients b_i for the quadratic slip curve approximation are calculated using a constrained linear least squares problem, minimizing the differences between the derivatives of the linear (parameters a_i) and quadratic (parameters b_i) approximations:

$$\min_{b_1,b_2} \quad \frac{1}{2}\left\| \frac{\partial q}{\partial \lambda} - a_1 \right\|_2^2 \tag{3.18}$$

$$\text{s.t.} \quad b_2 < 0 \tag{3.19}$$

$$-\frac{b_1}{2b_2} < \hat{\lambda}_{\text{ub}} \tag{3.20}$$

$$-\frac{b_1}{2b_2} > \hat{\lambda}_{\text{lb}}, \tag{3.21}$$

where $\hat{\lambda}_{\text{ub}}$ and $\hat{\lambda}_{\text{lb}}$ are upper and lower bounds on the $\bar{\lambda}_{\text{opt}}$ estimate, for example, $\hat{\lambda}_{\text{lb}} = 0.05$ and $\hat{\lambda}_{\text{ub}} = 0.5$. A history of recent slip curve slope values a_1 and corresponding slip ratios λ are used for estimating the b_i coefficients, e.g., the last 50 samples corresponding to approximately 500 ms. This means only the data from the last 500 ms is used for the λ_{opt} estimation, which corresponds to approximately 7 m of road surface at 50 km/h.

The final $\hat{\lambda}_{\text{opt}}[t]$ estimate is filtered as follows:

$$\hat{\lambda}_{\text{opt}}[t] = c_f\hat{\lambda}_{\text{opt}}[t-1] + (1-c_f)\,\bar{\lambda}_{\text{opt}}, \tag{3.22}$$

where c_f is the filtering constant. The algorithm for estimating the optimal slip ratio $\hat{\lambda}_{\text{opt}}$ is detailed in Algorithm 1.

Algorithm 1: RLS-based λ_{opt} estimation

Output: $\hat{\lambda}_{\text{opt}}[t]$
Input: a_1, λ, $\hat{\lambda}_{\text{opt}}[t-1]$, α
Parameters: $[c_f, \hat{\lambda}_{\text{ub}}, \hat{\lambda}_{\text{lb}}, \lambda_{\text{coast}}] \in (0,1)$
if $|\lambda| > \lambda_{coast}$ *and* $|\alpha| \approx 0$ **then**
  $[b_1, b_2] \leftarrow \text{lsqlin}(a_1, \lambda, \hat{\lambda}_{\text{ub}}, \hat{\lambda}_{\text{lb}})$
  `// lsqlin is defined in Equations (3.18) to (3.21)`
  $\bar{\lambda}_{\text{opt}} = -\frac{b_1}{2b_2}$
  $\hat{\lambda}_{\text{opt}}[t] = c_f \hat{\lambda}_{\text{opt}}[t-1] + (1-c_f)\,\bar{\lambda}_{\text{opt}}$
else
  $\hat{\lambda}_{\text{opt}}[t] = \hat{\lambda}_{\text{opt}}[t-1]$

3.3.3 UKF-based Estimation Algorithm

The UKF-based estimator is based on the wheel dynamics equation (3.3). Inputs to the estimator include the longitudinal traction force F_x, which can be measured or estimated as described in Section 3.3.1, the wheel normal force F_z, wheel torque τ, and wheel slip ratio λ. The estimator's state vector, containing the wheel's angular velocity, is expanded with the Pacejka magic formula parameters (see (2.16)). This state vector includes states x_i $(i = 2, 3)$ representing the Pacejka magic formula parameters B and D, as elaborated later in this section.

The decision to use only two Pacejka magic formula parameters is based on the fact that the other parameters do not vary significantly (the E parameter) and do not significantly influence the slip curve shape (see [70]), thereby simplifying the UKF filter complexity. This approach has been validated through simulations, with results presented in Section 3.5.2.

These parameter states have constant zero dynamics, similar to the Schmidt-Kalman algorithm (see [71]). The UKF model is described as follows:

$$f\left(\mathbf{x}[t], \tau_i[t]\right) = \begin{pmatrix} \frac{T_s}{J}\left(\tau_i[t] + r_{w_i} \cdot F_x\left(\mathbf{x}[t]\right)\right) + x_1[t] \\ \min\left(\max\left(x_2[t], B_{min}\right), B_{\max}\right) \\ \min\left(\max\left(x_3[t], D_{min}\right), D_{\max}\right) \end{pmatrix}, \tag{3.23a}$$

$$g\left(\mathbf{x}[t]\right) = \begin{pmatrix} x_1[t] \\ F_x\left(\mathbf{x}[t], F_z[t]\right) \end{pmatrix} \tag{3.23b}$$

$$F_x\left(\mathbf{x}, \lambda\right) = F_z x_3 \sin\left(C \arctan\left(x_2\lambda - E\left(x_2\lambda - \arctan\left(x_2\lambda\right)\right)\right)\right), \tag{3.23c}$$

where $\mathbf{x}[t+1] = f(\mathbf{x}[t], \tau_i[t])$, $\mathbf{y}[t] = g(\mathbf{x}[t])$, λ is the longitudinal slip ratio, T_s is the sampling time, F_x is the traction force, and F_z is the wheel normal force. The state x_1 corresponds to the wheel angular speed, and the states x_i $(i = 2, 3)$ relate to the Pacejka magic formula, as described above. These coefficients are bounded within a reasonable range using the min and max functions. The designer should determine the min and max bounds. The B parameter (state x_2) according to [70] ranges from 4 to 12, so the bounds could be set from 3 to 14. The $\mu_i D$ (state x_3) parameter product ranges from 0.1 to 1. The UKF algorithm benefits from being a derivative-free method, which means it can handle almost any nonlinearity easily without needing to calculate the derivatives of the model functions (Jacobian matrices). Therefore, the use of max functions is not deteriorating the UKF performance (in contrast to other Kalman Filter algorithms like Extended Kalman Filter).

To use the estimation algorithm, the following UKF input signals must be measured or estimated:

- v_x^i - i-th wheel's pivot point longitudinal speed (in wheel CS) needed for the slip ratio λ_i computation (see eq. (2.15)) - assumed measured/estimated.
- τ_i - i-th wheel's traction torque (input of the system) - assumed measured.
- ω_i - i-th wheel's angular speed - assumed measured.
- F_x^i - traction force generated by i-th wheel - estimated or measured (see Section 3.3.1).
- Optionally, F_z^i - i-th wheel normal force/loading - estimated or measured (see Section 3.3.1). If not available, it is advised to set it to 1, as explained in Section 3.3 (as was done for the validation platform – see Section 3.4). In this case, the state x_3 has a different meaning, representing the maximum force $\phi^i = F_{x,\max}^i = F_z^i \mu_i D^i$. This approximation introduces another limit on the operating conditions of the estimator. F_z should be approximately constant, which means that either braking or acceleration conditions must be selected to neglect the dynamic load transfer effects.

Updates to the estimator are driven by information about the system contained in the innovation $\boldsymbol{\epsilon}(t) = \mathbf{y}(t) - g(\hat{\mathbf{x}}(t))$. The sensitivity of the innovation to the parameters was implemented similarly to [68]. This approach computes the sensitivity of innovation (estimation error) on the parameters and marks all the measurements where the parameter sensitivity is low. Low-sensitivity measurements are not used in the UKF measurement update phase for the estimation of the Pacejka magic formula parameters. The measurement update phase is also conditioned by coasting ($|\lambda| > \lambda_{\text{coast}}$) and straight rides (wheel

slip angle α effect is small). The update is not performed for zero (and almost zero) torque τ applied to the wheel and significant steering wheel angle δ.

UKF-based λ_{opt} Estimation

From the estimated coefficients, the slip curve is reconstructed following the Pacejka magic formula (2.16). The maximum value of the curve is determined using an optimization method based on the SQP algorithm that seeks the maximum value of the $F_x(\lambda)$ manipulating λ. For safety reasons, the optimization is bounded to seek only within a defined region, e.g., $\hat{\lambda}_{\text{opt}} \in (0.05, 0.5)$ as the true λ_{opt} should be located within these boundaries.

3.4 Experimental Platform for Real-World Data Collection

Real-world data, including both camera and vehicle traction system data, were collected using an experimental subscale platform depicted in Fig. 3.6. For this purpose, a 1:5 scale LOSI buggy Radio Controlled car was modified. The vehicle's dimensions are: length 844 mm, width 501 mm, height 308 mm, with a ground clearance of 69 mm and a wheelbase of 552 mm. The bare platform, excluding data acquisition equipment, weighs 13.8 kg, including traction batteries.

The total weight of the platform is approximately 20 kg, and it can achieve a maximum speed of 70 km/h. This setup is ideal for proof-of-concept testing due to its simplicity and ease of use. Compared to full-scale vehicles, this platform reduces test overhead, team workload, and testing time. Moreover, it does not require special test tracks and offers other advantages. However, it presents some challenges, such as limited space for sensors and actuators.

The entire platform is equipped with four Electronic Control Units (ECUs) that handle vehicle motion control, camera and vehicle data acquisition, and essential safety features. The primary ECU hardware is an Intel *NUC7i7BNK* mini PC running Matlab & Simulink. Additionally, a Raspberry Pi 4, an Arduino Nano, and STM32 Nucleo (*STM32L432*) microcontrollers manage the traction control system, data acquisition, and safety functionalities. The platform operates at a sampling rate of 100 Hz.

Actuators

The model is powered by a central BLDC e-motor with Li-Po batteries, configurable for either 4WD or rear 2WD. For this study, rear 2WD was used to measure v_x from the

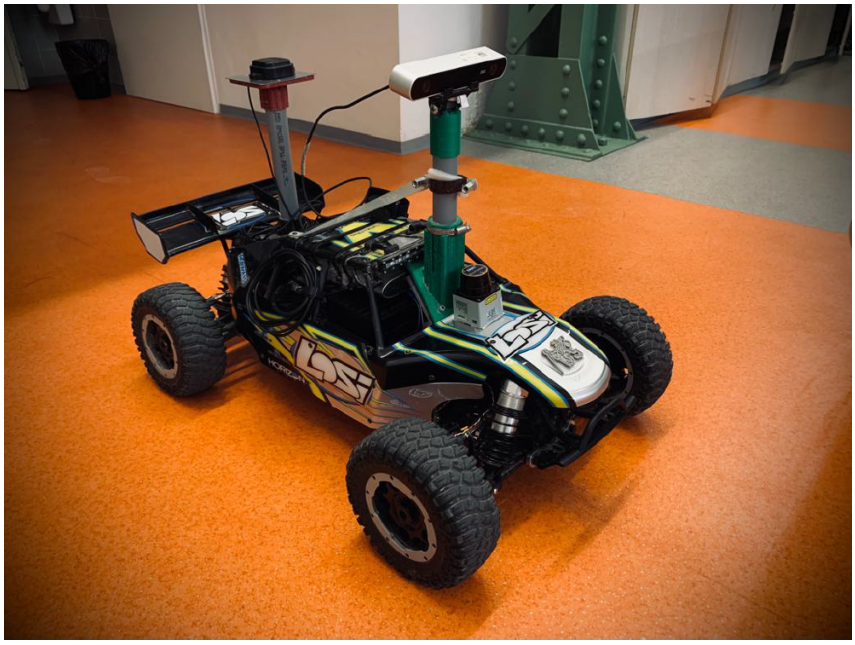


Figure 3.6: RC platform used for validation of the estimators.

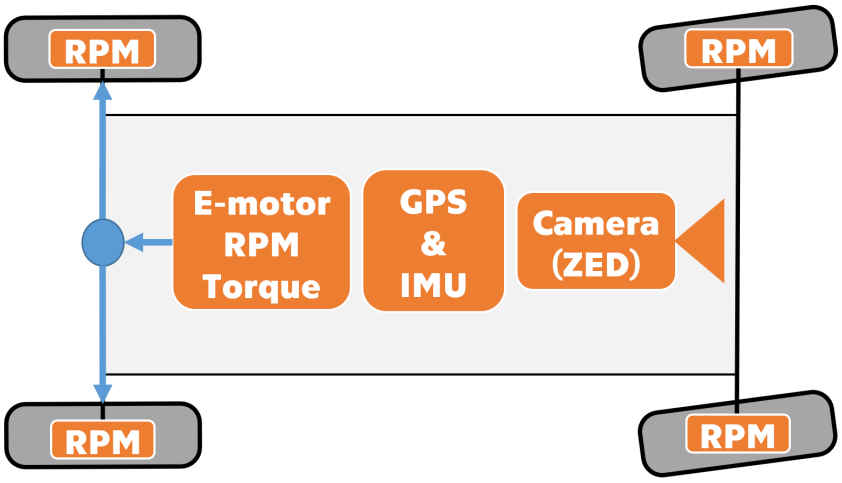


Figure 3.7: RC platform architecture used for validation of the estimators.

front wheels' speed. The model features independently actuated steering for each wheel, utilizing four servos. All actuators are remotely controlled by the human operator of the subscale platform.

Sensors

Each wheel is equipped with a speed sensor implemented using a Hall-effect sensor and magnetic markers distributed around the wheel sensor disk. The BLDC motor's current is measured in two phases (the third is calculated) to compute motor torque using the motor torque constant $k_p[Nm/A]$, where $\tau = k_p \cdot i$. An ideal operation of the differential is assumed, splitting torques equally between the rear wheels. The vehicle also includes a GPS & IMU unit for location measurements, and surface images are captured with a ZED stereo camera, using one camera of the rig.

Accurate vehicle translation speed measurements, required for the slip ratio in the Pacejka magic formula (see (2.16)), are achieved by calculating $v_x = \frac{\omega_1+\omega_2}{2} \cdot r$, where ω_1 and ω_2 are the nondriven front wheels' speeds, and r is the wheel radius. The normal force F_z is set to 1 as it is neither measured nor estimated on this platform.

The necessary signals for the estimators are obtained in this manner. The resistance torque τ_{res} is approximated as a function of the wheel speed $\tau_{res}(\omega)$. This resistance torque is subtracted from the measured motor torque to determine the net torque acting on the wheel.

The RC subscale platform used for validation is shown in Fig. 3.6 and the component block diagram is illustrated in Fig. 3.7.

The friction torque of the wheels was identified as a function of wheel angular speed $\tau_{drag} = \tau_{drag}(\omega)$. Parameters of the wheel dynamics equation were also identified from dynamic responses.

3.5 Simulation and Experimental Validation

This section presents the simulation and real-world experimental validation of the proposed estimation schemes. Initially, the simulation results for the force estimator are shown. Subsequently, the integrated RLS-based and UKF-based estimators are validated through simulations (using the model described in Section 3.2) and real-world experiments using the subscale platform discussed in Section 3.4.

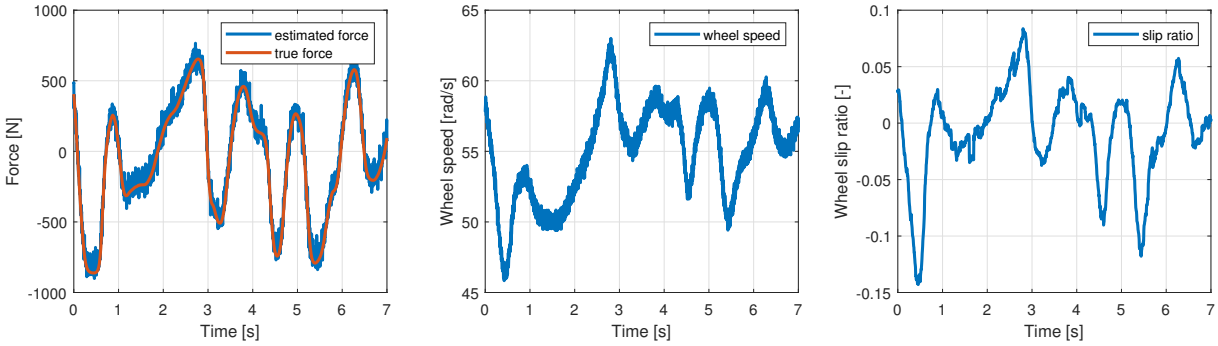


Figure 3.8: Comparison of estimated and true traction forces with torque, speed, and slip ratio at the wheel. Simulations using the nonlinear model (see Section 3.2).

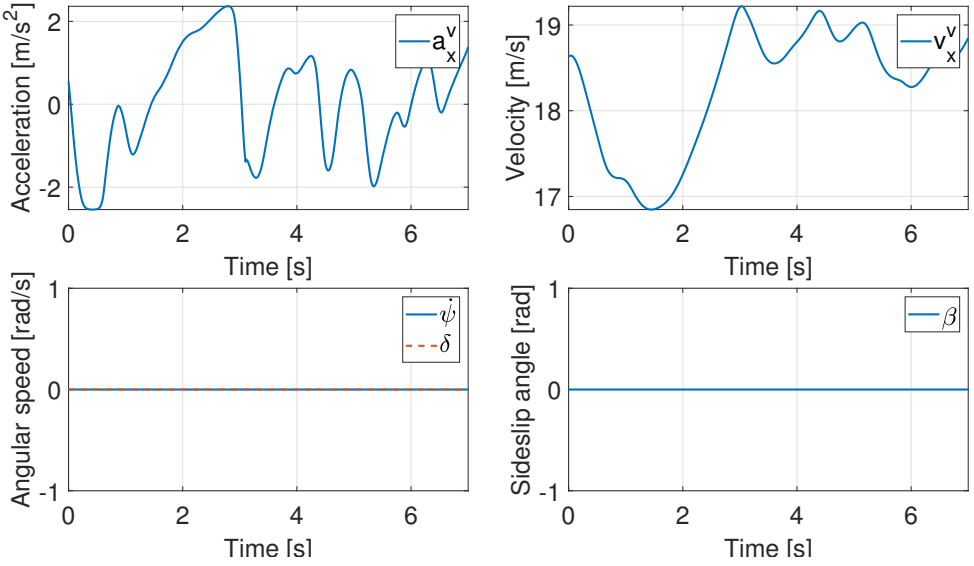


Figure 3.9: Vehicle body variables during the force estimation experiment. Simulations using the nonlinear twin-track model.

3.5.1 Traction Force Estimator - Simulation Results

The traction force estimation algorithm was validated using simulations with the high-fidelity nonlinear twin-track model described in Section 3.2. A comparison of the estimated and true values is shown in Fig. 3.8 (the vehicle body signals of the experiment are presented in Fig. 3.9). The middle plot of Fig. 3.8 displays the added measurement noise, which also affects the force estimation. Noise was introduced to the simulation model output to produce more realistic data. The overall performance appears satisfactory. Validation of the force estimator on the experimental platform is not feasible due to the absence of a force sensor for validation. However, the approach has been validated in real-world experiments by the authors of [3].

	B	C	D	E	t
Original Pacejka coefficients	7	2.5	1	1	$t < 8.5$
Modified Pacejka coefficients	3	2.3	0.7	1	$t > 8.5$

Table 3.1: Pacejka magic formula parameters for simulation experiment shown in Figures 3.10 to 3.12.

3.5.2 RLS and UKF Estimators - Simulations

This section presents a selected simulation experiment. Initially, the starting values for both RLS- and UKF-based estimators are set to an initial value, which is typically inaccurate. This setup demonstrates the convergence of the initial value. Following this idea, the slip curve changes abruptly, simulating a surface change at time 8.5s. The values of the Pacejka magic formula parameters are listed in Table 3.1. A comparison of the estimated and true λ_{opt} for both estimators is shown in Fig. 3.10.

Two plots are provided: the first shows the convergence of the initial value for both the RLS and UKF estimators' $\hat{\lambda}_{\text{opt}}$ estimate and the estimators' response to the slip curve change. The second plot illustrates the ΔF_x error resulting from the incorrect λ_{opt} estimate. The error ΔF_x is calculated as:

$$\Delta F_x = |F_x\left(\lambda_{\text{opt}}\right) - F_x\left(\hat{\lambda}_{\text{opt}}\right)|, \tag{3.24}$$

where $F_x\left(\lambda\right)$ represents the true traction force with the true Pacejka simulation parameters. The λ_{opt} value denotes the true simulation optimal slip ratio, while $\hat{\lambda}_{\text{opt}}$ is the estimated optimal slip ratio. An illustration of the maximum traction force error is shown in Fig. 3.5. The evaluated traction force error is displayed in the bottom plot of Fig. 3.10. The comparison indicates that the UKF estimator is noisier, whereas the RLS estimator is less noisy but exhibits slower convergence.

The values of the Pacejka coefficients are not as intuitive as the $\hat{\lambda}_{\text{opt}}$ estimates. To compare the UKF estimated slip curve shape with the true one, a measure of slip curve error $e\left(t\right)$ is introduced as follows:

$$e\left(\hat{F}_x\left(\lambda\right), F_x\left(\lambda\right), \lambda_{\max}, \lambda_{\min}\right) = \int_{\lambda_{\min}}^{\lambda_{\max}} |F_x\left(\xi\right) - \hat{F}_x\left(\xi\right)| d\xi, \tag{3.25}$$

where F_x is defined via the Pacejka magic formula with true simulation parameters and $\hat{F}_x$ is the same formula but with estimated parameters. The measure (3.25) integrates the force estimation error for a range of the slip curve between $\lambda_{\min} = -0.3$ and $\lambda_{\max} = 0.3$. The normalized error $e\left(t\right)$ is shown in the top plots of Figures 3.11 and 3.12. The bottom plots of Figures 3.11 and 3.12 show the convergence of the slip curve estimates and the true slip curves. The estimation convergence speed is illustrated in Fig. 3.11, while the

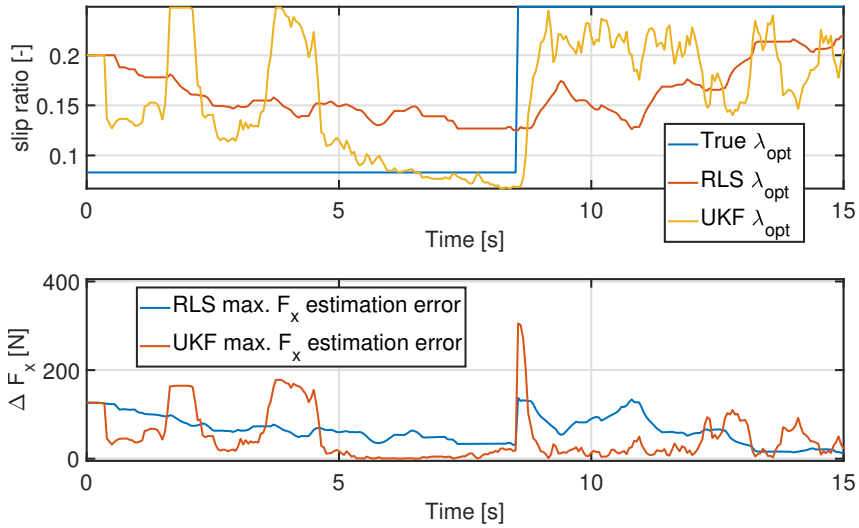


Figure 3.10: Estimation performance comparison of the RLS- and UKF-based estimators. Estimated values are compared with true values. The experiment was performed on the nonlinear model in the simulations (see Section 3.2).

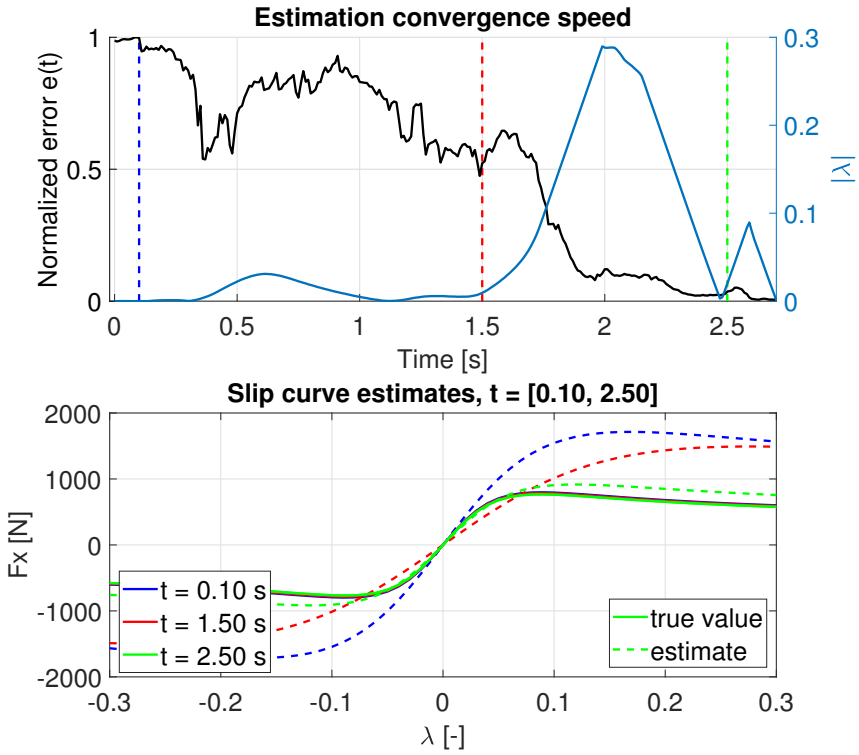


Figure 3.11: Pacejka slip curve estimation convergence. The top figure presents the slip curve estimation error with the slip ratio. Sample slip curves and corresponding slip curve estimates are plotted in the second figure.

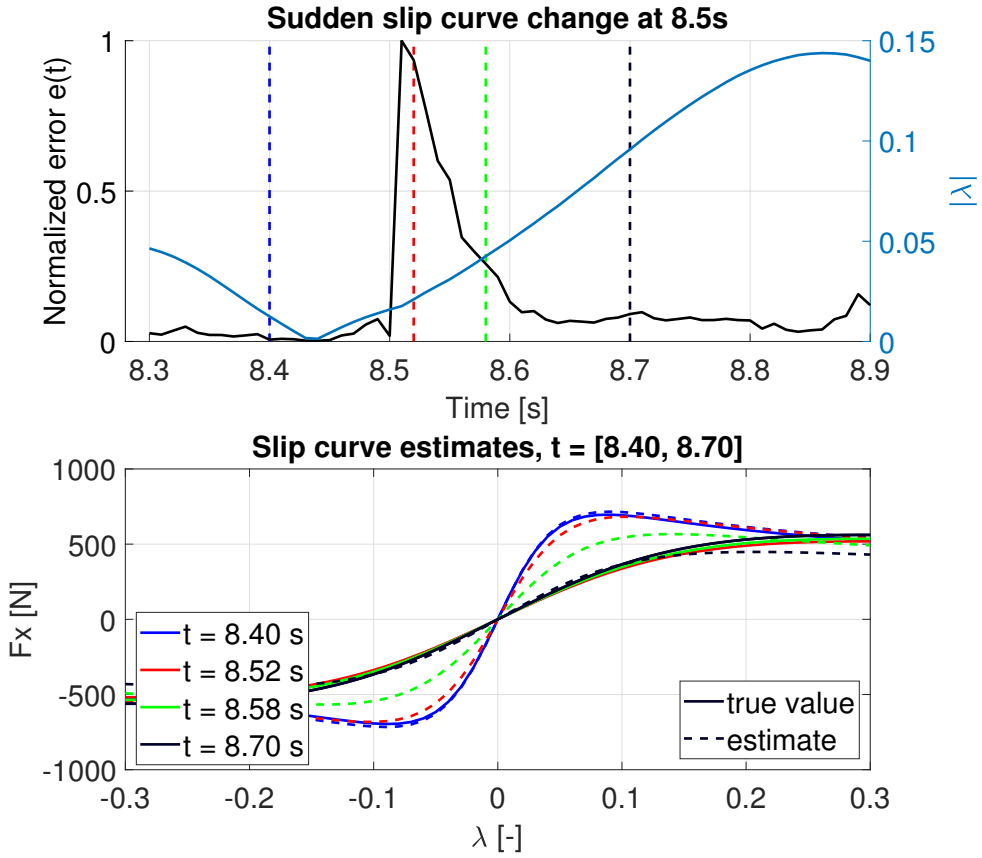


Figure 3.12: Pacejka slip curve estimation error for the slip curve change experiment at time 8.5s. The top figure presents the slip curve estimation error with the slip ratio. Sample slip curves and corresponding slip curve estimates are plotted in the second figure.

abrupt change in the slip curve representing the surface change is depicted in Fig. 3.12. It is evident that the estimation error decreases significantly once the slip ratio rises above a certain level. This behavior is expected because without system dynamics excitation, no information about the slip curve is available in the data. This theoretical limitation affects all dynamics-based estimators. Consequently, vehicle coasting does not provide any information, and data from coasting intervals are excluded from the update phase, as proposed in Section 3.3.

Optimal Slip Ratio Estimation Error Effect

To illustrate the effect of optimal slip ratio estimation error, a simple use case is presented. Assume a vehicle with a slip curve shape representing a slippery surface, as shown in Section 3.5.2 at the beginning of the scenario (refer to Fig. 3.12). The true optimal slip ratio value for the considered case is $\lambda_{\text{opt}} = 0.09$ as shown in Fig. 3.10.

Assume the vehicle travels at an initial speed of $v_0 = 50$ km/h and can theoretically perfectly control the wheel slip ratio to be equal to the optimal slip ratio estimate, i.e., $\lambda = \hat{\lambda}_{\text{opt}}$. Neglecting many other effects, the vehicle braking distance x is roughly determined

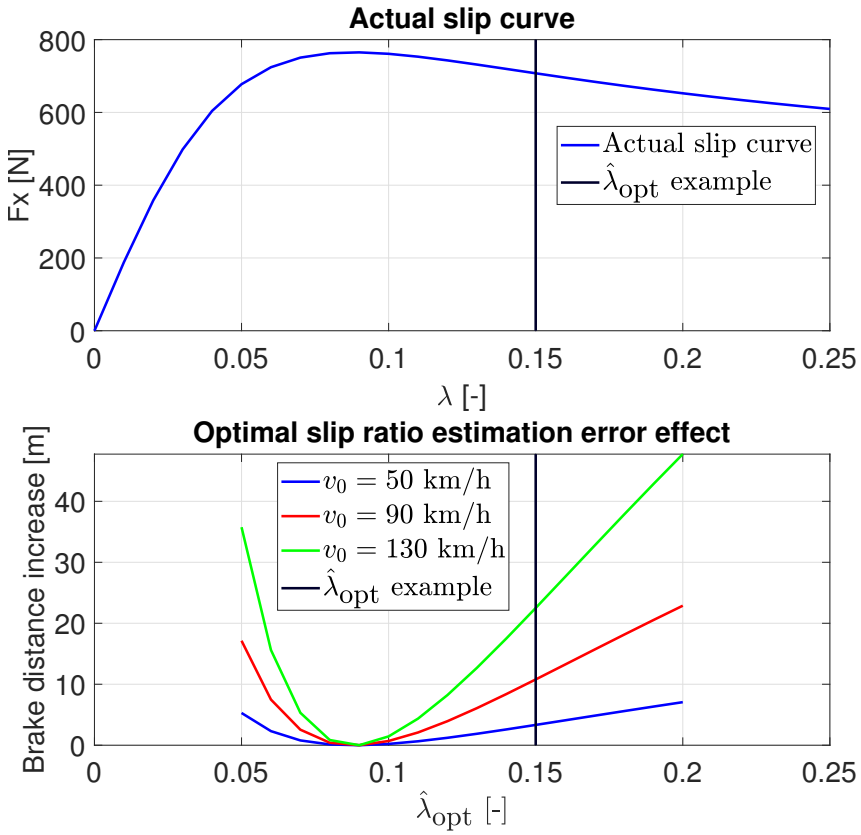


Figure 3.13: The theoretical braking distance increase as a function of the estimated $\hat{\lambda}_{\text{opt}}$ for the initial slip curve, as described in Section 3.5.2. The braking distance increase is shown for three initial vehicle speeds $v_0 = 50, 90, 130$ km/h. See Section 3.5.2 for details.

by Newton's second law as follows:

$$F_x = ma_x, \tag{3.26}$$

$$t = \frac{v_0}{a_x}, \tag{3.27}$$

$$x = \frac{1}{2}a_x t^2, \tag{3.28}$$

where the vehicle mass is $m = 1300$ kg. Assuming the constant $\hat{\lambda}_{\text{opt}}$ estimate as proposed in state-of-the-art studies focusing on traction control [3, 36, 41, 50], e.g., $\hat{\lambda}_{\text{opt}} = 0.15$ as in [36], one wheel generates $F_x^w\left(\hat{\lambda}_{\text{opt}}\right) = 707$ N, resulting in a vehicle braking force $F_x = 707 \cdot 4 = 2,828$ N, and thus a braking distance $x_1 \approx 44$ m. However, the true $\lambda_{\text{opt}} = 0.09$ generates a wheel traction force $F_x^w\left(\hat{\lambda}_{\text{opt}}\right) = 765$ N, resulting in a vehicle braking force $F_x = 765 \cdot 4 = 3,060$ N, which results in a braking distance $x_2 \approx 41$ m. Perfect knowledge of the true λ_{opt} reduces the braking distance theoretically by 3 m, which is a 7% reduction in braking distance. The theoretical braking distance increase as a function of the estimated $\hat{\lambda}_{\text{opt}}$ is shown in Fig. 3.13 for three initial vehicle speeds $v_0 = 50, 90, 130$ km/h.

3.5.3 Experimental Validation - Real World Experiments

Three test rides were conducted, each on a different surface: tarmac, gravel, and snow. All rides were performed using the platform described in Section 3.4 for both nondriven wheels. A wheel is selected, and the results are presented for clarity, as the results do not significantly differ between wheels. Time ranges with no update phase of the UKF, such as coasting and wheel side slips, are clipped from the data because only constant estimates are observable. Unfortunately, the platform measurements have high measurement noise for the slip ratio, as seen in Fig. 3.14. Subsequently, as the entire product ϕ is estimated, the bounds for UKF x_3 should be set accordingly. For the experimental platform (see Section 3.4), a range of 0.3 to 10 was used.

The measurement noise is mainly caused by wheel rotation velocity measurement noise and low traveling speeds. Once higher speeds (above 5 m/s) are achieved, the noise will be much less significant. Therefore, the UKF filter was tuned robustly, resulting in longer convergence times. However, it has been shown that both estimators work even with such noisy data. The noise level in a full-scale vehicle is expected to be lower than that in the validation platform, where the raw slip ratio measurement is employed.

A comparison of applied torques can be seen in Fig. 3.14. The separation of the three surfaces is already slightly visible in this comparison, but it is not as clear as in Fig. 3.15. Tarmac shows the highest torques, while gravel and snow exhibit medium to low torques with similar slip ratio values. The measured motor torque dependence on the measured slip ratio is shown in the figure. The mean torque for a particular slip ratio and surface type is also shown.

The $\hat{\lambda}_{\text{opt}}$ estimate for both UKF-based and RLS-based estimators is shown in Fig. 3.16 for a selected part of the tarmac surface ride. Both estimators have approximately the same estimate of the optimal slip ratio λ_{opt}.

The actual Pacejka slip curve with the measured slip ratio λ_i data and the corresponding force estimates are shown in Fig. 3.17 for the UKF-based estimator. The estimation of both parameters with UKF (B and ϕ) is shown in Fig. 3.18. These slip curves are shown together in Fig. 3.15 for better comparison. Finally, the maximum slip ratio $\hat{\lambda}_{\text{opt}}$ estimate comparison for the three surfaces is presented in Fig. 3.19. The tuning of the UKF-based estimator is shown in Table 3.2, and that of the RLS-based one is in Table 3.3. As seen in Fig. 3.19, both estimators approximately match for the tarmac and snow cases after initial convergence. It was demonstrated that despite the different initial conditions of the RLS- and UKF-based estimators, they converge to the same slip ratio λ_{opt} value. This supports the applicability of the RLS- and UKF-based estimators to real full-scale vehicles. The gravel optimal slip ratio λ_{opt} diverges the most among the three surface types. This discrepancy could stem from multiple sources, but the most probable cause

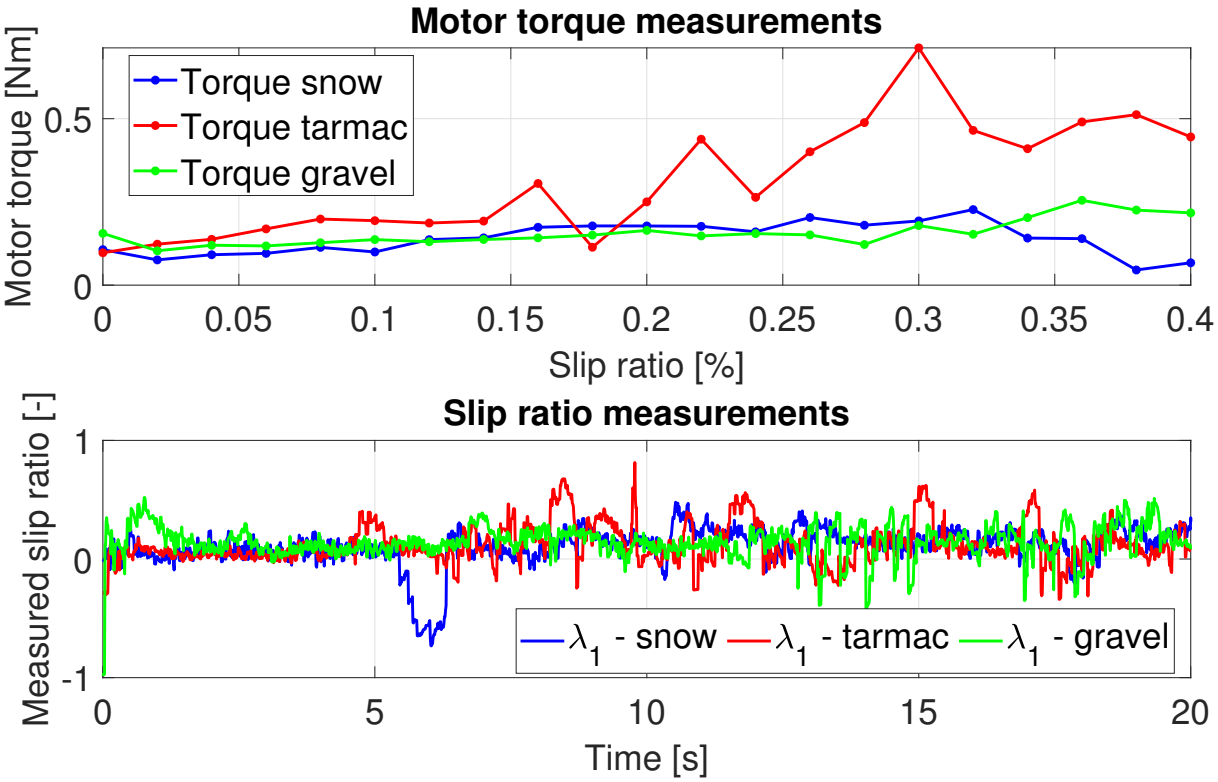


Figure 3.14: Comparison of traction motor torque and slip ratio for the first driven wheel. The first subplot shows the torque dependence on the slip ratio. The dots represent the mean measured values for a particular slip ratio. The slip ratio measurements are shown in the second subplot. The experiments were performed using the experimental platform described in Section 3.4.

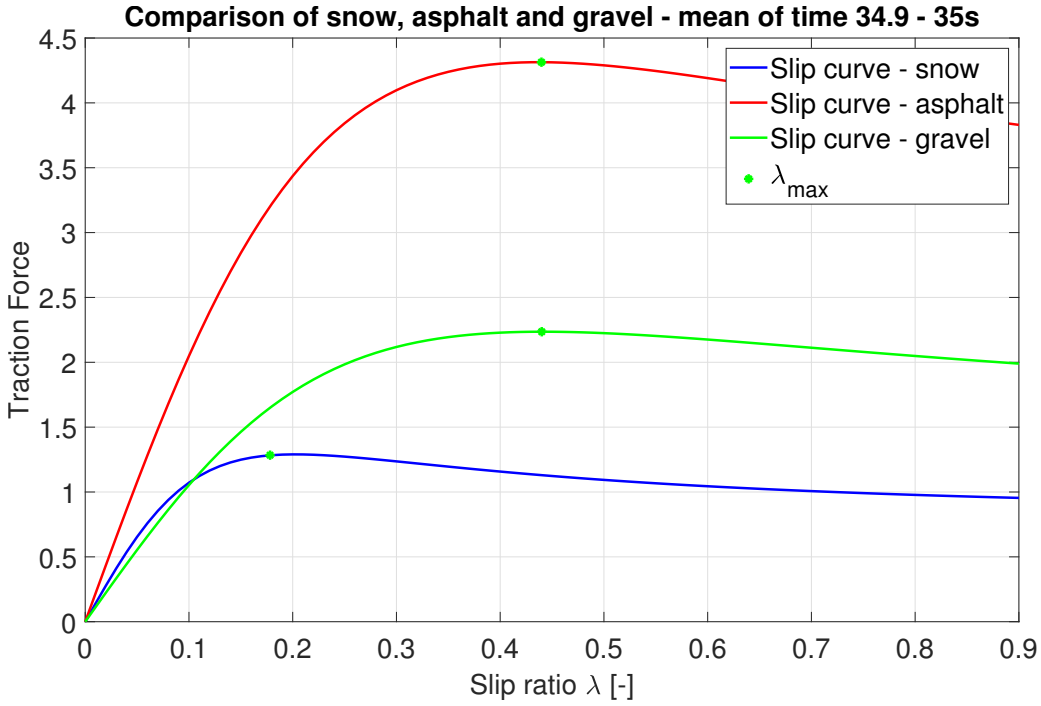


Figure 3.15: Comparison of Pacejka slip curve estimates for all three surfaces. Test rides were performed using the experimental platform described in Section 3.4.

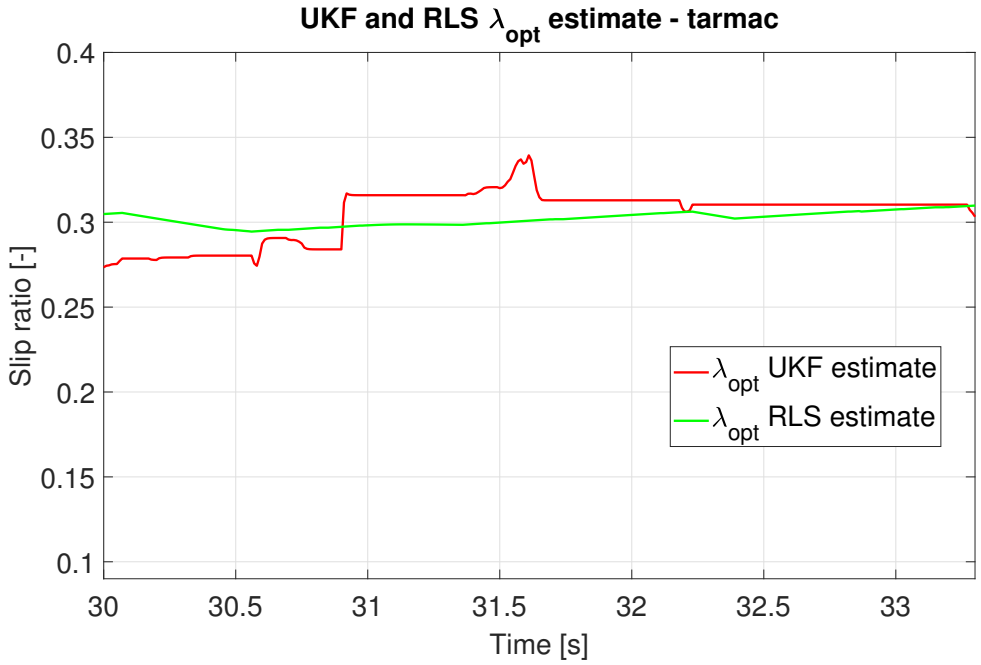


Figure 3.16: Detailed comparison of UKF- and RLS-based λ_{opt} estimates. Experiments were performed using the experimental platform described in Section 3.4.

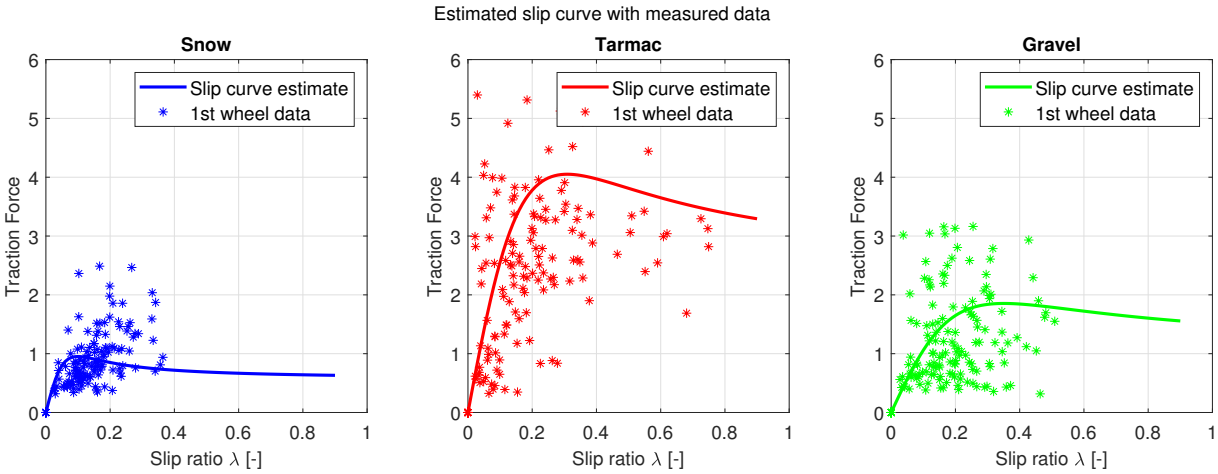


Figure 3.17: Measured data with the corresponding force estimate (see Section 3.3.1) and the UKF estimated slip curve (see Section 3.3.3). Test rides were performed using the experimental platform described in Section 3.4.

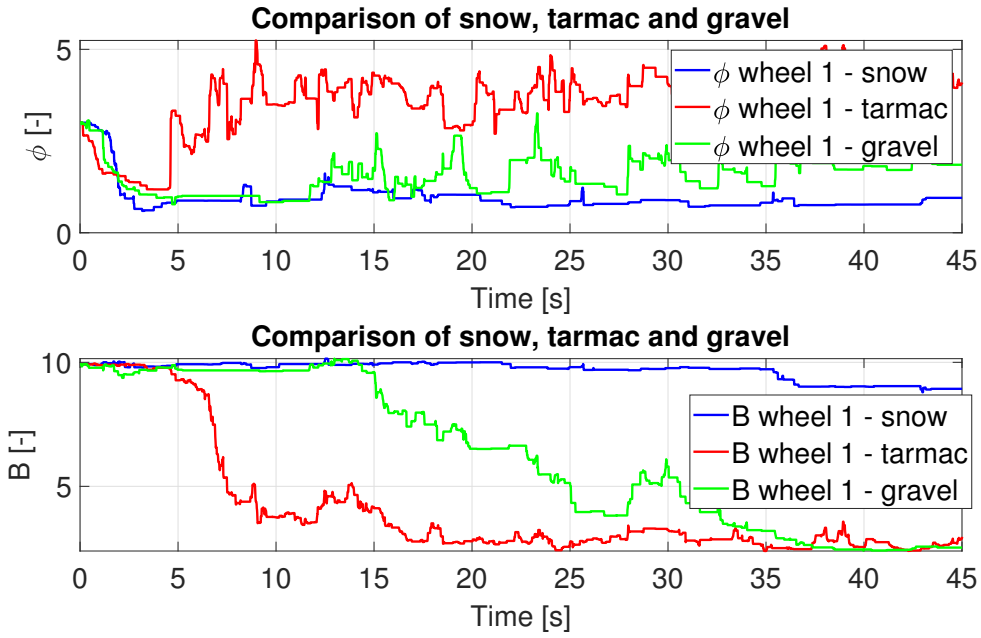


Figure 3.18: Estimated Pacejka slip curve parameters using UKF for different surfaces. Test rides were performed using the experimental platform described in Section 3.4.

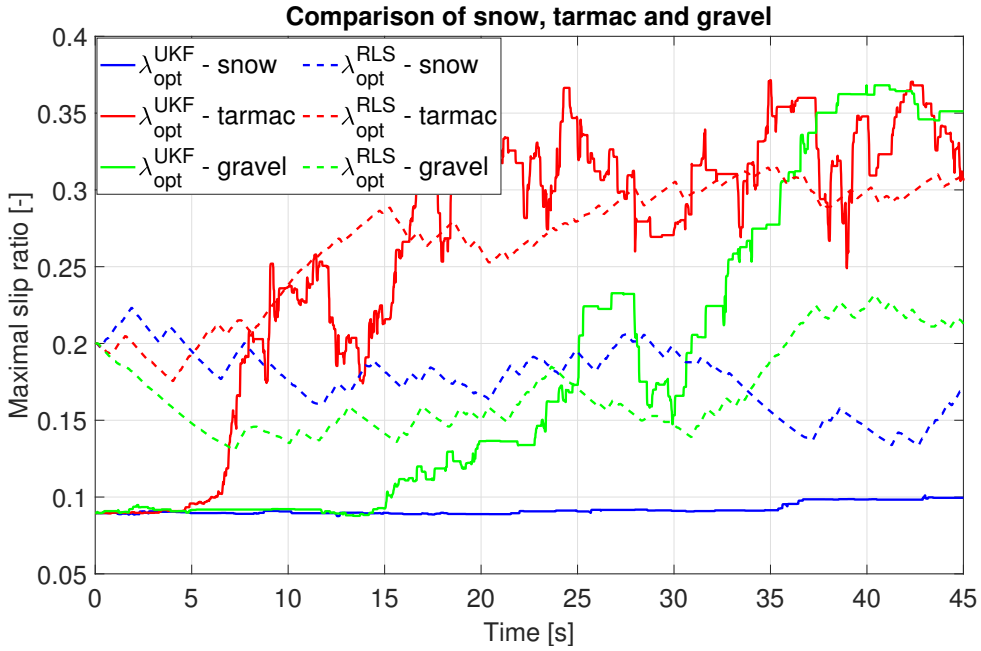


Figure 3.19: Estimated maximum slip ratio using UKF- and RLS-based estimators for different surfaces. Test rides were performed using the experimental platform described in Section 3.4.

is the noise in the slip ratio λ measurements (see Fig. 3.14). The noise disrupts the RLS slip curve slope measurement, affecting the optimal slip ratio λ_{opt} estimation. Due to the relatively slow vehicle speed in all experiments (up to $10m/s$ with mean velocity less than $5m/s$), the velocity measurement noise significantly influences the slip ratio measurement (see Fig. 3.14). Assuming a constant vehicle velocity/wheel speed noise level, the signal-to-noise ratio is high for low traveling speeds, which propagates to high slip ratio noise (see (2.15)). It is expected that the slip ratio measurement noise will be less significant for full-scale vehicle applications, where typically higher traveling speeds are assumed; hence, the estimators' performance is anticipated to improve.

Experiment	Q	R
Real world	diag([0.001, 0.1, 0.1])	diag([1, 1])
Simulation	diag([0.001, 0.09, 0.08])	diag([0.1, 8000])

Table 3.2: UKF-based estimator matrices tuning.

Experiment	c_f	$\hat{\lambda}_{\text{ub}}$	$\hat{\lambda}_{\text{lb}}$	λ_{coast}	φ
Real world	0.999	0.41	0.04	0.03	0.75
Simulation	0.992	0.3	0.04	0.03	0.75

Table 3.3: RLS-based estimator parameters tuning.

3.6 Experimental results of usage in the NN predictor

The UKF-based estimator intrinsically estimates the product of the Pacejka parameter D and the friction coefficient μ_i, represented as $D\mu_i$, or the maximum traction force (interpreted as surface friction) $\phi = F_z^i \mu_i D^i$, as part of its estimation process. For the experimental subscale platform, ϕ is estimated since F_z is neither measured nor estimated on the platform (refer to Section 3.5.3 and Section 3.4 for further details). This estimate serves as the ground truth for training a visual predictor. The visual predictor utilizes a Convolutional Neural Network (CNN) to forecast surface slipperiness ahead of the vehicle. The sections that follow elaborate on the visual predictor, its training, and its evaluation. The methodology for the visual predictor, as outlined here, was published in [18]. The work presented in [18] is multidisciplinary. My specific contribution was preparing the dataset, i.e., collection of the camera data and their labels using the UKF-based estimator used for neural network training and validation. The actual training and validation of the neural network were conducted by the other authors of the paper. Their results are presented in this Section 3.6 for the sake of completeness.

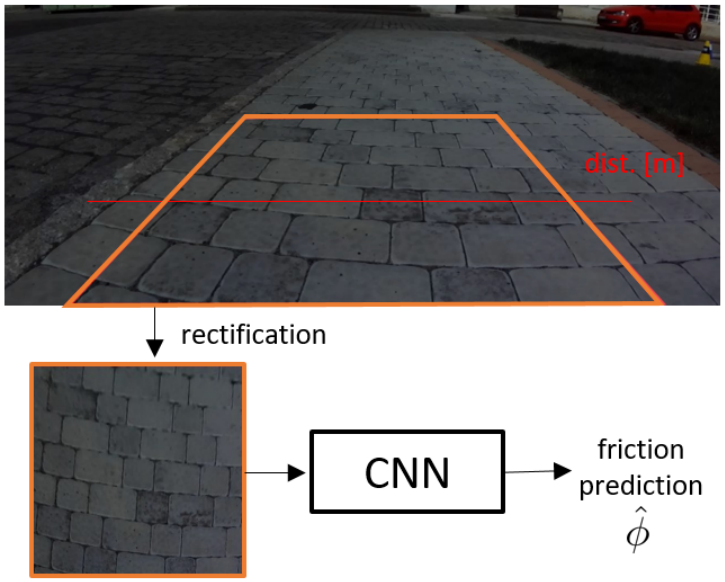


Figure 3.20: Visual predictor overview. An input image from the vehicle camera is rectified and cropped to capture a 1.5m × 1.5m rectangular area in front of the vehicle (highlighted in yellow). The image is then input to a CNN to predict friction at a measurement distance of d, set to 0.75m (marked by the red horizontal line).

3.6.1 Visual Predictor Training

The method described in [18] focuses on training a visual-only predictor using a CNN. This network processes images captured by a front-facing camera to estimate the corresponding surface friction ϕ. To train the CNN, data is collected by driving and recording both the camera feed and the vehicle's response signals. The training images are automatically labeled by correlating them with the surface friction estimates ϕ obtained using the method outlined in Section 3.3.

3.6.2 Image Labeling

An overview of the visual predictor process is illustrated in Fig. 3.20. Initially, a raw image from a camera is orthographically rectified to a bird's-eye view. Homography mapping transforms the raw image of a scene rectangle on the ground plane in front of the vehicle into a rectified image, ensuring the alignment of the rectangle's corners with the rectified image corners. This rectified image is then input into a CNN.

For training the CNN, images need appropriate labeling. To label an image captured at time t, the corresponding surface friction $\phi(t)$ is identified. A fixed measurement distance d in the middle of the scene rectangle is selected for all experiments. The time t when the vehicle travels over the measurement distance d is calculated using the vehicle velocity as $t = d/v(t)$, assuming a straight vehicle trajectory.

The optimal size of the input image for the CNN was determined experimentally. A

smaller region around the measurement distance provides insufficient context, while a larger region captures irrelevant background details.

Rectification of images, while not essential, serves as an efficient way to normalize images and potentially integrate different cameras with varying viewpoints without necessitating retraining of the CNN. In all experiments, the homography is estimated offline. Deviations from the estimated mapping, due to camera tilt from mounting on the vehicle body, are generally ignored since it does not introduce significant errors to the training process.

3.6.3 Convolutional Neural Network

Following the aforementioned process, a labeled dataset $\{(I_1, \phi_1), \ldots, (I_n, \phi_n)\}$ was compiled.

RESNET-50 [72] was employed as the backbone network. The input is the rectified RGB image I resized to a 224×224 px receptive field. The network outputs a single scalar, after applying ReLU, representing the estimated surface friction $\mathrm{CNN}(I; \Theta) = \hat{\phi}$. The vector Θ encompasses all the network weights.

An L_2-regression loss function $L(\Theta) = \sum \left(\phi_i - \hat{\phi}_i\right)^2$ was used to penalize differences between the ground truth and predicted surface friction labels. Batch-normalization layers were included, and network weights Θ were optimized using the ADAM optimizer [73]. Data augmentation techniques such as color jitter (adjustments in contrast, brightness, and hue) and random horizontal flipping were employed. Small image rotations of $\pm 4^\circ$ were also applied, considering the importance of texture orientation for perceiving surface friction. The training process converged after approximately 50 epochs, demonstrating the network's ability to learn the mapping from images to surface friction estimates effectively.

3.6.4 Experimental Platform Estimation Results

The estimation algorithm was experimentally validated using the platform described in Section 3.4. The sampling time for the estimator matches the platform's sampling rate of 10 ms.

Since lateral velocity is not measured or estimated on this platform, only straight driving maneuvers were considered in the UKF's measurement update phase, excluding the wheel side slip angle. This simplification aids in measuring the wheel pivot point velocity in the x direction, which is equated to the velocity measured by the nondriven wheels (as detailed in Section 3.4).

The estimation algorithm was executed offline following the rides, but it is designed for real-time application as follows from the nature of UKF filtering. Results from three

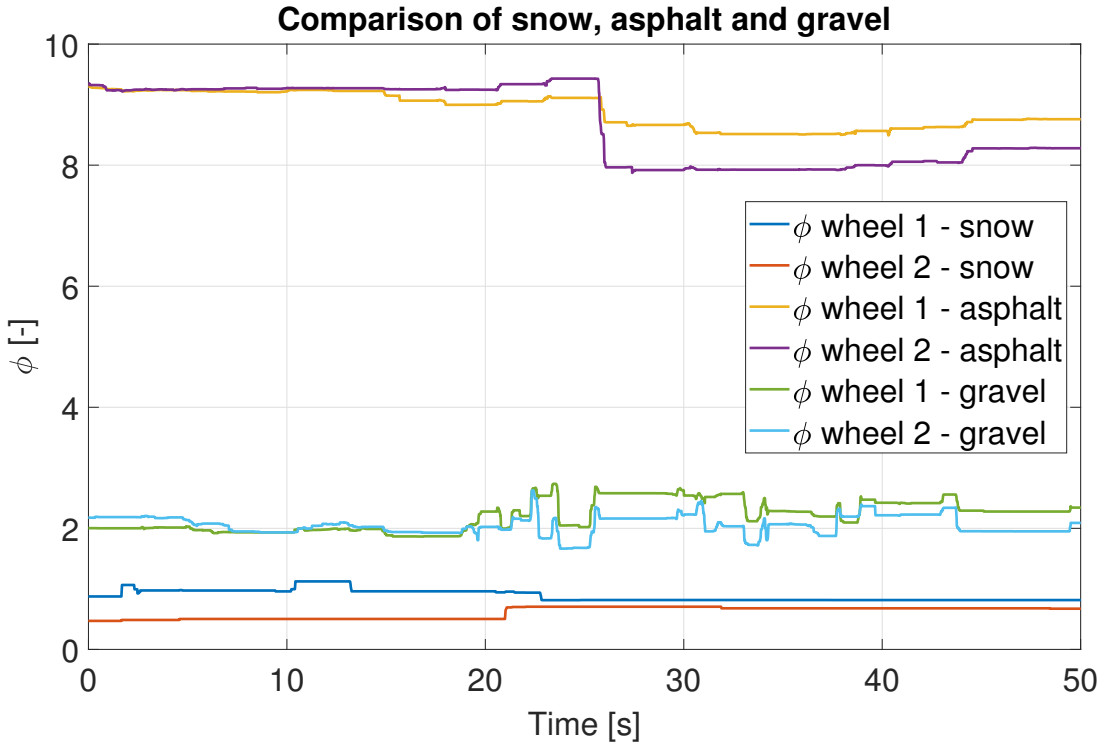


Figure 3.21: Comparison of estimated friction for different surfaces using subscale platform data.

experiments on different surfaces are presented in Fig. 3.21. The tested surfaces include snow, gravel, and asphalt, with asphalt exhibiting the highest adhesion among the surfaces tested. For each surface, two estimated values of $\hat{x}_3$ (for two driven wheels) are shown in Fig. 3.21. The $\hat{x}_3$ corresponds to the friction ϕ. The maximum of the friction estimates from both wheels was used as the surface label for training the neural network. The segmentation of friction estimates for different surfaces is clearly visible in Fig. 3.21.

The estimation algorithm requires sufficient excitation of the vehicle dynamics to provide accurate estimates, as discussed in Section 3.3. To ensure reliable neural network friction estimates that are unaffected by initial value convergence and poor dynamics excitation, two runs of the UKF algorithm were executed. The second run initializes with the average value from the last 30 seconds of the first run. This approach enhances results for surfaces with piecewise constant parameters.

3.6.5 Dataset

Data was collected by driving on various surfaces, including acceleration and deceleration phases, with the slip ratio typically ranging between [-0.4, 0.4]. During extreme maneuvers, the slip ratio reached values up to 0.9. Approximately 2.5 hours of synchronous raw recordings from all vehicle sensors and control signals, including the camera, were acquired. Test rides were conducted at 5 different locations, each featuring a variety of

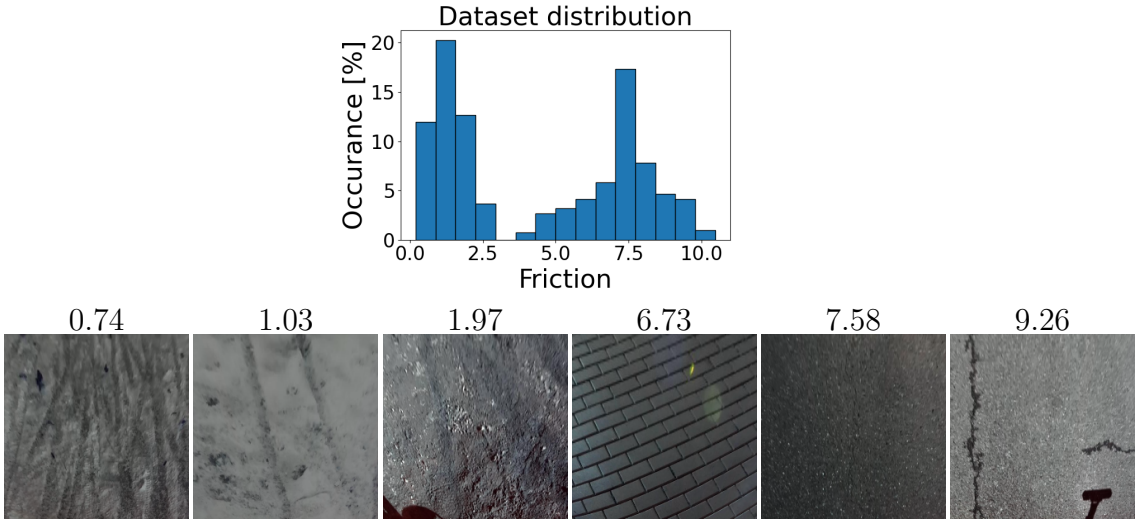


Figure 3.22: Distribution of surface friction ϕ in the acquired datasets. Samples sorted from the lowest to highest friction.

MAE	RMSE	Corr	P_{95}	$e_{0.5}$
0.36755	0.56746	0.9824	1.34325	78.17%

Table 3.4: Error statistics

surfaces such as dry/wet tarmac, snow, pavement, and gravel. Recordings were made on different days and times to capture the effects of varying illumination conditions.

Both the friction estimation method for the subscale platform and the image labeling process assume a straight vehicle trajectory. Consequently, data with significant deviations from a straight path were excluded during post-processing.

Images were captured every 0.5 seconds to maintain a manageable dataset size with sufficient diversity. The resulting dataset, which was automatically labeled, includes about 4,000 samples. This dataset was divided into disjoint subsets: 70% for training, 10% for validation (to select the best training epoch), and 20% for testing. Location and temporal metadata ensured no overlap between the training and test sets.

A sample of the dataset and the distribution of ground-truth friction labels is shown in Fig. 3.22.

3.6.6 Evaluation and Results

The testing was conducted on an independent test split of the dataset. The results are summarized in Table 3.4. The statistics include Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Correlation coefficient (Corr), Percentile-95 (P_{95}) indicating the prediction error lower or equal for 95% of test samples, and Error-0.5 ($e_{0.5}$) showing the percentage of samples with a prediction error lower than or equal to 0.5.

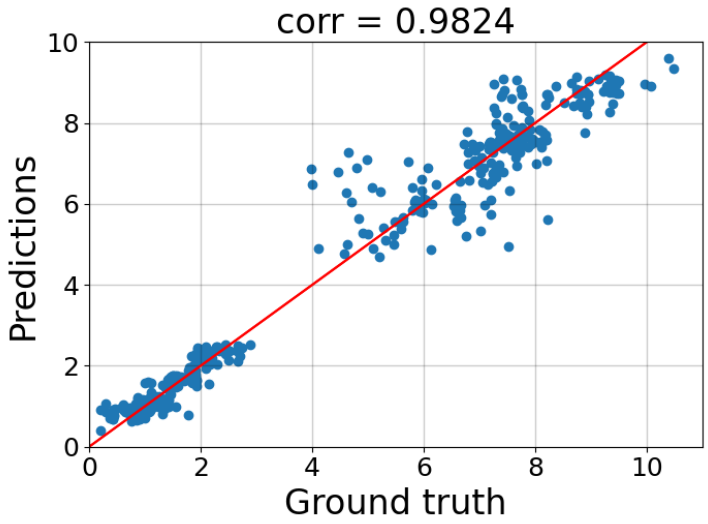


Figure 3.23: Scatter plots for the proposed CNN visual prediction. Ideal predictions lie on the red diagonal line. Correlation coefficients are shown in the title of the plots.

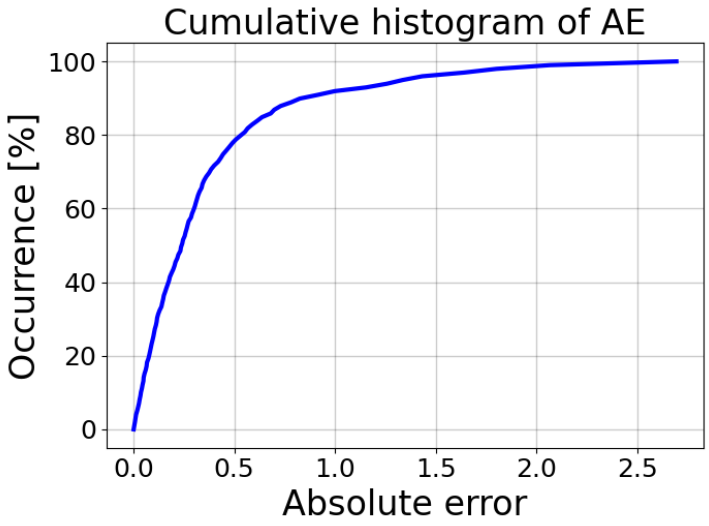


Figure 3.24: Cumulative histograms of absolute error.

In addition to the statistics, a scatter plot is presented in Fig. 3.23, visualizing the correlation between the ground truth and the predicted surface friction labels. Fig. 3.24 shows the cumulative histograms of absolute errors, providing insight into the error distributions. Statistics P_{95} and $e_{0.5}$ are easily seen in the plots. All statistics corroborate the high accuracy of the visual prediction.

3.6.7 Friction Maps – Qualitative Results

The following experiment demonstrates that the trained model generalizes to a slightly different problem: predicting a coarse map of surface friction.

The proposed CNN model was trained to predict the local surface friction in the center

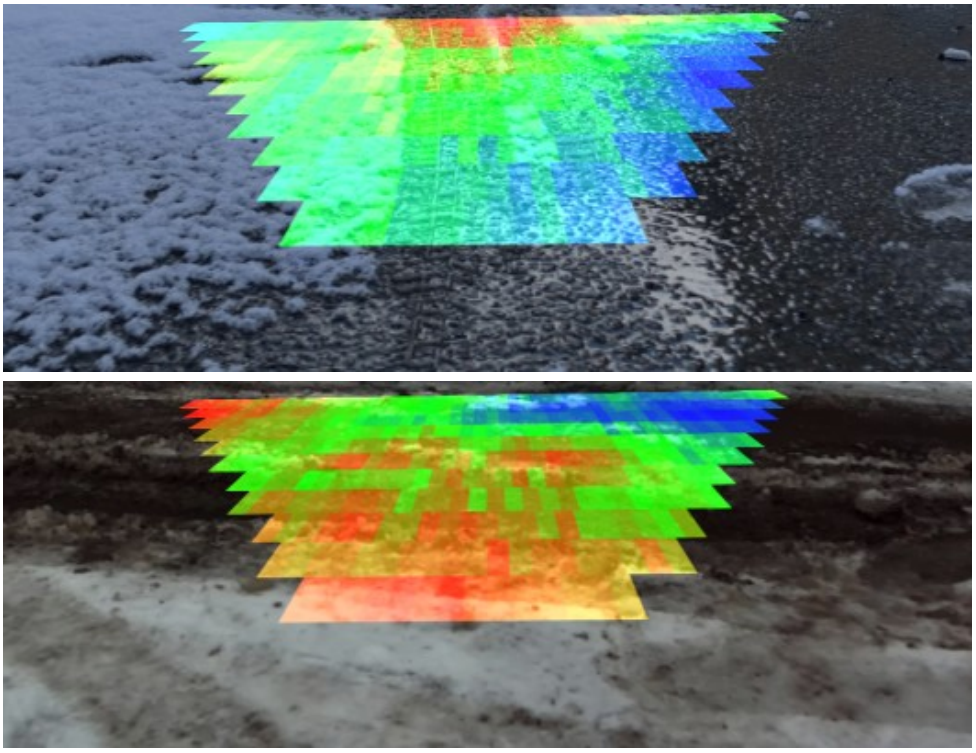


Figure 3.25: Color-coded surface friction maps calculated by the trained CNN executed in a scanning window over the input image rectified to bird's-eye view, colorized by the estimated friction, and finally back-projected to the raw camera image.

of the given image, i.e., a single scalar for an image. The input to the CNN represents a square of 1.5m $\times$ 1.5m in the scene. Therefore, a larger area of the surface seen by the camera is scanned with the CNN evaluated at many locations. Specifically, the scanning is done in the rectified bird's-eye view, with partially overlapping images of 1.5m $\times$ 1.5m windows fed to the CNN one by one. The outputs are stored, the center window locations are colorized based on the surface friction, and finally back-projected to the raw camera image.

Results are shown in Fig. 3.25. The boundary between surfaces of different frictions is visible. This experiment is presented as a qualitative result since ground-truth data for other locations than the measurement point (0.75m in front of the vehicle for an image) are not available. The CNN was not trained for other locations, making it surprising that it generalizes well for distant locations (up to 5 meters), where loss of resolution degrades image quality. This experiment demonstrates the accuracy and spatial consistency of the predictor.

3.7 Summary

This chapter presents two novel approaches for estimating the optimal slip ratio λ_{opt}. Both methods aim to solve the estimation problem of λ_{opt} efficiently. The estimators were rigorously tested in simulations using a high-fidelity nonlinear twin-track model and subsequently validated in real-world experiments across three different surface types using a subscale validation platform.

The RLS-based estimator is noted for its simplicity and ease of implementation, making it relatively straightforward to tune. In contrast, the UKF-based algorithm is more sophisticated and complex, offering a higher level of advancement. Simulation results demonstrated that both algorithms can estimate the optimal slip ratio λ_{opt} with remarkable accuracy. The UKF-based estimator, however, exhibited a faster convergence rate and higher precision in terms of estimation error. Despite this, the RLS algorithm proved to be more robust regarding individual tuning parameters, as observed during the author's tuning experiences.

Real-world experiments corroborated the high accuracy of both algorithms in estimating the optimal slip ratio λ_{opt}.

Additionally, the UKF-based estimator naturally provides an estimate of surface friction ϕ, which was utilized to predict surface slipperiness ahead of the vehicle using camera images and ResNet-based CNN [18].

The expectation is that both the RLS- and UKF-based estimators will perform even better in full-scale vehicle applications. This anticipation is based on the availability of more accurate sensors in full-scale vehicles, coupled with the fact that slip ratio λ measurement noise is less significant at higher speeds. At higher velocities, the absolute velocity measurement noise becomes less significant, improving the overall accuracy.

Given the easier implementation and robustness of the RLS-based estimator, its application to full-scale vehicles might be more straightforward. However, the UKF-based estimator's superior accuracy makes it more suitable for applications requiring higher precision in full-scale vehicles. Both algorithms demonstrate strong potential for deployment in real-time estimation of the optimal slip ratio λ_{opt} in full-scale vehicle applications. This λ_{opt} estimation can be used to initialize or otherwise inform traction control systems, thereby enhancing vehicle dynamics control performance.

It is important to note that both estimators are focused on longitudinal dynamics. Therefore, they may exhibit reduced accuracy in the presence of lateral slip ($\alpha_i \neq 0$). Addressing this systematic inaccuracy remains a challenge for future research.

Overall, the proposed estimation algorithms represent significant advancements in vehicle dynamics control, with practical applications in real-world scenarios. The successful implementation and validation of these algorithms pave the way for improved traction

control and vehicle stability, contributing to safer and more efficient vehicle operations. Future work will focus on refining these estimators to handle lateral dynamics and further enhance their robustness and accuracy in various driving conditions.

Chapter 4

Vehicle Trajectory Planning

Developing a fast online state trajectory (or path) planning algorithm for autonomous or self-driving vehicles is a well-recognized challenge. Self-driving vehicles are complex systems that encompass numerous subtasks such as environment sensing, prediction, safety features, and more.

Based on the sensed environment and the vehicle's target state, the autonomous vehicle makes decisions and executes them. Decision-making is a critical component of vehicle autonomy, implemented through a trajectory/path planning algorithm embedded within the navigation system's middleware and vehicle dynamics controller. The primary objective of this decision-making module is to plan a safe, collision-free trajectory towards the destination, considering factors like vehicle dynamics, static and dynamic obstacles, road boundaries, and numerous other constraints. Given the variability of real-world scenarios, the trajectory planning problem is inherently complex and often demands substantial computational and memory resources.

Trajectory/path planning for vehicles can typically be divided into several stages. The entire trip path planning, which considers the longest planning horizon (e.g., tens of kilometers), primarily falls within the domain of navigation planning and is not covered in this chapter. The output of such high-level planning might be a list of streets or general directions. This chapter focuses on state trajectory planning for shorter horizons (a few hundreds of meters at most), which directly feeds into the trajectory tracking controller. It is important to note that trajectory tracking is outside the scope of this chapter.

Most traditional planning methods assume a binary feasibility for the computed trajectory: feasible or not feasible (e.g., encountering an obstacle on the path). However, complex rules for determining trajectory feasibility can sometimes lead to contradictions, resulting in no feasible trajectory being planned. This limitation is addressed by the Minimum Violation Planning (MVP) framework [74], which introduces a continuous measure of trajectory feasibility with respect to safety, economics, and other constraints. This

makes MVP particularly interesting for this research, as it handles constraints (other than model dynamics) as logical statements that are transformed into cost functions with varying priorities.

The MVP framework guarantees a planned trajectory for every input scenario. For instance, it can provide a solution for scenarios where traversing a restricted area (e.g., a grassy patch instead of a road) is necessary to reach the goal. Traditional planning algorithms often fail in such scenarios. An example discussed in [75] involves an autonomous vehicle needing to overtake a stationary vehicle to reach its destination, even though overtaking is prohibited in that area. MVP always yields a trajectory, unlike many conventional algorithms like Rapidly-exploring Random Tree (RRT) and its variants.

Numerous studies have proposed fast planning algorithms, such as motion primitives in RRT [76, 77]. However, these methods lack MVP's features, such as multi-level cost function optimization and logical constraint handling, as provided by MVP [75].

The second algorithm examined in this chapter is Model Predictive Control (MPC), part of the optimization-based state trajectory planning family. Originally developed for process control, MPC has recently been applied to planning tasks [78, 79]. MPC is designed for convex minimization problems, which is a restrictive assumption for self-driving car trajectory planning, where many constraints create a non-convex space, such as navigating around obstacles.

This chapter first presents modifications and comparisons of Model Predictive Control (an optimization-based approach) and Minimum Violation Planning (a sampling-based approach) to address dynamic state trajectory planning. Similar ideas have been proposed in [80, 19].

Next, an integration of fast steering algorithms and the MVP framework is discussed, as proposed in [20]. A comparison between the original RRT* used in the MVP framework and the modified approach suggested in [20] is provided. This integration is crucial because traditional algorithms mentioned in [75] are inadequate for more complex dynamic models that consider additional factors beyond vehicle speed.

The MVP framework and the proposed modifications are evaluated using a selected benchmark scenario to plan a vehicle state trajectory encompassing vehicle velocity, north/east position, yaw, yaw rate, and battery state of charge.

MVP relies on sampling-based methods and can utilize various algorithms such as RRT#, RRG, RRT*, and others [75]. These methods provide performance benefits from the planned trajectory's optimality perspective, with all mentioned methods being asymptotically optimal. However, a fast steering function is crucial for real-time applications, especially when complex model dynamics are considered, necessitating dynamic state connection and state trajectory computation (i.e., solving a boundary value problem).

Minimum Violation Planning, introduced in [74] and extended in [75], serves as the foundation for this chapter's further study of MVP.

This chapter including the figures is based on my previously published work in [19, 20].

This chapter is organized as follows: First, the related work is reviewed. Next, the simulation environment used for validating the trajectory planning algorithms is introduced. Following this, the problem statement for minimum violation planning and model predictive control trajectory planning is presented. Subsequently, simulation results are compared and discussed. Finally, modifications to MVP that enhance calculation time without significantly compromising performance are presented and evaluated in a selected test scenario.

4.1 Related work

Path planning algorithms have their roots in the field of mobile robotics. This task is generally computationally intensive and memory demanding, necessitating simplifications in the mathematical models used. In robotics, vehicle dynamics are often considered less critical, leading to plans that are typically constrained to kinematic models (see [81, 82, 83]). Over time, the trajectory planning community has incorporated full trajectory planning, including differential equation constraints, to better suit autonomous driving applications.

Trajectory planning algorithms can be broadly classified into three main categories: sampling-based methods, search-based methods, and optimization-based methods. Below is a brief overview of each category.

Sampling-based Methods

Sampling-based methods can be further divided into probabilistic and deterministic sampling. Probabilistic methods include probabilistic roadmaps [84] and rapidly exploring random trees (RRT) [85, 86, 87]. In [85], the authors introduced a closed-loop controller for the vehicle model, allowing only the controller reference to be planned while simulating the entire vehicle state trajectory. [86] proposed different biased sampling strategies for generating the tree. A fast local steering algorithm based on half-car dynamics and RRT* was introduced in [87], although this strategy compromises the optimality of the resulting trajectory. In [88], the authors proposed RRT*, a variation of the RRT algorithm that asymptotically converges to the optimal solution.

Minimum Violation Planning (MVP) was introduced in [74] and relies on sampling-based methods. MVP employs RRT, RRT*, and RRG methods as mentioned in [75]. For more comprehensive details on sampling-based methods, readers are referred to [89].

Deterministic sampling-based methods are exemplified by control space sampling [90] and state-space sampling [91, 92]. [90] utilized discrete control inputs over a specific time to solve the initial value problem and compute the state. [91, 92] chose a goal state from the reference path and solved boundary value problems using predefined curves (polynomials) to derive a trajectory.

Optimization-based Methods

Optimization methods typically address trajectory planning problems by solving nonlinear two-point boundary value problems using optimization techniques [93, 94, 95, 96]. These methods often parameterize the trajectory/path using specific curve types, such as spirals [94], polynomials [95, 96], or piecewise constant functions [97]. Optimization methods are prevalent in control problems due to their fast convergence to local optima and high-quality solutions. However, they are most effective with convex problems, as local solvers are typically employed. Consequently, optimization can get stuck in local optima when dealing with more complex non-convex problems.

Obstacle handling, in particular, leads to non-convex problems, necessitating heuristics or good initial guesses to initialize the optimization process close to the global optimum, e.g., [98] proposed cost-to-go heuristics to address this issue. Model Predictive Control (MPC), studied in this chapter, falls under optimization-based methods.

Search-based Methods

Search-based methods utilize graph search algorithms such as A* [99] or Dijkstra [100]. These methods typically sample the configuration space uniformly and construct a directed graph, which is then searched for a solution. While these methods perform well for smaller problems, they suffer from the curse of dimensionality as the problem size increases.

4.2 High-Fidelity Nonlinear Single-track Model and Test Scenario

In this section, a high-fidelity nonlinear validation model and a specific test scenario for validating both Model Predictive Control (MPC) and Minimum Violation Planning (MVP) trajectory planning algorithms are introduced.

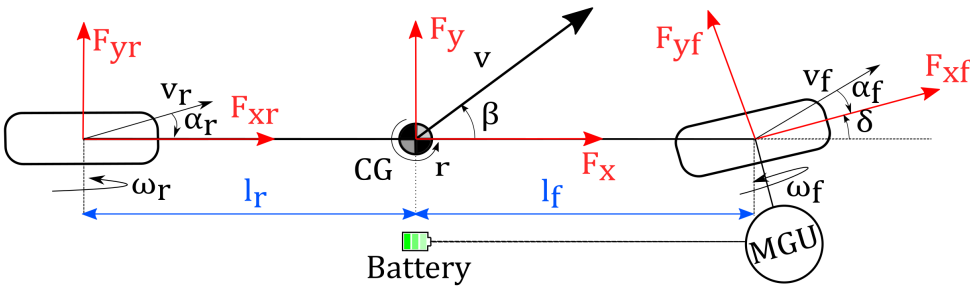


Figure 4.1: High-fidelity single-track model.

4.2.1 Nonlinear High-Fidelity Single-Track Model

The single-track model functions as the nonlinear mathematical framework essential for validating the proposed trajectory planning algorithms. This model, implemented in Matlab & Simulink and illustrated in Fig. 4.1, comprises four principal components:

- Rigid body
- Battery
- Electric Motor Generator Unit (MGU)
- Wheel

Rigid Body

The dynamics of the rigid body are represented by the following differential equations (refer to [2]):

$$\begin{pmatrix} \dot{\beta} \\ \dot{v} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{1}{mv} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{I} \end{pmatrix} \begin{pmatrix} -\sin\beta & \cos\beta & 0 \\ \cos\beta & \sin\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} \cos\delta & -\sin\delta & 1 & 0 \\ \sin\delta & \cos\delta & 0 & 1 \\ l_f \sin\delta & l_f \cos\delta & 0 & -l_r \end{pmatrix} \begin{pmatrix} F_{xf} \\ F_{yf} \\ F_{xr} \\ F_{yr} \end{pmatrix} - \begin{pmatrix} r \\ 0 \\ 0 \end{pmatrix}, \tag{4.1}$$

where β [−] denotes the vehicle sideslip angle, m [kg] represents the vehicle mass, v [m/s] is the vehicle velocity, I [kg/m^2] is the vehicle's moment of inertia, r [rad/s] denotes the vehicle yaw rate, and δ [rad] is the steering wheel angle. The variables l_f [m] and l_r [m] denote the distances from the center of gravity to the front and rear wheels, respectively,

while F_{ij} [N] represents the traction forces, with $i = x, y$ indicating the direction and $j = f, r$ specifying whether the force acts on the front or rear wheels.

The traction force F_{ij} incorporates the traction force generated by the wheel, alongside all drag and resistance effects, such as wheel friction, air drag, road grade, and rolling resistance. The yaw angle ψ [rad], along with the vehicle's east and north positions e [m] and n [m], are modeled as follows:

$$\dot{\psi} = r, \tag{4.2}$$

$$\dot{e} = v\cos(\psi), \tag{4.3}$$

$$\dot{n} = v\sin(\psi). \tag{4.4}$$

Wheel

The wheel model includes both wheel dynamics and the tire-to-road interface, described by the Pacejka magic formula (refer to [28]) with the implementation of the friction ellipse (Kamm's circle) adopted from [34] (refer to Section 2.1.4 for more details). The wheel dynamics are governed by:

$$J\dot{\omega} = \tau + \tau_b - \tau_{res} - F_x r_{whl}, \tag{4.5}$$

where ω [rad/s] denotes the wheel angular speed, τ [Nm] represents the e-motor drive torque, τ_b [Nm] indicates the mechanical brake torque, τ_{res} [Nm] encompasses all resistive effects (friction, rolling resistance, etc.), J [kg/m^2] signifies the wheel's moment of inertia, F_x [N] represents the traction force calculated via the Pacejka magic formula and friction ellipse, and r_{whl} [m] denotes the wheel's effective radius.

Battery

The battery is modeled as a 2 RC model (refer to [101]). Thermal dynamics are neglected under the assumption of a battery thermal management system. The battery model equations are:

$$\dot{v}_1 = -\frac{1}{R_1 C_1} v_1 + \frac{1}{C_1} i, \tag{4.6}$$

$$\dot{v}_2 = -\frac{1}{R_2 C_2} v_2 + \frac{1}{C_2} i, \tag{4.7}$$

$$\dot{SOC} = -\frac{100}{3600C} i, \tag{4.8}$$

$$i = \frac{P_{trm}}{V_{trm}}, \tag{4.9}$$

$$V_{trm} = V_{oc}(SOC) - v_1 - v_2 - R_0(SOC)\, i, \tag{4.10}$$

where i [A] represents the battery current, R_1, R_2 [Ω] and C_1, C_2 [F] are the resistances and capacitances of the RC circuits, respectively. Additionally, v_1, v_2 [V] denote the voltages of the RC circuits, C [Ah] indicates the battery capacity, SOC [%] stands for the state of charge, P_{trm} [W] represents the battery terminal power (input to the battery system), V_{trm} [V] denotes the battery terminal voltage, $V_{oc}(SOC)$ [V] is the open circuit voltage as a function of SOC, and $R_0(SOC)$ [Ω] represents the internal resistance, also as a function of SOC.

Motor Generator Unit

The Motor Generator Unit (MGU) is modeled with dynamics equations, assuming a rigid connection to the wheels, which determine the revolutions fed to the MGU model:

$$\dot{\tau}_{\text{int}} = -\frac{1}{t_\tau}\left(\tau_{\text{int}} - \min\left(\max\left(\tau_{\text{req}}, \tau_{min}\right), \tau_{max}\right)\right), \tag{4.11}$$

$$\dot{\omega}_{\text{em}} = -\frac{1}{t_\omega}\left(\omega_{\text{em}} - \omega_{\text{in}}\right), \tag{4.12}$$

$$\tau = \tau_{\text{int}} r_\omega - J_{em} r_\omega^2 \left(-\frac{1}{t_\omega}\left(\omega_{\text{em}} - \omega_{\text{in}}\right)\right), \tag{4.13}$$

$$P_{\text{em}} = \tau_{\text{int}} \omega_{\text{in}} r_\omega \eta\left(\tau_{\text{int}}, \omega_{\text{in}} r_\omega\right), \tag{4.14}$$

where τ [Nm] is the MGU mechanical torque, τ_{int} [Nm] is the MGU internal torque, t_τ [s] is the torque time constant, τ_{req} [s] is the requested mechanical torque, ω_{in} [rad/s] is the wheel angular speed, ω_{em} [rad/s] is the MGU angular speed, t_ω [s] is the speed time constant, r_ω [1] is the MGU to wheel gear ratio, J_{em} [kg/m^2] is the MGU inertia, P_{em} [W] is the MGU power consumed or regenerated, and $\eta(\tau_{\text{int}}, \omega_{\text{in}})$ [1] is the MGU efficiency map as a function of torque and MGU speed. The MGU speed state is used to filter out the input speed that is assumed to be given by the wheel speed measurement, which might be noisy.

4.2.2 Test Scenario for MVP and MPC Comparison

A scenario is selected to evaluate both the MVP and MPC approaches. This scenario includes the initial state of the vehicle, the desired goal state, and various static and dynamic obstacles. The primary objective is to reach the destination while minimizing the state of charge (SOC) consumption and travel time, without colliding with any obstacles.

The scenario is illustrated in Fig. 4.2, showcasing both static obstacles (such as buildings) and dynamic obstacles (like another moving vehicle).

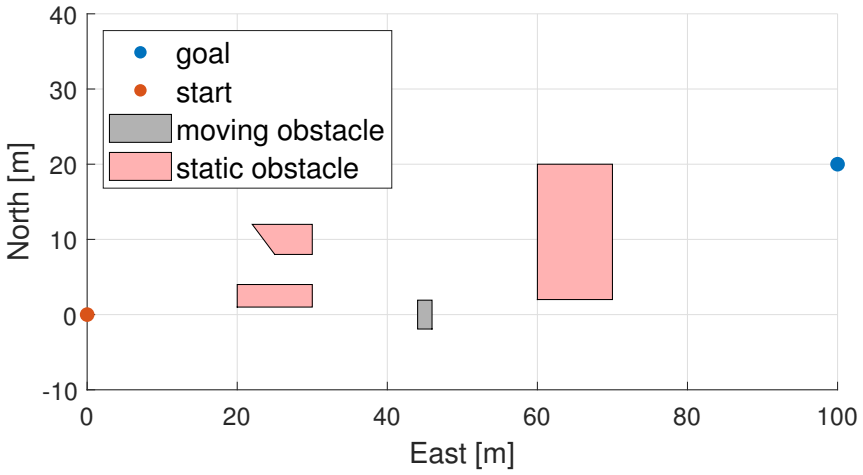


Figure 4.2: Validation scenario featuring static and dynamic obstacles.

4.3 Model used for planning

To streamline the computational demands for both MVP and MPC, the high-fidelity single-track model detailed in Section 4.2.1 is simplified.

The dynamics of the Motor Generator Unit (MGU) are disregarded, treating the MGU model as static, represented by:

$$\tau_{\text{int}} = \min\left(\max\left(\tau_{\text{req}}, \tau_{min}\right), \tau_{max}\right), \tag{4.15}$$

$$\tau = \tau_{\text{int}} r_{\omega}, \tag{4.16}$$

$$P_{\text{em}} = \tau_{\text{int}} \omega_{\text{in}} r_{\omega} \eta\left(\tau_{\text{int}}, \omega_{\text{in}} r_{\omega}\right). \tag{4.17}$$

The battery dynamics are reduced to a single state variable:

$$\dot{SOC} = -\frac{100}{3600C} i, \tag{4.18}$$

$$i = \frac{P_{trm}}{V_{trm}}, \tag{4.19}$$

$$V_{trm} = V_{oc}\left(SOC\right) - R_0\left(SOC\right) i, \tag{4.20}$$

Both the lateral and longitudinal directions of the Pacejka magic formula are neglected, simplifying the traction force calculation to:

$$F_x^{whl} = \min\left(F_{\max}, \frac{\tau + \tau_b}{r_{whl}}\right), \tag{4.21}$$

where F_x^{whl} [N] is the traction force generated by the wheel and $F_{\max}$ [N] represents the wheel's maximum traction force, incorporating saturation due to the tire-to-road interface.

The vehicle dynamics are also simplified as follows:

$$F_x = F_x^{whl} \cos(\delta) - F_{res}, \tag{4.22}$$

$$r = \frac{1}{k} v\delta, \tag{4.23}$$

$$\dot{v} = \frac{F_x}{m}, \tag{4.24}$$

where k [m] represents the minimum turning radius constant, and F_{res} [N] encapsulates all resistive forces such as air drag, road grade, and various frictions.

The vehicle position is modeled consistently with the high-fidelity model, as given by Equations (4.2), (4.3), and (4.4).

In the simplified planning model, the manipulated variables include the MGU torque τ, mechanical brake torque τ_b, and wheel steering angle δ.

4.4 Minimum violation planning (MVP)

The MVP algorithm, as detailed in [75], provides a structured approach to managing logically defined safety and performance criteria. This framework allows for prioritizing various criteria, such as placing safety above economic considerations. One significant advantage of MVP is its ability to consistently generate a trajectory, which can be asymptotically optimal depending on the steering function employed (see Section 4.7 and [75] for further details). The dynamics model used within the MVP framework for trajectory planning is described in Section 4.3.

Initially, the *steer* function is revisited. The authors in [75] do not provide a specific implementation for this function. The primary requirement is that the *steer* function must generate a trajectory from an initial state x_i to a goal state x_g (the function's arguments). This requirement naturally involves solving a Boundary Value Problem (BVP) due to the nonlinear differential equations describing the model dynamics (see Section 4.3). However, solving BVPs can be complex and computationally intensive. To simplify the calculations, an approach using input sampling instead of state sampling in the MVP algorithm is proposed. Later, in the section Section 4.7 the proposed MVP modifications are detailed and discussed.

MVP modifications

In certain cases, particularly at the start of the algorithm, the randomly sampled state might not have a nearby node to which it can be connected. To address this, a modification is introduced to reduce the unnecessary overhead when the sampled state cannot be connected.

Rather than sampling a state that should be connected, a new control input value is randomly sampled and applied to a random state from the structure of already connected states. This modification helps reduce the number of iterations where the initially sampled state remains unused due to the lack of a nearby node for connection. This significantly reduces computational effort, especially during the initial phase of the algorithm.

Furthermore, only a set of input values (discretized input space) is considered. The states are also discretized in space, allowing precomputation of trajectories for all combinations of discretized input and state values. Instead of solving an Initial Value Problem (IVP – integrating system inputs), a precomputed trajectory is selected from the set of precomputed trajectories. Although these modifications lead to the loss of asymptotic optimality (inherited from RRT*), they substantially reduce computational complexity while experimental results indicate no significant performance degradation (see Section 4.9 for further details).

MVP Problem Statement

The MVP framework accommodates multiple cost functions with different priorities (see [75]). For the validation scenario, the problem is stated as follows. The highest priority cost function for MVP states is defined by an LTL logic statement: the vehicle must avoid colliding with any obstacles.

Additionally, a lower-priority trajectory cost function that penalizes travel time and maximizes the state of charge (SOC) of the battery is defined as:

$$J_{\mathrm{MVP}}\left(SOC, t\right) = -w_{SOC} SOC + w_t t, \tag{4.25}$$

where w [−] are the penalizing weights, and t [s] is the travel time. The MVP cost functions, prioritized by order (highest priority first), are:

- Collision avoidance cost function.
- Time minimization and SOC maximization cost function – $J_{\mathrm{MVP}}\left(SOC, t\right)$.

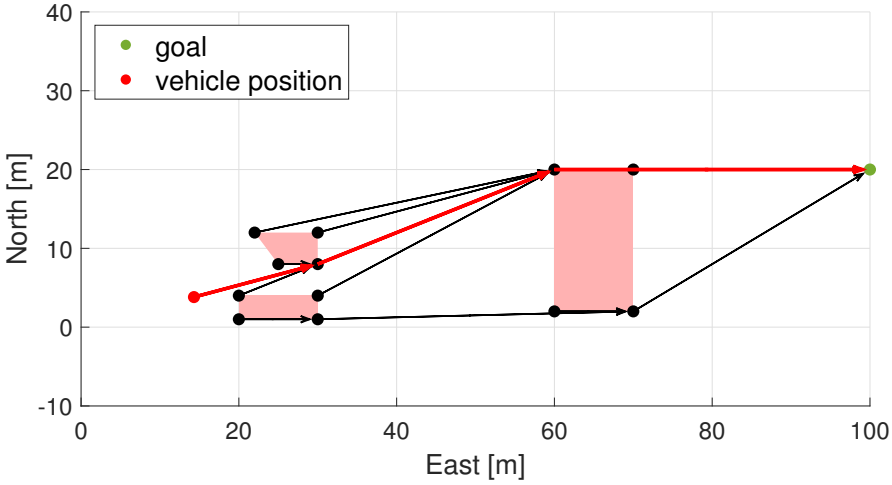


Figure 4.3: MPC cost-to-go heuristics evaluation for the validation scenario.

4.5 Model predictive control (MPC)

MPC is an optimization-based algorithm widely known in the control field [102]. However, most MPC solvers require a convex model, as discussed in Section 4.1. There are various approaches for addressing non-convex problems, with mixed integer programming being notable. Nevertheless, these approaches are computationally intensive and are often restricted to simpler problems.

In the validation scenario, non-convexity was managed using a heuristic approach with the cost-to-go $c_g(x_s, x_g)$ (see [98]), where the Euclidean distance to the goal state was chosen as:

$$c_g(x_s, x_g) = ||x_s - x_g||_2, \tag{4.26}$$

where x_s represents the initial state and x_g the goal state.

The goal state is not always directly visible from the current state, necessitating the construction of a tree that encompasses all possible visible paths, as illustrated in Fig. 4.3. The trajectory with the smallest Euclidean distance is selected, and state constraints are constructed as shown in Fig. 4.4. These constraints are introduced as soft constraints (with slack variables) into the MPC cost function. Given that the system (see Section 4.3) and the resulting optimization problem are nonlinear, a nonlinear MPC is employed.

The cost function is then presented. Unlike in MVP, it is not possible to directly use logically stated constraints in MPC, so the constraints were reformulated to fit the MPC framework. A penalty on the distance to the goal is introduced to minimize arrival time, as direct time penalization is not feasible using a fixed prediction horizon MPC (with a time-domain-based design model). Additionally, penalties for control input increments and SOC state maximization are included. The overall MPC cost function to be minimized

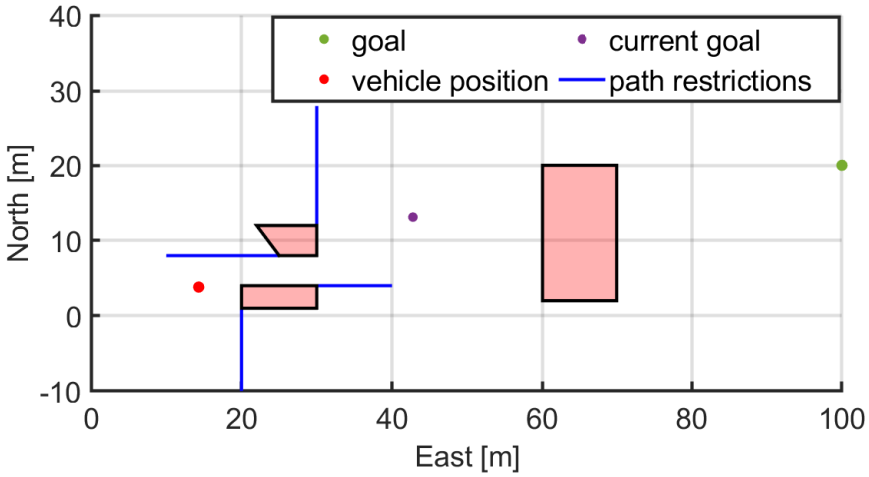


Figure 4.4: MPC constraints calculation based on the cost-to-go heuristics.

is:

$$J_{\text{MPC}} = -w_{SOC} SOC + w_{dst} c_g\left(x_0, x_{cg}\right) + w_{\Delta} \Delta u^2, \tag{4.27}$$

where w are the penalizing weights, x_{cg} is the current goal of the MPC, Δu are the input increments (motor torque, mechanical brake torque), and other symbols are as previously defined. The prediction and control horizon of the MPC is 3 seconds with a sampling time of 0.1 seconds. The current goal x_{cg} is set so that the MPC can reach it within the prediction horizon only when applying full traction torque (see Fig. 4.4). The x_{cg} lies on the path derived from the cost-to-go (see Fig. 4.3).

4.6 Comparison results of MVP and MPC

A comparison of the MPC and MVP approaches was conducted using the simulation scenario outlined in Section 4.2.2. This simulation utilized the high-fidelity nonlinear single-track model detailed in Section 4.2.1 in a closed-loop format. The trajectory was replanned iteratively, with the model being fed the manipulated variables of the planned trajectory, and no trajectory tracking controller was assumed that could enhance trajectory tracking.

Initially, a single step of trajectory planning is illustrated in Fig. 4.5. The MVP planning was halted after 2000 nodes were connected and evaluated, from which the optimal trajectory was selected (red path). The plan was computed for all the states considered (vehicle yaw ψ, yaw rate r, vehicle east and north positions e and n, velocity v, battery state of charge SOC), but for better readability, only the east and north coordinates are displayed to clearly observe the performance of both algorithms.

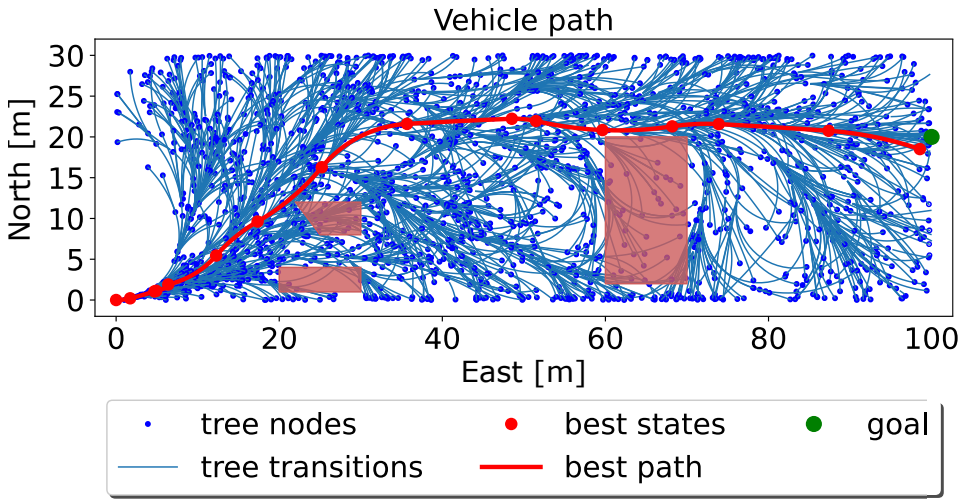


Figure 4.5: Planned trajectory using the MVP approach for the entire scenario. The moving obstacle is omitted for clarity.

Subsequently, the results of the closed-loop simulation are compared. The comparison of the east and north positions is shown in Fig. 4.6, where both planning algorithms reached approximately the same final position. The comparison of other selected inputs and states is displayed in Fig. 4.7. The MVP approach provides a slightly better final SOC, while the arrival time is roughly 1 second longer compared to the MPC.

Differences between both algorithms are anticipated due to the variations in their cost functions and structures. MPC incorporates a single cost function that must encompass all constraints, potentially leading to conflicting terms, such as collision avoidance and minimization of arrival time. Conversely, the MVP approach addresses this issue by prioritizing safety constraints (e.g., collision avoidance) first. Once safety is ensured, the MVP shifts focus to minimizing arrival time and optimizing SOC.

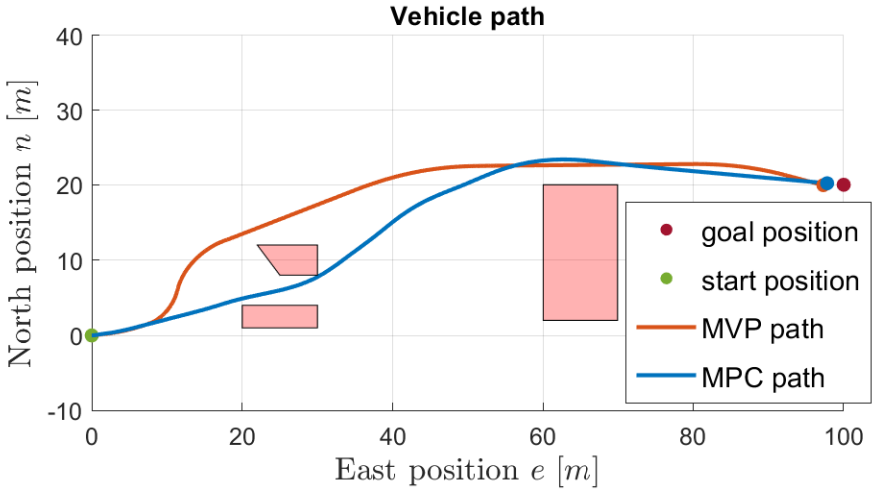


Figure 4.6: Comparison of vehicle paths in feedback loop simulation for both MPC and MVP algorithms in the validation scenario.

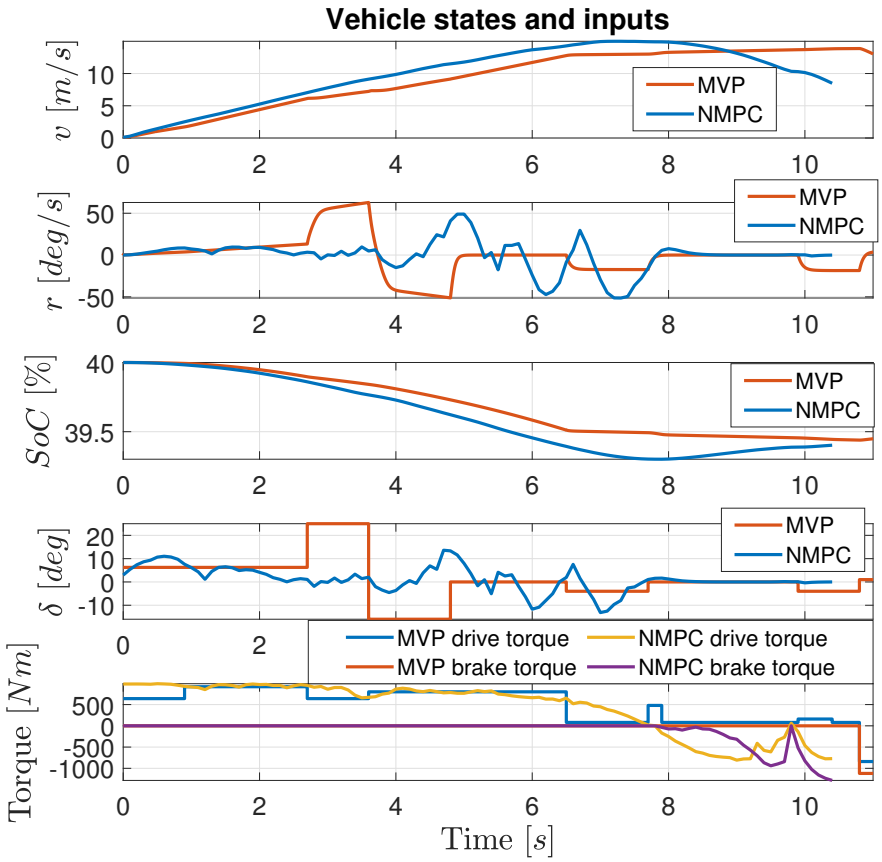


Figure 4.7: Comparison of other selected states and manipulated variables for MPC and MVP algorithms in feedback simulation of the validation scenario.

4.7 Modifications of MVP algorithm

This section will analyze the computational complexity of the MVP algorithm and systematically introduce the modifications mentioned earlier in Section 4.4. MVP is an iterative algorithm that outputs a trajectory typically as a sequence of discrete-time inputs and continuous-time dynamic states (since a continuous-time model is used). The *steer* function implementation can vary, with inputs being either continuous or discrete, as in multiple shooting or single shooting methods.

Trajectories are created by sampling states from the state space, forming a tree with trajectories that define the connections between tree nodes, known as the weighted Kripke structure. The primary requirement for connecting states is to adhere to trajectory dynamics (model equations) and input ranges, without considering other constraints at this stage. The state or trajectory itself does not need to be on a feasible path (e.g., collision-free), but states in constrained areas are penalized heavily. This feature of MVP is advantageous as it provides trajectories for all inputs and can handle scenarios where the trajectory is feasible but not ideal, such as choosing a gravel path where no asphalt one exists.

4.7.1 Overview of the Original MVP

Below is a summary of the original MVP algorithm (refer to Algorithm 1 from [75] for more details).

- Initialize the algorithm.
- Continue looping until a sufficient number of nodes is explored.
  - Sample a random node.
  - Find nodes near the sampled node.
  - Iterate through all near nodes and connect the sampled node with the cheapest available connection.
  - Iterate through all near nodes and reconnect them if a cheaper connection exists from the sampled node.
  - Check if the sampled node is within the goal state set.
  - Add the sampled node to the set of visited states.

The *connect* method from RRT*, which manages edge creation and reconnection, is detailed in Algorithm 2.

Algorithm 2: connect(s, s_{new}) for RRT*

Output: R_K
Input: s, s_{new}, R_K
`/* Check if a cheaper connection exists */`
1 **if** $\text{cost}(s) + \text{cost}(s, s_{new}) < \text{cost}(s_{new})$ **then**
  `/* Remove the more expensive and add the cheaper connection to the set of edges` R_K `*/`
2 $R_K \leftarrow (R_K \backslash \{(s_1, s_2) \in R_K | s_2 = s_{new}\}) \cup \{(s, s_{new})\}$;
  `/* Assign corresponding cost to the` s_{new} `*/`
3 $\text{cost}(s_{new}) = \text{cost}(s) + \text{cost}(s, s_{new})$;

Naive Implementation of the *Steer* Function

In the original MVP algorithm in [75] no specific *steer* function is provided. However, the *steer* function has a significant impact on the algorithm's performance and computational complexity. The *steer* function, as defined in [75], returns a state trajectory $x(t_0, t) \in S$ given states $s, s_{new} \in S$ (where S is the state space set). Its implementation is highly application-specific. Calculating the dynamic system state trajectory, generally described as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \tag{4.28}$$

can be particularly time-consuming, especially when dynamic constraints are considered. Methods that solve the Boundary Value Problem (BVP), such as single or multiple shooting methods, are naturally suggested for connecting two states under dynamic constraints. However, these methods are often computationally intensive and the randomly sampled state might be infeasible to connect given model dynamics and input constraints.

4.7.2 MVP with Input Sampling

A new modification to the MVP algorithm is proposed here, based on the original described in [75]. Algorithms 3 and 5 follow the same nomenclature. The main idea is to eliminate the first iteration loop (lines 7 - 9 in Algorithm 1 of [75]) and replace it with input integration on an already visited state. This modified approach is presented in Algorithm 3.

Instead of sampling a state, the input space U is used to sample a random input u_{new}. Then, a state s' is randomly sampled (as in the original algorithm). Subsequently, the *sampleNear* function samples a node $s \in S_K$, one of the near nodes of s'. A biased sampling is used in *sampleNear*, where near nodes with low cost function values are more likely to be sampled. Finally, the *IVP* function integrates the input u_{new} over a randomly chosen time t using the state s.

Algorithm 3: MVP with Input Sampling – IVP Variant

Output: $\bar{K}_n$, $S_{goal,K}$
Input: s_{init}, S_{goal}, U, $time$, S
$S_K \leftarrow \emptyset$; $R_K \leftarrow \emptyset$; $S_{goal,K} \leftarrow \emptyset$;
add(s_{init}, S_K);
foreach $i \in \mathrm{N}_{\leq n}$ **do**
  $u_{new} \leftarrow$ sample(U);
  $s' \leftarrow$ sample(S);
  $s \leftarrow$ sampleNear(S_K, s');
  $t \leftarrow$ sample$(time)$;
  $s_{new} \leftarrow \mathrm{IVP}(s, u_{new}, t)$;
  connect(s, s_{new}, R_K);
  $S_{near} =$ near(s_{new});
  foreach $s \in S_{near}$ **do**
    if steer$(s_{new}, s) \neq \emptyset$ **then**
      connect(s_{new}, s, R_K);
  if $s_{new} \in S_{goal}$ **then**
    $S_{goal,K} = S_{goal,K} \cup \{s_{new}\}$;
  add(s_{new}, S_K);
return $\bar{K}_n = (S_K, s_{init}, R_K, \Pi, \mathcal{L}, \mathcal{W}_K), S_{goal,K}$

The proposed modification maintains probabilistic completeness, ensuring the entire reachable state space is eventually included as the number of nodes increases. Every sampled state $s \in S_K$ and every input $u \in U$ has a nonzero probability of being selected and integrated over an infinitely small time value. This ensures the entire reachable state space is eventually added to S_K.

This modification reduces unnecessary iterations by replacing them with input space sampling and solving a single IVP (instead of multiple BVPs in the naive *steer* implementation). Another significant benefit is that the original MVP's random new node sample might not connect to the tree of already explored states due to restrictive dynamics and inputs. Additionally, solving IVPs is often less computationally demanding than solving BVPs.

4.7.3 MVP with Precomputed Trajectories

This section presents another modification aimed at further speeding up computation. The primary idea is to eliminate IVP/BVP solving by introducing an offline precomputed trajectory. This precomputation occurs before the actual trajectory planning – a set of motion primitives is precomputed. Thus, the keyword offline is used. The main idea is to precompute trajectories initiating in the state space origin for a discretized set of inputs, which is used to calculate the trajectory set.

Furthermore, the dynamics model symmetry is leveraged to reduce the computation burden during the online phase. The state space subset that does not affect the state derivatives function $f(\mathbf{x}, \mathbf{u})$ (see eq. (4.28)) is denoted as $\mathrm{X_d} \subseteq \mathrm{X}$, where X is the entire state space. It is defined as

$$f(\mathbf{x}_1, \mathbf{u}) = f(\mathbf{x}_2, \mathbf{u}); \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathrm{X_d}. \tag{4.29}$$

The trajectories for states $\mathbf{x} \in \mathrm{X_d}$ can be shifted and rotated without precomputation. An example of such states is the north/east position (assuming a flat road), where the precomputed trajectory is not dependent on the absolute north/east position and can be rotated and shifted from the origin where the offline trajectory is precomputed. For other states $\mathbf{x} \notin \mathrm{X_d}$, the space is discretized (in space) within a particular range for which the trajectories are precomputed. The discretization is also done for the input space U. Finally, trajectories are precomputed for a grid of all possible combinations of the discretized state space, specified time range, and input space as shown in the Algorithm 4.

Algorithm 4: Trajectory Precomputation

Output: T
Input: $\mathrm{X}/\mathrm{X_d}$, U, *time*

```
T ← ∅;
x_vect ← discretize(X/X_d);
u_vect ← discretize(U);
t_vect ← discretize(time);
foreach x ∈ x_vect do
    foreach u ∈ u_vect do
        foreach t ∈ t_vect do
            T ← T ∪ IVP(x, u, t)
return T;
```

The MVP for precomputed trajectories is proposed in Algorithm 5. The *Append* function used in the algorithm takes the applicable trajectory (depending on the input u_{new}) from the set of precomputed trajectories T and applies it to the state s. The *steer* function needs to be redefined for precomputed trajectories.

Steer Function – Precomputed Trajectories

Given two states $s, s_{new} \in S$ and the set of trajectories T, the *steer* function returns a trajectory $\phi \in \mathrm{T}$ from state s if a trajectory ϕ starting in state s using any input $u \in u_{vect}$ ends in state s' such that $||s' - s_{new}||_2 < \epsilon$, where ϵ is a sufficiently small number (designer's choice). Setting $\epsilon > 0$ allows easier connection of two trajectories accounting

for numerical solution existence of the trajectory calculation. Continuity of the planned trajectory is not crucial since the planning model is always an approximation, and the planned trajectory is assumed to be tracked with a controller. If continuity is required, ϵ can be set to zero.

Algorithm 5: MVP with Precomputed Trajectories

Output: $\bar{K}_n$, $S_{goal,K}$
Input: s_{init}, S_{goal}, T, u_{vect}, S
$S_K \leftarrow \emptyset$; $R_K \leftarrow \emptyset$; $S_{goal,K} \leftarrow \emptyset$;
add(s_{init}, S_K);
foreach $i \in \mathrm{N}_{\leq n}$ **do**
  $u_{new} \leftarrow$ sample(u_{vect});
  $s' \leftarrow$ sample(S);
  $s \leftarrow$ sampleNear(S_K, s');
  $s_{new} \leftarrow$ append$(s,\ u_{new},\ \mathrm{T})$;
  connect(s, s_{new}, R_K);
  $S_{near} =$ near(s_{new});
  foreach $s \in S_{near}$ **do**
    if steer$(s_{new}, s, \mathrm{T}) \neq \emptyset$ **then**
      connect(s_{new}, s, R_K);
  if $s_{new} \in S_{goal}$ **then**
    $S_{goal,K} = S_{goal,K} \cup \{s_{new}\}$;
  add(s_{new}, S_K);
return $\bar{K}_n = (S_K, s_{init}, R_K, \Pi, \mathcal{L}, \mathcal{W}_K)\,, S_{goal,K}$;

4.8 MVP modifications: Model and test scenario

The MVP algorithm used for the trajectory planning encapsulates the dynamic model presented in this section. Choosing such complex model as benchmark model for trajectory planning is intentional to show the reduction of the computation time for such level of complexity. The model is based on the vehicle dynamics and the battery state of charge (SOC) dynamics. The model is presented in Section 4.3

4.8.1 Test scenario

A scenario was selected to test the original MVP and both modifications. The scenario comprises the initial vehicle state, the goal state, and various static obstacles. There are two types of obstacles – a lawn and a building. The goal is to arrive at the destination while minimizing SOC consumption and travel time. At the same time, the vehicle is

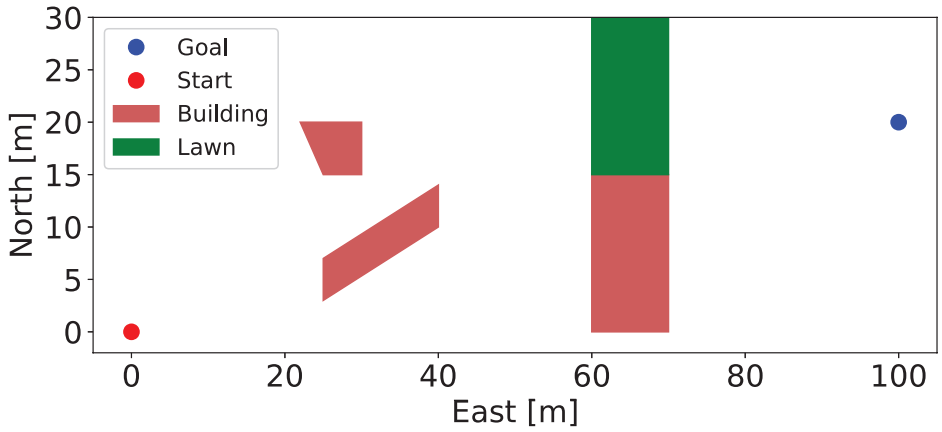


Figure 4.8: Validation scenario - environment with static obstacles.

not allowed to collide with building. It also shouldn't go over the lawn, but with lower priority then collision with building.

The scenario layout is shown in Fig. 4.8, where both types of obstacles – buildings and lawn – are present.

4.9 MVP modifications: Trajectory planning and results

Initially, the MVP cost function is proposed for trajectory planning across the three variants. Following this, both the original MVP algorithm and the proposed modifications were evaluated using the planning model detailed in Section 4.3 to plan the vehicle's state trajectory.

MVP Problem Statement

The MVP framework allows for the definition of multiple cost functions with varying priorities (refer to [75]). In the validation scenario, the hierarchy of the cost functions is defined as follows:

- The highest priority cost function for MVP states is articulated via an LTL logic statement: The vehicle must avoid colliding with any buildings.
- The second level of priority specifies: The vehicle should not traverse over lawn areas.
- The lowest priority cost function penalizes time of arrival and aims to maximize the

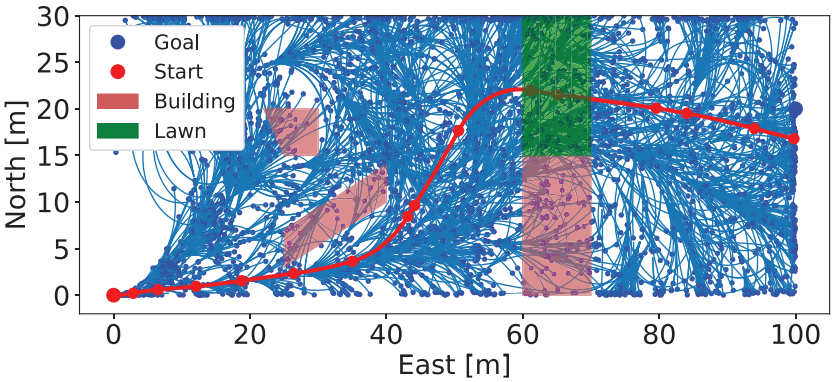


Figure 4.9: MVP planned trajectory using the MVP variant with precomputed trajectories in the validation scenario. 4000 nodes were connected to obtain the trajectory.

battery's state of charge (SOC) as:

$$J\left(SOC, t\right) = -w_{SOC} SOC + w_t t, \tag{4.30}$$

where w represents the penalizing weights, and t is the travel time. The MVP prioritizes minimizing higher priority cost functions before addressing lower priority ones. For the precomputed trajectories variant, inputs were discretized into 22 values for δ ranging between 25 and -25 degrees, and 52 values for τ ranging between -1000 and 1000 Nm.

Results

The performance of the MVP algorithm variants was evaluated on a PC equipped with a 10th generation Intel CORE i9 CPU running at 3.7 GHz. The MVP algorithm variants were implemented in Python. The computation time and cost function values for each MVP variant were recorded over 10 runs, as summarized in Table 4.1. A total of 161,721 trajectories were precomputed offline and utilized by the precomputed trajectories MVP variant. Search through this set was optimized using a KD tree, and the offline trajectory computation time was not included in the evaluation.

The computation time and cost function values for the test scenario (detailed in Section 4.8.1) with 6000 nodes forming the tree of trajectories are presented in Table 4.1. The prolonged calculation time of the BVP variant is attributed to the complexity of computing trajectories to nodes where no feasible trajectory exists due to model constraints and limited inputs. This limitation is not present in the IVP and precomputed trajectories MVP variants (refer to Section 4.7), resulting in significantly reduced computation times for these variants.

The performance degradation of the new variants is minimal, as evidenced by the cost function values for each variant's planned trajectory in Table 4.1.

Table 4.1: Comparison of MVP variants. Computations were performed on a laptop using the model described in Section 4.3. Values represent the mean and standard deviation of trajectory computation time using 6000 nodes.

MVP Variant	BVP	IVP	Precomputed trajectories
calculation time N(μ, σ) [s]	(31850, 0)	(235, 14)	(20, 1)
calculation time reduction [%]	100	0.7	0.07
cost function value N(μ, σ) [-] $\begin{bmatrix}\text{highest priority}\mu \\ \sigma \\ \text{middle priority}\mu \\ \sigma \\ \text{low priority}\mu \\ \sigma\end{bmatrix}$	$\begin{bmatrix}0\\0\\11.5\\0.5\\98000\\58000\end{bmatrix}$	$\begin{bmatrix}0\\0\\14.6\\1.6\\100000\\47000\end{bmatrix}$	$\begin{bmatrix}0\\0\\15.6\\0.9\\33000\\27000\end{bmatrix}$

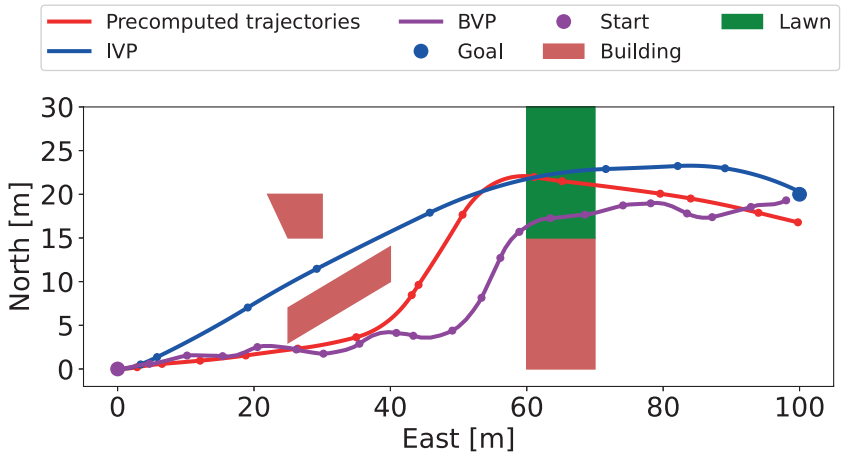


Figure 4.10: Vehicle path plan comparison for the original and the two newly proposed MVP variants.

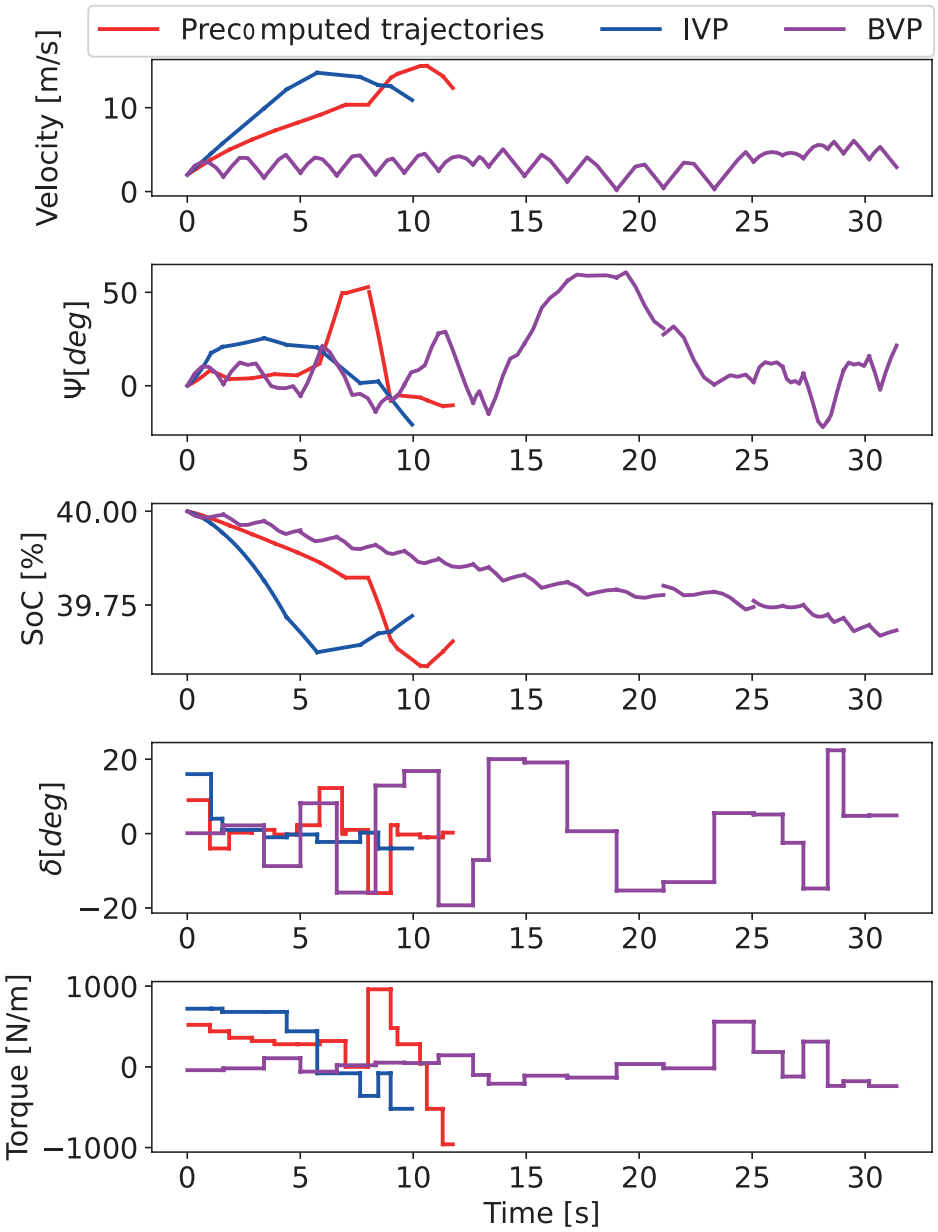


Figure 4.11: Vehicle trajectory plan comparison for selected states and inputs. The comparison is shown for the original and the two proposed MVP variants.

The tree of trajectories for the north/east position is illustrated in Fig. 4.9 using the MVP variant with precomputed trajectories (detailed in Section 4.7.3). The optimal trajectory is highlighted in red. The planning was conducted for all states in the model (vehicle yaw ψ, yaw rate r, velocity v, east and north positions e and n, and battery SOC), but only east and north coordinates are shown for clarity. It can be observed that the state space is not fully covered due to the limited number of nodes. This mirrors real-world scenarios where planning must be stopped after a certain time limit, resulting in potential variations between individual runs of the probabilistic planner. Therefore, statistics based on 10 runs are presented in Table 4.1.

A comparison of the north/east path plans for all three variants is shown in Fig. 4.10. Note that the MVP BVP plan is not continuous due to imperfections in node reconnections, where a trajectory ending in the near vicinity ($||s' - s_{new}||_2 < \epsilon$) of the state is sufficient for reconnection – strict continuity is not enforced (see Section 4.7.3 for details). Selected states and inputs from the trajectory plan are compared in Fig. 4.11. The BVP variant takes significantly more time to reach the destination, potentially due to its higher cost function minimization capability or finding a narrow local optimum. Despite the BVP's superior overall cost function performance, the performance degradation of the other variants is not significant.

4.10 Summary

This chapter delves into the comparison of two distinct trajectory planning algorithms, each stemming from different algorithmic families. The Minimum Violation Planning (MVP) and Model Predictive Control (MPC) algorithms were rigorously tested within a simulated environment using an identical test scenario and closely matched cost function tuning. The results from these tests indicate that both algorithms achieve roughly comparable outcomes, exhibiting only minimal differences in performance.

The MVP algorithm, however, appears particularly well-suited for more complex and logically constrained problems. Its ability to directly incorporate logical constraints and operate effectively in non-convex problem spaces offers significant advantages. This flexibility is especially beneficial in scenarios where traditional optimization methods may struggle due to their reliance on convex problem formulations.

The chapter further explores enhancements to the MVP algorithm aimed at reducing computation time while maintaining its inherent strengths. The proposed MVP modifications were evaluated against the original algorithm, focusing on their ability to maintain performance levels while improving computation efficiency. The benchmark tests, which concentrated on vehicle trajectory planning, revealed that the variant employing the Ini-

tial Value Problem (IVP) solution significantly reduced calculation time compared to the original MVP variant utilizing the Boundary Value Problem (BVP) approach. This reduction in computation time did not come at the expense of performance, highlighting the efficiency of the IVP-based modification.

Moreover, the variant utilizing precomputed trajectories demonstrated even greater efficiency. This approach not only maintained performance levels but also achieved an approximate tenfold increase in speed compared to the IVP-based variant. The precomputed trajectory method simplifies the online computation process by leveraging offline precomputed paths, which significantly accelerates the planning process during runtime. This method's efficacy underscores the potential for substantial performance gains through strategic algorithm modifications.

In summary, the comparison between MVP and MPC algorithms highlights the strengths and weaknesses of each approach. While both algorithms perform similarly in terms of trajectory planning under the tested conditions, the MVP's flexibility and ability to handle complex logical constraints make it particularly advantageous for intricate planning problems. The enhancements introduced to the MVP algorithm further bolster its suitability for real-time applications by drastically reducing computation times without compromising on performance. For a detailed discussion on the performance and computation time comparisons, please refer to Section 4.9.

The insights gained from this chapter emphasize the importance of selecting the appropriate algorithm based on the specific requirements of the application. For scenarios demanding high flexibility and the incorporation of logical constraints, the MVP algorithm stands out as a robust choice. Conversely, for applications where the problem space is more structured and computational efficiency is paramount, the MPC algorithm offer a compelling solution. The continued refinement of these algorithms holds promise for advancing the capabilities of autonomous vehicle trajectory planning, paving the way for more efficient and reliable autonomous systems.

Chapter 5

Conclusion

In this thesis three main topics were studied – control allocation problem, optimal slip ratio estimation, and trajectory planning algorithms.

Firstly, an innovative control system was proposed that fundamentally changes traditional control allocation methods. Instead of directly allocating torques or forces to the wheels, the system translates the vehicle's velocity or acceleration tracking problem to the wheel pivot points. This approach was published in the *IEEE Transactions on Intelligent Transportation Systems* journal. This architecture provides inherent control allocation and offers several benefits: unique wheel-level reference signals, robustness to variations in tire-to-road interface, robustness to variations in vehicle system parameters like CoG location, etc., adaptability to different vehicle configurations, ease of implementation, etc. Additionally, it maintains wheel safety limits, integrates seamlessly with existing vehicle systems, and simplifies vehicle control architecture. The system's validation through simulations experiments demonstrated its robustness, scalability, and cost-effectiveness, making it a promising alternative to traditional traction control systems.

Secondly, two novel approaches for estimating the optimal slip ratio (λ_{opt}) were introduced, using Recursive Least Squares (RLS) and Unscented Kalman Filter (UKF) algorithms. This approach was published in the *Control Engineering Practice* journal. Both methods exhibit high accuracy in simulations and real-world experiments, with the UKF-based estimator providing faster convergence and higher precision. These algorithms show significant potential for real-time application in full-scale vehicles, improving vehicle stability and safety. These algorithms estimate variations in tire-to-road interface properties and thus potentially enhancing traction control systems' performance, particularly in challenging conditions.

Finally, a comparison of two trajectory planning algorithms—Minimum Violation Planning (MVP) and Model Predictive Control (MPC) was presented. Both algorithms perform comparably in trajectory planning tasks, but MVP excels in handling complex,

logically constrained problems. Enhancements to the MVP algorithm, including the use of Initial Value Problem (IVP) and precomputed trajectories, notably reduce computation time without significantly compromising performance. These modifications make MVP particularly suitable for real-time applications. The insights from this comparison highlight the importance of algorithm selection based on application requirements, with MVP being advantageous for flexible, complex planning scenarios, and MPC for structured, efficiency-focused applications.

In summary, the proposed traction allocation control system, the optimal slip ratio estimators, and trajectory planning algorithms provide significant improvements in vehicle dynamics control and autonomous systems. These advancements pave the way for safer, more efficient, and reliable vehicle operations.

5.1 Future Work

The thesis proposed a solution for the traction allocation problem by allocating torques to each wheel. Similar architecture and results might also be drawn for the lateral dynamics of the wheel, where individual steering of each wheel can be employed. Such an architecture needs to be rigorously introduced in future work. Furthermore, the integration of the traction allocation and steering allocation systems requires development and analysis, which will be addressed in future research.

Next, the integration of the proposed optimal slip ratio λ_{opt} estimation with the traction allocation control system needs to be validated and analyzed in future work. The proposed estimators can be further improved by incorporating additional sensors and data sources, such as cameras, LiDAR, or radar, to enhance estimation accuracy and robustness.

Finally, the trajectory planning algorithms can be validated on the test platform presented in Section 3.4. Currently, the work is ongoing to validate these algorithms in real-world scenarios using the test platform. The results of this validation will be published in the *IEEE Transactions on Intelligent Transportation Systems*.

Bibliography

[1] SAE International Recommended Practice, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," in *SAE Standard J3016 202104*, April 2021.

[2] D. Schramm, M. Hiller, and R. Bardini, *Vehicle Dynamics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. [Online]. Available: http://link.springer.com/10.1007/978-3-540-36045-2

[3] K. Maeda, H. Fujimoto, and Y. Hori, "Four-wheel driving-force distribution method based on driving stiffness and slip ratio estimation for electric vehicle with in-wheel motors," in *2012 IEEE Vehicle Power and Propulsion Conference*. IEEE, oct 2012, pp. 1286–1291. [Online]. Available: http://ieeexplore.ieee.org/document/6422490/

[4] Center for Sustainable Systems, University of Michigan, "Personal transportation factsheet, pub. no. css01-07," in *Tech. Rep.*, 2022.

[5] Eurostat, "Road safety statistics in the EU." [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Road_safety_statistics_in_the_EU&oldid=630784#The_number_of_persons_killed_in_road_traffic_accidents_fell_by_22_.25_between_2012_and_2022

[6] M. Carlier, "Number of electric vehicles in use by type 2016-2022." [Online]. Available: https://www.statista.com/statistics/1101415/number-of-electric-vehicles-by-type/

[7] Z. Wang, H. Zheng, C. Zong, and C. Kaku, "Research on Control Strategy of Hierarchical Architecture Based on Drive-by-Wire Chassis," in *SAE Technical Paper 2023-01-0819*, apr 2023. [Online]. Available: https://www.sae.org/content/2023-01-0819

[8] J. Ni, J. Hu, and C. Xiang, "A review for design and dynamics control of unmanned ground vehicle," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 4, pp. 1084–1100, mar 2021. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0954407020912097

[9] P. Song, C. F. Zong, H. Y. Zheng, and L. He, "Rapid Prototyping for the Vehicle Control Unit of a Full Drive-by-Wire Electric Vehicle," *Advanced Materials Research*, vol. 694-697, pp. 1573–1581, may 2013. [Online]. Available: https://www.scientific.net/AMR.694-697.1573

[10] A. Foundation, "Autoware Project: The world's leading open-source software project for autonomous driving." [Online]. Available: https://autoware.org/

[11] Apollo, "Apollo autonomous driving." [Online]. Available: https://en.apollo.auto/

[12] Tesla, "Autopilot and Full Self-Driving Capability." [Online]. Available: https://www.tesla.com/support/autopilot

[13] D. Vošahlík and T. Haniš, "Traction Control Allocation Employing Vehicle Motion Feedback Controller for Four-Wheel-Independent-Drive Vehicle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 14 570–14 579, dec 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10194411/

[14] D. Vošahlík, T. Veselý, T. Hanis, and J. Pekar, "Brake Control Allocation Employing Vehicle Motion Feedback for Four-Wheel-Independent-Drive Vehicle," in *SAE Technical Papers*, 2023, pp. 1–12.

[15] J. Guo, Y. Luo, and K. Li, "An adaptive hierarchical trajectory following control approach of autonomous four-wheel independent drive electric vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2482–2492, 2018.

[16] R. de Castro, R. E. Araújo, M. Tanelli, S. M. Savaresi, and D. Freitas, "Torque blending and wheel slip control in EVs with in-wheel motors," *Vehicle System Dynamics*, vol. 50, no. sup1, pp. 71–94, jan 2012. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00423114.2012.666357

[17] D. Vošahlík and T. Hanis, "Real-time estimation of the optimal longitudinal slip ratio for attaining the maximum traction force," *Control Engineering Practice*, vol. 145, no. July 2023, p. 105876, apr 2024. [Online]. Available: https://doi.org/10.1016/j.conengprac.2024.105876https://linkinghub.elsevier.com/retrieve/pii/S0967066124000364

[18] D. Vosahlik, J. Cech, T. Hanis, A. Konopisky, T. Rurtle, J. Svancar, and T. Twardzik, "Self-Supervised Learning of Camera-based Drivable Surface Friction," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, vol. 2021-Septe, no. 1, pp. 2773–2780, 2021.

[19] D. Vosahlik, P. Turnovec, J. Pekar, and T. Hanis, "Vehicle Trajectory Planning : Minimum Violation Planning and Model Predictive Control Comparison," in *33rd IEEE Intelligent Vehicles Symposium*, 2022.

[20] D. Vosahlik, P. Turnovec, J. Pekar, M. Bohac, and T. Hanis, "Vehicle Dynamics Trajectory Planning: Minimum Violation Planning Modifications Reducing Computational Time," in *2023 SICE International Symposium on Control Systems (SICE ISCS)*. IEEE, mar 2023, pp. 27–32. [Online]. Available: https://ieeexplore.ieee.org/document/10079204/

[21] V. Ivanov, D. Savitski, and B. Shyrokau, "A Survey of Traction Control and Antilock Braking Systems of Full Electric Vehicles with Individually Controlled Electric Motors," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 9, pp. 3878–3896, 2015.

[22] H. Park and J. C. Gerdes, "Optimal tire force allocation for trajectory tracking with an over-actuated vehicle," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2015-Augus, no. Iv, pp. 1032–1037, 2015.

[23] Y. Tian, X. Cao, X. Wang, and Y. Zhao, "Four wheel independent drive electric vehicle lateral stability control strategy," *IEEE/CAA Journal of Automatica Sinica*, pp. 1–13, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8766204/

[24] W. Li, H. Du, and W. Li, "Four-Wheel Electric Braking System Configuration with New Braking Torque Distribution Strategy for Improving Energy Recovery Efficiency," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 87–103, 2020.

[25] J. Hu, J. Tao, F. Xiao, X. Niu, and C. Fu, "An Optimal Torque Distribution Control Strategy for Four-Wheel Independent Drive Electric Vehicles Considering Energy Economy," *IEEE Access*, vol. 7, pp. 141 826–141 837, 2019.

[26] Y. Ge and C. Chang, "Torque distribution control for electric vehicle based on traction force observer," *Proceedings - 2011 IEEE International Conference on Computer Science and Automation Engineering, CSAE 2011*, vol. 2, no. 3, pp. 371–375, 2011.

[27] G. Amato and R. Marino, "Distributed Nested PI Slip Control for Longitudinal and Lateral Motion in Four In-Wheel Motor Drive Electric Vehicles," *Proceedings of the IEEE Conference on Decision and Control*, vol. 2019-Decem, no. Cdc, pp. 7609–7614, 2019.

[28] H. Pacejka, *Tyre and Vehicle Dynamics*. Elsevier LTD, Oxford, 2002. [Online]. Available: https://www.ebook.de/de/product/18341528/hans{_}pacejka{_}tire{_}and{_}vehicle{_}dynamics.html

[29] V. Cibulka, "MPC Based Control Algorithms for Vehicle Control," 2019.

[30] Smart Driving Solutions Research Center, "SDS TwinTrackGit," 2022. [Online]. Available: https://github.com/SDS-RC-FEE-CTU-in-Prague/TwinTrack

[31] L. Pugi, F. Grasso, M. Pratesi, M. Cipriani, and A. Bartolomei, "Design and preliminary performance evaluation of a four wheeled vehicle with degraded adhesion conditions," *International Journal of Electric and Hybrid Vehicles*, vol. 9, no. 1, p. 1, 2017. [Online]. Available: http://www.inderscience.com/link.php?id=82812

[32] C. Riese and F. Gauterin, "Evaluation of a State of the Art Hydraulic Brake System with Regard to Future Requirements," *SAE International Journal of Passenger Cars - Mechanical Systems*, vol. 9, no. 3, pp. 2016–01–1927, sep 2016. [Online]. Available: https://www.sae.org/content/2016-01-1927/

[33] M. Z. Islam, A. Singh, A. Arvanitis, M. Younkins, S. Leng, P. Carvell, A. Tripathi, Z. Chen, and A. Phillips, "Efficiency Improvement of Electric Motor Drives Using Dynamic Motor Drive Technology," in *SAE Technical Paper 2022-01-0721*, mar 2022. [Online]. Available: https://www.sae.org/content/2022-01-0721/

[34] Adams/Tire, "Using the PAC2002Tire Model." [Online]. Available: https://docplayer.net/54206752-Using-the-pac2002tire-model.html

[35] H. Pacejka, *Tire and Vehicle Dynamics*. Elsevier LTD, Oxford, 2012. [Online]. Available: https://www.ebook.de/de/product/18341528/hans{_}pacejka{_}tire{_}and{_}vehicle{_}dynamics.html

[36] F. Jia, Z. Liu, H. Zhou, and W. Chen, "A novel design of traction control based on a piecewise-linear parameter-varying technique for electric vehicles with in-wheel motors," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 10, pp. 9324–9336, 2018.

[37] M. Ringdorfer and M. Horn, "Development of a wheel slip actuator controller for electric vehicles using energy recuperation and hydraulic brake control," in *2011 IEEE International Conference on Control Applications (CCA)*. IEEE, sep 2011, pp. 313–318. [Online]. Available: http://ieeexplore.ieee.org/document/6044472/

[38] M. Lehtla and H. Hõimoja, "Slip control upgrades for light-rail electric traction drives," *2008 13th International Power Electronics and Motion Control Conference, EPE-PEMC 2008*, pp. 1581–1584, 2008.

[39] "EVC 1000, European Union's Horizon 2020 Project." [Online]. Available: http://www.evc1000.eu/en

[40] S. Seyedtabaii and A. Velayati, "Adaptive optimal slip ratio estimator for effective braking on a non-uniform condition road," *Automatika*, vol. 60, no. 4, pp. 413–421, 2019. [Online]. Available: https://doi.org/00051144.2019.1637053

[41] L. Zhai, T. Sun, and J. Wang, "Electronic Stability Control Based on Motor Driving and Braking Torque Distribution for a Four In-Wheel Motor Drive Electric Vehicle," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 4726–4739, jun 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7404035/

[42] Smart Driving Solutions Research Center, "SDS Formula Git," 2022. [Online]. Available: https://github.com/SDS-RC-FEE-CTU-in-Prague/FormulaCarMaker

[43] Smart Driving Solutions Research Center, "SDS youtube," 2022. [Online]. Available: https://www.youtube.com/watch?v=9BoSNcziuhw

[44] T. Veselý, "Brake-by-Wire System Development," Master's thesis, Czech Technical University in Prague, 2022.

[45] H. Peng, W. Wang, C. Xiang, L. Li, and X. Wang, "Torque Coordinated Control of Four In-Wheel Motor Independent-Drive Vehicles with Consideration of the Safety and Economy," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9604–9618, 2019.

[46] V. Rodrigo Marco, J. Kalkkuhl, J. Raisch, W. J. Scholte, H. Nijmeijer, and T. Seel, "Multi-modal sensor fusion for highly accurate vehicle motion state estimation," *Control Engineering Practice*, vol. 100, jul 2020.

[47] G. Papa, M. Tanelli, G. Panzani, and S. M. Savaresi, "Wheel-slip estimation for advanced braking controllers in aircraft: Model based vs. black-box approaches," *Control Engineering Practice*, vol. 117, dec 2021.

[48] F. Sun, X. Huang, J. Rudolph, and K. Lolenko, "Vehicle state estimation for anti-lock control with nonlinear observer," *Control Engineering Practice*, vol. 43, pp. 69–84, oct 2015.

[49] The MathWorks, "Modeling an Anti-Lock Braking System," 2020. [Online]. Available: https://www.mathworks.com/help/simulink/slref/modeling-an-anti-lock-braking-system.html

[50] S. Stefano, M. Luca, D. Daniele, and T. Marco, "Antilock braking systems, devices, and methods using sensorized brake pads," U.S. Granted Patent US 10 227 064 B2, 2019. [Online]. Available: https://lens.org/046-284-213-667-359

[51] E. Ono, K. Asano, M. Sugai, S. Ito, M. Yamamoto, M. Sawada, and Y. Yasui, "Estimation of automotive tire force characteristics using wheel velocity," *Control Engineering Practice*, vol. 11, no. 12, pp. 1361–1370, 2003.

[52] J. Villagra, B. D'Andréa-Novel, M. Fliess, and H. Mounier, "A diagnosis-based approach for tire-road forces and maximum friction estimation," *Control Engineering Practice*, vol. 19, no. 2, pp. 174–184, 2011.

[53] X. Zhang and D. Göhlich, "A hierarchical estimator development for estimation of tire-road friction coefficient," *PLoS ONE*, vol. 12, no. 2, pp. 1–21, 2017.

[54] A. Kunnappillil Madhusudhanan, M. Corno, M. A. Arat, and E. Holweg, "Load sensing bearing based road-tyre friction estimation considering combined tyre slip," *Mechatronics*, vol. 39, pp. 136–146, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.mechatronics.2016.03.011

[55] K. Han, E. Lee, M. Choi, and S. B. Choi, "Adaptive Scheme for the Real-Time Estimation of Tire-Road Friction Coefficient and Vehicle Velocity," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 4, pp. 1508–1518, 2017.

[56] H. Mirzaeinejad and M. Mirzaei, "A novel method for non-linear control of wheel slip in anti-lock braking systems," *Control Engineering Practice*, vol. 18, no. 8, pp. 918–926, aug 2010.

[57] H. Kataoka, H. Sado, I. Sakai, and Y. Hori, "Optimal slip ratio estimator for traction control system of electric vehicle based on fuzzy inference," *Electrical Engineering in Japan (English translation of Denki Gakkai Ronbunshi)*, vol. 135, no. 3, pp. 56–63, may 2001.

[58] Z. Huang, Z. Xu, B. Chen, R. Zhang, Y. Chen, and Q. Peng, "Sliding Mode Control for Urban Railway Anti-Slip System Based on Optimal Slip Ratio Estimation with Forgetting Factor Recursive Least-Squares," Tech. Rep., 2017.

[59] B. Chen, Z. Huang, R. Zhang, F. Jiang, W. Liu, H. Li, J. Wang, and J. Peng, "Adaptive slip ratio estimation for active braking control of high-speed trains," *ISA Transactions*, vol. 112, pp. 302–314, jun 2021.

[60] F. Holzmann, M. Bellino, R. Siegwart, and H. Bubb, "Predictive estimation of the road-tire friction coefficient," *Proceedings of the IEEE International Conference on Control Applications*, no. 1, pp. 885–890, 2006.

[61] M. Riehm, T. Gustavsson, J. Bogren, and P. E. Jansson, "Ice formation detection on road surfaces using infrared thermometry," *Cold Regions Science and Technology*, vol. 83-84, pp. 71–76, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.coldregions.2012.06.004

[62] M. Ohori, T. Ishizuka, T. Fujita, N. Masaki, and Y. Suizu, "Fundamental study of smart tire system," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1519–1524, 2006.

[63] A. Tuononen, "Optical position detection to measure tyre carcass deflections," *Vehicle System Dynamics*, vol. 46, no. 6, pp. 471–481, jun 2008. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00423110701485043

[64] A. Soltani, S. Azadi, and R. N. Jazar, "Integrated control of braking and steering systems to improve vehicle stability based on optimal wheel slip ratio estimation," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 44, no. 3, p. 102, mar 2022. [Online]. Available: https://link.springer.com/10.1007/s40430-022-03420-2

[65] R. Rajamani, G. Phanomchoeng, D. Piyabongkarn, and J. Y. Lew, "Algorithms for real-time estimation of individual wheel tire-road friction coefficients," *IEEE/ASME Transactions on Mechatronics*, vol. 17, no. 6, pp. 1183–1195, 2012.

[66] C. Lee, K. Hedrick, and K. Yi, "Real-Time Slip-Based Estimation of Maximum Tire–Road Friction Coefficient," *IEEE/ASME Transactions on Mechatronics*, vol. 9, no. 2, pp. 454–458, jun 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1306460/

[67] E. Joa, K. Yi, K. Sohn, and H. Bae, "Four-wheel independent brake control to limit tire slip under unknown road conditions," *Control Engineering Practice*, vol. 76, no. March, pp. 79–95, 2018. [Online]. Available: https://doi.org/10.1016/j.conengprac.2018.04.001

[68] M. Wielitzka, M. Dagen, and T. Ortmaier, "Sensitivity-based Road Friction Estimation in Vehicle Dynamics using the Unscented Kalman Filter," *Proceedings of the American Control Conference*, vol. 2018-June, pp. 2593–2598, 2018.

[69] V. Peterka, "Control of uncertain processes: applied theory and algorithms," *Kybernetika*, vol. 22, no. 7, pp. 3–101, 1986.

[70] I. The MathWorks, "Tire-Road Interaction (Magic Formula)," 2011. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/tireroadinteractionmagicformula.html

[71] A. H. Jazwinski, *Stochastic processes and filtering theory*. Courier Corporation, 2007.

[72] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[73] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." in *Proc. ICLR*, 2015.

[74] J. Tumova, L. I. Reyes Castro, S. Karaman, E. Frazzoli, and D. Rus, "Minimum-violation LTL planning with conflicting specifications," *Proceedings of the American Control Conference*, pp. 200–205, 2013.

[75] T. Wongpiromsarn, K. Slutsky, E. Frazzoli, and U. Topcu, "Minimum-Violation Planning for Autonomous Systems: Theoretical and Practical Considerations," *Proceedings of the American Control Conference*, vol. 2021-May, pp. 4866–4872, 2021.

[76] L. Palmieri and K. O. Arras, "A novel RRT extend function for efficient and smooth mobile robot motion planning," in *IEEE International Conference on Intelligent Robots and Systems*, no. Iros. IEEE, sep 2014, pp. 205–211. [Online]. Available: http://ieeexplore.ieee.org/document/6942562/

[77] V. Vonasek, M. Saska, K. Kosnar, and L. Preucil, "Global motion planning for modular robots with local motion primitives," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, may 2013, pp. 2465–2470. [Online]. Available: http://ieeexplore.ieee.org/document/6630912/

[78] A. Katriniok, P. Sopasakis, M. Schuurmans, and P. Patrinos, "Nonlinear Model Predictive Control for Distributed Motion Planning in Road Intersections Using PANOC," *Proceedings of the IEEE Conference on Decision and Control*, vol. 2019-Decem, no. Cdc, pp. 5272–5278, 2019.

[79] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Predictive Control for Multi-Robot Motion Planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, 2020.

[80] P. Turnovec, "Optimal Planning and Control of Vehicle Dynamics," 2021.

[81] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, mar 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/rob.20285

[82] G. Bitar, V. N. Vestad, A. M. Lekkas, and M. Breivik, "Warm-Started Optimized Trajectory Planning for ASVs," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 308–314, 2019. [Online]. Available: https://doi.org/10.1016/j.ifacol.2019.12.325

[83] Y. Kuwata, G. Fiore, J. Teo, E. Frazzoli, and J. How, "Motion planning for urban driving using RRT," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 196, no. 3. IEEE, sep 2008, pp. 1681–1686. [Online]. Available: http://ieeexplore.ieee.org/document/4651075/

[84] L. E. Kavraki, M. N. Kolountzakis, and J. C. Latombe, "Analysis of probabilistic roadmaps for path planning," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 166–171, 1998.

[85] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[86] J. H. Ryu, D. Ogay, S. Bulavintsev, H. Kim, and J. S. Park, "Development and experiences of an autonomous vehicle for high-speed navigation and obstacle avoidance," *Studies in Computational Intelligence*, vol. 466, pp. 105–116, 2013.

[87] J. H. Jeon, R. V. Cowlagi, S. C. Peters, S. Karaman, E. Frazzoli, P. Tsiotras, and K. Iagnemma, "Optimal motion planning with the half-car dynamical model for autonomous high-speed driving," *Proceedings of the American Control Conference*, pp. 188–193, 2013.

[88] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Robotics: Science and Systems*, vol. 6, pp. 267–274, 2011.

[89] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[90] F. von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H. J. Wuensche, "Driving with tentacles: Integral structures for sensing and motion," *Journal of Field Robotics*, vol. 25, no. 9, pp. 640–673, 2008.

[91] P. Resende and F. Nashashibi, "Real-time dynamic trajectory planning for highly automated driving in highways," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 653–658, 2010.

[92] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu, "Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 2, pp. 740–753, 2016.

[93] X. Qian, F. Altché, P. Bender, C. Stiller, and A. De La Fortelle, "Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 205–210, 2016.

[94] M. Elbanhawi, M. Simic, and R. Jazar, "Randomized Bidirectional B-Spline Parameterization Motion Planning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 406–419, 2016.

[95] J. W. Lee and B. Litkouhi, "A unified framework of the automated lane centering/changing control for motion smoothness adaptation," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 282–287, 2012.

[96] A. Piazzi, C. Guarino, L. Bianco, and M. Bertozzi, "Quintic G2 -Splines for the Iterative Steering of vision base autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 2, pp. 27–36, 2002.

[97] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha - A local, continuous method," *IEEE Intelligent Vehicles Symposium, Proceedings*, no. Iv, pp. 450–457, 2014.

[98] A. Richards and J. How, "Mixed-integer programming for control," *Proceedings of the American Control Conference*, vol. 4, pp. 2676–2683, 2005.

[99] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[100] J. Bohren, T. Foote, J. Keller, A. Kushleyev, D. Lee, A. Stewart, P. Vernaza, J. Derenick, J. Spletzer, and B. Satterfield, "Little Ben: The Ben Franklin Racing Team's entry in the 2007 DARPA Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 9, pp. 598–614, sep 2008. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/rob.21514/abstracthttps://onlinelibrary.wiley.com/doi/10.1002/rob.20260

[101] H. Bouchareb, K. Saqli, N. Mapos, . Sirdi, M. Oudghiri, A. Naamane, and N. K. M'sirdi, "Electro-thermal coupled battery model : State of charge, core and surface temperatures estimation," *ICEERE2020 2nd International Conference on Electronic Engineering and Renewable Energy*, 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02486440

[102] M. Morari and J. H. Lee, "Model predictive control: Past, present and future," *Computers and Chemical Engineering*, vol. 23, no. 4-5, pp. 667–682, 1999.

Appendix A

Appendix

A.1 Author's publications

This section presents author's publications and patents

A.1.1 Related to The Thesis

Journal publications

David Vošahlík, Tomas Hanis, **"Real-time estimation of the optimal longitudinal slip ratio for attaining the maximum traction force"** in *Control Engineering Practice*, Volume 145, 2024, 105876, ISSN 0967-0661

D. Vošahlík and T. Haniš, **"Traction Control Allocation Employing Vehicle Motion Feedback Controller for Four-Wheel-Independent-Drive Vehicle,"** in *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 14570-14579, Dec. 2023

Conference papers

Vošahlík, D., Veselý, T., Hanis, T., and Pekar, J., **"Brake Control Allocation Employing Vehicle Motion Feedback for Four-Wheel-Independent-Drive Vehicle,"** in *SAE Technical Paper* 2023-01-1866, 2023

D. Vosahlik, P. Turnovec, J. Pekar, M. Bohac and T. Hanis, **"Vehicle Dynamics Trajectory Planning: Minimum Violation Planning Modifications Reducing Computational Time,"** in *2023 SICE International Symposium on Control Systems (SICE ISCS)*, Kusatsu, Japan, 2023

D. Vosahlik, P. Turnovec, J. Pekar and T. Hanis, **"Vehicle Trajectory Planning: Minimum Violation Planning and Model Predictive Control Comparison,"** in *2022 IEEE Intelligent Vehicles Symposium (IV)*, Aachen, Germany, 2022, pp. 145-150

D. Vosahlik et al., **"Self-Supervised Learning of Camera-based Drivable Surface Friction,"** in *2021 IEEE International Intelligent Transportation Systems Confer-*

ence (ITSC), Indianapolis, IN, USA, 2021, pp. 2773-2780

D. Vošahlík, T. Hanis and M. Hromčík, **"Vehicle longitudinal dynamics control based on LQ,"** in *2019 22nd International Conference on Process Control (PC19)*, Strbske Pleso, Slovakia, 2019, pp. 179-184

A.1.2 Not Related to The Thesis

Patents

David Vosahlik, Matej Pcolka, **"Driver and lead vehicle prediction model,"** *US20240132066A1*

Martin Herceg, Matej Pcolka, Bohumil Hnilicka, Daniel Youssef, Kamil Dolinsky, Jaroslav Pekar, David Vosahlik, **"Energy efficient predictive power split for hybrid powertrains,"** *US20240132047A1*

Matej Pcolka, David Vosahlik, **"Non-selfish traffic lights passing advisory systems,"** *US20230286508A1*

A.2 Declaration of Generative AI and AI-assisted Technologies in the Writing Process

During the preparation of this work the author used ChatGPT in order to improve readability. After using this tool, the author reviewed and edited the content as needed and take full responsibility for the content of the publication.