UC Davis
UC Davis Electronic Theses and Dissertations

Title

Cheap Lateral Motion Control of Automated Driving Systems in Adverse Conditions

Permalink

https://escholarship.org/uc/item/4p0931df

Author

Vidano, Trevor

Publication Date

2024

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Cheap Lateral Motion Control of Automated Driving Systems in Adverse Conditions

By

TREVOR VIDANO

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Mechanical and Aerospace Engineering

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Francis Assadian, Chair

Shima Nazari

Iman Soltani

Committee in Charge

2024

To Erica,

who's unwavering support made this possible.

Contents

List of Figures

List of Tables

Abstract

Adverse weather conditions, mainly uncertain road surface conditions, are a remaining challenge in the field of lateral motion control of an Automated Driving System. This thesis proposes treating this challenge as one of addressing model parameter uncertainty. This perspective is framed as a multi-objective optimal robust control problem. The first objective is to simultaneously minimize yaw and lateral position reference tracking errors despite the condition of the road surface. The second objective is to minimize actuator effort. To ensure that the solutions remain practical, this thesis constrains the possible solutions to *cheap* control techniques (techniques that use low amounts of runtime and memory).

Before developing controllers, this thesis first develops a new methodology that can be used to determine the amount of tracking error that can be tolerated to achieve an overall target level of safety for the entire Automated Driving System. The new methodology combines three techniques: (1) risk allocation, (2) geometric analysis of the driving task, and (3) statistics. The methodology argues that the control requirements must be designed with consideration of the performance of upstream modules such as planning and localization.

To investigate and compare the current state-of-the-art in this field, a high-fidelity simulator is developed with the commercial software CarSim. This simulator enables the simulation of many lateral motion controllers in a wide variety of maneuvers and environmental conditions. Following the development of this simulator, the bicycle car model is used to develop different approaches to lateral motion control. This analysis presents new understandings of how the controller's performance is influenced by the many possible definitions of the system's state. Lateral motion control is then separated into trajectory tracking and path tracking. For the remainder of the thesis, path-tracking is used because it is shown to better decouple lateral motion control performance from longitudinal motion control.

One of the challenges of designing a robust controller is balancing conservatism with performance.

To address this, a new Linear Parameter Varying (LPV) control technique is developed. This new LPV control technique allows for the design of a controller whose parameters are continuously varying with the scheduled parameters without requiring special characteristics of the model (such as being affine with the scheduling parameters). This approach is used to design a high-performance controller in simulation. Real-world vehicle results on closed-course maneuvers and public road routes show that this new design performs well. The most significant result is that this new LPV control achieves a sufficient balance between tracking performance and passenger comfort for the velocity range of 5-30 m/s on concrete road surfaces and for highway-like and collision-avoidance maneuvers. This same controller also achieves this satisfactory performance on a gravel road between 5 and 15 m/s.

The next investigation compares the performance of several controllers that require low computational resources. These results, collected in simulation, argue that many path-tracking control problems can be solved by cheap controllers. More complex controllers are not necessary and instead the researchers in this field should focus less on the commonly solved problems such as lane-keeping, lane-changes, collision-avoidance and more on limit handling, adverse road surface conditions, fault tolerance, and other practical challenges.

The final investigation shows that steering dynamics (which are often neglected in lateral motion control) can play a significant role in performance and need to be included in the overall control design. In this investigation, the problem of having little information on the steering actuator is addressed by developing a new data-driven model that captures key characteristics of the actuator. The first characteristic accurately modeled is the dependence of the static gain on velocity. The second characteristic modeled is the decrease in system damping as velocity increases. A steering controller is then developed using this new model. An outer loop controller performs path-tracking control. On-vehicle results show that the resulting design works well on highways in ideal and rainy conditions.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Assadian, for his unwavering support, guidance, and patience throughout my PhD journey. His invaluable insights and expertise have been instrumental in shaping this dissertation and my academic growth.

I am also profoundly grateful to my committee members, Prof. Nazari and Prof. Soltani, for their thoughtful feedback, and for challenging me to refine and improve my work. Their contributions have significantly enriched my research.

I would like to extend my heartfelt thanks to my fellow students and colleagues: Dr. Louis Filipozzi, Dr. Ehsan Arasteh, George Martin, Yin-Hsuan Huang, Jonas Lossner, and Shivam Vashi. Your camaraderie, collaborative spirit, and intellectual discussions have made this journey both enjoyable and intellectually stimulating.

Special thanks to Texas A & M University and Prof. Langari for leading the Autonomous Vehicles for All project, which is generously funded by the US Department of Transportation. This project has provided me with invaluable research opportunities and resources.

I would also like to express my gratitude to Nuro and my managers, Dan Ratner, Dr. Kiumars Jalali, and Dr. Mohammad Shakiba, for their mentorship and support throughout the summers that I worked there. Their guidance has greatly influenced my deep interest in connecting theory to practice and considering the practicality of my work throughout my professional development.

To my fiancé, Erica Skytte, your love, support, and understanding have been my anchor throughout this journey. Thank you for your endless encouragement and for being my rock during the most challenging times.

I am deeply indebted to my parents, Paul and Sally Vidano, for their unwavering belief in me and for their constant support in every endeavor I pursue. Your love and sacrifices have made all of this possible.

Lastly, I want to thank my dear friends Shane Cody, Mario Zendejas, and Dr. Aimee Bains for reminding me to maintain a balance between work and life. Your friendship has been a source of joy and a much-needed escape during this intense period.

Thank you all for being a part of this journey.

Chapter 1

Introduction

1.1 Research Goal and Scope

Automobiles that can safely transport passengers without a driver manually controlling the steering wheel have been a research goal for control engineers since the 1950s [73]. It wasn't until recently that this dream was realized in limited domains by self-driving vehicle companies such as Waymo, Cruise, and Nuro (to name a few) [62, 56, 152]. Recently, the self-driving car industry has prioritized demonstrating the safety of their technology in urban environments such as San Francisco, Mountain View, Phoenix, and Houston. This choice balances constraining the operational domains to limit the engineering problem and accessing a large enough customer basis for profitability. This raises the potential issue that populations that are only accessible via rural highways or that live in areas that frequently experience adverse weather conditions will likely not benefit from the advances of the self-driving car industry. To overcome this issue, significant advances in the research and development of Automated Driving Systems (ADS) must be realized. One of the major challenges in extending the operating domains of ADS into adverse weather and road conditions is the difficulty in tracking lateral and yaw references with a front steering wheel actuator.

For much of driving, the longitudinal dynamics are constant. However, there are critical situations in which they are not. The coupling between longitudinal and lateral dynamics appears mostly in the kinematics and the tire forces (as will be covered in Chapter 3). The traditional approach to solving ADS motion control treats the lateral dynamics as independent of the longitudinal dynamics. In each motion controller's design, the other motion's dynamics are treated as model uncertainties and disturbances. For instance, in the design of a lateral motion controller, the longitudinal dynamics are commonly assumed constant, thus

treating the coupling as unmodeled uncertainty. The control task is made more difficult because robust performance must be guaranteed when the longitudinal dynamics are not constant. There have been numerous examples of impressive feats achieved with this decoupled architecture. The work in [11, 121, 122, 176, 178, 189, 179, 132, 76, 96] has demonstrated significant advances in limit-handling control. Much of this work uses decoupled lateral and longitudinal motion controllers. Additional work by the Technical University of Munich in the application of Autonomous Racing provides further evidence that separating functionalities into lateral and longitudinal controllers can provide safe and acceptable performance at the limits of handling [92, 26, 90, 191, 25, 192, 190, 193].

With this growing set of evidence that separating lateral and longitudinal motion control functions can achieve acceptable performances, this thesis focuses only on the lateral motion control problem. Specifically, this is the problem of tracking lateral position references with a vehicle. However, throughout this dissertation yaw angle references will also be tracked. This is because the yaw dynamics are closely coupled with the lateral dynamics. Throughout this thesis, lateral dynamics will refer to both lateral and yaw dynamics.

Much of the literature around solving this problem has focused on proposing new control techniques to solve the lateral motion control problem under conservative constraints. The historical perspective presented in Section 1.3 will illustrate this. Gradually, new problems have been solved, but there is little cohesive understanding of the literature's progress in solving this problem more broadly. Specifically, the ideal solution would be a single control algorithm that requires minimal computational resources (memory and runtime) and human resources (tuning and experimentation) to achieve safe driving through all environmental conditions in which humans operate vehicles. The existence of such a solution is unknown. However, this thesis takes a systematic approach towards trying to understand how to get to this solution. The goal is to push the research community towards reaching this ideal solution rather than proposing new solutions for constrained problems that may already have a simple solution.

There are three major aspects of this ideal solution. The first is to have a single control algorithm. This is ideal because it constrains the solution's complexity. To achieve this, the controller must satisfy conflicting requirements such as passenger comfort, safe tracking performance, fault reliability, and security against adversarial cyber-attacks. This dissertation focuses primarily on satisfying safe tracking performance because it is considered the most important requirement. However, some considerations will be made at appropriate points for passenger comfort. The rest are outside the scope. The second aspect of the ideal

solution is minimal computational resources. To address this, the control methods explored in this thesis are constrained to control methodologies that require low amounts of CPU runtime and computer memory (called “Cheap" controllers). This restriction excludes any methods that require online iteration to solve an optimization problem. The third aspect of this ideal solution is to consider the tuning effort of each controller. To address this, model-based control approaches are used in which the model is low-fidelity and identified from limited amounts of data. This eliminates more recent modeling advances such as neural networks from being considered. The controllers are then designed to be robust against model uncertainty. Alternative approaches exist such as adaptive or control solutions that learn a model online [34]. These approaches show promise but are not considered in this thesis.

1.2 Dissertation Outline

To this end, the thesis is organized around exploring the answers to questions that all relate to the overall goal and are bound within the scope. Each chapter addresses a different question, but concepts span multiple chapters. The chapters are organized as follows:

1. The first chapter (Section 1.3) addresses the question of what has been done in the past to solve the lateral motion control problem. Section 1.3 presents a somewhat linear timeline of some significant breakthroughs in the literature. It partitions the history into before and after the DARPA Grand Challenge [141].

2. The second chapter asks how well a controller must perform to ensure safe, public road deployment. To date, this question has not been rigorously answered in the literature. Such an answer must be constrained with realistic assumptions about acceptable levels of risk and current ADS subsystem performances. More specifically, the aim is to propose a methodology for designing requirements. The goal is not to develop a verification and validation scheme. Verification and validation efforts are better left to the industry and the well-established practices in the automotive space. Developing a method to derive performance requirements is still significant because it is a prerequisite for verification and validation efforts.

3. The third chapter develops a simulation framework that can safely, repeatably, and reliably test different lateral motion controllers. The system design is based on the existing autonomous vehicle used in

the Automated Vehicles for All Project [22]. In this vehicle, measurements from an Inertial Measurement Unit and a Real-Time Kinematic Global Positioning System are fused to provide accurate pose estimates. The lateral motion controller commands either a steering wheel angle or a steering torque (applied to the rack and pinion by an electric motor). The longitudinal motion controller interface exposes a throttle and brake pedal command. To ground the testing methodology in reality, the controller design does not have access to the true parameters of the vehicle. Instead, such parameters must be estimated from system identification techniques. Finally, five Operational Design Domains are designed to simulate various environmental conditions. All of this is interfaced with the commercial software CarSim (version 2020.0) to capture the nonlinear effects of the vehicle dynamics.

4. The fourth chapter asks how some modeling and controller implementation choices affect the control design and the achievable performance. Specifically, the dynamic bicycle car model is explored for lateral motion control by proposing three unique state variable definitions. Each resulting model is considered in terms of controllability and implementation nuances. Then a series of simulation experiments are performed to explore the relative effects of each modeling and implementation choice on the lateral motion tracking performance. There are five contributions made in this chapter. (1) It is shown that the commonly used error state model [160] is useful because it reduces control performance sensitivity to the references (explained further in this Chapter) and linearizes the model for position tracking. (2) Several different error state formulations are developed to demonstrate that properties that are beneficial for control can be achieved by different varieties (suggesting further directions of research). (3) An implementation technique is proposed to use the linearized global state model without needing gain scheduling on the yaw angle (which is often required to guarantee the validity of the small-angle assumption). (4) A new modification proposed for regulator design with the error state model introduces little additional complication while significantly improving performance. (5) It is shown that the coupling between the lateral and longitudinal motion control performance is increased when the references are parameterized by time rather than by space (tracking a trajectory or a path).

5. The fifth chapter asks if the lateral controller needs to be gain scheduled on the vehicle velocity to achieve robust performance throughout a reasonable velocity range. Several approaches are employed to answer this question. First, the question is redefined as a robust control design problem and the

improvement in robust performance (quantified by the H-infinity norm) is shown for gain-scheduling. Secondly, simulation results are used to show that a Linear Parameter Varying (LPV) controller can achieve excellent robust performance. To accomplish this, the Linear Time-Invariant (LTI) control design method using Interpolation Conditions [61] is extended to LPV systems and applied to solve the lateral motion control problem. Finally, empirical evidence collected on the AVA Project's 2019 Jeep Grand Cherokee is presented to show the achievable robust performance improvement with an LPV control design. The control architecture used to achieve the results on the Jeep (a cascade or inner-outer architecture) differs from the one used to achieve the results in the Simulator because of the additional challenges presented by the Jeep. The main difference is that the Simulator requires a steering wheel angle as a command (as explained in Chapter 3) whereas the Jeep uses a steering torque (further explained in Chapter 7). The steering control development is presented in Chapter 7 and is not discussed in this Chapter since Chapter 7 addresses the challenges presented by this change in interface. This Chapter is concerned only with the development and demonstration of the LPV lateral motion controller. This Chapter is presented before Chapter 7 only because this control technique will be used in Chapter 6.

6. The sixth chapter explores the current state of the literature in solving the lateral motion control problem. This study proposes five maneuvers to test each controller. It also uses the five Operational Design Domains proposed in Chapter 3 and on safe performance requirements. A selection of four controllers from the literature are tuned and applied to this problem. They are each an example of a unique control design category: nonlinear feedforward, optimal state feedback, driver-model-based, and frequency domain. The results show that several of these cheap controllers (control techniques that require low amounts of CPU runtime, computer memory, and control-parameter tuning effort) solve each maneuver and Operational Design Domain combination.

7. The seventh chapter addresses how to account for steering dynamics in the design of lateral motion controllers. Many publications neglect the effects of steering dynamics, but it is shown that they play a significant role and must be carefully accounted for in reality. To study this topic, parallel efforts are carried out on the simulation proposed in Chapter 3 and the 2019 Jeep Grand Cherokee. This firmly grounds all approaches in reality. Two approaches are studied in simulation: a cascade control architecture and a model inversion-based control architecture. Simulation results show that the

cascade control architecture presents the most promise for high-quality, robust control. These results are then built upon and a cascade control architecture is constructed for the 2019 Jeep Grand Cherokee. This control architecture is then tested on closed-closed maneuvers and public road routes during different weather conditions to demonstrate the approach's efficacy. There are several novelties in this chapter. First, an LPV controllable canonical form state space model is used to capture the Jeep's steering system dynamics. This method is motivated by the lack of knowledge about the system. Then an LPV H-infinity dynamic output feedback controller is developed to track steering wheel angle commands. The outer loop controller is a feedforward-feedback architecture that commands a steering wheel angle. Despite significant communication delay in the system safe, robust, and comfortable performance is achieved. This Chapter develops the inner loop controller used to achieve the empirical results presented in Chapter 5. While Chapter 5 uses an LPV lateral motion controller, this Chapter uses an LTI lateral motion controller because the focus is primarily on the torque-based steering controller as opposed to the development of an LPV lateral motion controller. The reader is suggested to recall the empirical results of Chapter 5 collected on the Jeep to further understand the full capability of a torque-based lateral motion control scheme.

8. The final chapter reflects on the results presented in this thesis. It identifies themes throughout the thesis and discusses some lessons that have been learned. To conclude the thesis, recommendations are proposed for the future direction of the field of lateral motion control.

1.3 A Historical Perspective on Lateral Motion Control

1.3.1 Before the DARPA Grand Challenge

The design of a lateral motion controller for automobiles has been an active research topic since the World's Fair in 1939-40. The General Motors Pavilion proposed the concept of a vehicle that drives itself [107]. In the 1950s, they demonstrated this was possible on a test track [73]. After that work, it appears that the interest in ADS only grew (although the terminology was quite different from what is used now). In the late 1960s, the Bureau of Public Roads, which would later become the Federal Highway Administration (FHWA), proposed the concept of an Automated Highway System. During this time the initial foundations for understanding vehicle dynamics for control were laid. Ohio State University (OSU) was one of the first

academic institutions to begin research in this area. By 1976 they had a semi-autonomous vehicle capable of driving between 50 and 80 mph in wet and dry road conditions on relatively low curvature roads ($< \frac{1}{841}\frac{1}{\text{m}}$) [64, 65]. They state that their controller did not perform well on curves, a challenge that persists today. Their techniques used classical control theory and they used a modified bicycle car model to capture high-velocity vehicle phenomena. While not widely agreed-upon, they argue that the bicycle car model poorly models the understeer characteristics of vehicles at these velocities, but this can be mitigated with additional algebraic terms in the dynamic model. They also use a simple model of their actuator to capture the subsystem's bandwidth of 4.8 rad/s. Finally, they employ a form of look-ahead to compute their reference. Although it appears that they were not fully aware of the benefits offered by this choice. It wouldn't be until decades later, under the California Program on Advanced Technology for the Highway, which is now the California Partners for Advanced Transportation Technology (PATH), that the implications of this choice would be better understood. Before explaining this, we first need to introduce California PATH.

This age of research and development came to an end in 1981 when the new US administration shifted priorities away from advanced research in transportation systems [107]. It wasn't long until the country was beginning to struggle with increasing amounts of traffic which became a major priority of the FHWA, state transportation departments, and related agencies. In 1986 the California PATH began research on the Automated Highway System. Under these research efforts, new concepts were developed. The general architecture of modern ADS was coming into focus. The direction of research and development was towards a decomposed architecture composed of interconnected modules. For instance, the sensing system provided a supervisory controller (now called a planner) with information about the ego vehicle and its environment. The planner then provides the motion controller (often decoupled into lateral and longitudinal control) with a reference. The motion control efforts were divided into different tasks. The simplest tasks were grouped into what is now known as SAE Level 2 functions such as Lane Keeping Assist [50]. It was identified that emergency control (collision avoidance) tasks deserved their separate investigation [89].

While California PATH dominated much of the experimental literature in the 1980s, work continued throughout the US. For information regarding research outside the US at this time, see [107, 171]. Ohio State University (OSU), for instance, continued its work in lateral motion control. In 1980-87 work was done on scheduling a lateral motion controller with the longitudinal velocity of the vehicle [66, 54]. Several controllers were developed including a cascade controller and an optimal state feedback controller, each scheduled with longitudinal velocity. This group of work determined that steering performance was more

sensitive to changes in cornering stiffness and longitudinal velocity than other bicycle model parameters. These controllers performed well between velocities of 10 and 35 m/s on a test track with curvatures as high as 1/100 (1/m), on straight-line waterlogged routes, and lane changes subject to side winds of 13 m/s. However, it should be noted that the high speeds were primarily in low curvature portions, while low speeds were used on high curvature portions. In other words, the operation was not at the vehicle's limits of handling. The high-level outcome of these efforts is an argument that lateral motion control on highways is achievable with very simple feedback controllers. This is qualified by the fact that their experiments did not push the vehicle to lateral accelerations above 2 m/s^2.

Before the late 1990s, most controllers focused on speeds lower than typical highway driving, except for the OSU efforts [83]. By 1997, more than 200 papers had been published proposing unique lateral motion controllers [81]. Few were demonstrated on actual vehicles. The California PATH program provided unique experimental opportunities. Several lateral motion controllers were developed including frequency-shaped linear quadratic control (FSLQ) [158], PID, fuzzy rule-based control [95], feedback linearization, and surface control [159]. Overall these controllers achieve less than 10 cm error at 50 km/hr (31 mph) on a road that demands less than 2 m/s^2 lateral acceleration. The FSLQ controller, in particular, showed good robustness against feedback discontinuities, parameter mismatch, and wet road surfaces [171]. However, this work focused only on lane-keeping.

As several lateral motion controllers were being developed and investigated for the Automated Highway System, several investigations showed significant benefits of using reference path information in front of the vehicle [83, 81]. When using information collected at the vehicle's center of gravity (look-down systems), the dynamic system quickly becomes under-damped as velocity increases. Furthermore, the transfer function from the front steering angle to lateral error produces increasing phase lag with increasing velocity. This is shown in Figure 1.1 where as velocity increases, the lines get darker.

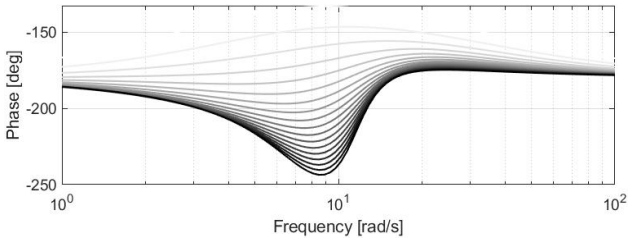


Figure 1.1: Phase plot of the transfer function from front steer angle to lateral error as velocity increases from 0-40 m/s

In general, it seems that most low-speed lateral motion control problems are solvable with look-down systems as the vehicle dynamics are negligible and depend little on road friction [83]. At higher velocities, it becomes beneficial to use information collected at some look-ahead distance in front of the vehicle. When

using a look-ahead system, phase lead is introduced to the system, mitigating the phase lag shown in Figure 1.1. The complexity in design occurs when speed is increased further and more phase lead is required for stability and robustness requirements. This increasing demand results in higher gains above 1 Hz, which can conflict with steering actuator dynamics [83]. An additional consideration that was investigated and solved under some limitations is the detection and lateral control of a vehicle in the event of a tire blowout [156]. In the nomenclature of this thesis, this can be incorporated as a specific Operational Design Domain.

In contrast to the results concerning look-ahead systems and the argument that the controller must be scheduled with velocity, Ackermann et al. developed fixed-gain controllers for a bus. These controllers are robust enough to operate throughout the full speed range, under different payloads, and varying road surface conditions all without preview information [3, 82, 2]. The underlying argument this group makes is that rather than measure or estimate model parameters online, the design of a robust lateral motion controller results in a simpler controller that still meets performance requirements. The control architecture they use is quite different than the look-ahead controllers. Rather than directly using a steering angle, they command a steering angle rate. Several decades later, [181] studies how human drivers steer a car and argues that the human actuates a steering wheel angular velocity rather than a steering wheel angle. Furthermore, in the absence of preview information, they make use of yaw rate information and perform control on tracking a reference yaw rate.

By the mid-1990s most lateral motion controllers focused on lane keeping and some on lane changes. A notable exception is [175] which argues that low-order models, such as the 2DOF lateral bicycle car model, are insufficient when lateral accelerations exceed 0.2 g's (a claim that is disproven by the results of Chapter 3 as well as throughout this Thesis. Most maneuvers do not exceed this lateral acceleration limit due to passenger comfort considerations. So work that is constrained only to lane keeping and lane changing can likely use low-order models. However, when collision avoidance maneuvers are considered, [175, 174] argues that low-order models should not be used (a claim that is argued against in this Thesis). The most significant effect is seen from tire non-linearity. The second most significant effect is lateral tire lag. Similar arguments are made in more recent control papers [121, 76], where it is determined that modeling tire saturation is critical when performing control in high slip conditions. The major conclusion of [174] is that it might be possible to design a linear controller for these highly nonlinear regions of operation. They suggest gain scheduling the linear controller to optimize the nonlinear model performance in simulation.

In 1997, California PATH performed a demonstration of platooning vehicles that steer automatically on

a California freeway [172]. Despite its widespread news coverage, the US Department of Transportation (USDOT) terminated its joint effort with California PATH called the National Automated Highway Systems Consortium (NAHSC) in 1998, disrupting much of the ongoing research and development. This also caused a large body of research to go unpublished to the public until they were digitized and released by California PATH in 2017. Despite this loss of funding, several projects continued well into the mid-2000s focusing on transit buses and tractor-trailer trucks [172].

One key point about the control system development during these decades is that most controllers relied on infrastructure-based sensing systems. Magnets were embedded into the road and were capable of providing precise localization information to the vehicle regardless of weather conditions [186]. However, some work in Japan during the late 1970s through the mid-1990s focused on Computer Vision-based sensors [171]. This is interesting to compare with the now more popular technique of using forward-facing cameras and LiDARs. While the magnetic-based sensing systems required extensive infrastructure modifications, they were very robust. The modern techniques require relatively cheaper sensor installations but suffer from robustness to weather conditions [28, 201].

The result of [170], in terms of defining performance requirements, is two separate specifications: one for safety and one for comfort. The safety requirement specifies that the lateral motion controller should maintain the vehicle within 10 cm of lateral deviation (no more than 3 cm r.m.s) when driving on roads requiring up to 1.96 m/s^2 (0.2 g). The comfort requirement states that the r.m.s. lateral acceleration be no greater than 0.03 g above steady-state lateral acceleration and whose peaks are no more than 20% above steady-state lateral acceleration. Most of the passenger comfort is attributable to infrastructure design such as speed limits and road radii.

A convenient point to delineate past from present is the shift in research focus from Automated Highway Systems to what is now called Automated Driving Systems (ADS). This delineation is most conveniently found in the DARPA Grand Challenge [141] and the DARPA Urban Challenge [142], which gave rise to the modern self-driving vehicle industry. Before moving on to lateral motion control of ADS, it is worth noting the current automotive development of Active Safety Systems built on the historical work discussed in this section.

Due to the maturity of Automated Highway Systems, it is not surprising that consumers have access to Active Safety Systems. In 2021, 50% of new vehicle models offered sustained Lane Keeping Assistance [187]. Insight into the control techniques used by these systems is not publicly available, except for Comma

Ai's open-source OpenPilot [49]. This system uses a Deep Neural Network to extract a future waypoint directly from a forward-facing camera. This waypoint is then used in separate Longitudinal and Lateral Model Predictive Control-based local planners to generate a reference path. The reference path is then tracked by a lateral motion controller. The lateral motion controller uses a PID feedback control law with a feed-forward controller.

The presence of these systems on the market suggests that the automotive industry believes that the lane-keeping problem is mostly solved to an acceptable level of safety. However, that level of safety is not explicitly stated. Recent testing from AAA has shown that moderate to heavy rain conditions drastically deteriorate the performance of lane-keeping systems [59]. Furthermore, even in normal highway driving conditions, an adverse event was observed once every 8 miles in the lane-keeping systems studied [1]. Therefore, it is easy to conclude that the industrial implementation of lateral motion controllers still has a way to go to achieve the promised performance of California PATH's research. Furthermore, an open question that was partially addressed throughout research on Automated Highway Systems is how to design a lateral motion controller that is simple and robust to adverse weather and other environmental disturbances.

1.3.2 State-of-the-Art

By 2003 the field of advanced robotics had progressed so much that the Defense Advanced Research Projects Agency (DARPA) thought it possible to put a vehicle in the desert of California and traverse a route of 140 miles without human intervention. In the first attempt, none of the 15 vehicles that were tested in 2004 completed the course. However, by 2005 more than 23 groups were ready to try again, and 5 of them would be successful [141]. This challenge was the largest experiment of off-road lateral motion control. The most significant lateral motion controller developments consist of the Stanley controller [99], which demonstrated that robustness against surface conditions and road disturbances can be performed using a simple nonlinear feedback law using a kinematic model of the vehicle system. It should be noted that additional tuning parameters were included to account for vehicle and steering actuator dynamics [141]. Another nonlinear controller, pure pursuit, was used by Carnegie Mellon University's entry to track a path with the steering actuator [47, 48]. The basic pure pursuit algorithm was found to be inadequate in the presence of steering actuator nonlinearities, so it was augmented with an integral function [141]. These two works popularized the use of robotics-inspired lateral motion controllers on automobiles. In contrast to this, the KAT-5 vehicle used a lead-lag controller based on the dynamic bicycle car model with look-ahead [141]. It was determined

that scheduling gains with velocity for their vehicle and test conditions were not necessary. The controller is shown to perform with less than a 5 cm standard deviation of lateral error from the reference for 28 miles. Another model-based controller was used by Auburn University's vehicle that was based on controlling the heading angle directly. Both the Ohio State University's vehicle and Virginia Tech's vehicle use fuzzy logic lateral motion controllers. Overall, the most significant outcome of the DARPA Grand Challenge for the field of lateral motion control was the demonstration that robotics-inspired controllers could achieve similar, if not better, performance as controllers based on vehicle dynamic models. However, these results should be qualified by noting that the maximum speed was 35 mph.

Since then, numerous survey papers have experimentally compared several lateral motion control implementations either in reality or in simulation [46, 164, 27, 125]. In aggregate, lateral motion controllers can be grouped into the following categories: geometric/kinematic, offline optimal, online optimal, dynamic model-based nonlinear, and data-driven.

Examples of geometric or kinematic controllers are the Stanley and Pure Pursuit control algorithms. While these have shown excellent performance experimentally, they are generally argued as being insufficient at highway speeds (> 50 mph). However, kinematic model-based controllers have had some success in adverse conditions [130, 60]. These are based on the Model Predictive Control algorithms though, and are better classified as online optimal.

The offline optimal controllers mostly consist of Linear Quadratic Regulators and H-infinity control techniques. The application of these control techniques is very diverse, utilizing many different formulations of dynamic vehicle models and Linear Parameter Varying extensions [151, 101, 160, 135, 117, 55, 124, 118].

The online optimal control techniques are mostly focused on Model Predictive Controllers and their various extensions to Linear Parameter Varying and Linear Time-Varying Systems. The main challenge observed by these controllers is balancing predictive performance with computational complexity. It is common to see kinematic models extended with various terms to account for significant nonlinearities [130, 60, 143, 182, 189]. This trade-off tends to make online optimal controllers less attractive to the industry. However, it appears that they also occupy the spot of the most performant lateral motion controllers in the literature [164, 27].

The dynamic model-based nonlinear controllers are mostly composed of sliding mode control and Lyapunov techniques [177, 2, 88, 178, 121, 76]. The sliding mode control techniques suffer from chattering, necessitating a more involved fix. This typically requires designing a smoother nonlinear function, using an

observer-based sliding mode controller, or using higher-order sliding mode controllers. Despite this, low online computational complexity is achieved with good robustness. Other nonlinear control techniques make use of potential fields or nonlinear tire model inversion [178, 121, 76]. These show excellent performance in highly nonlinear operating domains.

Finally, data-driven approaches are increasingly popular [27]. In general, they seek to achieve similar breakthroughs as those experienced in applying Machine Learning to Computer Vision or Markov Decision Processes. In general, these techniques lack sufficient experimental demonstration on full-scale vehicles [164, 27]. One notable exception is the application of Neural Networks to model lateral vehicle dynamics in all weather conditions [176]. By using the Neural Network in the feed-forward controller, they sidestep the challenges presented by achieving stability when a Neural Network is used in the feedback. This application also shows that the Neural Network can learn a function that outputs an appropriate steering angle on various road surface conditions without explicitly estimating the friction coefficient.

Based on this Subsection and Subsection 1.3.1's review of the literature, the following gaps in the literature can be identified:

- What is the required lateral motion performance of a typical passenger vehicle that guarantees a safety requirement on typical roadways in the United States?
- Which, if any, of the existing lateral motion controllers meet these safety requirements under ideal weather conditions?
- Under which conditions do the existing front lateral motion controllers fail to meet these requirements?
- What control technique offers the best balance between performance, passenger comfort, robustness to non-ideal road conditions, and computational complexity?

Chapter 2

Control Performance Requirements of Automated Driving Systems

This research investigates the development of risk-based performance requirements for Automated Driving System (ADS) control. The proposed method begins by determining the target level of safety for the Virtual Driver of an ADS. Publicly-available data informs the underlying assumptions, and modifications are suggested to improve these assumptions. Next, a geometric technique is used to derive deterministic performance levels of the Virtual Driver. This technique is based on typical roadway designs and vehicle sizes. To integrate the risk and performance requirements seamlessly, this Chapter proposes new definitions for errors associated with the Planner, Pose, and Control modules. These definitions facilitate the derivation of stochastic performance requirements for each module, ensuring an overall target level of safety. Notably, these definitions enable real-time controller performance monitoring, potentially enabling fault detection linked to the system's overall safety target. At a high level, this approach argues that the requirements for the Virtual Driver's modules should be designed simultaneously instead of separately. To illustrate this approach, this technique is applied to a research project available in the literature that developed an automated steering system for an articulated bus. This case study illustrates that this method supports unique vehicle geometries. Most importantly, the example demonstrates that this method can generate experimentally-verifiable performance requirements.

2.1 Introduction

Automated Driving Systems (ADS) have the opportunity to reduce on-road collisions and fatalities. Proving that a particular ADS has achieved this is a major challenge to the industry [62, 56]. While various standards have been proposed to aid in achieving high levels of safety [116, 112], this is still an open question and often debated topic in the literature [126, 140, 74]. Furthermore, the proposed safety goals commonly lack performance metrics (with one notable exception [162]). This makes it difficult to design a control system

that achieves a level of safety.

Most performance requirements for a lateral or longitudinal controller have been based on some combination of values such as maximum lateral error, maximum lateral jerk, or maximum lateral acceleration [81, 180, 164]. Many papers that propose new lateral path tracking controllers do not present a required level of performance. They tend to focus on proving feasibility rather than a sufficiently safe performance by displaying real-world or simulation results from one or a few tests [164, 46, 100, 176]. While many controllers achieve impressive performance, it is not always clear if they deliver performance sufficient for safe autonomous driving. Without a predefined requirement, the control engineer is forced to resort to a trial-and-error design procedure. However, errors can be severe when the ADS is deployed on public roads.

To aid in the design of a controller, it is often useful to impose an architecture on ADS functionalities. A convenient architecture is to define a Virtual Driver System (VDS) and a Vehicle System. The VDS is composed of the tasks that a human performs when manually driving. The Vehicle System is composed of the physical vehicle and its low-level controllers. This partition is useful because it allows ADS developers to leverage the design and manufacturing experience of existing Original Equipment Manufacturers. This leaves the ADS developers with more resources to dedicate to the VDS.

Figure 2.1 shows how the VDS is further broken down into modules. Most modules in Figure 2.1 are well-known, except for the Pose module. In this architecture, the Pose module consumes sensor outputs and the output of the Localization to provide feedback to the Control module. Some architectures simply use the Localization module in place of the Pose module. The remainder of this Chapter will assume the existence of the Pose module. For architectures that do not use a Pose module, the following methods applied to the Pose module can be directly applied to the Localization module.

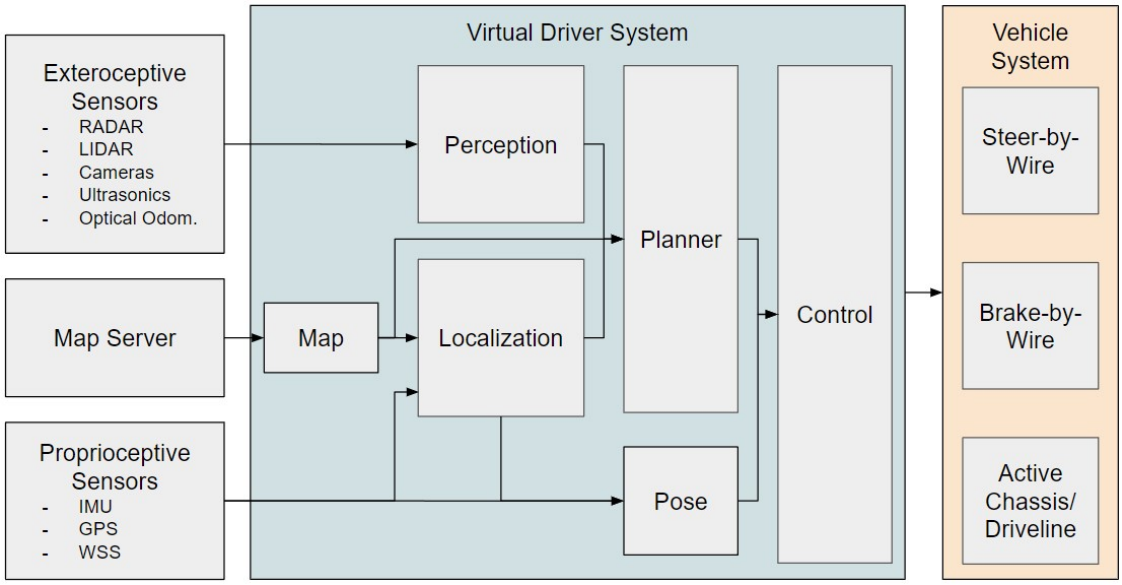


Figure 2.1: A generic ADS architecture separating the VDS and Vehicle System

The main contributions of this chapter are:

1. A new way to determine performance requirements for the Control module is presented in Sections 2.2 and 2.3. This method allocates performance requirements to the Planner, Pose, and Control modules simultaneously. In Section 2.4, this method is shown to generate feasible requirements that are also verifiable using experimental data.

2. Rather than apply position and orientation requirements to the Localization Module, as is done in [162], this Chapter applies these requirements to the VDS as a whole. The result is a new set of definitions for each module's performance metrics. A consequence of this is that the Control module's performance requirements are directly verifiable online.

3. A missing assumption in [162]'s definitions underlying their risk allocation is discussed and addressed in Section 2.3.

4. The performance requirements for the Planner, Pose, and Control modules are all formulated as stochastic characteristics, aiding in their ability to be verified with experimental data.

The remainder of the Chapter is organized as follows: Section 2.2 introduces safety terminology such as Alert Limits and Protection Levels. In the same section, an example vehicle is used to demonstrate how these definitions generate deterministic requirements. Section 2.3 reviews the risk allocation in [162] and

identifies a deficiency in their definition of failure. A fix to this deficiency is then proposed. The same section then allocates risk to the Planner, Pose, and Control modules, which enables the conversion of deterministic requirements from Section 2.2 into statistical metrics. Section 2.4 provides a case study demonstrating: (1) how to use this new method, (2) some shortcomings of the method, and (3) the method's feasibility. Section 2.5 concludes this Chapter with a summary of its findings and a brief discussion of its limitations.

2.2 Terminology

[162] proposes a geometric way to determine target performance levels called Protection Levels. This is a more rigorous extension of the traditional derivation of maximum lateral position from the lane's width. The result is a set of equations describing the coupling between lateral and longitudinal positional errors for various road types and passenger vehicles in the United States. The method begins with defining a failure as any part of the vehicle departing from the lane. First, a rectangular Alert Limit with length y and width x is drawn within a lane of constant radius, as shown in Figure 2.2. The road's radius, r, is measured from the road's center of curvature.

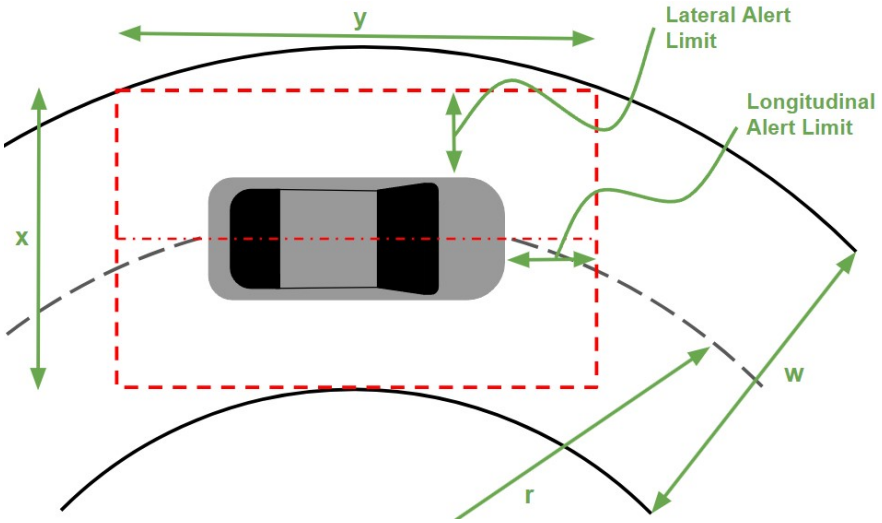


Figure 2.2: Lateral and Longitudinal Alert Limits

The Lateral and Longitudinal Alert Limits are computed with the following equations from [162]:

$$x = \sqrt{(r + \frac{w}{2})^2 - (\frac{y}{2})^2} + \frac{w}{2} - r \tag{2.1}$$

$$AL_{lat} = \frac{(x - w_v)}{2} \tag{2.2}$$

$$AL_{lon} = \frac{(y - l_v)}{2} \tag{2.3}$$

where w_v is the vehicle's width in meters, l_v is the vehicle's length in meters, AL_{lat} is the Lateral Alert Limit, AL_{lon} is the Longitudinal Alert Limit, and the remaining variables are as defined in Figure 2.2.

To demonstrate the application of Equations 2.1, 2.2, and 2.3, this Section uses a 2019 Jeep Cherokee, whose width is 1.9 m and length is 4.6 m. The Alert Limits are computed for Freeway, Interchange, Arterial, Collector, and Local roads and shown in Figure 2.3. The road specifications come from [162], where the Local roads are assumed to be 3.0 m wide and have a minimum radius of curvature of 10 m. The results show that Freeways are the least restrictive, while Local roads are the most restrictive.

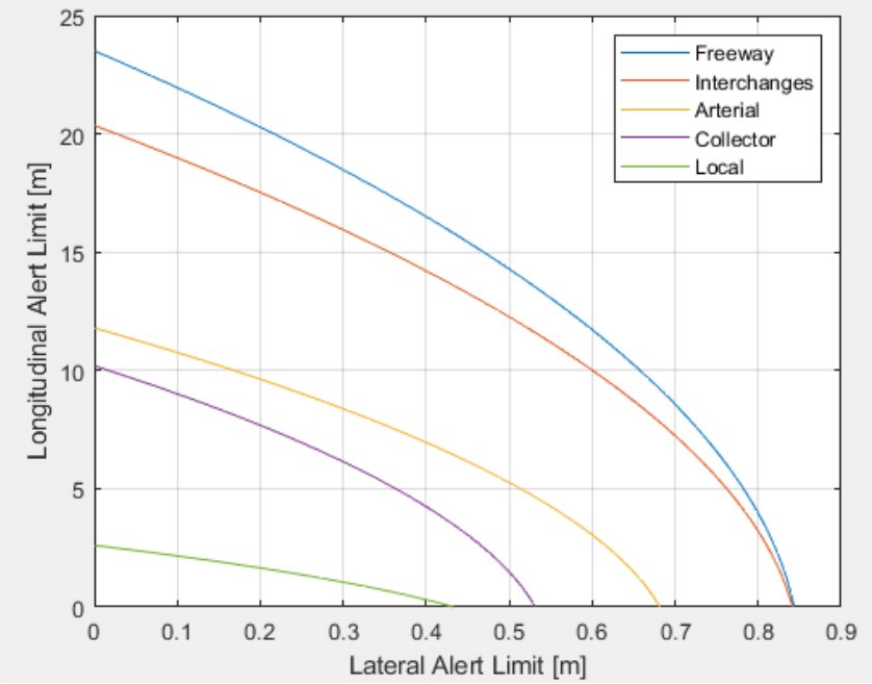


Figure 2.3: The trade-off curves between the lateral alert limit and the longitudinal alert limit for a 2019 Jeep Cherokee

The Alert Limits define only lateral and longitudinal limits for failure. To incorporate yaw angle limits, Protection Levels are introduced. Protection Levels define the maximum lateral and longitudinal position error and yaw angle error. Their combination defines a rectangle that must be within the Alert Limits. While [162] includes vertical errors, this Chapter focuses only on planar errors because the Control module is typically incapable of controlling vertical movement. Figure 2.4 shows a visualization of these Protection

Levels.

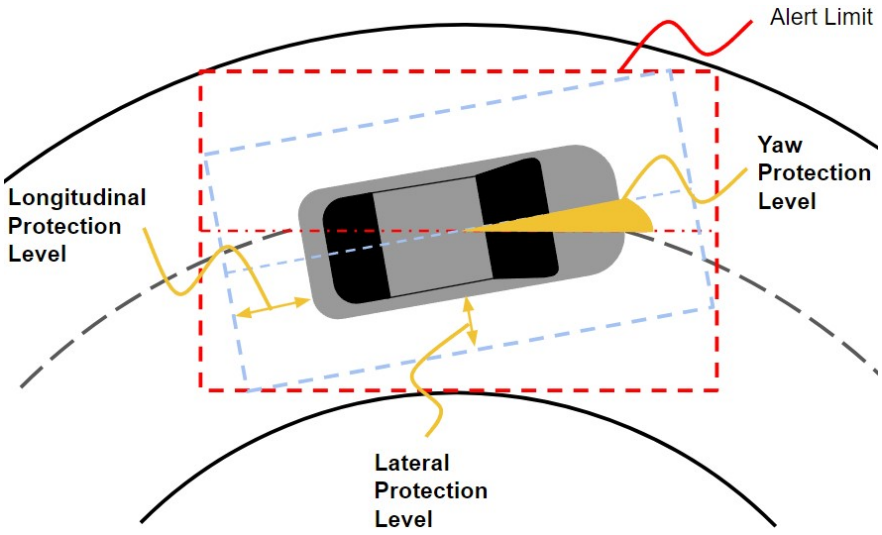


Figure 2.4: The Protection Levels depicted as a rotated rectangle within the Alert Limit

The next step is to derive the equations governing the trade-off between Lateral, Longitudinal, and Yaw Protection Levels. Using a small angle assumption and considering only planar components, the following can be derived:

$$\delta_{lat} + (\delta_{lon} + \frac{l_v}{2})\delta_\psi \leq AL_{lat}$$
$$\delta_{lon} + (\delta_{lat} + \frac{w_v}{2})\delta_\psi \leq AL_{lon}$$

where δ_{lat}, δ_{lon}, and δ_ψ are the Lateral, Longitudinal, and Yaw Protection Levels, respectively.

Next, rearrange these equations and assume equality instead of inequality to get:

$$\begin{bmatrix} \delta_{lat} \\ \delta_{lon} \end{bmatrix} = \begin{bmatrix} 1 & \delta_\psi \\ \delta_\psi & 1 \end{bmatrix}^{-1} \begin{bmatrix} AL_{lat} - \frac{l_v}{2}\delta_\psi \\ AL_{lon} - \frac{w_v}{2}\delta_\psi \end{bmatrix}$$

After substituting Equations 2.1, 2.2, and 2.3 in, the Protection Levels can be shown to be functions of y and δ_ψ. The engineer now only needs to decide on the necessary Protection Levels rather than determining both Protection Levels and Alert Limits. It is useful to generate plots of the trade-off between each Protection Level as is done in Figure 2.5. Figure 2.5 shows the computed Protection Levels when $\delta_\psi = 0.05$ rad and

for a range of y. To use these curves, the engineer need only select a point along these curves and the two remaining Protection Levels are determined. If needed, the Alert Limits can then be computed from this selection.

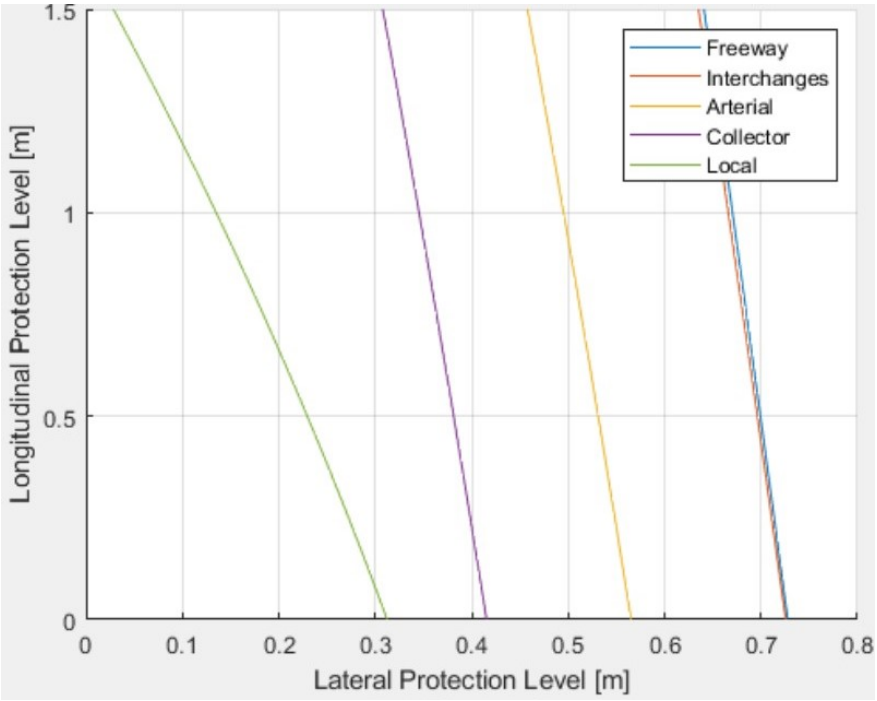


Figure 2.5: Protection Levels of a 2019 Jeep Cherokee when the Yaw Protection Level is set at 0.05 rad (2.9 deg).

The derivation of Protection Levels is based only on the lane and vehicle geometries. Therefore, these limits are independent of the environmental conditions. If the ADS cannot meet these Protection Levels under adverse weather conditions, then the ADS should not be operating in those conditions.

It has been pointed out that using rectangles to model the vehicle in the lane yields an overly conservative Protection Level [63, 127]. A less conservative approach is to model the vehicle as a rectangle with curved corners modeled as ellipsoids. This produces larger Protection Levels and the trade-off curves in Figure 2.2 have a smaller curvature (which makes choosing tradeoffs more favorable). This alternative geometric model is recommended for most vehicles.

The Protection Levels derived in [162] are assigned to the Localization module. Those authors define a Localization module's failure as the error of the vehicle's pose estimate exceeding the Protection Levels. However, this Chapter argues that Protection Levels are better applied to the VDS as a whole. When applied in this way, the performance requirements can be derived for the Control module. In other words, a failure is

when a part of the vehicle's position leaves the lane, not when the Localization error exceeds the Protection Levels.

Furthermore, this failure definition can be abstracted to all driving tasks if the lane is replaced with a Virtual Corridor. A Virtual Corridor is a space that guarantees a driving task (lane keeping, collision avoidance, overtaking, etc.) is accomplished. For example, when the driving task is an overtaking maneuver, the Virtual Corridor is a collision-free space that results in the ego vehicle passing another vehicle. The Virtual Corridor could be the lane for the start and end portions of the maneuver, but not necessarily in between.

2.3 Risk Allocation

The Protection Levels derived in Section 2.2 can be converted into characteristics of stochastic distributions by applying a system integrity risk allocation. This application will follow [162]'s example of deriving a Target Level of Safety (TLS) for the ADS. The TLS is set to be similar in magnitude to that of the Aerospace industry: 2×10^{-10} fatalities per mile. To convert this target into failures per mile, [162] defines a "fatal crash to incident ratio as $P_{F:I} = 10^{-2}$ fatal crashes/failure, ... where an incident could be seen as a lane departure or minor crash". The error in this statement is the definition of an incident being a lane departure or a minor crash. This assumption is overly conservative. Instead, there should be an additional ratio that converts a lane departure to a collision, $P_{LD:C}$.

Determining this ratio from publicly available data is challenging. According to [44], 63% of new US passenger vehicles for the model year 2017 offer Lane Departure Warning (LDW) systems. The companies that provide LDW, if they collect this data, can provide accurate estimates for the number of warnings issued to the driver, which can be used to estimate $P_{LD:C}$. Concerning publicly available data, there are few open-source data sets that contain public driving such as [167]. From this data (if there are any collisions observed), significantly more effort would be required to estimate $P_{LD:C}$, but it may be possible. For now, continue similarly to [162] and assume $P_{LD:C} = 1$ (in the case study of Section 2.4 this will be made less conservative). This results in a failure per mile of $P_{F:M} = 2 \times 10^{-8}$ for the entire ADS (from [162] assumptions and calculation). [162] identifies that current passenger vehicles have a $P_{F:M} = 1 \times 10^{-8}$. Therefore, it is reasonable to split the ADS $P_{F:M}$ evenly between the VDS and Vehicle System.

Further allocation of this probability to the Pose, Control, and Planner modules should be informed by

expected performance data from each module. For now, use the allocation proposed in [162]: the Planner is allocated $P_{traj} = 5.5 \times 10^{-9}$ failures per mile, the Pose module is allocated $P_{pose} = 10^{-9}$ failures per mile, and the Controller module is allocated $P_{ctrl} = 3.5 \times 10^{-9}$ failures per mile (the case study in Section 2.4 will use a more informed allocation).

By defining errors of the Planner, Pose, and Control modules as Gaussian variables, then the allocated risk and Protection Levels define stochastic requirements. Begin with the following definitions:

1. An Ideal Trajectory is a trajectory that perfectly tracks the center line of the Virtual Corridor, guaranteeing that the current driving task is accomplished collision-free. (As will be explained more precisely in Chapter 4, this trajectory can be parameterized in time or space)

2. A Planner failure is when any of the lateral, longitudinal, or yaw trajectory errors, $e_{traj,*}$ (where $*$ is used to represent lat, lon, and ψ) exceed their corresponding thresholds: $\epsilon_{traj,*}$. The trajectory error is the difference between the generated trajectory and the Ideal Trajectory. Under a Gaussian assumption: $e_{traj,*} \sim \mathcal{N}(0, \sigma^2_{traj,*})$ (where $\sigma_{\cdot}$ denotes the standard deviation of the quantity specified by "$\cdot$").

3. A Pose failure is when any of the pose errors, $e_{pose,*}$ exceed their corresponding thresholds: $\epsilon_{pose,*}$. The pose error is the difference between where the VDS believes the ego vehicle is and where the ego vehicle truly is in the world. Under a Gaussian assumption: $e_{pose,*} \sim \mathcal{N}(0, \sigma^2_{pose,*})$.

4. A Control failure is when any of the control errors, $e_{ctrl,*}$ exceed their corresponding thresholds $\epsilon_{ctrl,*}$. The control error is the difference between the generated trajectory and where the VDS believes the ego vehicle is in the world. Under a Gaussian assumption: $e_{ctrl,*} \sim \mathcal{N}(0, \sigma^2_{ctrl,*})$.

Before continuing with combining the risk allocation and Protection Levels, it is useful to digress a little on the above definitions. The definition of $e_{ctrl,*}$ allows for its direct measurement. Both the generated trajectory and the estimated pose are available to the controller. Therefore, these quantities can be monitored online. The definition of $e_{pose,*}$ does not share this property because it is computed based on where the VDS believes the ego vehicle is and where the ego vehicle actually is (also known as the ground truth). Without access to the ground truth, $e_{pose,*}$ cannot be measured. Finally, the definition of $e_{traj,*}$ depends on the Ideal Trajectory definition and the Virtual Corridor definition, by extension. Recall that the Virtual Corridor is specified similar to that of a lane and has a direct impact on the Protection Levels. Therefore, a task that

the Planner must complete is to estimate the Virtual Corridor, with its pre-defined specifications. The next task that the Planner must do is to supply a time-parameterization of the Virtual Corridor to generate an estimation of the Ideal Trajectory (Chapter 4 will demonstrate how a trajectory can be converted to a path and vice versa).

The underlying logic for these specific definitions is that they cause the modules' errors to be additive:

$$e_{vds,*} = e_{traj,*} + e_{pose,*} + e_{ctrl,*}$$

Applying the Gaussian assumptions:

$$\sigma^2_{vds,*} = \sigma^2_{traj,*} + \sigma^2_{pose,*} + \sigma^2_{ctrl,*} \tag{2.4}$$

Now what is remaining is to relate the probability of failures per mile to the respective threshold and the standard deviation. For example: what value of $\sigma_{traj,lat}$ satisfies $P_{traj} = P(-\epsilon_{traj,lat} < e_{traj,lat} < \epsilon_{traj,lat})$? We begin by defining the Z-score:

$$Z = \frac{x - \mu}{\sigma} = F^{-1}(P|\mu = 0, \sigma = 1) \tag{2.5}$$

where

$$P = F(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-(t-\mu)^2}{2\sigma^2}} dt$$

Where $\mu = 0$ for all distributions by assumption and x is a sample from the distribution. Note that the second equality in Equation 2.5 uses the fact that the Z score is the same as a sample from the Standard Normal Distribution (which has zero mean and unit standard deviation). $F^{-1}(P_{traj}|\mu = 0, \sigma = 1)$ can be computed using a table of Standard Normal Distribution values or a computer. Solving for σ gives:

$$\sigma = \frac{x}{Z}$$

The z-scores are 5.61, 5.71, 6, and 5.79 for the VDS, Planner, Pose, and Control modules, respectively. These z-scores then relate the Protection Levels to their corresponding standard deviations and thresholds:

$$
\begin{aligned}
\sigma_{vds,*} &= \frac{\delta_*}{5.61} \\
\sigma_{traj,*} &= \frac{\epsilon_{traj,*}}{5.71} \\
\sigma_{pose,*} &= \frac{\epsilon_{pose,*}}{6} \\
\sigma_{ctrl,*} &= \frac{\epsilon_{ctrl,*}}{5.79}
\end{aligned}
\tag{2.6}
$$

Where δ_* represents the Lateral, Longitudinal, and Yaw Protection Levels introduced in Section 2.2: δ_{lat}, δ_{lon}, δ_{ψ}.

Now substitute Equation 2.6 into Equation 2.4 to get:

$$
(\frac{\delta_*}{5.61})^2 = (\frac{\epsilon_{traj,*}}{5.71})^2 + (\frac{\epsilon_{pose,*}}{6})^2 + (\frac{\epsilon_{ctrl,*}}{5.79})^2 \tag{2.7}
$$

Equation 2.7 shows that the required error threshold for the Planner, Pose, and Control modules cannot be designed independently. Continuing with the 2019 Jeep Cherokee, these equations are numerically solved when $\delta_{\psi} = 0.05$ rad, δ_{lat} = 0.65 m, and $\delta_{lon} = 1.32$ m. The resulting surfaces are shown in Figure 2.6.

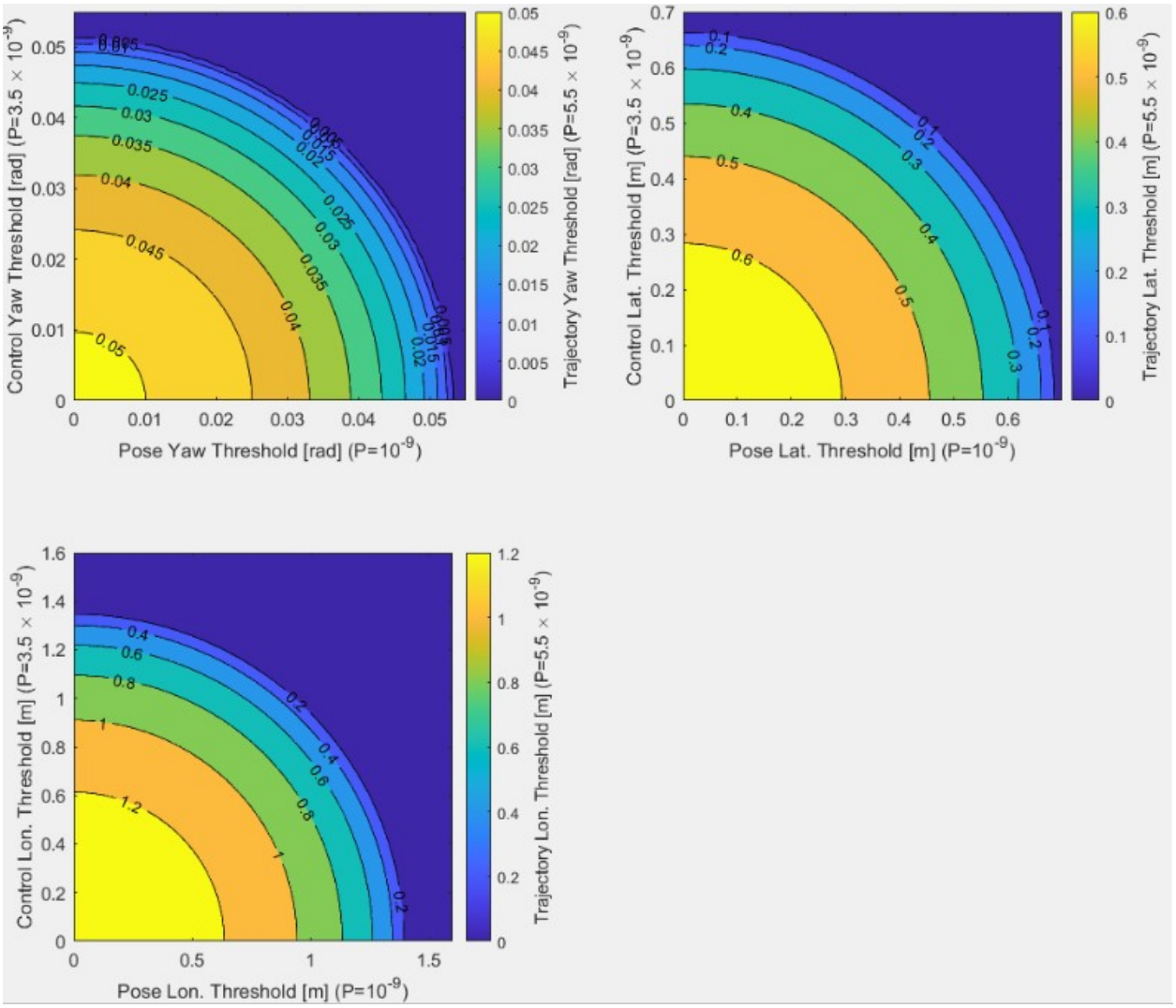


Figure 2.6: The Ellipsoid-like surfaces showing the trade-off between Planner, Pose, and Control module thresholds

The engineer may now choose a point on these surfaces to design the thresholds. When verifying that the module performance meets the desired thresholds, the engineer must acknowledge that these selected thresholds are not deterministic maximum limits. Instead, they describe a distribution at the module's selected risk allocation. Therefore, the engineer may prefer to present the surfaces from Figure 2.6 as Standard Deviations rather than the designed Probability threshold.

Before proceeding with an example to illustrate how this works in practice, it is important to pause and reflect on the critical assumption that $e_{traj,*}$, $e_{pose,*}$, and $e_{ctrl,*}$ are Gaussian and Independent. The Gaussian assumption is made because of its mathematical simplicity and the potential validity of the Central Limit Theorem. The Independent assumption is also made for mathematical convenience. Both of these

assumptions will be validated by the case study in Section 2.4. However, it is important to note that these assumptions are likely dependent on the specific design of each module.

It can also be argued that Independence is not valid when the Gaussian assumption is made. The argument begins with assuming the typical frequency-based robust control design principle of shaping the Sensitivity transfer function, $S(s)$, as a high-pass filter [119, 61, 93]. $S(s)$ is the transfer function from the reference, $r(s) = \hat{r}(s) + e_{ref,*}(s)$, (where $\hat{r}$ is the noise-free reference or Ideal Trajectory and $e_{ref,*}$ is the reference's noise) to the controller's error, $e_{ctrl,*}(s)$. It is also the transfer function from the sensor noise, $e_{pose,*}(s)$, to $e_{ctrl,*}(s)$. Therefore, if both $e_{pose,*}$ and $e_{ref,*}$ are Gaussian, then their frequency content spans the full frequency range. $S(s)$'s shape as a high-pass filter attenuates the low-frequency components of $e_{pose,*}$ and $e_{ref,*}$ while passing the high-frequency components to $e_{ctrl,*}$. The result is that $e_{ctrl,*}$ now has high-frequency errors and is not Gaussian. This argument conceptually shows that the Gaussian and Independence assumptions are flawed. Despite this, these assumptions are useful because they are conservative and provide useful properties as previously described.

In all, the real-world distributions of and relationships between $e_{traj,*}$, $e_{pose,*}$, and $e_{ctrl,*}$ should be explored more deeply. However, this subject is outside the scope of this Thesis and is suggested for further research.

To illustrate how these specifications can be used to verify that requirements are met, the following example is proposed: consider the design of a VDS for a 2019 Jeep Cherokee on a freeway. The lateral error thresholds are set at 0.1 m, 0.3 m, and 0.5727 m for the Pose, Control, and Planner modules (recall that the variables relating to the Planner use the term Trajectory), respectively, by selecting a point on the lateral plot of Figure 2.6. To simulate independent experiments of each module's lateral errors, 100,000 samples are taken from a normal distribution characterized by the module's standard deviations computed using Equation 2.6. These samples are then summed together to compute the experimental distribution of the VDS's lateral position. Separately, the probability density function is computed for each module's distribution as well as the expected VDS distribution. Figure 2.7 shows that the histograms of the distributions fit perfectly with the expected probability density functions. The subfigure shows this more clearly for the VDS. This fit demonstrates the validity of Equations 2.4, 2.6, and 2.7.

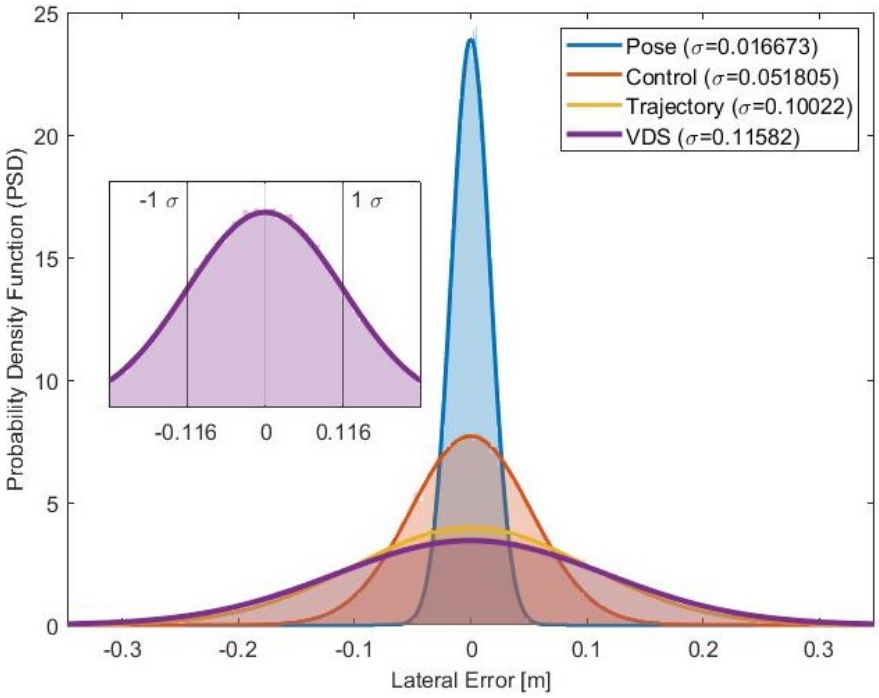


Figure 2.7: The probability density of the Pose, Control, Trajectory, and Virtual Driver System lateral errors.

This example proposes a way to validate the safety of the VDS without full system testing. Rather than requiring millions or billions of miles driven on public roads to prove safety, each module may be independently tested. If the Gaussian assumption is valid, and the distributions of each module's errors satisfy their required characteristics, then the full system is proven safe at the Target Level of Safety. These assumptions must also be checked, which may necessitate some amount of full system testing.

The method developed in this section to compute error thresholds can also be applied to upstream modules. However, the errors need to be defined such that they are additive. This is not trivial. For instance, it is not so obvious how errors should be defined for the Map and Perception outputs.

2.4 A case study showing feasibility

The previous sections developed a new way of simultaneously allocating risk to the Planner, Pose, and Control modules of a Virtual Driver System (VDS). In this section, a case study from the literature is used to demonstrate (1) that this method is feasible, and (2) how to apply this method when there is data to support trade-off decisions.

[102, 104, 180] develop new lateral controllers and validate them on an articulated bus. The bus is 18.3

m long and 2.6 m wide. Its Pose module uses DGPS, INS, and magnetic markers embedded in the road to provide positional feedback to the steering controller. The bus drives on an 1800 m portion of Eugene Oregon's EmX bus route. During this stretch, the bus operates in a special lane for most of the distance. This lane is mostly separated from traffic by curbs, but not continuously for the full route [70]. The route encompasses various "S"-shaped curves where the smallest radius of curvature is 26 m. The speeds at these curvatures are in the range of 6-12 m/s. The road widths range from 3.3 to 3.05 m.

The automatically-steered buses operated for eight months and drove approximately 15,000 miles (estimated from publicly-available bus timetables [106]). [102, 104, 180] suggests that at some points in this portion of the EmX route, the buses were driven manually. The exact amount of miles driven manually is unknown, so it is assumed that half (7,500 miles) were driven manually and the other half were driven automatically.

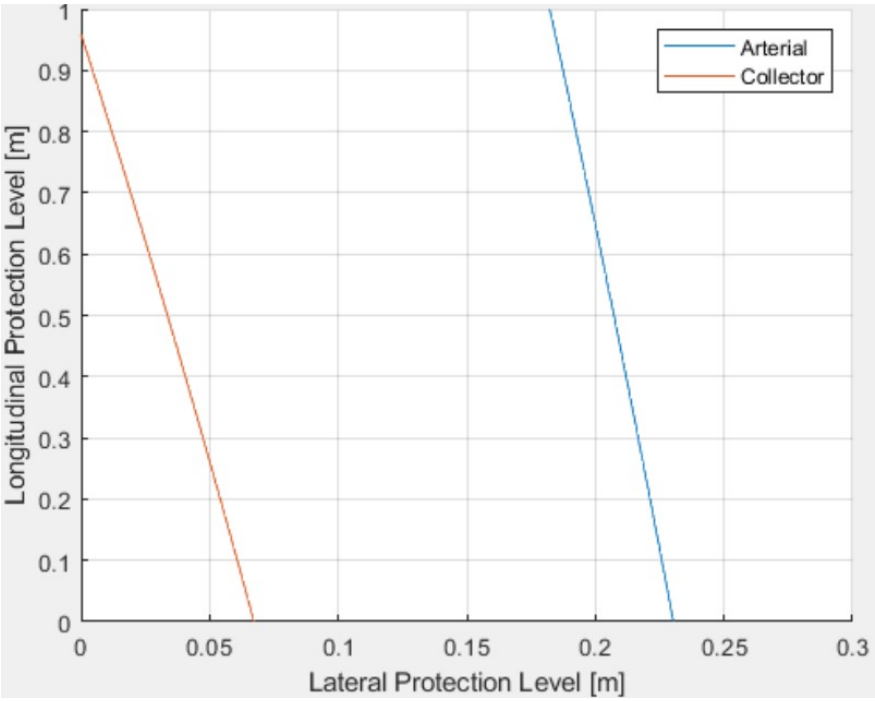


Figure 2.8: Trade-off curve for the Protection Levels of the EmX Bus when $\delta_\psi = 1$ deg

Selecting a vehicle length for this risk allocation is not obvious for an articulated bus because the center joint of the bus allows it to bend around corners. [102, 104, 180] claims that the lane is so narrow at times that the bus must be steered towards the inner curve to prevent the rear wheels from hitting the curbs. Since this indicates that the Alert Limit is not the vehicle's edge but rather the tires, the largest wheelbase will be used instead of the length of the bus. The front wheelbase is 5.8 m and the rear wheelbase is 7.7 m (these do

not add up to the overall vehicle length because the vehicle has front and rear overhangs). The articulation allows for the vehicle to bend around corners, relaxing the required cornering space. Therefore, the modified dimensions of the vehicle are 7.7 m long and 2.6 m wide. The wheelbase is also best modeled by a rectangle rather than a rectangle with rounded corners.

The trade-off curves of the Protection Levels for the articulated bus are shown in Figure 2.8. The Protection Levels are far more stringent than those for the 2019 Jeep Cherokee. When the Yaw Protection Level is 1 degree, the largest Lateral Protection Level is 6.7 cm and 23 cm for the Collector and Arterial roads, respectively. When the Yaw Protection Level is increased more than this, the resulting Lateral Protection Level is too restrictive. When the Yaw Protection Level is decreased, it becomes unrealistic. Therefore, the choice of 1 degree is used to balance these two trade-offs. Unfortunately, [102, 104, 180] does not provide Yaw or Longitudinal tracking data, so the rest of this case study will focus only on the Lateral components.

Using data collected from 8 months of operation, the automated steering system achieved a lateral error standard deviation of 7.15 cm and zero mean. The manually-steered vehicles had a lateral error standard deviation of 16.81 cm. Comparing these performances with the largest Lateral Protection Levels for each road type suggests that the manually steered bus commonly placed the wheelbase outside the lane on Collector roads. This reinforces the argument made in Section 2.3 that there should be an additional ratio to convert failures (lane departures) per mile into collisions per mile.

It is now possible to compute an estimate of this ratio. Begin with computing the number of lane departures that occurred during operation. For simplicity, the conservative assumption is made that all miles are collected on Arterial roads. Converting the Protection Level ($\pm$ 23 cm) into a probability using the manually-steered standard deviation results in 82% of the miles being free of lane departures. Therefore, there should be approximately 1,350 lane departures in the 8 months of data collection. Since [102, 104, 180] do not mention any collisions, and there is no public crash data for those months of operation [153], it is assumed that no collisions occurred. As a result, the articulated bus' estimated lane departure per collision ratio is greater than 1:1,350. The 82% of miles driven collision-free is then used to extrapolate with more data and can provide a better estimate of the lane departure per collision ratio. According to [153] there were 13 collisions (no fatalities) in the location of interest on the EmX route between the years 2008 and 2021. None were attributed to Vehicle System failures. During this period, approximately 302,000 miles were driven (also estimated from timetable data [106]). Therefore, a more accurate ratio would be 54,360 lane departures per 13 collisions (or approximately 4,200 lane departures per collision).

Using this new ratio, the Virtual Driver System's (VDS) probability of failure per mile is:

$$TLS = (P_{veh} + P_{vds})P_{LD:C}P_{C:F}$$

where TLS is the Target Level of Safety, P_{veh} is the probability of the Vehicle System, P_{vds} is the probability of the VDS, $P_{LD:C}$ is the ratio of lane departures (failures) per collision, and $P_{C:F}$ is the ratio of collisions per fatality.

Solving this equation for P_{vds}:

$$2 \times 10^{-10} = (10^{-8} + P_{vds})(\frac{1}{4,200})(10^{-2})$$

$$P_{vds} = 8.4 \times 10^{-5}$$

This probability corresponds to a z-score of 3.76 and can now be used to convert the Protection Levels into the VDS standard deviation, σ_{vds}, using Equation 2.6. Solving Equation 2.6 results in $\sigma_{vds,lat} = 3.56$ cm for Collector roads and $\sigma_{vds,lat} = 12.2$ cm for Arterial roads (recall that Equation 2.6 does not include the halving required because of symmetry). Since [102, 104, 180] present performances as standard deviations, Equation 2.4 will be used rather than Equation 2.7 to compute $\sigma_{ctrl,lat}$. Solving Equation 2.4 requires the other standard deviations. This information can be extracted from the description of the VDS reference path system in [186] and from the VDS position estimation system in [40]. The reference path is a series of magnets embedded into the road within 15mm of the road center-line [186]. This will be used as the value for $\sigma_{traj,lat}$. The magnetic reference system allows the VDS to localize itself within 5mm to 3cm of the ground truth [40]. The Pose module's performance is therefore set at $\sigma_{pose,lat} = 3$ cm. Substituting these values into Equation 2.4 results in $\sigma_{ctrl,lat} = 1.2$ cm for Collector roads and $\sigma_{ctrl,lat} = 11.7$ cm for Arterial roads.

Comparing the target standard deviations: (Collector) $\sigma_{ctrl,lat} = 1.2$ cm, (Arterial) $\sigma_{ctrl,lat} = 11.7$ cm; with the experimental standard deviation: $\sigma_{ctrl,lat} = 7.15$ cm, shows that the Target Level of Safety is met for Arterial roads, but not for Collector roads. This comparison shows that this risk allocation is feasible because it is partially satisfied by an existing Public road deployment of a VDS. However, the fact that the Collector road requirements are not satisfied suggests that either (1) the automatic steering system was not operated long enough to observe collision data or (2) this approach can be overly sensitive to road

specifications.

2.5 Conclusion

This Chapter begins with a brief introduction to Alert Limits and Protection Levels in Section 2.2. In the same section, a rectangular model is used to compute the Protection Levels for a Jeep Cherokee. The next section follows [162] partially. Two major differences are argued: (1) there should be an additional conversion factor between lane departures per mile and collisions per mile, and (2) the Protection Levels should be applied to the VDS rather than to the Localization module. By imposing the Protection Levels on the VDS, definitions of failure can be set for each module such that their errors are additive. This allows for a simple approach to allocating risk to the three modules simultaneously.

In Section 2.4, an articulated bus performing lane-keeping in a narrow lane is used to demonstrate this method's feasibility. This case study further demonstrates the risk allocation method developed in Sections 2.2 and 2.3. It also provides data to support the argument that an additional conversion factor should be used in computing the Target Level of Safety.

Most importantly, the case study in Section 2.4 demonstrates that this method is realistic in application and can be verified with experimental data. This verification shows that the system developed in [102, 104, 180] meets requirements for Arterial roads, but not for Collector roads. Furthermore, this method argues that had the ADS developed in [102, 104, 180] continued to operate on Arterial roads, it would have achieved the Target Level of Safety; approximately 200 times better than a human driver [162].

Throughout the remainder of this Thesis, the content presented in this Chapter will not be used very extensively. This is mainly because such a methodology is not necessary for simulation for which the ground truth is always known. Furthermore, this Thesis uses a Jeep Grand Cherokee to perform real-world experiments for different control methodologies. The goal for the vehicle is not for it to achieve some target level of safety. So this Chapter's developments are largely not applied. However, during some experimentation, the Planner module could not reliably produce reasonable references. This problem was largely solved by using the geometry and definitions presented in this Chapter to implement layers of safety checks on the vehicle. This is discussed in more detail in Chapter 7.

Chapter 3

Parameter Identification and Modeling for Lateral Motion Control

This chapter develops a simulator that will be used throughout this Thesis to investigate lateral motion control in a realistic, safe (by not risking real-world challenges), and repeatable way. The important aspects of driving in realistic Operational Design Domains are modeled including road friction and vertical road noise. This chapter then introduces the dynamic, lateral single-track vehicle model (commonly known as the bicycle car model) that is used extensively throughout this dissertation. A nonlinear grey-box identification scheme is used to identify the parameters of this model. This strategy is applied first to a simulated dataset. The identified model is compared to the CarSim model using tests defined in ISO standards

. The strategy is then applied to identify the parameters of a small-scale vehicle where input-output delays are significant, friction is low, and there are unknown actuator dynamics. Finally, the strategy is applied to a 2019 Jeep Grand Cherokee.

3.1 Introduction

This Thesis requires testing controllers on a vehicle under adverse weather conditions. Few companies, let alone University laboratories, have the resources to perform this safely and repeatably on a full-scale passenger vehicle. Several solutions have been proposed using small-scale cars [140, 30, 29, 31] to mitigate the required resources. Some have even suggested using scaled vehicles for studying drifting [161]. Each of these techniques shows significant promise. However, these systems are typically designed for indoor operations and struggle to model a car driving in snow or rainy conditions. Additionally, because of their

small scale, it is common to use sensor configurations, computing systems, and hardware that would not be used on a full-scale car. This is not to say that these limitations could not be overcome. However, it points out that there may exist a solution alternative to using a scaled testbed to study the control of vehicles in adverse weather conditions that better balance safety, repeatability, validity, and human resources. To this end, computer simulation is proposed in this Thesis to satisfy all requirements.

The development of such a simulator is explained in Section 3.2. It begins with selecting an appropriate software to simulate the vehicle dynamics. Next, models are introduced to simulate sensors and localization algorithms. Then, additional features are described to simulate environmental conditions such as vertical road noise, wind gusts, and road friction. The section also proposes a method to design realistic maneuvers for controller testing. Finally, all of these effects are categorized into definitions of several Operational Design Domains (ODDs).

Having defined a high-fidelity simulator capable of capturing some of the most important phenomena observed when driving in adverse weather, models must now be developed to design controllers. The models described in Section 3.2 are too complex to facilitate the design of controllers. Instead, two low-order models are proposed for control design: the kinematic bicycle car model [160, 143] and the dynamic bicycle car model [160, 154, 75]. In Subsection 3.3.4 both of these models are fit to experimental data collected on the AVA Project's Jeep Grand Cherokee. Model fit results show that the dynamic bicycle car model is better suited to modeling the system than the kinematic bicycle car model. In Subsection 3.3.5, the dynamic bicycle car model parameters will be identified from simulation experiments of a CarSim vehicle similar to the AVA Project's Jeep Grand Cherokee.

Finally, Section 3.4 summarizes the simulator and the control-oriented modeling and system identification developed in this Chapter. Additional work that could enrich the developments of this Chapter is also proposed.

3.2 Simulator Design

By selecting computer simulation, financial and time costs are greatly reduced compared to physical experiments. Furthermore, repeatability often comes with little additional effort (in stochastic studies it can be as simple as setting a seed value for a random number generator). The challenge to overcome is validity. The growth in computer simulation in the last several decades has resulted in a wealth of high-fidelity com-

puter simulation tools for engineers. To simulate the complex multi-body dynamics of a full-size vehicle there is commercial software such as MSC Adams/Car [4], Mechanical Simulation's CarSim [38], and IPG Automotive's CarMaker [39]. Several open-source simulation tools also exist that provide varying levels of vehicle dynamics fidelity. The most mature and well-resourced is likely Project Chrono [148], which provides a full multi-physics simulation engine and several Application Programming Interfaces to define different vehicle models. The challenge with using Project Chrono is the lack of existing vehicle models with published empirical validation studies. Therefore, to simulate vehicle dynamics with reliable results, this Thesis will use CarSim.

However, there is much more than just the vehicle to model when simulating Automated Driving Systems (ADS) in adverse weather conditions. A full system architecture will have to be selected and constructed in simulation. These additional simulation features are implemented in Matlab/Simulink (version R2019b) since it interfaces well with CarSim. The simulator architecture shown in Figure 3.1 is motivated by the Automated Vehicles for All (AVA) Project's 2019 Jeep Grand Cherokee [22]. This project, funded by the Department of Transportation, is a collaboration between Texas A & M University, the University of Illinois Urbana-Champaign, George Washington University, and the University of California-Davis. This project's goal is to explore ADS technology for rural America. The AVA Project has selected a retrofitted 2019 Jeep Grand Cherokee from AutonomousStuff (A Hexagon AB company) that supports some ADS research [21].

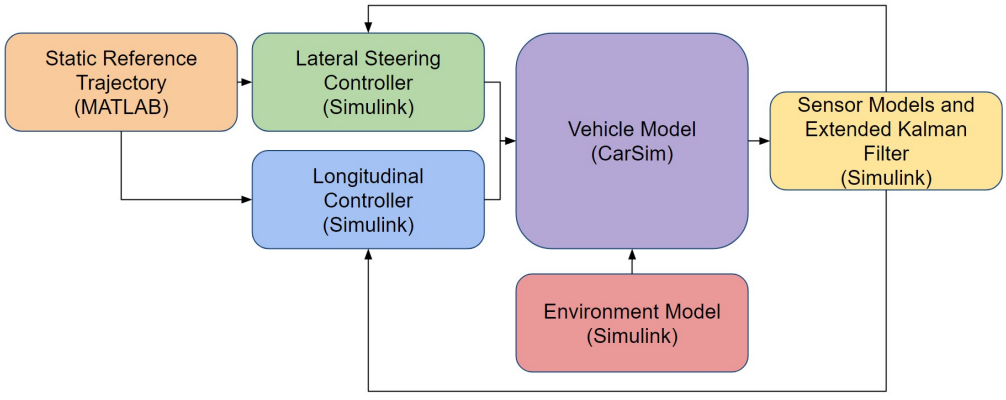


Figure 3.1: High-level overview of simulator architecture.

The system architecture for the Jeep is largely software-defined and is, therefore, flexible. However, several devices have been added to the Jeep to enable ADS functionality. An Inertial Navigation System (INS) and Global Navigation Satellite System (GNSS) device called AsteRx-i3 Pro+ from Septentrio [14] is

installed to provide the Virtual Driver System (VDS) with estimates of the ego vehicle's pose. This device fuses Real-Time Kinematic Global Positioning System (RTK-GPS) and Inertial Measurement Unit (IMU) measurements to provide centimeter-level positioning accuracy. Actuation is provided by a drive-by-wire module from New Eagle [150]. This device exposes an interface to command several actuators such as throttle, brake, and steering.

The selection of the vehicle to model has a significant impact on the results of this study. Many controllers available in the literature have already been demonstrated on high-performance sedan-like vehicles [74, 121, 122, 135]. Furthermore, according to data collected on vehicle model sales in the US in 2023, [173] 15 of the top 20 best-selling vehicle models are either pickup trucks or SUVs (including smaller hatchbacks). Only 5 of the top 20 best-selling vehicle models are sedans. When totaling the number of sales for the top 20 sold models [173], sedans account for approximately less than 20%. Furthermore, the current vehicle models operated by robo-taxi companies (Waymo, Cruise, and Motional) are SUV vehicles (Jaguar I-Pace, Chevy Bolt, and Hyundai Ioniq 5, respectively). Therefore, by simulating an SUV, this Thesis (1) differentiates itself from what has been done before, (2) uses the most popular vehicle type sold in the US, and (3) is very practical and useful for the ADS industry.

Finally, before moving on, it is worth noting that selecting an SUV model makes the control task more challenging. The performance requirements are tighter since the vehicle is larger. Furthermore, it is generally true that SUVs are less maneuverable than smaller sedans.

The vehicle model in Figure 3.1 models an SUV using the commercial software CarSim 2020. This vehicle is a full-size SUV with Electronic Stability Control and Antilock Braking System. CarSim provides the user with an interface to select parameters for a vehicle. The user may select from default models (as is done in this case) to build a vehicle model. The interface breaks down modeling a vehicle into vehicle body, powertrain, brake, steering, front suspension, and rear suspension parameters and configurations. The vehicle body parameters are detailed in Table 3.1.

Parameter	Value
C.G. Height from ground (mm)	781
C.G. Offset from Front Axle (mm)	1330
C.G. Lateral Offset from Centerline (mm)	0
Wheelbase (mm)	3140
Sprung Mass (kg)	2257
Roll Inertia (kg-m^2)	846.6
Pitch Inertia (kg-m^2)	3524.9
Yaw Inertia (kg-m^2)	3524.8

Table 3.1: CarSim Vehicle Body Parameters.

The powertrain is composed of the engine, a torque converter, a 7-speed transmission, a transfer case between front and rear differentials, and a front and rear differential. The engine is selected to be a 250 kW engine, which is implemented as a nonlinear, 3-dimensional lookup table (linear interpolation) between engine speed, engine torque, and throttle position. The torque converter models the input and output torque and rotation speeds as lookup tables. The inputs are speed ratio (output / input) and the outputs are torque ratio and inverse capacity factor $\left(\frac{\sqrt{\text{(N-m)}}}{\text{rpm}}\right)$. The transfer case models viscous friction as a nonlinear lookup table. The powersplit is 50% for the front and 50% for the rear. The difference between the front and rear axle speeds is the input to the lookup table, which outputs a torque to the front and rear axles. The torsional stiffness is 80 N-m/deg, and the torsional damping is 0.8 N-m-s/deg. The front and rear axles also model viscous friction as a nonlinear lookup table. The associated parameters are detailed in Table 3.2.

Parameter	Front Axle Value	Rear Axle Value
Stiffness (N-m/deg)	100	80
Damping (N-m-s/deg)	1	0.8
Gear ratio	2.65	2.65

Table 3.2: CarSim Front and Rear Axle Parameters

The brake system is configured to use the "Control with master cylinder pressure" option. No transport delay is modeled. The front and rear actuator time constants are set at 0.06 s. The built-in ABS control is configured to use the "Two-channel front [rear] ABS", respectively. The steering system is configured to use the "rack assist rack and pinion" with a C factor of 40 mm/rev. The power steering function is modeled as the a nonlinear lookup table that inputs a steering wheel torque and outputs a rack motor boost torque. Additional parameters are presented in Tables 3.3 and 3.4.

Parameter	Value
Column Inertia (kg-m^2)	0.02
System Inertia (kg-m^2)	0.005
Column Damping (N-m-s/deg)	0.02
Column Hysteresis (N-m)	0.1
Hysteresis ref. angle	0.5

Table 3.3: CarSim Steering Column Properties

Parameter	Value
Lateral offset at center (mm)	39.5
Kingpin inclination (deg)	8.0
X coord. of kinpin at center (mm)	-1.0
caster angle (deg)	3.5

Table 3.4: Kingpin Geometry (same left and right values)

The front suspension is configured to use independent suspension. The parameters are presented in Table 3.2. The dampers are modeled with a nonlinear lookup table mapping spring displacement rate to force. The rear suspension is configured to use a rigid axle suspension. The parameters are presented in Table 3.6. The springs and dampers are modeled with nonlinear lookup tables.

Parameter	Value
Unsteered unsprung mass (kg)	21.01
Steered unsprung mass (kg)	48.69
Spin inertia (kg-m^2)	3.88
XX/ZZ inertia (kg-m^2)	1.96
Distance between wheel centers (mm)	1725
Spring friction (N)	20
Linear spring rate (N/mm)	189
Max rebound stop (mm)	150
Min rebound stop (mm)	-110

Table 3.5: CarSim Front Independent Suspension Parameters

Parameter	Value
Steered unsprung mass (kg)	0
Spin inertia (kg-m^2)	4.29
XX/ZZ inertia (kg-m^2)	2.17
Axle mass (kg)	172
Axle roll and yaw inertia (kg-m^2)	44

Table 3.6: CarSim Rear Rigid Axle Suspension Parameters

Finally, the tires are modeled using CarSim's "Internal Table Model with Simple Camber" with the

parameters for a 265/70 R17 tire. The parameters are presented in Table 3.7. The longitudinal force, lateral force, aligning moment, camber thrust, and relaxation length are all modeled with 3-dimensional nonlinear lookup tables.

Parameter	Value
Reference vertical force (N)	11500
Effective rolling radius (mm)	368
Unloaded radius (mm)	395
Spring rate (N/mm)	440
Mass (kg)	31
Spin moment of inertia (kg-m^2)	3.1
Yaw/roll moment of inertia (kg-m^2)	1.8

Table 3.7: CarSim Internal Table Model Parameters for 265/70 R17

The inputs to the CarSim model are steering wheel torque, throttle percentage, and brake pressure. There are two alternative modes of steering operation: (1) steering wheel torque control and (2) steering rack torque control. The first mode expects Simulink to send a steering wheel torque to CarSim and an additional boost torque is applied from a motor on the steering rack. To allow the lateral motion controller to command a steering wheel angle (or road wheel angle converted statically to a steering wheel angle) a PID controller is developed to track a reference steering wheel angle with a steering wheel torque. This is the interface used for all simulations in this Thesis except for the results in Chapter 7 (which compares the different modes).

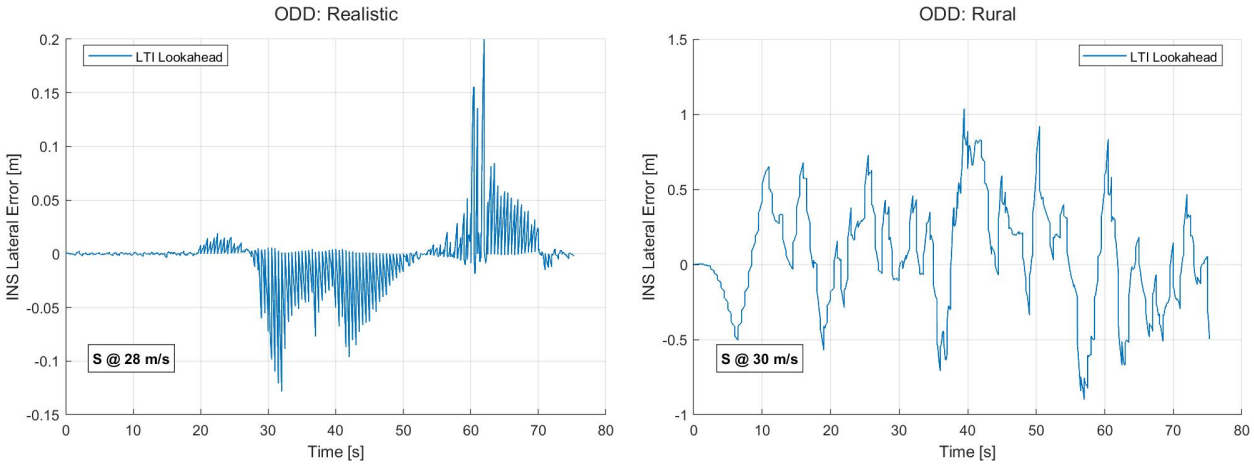
The longitudinal motion controller is simply a PID controller implemented in Simulink to track a desired velocity using a combination of throttle percentage and brake pressure. To separate the two commands, the output of the second PID controller is split into positive and negative components. When positive, throttle percentage is commanded. When negative, brake pressure is commanded.

The CarSim model also supports road friction inputs for each tire-road interface, vertical displacement for each tire, and external wind magnitude and direction. These inputs constitute the Environment Model's interface with the vehicle model. These Environment Models will be explained further in Section 3.2.2.

In addition to the vehicle and environmental models, sensor models and a sensor fusion algorithm are implemented. This is shown in Figure 3.1 as consuming the output of the vehicle model. The three sensors that are modeled are the Inertial Measurement Unit (IMU), Wheel Speed Sensor (WSS), and the Global Positioning System (GPS). The IMU provides accelerations and angular rotations sampled at 200 Hz. It is

simulated using Matlab/Simulink's IMU model [105]. The WSS provides longitudinal velocity sampled at 200 Hz. It is simulated by adding Gaussian noise to the vehicle's longitudinal velocity and quantizing the signal. The GPS provides position, orientation, and velocity in the global coordinate frame sampled at 2 Hz. It is simulated using Matlab/Simulink GPS model [78]. The Extended Kalman Filter (EKF) proposed in [193] fuses these three signals to improve localization sampled at 200 Hz. The algorithm developed in [193] fuses more sensors than those modeled here, so those components of the update steps are not used.

The simulation supports two different modes of operation for the sensor models and EKF: RTK-GPS and Differential GPS (DGPS). The parameters of the sensor models and EKF are tuned to match the RTK INS performance and the DGPS INS performance of the AsteRx-i3 D Pro + from Septentrio [14]. In RTK mode, the EKF estimates the global positions approximately 6-15 cm within the ground truth (root mean square). In DGPS mode, the EKF provides global position accuracies between 10-40 cm (root mean square). The lateral error for the RTK mode is shown in Figure 3.2a, and the lateral error for the DGPS mode is shown in Figure 3.2b. To produce these results a Linear Time-Invariant (LTI) controller is tested on an "S" shaped maneuver in two Operational Design Domains (ODDs). The controller will be described in Chapter 4. The maneuver will be described in Section 3.2.1. The ODDs will be explained later in Section 3.2.2.



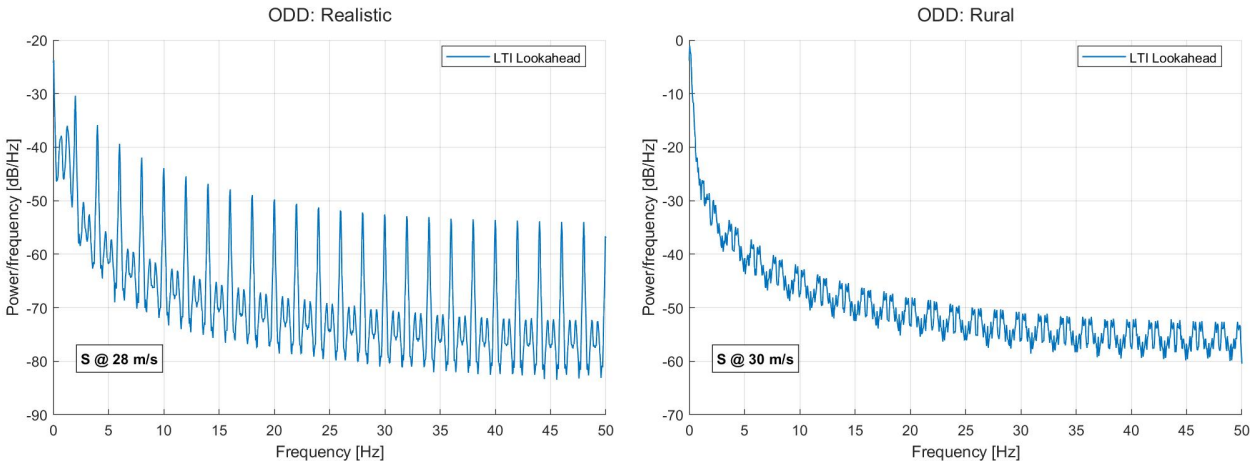
(a) Lateral position error in RTK-GPS mode

(b) Lateral position error in DGPS mode

Figure 3.2: Lateral position error of both sensor fusion modes

Before moving on from the sensor models, it is worth noting a phenomenon that is also modeled: jumps in position estimates. The particle mass model uses the IMU and WSS measurements (assumed to be longitudinal velocity measurements) as inputs to estimate position sampled at 200 Hz. This is the so-called model update step. The noise in each sensor is integrated in this step such that there is drift in

the position update. Then, at 2 Hz, the GPS provides high-precision position information in the so-called measurement update step. Tuning the gains of the EKF is a balance between trusting the measurement and the model estimations. If the model updates are trusted too much, the drift in the model updates can be large. Then, when the measurement update step occurs, the estimate jumps to a more accurate estimate. If the model updates are trusted too little, the estimation quality deteriorates until a measurement update occurs. Therefore, a balance is struck in the EKF tuning between smoothness and variance. In the RTK configuration, the jumps are more prominent because the RTK GPS is trusted more than the model-update steps. In the DGPS configuration, a smoother, but less accurate, estimate is achieved. This can be perceived in another way as concentrating the Power Spectrum Density (PSD) of the estimation error at 2 Hz for the RTK mode. By contrast, while the DGPS mode has larger overall power, its PSD is more evenly distributed. This is shown in Figures 3.3a and 3.3b. Figure 3.2a shows harmonics every 2 Hz which compose the step changes in RTK measurement update steps.



(a) Welch power spectral density estimate for lateral position error in RTK-GPS mode

(b) Welch power spectral density estimate for lateral position error in DGPS mode

Figure 3.3: Welch power spectral density estimates for lateral position error of both sensor fusion modes

In addition to modeling the sensors, the output of the EKF is delayed by a value that is sampled from a Gaussian distribution. This is to model the delay associated with the processing time of various sensor information, which varies stochastically. With a proper estimate of the mean of this delay, its effects can be mitigated by forward extrapolating the signals, as is done in [193]. However, to keep the control design realistic, the delay is assumed unknown and not accounted for in the design of the lateral or longitudinal controllers.

As Figure 3.1 shows, the control architecture uses decoupled lateral and longitudinal controllers. Typical Automated Driving System software architectures require perception modules and planner modules [107, 49, 46, 134, 103]. However, since neither of these modules is the subject of our current study, their contribution is eliminated by assuming static trajectories. These trajectories are provided to both the longitudinal and lateral controllers. The full path (the trajectory without the time parameter) is provided to the lateral controller. The distance along the path and the time associated with each waypoint are given to the longitudinal controller. In this way, the longitudinal controller's task is to position the vehicle along the path exactly where the trajectory desires. The lateral controller's task is to keep the vehicle as close to the path as possible.

This separation of lateral and longitudinal dynamics has significant impacts on the overall control of the system, especially when operating in the nonlinear tire region, which is more easily reached on low friction surfaces. All of the controllers investigated in this Thesis control only lateral and yaw vehicle dynamics. This choice is motivated by the successes realized in [122, 76, 176]. These works present evidence suggesting that high lateral motion control performance can be achieved using such a decoupled architecture in a wide range of tasks and ODDs. In particular, [76, 176] demonstrates steering controllers that sustain large body-slip angles when drifting, good performance when driving at the limits of handling, and achieve good performance in reduced road friction.

3.2.1 Reference Generation

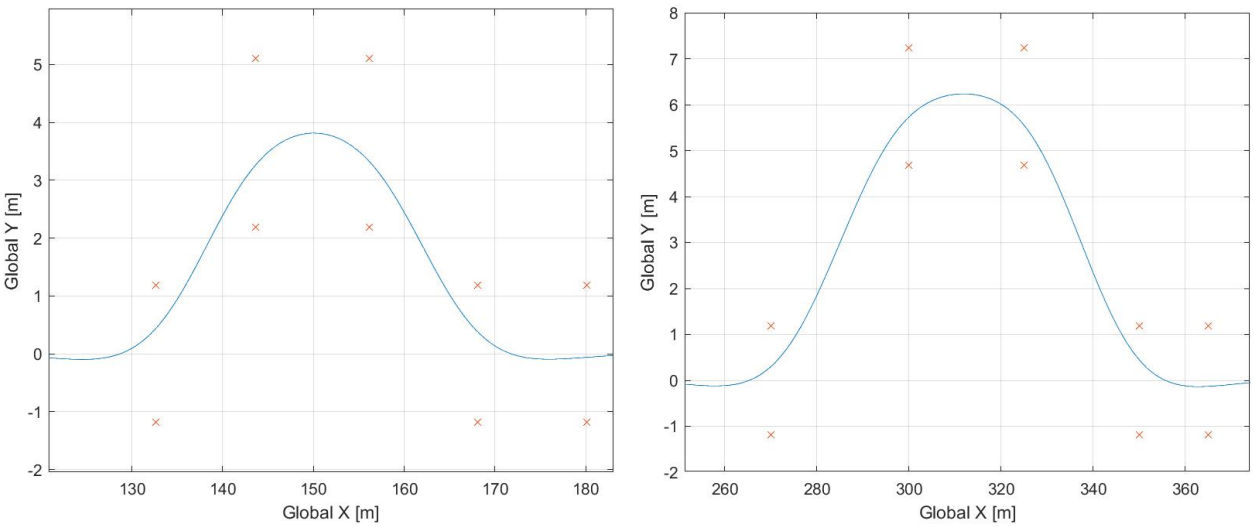
The goal of this Thesis is to experimentally evaluate lateral motion controllers in rural roads and highways across a variety of environmental conditions. This necessitates the generation of references that represent driving on typical highways and rural roads. However, driving on typical roads includes a variety of maneuvers that must be safely completed such as lane keeping, lane changing, overtaking, and collision avoidance.

Lane-keeping maneuvers are captured in this work by using two existing sections of highways in California: CA-17 and I-15; both of these are collected from OpenStreetMap [53]. These sections are selected because they are the locations of a large number of vehicle collisions every year in California. To capture lane-changing and overtaking maneuvers another trajectory, called “S Road", is manually created to present higher-than-normal curvatures. This maneuver is generated by first creating a high-curvature road. Then, an overtaking maneuver is manually created on this high-curvature road. The result is an aggressive highway maneuver.

Collision avoidance maneuvers are quite different from these three trajectories. Collision avoidance maneuvers are typically composed of a series of very high, but short-duration, peaks in road curvature. The Single Lane Change (SLC) and the Double Lane Change (DLC) maneuvers are commonly used in the literature to test the performance of steering controllers [41, 46, 164]. Both names come from the number of lanes crossed during an abrupt overtaking maneuver. They are specified by ISO 3888 [113]. While the design of the single and double-lane changes is motivated by actual driving, it is not clear how well they predict steering controller performance on actual highways. The inclusion of actual highway trajectories allows for the direct study of this question.

ISO 3888 specifies that the SLC and DLC tests are to be conducted with an expert driver, whose task is to enter the test at a high speed and keep the vehicle within a bounded area. When this test is performed by an ADS, the local planner must determine a path that keeps the vehicle within the bounded area. Because this investigation is not focused on trajectory generation or optimization, the trajectory is generated offline and held static for all controllers and ODDs.

The trajectories are generated in two steps: (1) the generation of a two-dimensional, C^2 continuous path, and (2) the assignment of a velocity profile to the path. A C^2 continuous path is one whose zeroth, first, and second derivatives are continuous. This provides a smooth reference that can be followed with low error. To generate the path, a cubic spline, parameterized by the cumulative chord length [68], interpolates the waypoints extracted from the OpenStreetMap data. This allows for any amount of spatial sampling of the path (i.e., the path is composed of waypoints that are spaced 0.1 m from each other along the path). For the collision avoidance paths, the waypoints are initially centered between each cone in the ISO specification. Then, a cubic smoothing spline interpolation algorithm [57] is used to adjust the waypoints so that they generate a smoother path. This step is iterated by increasing the desired smoothness of the spline until the path is approximately the vehicle's track width away from the nearest cone. Finally, the same method used for the highway maneuvers is applied to generate the final trajectory. The SLC and DLC paths are shown in Figures 3.4a and 3.4b. The crosses indicate the cone locations specified by the associated standard.



(a) Top-down view of the single lane change path.

(b) Top-down view of the double lane change path.

Figure 3.4: Collision avoidance paths.

The specifications of the five trajectories used in this study are listed in Table 3.8.

Name	Maximum Absolute Curvature [1/m]	Average Absolute Curvature [1/m]	Path Length [m]
Single lane change (SLC)	0.033	0.006	210 *
Double lane change (DLC)	0.015	0.002	424 *
CA-17	0.007	0.001	13,825
I-15	0.003	6×10^{-4}	9,924
S Road	0.008	0.003	1,609

Table 3.8: Trajectory specifications.
(*: The straight portion of the route adjusts to provide enough space to get up to speed.)

Dynamic Feasibility

The modular architecture design for complex systems such as the one for Automated Driving Systems shown in Figure 2.1 provides several benefits. However, these benefits come at the price of having to carefully define the requirements and interfaces of each module. One particularly difficult interface to design is the one between the Planner and Controller. This is because the combined system must achieve a target level of safety subject to constraints imposed by the vehicle dynamics and environmental conditions. This Thesis requires the Planner to generate references that satisfy these constraints and the Controller to achieve safe performance. References that satisfy these constraints are called *feasible* references.

A classical system theoretic approach is to consider the vehicle and its environment as a function relating

some state x with its derivative and some system inputs: $\dot{x} = f(x, u)$. A final state x_f is called *reachable* (specifically, locally reachable) if x can be driven from an initial state x_i to x_f in finite time. Now, suppose that there are constraints on u: $\underline{u} \leq u \leq \bar{u}$. If x_f can be reached with a $u(t)$ that satisfies these constraints, then x_f is said to be reachable and feasible. Finally, suppose that we wish to transition from x_0 to x_f with a duration less than t. A trajectory of states $x(t)$ is *dynamically feasible* if all $x \in x(t)$ are reachable and feasible, and the durations required to transition between each state are equal to the durations between time parameterizations of each state.

The definition of dynamic feasibility imposes an equality constraint between the time it takes to transition between each state and the trajectory's duration between each state. This equality constraint is highly conservative. To show this, consider two dynamical systems: $\dot{x}_1 = f_1(x_1, u_1)$ and $\dot{x}_2 = f_2(x_2, u_2)$. A reference state trajectory $x_r(t)$ is generated by applying the input trajectory $u_1(t)$ to the first system. If $f_2 = f_1$ and $x_1(0) = x_2(0)$, then $x_r(t)$ can be tracked perfectly by applying $u_2(t) = u_1(t)$ to the system described by f_2. However, if $x_1(0) \neq x_2(0)$, then $x_r(t)$ is not dynamically feasible. The fragility of this definition limits its use, so we will relax the definition by allowing the term to also refer to control error, $e(t) = x_r(t) - x_2(t)$. If $x_r(t)$ can be followed by the second system such that $|e(t)| \leq \epsilon$, then the state reference trajectory is called dynamically feasible.

The key aspect of this definition is now ϵ. An added benefit of this definition is that $x_r(t)$ can be dynamically feasible if $f_2 \neq f_1$. In other words, a system different from the plant can be used to generate a dynamically feasible trajectory. To illustrate this concept consider the hypothetical scenario of two cars on a race track. The first car is a Formula One race car, and the second car is an SUV. The first car's state is recorded as it gently performs its first lap. However, on the second lap, the race car is driven aggressively to achieve the lowest lap time. The driver of the SUV is now tasked with following the recorded race car states for both laps. We might expect the SUV driver to track well during the first lap. However, on the second lap, we would expect the driver to track very poorly. In this scenario, we might say that the first lap is more dynamically feasible for the SUV and its driver than the second lap.

Another example is to consider two vehicle models parameterized by an understeer gradient. The first model is used in the planner to generate references. The second model is used in the controller to track the references. The more the understeer gradients of the planner and control models diverge, the less dynamically feasible the trajectory. This example shows that parameter accuracy influences dynamic feasibility. An open question to investigate is how much model accuracy affects dynamic feasibility and ultimately control

performance.

The Planner problem is now the task of generating the most dynamically feasible trajectory for the vehicle. More precisely, the Planner must minimize $e(t)$. Such a task is well positioned to be solved with optimal control techniques [77].

The trajectories in Table 3.8 are generated using the chord-length-parameterized cubic spline model. This model does not capture the vehicle dynamics. It simply approximates the vehicle as a particle and its state constraints are imposed by cubic spline constraints. Therefore, it is expected that the generated trajectories are not very dynamically feasible. This is made worse when the road conditions vary because the velocity profile is generated on assumptions of maximum lateral and longitudinal accelerations.

One option to address the problem of decreased dynamic feasibility when the road conditions are adverse (such as low friction) is to modify the maximum lateral and longitudinal accelerations. Essentially, adapt (offline) the acceleration constraints to the tire-road interface.

To determine how much to lower the maximum velocity of the maneuvers, we can derive the relationship between the vehicle's longitudinal velocity, U_x, and the road-tire static friction coefficient, μ, as is shown in Equation 3.1. This relationship attempts to make the references more dynamically feasible even when the road friction is diminished.

$$\begin{aligned} F_y &= ma_y \\ F_y &= mg\mu \\ a_y &= \kappa U_x^2 \\ U_x &= \sqrt{\frac{g\mu}{\kappa}} \end{aligned} \tag{3.1}$$

where κ is the road's curvature and g is Earth's gravity constant. This relationship can now be used to determine a safe velocity for each maneuver at different values of μ.

It is important to note that this methodology is based on the idea of pulling out known nonlinearities and constraints and placing them upstream of the controller. In other words, the planner has been made more complex because it is now responsible for generating more dynamically-feasible maneuvers. In this Thesis, this increase in complexity is acceptable because the Planner is simply the generation of a static reference. However, in practice, where the Planner is responsible for generating a path that adapts to the changing

road environment, this increase in complexity may not be tolerable. This is largely an architectural design decision that does not appear to be well-studied in the literature.

More advanced methods such as reference governors [129] and Model Predictive Control can be used to modify or generate trajectories that are more dynamically feasible. If the references are made more dynamically feasible than the references used in this Thesis, then it is likely true that the lateral motion control performance will improve (by definition of dynamic feasibility). Therefore, the experimental results (both simulation and real-world) that are presented throughout this Thesis (the adverse weather conditions and aggressive maneuvers in particular) can likely be improved upon with more advanced Planner algorithms. However, further investigation is warranted because there is still an open debate over how to address dynamic feasibility to achieve tight requirements on computing resources, calibration effort, safety, and comfort.

Reference Extraction

Once the Control module receives the reference trajectory, there are several ways to pass the references to the lateral and longitudinal controllers. One way to provide references to the lateral controller is to discard the temporal information and parameterize the trajectory in space. For instance, we might change $x_r(t)$ to $x_r(s)$, where s is the distance along the path. However, now the challenge is to determine which point on the path to use as a reference. This is solved with a closest-point algorithm. At every time step, the following equation is solved $x_r(t) = \min_{x_r(s)} |x_1(t) - x_r(s)|_2$ (where $x_1(t)$ is the current vehicle's state and $x_r(s)$ is the reference path). This comes at the cost of additional computing and, more significantly, introduces the pose estimator's dynamics and noise into the reference. This last cost occurs because $x_1(t)$ is measured with sensors that contain noise. So when performed on a real vehicle, the reference extraction is given by: $x_r(t) = \min_{x_r(s)} |(x_1(t) + n_1(t)) - x_r(s)|_2$ (where $n_1(t)$ is the measurement noise). If $n_1(t)$ has spectral power in the same frequency range as the controller, no feedback controllers can mitigate this noise because it is in the reference, not the feedback. This discussion of reference extraction is more deeply explored in Chapter 4.

Further expanding on the lateral controller, once the steering angle is computed, it is typically passed to a low-level steering controller, sometimes referred to as a smart actuator. This is the motivation for the PID controller developed in Simulink to track a desired steer angle by providing a torque request to CarSim 2020. While some lateral controllers consider steering actuator dynamics [55], we assume the dynamics are unknown. The reason for this is that it is often the case that Automated Driving Systems are developed on an

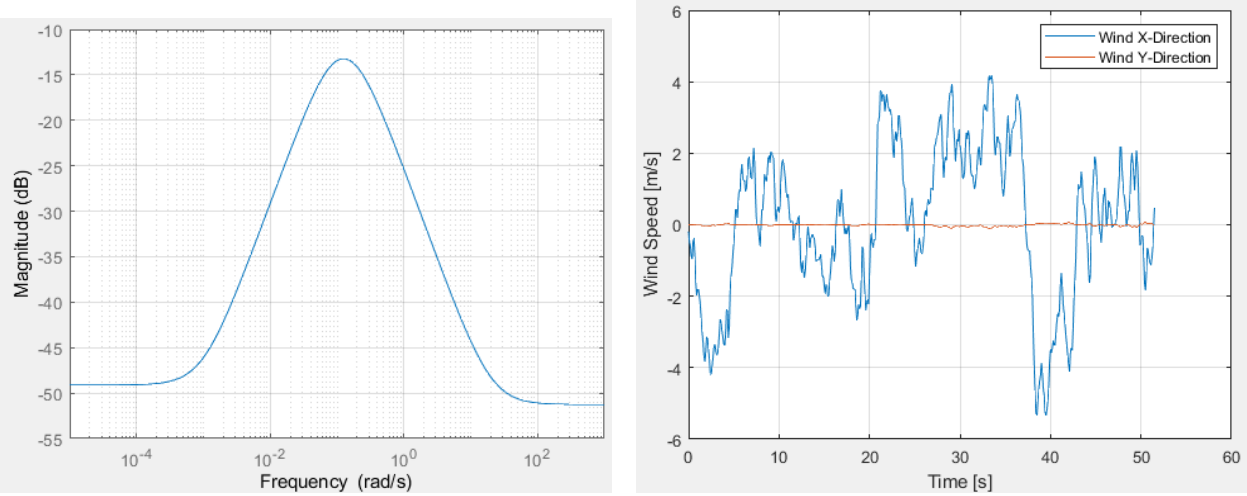
existing vehicle with its own low-level steering actuator controllers. Depending on the Original Equipment Manufacturer, it can be difficult to obtain information on these controllers or to model them accurately. To account for this, each controller is tuned to account for actuator constraints indirectly. This is part of the reason for the hyperparameter tuning of each controller. This more closely resembles a realistic control development task when the control developers are given a system and not given full information on the system. This restriction will be lifted in Chapter 7 where steering dynamics are explicitly addressed.

To control the longitudinal dynamics, a cascade controller is developed. The outer loop controller tracks the time-scheduled distance along the path. The outer loop controller provides a target velocity that is then tracked by the inner loop controller, which uses the throttle and brake pedals. Both the inner and outer controllers are PID controllers for simplicity. Sometimes, the single and double-lane change tests specify that the throttle should not be used once entering the lane change (known as an off-throttle test). However, in this Thesis, the longitudinal controller will be used to maintain proper longitudinal position, allowing these tests to be completed using the throttle or brakes (if needed).

3.2.2 ODD Definitions

In addition to the vehicle dynamics, the environment is modeled by separating it into three disturbances: wind, road friction, and vertical displacement. The wind is assumed planar and is decomposed into a magnitude and direction: $\boldsymbol{w} = w\angle\theta_w$. The wind disturbance is separated into a constant wind speed, (w_c, θ_{wc}), and turbulence, (w_t, θ_{wt}): $w = w_c + w_t$, $\theta_w = \theta_{wc} + \theta_{wt}$. The turbulence is generated by applying a band-pass filter to white noise. The filter is shaped similarly to observed wind speed spectra [120] and its frequency response is shown in Figure 3.5a. A slow, random walk is used to model the turbulent wind direction (ie. $\theta_{wt}(6\text{min}) - \theta_{wt}(0) = 12\text{deg}$). Combining the magnitude and direction produces the wind speed in global coordinates shown in Figure 3.5b.

The makeup of a road's surface is rarely perfectly uniform, suggesting that the road friction coefficient is not perfectly constant. This is modeled as colored noise, generated from the filter shown in Figure 3.6a. The lower cutoff frequency is tuned to eliminate any long-term drift, and the higher cutoff frequency is set at approximately 10^{-2} rad/s to limit variation. Separate profiles are generated for each tire. In addition to this, an offset is added to the noise so that the mean is set to approximately 1.0. This offset can be varied with distance, enabling the ability to simulate ice patches and split-mu test conditions. An example of the simulated road friction is shown in Figure 3.6b.

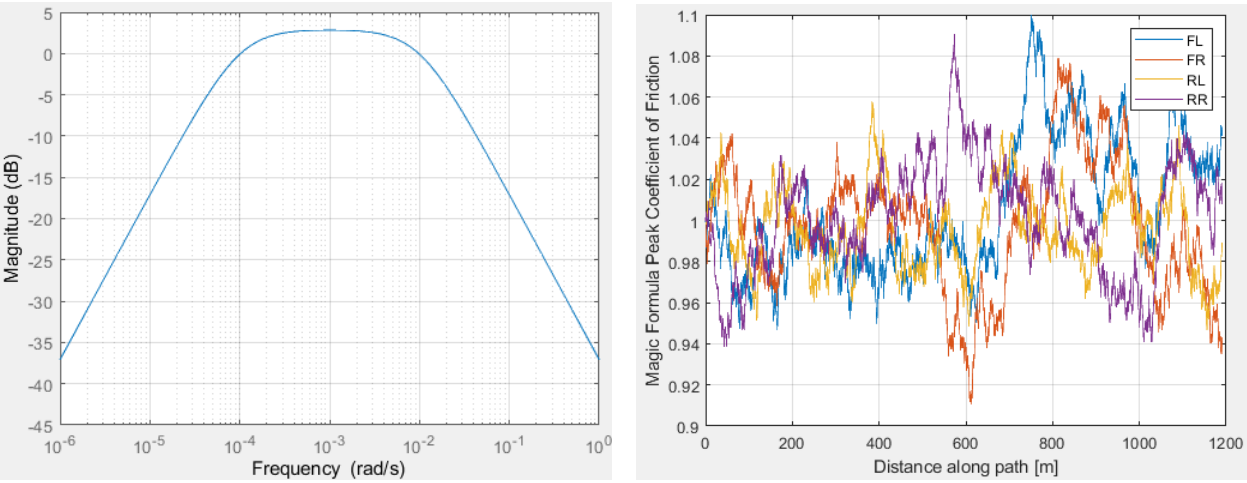


(a) Wind model

(b) Wind speed in global coordinates

Figure 3.5: Simulated Wind Disturbances

Finally, the vertical road position is rarely perfectly smooth. ISO 8608 defines a way to model different road surface profiles [115]. To generate surface profiles for each tire independently, white noise is filtered to achieve the desired spectrum, as shown in Figure 3.7. In this figure, the acronyms: FL, FR, RL, and RR correspond to the front left, front right, rear left, and rear right vertical wheel displacements. The alphabetical values correspond to the specifications in ISO 8608 [115]. However, this only provides highly granular distortions. Road elevation and super-elevation can be incorporated into this model by adding an offset to each tire's profile independently. Additional disturbances can be modeled using this architecture such as potholes and speed bumps.



(a) Road friction model

(b) Road friction profiles when the mean is set to 1.0

Figure 3.6: Simulated Road Disturbances

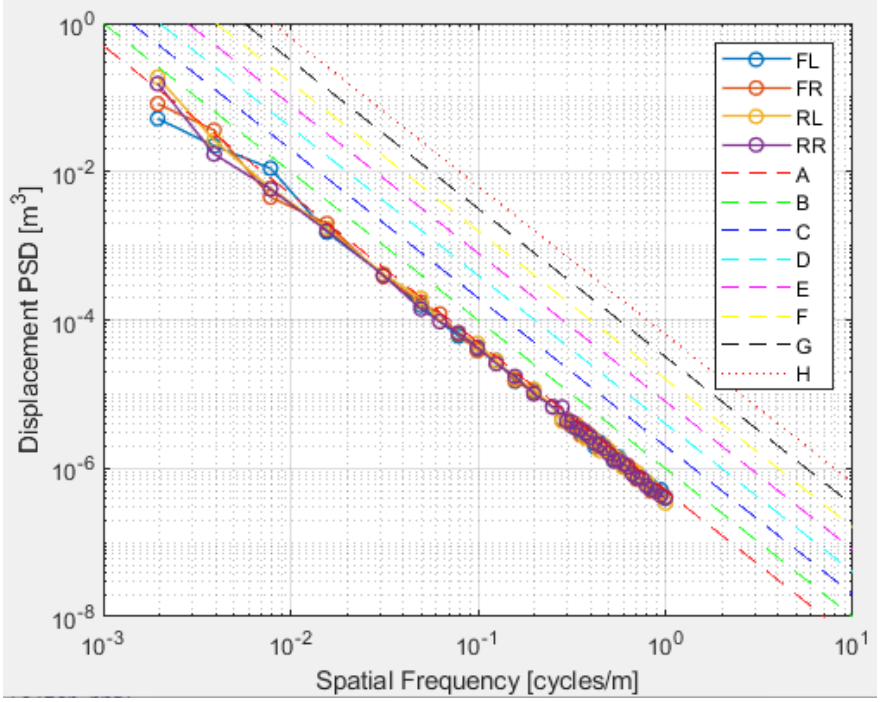


Figure 3.7: Road surface profile for each tire

Having defined the trajectories, control architecture, and environmental models, the next step is to define the test conditions. By selecting specific parameters of the disturbance models, we can create numerous ODDs. Table 3.9 defines disturbance parameters for 5 ODDs. The first ODD is called nominal and is where the feedback is without noise or delay, the friction is maximal, there is no wind, and the vertical road noise is minimal. The goal of this domain is to provide an upper bound on performance. All controllers should perform well in this domain.

	Nominal	**Realistic**	**Rural**	**Rainstorm**	**Blizzard**
Noise level	A	A	C	A	D
Wind speed [m/s]	N/A	0	5	13.4	13.4
Friction	1.0	1.0	1.0	0.7	0.4
Feedback	Perfect	RTK	DGPS	RTK	RTK
Speed adjustment [%]	0	0	0	−16	−37

Table 3.9: Operational Design Domain (ODD) Specifications.

The second ODD is realistic. It uses minimal vertical road noise, minimal wind (but with gusts), maximal friction, and feedback provided by the Extended Kalman Filter tuned to match Real-Time Kinematic Inertial Navigation System (RTK INS) performance levels. It also includes stochastic feedback delay as previously described. All ODDs, except for Nominal, use feedback delay with a mean of 60 ms with a

standard deviation of 10 ms (clipped to be strictly positive). Altogether, the Realistic ODD simulates a well-maintained road in ideal weather conditions.

The third ODD, Rural, is identical to the second except that the vertical road noise is increased and so is the wind speed. This simulates a moderately maintained road with moderate to high wind speeds. It also uses degraded feedback performance. This is reasonable since RTK requires a ground station, and some rural roads might be outside the range of that ground station.

The fourth ODD, Rainstorm, simulates driving in a rainstorm on a moderately-maintained road. The last ODD uses the highest vertical road noise, high wind speed, and the lowest road friction to simulate driving in a blizzard. Both of these ODDs reduce the reference speed to improve the reference's dynamic feasibility.

3.3 Control-Oriented Modeling and System Identification

The previous section presented a full simulator capable of testing lateral motion controllers on several maneuvers and ODDs. To design a model-based controller, a model must be developed. Several types of models can be used. The two considered in this Chapter will be the kinematic bicycle car and the dynamic bicycle car model. Both of these models approximate a four-wheel vehicle as having only a front and rear wheel in the plane because they are derived from the assumption of constant cornering. Each model's parameters will be identified from data collected on the AVA Project's Jeep Grand Cherokee, and the dynamic bicycle car model will be shown to be the best-fitting model. Then the dynamic bicycle car model's parameters will be identified from experiments collected on the simulator presented in Section 3.2.

3.3.1 Kinematic Bicycle Car Model

The kinematic bicycle car model equations of motion are developed in [160, 143]. The derivation begins with the assumption that longitudinal velocity is small and therefore the velocity vector at each wheel is in the direction of the wheel. This is the no-slip condition. Next, it is assumed that the vehicle is traveling about a constant circle of radius R. Then, based on geometry, the front steering angle can be related to the vehicle's rotational velocity $\dot{\psi}$. The equations of motion are covered in Equation 3.2 where L is the vehicle's wheelbase [m], b is the distance from the center of gravity to the rear axle [m], δ_f is the front road wheel angle [rad], β is the body slip angle [rad], and U_x is the longitudinal velocity [m/s].

$$\dot{\psi} = \frac{U_x \cos(\beta)}{L}(\tan(\delta_f))$$
$$\beta = \tan^{-1}(\frac{b\tan(\delta_f)}{L}) \tag{3.2}$$

Vehicles commonly have a steering wheel angle sensor and an IMU to support the Electronic Stability Control (ESC) functionalities required by the Federal Motor Vehicle Safety Standards (FMVSS). Therefore, the input to the model should be U_x and the steering wheel angle: $\delta_{sw} = \delta_f K_{st}$, where K_{st} is the static ratio between the steering wheel angle and the road wheel angle. The mechanical linkages of the steering system typically make this ratio nonlinear, but if the left and right road wheel angles are averaged, a static ratio is a good approximation. This is shown for the SUV vehicle in CarSim in Figure 3.8 where RWA is the road wheel angle and SWA is the steering wheel angle. The CarSim vehicle is used to illustrate this because measuring the RWA on the Jeep Grand Cherokee is difficult to do precisely.

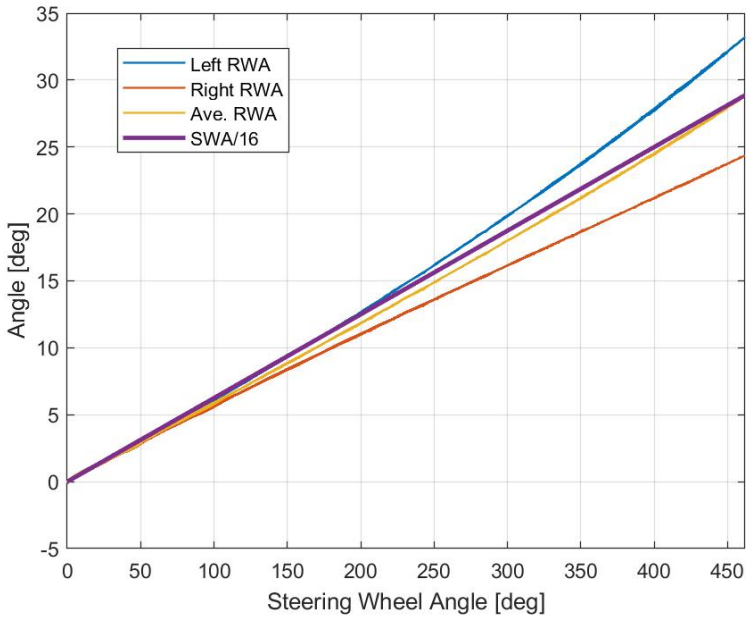


Figure 3.8: Steering angles for the SUV vehicle in CarSim

The lateral acceleration is also included as a system output $a_y = U_x^2/R = U_x\dot{\psi}$ so that this model has identical input and output signals to the dynamic bicycle car model that will be developed next. This is important to compare the quality of fit. Finally, the parameters of the kinematic bicycle car model (L,b) are readily measurable or obtainable from publicly available data [199]. So when system identification is

performed on the kinematic model, only K_{st} will be estimated from the data (although this is also commonly available on manufacturer's websites).

3.3.2 Dynamic Bicycle Car Model

The dynamic bicycle car model will be developed more carefully in Chapter 4. For now, this Chapter presents the state space model without derivation (this model is more carefully presented in Chapter 4 where the derivation is of particular interest):

$$\begin{gathered}\dot{x} = Ax + Bu \qquad x = \begin{bmatrix} \dot{y} \\ \dot{\psi} \end{bmatrix} \qquad u = \delta_f \\ A = \begin{bmatrix} \frac{-(C_f+C_r)}{mU_x} & \frac{(bC_r-aC_f)}{mU_x} - U_x \\ \frac{(bC_r-aC_f)}{JU_x} & \frac{-(a^2C_f+b^2C_r)}{JU_x} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{C_f}{m} \\ \frac{aC_f}{J} \end{bmatrix}\end{gathered} \tag{3.3}$$

where the parameters are defined in Table 3.10.

Symbol	Name	Unit	Symbol	Name	Unit
δ_f	Front tire angle	rad	C_f	Front effective stiffness	N/rad
U_x	Longitudinal velocity	m/s	C_r	Rear effective stiffness	N/rad
$\dot{y}$	Lateral velocity	m/s	a	C.G. distance from front axle	m
b	C.G. distance from rear axle	m	m	Vehicle mass	kg
$\dot{\psi}$	Yaw rate	rad/s	J	Vehicle yaw moment of inertia	kg-m^2

Table 3.10: Parameters and States for the Lateral Bicycle Car Model

The input to this model is the Steering wheel angle measurements, which are converted statically to the road wheel angle and the longitudinal velocity (formally a state space parameter). The outputs of the model that will be used for system identification are yaw rate and lateral acceleration (which are measurable with an IMU).

3.3.3 Formulating the Parameter Identification Problem

Before developing any control algorithm that uses either model, the parameters must first be identified. There are several ways of doing this. The parameters such as mass, and center of gravity location are the easiest parameters to measure directly. These parameters are commonly recorded on Original Equipment

Manufacturer's websites and in publications such as [199]. Measuring C_{yf} and C_{yr} directly requires specialized tire testing equipment. To measure J, specialized test rigs can be used. However, the control engineer might be tempted to use assumptions on the weight distribution and geometry to compute an estimate of J with the Parallel Axis Theorem. For small-scale vehicles, specialized equipment can be affordably constructed, but for full-scale vehicles, this is often cost-prohibitive. Therefore, one of the cheapest techniques is estimating parameters from data collected when the vehicle is driven in maneuvers that excite the full range of its dynamics. This breaks the task of estimating parameters into several subtasks:

1. Experimental design for proper excitation of the vehicle's dynamic range
2. Data processing and sensing
3. Computation of the model's parameters that best fit the data
4. Model validation

The first subtask has been the subject of research for decades. While each Original Equipment Manufacturer might argue for some specific experimental design, the industry standard is codified in ISO 19364, ISO 7401, and ISO 22140 [110, 114, 111]. The random steer test has a nice theoretical interpretation since it uses a chirp signal as an input to the steering angle as the vehicle is driven at various speeds. By sweeping through the expected frequency range of the vehicle's response to steering angle (0 - 4 Hz), linear systems theory argues that all frequencies are properly excited.

However, the linear system theory argument is qualified by the assumptions required to linearize the bicycle car model (explained thoroughly in Chapter 4). Therefore, it is required that the random steer test be conducted such that the linearizing assumptions are valid; specifically the small angle approximation. To account for the assumption of constant longitudinal velocity, the test can be conducted at various constant velocities throughout the full velocity range of the vehicle. In a sense, this experimental design assumes the construction of a family of linear models. However, there may be some dynamic phenomena that are not captured in this family as a result of its linear assumptions. To this end, the fourth subtask must acknowledge these assumptions and attempt to argue that such a family of models is useful.

The second subtask is highly dependent on the availability of sensors and onboard computation. It has already been explained that the inputs to both models will be the steering wheel angle and the longitudinal velocity. The outputs will be the lateral acceleration and yaw rate.

The third subtask is to compute the bicycle car model's parameters that best fit the data. Decades ago this might have required significant software development, but now the Mathworks System Identification Toolbox supplies many advanced model identification tools. The one that is most appropriate for identifying the model parameters is the nonlinear grey-box identification tool. This tool formulates the parameter identification problem as an optimization problem.

We begin with the nonlinear least squares problem. Given measurements of the output $y_{meas}(t) \in \mathbb{R}^{n_y}$ and the model predictions $y_{pred}(t, \theta) \in \mathbb{R}^{n_y}$ that depends on parameters $\theta \in \mathbb{R}^{n_\theta}$, we define the cost to be minimized:

$$e(t, \theta) = y_{meas}(t) - y_{pred}(t, \theta)$$

The objective is to find θ that minimizes the weighted sum of squared errors:

$$J(\theta) = \frac{1}{N} \sum_{t=1}^{N} e^T(t, \theta) W(\theta) e(t, \theta) \tag{3.4}$$

Here, $W(\theta) \in \mathbb{S}^{(n_y \mathrm{x} n_y)}$ is a positive definite matrix and $e(t, \theta) \in \mathbb{R}^{n_y}$.

The Levenberg-Marquardt algorithm will be used to solve this problem. We begin by computing the gradient, $\nabla J(\theta)$, and the Hessian $H(\theta)$. This is done in three steps:

1. Compute the residual vector for all time steps, stacking them as follows:

$$E(\theta) = \begin{bmatrix} e(t_1, \theta) \\ e(t_2, \theta) \\ \vdots \\ e(N, \theta) \end{bmatrix} \in \mathbb{R}^{N n_y}$$

   We then stack copies of $W(\theta)$ diagonally to form a block diagonal matrix $\tilde{W}$. Then the cost function can be written as:

$$J(\theta) = \frac{1}{N} E(\theta)^T \tilde{W} E(\theta)$$

2. The gradient of $J(\theta)$ with respect to θ is:

$$\nabla J(\theta) = \frac{1}{N}\left(\frac{\partial E(\theta)}{\partial \theta}\right)^T \tilde{W} E(\theta)$$

The Jacobian matrix:

$$J_\theta(\theta) = \frac{\partial E(\theta)}{\partial \theta} \in \mathbb{R}^{(N n_y \times n_\theta)}$$

which is approximated using with finite differences:

$$J_\theta(\theta) \approx \frac{E(\theta + \delta\theta_i p_i) - E(\theta)}{\delta\theta_i}$$

where p_i is a unit vector in the direction of the ith parameter. We then have:

$$\nabla J(\theta) = \frac{1}{N} J_\theta(\theta)^T \tilde{W} E(\theta)$$

3. The Hessian,$H(\theta)$ is the second derivative of $J(\theta)$ with respect to θ:

$$H(\theta) = \frac{\delta^2 J(\theta)}{\delta\theta^2}$$

and is approximated by the Gauss-Newton assumption (assumes that the residuals are linear in the neighborhood of the current estimate θ):

$$H(\theta) \approx \frac{1}{N} J_\theta(\theta)^T \tilde{W} J_\theta(\theta)$$

To find the solution, $\theta^* = \text{argmin} J(\theta)$ we begin with an initial guess of $\theta^{(k)}$ and compute a new estimate $\theta^{(k+1)}$. The Levenberg-Marquardt algorithm is a combination of Gradient Descent and Gauss-Newton. In Gradient Descent, $\theta^{(k+1)} = \theta^{(k)} - \lambda \nabla J(\theta^{(k)})$. In Gauss-Newton, $\theta^{(k+1)} = \theta^{(k)} - H(\theta^{(k)})^{-1} \nabla J(\theta^{(k)})$. The Levenberg-Marquardt method is $\theta^{(k+1)} = \theta^{(k)} - (H(\theta^{(k)}) + \lambda I)^{-1} \nabla J(\theta^{(k)})$. With the above simplifications:

$$\theta^{(k+1)} = \theta^{(k)} - \left(\frac{1}{N} J_\theta(\theta^{(k)})^T \tilde{W} J_\theta(\theta^{(k)}) + \lambda I\right)^\dagger \frac{1}{N} J_\theta(\theta^{(k)})^T \tilde{W} E(\theta^{(k)}))$$

Where † denotes the Moore-Penrose pseudoinverse.

This formulation is applicable to both linear and nonlinear models. The kinematic bicycle car model is nonlinear (due to the trigonometric functions). These are commonly removed by acknowledging that they are functions of the steering angle which is typically very small in highway driving. However, this approximation is not necessary because of the simplicity of the model. There is only one parameter, wheelbase, which can be easily obtained from manufacturer data. Since the kinematic model is so simple, two forms of model parameters for the kinematic model will be explored. Both forms will add an additional model parameter K_{str}, which is the steering ratio and acts to multiple δ_f in equation 3.2. The first form of the kinematic model does not have any free model parameters because it is assumed that both the wheelbase and steer ratio can be taken directly from vehicle manufacturer data. The second form estimates K_{str} and will be called "Kinematic+Steer Ratio" to differentiate it from the "Kinematic" model.

The challenge in nonlinear parameter estimation is converging to the global minima. The Kinematic+Steer Ratio model has only one parameter so a simple parameter sweep can be used to verify that the solution is the global minima. In contrast, the lateral bicycle car model is linear because of the assumed linear tire model and the small angle approximation of the steering wheel angle. This need not be the case. A nonlinear tire model can used and the small angle approximation can be removed. However, this makes the parameter estimation task more challenging due to the larger dimension and size of the parameter space. So the the model is left linear.

The predicted output for both models is the lateral acceleration and the yaw rate. $W(\theta)$ is the identity matrix for both models. For both kinematic models, the lateral acceleration is just $U_x\dot{\psi}$. For the dynamic bicycle car model, the lateral acceleration is computed with the first row of the A and B matrices in equation 3.3. The yaw rate is a state in both models.

The dynamic bicycle car model has seven total parameters, three of which are assumed to be collected from the vehicle manufacturer (the CarSim parameters in this case). The remaining four parameters (presented in Table 3.10) are estimated by solving equation 3.4.

Once the Levenberg-Marquardt algorithm converges on θ^*, the next subtask is to validate the estimated parameters. This is the subject of several ISO standards [110, 114, 111]. These standards describe a set of tests that a modeler can choose from to collect data on the vehicle. They also define data processing techniques and metrics to compute the quality of the model fit. In particular, ISO 19364 defines metrics and suggests values for valid levels of those metrics to validate a simulation model with a vehicle in steady-state conditions. ISO 22140 defines tests with transient conditions. The methods taken from these two standards

are the constant steering wheel angle test as well as the random steer test. The constant steering wheel angle test is very similar to a constant radius test, but it is easier to perform. It requires the driver to maintain a constant steering wheel angle as the vehicle is very gradually accelerated from rest. The random steer test is a transient test that consists of applying a chirp signal whose frequency linearly increases with time from 0 Hz to 4 Hz. The model can also be compared against the same data used to train. Overfitting is addressed by the fixed structure (limited complexity) of the three models.

3.3.4 Jeep Grand Cherokee Results

Experimental Design

To collect the data for estimating each model's parameters the AVA Project's Jeep Grand Cherokee is tested in a closed course on a retired aircraft runway near College Station, Texas. Several days of testing were performed. The Jeep Grand Cherokee's New Eagle Drive-by-wire system uses a steering controller to track a reference steering wheel angle. This controller performs insufficiently at frequencies above 1 Hz so a driver performed some of the chirp commands manually. The IMU measurements were recorded using ROS 1 rosbags. Because the input and output data are recorded from different systems and ROS nodes, they are not all synchronized when sampling and do not have the same sampling rate. Therefore, linear interpolation is used to synchronize all recorded inputs and outputs at a sample rate of 50 Hz so that they are time-synchronized. A zero-phase anti-aliasing filter is applied before down-sampling any signals that require down-sampling.

Model Fit

Two different versions of the kinematic bicycle car model will be compared. The first model uses the steering ratio that is estimated in the dynamic bicycle car model system identification. The second model directly estimates the steering ratio using the kinematic bicycle car model. The second model's identified steering ratio is 30.7. The parameters estimated for the dynamic bicycle car model are shown in Table 3.11.

Name	Fixed Value	Name	Estimated Value
Vehicle mass [kg]	2300	Front effective stiffness [N/rad]	107,816
C.G. distance from front axle [m]	1.4025	Rear effective stiffness [N/rad]	173,478
Wheelbase [m]	2.908	Yaw rotational inertia [kg-m^2]	3072
		Steering ratio	15.88

Table 3.11: Jeep Grand Cherokee Estimated Dynamic Bicycle Car Model Parameters

The time-domain results for an experiment that contains relatively low-frequency sine wave inputs are shown in Figure 3.9. Two types of kinematic models are identified. The first model, called “Kinematic", uses a steering ratio of 15.88. The second model, called “Kinematic+Steer Ratio" estimates the steering ratio from the data. Figure 3.9 results show that “Kinematic+Steer Ratio" model does not fit the data as well as the other two models. This is because the steering ratio is used to improve the fit quality for experiments that contain higher frequency and higher velocity data. The results for this higher frequency and higher velocity experiment are shown in Figure 3.10.

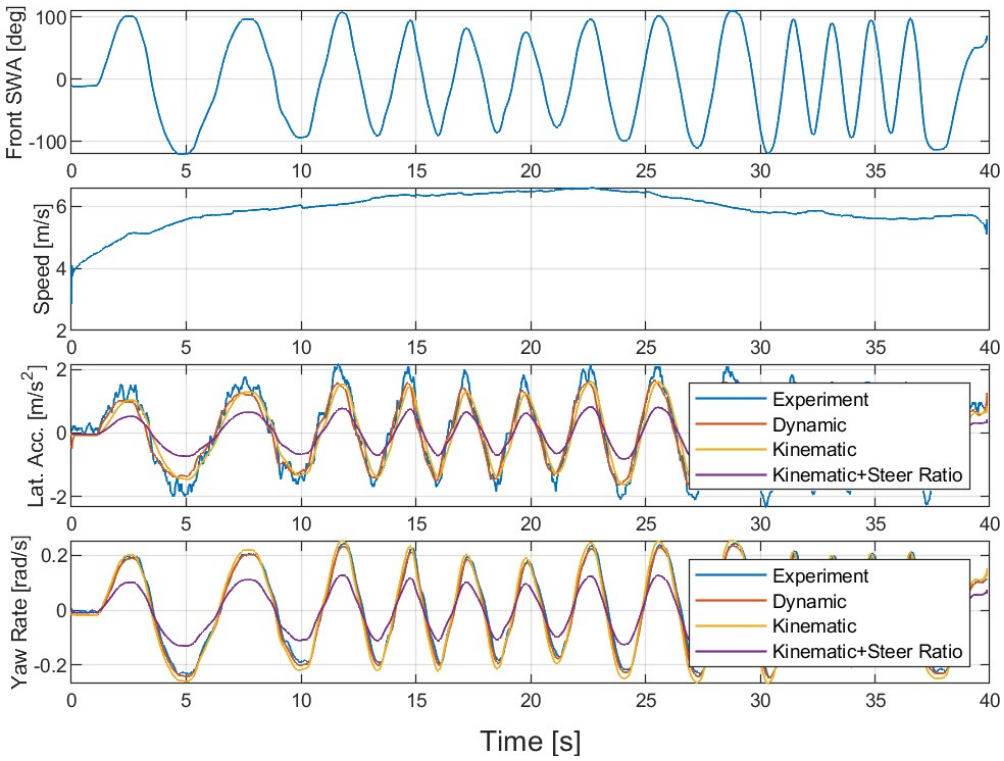


Figure 3.9: Model fit comparison for low-frequency input data

The results shown in Figure 3.10 are at a higher velocity than the experiment shown in Figure 3.9. Despite this, the “Kinematic+Steer Ratio" model fits the data nearly as well as the dynamic bicycle car

model.

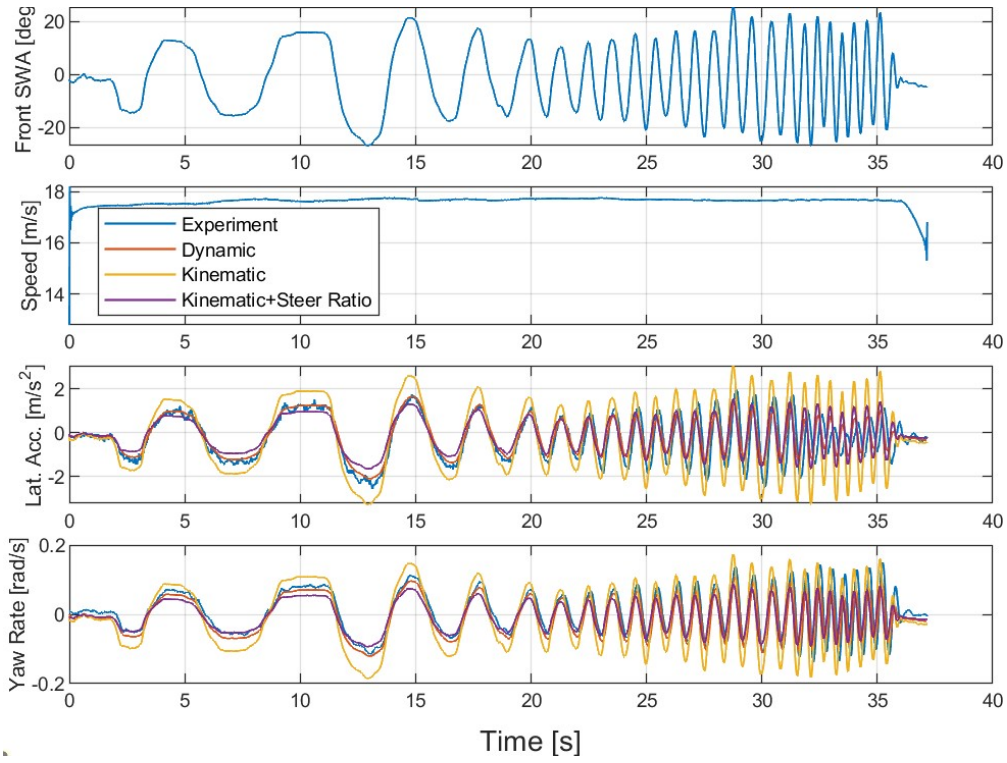


Figure 3.10: Model fit comparison for high-frequency input data

By comparing the fit results in Figures 3.10 and 3.9, a single kinematic model is shown to be incapable of fitting high and low-velocity experiments as well as the dynamic model. This is better shown in Figure 3.11 where each value is evaluated using a normalized root mean square error:

$$\text{fit } \% = 100\left(1 - \frac{||y - \hat{y}||}{||y - \text{mean}(y)||}\right)$$

where y is the experimental data output and $\hat{y}$ is the output of the model. In this metric, the higher the value the better the fit. Figure 3.11 shows that the Kinematic models deteriorate in fit quality as velocity increases, while the Dynamic model remains at approximately greater than or equal to 50 %.

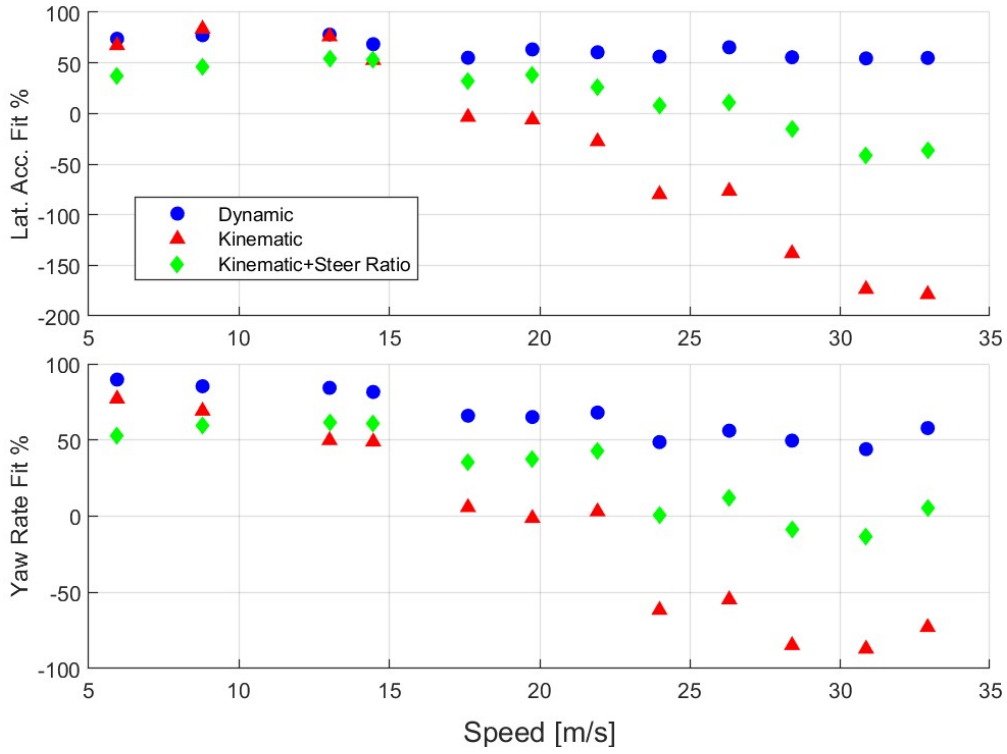


Figure 3.11: Model fit of each model as velocity increases

3.3.5 CarSim Results

This section now applies the methodology used in the previous Subsection to the SUV vehicle in CarSim. Since the previous section showed that the dynamic bicycle car model outperformed the kinematic model, only the dynamic bicycle car model will be identified from the data. This identified model will be used throughout this Thesis for controller design.

Experimental Design

The same chirp steering experiments used in the previous Subsection are conducted in CarSim to generate data. While many of these experiments had to be performed manually on the Jeep Grand Cherokee, in CarSim the input was automated by a steering wheel angle position input. The longitudinal velocity is measured from the wheel speed sensors described in Section 3.2. The lateral acceleration and yaw rate are corrupted by the sensor models described in Section 3.2. These corrupted signals are used to simulate a more realistic data collection and system identification process.

In addition to the chirp steering wheel inputs, several constant steer angle inputs are also included. These

constant steer angle inputs hold the steering wheel angle constant while increasing longitudinal velocity and are easier to perform by a human driver than a constant radius test. These experiments require large amounts of space, which is why they are not performed with the Jeep Grand Cherokee.

Model Fit

The model fit quality can be visually assessed in Figures 3.12 and 3.13. The model fit for the chirp steer input in Figure 3.12 shows an excellent fit for the lateral acceleration throughout the frequency range. At higher frequencies, the model begins to overpredict the measured yaw rate. The results shown in Figure 3.13 show an excellent fit between the model and the experimental data. This shows that the model is capable of capturing the quasi-steady-state behavior of the CarSim vehicle throughout a velocity range (0-15 m/s) and lateral acceleration range (0-5 m/s^2).

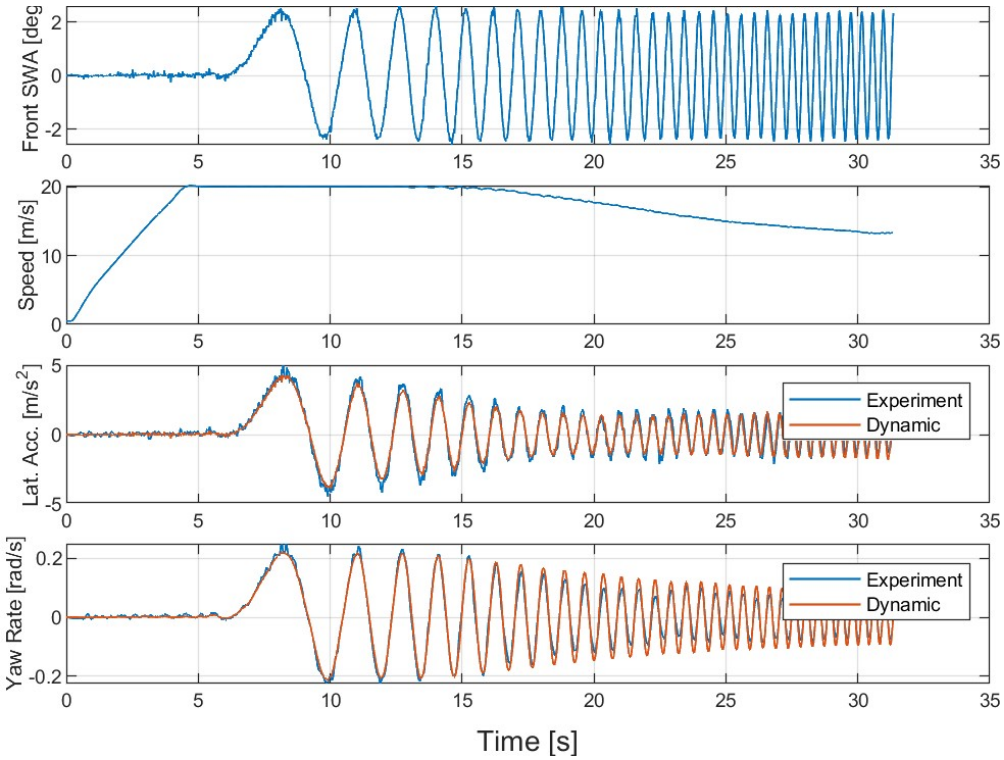


Figure 3.12: Model fit of dynamic model to CarSim data for a chirp steering input

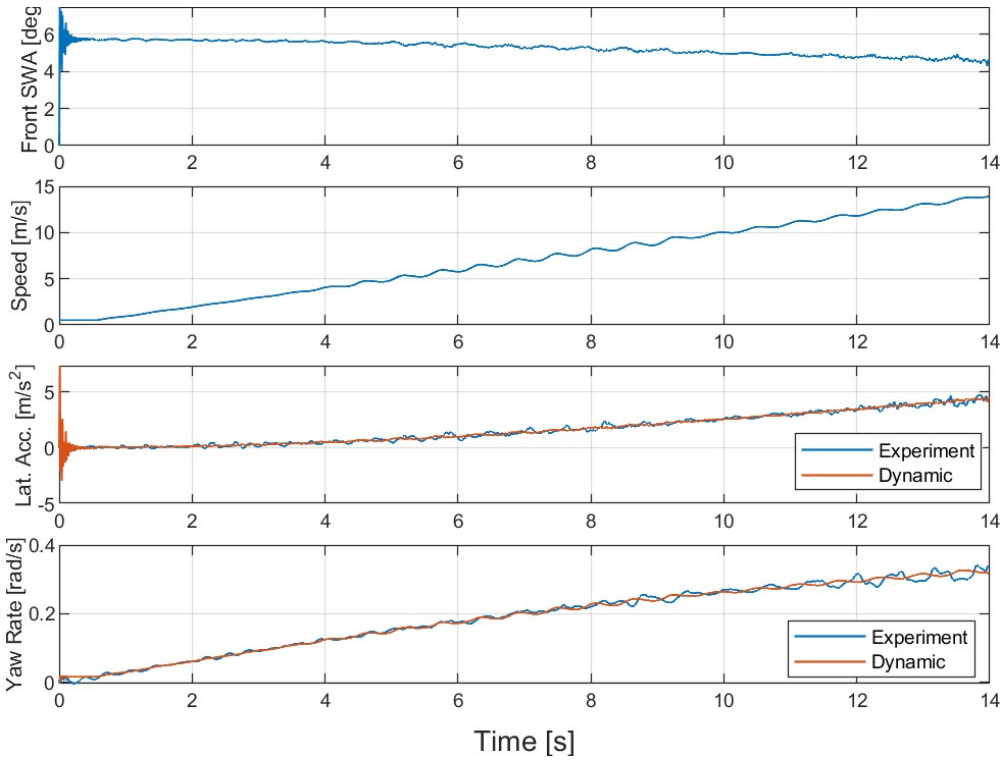


Figure 3.13: Model fit of dynamic model to CarSim data for a constant steer input

All of the experiments used to identify the dynamic model's parameters that best fit the CarSim model are shown in Figure 3.14.

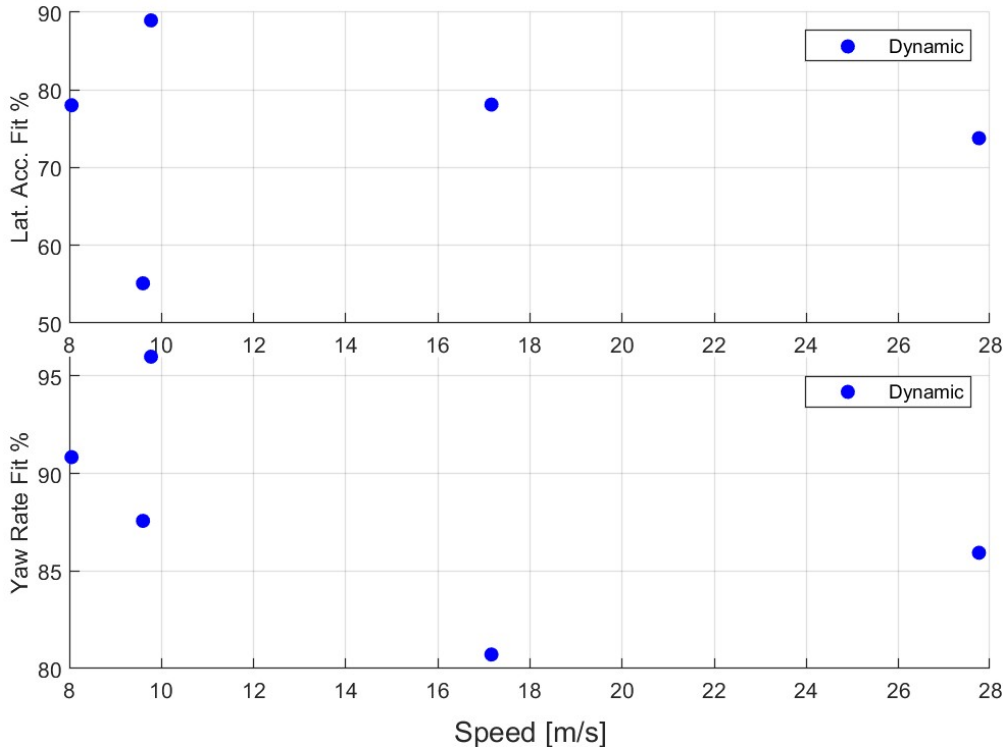


Figure 3.14: Model fit of dynamic model to CarSim data as velocity increases

The identified parameters are presented in Table 3.12. overall, the dynamic model fits the CarSim vehicle better than it fits the Jeep Grand Cherokee. This is likely because the CarSim simulations are more idealized. They have a more ideal steering input, they have perfect nominal conditions (reduced external disturbances), and the simulation is not being recorded using ROS (which may introduce unknown delays and more corrupted data). The main reason for not developing a vehicle model for the Jeep Grand Cherokee in CarSim is the extensive amount of work and resources required. The models in CarSim are highly complex and require extensive amounts of data, identification, and validation efforts to establish a reliable model of a real vehicle. Instead, a pre-parameterized model in CarSim is used so that the simulation results are repeatable by anyone with the same CarSim software version. The drawback is that the effort of identifying model parameters has to be duplicated for the CarSim vehicle and for the Jeep Grand Cherokee.

Name	**Fixed Value**	**Name**	**Estimated Value**
Vehicle mass [kg]	2691	Front effective stiffness [N/rad]	153,465
C.G. distance from front axle [m]	1.4303	Rear effective stiffness [N/rad]	153,541
Wheelbase [m]	3.14	Yaw rotational inertia [kg-m^2]	5502
		Steering ratio	16

Table 3.12: CarSim SUV Estimated Dynamic Bicycle Car Model Parameters

3.4 Conclusions

This Chapter developed the complex simulator that will be used throughout this Thesis to experimentally test lateral motion controllers. This simulator is centered around the vehicle dynamics software CarSim. The outputs of the model are corrupted with sensor models and then fused with an Extended Kalman filter. After being delayed by a stochastic duration, these signals can then be used for feedback in motion control. This makes the simulator highly realistic because noise and communication delays are always present in real-world systems.

In addition to the sensor and fusion models, the environment is modeled by separating disturbances into wind, vertical road noise, and road friction. The specifications of each model are varied and summarized in five unique Operational Design Domains. This enables the evaluation of motion controllers throughout a wide variety of realistic environmental conditions.

Five unique maneuvers are generated to be tracked by motion controllers. Three of which model highway-like maneuvers and are exceptionally long (more than a kilometer). Two of these three are modeled from existing highways in California which contain high curvatures and have a history of collisions. Two of the maneuvers are collision avoidance maneuvers and are properly motivated by ISO 3888 [113]. These five maneuvers will be useful in evaluating each lateral motion controller's ability to perform multiple objectives: lane-keeping, overtaking, and collision avoidance.

Section 3.3 then presented two models that are useful in lateral motion control development. The results in Section 3.3.4 show that the dynamic bicycle car model is a more accurate model for capturing the important phenomena in driving at low and high speeds. These findings are then used to identify the dynamic model parameters that best fit the data collected from CarSim experiments.

Chapter 4

Effects of Modeling and Implementation Choices in Lateral Motion Controllers

Several modeling and implementation choices often go unstated when presenting new lateral motion controllers for Automated Driving Systems. This Chapter seeks to fill this gap by investigating the effects of a selection of modeling and implementation choices. The first half of this Chapter is devoted to analyzing three different forms of the dynamic bicycle car model to be used for lateral motion control. The first model uses the vehicle's position in global coordinates, the second model uses the vehicle's position in local coordinates, and the third model uses error states derived from assuming a particle mass generates reference states. This same half of the Chapter presents the bicycle car model derivation. The second half of this Chapter presents simulation-based experiments to show an equivalency between the global state and the error state models. First, the similarities between these two modeling choices are shown. Then, their differences are investigated in independent Subsections. Modifications are proposed to mitigate the challenges proposed by each model. Another implementation detail investigated is the choice of using a trajectory or a path as the reference to the lateral controller. Simulation results show little differences in performance or robustness when using either a trajectory or a path as the reference. However, it is shown that lateral motion control becomes more sensitive to longitudinal motion control performance when using a trajectory.

4.1 Introduction

This Chapter investigates the effects of modeling and control implementation choices when developing lateral motion controllers. Lateral motion control refers to tracking lateral position and yaw references. This discussion seeks to draw connections between concepts and provide useful background information for anyone designing a lateral motion controller. The first half of this Chapter, Section 4.2, develops the equations of motion for the bicycle car model. Then, three different selections are made for the model's

states:

1. Local state model (Section 4.2.2): which uses the vehicle's position in the vehicle's (local) coordinate frame.

2. Global state model (Section 4.2.3): which uses the vehicle's position in the inertial (global) coordinate frame.

3. Error state model (Section 4.2.4): which converts the local state model to error states by assuming the references are generated from a particle mass.

This discussion then informs this Chapter's second half (Section 4.3), which addresses different ways controllers can be implemented. Section 4.3 uses simulation results to illustrate the impact of some implementation techniques on the control performance.

Model-based control development begins with selecting an appropriate model to describe the physical phenomena to be controlled. Similar to the other Chapters in this thesis, a wide range of vehicle velocities is to be considered. At high velocities, inertial effects become important, so the dynamic bicycle car model is selected for investigation [160, 64, 164, 175]. The discussion of modeling choices begins after developing the bicycle car equations of motion where the local coordinate velocity states are exposed. There are several ways to derive these equations of motion. Since we wish to include the lateral position of the vehicle, the equations will be derived with the lateral acceleration.

Each of these models will be compared in terms of their controllability, their simplicity, and their potential for control. At a high level, the first model (local state model) is ill-suited for lateral motion control because it is not stabilizable. The second model (global state model) is well-suited for lateral motion control because it is controllable and presents the servomechanism control problem (reference tracking). One potential benefit of having non-zero, time-dependent references, is that it allows for a prefilter to be used as an additional design degree of freedom [93]. The third model (error state model) is also well-suited for lateral motion control because it is controllable and presents the regulator control problem. The regulator control architecture is simple and well-suited for optimal state feedback design such as the Linear Quadratic Regulator [6]. The error state model also has a physically interpretable output (the lateral error at some look-ahead distance) that results in a single-input, single-output system [160]. It is a popular choice for output feedback control [83, 164, 121, 135, 165]. Another benefit is that the model directly exposes a measurable

disturbance (curvature) that can be rejected with appropriately designed feedforward control. Section 4.3 presents several simulation experiments that illustrate the high-performance control achievable with each of the last two models.

After selecting the model states, the nuances in how each controller is implemented become significant. Section 4.3 is separated into four Subsections. The first Subsection (4.3.1) shows how both the global state model and the error state model are equivalent under a set of assumptions. The second Subsection (4.3.2) explores how the servomechanism control problem exposed by the global state model can leverage an additional design degree of freedom. It also shows that the ability to know future references within a limited time horizon can improve the control system's performance. The third Subsection (4.3.3) explores two different methods of implementing output feedback with the error state model. The first method is causal. To improve the performance of this causal controller, a new modification is shown to improve performance. The second method is non-causal and also improves performance. The final Subsection (4.3.4) presents an additional implementation detail that applies to all controllers. This is whether a path or a trajectory is used as a reference. The key distinction is that path-tracking control assumes the references are parameterized by space, while the trajectory-tracking control assumes the references are parameterized by time (hence the need for a third term encompassing both such as "Lateral Motion Control"). This distinction is studied with simulation experiments to show some of the trade-offs found in each method.

4.2 Modeling Choices

4.2.1 Bicycle Model Equations of Motion

The application of control theory to solve real-world problems involves choosing assumptions that make the problem solvable. Typically those assumptions are based on the problem's constraints. For instance, the design of a lateral motion controller for a vehicle driving on the highway might assume that the system behaves with linear dynamics. Such assumptions introduce modeling errors because some phenomena of the system are ignored, while others are not. The challenge in modeling is the selection of which phenomena to capture with a model and which to ignore.

The use of a bicycle-car model (also known as the single-track model) has been standard practice for controller development and system analysis for decades [64, 107, 175, 36, 154, 75]. The key assumption in this model is that the vehicle is undergoing steady-state cornering, a constraint that is often satisfied on

highway-like maneuvers. Under this assumption, the difference in forces of the left and right wheels can be lumped together without introducing significant modeling errors (which gives the model its name).

Typically, the bicycle-car model is used to analyze steady-state handling characteristics such as the understeer gradient [75]. When using the bicycle-car model for lateral motion control, it is useful to include the position coordinates as a model state. The subsequent development of this model closely follows what is explained in [160]. However, each assumption is more carefully elaborated.

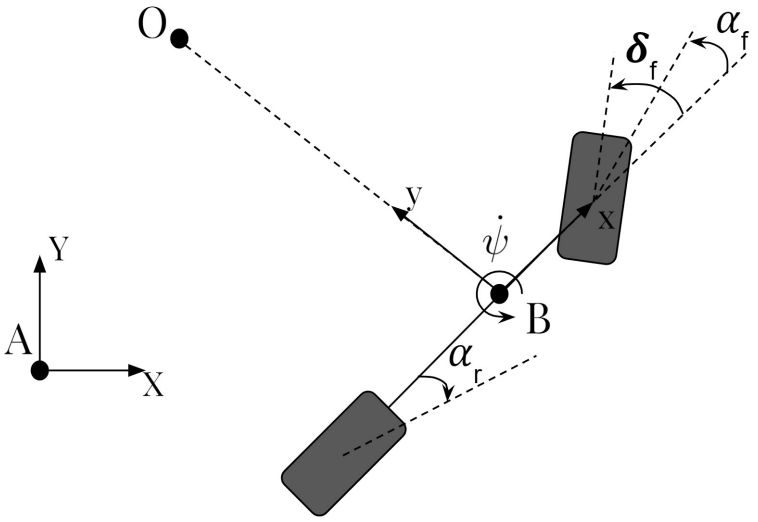


Figure 4.1: Bicycle Car Model Diagram.

Figure 4.1 shows a diagram of the bicycle car model in a global coordinate frame A with directions X and Y. The local coordinate frame, B, is fixed to the vehicle's chassis with directions x and y. The point O is the *vehicle chassis's* instantaneous center of rotation. The previously stated steady-state cornering assumption places the chassis in constant circular motion about point O. The acceleration in the y direction is $a_y = \ddot{y} + U_x\dot{\psi}$. The front and rear tires generate forces orthogonal to the tire's direction. At high velocities between the tire center and the ground experimental results show that the lateral tire forces are well modeled as a function of lateral slip angle [75, 154]. The lateral slip angle is the ratio between the tire's lateral and longitudinal velocities. They are:

$$\begin{aligned} \alpha_f &= \delta_f - \tan^{-1}\left(\frac{\dot{y} + a\dot{\psi}}{U_x}\right) \\ \alpha_r &= -\tan^{-1}\left(\frac{\dot{y} - b\dot{\psi}}{U_x}\right) \end{aligned} \tag{4.1}$$

where the parameters are defined in Table 4.1.

Symbol	Name	Unit	Symbol	Name	Unit
α_f	Front lateral slip angle	rad	F_{yf}	Front lateral tire force	N
α_r	Rear lateral slip angle	rad	F_{yr}	Rear lateral tire force	N
δ_f	Front tire angle	rad	C_f	Front effective stiffness	N/rad
U_x	Longitudinal velocity	m/s	C_r	Rear effective stiffness	N/rad
$\dot{y}$	Lateral velocity	m/s	X	C.G. global X position	m
a	C.G. distance from front axle	m	Y	C.G. global Y position	m
b	C.G. distance from rear axle	m	ψ	Vehicle yaw angle	rad
m	Vehicle mass	kg	$\dot{\psi}$	Yaw rate	rad/s
J	Vehicle yaw moment of inertia	kg-m^2			

Table 4.1: Parameters and States for the Lateral Bicycle Car Model

This tire model using lateral slip angles is valid for small lateral slip angles, so a small angle approximation can be used to simplify the tire slip angles:

$$\begin{aligned} \alpha_f &= \delta_f - \frac{\dot{y} + a\dot{\psi}}{U_x} \\ \alpha_r &= -\frac{\dot{y} - b\dot{\psi}}{U_x} \end{aligned} \tag{4.2}$$

A more accurate model of the relationship between lateral slip angle and lateral tire forces is nonlinear and dynamic [154]. However, for sufficiently small slip angles (typically less than 5 or 6 degrees) the tire force is well modeled as a linear relationship with lateral slip angle. In this operating region, the proportionality constant is called the lateral cornering stiffness. When identifying these parameters from data, there are a large number of additional effects that are lumped into this proportionality constant [75]. We will refer to these as “effective" cornering stiffnesses to acknowledge this. Now the linear tire model is:

$$\begin{aligned} F_{yf} &= C_f \alpha_f \\ F_{yr} &= C_r \alpha_r \end{aligned} \tag{4.3}$$

Where the parameters are defined in Table 4.1. Now, applying Newton's law:

$$\begin{aligned} m(U_x\dot{\psi} + \ddot{y}) &= F_{fy} + F_{ry} \\ J\ddot{\psi} &= F_{fy}a - F_{ry}b \end{aligned} \tag{4.4}$$

Now combine Equations 4.4, 4.3, and 4.2, and rearrange to form the following state space equations:

$$\begin{aligned} &\dot{x} = Ax + Bu \qquad x = \begin{bmatrix} \dot{y} \\ \dot{\psi} \end{bmatrix} \qquad u = \delta_f \\ &A = \begin{bmatrix} \frac{-(C_f+C_r)}{mU_x} & \frac{(bC_r-aC_f)}{mU_x} - U_x \\ \frac{(bC_r-aC_f)}{JU_x} & \frac{-(a^2C_f+b^2C_r)}{JU_x} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{C_f}{m} \\ \frac{aC_f}{J} \end{bmatrix} \end{aligned} \tag{4.5}$$

4.2.2 Local States

To use the bicycle car model for control, we may consider including the lateral position (in local coordinates), y, and yaw angle, ψ, of the vehicle in the state space model:

$$\frac{d}{dt}\begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_f+C_r}{mU_x} & 0 & -U_x + \frac{-aC_f+bC_r}{mU_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{aC_f-bC_r}{JU_x} & 0 & -\frac{a^2C_f+b^2C_r}{JU_x} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{aC_f}{J} \end{bmatrix} \delta_f \tag{4.6}$$

However, this plant is not straightforward to use in control development. To show this, consider the controllability of the system given in Equation 4.6. The controllability matrix, R, is given by $R = \begin{bmatrix} B & AB & A^2B & A^3B \end{bmatrix}$ [93]. By computing R, one can show that the Controllability matrix is rank-deficient, and therefore the model is uncontrollable.

A more general state space model, given in Equation 4.7, is also uncontrollable. Therefore, any (A,B) pair having this structure is uncontrollable.

$$\dot{x} = \begin{bmatrix} 0 & a_{12} & 0 & a_{14} \\ 0 & a_{22} & 0 & a_{24} \\ 0 & a_{32} & 0 & a_{34} \\ 0 & a_{42} & 0 & a_{44} \end{bmatrix} x + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} u \tag{4.7}$$

Furthermore, it can be shown that if y or ψ are dropped as a state variable from Equation 4.6, the resulting state space models are controllable. From this observation, it can be concluded that there is some property about including both y and ψ in the model that results in a loss of controllability.

To further investigate this result, the controllable decomposition of Equation 4.6 may be computed by finding a suitable transformation matrix T that maps the realization (A, B, C) to $(\bar{A}, \bar{B}, \bar{C})$, where

$$(\bar{A}, \bar{B}, \bar{C}) = (T^{-1}AT, T^{-1}B, CT)$$

$$\bar{A} = \left[\begin{array}{c|c} \bar{A}_c & \bar{A}_{12} \\ \hline 0 & \bar{A}_{uc} \end{array}\right] \qquad \bar{B} = \left[\begin{array}{c} \bar{B}_c \\ \hline 0 \end{array}\right] \qquad x = T\bar{x}$$

and $(\bar{A}_c, \bar{B}_c)$ form a controllable system.

To compute T symbolically, three linearly independent columns of R formed from the A and B matrices in Equation 4.6 can be combined with an arbitrary column to form a full rank T:

$$T = \begin{bmatrix} 0 & \cdot & \cdot & 1 \\ \frac{C_f}{m} & \cdot & \cdot & 0 \\ 0 & \cdot & \cdot & 0 \\ \frac{C_f\, a}{J} & \cdot & \cdot & 0 \end{bmatrix} \qquad \bar{A} = \left[\begin{array}{ccc|c} 0 & 0 & 0 & 0 \\ \cdot & 0 & \cdot & 0 \\ \cdot & 1 & \cdot & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \qquad \bar{B} = \left[\begin{array}{c} 1 \\ 0 \\ 0 \\ \hline 0 \end{array}\right] \tag{4.8}$$

where $\cdot$ represents non-zero elements that are too long to present here. The uncontrollable state is $(\cdot)y + (\cdot)\dot{y} + (\cdot)\psi$ (a linear combination of all states except $\dot{\psi}$).

When designing controllers, stabilizability is often a minimum condition. The uncontrollable mode is unstable since its eigenvalue is 0. However, since $\bar{A}_{12} = \mathbf{0}$ this uncontrollable mode does not influence the other states. Furthermore, since the last rows of $\bar{A}$ and $\bar{B}$ are zero, the uncontrollable state, x_{uc}, is constant ($\dot{x}_{uc} = 0$). Therefore, to overcome the issue of an uncontrollable, constant mode, we may simply ignore it and treat the controllable system as our plant.

In all, we have shown that the bicycle car model with local position states is not stabilizable. It can, however, be decomposed into a lower-order controllable state space model. The next Subsections will present a selection of states that present a more useful model for control because this decomposition does not need to be performed.

4.2.3 Global and Local States

Rather than ignoring the unstable mode of the model, there exists an alternative approach to making the bicycle car model more suitable for control. The structure of Equation 4.7 suggests that some of the zero terms in the state matrix should be somehow made nonzero to make the system controllable. It turns out that this is indeed true.

To show this, include the global lateral position state, $\hat{Y}$, with Equation 4.5. The geometric relationship is:

$$\hat{Y} = \int (U_x \sin(\psi) + \dot{y}\cos(\psi))dt \tag{4.9}$$

Then, a small yaw angle assumption gives: $\dot{\hat{Y}} = U_x \sin(\psi) + \dot{y}\cos(\psi) \approx U_x\psi + \dot{y}$. The resulting state space model, given in Equation 4.10, is controllable.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ \frac{d}{dt}\begin{bmatrix} \hat{Y} \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & U_x & 0 \\ 0 & -\frac{C_f+C_r}{mU_x} & 0 & -U_x + \frac{-aC_f+bC_r}{mU_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{aC_f-bC_r}{JU_x} & 0 & -\frac{a^2C_f+b^2C_r}{JU_x} \end{bmatrix} \begin{bmatrix} \hat{Y} \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{aC_f}{J} \end{bmatrix} \delta \end{aligned} \tag{4.10}$$

The new, non-zero term in the third column of the A matrix in Equation 4.10 results in a controllable (A, B) pair. This can be conceptually explained by considering the new U_x term as coupling $\dot{y}$ and ψ.

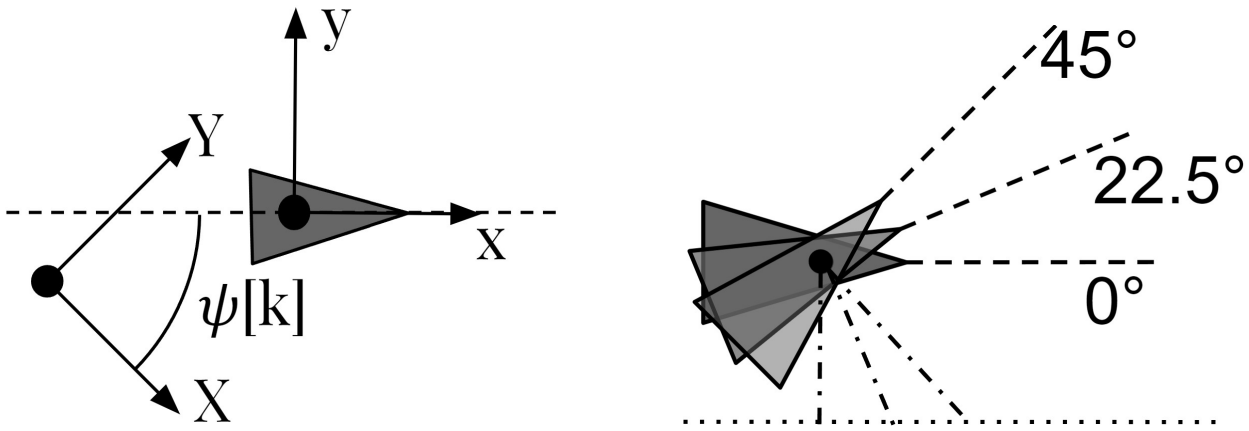
In addition to being controllable, this model is also useful because it allows tracking control. Specifically, a 2 Degree of Freedom (DoF) controller can be designed with this model. In addition to this, if the references are assumed to be known within some future time horizon, non-causal prefilters can be designed to significantly improve tracking performance.

The challenge presented by this controller is that it assumes ψ is small, which is likely not true for many types of maneuvers. However, a unique implementation method can be used to overcome this.

This unique implementation method begins by realizing that the choice of coordinate frame is arbitrary. As long as the references and feedback are in the same coordinate frame, control is possible. We begin

by redefining the final coordinate frame for which our dynamics are developed at each timestep such that assuming ψ is small is valid. To illustrate this, Figure 4.2a shows the final coordinate frame being redefined as the previous timestep's vehicle's pose (position and orientation). Figure 4.2a depicts the current position of the vehicle as a triangle. The global coordinates are denoted with X and Y. The local coordinates are denoted with x and y and are fixed to the vehicle. The angle between the reference x and global X is $\psi[k]$ (where k is the current timestep). If the vehicle's yaw rate is sufficiently small, then $\psi[k]$ will also be small and guarantee the small angle approximation's validity. Furthermore, since we assume we can redefine the final coordinate frame at each timestep wherever we wish, we can also define it as the current vehicle's pose (position and orientation). Note that this choice is not illustrated because it is difficult to show when the local and global coordinate frames are the same. This choice will also make the vehicle's yaw angle zero. Furthermore, to ensure that control is not adversely affected, the references also need to be transformed into this new final coordinate frame at each timestep.

While this method addresses the small angle assumption limitations, it comes at the cost of increasing the reference lateral position's sensitivity to the angle between the vehicle's heading and the reference heading. This is shown geometrically in Figure 4.2b where there are three instances of the vehicle overlaid at the same position, but with different orientations. The dotted line represents the reference. The dash-dot line represents the lateral position of the reference in the vehicle's coordinate frame. As the angle between the vehicle's heading and the reference heading (yaw error) increases, the reference's lateral position also increases.



(a) Geometry of redefining the final coordinate frame as the vehicle's previously sampled position

(b) Sensitivity of local lateral position when using vehicle position

Figure 4.2: Geometric representation of coordinate transformation

The previous implementation method decided to select the current vehicle's pose as the new final coordinate frame at each timestep. This results in $\psi = 0$, thus guaranteeing the validity of the small angle approximation. An alternative pose to define as the final coordinate frame is the current reference pose. If the lateral motion controller tracks the references well, then the angle between the vehicle's orientation and the reference orientation is small by definition. Therefore, this alternative choice also guarantees that the small angle approximation is valid (as long as good reference tracking is achieved). However, the reference lateral position and orientation in the new final coordinate frame would then be zero, which converts this problem from a servomechanism problem to a regulator problem. A benefit of doing this is that the vehicle's lateral position in the reference coordinate frame would be nonzero and independent of the angle between the reference and vehicle headings. This is shown geometrically in Figure 4.3, where three vehicle poses are overlaid. Each vehicle has the same position but a different orientation. The reference path is the dotted line, and the reference coordinate frame is denoted with x and y. The lateral position of all the vehicles in the reference coordinate frame is shown with the dash-dot line. All three of the vehicle's lateral positions in the reference coordinate frame are the same.

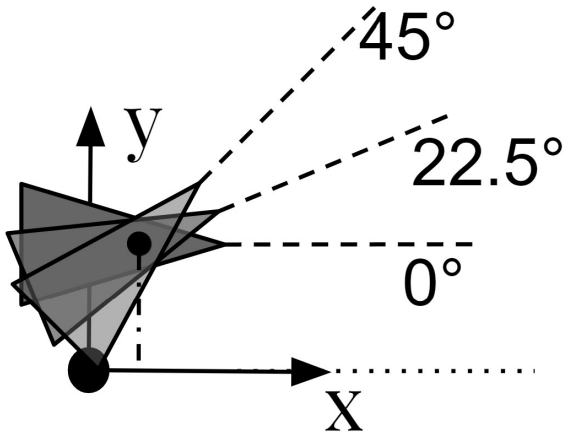


Figure 4.3: Geometric representation of the vehicle's lateral position's insensitivity to the yaw error

Figures 4.3 and 4.2b shows a sensitivity trade-off between the two final coordinate frame options: (1) the final coordinate frame is defined as the current vehicle's pose and (2) the final coordinate frame is defined as the current reference pose. For the global coordinate model, it is preferred to have a non-zero reference, so the first option is selected. The second option is reserved for the error state model that will be developed in Section 4.2.4.

Coordinate transforms are performed by first rotating and then translating. Suppose there are two co-

ordinate frames A and B. Frame B is represented with respect to frame B as a rotation matrix R_A and translation t_A. The point p^A is the coordinates of point p in frame A; p^B is the point p in frame B. Their relationship is then: $p^A = T_A(p^A) = R_A p^A + t_A$.

Option 1 can be implemented online by suitably transforming reference and vehicle poses from the global coordinate frame into the current vehicle's coordinate frame each timestep. The control diagram including this coordinate system change is shown in Figure 4.4. The transformation from the global coordinate frame to the vehicle coordinate frame is denoted T_{veh} and is depicted as a block that applies the transformation to the block's input.

The reference, after being transformed into the vehicle coordinate frame is the same as the lateral position error state in Equation 4.14, e_1. The yaw error is $e_2 = \psi_{ref} - \psi = \psi_{ref} - 0$. The block K is the controller. The dashed line encircling the vehicle and coordinate transform is the representation of the model in Equation 4.10. Figure 4.4 assumes the measurements are of the global vehicle position Y and the yaw angle ψ. However, there are also measurements of lateral acceleration and yaw rate that can be output of the model. This figure is simply for illustration. Later in Section 4.3.1 the control architecture will become more specific to the desired control design.

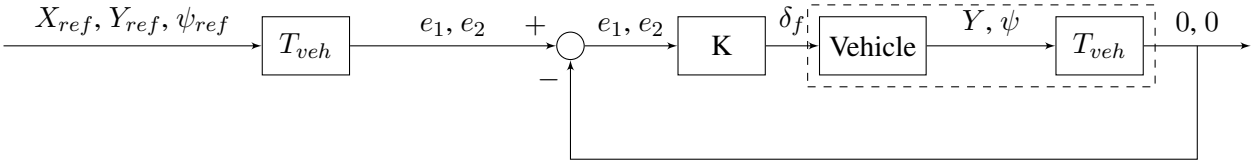


Figure 4.4: Control block diagram of plant using global coordinates redefined at each step

4.2.4 Error States

When using the dynamic bicycle car model to analyze or design a controller to track a reference it is common to rewrite the dynamics in terms of error states [121, 76, 81, 64]. Most ways of reformulating these dynamics into error states assume the references are well modeled by a particle undergoing circular motion. The states of this particle are used as a reference: Y_{ref}, $\dot{y}_{ref}$, ψ_{ref}, and $\dot{\psi}_{ref}$.

The reference kinematics in the inertial reference frame are:

$$\begin{aligned} a_{y,ref} &= R\dot{\psi}_{ref}^2 = U_x\dot{\psi}_{ref} \\ v_{y,ref} &= U_x\psi_{ref} \\ \dot{\psi}_{ref} &= \dot{\psi}_{ref} \\ \hat{Y}_{ref} &= -R_{ref} \end{aligned} \tag{4.11}$$

Before proceeding it is useful to introduce the following definition:

$$Y \coloneqq U_x \int \psi dt \tag{4.12}$$

which is derived from an integration of $v_{y,ref}$ and treating U_x as a constant. Using this definition and the assumption that the control does a good job of bringing the vehicle states close to reference states allows for the formulation of the following error states:

$$\begin{aligned} e_1 &= r_y - r_{y,ref} \\ &= Y - Y_{ref} = U_x \int \psi dt - U_x \int \psi_{ref} dt \\ &= U_x \int e_2 dt & e_2 &= \psi - \psi_{ref} \\ \dot{e}_1 &= v_y - v_{y,ref} & \dot{e}_2 &= \dot{\psi} - \dot{\psi}_{ref} \\ &= \dot{y} + U_x\psi - U_x\psi_{ref} = \dot{y} + U_x e_2 & \ddot{e}_2 &= \ddot{\psi} \\ \ddot{e}_1 &= a_y - a_{y,ref} \\ &= \ddot{y} + U_x\dot{\psi} - U_x\dot{\psi}_{ref} = \ddot{y} + U_x\dot{e}_2 \end{aligned} \tag{4.13}$$

The small angle assumption is used to linearize the lateral velocity of the vehicle and particle. To ensure this is valid we may use the second option for redefining the global coordinate frame: setting at it the current reference pose. Combining these definitions with bicycle-car dynamics, the resulting state space model is:

$$\dot{x} = Ax + B_1\delta + B_2\dot{\psi}_{ref} \tag{4.14}$$

$$\frac{d}{dt}\begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_f+C_r}{mU_x} & \frac{C_f+C_r}{m} & \frac{-aC_f+bC_r}{mU_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{aC_f-bC_r}{JU_x} & \frac{aC_f-bC_r}{J} & -\frac{a^2C_f+b^2C_r}{JU_x} \end{bmatrix} \begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{aC_f}{J} \end{bmatrix} \delta + \begin{bmatrix} 0 \\ -\frac{aC_f-bC_r}{mU_x} - U_x \\ 0 \\ -\frac{a^2C_f+b^2C_r}{JU_x} \end{bmatrix} \dot{\psi}_{ref} \tag{4.15}$$

The pair given by (A, B_1) forms a controllable system. The control block diagram for the error state model is shown in Figure 4.6. The dashed box encircles the system that is modeled by Equation 4.14 (excluding the controller). The difference between Figures 4.5 and 4.4 is that the model encompasses both the model and the references (excluding the controller). Also, the alternative coordinate frame transformation is assumed. This means that the vehicle pose is transformed into the reference pose coordinate frame. This transformation is denoted T_{ref}. The result is that the reference lateral position is zero and the feedback lateral position is nonzero.

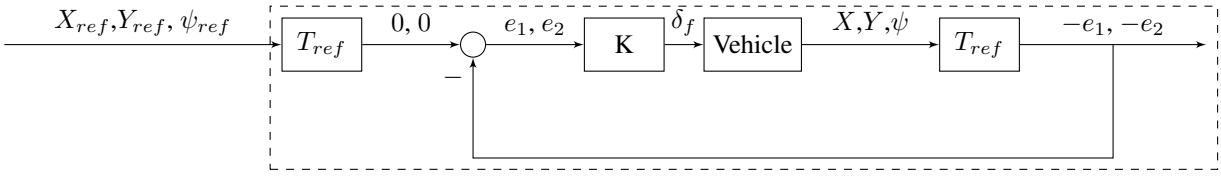


Figure 4.5: Control block diagram of error state model including rotations

The benefit of this model is that the control diagram can more simply be considered as the one shown in Figure 4.6. Furthermore, the error at some lookahead distance, d_s, is linearly approximated by $e_{LA} = e_1 + d_s e_2$. This has been shown to provide phase lead that helps compensate for the phase lag in the system [83, 81]. The challenge, however, is that the reference is always zero (since the error states should be driven to 0) and so the benefits of 2 DoF control and non-causal prefilters cannot be realized with this model.

4.3 Implementation Choices

The previous Section analyzed the different modeling choices so a controller can be designed. It also briefly discussed some implementation details that enable the use of the small angle approximation. This Section

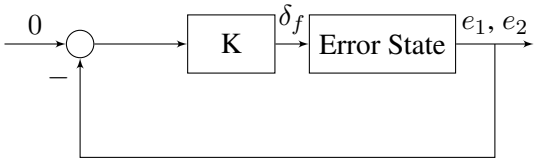


Figure 4.6: Control block diagram of error state model

now focuses on implementing controllers using the models developed in Sections 4.2.3 and 4.2.4.

The first Subsection develops controllers using the models in Equation 4.10 (global state model) and 4.14 (error state model). It is argued that these models are equivalent (under some assumptions that will be stated later). To show this, a controller is designed using the global state model, but implemented on the error state model.

Throughout this Section, the same maneuvers and environmental conditions, or Operational Design Domains (ODD), are used. There are two maneuvers. The first is the double lane change to capture collision avoidance maneuvers. The second is a maneuver on an “S" shaped highway that also contains an overtaking maneuver. This is intended to represent highway driving with larger-than-normal path curvatures. The two ODDs that will be studied are the Realistic ODD and the Rainstorm ODD. Refer to Chapter 3 for specific definitions of these ODDs.

The second Subsection explores how implementing the global state model (Equation 4.10) exposes references. If a preview of the references is assumed, it also enables using non-causal prefilters such as the Zero Phase Error Tracking Control [185] to improve tracking performance drastically. Some simulation results show the improvement that the Zero Phase Error Tracking Control achieves.

The third Subsection explores the concept of lateral error at a look-ahead distance with the error state model of Equation 4.14. The linear approximation of the look-ahead error is shown to introduce modeling error. This is addressed by modifying the regulator architecture of the controller and providing a reference that is a nonlinear function of the path curvature. Finally, the concept of non-causal control design (controllers that use future information of the references) is applied to the error state model to implement a controller that improves performance. The only difference between the non-causal and causal controllers is how the model output is computed.

The final Subsection explores the differences between treating the references as time-parameterized (trajectory) or space-parameterized (path). It is argued that all controllers developed so far can be implemented with either a trajectory or a path. Under good lateral and longitudinal control, there is little difference be-

tween each reference type. However, it is shown experimentally that when the longitudinal control degrades in performance, lateral controllers based on trajectory references suffer more than those developed with path references.

4.3.1 Equivalency of Error States and Global States

The global position model in Equation 4.10 and the error state model in Equation 4.14 both permit several control architectures and tuning techniques. Under the following assumptions, there is little difference between the control design of each plant:

1. The controller is Linear Time Invariant.
2. The controller is causal.
3. The control architecture consists only of feedback control (no additional degree of freedom or feed-forward).
4. The references are modeled well by a particle mass.

The first assumption eliminates nonlinear and parameter-varying control. These controllers may leverage the unique properties of each model differently. The second assumption eliminates any use of reference preview that might make a controller designed with one model perform differently than a controller designed with the other model. The third assumption prevents using a prefilter to provide an additional degree of freedom, which is only possible with the global position model. The fourth assumption ensures the error state model is valid. Under these assumptions, the two models are identical, as shown by the derivation of the error state model in Section 4.2.4. However, to further illustrate this equivalency a controller will be designed using the error state model and implemented on both architectures shown in Figures 4.5 and 4.4.

For the error state space model, geometry is used to compute e_1 during simulation. The geometric relationship is simply a projection of the vehicle's position onto the path:

$$e_1 = \begin{bmatrix} (X_{veh} - X_{ref}) & (Y_{veh} - Y_{ref}) \end{bmatrix} \begin{bmatrix} -\sin(\psi_{ref}) \\ \cos(\psi_{ref}) \end{bmatrix} \tag{4.16}$$

This is used to provide the output error state e_1 and the difference between the vehicle's yaw and the reference yaw is used to provide e_2. The derivatives (if needed) are approximated online with digital derivatives

and low pass filters to attenuate high-frequency noise.

Control Design

A simple Single-Input, Single-Output controller is selected as the structure of the controller to be designed. This simplifies the control design. The output of the error state model is selected as the approximation of the lateral error at some look-ahead distance: $e_{LA} = e_1 + d_s e_2$, where d_s is the look-ahead distance. This output can be translated to the global position model by setting that model's output to $y + d_s\psi$ and the references to $y_{ref} + d_s\psi_{ref}$. Therefore, the error that the controller responds to is $e_{LA} = (y_{ref} - y) + d_s(\psi_{ref} - \psi) = e_1 + d_s e_2$.

The parameters that represent the model in CarSim for the full-size SUV with Electronic Stability Control and Antilock Braking System are presented in Table 4.2. The resulting numerical transfer functions are:

$$P_Y(s) = \frac{655.41(s^2 + 4.247s + 7.624)}{s^2(s^2 + 8.424s + 25.25)} \qquad P_e(s) = \frac{655.41(s^2 + 4.247s + 7.624)}{s^2(s^2 + 8.424s + 25.25)} \tag{4.17}$$

Where $P_Y(s)$ is the transfer function from δ_f to $y + d_s\psi$ and $P_e(s)$ is the transfer function from δ_f to e_{LA}. The transfer functions are identical, so there is no difference in designing a controller with one model over the other (under the previously stated assumptions).

Parameter	Name [units]	Value
$C_{\alpha,f}$	Front cornering stiffness [N/rad]	153465
$C_{\alpha,r}$	Rear cornering stiffness [N/rad]	153541
m	Vehicle mass [kg]	2691
J	Vehicle yaw inertia [kg-m^2]	5502.39
l_f	Front axle location [m]	1.4303
l_r	Rear axle location [m]	1.7097
d_s	Look-ahead distance [m]	15
V_x	Vehicle speed [m/s]	30

Table 4.2: CarSim SUV Bicycle Car Model Parameters

The H-infinity mixed sensitivity technique is used to design a dynamic output feedback controller. The purpose of this design is to generate a controller that performs well enough to compare the influence of different implementations. To retain some practicality in the design, the frequency weight on the Youla

transfer function, $Q(s)$, is shaped to limit the actuator effort. Then the weights on the sensitivity transfer function, $S(s)$, and the complementary sensitivity transfer function, $T(s)$, are tuned to achieve an acceptable $T(s)$ bandwidth while maintaining robust frequency shapes. The resulting design and weights are shown in Figure 4.7.

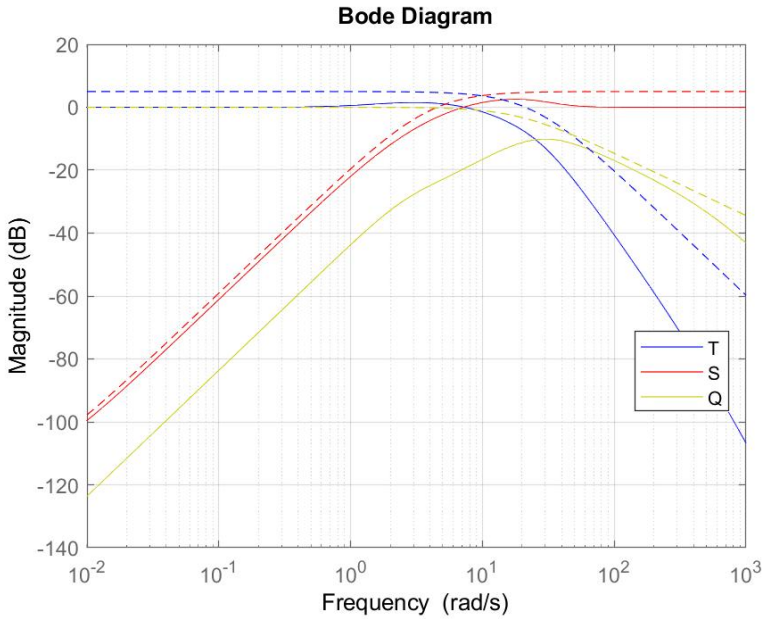


Figure 4.7: Magnitude of Bode Plot for the H-infinity loop shaping controller. The dashed lines are the inverse of the frequency weights.

The resulting controller has nine poles and eight zeros. The controller is reduced to a fourth-order transfer function using the balanced truncation technique and state space to transfer function transformations. The resulting transfer function is:

$$K(s) = \frac{0.0093121(s + 10.24)(s + 4.558e - 05)(s^2 - 3725s + 6.103e06)}{(s + 4388)(s + 0.01235)(s^2 + 49.14s + 1498)} \tag{4.18}$$

The controller is then converted to a discrete-time state-space model using the Bilinear Transform.

Simulation Results

The simulation results are shown in Figure 4.8, where “LTI Lookdown" refers to the controller implemented using the error state architecture and “Global Coord" is the controller implemented using the global position state architecture. The specific reason for why this controller is labeled look-down as opposed to look-ahead

is addressed later in Section 4.3.3.

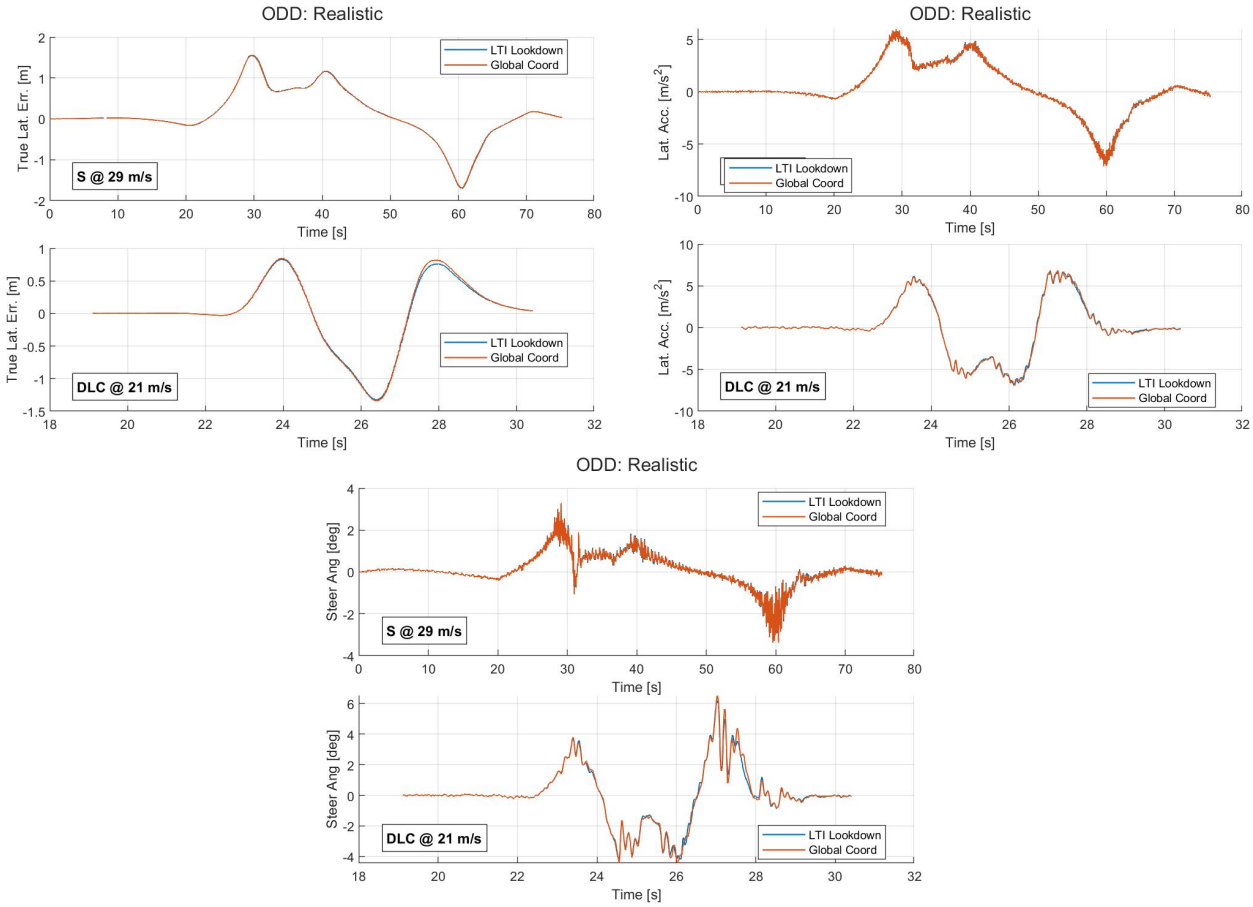


Figure 4.8: Simulation results of the H-infinity controller implemented in both architectures

The results show nearly identical performance despite the differences in implementations (mainly the coordinate transformations). This shows that controllers following the previously described assumptions designed with the error state model can also be implemented in the architecture shown in Figure 4.4.

4.3.2 Prefilter Design

The previous Subsection argued that there is little difference in using the global position model or the error state model under a set of assumptions. It is worth noting now that those assumptions were overly restrictive. In reality, additional degrees of freedom should be used if the increase in control complexity is acceptable. The same applies to the use of a non-causal pre-filter. This Subsection addresses the design of such pre-filters.

A prefilter can be placed before the feedback control architecture as shown in Figure 4.9, where $\bar{y} =$

$e_1 + d_s e_2$, r is the reference pose, and $(\cdot)^*$ represents prefiltered signals.

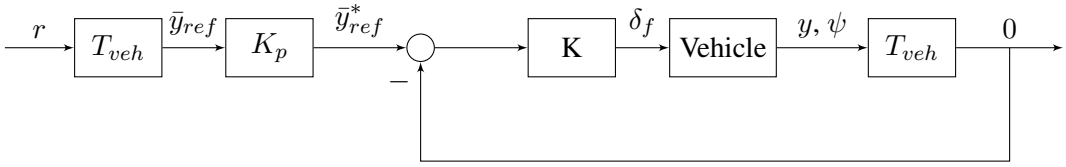
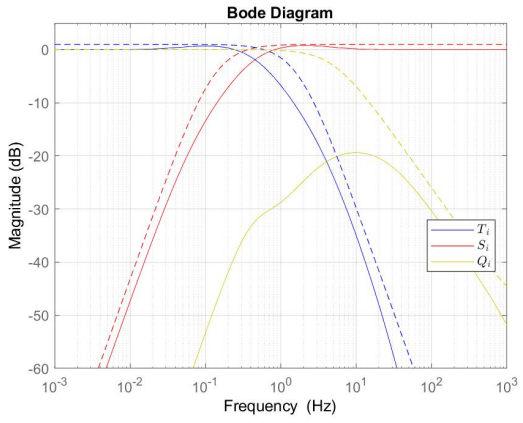


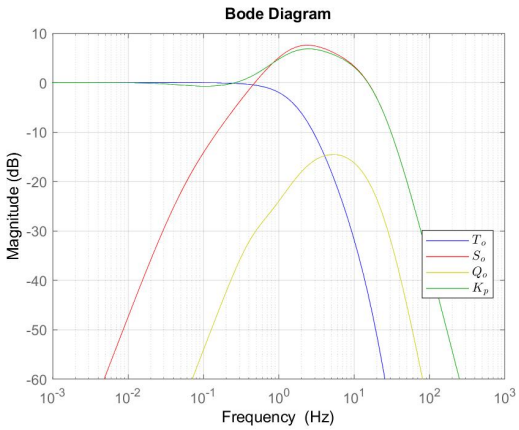
Figure 4.9: Two DoF control block diagram of global Y model with prefilter K_p

Control Design

To show the tradeoffs when using a prefilter K_p, the inner controller, K, is detuned such that the bandwidth of the transfer function $T_i(s) = \frac{\bar{y}(s)}{\bar{y}^*_{ref}(s)}$ is approximately 0.5 Hz. The same frequency weights and model parameters in Section 4.3.1 are used, but shifted to a lower frequency. Then they are lowered slightly to enforce smaller peaks in S_i and T_i. This detuning results in an increase in robustness and disturbance rejection as well as a decrease in actuator effort. However, these benefits come at the cost of reducing reference tracking performance. Note that the velocity for the controller is set at 30 m/s but will be tested at different velocities. In this sense, robustness against model uncertainty (the velocity parameter) will be shown in these simulations. The resulting design is shown in Figure 4.10a.



(a) Magnitude of Bode Plot for the detuned H-infinity loop shaping controller. The dashed lines are the inverse of the frequency weights.



(b) Magnitude of Bode Plot for the Prefilter and outer closed loop transfer functions.

To improve tracking performance, K_p is computed from $K_p = T_{des} T_i^{-1}$, where T_{des} is a desired shape for $T_o = T_i K_p$. T_i^{-1} is improper so additional poles are placed at a suitably high frequency. Therefore $T_o \approx T_{des}$. The bandwidth of T_o is tuned to be 2 Hz. If the bandwidth of T_o is significantly greater than

T_i's, there will be increased actuator effort because $Q_o = \frac{u}{\bar{y}_{ref}} = QK_p$. Therefore, this trade-off is balanced by selecting the bandwidth of 2 Hz. Furthermore, $S_o = \frac{e^*_{LA}}{\bar{y}_{ref}} = S_i K_p$ will have a significantly larger peak. Such a design is shown in Figure 4.10b.

This tradeoff results from position tracking control with the steering wheel angle. The transfer function from the steering angle to the lateral position contains two integrators. Internal stability requires that $T_i(0) = 1$ and $\frac{dT_i}{ds}(0) = 0$ (a result that can be found by applying the interpolation methods of [61]). If these were not conditions for internal stability, then T_i could be tuned to have non-unity dc gain, which is then corrected by the K_p. This would provide more freedom to balance the bandwidth of T_o and T_i.

Simulation Results

The above design is simulated to show that even though the bandwidth of T_o is significantly greater than T_i, little to no performance increase is achieved. This is shown in Figure 4.11. The linear model of the system, T_o, predicts that the system will behave with a high bandwidth. However, the simulation results show otherwise. To explain this mismatch, reconsider Figure 4.4 and notice that the references are nonlinear functions of the current vehicle's pose (the transformation to a coordinate frame fixed on the current vehicle's pose is a nonlinear function of the current vehicle's pose). Therefore, the vehicle pose dynamics are inadvertently embedded in the references. The closed-loop response affects the references. How might we analyze this effect? We can create a new error state model with these prefilters references similar to what is presented in Section 4.2.4. However, this analysis increases the complexity of tuning the prefilters to achieve suitable improvements in tracking performance.

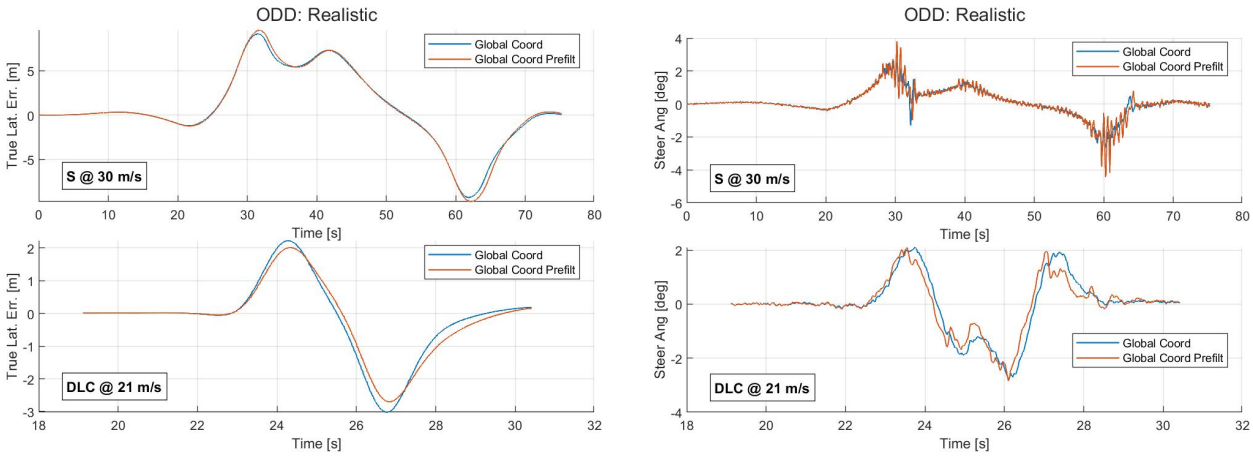


Figure 4.11: Simulation results of the 1 DoF and 2DoF Controllers

Another potential remedy to this issue is to take advantage of two observations: (1) the yaw angle reference is independent of the current pose, and (2) the output to be tracked is a linear combination of lateral position and yaw angle. The first observation comes from the realization that the error the controller responds to is $e_1 + d_s e_2$, where $e_2 = \psi_{ref} - \psi$. Therefore, instead of using the transformed reference and vehicle yaw angles to compute e_2, we may use the untransformed yaw angles. For instance, the yaw angles can be quite large, but as long as e_2 is small, then the small angle approximation is valid. The second observation leads to interpreting the model used for control design as one that interpolates between y and ψ. From this perspective, the controller closes the loop around both the yaw angle and the lateral position. If it is desired to decouple the references from the current vehicle pose, and the yaw angle reference is independent of the current pose, the yaw angle reference should be tracked more closely than the lateral position reference. In essence, the gain on yaw should be larger than the gain on lateral position so that the transformed reference $\bar{y}_{ref}$ is less dependent on the current pose.

This is illustrated by a second simulation experiment where $d_s = 60$ (four times its previous value). In addition to this gain change, the controller is also tuned so that the inner loop's bandwidth is approximately 1 Hz (twice the previous bandwidth). The prefilter is tuned to achieve an outer loop bandwidth of 3 Hz (1.5 times the previous bandwidth). This change in tuning makes the impact of the different implementations more obvious in Figure 4.11. In all, we should expect that the controller's performance improves because of the increase in the inner loop bandwidth. We should also expect that the difference between the prefilter and non-prefilter controllers becomes more substantial than in Figure 4.11.

The simulation results are shown in Figure 4.12. They show two controller simulations: a controller with prefilter ("Global Coord Prefilt") and a controller without prefilter ("Global Coord"). The "Global Cord Prefilt" lateral error is much lower in the DLC maneuver than the "Global Coord" controller. The predicted improvement is partially realized. However, there is an increase in error for the "S" maneuver. This shows that this controller does a poorer job in rejecting the curvature or reference yaw rate (which dominates in the "S" maneuver).

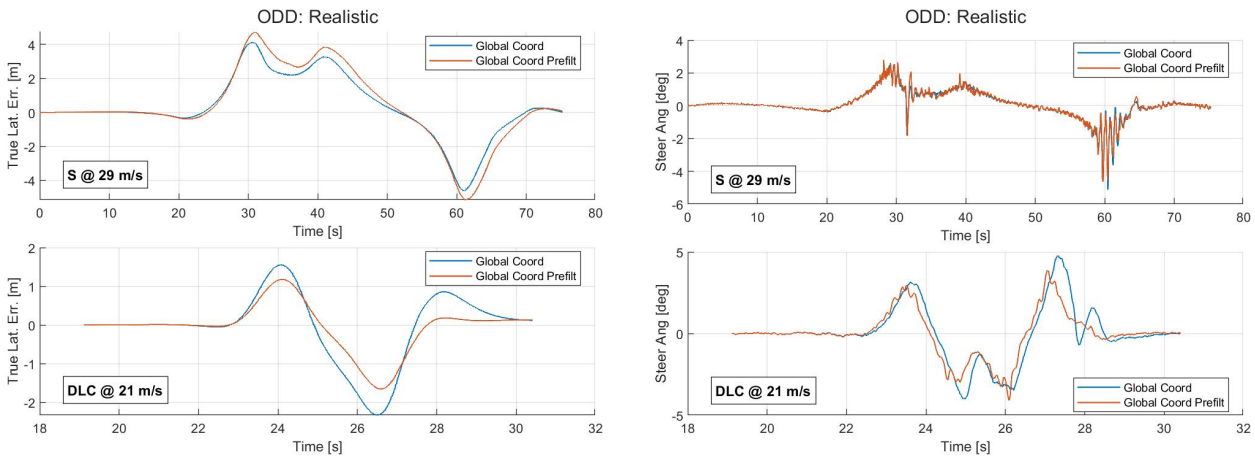


Figure 4.12: Simulation results of the 1 DoF and 2DoF Controllers where $d_s = 60$

To show that the reference yaw rate has a significant impact on the lateral error for the “S" maneuver, we will design a feedforward controller that will improve the overall tracking. This feedforward controller is a static gain on the reference yaw rate. The gain is tuned through iterative simulations. The simulation results are shown in Figure 4.13. This shows a lateral error that is approximately half what is shown in Figure 4.12. This dramatic increase in performance shows the significance of the yaw rate on the controller's performance. However, the prefilter controller is still more sensitive to the curvature than the non-prefilter controller.

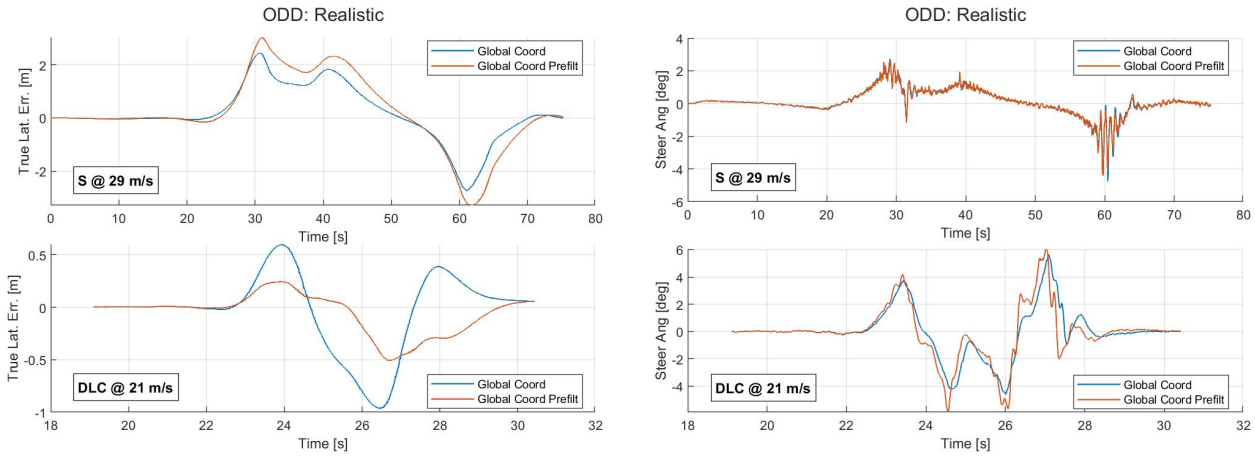


Figure 4.13: Simulation results of the 1 DoF and 2DoF Controllers where $d_s = 60$ and a feedforward controller is used to improve yaw rate tracking

Finally, we may assume that the references are known within a reasonable time horizon (up to a few time samples) because the Automated Driving System software typically generates plans that can be several seconds long. We can take advantage of this preview with a non-causal prefilter. Several control techniques

leverage preview in a non-causal way [185, 137, 139, 87, 35]. These techniques are typically some form of discrete-time model inversion with some strategy for handling non-minimum phase zeros in the model. Non-minimum phase zeros can occur when discretizing plants with all stable zeros. Unstable zeros can also be a result of using the Pade approximation for time delay. This makes these strategies important in practice.

The performance improvement in non-causal prefilters comes from the model inversion and the way that the resulting prefilter is made proper. The model inverse technique typically results in an improper system. To make the prefilter proper, pure delay is added to the filter. However, this delay is canceled out by time shifting forward the input to the prefilter, thus requiring knowledge of future references. The Zero-Phase Error Tracking Control (ZPETC) is a relatively simple technique that involves replacing unstable zeros in the complementary sensitivity transfer function (discretized) with stable versions of those zeros [185]. To do this, the coefficients of the unstable zero portion are flipped. For instance, let $T(z) = \frac{B(z)}{A(z)} = \frac{B_s(z)B_u(z)}{A(z)}$, where $B_s(z)$ contains the stable zeros and $B_u(z)$ contains the unstable zeros. Let $B_u(z) = b_{un}z^n + b_{u(n-1)}z^{n-1} + ... + b_{u0}$. Now, $T^{-1}(z) \approx \frac{z^{-q}A(z)B_u^f(z)}{B_s(z)(B_u(1))^2}$. Where, $B_u^f(z) = b_{u0}z^n + b_{u1}z^{n-1} + ... + b_{un}$ and q is the number of additional poles needed to make T^{-1} proper. Note that the difference between $B_u(z)$ and $B_u^f(z)$ is that the coefficients are flipped.

The ZPETC technique is now applied to the lateral motion control problem. The control design is left unchanged from the 1 Hz inner loop and the yaw rate feedforward controller. What differs from previous controllers is that the prefilter is now a ZPETC. The simulation results are shown in Figure 4.14. There is an improvement in performance between the results shown in Figure 4.14 and 4.13. The ZPETC improves performance in the “S" maneuver, where the causal prefilter degrades performance. The dynamic response in the DLC maneuver for the ZPETC is quite different from both the causal prefilter and the causal 1DoF controller. One tradeoff of the ZPETC is that the commanded steering angle is noticeably noisier. This is likely due to the full model inversion. The causal prefilter mitigates this issue by adding poles that act like a low-pass filter.

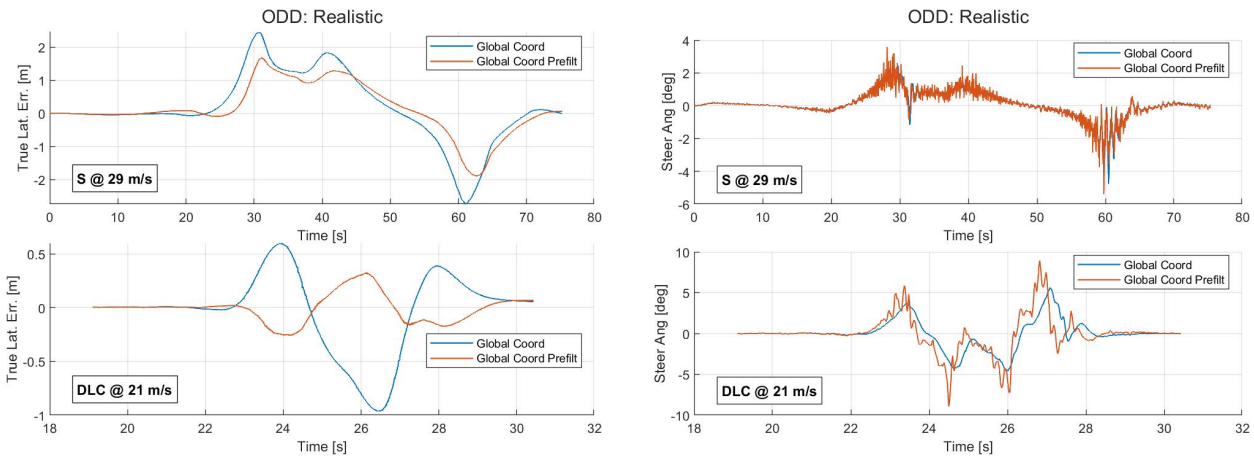


Figure 4.14: Simulation results of the 1 DoF and 2DoF Controllers where $d_s = 60$, a feedforward controller is used to improve yaw rate tracking, and a ZPETC is used as a prefilter

One concern in model inversion techniques is robustness. If the model is not accurate in the frequency ranges in which the model inversion control acts, then poor performance can be expected. In lateral motion control problems, a concern is the change in environmental conditions. Driving in rain, for instance, reduces the road-tire friction and can decrease the vehicle's gain between the steering wheel angle and lateral acceleration. To explore the robustness of the ZPETC, another simulation is run on the Rainstorm ODD. This uses decreased road friction and increased wind speeds. The simulation results are shown in Figure 4.15. The results show that the ZPETC improves performance for the "S" maneuver, but does not for the DLC maneuver. It is also important to point out that there appears to be a drift in lateral error in both maneuvers. This is a result of placing more emphasis on the yaw angle and thus reducing the controller's ability to track lateral position and reject lateral wind forces.

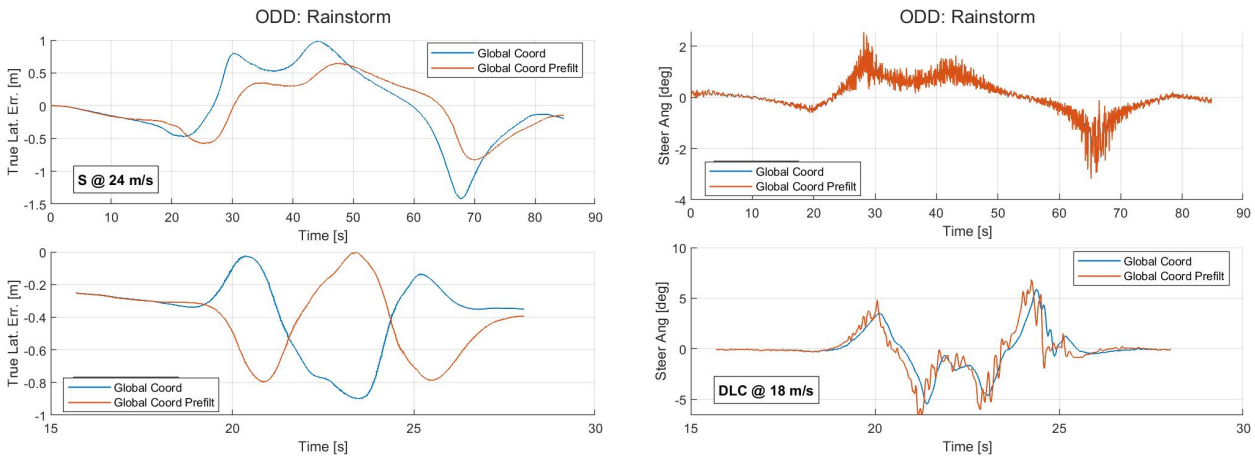


Figure 4.15: Simulation results in Rainstorm ODD of the 1 DoF and 2DoF Controllers where $d_s = 60$, a feedforward controller is used to improve yaw rate tracking, and a ZPETC is used as a prefilter

This Subsection explored several types of prefilters and controller designs. It was shown first that the coupling of the current pose and the coordinate transformations limit the effectiveness of the prefilter. To mitigate this, more weight was placed on the yaw angle as the output of the system. This resulted in increased improvements for the causal prefilter and the ZPETC. However, it was shown that this comes at the cost of reducing tracking performance and wind disturbance rejection capabilities in the Rainstorm ODD.

4.3.3 Look-ahead in the Error State Model

The previous Subsection explored some methods that leverage the unique opportunities presented by the global position model. In this Subsection, the unique benefits of the error state model will be shown. When the output of the error state model is set as a linear combination of e_1 and e_2 such that $e_{LA} = e_1 + d_s e_2$, the term e_{LA} can be physically interpreted as the lateral error at a look-ahead distance d_s.

When the controller uses information from the reference at a distance ahead of the vehicle (requiring access to future references and therefore is a non-causal controller), it is called a look-ahead system. A controller that uses measurements made at the vehicle's center of gravity is called a look-down controller (requiring access only to current references and is thus a causal controller).

These options come from the model used to analyze the input-output behavior of the system that will be controlled. One can begin with the A and B matrices for the model in Equation 4.14. When the path is straight (or has a sufficiently large radius, R) the lateral error at a look-ahead distance is $y = e_1 + d_s e_2$, where d_s is some look-ahead distance in meters (assuming e_2 is small enough for a small angle approximation).

The geometry of this situation is depicted in Figure 4.16.

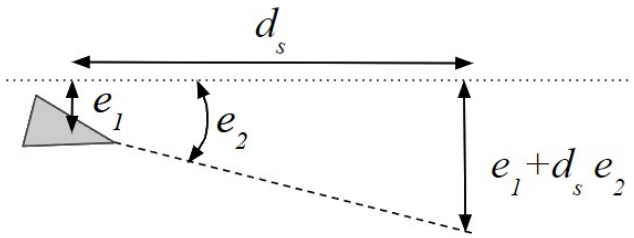


Figure 4.16: Geometry of the lateral error when the reference path is straight. The references are the dotted, horizontal line.

This output enables linear systems analysis and control. However, it is a poor approximation when R is small. When the path is a constant radius, R, the true look-ahead cross-track error is:

$$e_{LA} = e_2 d_s + e_1 + (R - \sqrt{R^2 - d_s^2}) \tag{4.19}$$

This is derived from the geometry depicted in Figure 4.17.

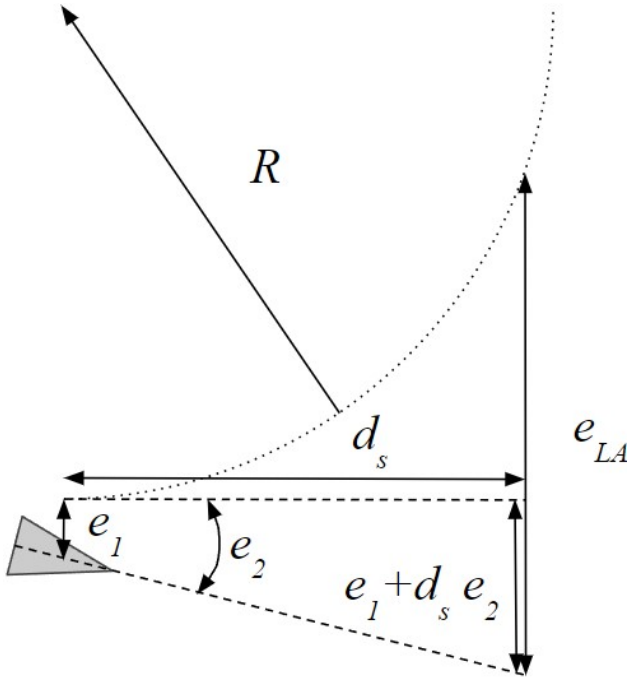


Figure 4.17: Geometry of the lateral error when the reference path is an arc of constant radius. The references are dotted arc.

Equation 4.19 is well approximated by $e_2 d_s + e_1$ when $R >> d_s$.

If the system has access to the future trajectory (at least for a distance up to d_s) then e_{LA} can be computed

online. A simple way to do this (when the references are in the form of a path) is to use a closest-point algorithm to find the point on the path that is closest to the vehicle. Once that point is found, the path is traversed for the look-ahead distance. The point at the end of the look-ahead distance is used as a reference. This method is more accurate than e_{LA} because it does not make any assumption on constant radius.

Control Design

This Subsection proposes three alternative implementations of a dynamic output feedback controller that uses cross-track error at a look-ahead distance. They are as follows:

1. Look-ahead: Use forward path information to measure the system's output for feedback control.
2. Look-down: Use measurements of the path at the vehicle's center of gravity with the linear approximation of $e_{LA} \approx e_1 + d_s e_2$ to compute the system's output (no curvature information is used).
3. Look-down+: Use center of gravity measurements and curvature measurements with Equation 4.19 to compute the system's output.

The plant transfer function will be held constant for all three implementations. This is because these variations are viewed as controller implementation variations and not controller design variations. The resulting plant and controller will be the same for all three variations. This isolates the influence of the implementation so that any resulting changes in performance can be directly attributed to the implementation and not the control design. The plant transfer function is derived using the linear approximation of e_{LA} as the output.

Simulation Results

The simulation results are shown in Figures 4.18 and 4.19.

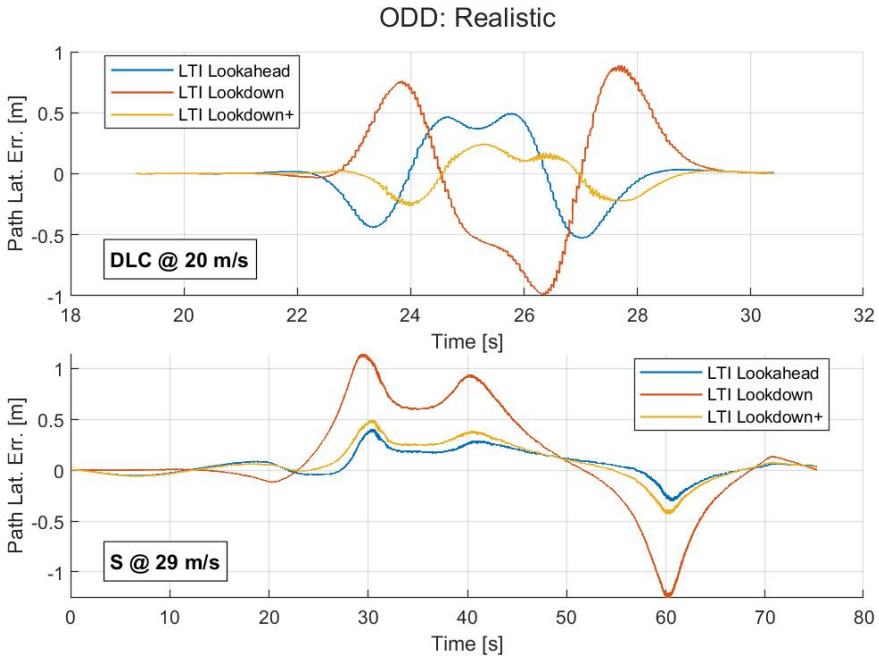


Figure 4.18: True Lateral Error for Realistic ODD

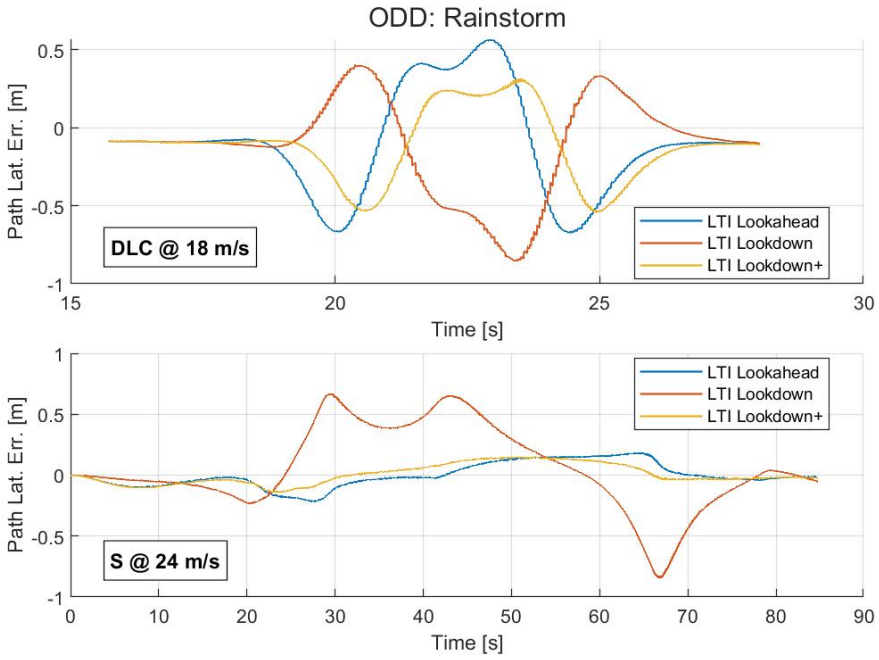


Figure 4.19: True Lateral Error for Rainstorm ODD

The Look-down+ controller performs very similarly to the Look-ahead controller. This indicates that the additional terms of Equation 4.19 reduce the causal filter's error by approximately half. The difference in performance between the Look-down and the Look-down+ performance indicates how important

the accuracy of the look-ahead error is for the controller performance. Furthermore, the observation that the Look-down+ controller achieves lower error in the Rainstorm ODD indicates that this additional term improves robustness against vehicle modeling uncertainties.

4.3.4 References as a Path or a Trajectory

One often neglected nuance in lateral control for Automated Driving Systems is the difference between a path and a trajectory. Such a distinction becomes important when designing the architecture of the system that will safely control the Automated Driving System. A common approach, as described in Chapter 1, is to separate the functionalities of the driver into Perception, Planner, and Control modules. This modularity promotes designing the Control module almost independently of the other modules. However, one design choice that must consider both the Planner and Control module is the Control module's interface. This represents the way information is shared between the Planner and Control modules.

Planner algorithms provide references in the form of discrete or continuous references. They may be parameterized by a spatial parameter (typically the distance along the curve s) or by time [77, 45, 183]. For instance, when the ego vehicle must overtake a slower vehicle that is in the same lane and ahead of the ego vehicle, the Planner is responsible for providing a path that brings the ego vehicle around and in front of the slower vehicle. If the output is discrete and parameterized by a spatial variable, the planner output, r, may be an array of global X and Y positions and orientations, ψ, that are parameterized by the distance along the curve, s. The result is: $r(s) \in \mathbb{R}^{m\times 4}$. Where m represents the number of discrete points in the path and each row is a vector of $[X, Y, \psi, s]$.

When the references are a path, an additional step is required before passing a reference to the controller. The controller requires a reference with which the feedback can be compared. One approach is to add an intermediary module that receives the reference path and the current vehicle's position. Then the reference path is searched for the point that is closest to the vehicle's position (a 2-dimensional Euclidean distance minimization problem). Look-ahead controllers can then be implemented as described in the previous Section.

However, a path can be converted to a trajectory by assigning temporal information to each point of the path. One way to do this is to generate a velocity profile at which the path should be traversed. The resulting path would contain pose and velocity information. Using simple kinematic relations, a reference time can then be computed for each reference point. The result is $r(t) \in \mathbb{R}^{m\times 6}$, where the reference is now

parameterized by time instead of s. Each row is now a vector of $[X, Y, \psi, s, \dot{s}, t]$. To pass this as a reference to the Controller, the Planner need only use the waypoint that corresponds to the current time.

Both $r(s)$ and $r(t)$ can be implemented as a continuous function or a 2-D array. When $r(t)$ is sampled, it is done at fixed time intervals to produce a 2-D array. The effects of this can be well understood with digital systems approaches (ie. Nyquist-Shannon sampling theorem). when $r(s)$ is sampled, it is done at fixed space intervals (values of s a uniform distance apart). For both a trajectory and a path, if the sampling is too coarse (too much time between trajectory samples or too much distance between path samples) the resulting references will not be smooth, potentially deteriorating control performance and passenger comfort. The determination of a suitable sampling rate is task-dependent.

These distinctions are often neglected in much of the literature on controller design. The effects of specific implementation methods have not yet been studied. To do so, we will design a set of simulations. To maintain repeatability we will use the simulator presented in Chapter 3. Two controllers: a dynamic output feedback and state feedback; will be simulated on two maneuvers: Double Lane Change and “S" maneuver. Each combination will be simulated when the references are given as a path and when they are given as a trajectory. When the references are a path, the static path is sampled every 20 cm. The closest point is used for the state feedback controller (a look-down controller). The point which is the look-ahead distance along the curve in front of the closest point is used for the dynamic output feedback controller (a look-ahead controller). The path is converted to a trajectory by imposing a velocity profile. The result is that the trajectory does not have a consistent time sampling. To account for this, the trajectory array is linearly interpolated online, enabling any desired time sampling. The selected time sample matches the sampling rate of the controller: 50 Hz. To use look-ahead distance with trajectory-based references, instead of using the current time, we use the reference at $t = t_0 + d_s/U_x$. t_0 is the current time, d_s is the look-ahead distance, and U_x is the current longitudinal velocity.

The controller performance is shown in Figure 4.20. The results show little change in the performance as a result of the different implementations of the references. In the Dynamic Output Feedback (called “LTI Look-ahead"), the performance is indistinguishable.

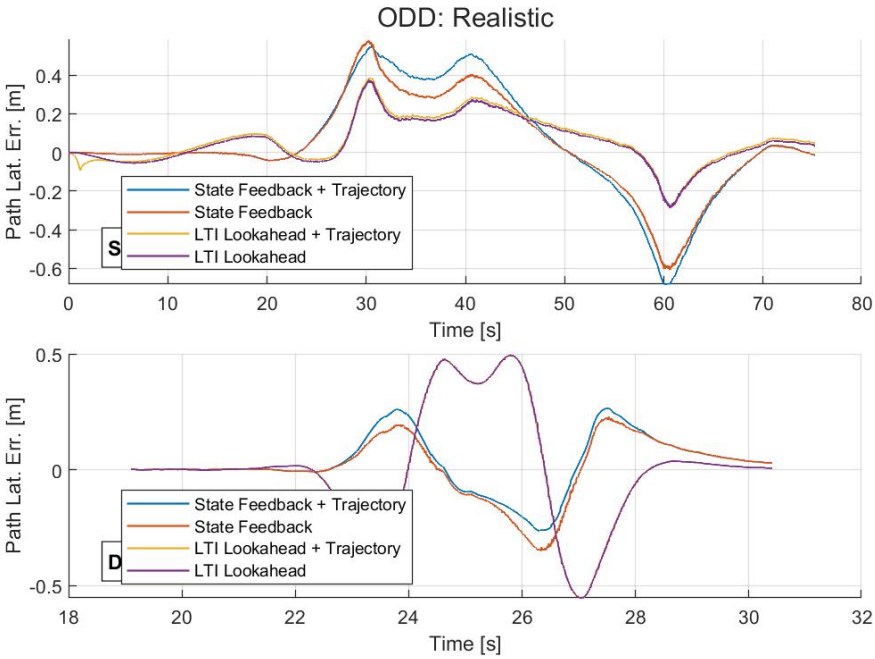


Figure 4.20: True Lateral Error for Trajectory and Path References in the Realistic ODD

This lack of performance change between reference implementations suggests that there is little reason to prefer using a trajectory over a path. However, these results are under the assumption of good lateral and longitudinal control performance.

The control architecture, presented in Chapter 3, separates lateral motion control from longitudinal control. The longitudinal motion control tracks the distance along the path, while the lateral motion control tracks the lateral position of the path or trajectory. The results in Figure 4.20 also correspond to low longitudinal tracking error. If we detune the longitudinal controller or add a bias to the longitudinal controller then we may produce different results. The results shown in Figure 4.21 show more drastic differences between controllers that use a trajectory and a path. These results occur when the longitudinal reference (the distance along path) is offset by 10 meters. The magnitude of this offset is rather extreme, but is selected so that the difference in performance is easily discernable.

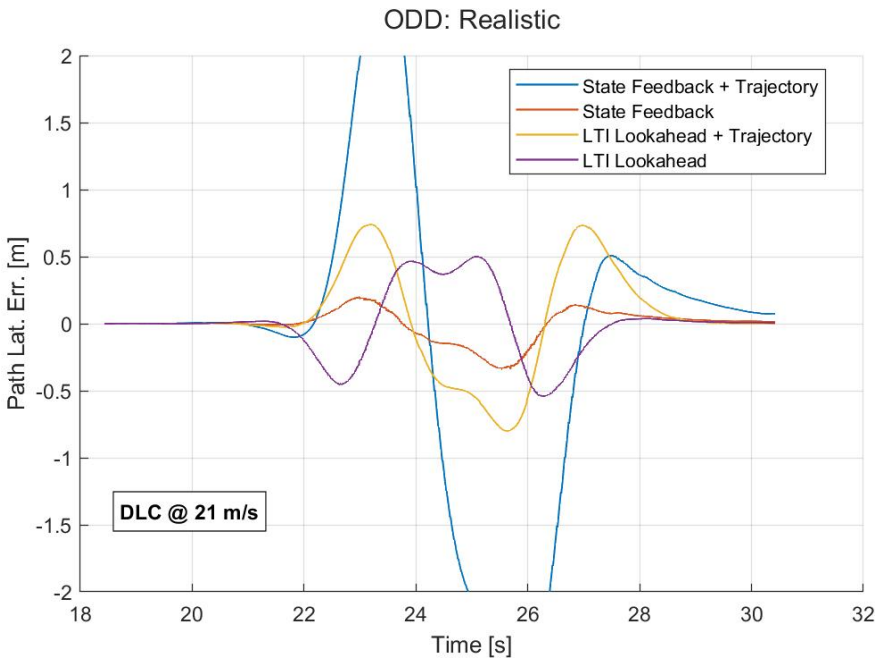


Figure 4.21: True Lateral Error for Trajectory and Path References Under Increased Longitudinal Error

Figures 4.21 and 4.20 show that the performance of lateral motion controllers that use a reference path are not affected by a bias in longitudinal tracking performance. However, the controllers that use a reference trajectory drastically change performance when there is bias in the longitudinal tracking performance. This shows that using a reference path can better decouple lateral and longitudinal motion control than using a reference trajectory.

4.4 Conclusion

This Chapter performed a series of analytical and simulation experiments to show the effects of modeling choices that need to be made when designing and implementing a lateral motion controller. Several models were presented in Section 4.2, but only two were simulated because they present simple and effective choices for lateral motion control design. The models investigated are the global state model and the error state model.

The two models are shown to be identical under a set of assumptions in Subsection 4.3.1. These assumptions are then relaxed to show the potential benefits of using a prefilter with the global state model in Section 4.3.2. The additional degree of freedom does improve performance, but only if the coupling between the references and the current vehicle pose is reduced. Furthermore, a non-causal prefilter in the form

of a Zero-Phase Error Tracking Controller [185] is developed to show how preview can be used to improve tracking performance.

Having demonstrated the challenges and benefits when using the global state model for control, Subsection 4.3.3 investigates the potential benefits of using the error state model. Three different implementation choices are proposed for the dynamic output feedback control design with the error state model. The look-down implementation is the worst-performing controller, while the look-down+ and look-ahead controllers perform very similarly. This similar performance shows the significant impact a simple, nonlinear term can have on the control performance. The terms can make a causal controller behave similarly to a non-causal controller.

The final implementation detail investigated is to provide the reference from the Planner to the Controller as a path or a trajectory. Simulation results show that the performance is not significantly influenced. However, using a reference trajectory increases the coupling between longitudinal and lateral motion controllers. Simulation experiments show that when the lateral motion controller uses a reference trajectory its performance can degrade with a bias in longitudinal tracking error.

All of these modeling and implementation details arise in practical implementations of lateral motion controllers. They have not yet been isolated and studied in the literature. Investigating the impacts of these implementations is difficult to do analytically, so simulation experiments are conducted. However, these simulation results are highly specific to the selected controllers. The conclusions drawn from them may not generalize to controllers not studied. However, the selection of controllers studied are dynamic output feedback controllers as well as state feedback controllers, which constitute a wide variety of control designs. In all, this Chapter contributes useful analysis and evidence for understanding the effects of nuances that often go overlooked.

Chapter 5

Linear Parameter Varying Control Based on Interpolation Conditions

This chapter develops a new Linear Parameter Varying dynamic output feedback controller design technique. This new method is based on the Interpolation Conditions for internal stability. The original method, used for Linear Time Invariant systems, is extended to Linear Parameter Varying systems by exposing the transfer function coefficients as functions of scheduling parameters. This technique is then demonstrated to enable high-performance control design by applying it to an Automated Driving System for lateral motion control. The control design is validated using the Simulator developed in Chapter 3. The results for two aggressive maneuvers show that the controller performs well in normal and low-friction environments with severe external disturbances. Real-world test results collected by implementing an LPV lateral motion controller on a 2019 Jeep Grand Cherokee are then presented to empirically evaluate the technique's performance on several maneuvers.

5.1 Introduction

This chapter addresses the problem of designing a gain-scheduled dynamic output feedback controller. The use of gain-scheduled control for Linear Parameter Varying (LPV) systems has been popular for decades [17, 20]. Before the work of Shamma [169], much of the control design was based on analyzing the Linear Time Invariant (LTI) model at fixed values of one or more scheduling parameters. This is referred to as a frozen-parameter version of the LPV model. However, this form of analysis overlooks the fact that the stability of the LPV system is not always a direct extension of its frozen-parameter version [169, 166]. The scheduling parameter(s) dynamics can play a significant role in the stability of the full system. Following this realization, various control methods emerged with rigorous stability proofs [98].

Many techniques are based on finding a Lyapunov function that guarantees stability across all parameter variations. Assuming a particular structure of the Lyapunov function allows one to formulate the search for such a function as a Linear Matrix Inequality (LMI) [8, 9, 10, 7]. However, the resulting optimization problem involves an infinite number of LMIs. One way to address this is to constrain the parameter variations to a polytope [10]. By constraining to a polytope a convexity argument can be made so that controllers need only be computed at the polytope's vertices. A summary of other approaches to use LMIs in LPV control can be found in [20]. In general, LMIs play a critical role in the post-modern control theory [58]. A notable work to mention is [18], which presents a comparative analysis of some of these techniques applied to the problem of path tracking by an Automated Driving System (ADS).

One tool that has been shown to benefit the design of LPV controllers is the Youla-Kucera (YK) parameter [136, 194, 135]. In [195, 196, 131], a variable, Q, is shown to parameterize all stabilizing controllers of a plant P. YK parameterization is popular in optimal control because it can be used to find H-infinity and H-2 solutions [5]. Many applications of YK parameterization use doubly coprime factorizations of the plant [188].

However, the original development was closely related to mathematical interpolation theory [197]. Specifically, Interpolation Conditions can ensure the computation of a stabilizing controller [61]. Proofs and reviews of these Interpolation Conditions can be found in [91, 119, 61]. The benefit of these Interpolation Conditions is that the resulting design process can be performed manually. When designing a control system manually a great amount of insight can be gained about the system. For instance, the trade-off between performance and actuator effort is often readily apparent.

This chapter shows how the Interpolation Conditions can be used to compute a gain-scheduled controller for an LPV system. Until now, these conditions were only applied to LTI systems [67, 12, 13, 80]. This control technique will be used to solve the problem of path-tracking with an ADS. This problem has been studied extensively with LPV control techniques [123, 17, 18, 64, 54]. This chapter will show that this technique is useful because it:

1. Permits manual control design;

2. Computes a controller that is symbolic so that can be efficiently implemented;

3. Leverages frequency domain design;

The Interpolation Conditions alluded to earlier are presented in Section 5.2. This section will introduce the assumed control structure and the concepts of internal stability. Then, in Section 5.3 the method will be extended to compute a gain-scheduled controller. This method is applied to design a path-following controller in Section 5.4. The example uses a high-fidelity co-simulation of Simulink and CarSim developed in Chapter 3. The computed controller will be shown to achieve good tracking performance ($<$ 0.6m lateral error) in two environmental conditions: dry and blizzard. Two different maneuvers will be studied to show that the resulting controller works well on all dynamically-feasible maneuvers. The first maneuver is based on a collision avoidance scenario. The second maneuver is based on a highway overtaking scenario. Finally, concluding thoughts and future work are discussed in Section 5.6.

5.2 Interpolation Conditions

Consider the system shown in Figure 5.1. P is the plant transfer function (we drop the Laplace variable to simplify notation). C is the controller transfer function. A well-designed controller will ensure that the reference signal, r, is tracked by the plant output signal, y, with reasonable limits on the actuator command, u.

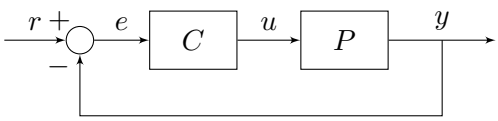


Figure 5.1: Feedback control architecture

Internal stability is necessary for any LTI controller. Internal Stability is achieved when the following four transfer functions are stable [119]:

$$\frac{1}{1+PC}\begin{bmatrix} 1 & P \\ C & PC \end{bmatrix} = \begin{bmatrix} S & PS \\ Q & T \end{bmatrix} \tag{5.1}$$

where S is the sensitivity transfer function, Q is the YK transfer function, and T is the complementary-sensitivity transfer function. The relationship $S + T = 1$ motivates the names of S and T.

Conditions for Internal Stability are well known and succinctly presented in [119] and [15]. These conditions require that T and S interpolate to specific values determined by the plant's poles and zeros. The first Interpolation Condition is presented in [119] as a design constraint to ensure good tracking characteristics.

The full development of the Interpolation Conditions is covered with more mathematical rigor in [188]. A more approachable summary is [61]. The Interpolation Conditions are:

Theorem 5.2.1 *Let p be a pole of the return ratio $L = PC$, of multiplicity a_p, and z be a zero of L, of multiplicity a_z, then*

$$T(p) = 1 \qquad S(p) = 0 \tag{5.2}$$

and

$$\frac{d^k}{ds^k}T(p) = 0 \qquad \frac{d^k}{ds^k}S(p) = 0 \qquad 1 \leq k \leq a_p - 1 \tag{5.3}$$

while

$$T(z) = 0 \qquad S(z) = 1$$

and

$$\frac{d^k}{ds^k}T(z) = 0 \qquad \frac{d^k}{ds^k}S(z) = 0 \qquad 1 \leq k \leq a_z - 1 \tag{5.4}$$

Furthermore, since Q parameterizes S and T, these conditions can also be applied to Q. Applying Theorem 5.2.1 to SISO systems is well described in [61]. Alternative control methods of utilizing Q solve coprime equations [188]. Applying Theorem 5.2.1 for manual synthesis of a stabilizing controller is an iterative process. The engineer typically begins with a low-order Q or T and checks that the Interpolation Conditions are met and all transfer functions are proper. If they are not, the complexity is increased slightly and repeated. Sometimes meeting all Interpolation Conditions can be challenging, but this iteration often provides insight into fundamental trade-offs in the stabilization problem. Refer to [61] for a more in-depth discussion of Interpolation Conditions to Single-Input Single-Output (SISO) and Multi-Input and Multi-Output (MIMO) systems.

5.3 Parameter Varying Extension

The previous section introduced the Interpolation Conditions for LTI systems. Directly extending Theorem 5.2.1 to LPV systems is not straightforward since it is derived from the fact that pole locations determine the system's stability properties. For LPV systems, this is not always the case [169]. It is only true for the frozen-parameter interpretation of an LPV model under the assumption that the parameter is constant or slowly varying.

Therefore, in extending the Interpolation Conditions to LPV systems, the parameter dynamics are assumed to be sufficiently slow. The classical gain scheduling method now is to select a set of operating points. However, rather than choosing a finite set of points, the Interpolation Conditions can be used to derive a controller transfer function that is a symbolic function of the scheduling parameters (as opposed to a numerical polytopic model that is produced in [10]). Leveraging this strength of the Interpolation Conditions allows the resulting controller to be a continuous function of the scheduling parameters. This process begins with an LPV state space model:

$$\dot{x} = A(\theta)x + B(\theta)u$$
$$y = C(\theta)x + D(\theta)u$$

where θ denotes the scheduling parameters.

The plant transfer function matrix, $P(\theta)$, is therefore,

$$\begin{aligned} P(\theta) &= \frac{Y}{U} \\ &= C(\theta)(sI - A(\theta))^{-1}B(\theta) + D(\theta) \end{aligned}$$

where Y is the Laplace transform of the system's output signal y, and U is the Laplace transform of the system's input signal u.

This translation to a transfer function is the reason for assuming slowly varying parameter dynamics. Without it, this transfer function could not be defined so trivially. Furthermore, this step is critical in enabling the frequency domain analysis of the model, the controller, and the resulting closed-loop system.

Expanding the plant transfer function exposes that the poles and zeros are allowed to be functions of θ:

$$P(\theta) = \frac{b_0(\theta)s^n + b_1(\theta)s^{n-1} + ... + b_n(\theta)}{a_0(\theta)s^n + a_1(\theta)s^{n-1} + ... + a_n(\theta)}$$

Applying the Interpolation Conditions may result in T, S, Q, and C becoming functions of θ.

Once a suitable controller transfer function, $C(\theta)$, is found, it must be implemented in a digital computer on a real system [16, 109].

One challenge of designing a gain-scheduled controller is the need to tune the controller at nominal values throughout the scheduling parameter ranges. This control technique mitigates this tuning effort by exposing the parameter-dependent terms in the transfer functions. Thereby, resulting in a parameter-dependent controller that is formulated as a transfer function. To implement, the transfer function can be converted to a state space model and the resulting controller is now a state space model with terms that are functions of the scheduling parameter. Therefore, different controllers do not need to be designed at each nominal condition and then interpolated. Instead, the controller's parameters are tuned as functions of the scheduling parameters.

Furthermore, this method makes no assumption on the controller's dependency on the scheduling parameters. In some LPV techniques, the controller must be an affine function of the scheduling parameters [10]. This provides a less conservative control design solution. However, it can make analyzing the closed-loop stability difficult.

In this chapter, we establish stability by experimentally demonstrating the controller's performance in Section 5.4. Future research may focus on applying the theorems developed in [94, 135] (and references therein) to connect Interpolation Conditions to YK-based gain-scheduling or controller switching.

5.4 Lateral Motion Control

This section demonstrates the simplicity of the previously explained extension of Interpolation Conditions to LPV systems. The example studied in this section will design a path-tracking lateral motion controller for an ADS. More specifically, the task is to design a controller that commands an appropriate steering angle so that the vehicle follows a given path in various weather conditions and vehicle speeds with less than 80 cm of lateral position error.

This problem can be framed as a feedback control problem in which the plant (the vehicle) takes a road wheel angle (or a steering wheel angle that is statically converted to a road wheel angle) and whose output

must asymptotically approach some reference path. Two sources of parametric uncertainties are considered: (1) vehicle velocity and (2) road friction.

A simple and useful model for an ADS tracking a path is the error state model developed in [160]. The bicycle car model captures the vehicle dynamics. The references are modeled as a point mass. The output, $y = e_1 + d_s e_2$, is the lateral error at some look-ahead distance, d_s. The input to the model is the road wheel angle δ_f. The model also considers the reference yaw rate, $\dot{\psi}_{des}$, as a disturbance since the states are in error coordinates and should asymptotically approach zero.

The LPV state space model is:

$$
\begin{aligned}
\dot{x} &= A(U_x)x + B_1\delta_f + B_2(U_x)\dot{\psi}_{des} \\
A(U_x) &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_f+C_r}{mU_x} & \frac{C_f+C_r}{m} & \frac{-aC_f+bC_r}{mU_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{aC_f-bC_r}{I_zU_x} & \frac{aC_f-bC_r}{I_z} & -\frac{a^2C_f+b^2C_r}{I_zU_x} \end{bmatrix} \\
B_1 &= \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{aC_f}{I_z} \end{bmatrix} \qquad B_2(U_x) = \begin{bmatrix} 0 \\ -\frac{aC_f-bC_r}{mU_x} - U_x \\ 0 \\ -\frac{a^2C_f+b^2C_r}{I_zU_x} \end{bmatrix} \\
y &= C(U_x)x = \begin{bmatrix} 1 & 0 & f(U_x) & 0 \end{bmatrix} x \\
\dot{\psi}_{des} &= \frac{U_x}{R_{ref}} \qquad x = \begin{bmatrix} e_1 & \dot{e}_1 & e_2 & \dot{e}_2 \end{bmatrix}^T
\end{aligned}
\tag{5.5}
$$

where $f(U_x)$ is the function that gives the look-ahead distance d_s and allows the control engineer to schedule d_s on U_x.

5.4.1 Achievable Performance of LPV

Before proceeding it is worth pausing and reflecting on the question: What is the benefit of applying LPV control to this problem? One way to answer this question is to frame it as a robust control design problem and quantify the best achievable performance with the H-infinity norm. Such a framing can also be done

with other forms of cost functions and control architectures. However, in this Chapter, we are primarily focused on SISO Dynamic Output Feedback controllers. Therefore, the controller is assumed to be either an LPV or an LTI SISO continuous-time dynamic system, $K(s, U_x)$ or $K(s)$, respectively. The plant is studied in a frozen-parameter fashion to evaluate performance. This can also be done in an LPV way by using the quadratic stability and LMI-based robust stability [157, 33]. However, such an analysis is conservative [157].

We begin by setting up the mixed sensitivity H-infinity minimization problem. In this problem, the closed-loop system requirements are embedded in frequency weightings. Determining these frequency weightings to achieve acceptable performance in reality can be challenging. However, for the purpose of analyzing the achievable performance of an LTI and an LPV controller, the requirements need not be complicated. They only need to represent a reasonable design. To this end, the sensitivity transfer function $S(s)$ is weighted by $W_1(s)$ and the Youla transfer function, $Q(s)$, is weighted by $W_2(s)$:

$$W_1(s) = \frac{s + 0.924}{1.778s + 0.00924}$$
$$W_2(s) = \frac{s + 123.4}{0.01s + 39.02}$$

$W_1(s)$ is designed to enforce a reasonable steady-state error and a minimum closed-loop bandwidth of about 0.1 Hz. $W_2(s)$ is designed to limit low-frequency actuator effort and to make the controller's high-frequency response roll off at -20 dB/decade above 10 Hz. Using these weights, the model presented in Equation 5.5 (neglecting the yaw rate reference as a disturbance for simplicity), and the parameters in Table 5.1, the robust control problem is well-defined.

Table 5.1: Full Size SUV Parameters

Variable	Name	Value
m	Mass	2691 [kg]
L	Wheelbase	3.14 [m]
a	CG Position from Front	1.4303 [m]
b	CG Position from Rear	1.7097 [m]
I_z	Rotational Inertia	5502.39 [kg$\times$m^2]
C_f	Front Corner Stiffness	153465 [N/rad]
C_r	Rear Corner Stiffness	153541 [N/rad]

In nominal conditions $\mu_f = 1$, and in the extreme, $\mu_f = 0.5$. The task is now to minimize the largest

singular value for the closed loop system from the control references, r, to the outputs of $W_1(s)$ and $W_2(s)$ throughout the ranges of the uncertain parameters U_x, C_f, and C_r. To reduce the number of uncertain parameters, both C_f and C_r are assumed certain but are scaled by an uncertain friction coefficient μ_f. Now the three uncertain parameters U_x, C_f, and C_r are reduced to two uncertain parameters: U_x and μ_f. The optimal LTI H-infinity controller minimizes the H-infinity norm of the augmented closed-loop transfer function matrix:

$$M_{lti}(s, U_x, \mu_f) = \begin{bmatrix} W_1(s)S(s, U_x, \mu_f) \\ W_2(s)K(s)S(s, U_x, \mu_f) \end{bmatrix}$$

The LPV controller forms the augmented closed-loop transfer function matrix:

$$M_{lpv}(s, U_x, \mu_f) = \begin{bmatrix} W_1(s)S(s, U_x, \mu_f) \\ W_2(s)K(s, U_x)S(s, U_x, \mu_f) \end{bmatrix}$$

This problem can be solved by Mu-Synthesis. However, since there are several complications when dealing with real-valued uncertain parameters [24] the solution may be conservative. Instead, we will solve the H-infinity mixed sensitivity problem when $U_x = 25$ and $\mu_f = 1$ (the Nominal Condition). Then, we compute $|M_{lti}(s, U_x, \mu_f)|_\infty$ when the plant's uncertain parameters are varied throughout their ranges. The analysis of this first design is shown in Figure 5.2. The grid spans the full uncertain parameter range and shows how the performance changes throughout the uncertain parameter space. When the H-infinity norm is less than 1, the requirements are satisfied. Figure 5.2 shows that the controller achieves robust performance through the combined ranges $U_x = (18.9, 30]$, $\mu_f = (0.5, 1.0]$, excluding the range $\mu_f = [0.5, 0.61]$ for $U_x = 18.9$. The worst performance (largest H-infinity norm) occurs at $U_x = 5$, $\mu_f = 0.5$, the parameter combination that is furthest from the nominal value).

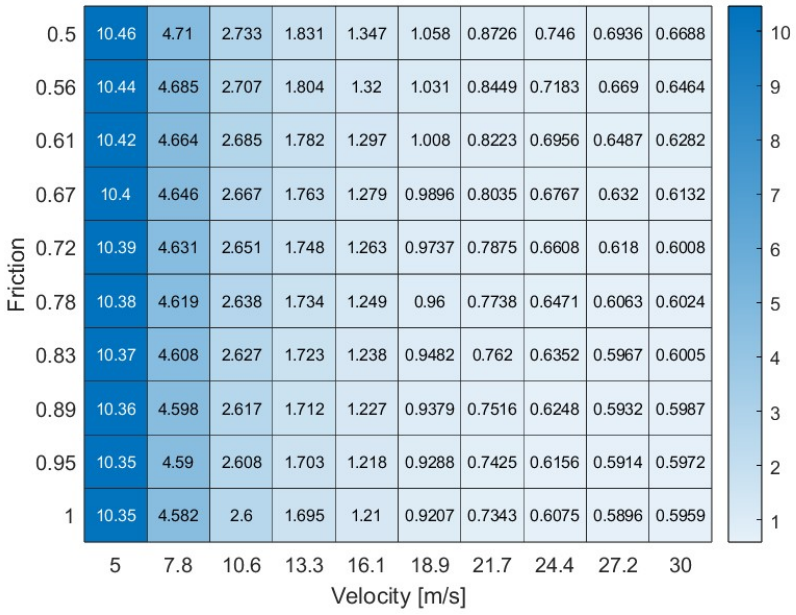


Figure 5.2: $|M_{lti}(s, U_x, \mu_f)|_\infty$ when the controller is designed at nominal conditions ($U_x = 25$,$\mu_f = 1$).

Designing the LTI controller at the plant nominal conditions may not result in the smallest $|M_{lti}(s, U_x, \mu_f)|_\infty$. To determine which parameter values result in the LTI controller that achieves the smallest $|M_{lti}(s, U_x, \mu_f)|_\infty$, the H-infinity problem can be solved (using Matlab's hinfsyn command) over the grid of parameter values. The result of this grid-search is shown in Figure 5.3. The following optimization problem is solved when $U_x^* = 5$ $\mu_f^* = 0.61$ and $\gamma^* = 0.69$:

$$U_x \in [5, 30], \qquad \mu_f \in [0.5, 1.0]$$

$$\min_{U_x, \mu_f} |M_{lti}(s, U_x, \mu_f)|_\infty = \gamma^*$$

The fact that $\gamma^* > 1$ for some parameter combinations indicates that there is no single LTI controller that achieves the robust performance requirements (as defined by $W_1(s)$ and $W_2(s)$) throughout the parameter range.

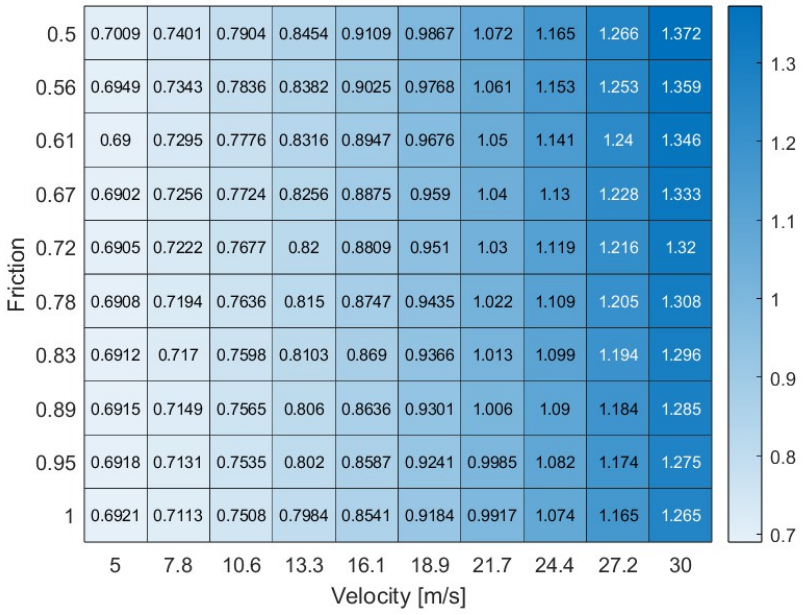


Figure 5.3: $|M_{lti}(s, U_x, \mu_f)|_\infty$ when the controller is designed at optimal conditions ($U_x = 5$,$\mu_f = 0.61$).

It is therefore reasonable to next consider gain scheduling on either U_x, μ_f or both to check if the desired performance can be met. Since μ_f is not as readily measureable as U_x, gain-scheduling will be performed on U_x. To evaluate the best possible gain-scheduled controller, the mixed sensitivity H-infinity problem will be solved at each U_x in the grid with the nominal μ_f value of 1.0. This analysis therefore assumes that the gain scheduled controller is otherwise stable and robustly performant between the gridded parameters. One constraint that is worth considering is that the LPV control technique presented in this Chapter requires that the controller be the same order. Therefore, the H-infinity problem is further restricted to using a 4th order controller (which can be solved with Matlab's hinfstruct command). The resulting robust performance is shown in Figure 5.4. All maximum singular values are below 0.68, indicating that robust performance is achieved. Furthermore, the requirements can be made more demanding (which is important since the assumed requirements are quite relaxed). A comparison of Figures 5.3 and 5.4 shows that gain scheduling can result in a significant improvement: between a 3 and 55 % improvement.

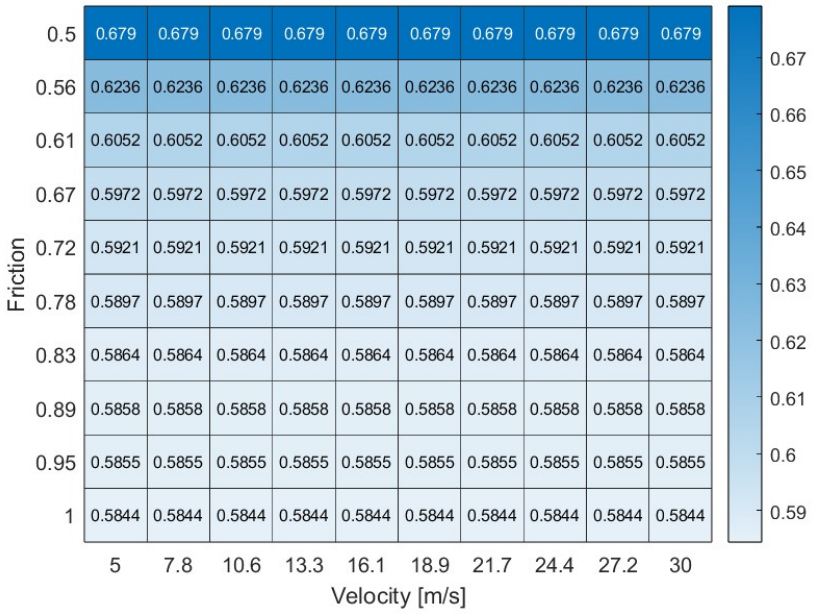


Figure 5.4: $|M_{lpv}(s, U_x, \mu_f)|_\infty$ when the H-infinity optimal controller is computed at each U_x and when $\mu_f = 1.0$.

Now that it has been demonstrated that gain scheduling on U_x can result in significant robust performance improvements, the next task is to design the controller.

5.4.2 Control Design

If the polytopic LPV techniques were applied to this model, then two parameters would have to be introduced: U_x and $\frac{1}{U_x}$ so that the model is affine with these parameters. This increases the number of scheduling parameters and both the affine and quadratic stability requirements impose conservatism. This conservatism can be reduced somewhat by finding a minimal polytope that encompasses the parameter variations [18]. This decrease in conservatism results in about 3% RMS tracking error improvement [18]. An alternative approach that has not yet been sufficiently applied to the polytopic or LFT LPV techniques is using dimensional analysis to reduce the number of scheduling parameters as is done in the scale ground vehicle research literature [97, 30, 32, 86, 84, 85].

A suitable transformation matrix can be applied to the model in Equation 5.5 to put the model into a Controllable Canonical form:

$$A(U_x) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{a_2(U_x)}{a_0(U_x)} & -\frac{a_1(U_x)}{a_0(U_x)} \end{bmatrix} \qquad B(U_x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$
$$C(U_x) = \begin{bmatrix} \frac{b_2(U_x)}{a_0(U_x)} & \frac{b_1(U_x)}{a_0(U_x)} & \frac{b_0(U_x)}{a_0(U_x)} & 0 \end{bmatrix} \tag{5.6}$$

Which corresponds to the transfer function from δ_f to $(y + f(U_x)\psi)$:

$$P(U_x) = \frac{b_0(U_x)s^2 + b_1(U_x)s + b_2(U_x)}{a_0(U_x)s^4 + a_1(U_x)s^3 + a_2(U_x)s^2} \tag{5.7}$$

The coefficients, b_i, a_i $\forall i \leq 2$ are functions of U_x. Two poles are always at the origin. For understeer vehicles, the other two poles are stable throughout the range of U_x. The error state model exposes an additional tuning parameter d_s. This may be left static, treated as an additional scheduling parameter, or set as a function of U_x. In this case, $d_s = f(U_x) = 0.8U_x$ (the value of 0.8 was found by iteratively tuning in simulation to achieve satisfactory performance throughout the velocity range).

Figure 5.5 shows the movement of the poles and zeros as U_x increases from 5 to 40 m/s. The model parameters used in this example are presented in Table 5.1. They have been identified using Grey Box identification techniques to best fit the dynamic response of the Full-Size SUV vehicle in CarSim.

Figure 5.5 shows that two poles always remain at the origin. The other two poles begin on the real axis and leave the real axis at approximately (-9,0). After this, the two poles become increasingly less damped. The two zeros follow similar behavior to the poles, except they depart the real axis at approximately (-2,0). These zeros therefore have a significant impact on the response of the system.

With two poles always at the origin, the Interpolation Conditions need not be a function of U_x. This makes it possible to design T so it is not a function of U_x. There are two conditions to meet:

$$T(0) = 1 \tag{5.8}$$

and

$$\frac{dT(0)}{ds} = 0 \tag{5.9}$$

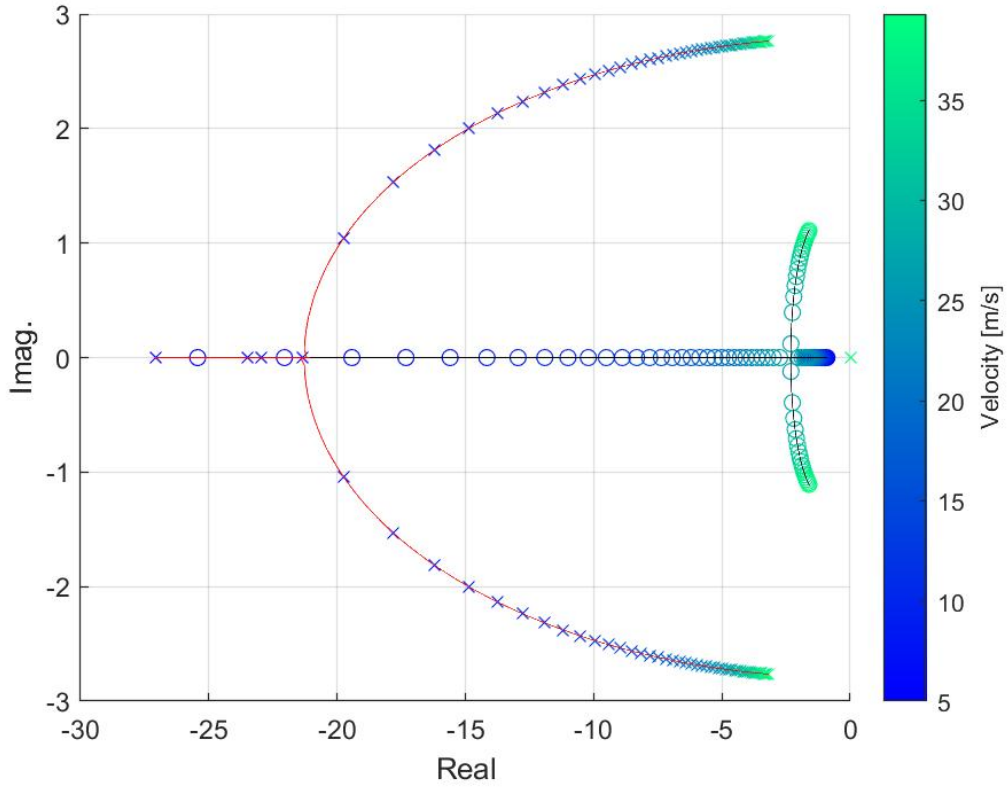


Figure 5.5: Pole-Zero Locus of Bicycle Car Model

For illustration, a simple T is chosen. However, a higher order T will be used in the actual design to provide more free parameters to tune.

Let

$$T = \frac{k\ (s\,\tau_1 + 1)}{(s\,\tau_2 + 1)^3}$$

The first Interpolation Condition is met when,

$$k = 1$$

The second Interpolation Condition is met when,

$$\tau_1 = 3\tau_2$$

Therefore,

$$T = \frac{3\,s\,\tau_2 + 1}{(s\,\tau_2 + 1)^3} \tag{5.10}$$

The control engineer is left with one free parameter, τ_2, to achieve a desired closed-loop response.

At this point, it is worth pausing and reflecting on the form of T. The second pole at the origin requires the existence of zero. The location of that zero is also a function of the poles. This is true even when T is made higher order. This zero greatly impacts the cost of decreasing robustness to achieve improved performance. This is because the placement of this zero becomes a function of the closed-loop poles to satisfy Interpolation Conditions. The control engineer has limited freedom to place this zero in a favorable location. Such an observation illustrates how the LPV Interpolation Conditions can provide added insight into the design of a stabilizing controller.

Using the relations $T = QP$, $S + T = 1$, and $Q = KS$, the resulting controller is:

$$C(U_x) = \frac{(3s\tau_2 + 1)(a_0(U_x)s^2 + a_1(U_x)s + a_2(U_x))}{(s{\tau_2}^3 + 3{\tau_2}^2)(b_0(U_x)s^2 + b_1(U_x)s + b_2(U_x))}$$

This $C(U_x)$ cancels the stable poles and zeros of $P(U_x)$ while including additional poles and zeros to achieve a desired closed-loop tracking response. Alternatively, the stable poles and zeros of $P(U_x)$ may be left in T, resulting in a simpler controller.

An added benefit of this controller is that τ_2 can be treated as a variable parameter with U_x. This allows

online adaptation of the closed-loop bandwidth, which may be used to continuously adjust the balance between high-performance control (high bandwidth) and actuator effort.

The next step is to discretize this controller. This can be done directly on the transfer function $C(U_x)$ or on the LPV state space realization of $C(U_x) = \begin{bmatrix} A_c(U_x) & B_c(U_x) \\ C_c(U_x) & D_c(U_x) \end{bmatrix}$.

5.4.3 Stability

The Interpolation Conditions guarantee the internal stability of the frozen parameter closed-loop system throughout the parameter range. However, this is not the same as the closed-loop stability of an LPV system. The techniques proposed in [169] may be used to prove stability. Alternatively, the closed-loop system may be converted to a polytopic LPV model, and the quadratic stability checked with a finite set of LMIs [10]. For the simple T proposed in Equation 5.10 the LMIs are straightforward.

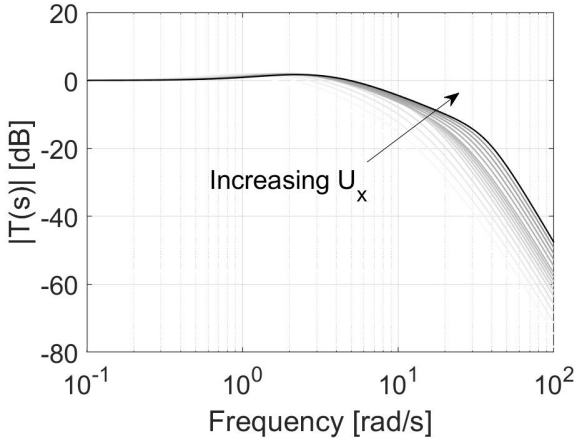
For a higher order T, as will soon presented, the LMIs may become numerically ill-conditioned and unreliable. Similar challenges may occur with the methods in [169]. Therefore, experimental validation is used to check the stability of the controller.

5.4.4 Controller Design

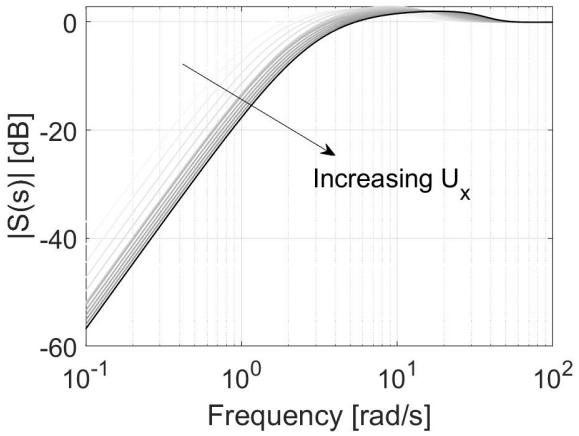
The previously proposed T is useful for illustrating how the Interpolation Conditions are met and their implications. However, its simplicity reduced the flexibility in balancing performance and robustness. A new T is therefore designed to allow more tuning parameters and faster high-frequency roll off:

$$\begin{aligned} T &= \frac{k(s\tau_1 + 1)}{(s\tau_2 + 1)p_1 p_2} \\ p_1 &= ({\omega_{n1}}^2 + 2\zeta_1\omega_{n1}s + s^2) \\ p_2 &= ({\omega_{n2}}^2 + 2\zeta_2\omega_{n2}s + s^2) \end{aligned} \tag{5.11}$$

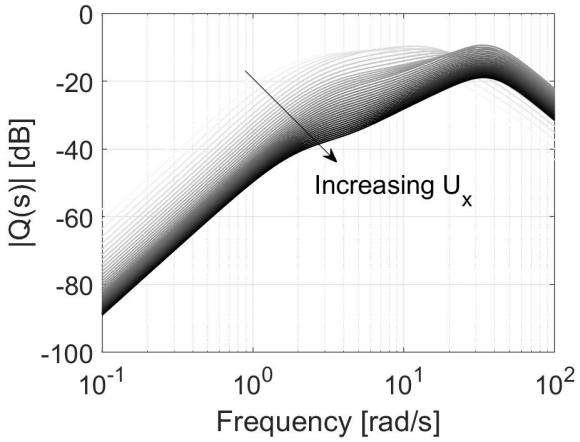
Through tuning the parameters of this system it was found that if T is left independent of U_x, unreasonable amounts of actuator effort (with respect to realistic steering actuator performances) are required when velocity is less than 20 m/s. To mitigate this, the tuning parameters (τ_2, ω_{n1}, ζ_1, etc.) are also scheduled with velocity (which indirectly schedules the closed-loop bandwidth with velocity). This tuning is per-



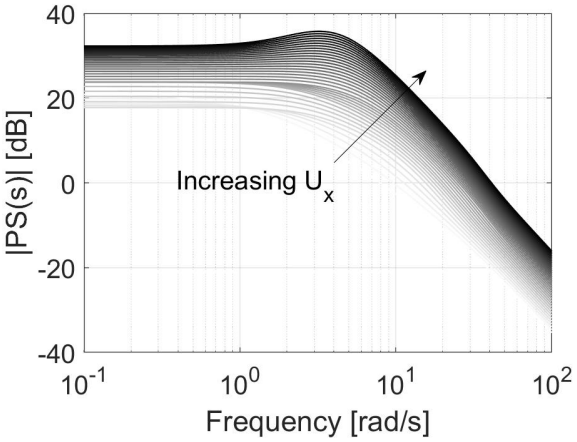
(a) Complementary Sensitivity Frequency Response



(b) Sensitivity Frequency Response



(c) Youla Frequency Response



(d) Output Sensitivity Frequency Response

Figure 5.6: Bode magnitude plot of the gang of four as U_x increases

formed over a coarse grid of U_x between 5 and 20 m/s. Then, when implemented, a linear interpolation is performed between the tuned values.

The resulting bode plots of the four transfer functions in Equation 5.1 as velocity increases are shown in Figure 5.6.

Several observations are worth noting.

- The bandwidth of $T(U_x)$ increases as velocity increases.
- $|S(U_x)|_\infty$ remains less than 5 dB.
- $|Q(U_x)|_\infty$ remains close to -10 dB until higher velocities are reached ($\approx$20 m/s). After that, as velocity increases, $|Q(U_x)|_\infty$ decreases. This is a direct result of setting $T(U_x)$ constant for velocities

above 20 m/s.

- $PS(U_x)$ has a rather large DC gain that increases with velocity.
- $|PS(U_x)|_\infty$ increases as velocity increases.

These observations suggest that this controller will be robust to parameter variations, require less actuator effort as velocity increases, but suffer from actuator disturbances such as bump steer and unmodeled steering actuator effects. The tuning of the design can be adjusted to mitigate these effects at the cost of reduced tracking performance. This is the fundamental trade-off between robustness and performance. Furthermore, it is worth noting that the look-ahead error response to curvature is captured by $S(U_x)$. Therefore, this closed-loop system should reject the reference path's curvature when its frequency content is below 2–3 rad/s. However, when this frequency content is high, the system cannot reject this.

It is also worth noting that the $Q(U_x)$ transfer function has different input and output engineering units. The input units are of the reference and the output units are of the actuator. Therefore, the DC gain of $Q(U_x)$ can be changed arbitrarily by selecting different units of measure. Evaluating $|Q(U_x)|$ only becomes meaningful when a requirement is placed on the actuator's frequency response. Thereby providing a reference for which $|Q(s)|$ can be evaluated.

5.4.5 Simulation Results

The control is tested using the simulator from Chapter 3. Two maneuvers are selected to test this controller. The first is a Double Lane Change (DLC) to represent collision avoidance maneuvers. The second is an overtaking maneuver on an S-shaped highway (S) representing aggressive highway driving. Two ODDs are tested: Realistic and Blizzard to demonstrate the robustness of the designed controller.

The simulation results for the Realistic ODD are shown in Figures 5.7 and 5.8. The peaks in the S maneuver correspond to the maximum lateral accelerations and route curvatures. The addition of a feedforward controller can improve the sensitivity to curvature. This is left out though to showcase the performance of the LPV Interpolation Controller.

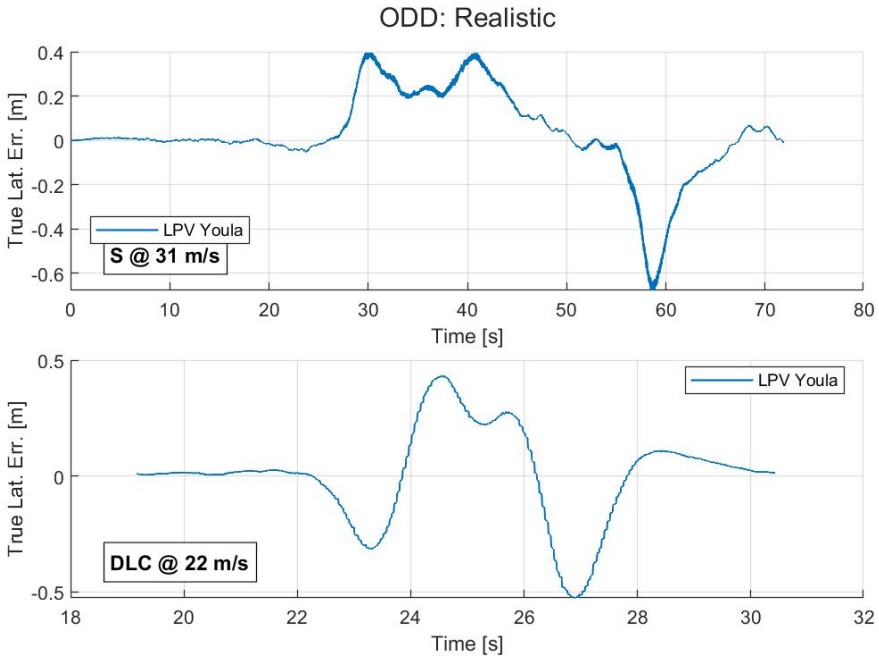


Figure 5.7: Lateral error between vehicle position and path for S and DLC maneuvers in the Realistic ODD

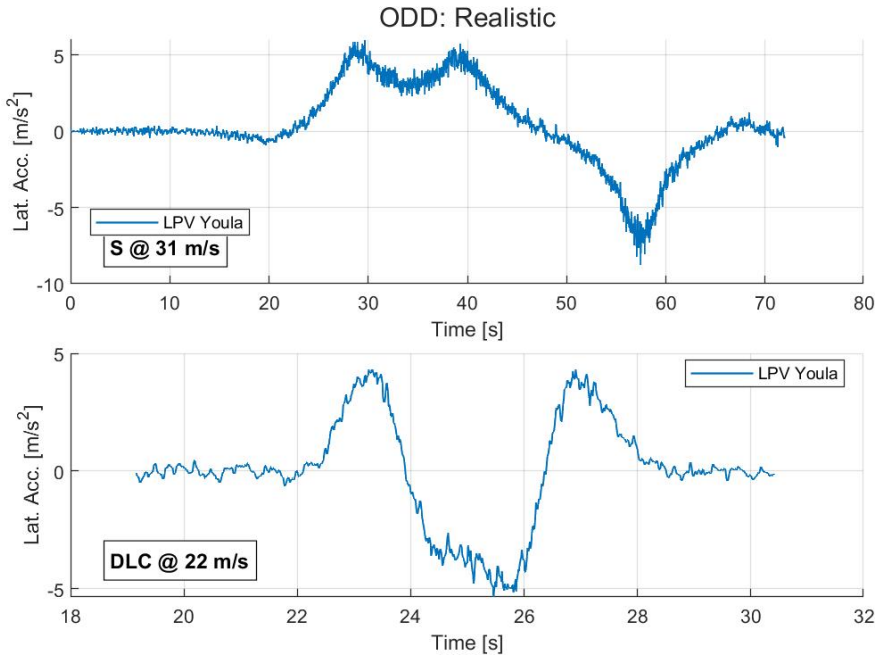


Figure 5.8: Lateral acceleration for S and DLC maneuvers in the Realistic ODD

The results in Figure 5.9 show lower lateral errors than those in Figure 5.7 despite a significant increase in external disturbances and a decrease in road friction. Figure 5.10 is included to show that despite the speed adjustment to the maneuver, the test remains aggressive and pushes the vehicle to its limit in a low-friction

environment.

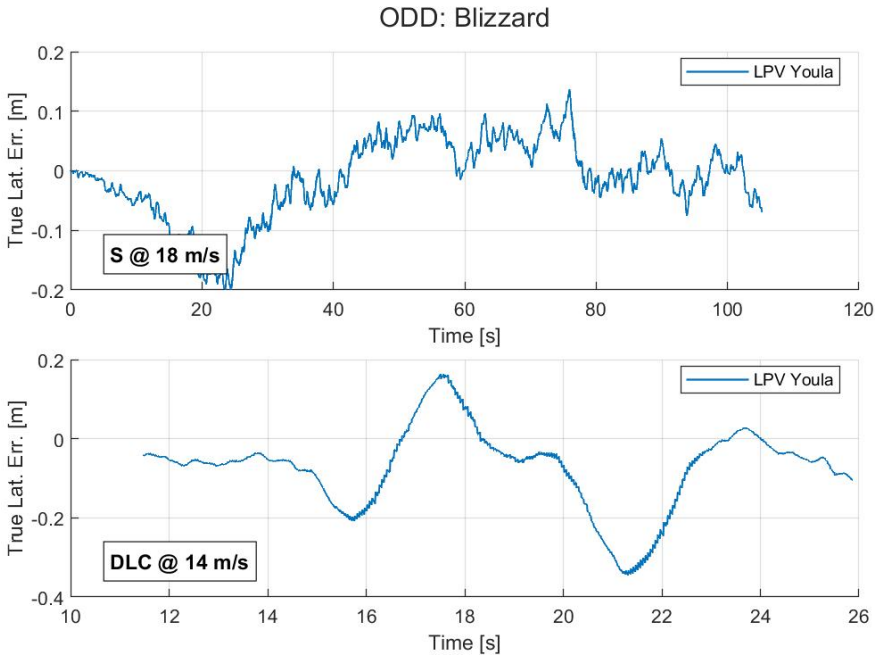


Figure 5.9: Lateral error between vehicle position and path for S and DLC maneuvers in the Blizzard ODD

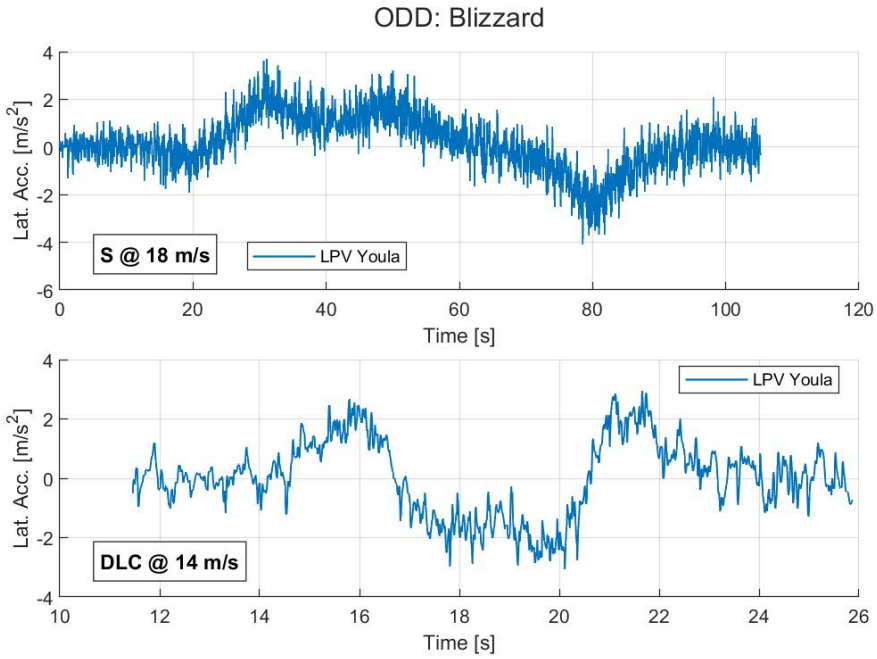


Figure 5.10: Lateral acceleration for S and DLC maneuvers in the Blizzard ODD

5.5 Gain-Scheduled Lateral Motion Control for the Jeep Grand Cherokee

5.5.1 Identifying model parameters for the Jeep Grand Cherokee

Several sets of bicycle car model parameters for the 2019 Jeep Grand Cherokee are presented in Chapters 3 and 7. They differ in their values and in the specific method in which they are estimated. The method employed in Chapter 3 is less precise and is focused primarily on demonstrating that the system identification methodology that is applied to the CarSim data can also work on data collected from a real-world vehicle. The methods presented in Chapter 7 are more precise in that they address the issue of unknown dead time (or pure time delay). Chapter 7 also presents an LPV black box model for the steering system that is assumed in series with the bicycle car model.

This Chapter is primarily concerned with the development and application of a new LPV control technique. To this end, the methods used to compute the parameters of the bicycle car model are secondary. They will be the primary focus in Chapter 7. This section's goal is simply to demonstrate that LPV control technique does not generate a destabilizing controller when applied to a real-world vehicle. Therefore, to avoid repeating what is presented in Chapter 7, the bicycle car model parameters used for control design are presented in Table 5.2. The detailed results for parameter identification is presented in Chapter 7 since they are most relevant there. However, what that chapter does not approach as explicitly as this chapter is robust performance against road surface changes. This is why this section will present the parameter identification results for when the vehicle is on a gravel road.

Table 5.2: 2019 Jeep Grand Cherokee Bicycle Model Parameters for Concrete and Gravel

Variable	Name	Concrete	Gravel
m	Mass [kg]	2300	2300
L	Wheelbase [m]	2.908	2.908
a	CG Position from Front [m]	1.4025	1.4025
b	CG Position from Rear [m]	1.5055	1.5055
I_z	Rotational Inertia [kg$\times$m^2]	2,453	2,453
C_f	Front Corner Stiffness [N/rad]	9.82×10^4	5.75×10^4
C_r	Rear Corner Stiffness [N/rad]	1.44×10^5	5.90×10^4
K_s	Steering Ratio	16	16

A series of constant steering wheel angle tests as defined in [52] are performed to identify the understeer gradient. The test results are shown in Figures 5.11 and 5.12. The analysis method for these tests is described in [52] and presented here. Figure 5.11 shows the computed vehicle curvature and corresponding lateral

accelerations for different tests. The data is filtered with a zero-phase filter to mitigate the effects of increased noise from the gravel road. The thick, dashed line is the linear fit of the data. The slope of this line is ultimately used to compute the understeer gradient (refer to equations in [52]).

Figure 5.12 shows the resulting understeer gradient computations from each test. Most tests agree that the understeer gradient is between 0.5 and 1.0. The value of 1.0 is assumed true and the remaining bicycle car model parameters (rear cornering stiffness is computed from the understeer gradient and the estimated front cornering stiffness) are estimated from chirp data. The quality of this model fit can be assessed in Figures 5.13, 5.14, and 5.15.

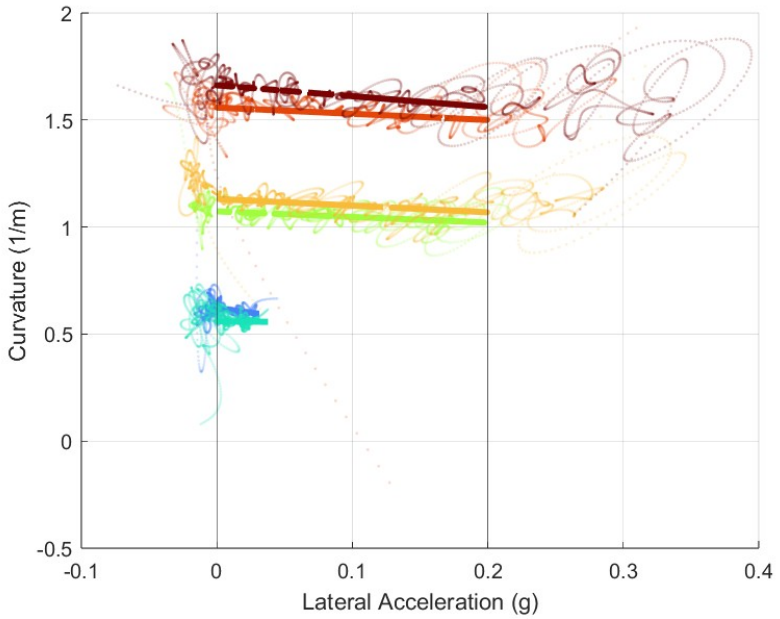


Figure 5.11: Results from constant steering wheel angle tests to identify the understeer gradient. The data is shown as scatter plots and the linear model fit to each test is shown as the thick dashed line.

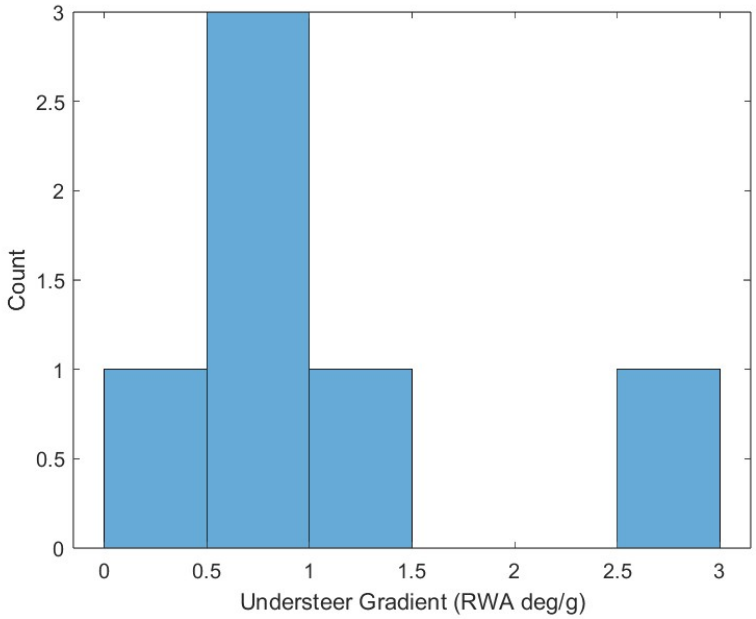


Figure 5.12: Estimated understeer gradient from each test.

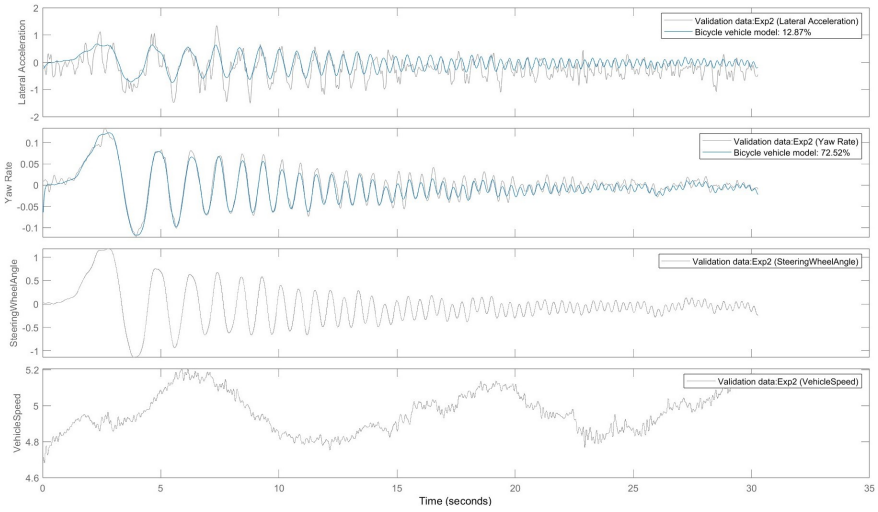


Figure 5.13: Comparison of the model's fit when a chirp torque command is used to excite the vehicle on a gravel road at 5 m/s. Units are m/s^2, rad/s, rad, and m/s for lateral acceleration, yaw rate, steering wheel angle, and vehicle speed, respectively.

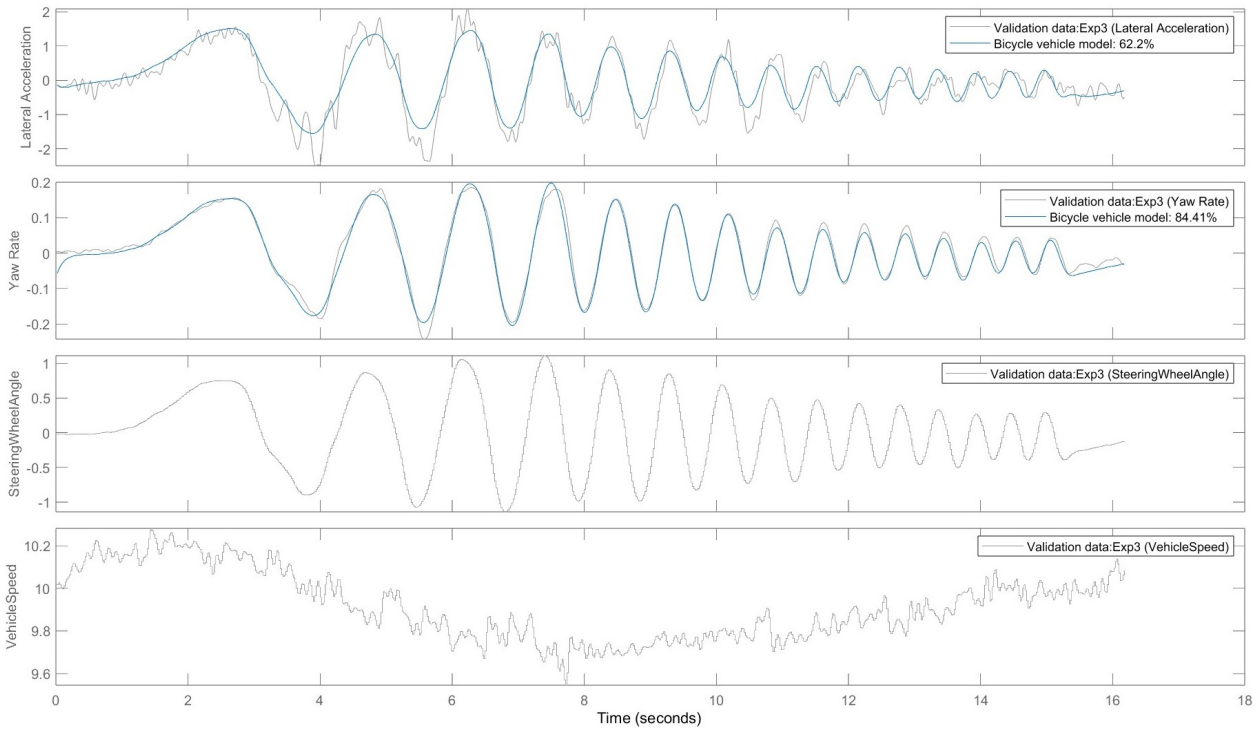


Figure 5.14: Comparison of the model's fit when a chirp torque command is used to excite the vehicle on a gravel road at 10 m/s. Units are m/s^2, rad/s, rad, and m/s for lateral acceleration, yaw rate, steering wheel angle, and vehicle speed, respectively.

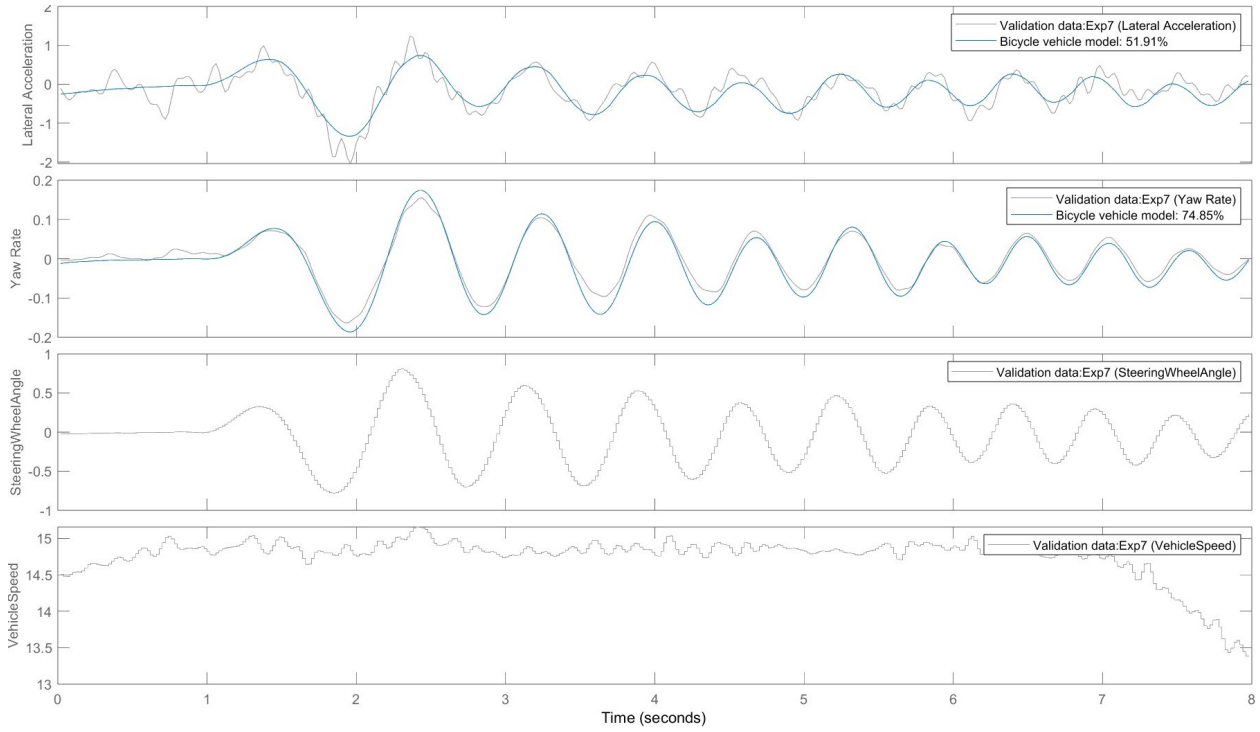


Figure 5.15: Comparison of the model's fit when a chirp torque command is used to excite the vehicle on a gravel road at 15 m/s. Units are m/s^2, rad/s, rad, and m/s for lateral acceleration, yaw rate, steering wheel angle, and vehicle speed, respectively.

5.5.2 Control Design

The control architecture for the 2019 Jeep Grand Cherokee is a cascade control system as shown in Figure 5.16. The presence of U_x in the blocks is to indicate that each controller and plant is dependent on U_x. To simplify the control design the inner control ($K_2(U_x)$ and $G_{act}(U_x)$) loop is assumed to be perfect up to a frequency 1 Hz and 2 Hz. In Chapter 7 the design of this inner controller ($K_2(U_x)$) is presented. This low bandwidth makes designing a dynamic output feedback controller $K_1(U_x)$ more challenging since the general rule is to have the outer loop 5-10 times slower than the inner loop to achieve sufficient disturbance rejection [69]. From experience tuning the bandwidth of $K_1(U_x)$ on the Jeep, if the bandwidth of $K_1(U_x)$ is too great (> 0.3 Hz), there are persistent oscillations in the steering wheel which cause discomfort. Due to this bandwidth restriction, the performance achieved with the pure feedback in Section 5.4.5 are not possible. To overcome this limitation a feedforward and feedback architecture is selected as shown in Figure 5.16. This architecture closely follows some of the work done in [121] where greater control authority is placed in the feedforward controller than in the feedback controller. In this architecture, the feedback controller's role

is to ensure robust stability and the feedforward controller's role is to achieve high performance tracking.

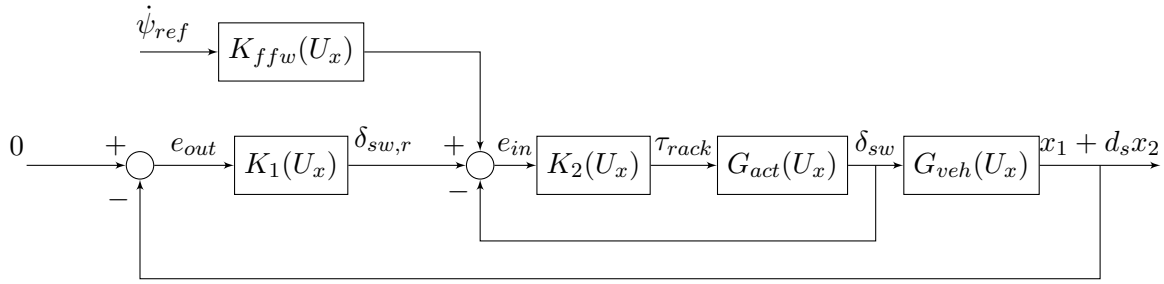


Figure 5.16: Cascade control architecture

The feedforward controller is designed based on the relationship between steering angle and yaw rate. The bicycle car model has a steady-state gain between the steering angle and yaw rate given by [75]:

$$\frac{\dot{\psi}}{\delta} = \frac{U_x/L}{1 + \frac{KU_x^2}{57.3Lg}}$$
$$K = \left(\frac{mb}{C_f L} - \frac{ma}{C_r L}\right)$$

where K is the understeer gradient, g is Earth's gravity, and 57.3 is the conversion from radians to degrees. This equation can then be used to predict the appropriate steering angle to achieve a desired yaw rate and subsequently as a feedforward control law. More specifically, the feedforward control equation is:

$$\delta_{ffwd} = \frac{1 + KU_x^2/L}{U_x/L}\dot{\psi}_{ref} \tag{5.12}$$

when K is in rad/(m/s^2). Beginning with the model's value of K and then tuning iteratively on the vehicle, a value of 0.0034 rad/(m/s^2) appears to achieve the lowest lateral error across a velocity range of 5 - 30 m/s.

The feedback controller begins with solving the Interpolation Conditions using a $T = \frac{k(s\tau_1+1)}{p_1 p_2}$ where $p_1 = ({\omega_{n1}}^2 + 2\zeta_1\omega_{n1}s + s^2)$ and $p_2 = ({\omega_{n2}}^2 + 2\zeta_2\omega_{n2}s + s^2)$. Therefore, the closed-loop tracking performance should be independent of longitudinal velocity. This T design differs from the previous T designed for the CarSim vehicle because the lower closed-loop bandwidth design does not benefit much from the increased order or the scheduling on U_x. The closed loop frequency responses are shown in Figure 5.17.

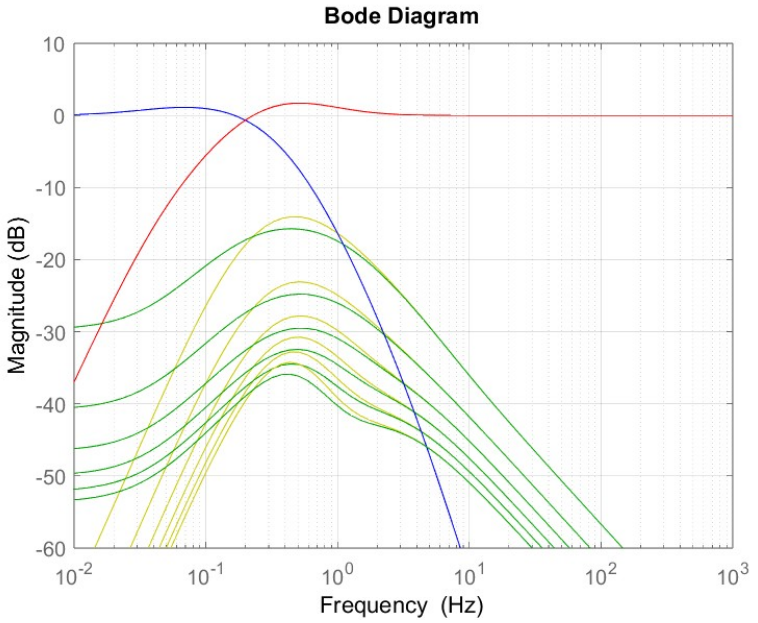


Figure 5.17: Closed loop frequency response for the lateral motion controller designed for the Jeep Grand Cherokee

This produces a controller with four states. The model's look-ahead distance, d_s, is set as $0.5U_x$ (set at 0.5 by iterative tuning on the vehicle). The implementation is restricted to the look-down type presented in Chapter 4 because the module that produces the reference path is unable to provide a sufficient look-ahead distance. Note that this also differs from the controller designed for the CarSim vehicle, which uses the look-ahead implementation because there is no limit on the look-ahead distance from the static trajectory.

After designing the controller, it must be implemented in the ROS-based control system on the Jeep Grand Cherokee. This is written in Python 2. The controller, which forms the closed-loop system frequency responses shown in Figure 5.17, is a continuous-time transfer function with coefficients (a_i, b_i) of the laplace variable, s, are functions of U_x:

$$K(s) = \frac{a_0(U_x)s^3 + a_1(U_x)s^2 + a_2(U_x)s + a_3(U_x))}{s^4(U_x) + b_0(U_x)s^3 + b_1(U_x)s^2 + b_2(U_x)s + b_3(U_x))}$$

These coefficients are often nonlinear functions of the scheduling parameter. Based on prior experience implementing digital controllers, the state space form of the controller is preferred over the transfer function form because of its improved numerical stability. Converting $K(s)$ into a state space equation can be performed with the controllable canonical state space form:

$$
\begin{aligned}
A(U_x) &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ b_3(U_x) & b_2(U_x) & b_1(U_x) & b_0(U_x) \end{bmatrix} \\
B(U_x) &= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \\
C(U_x) &= \begin{bmatrix} a_3(U_x) & a_2(U_x) & a_1(U_x) & a_0(U_x) \end{bmatrix}
\end{aligned}
\tag{5.13}
$$

The $A(U_x)$ matrix can have a poor condition number (which may indicate that the numerical output is highly sensitive to inputs) because of its structure and the values of $b_i(U_X)$. To address this a suitable state space coordinate transformation may be found. Specifically, a diagonal matrix T such that $TA(U_x)T^{-1}$ has a better condition number. Several approaches can be used to solve for T such as balanced realization [149] or a variation of more general matrix balancing as in [155].

To do this, the methods employed in the Matlab Robust Control Toolbox are used. For each integer value of U_x, the $A(U_x)$ matrix is computed. The Matlab command "balance" is then used to compute $TA(U_x)T^{-1}$. T is assumed diagonal and derived from the result. Furthermore, an additional scalar multiplier t (rounded to the nearest multiple of 2) is computed such that the norms of $C(U_x)T^{-1}\frac{1}{t}$ and $tTB(U_x)$ are close to equal (equality is not always achievable because of the desire to round t to the nearest multiple of 2). Then, T is suitably redefined: $T = tT$. The most commonly computed T throughout the range of U_x is used as the static T to transform the state space coordinates. This method improves the $A(U_x)$ condition number by several magnitudes. The T used for the controller shown in 5.17 is:

$$
\begin{bmatrix} 4096 & 0 & 0 & 0 \\ 0 & 512 & 0 & 0 \\ 0 & 0 & 64 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}
$$

Once in a transformed continuous state space form, the next step for the LPV controller is to be dis-

cretized. The state space version of the Tustin Transform [146] is used in this step. The resulting state space equations have matrices that are full and well-conditioned (even the $D(U_x)$ is nonzero since the Tustin Transform converts the strictly proper continuous-time model into a proper discrete-time model). The coefficients for each element of each state space matrix are a rational function of U_x. In this instance, they all are of the form: $\frac{\alpha_1 U_x^3+\alpha_2 U_x^2+\alpha_3 U_x^1+\alpha_4}{\beta_1 U_x^4+\beta_2 U_x^3+\beta_3 U_x^2+\beta_4 U_x+\beta_5}$ (each α_i and β_i can be unique to each state space matrix element). In Matlab, these functions of symbolic variables can be easily converted to numerical functions using the Symbolic Math Toolbox. However, this is more challenging to accomplish when implementing in Python 2. Therefore, nonlinear regression is performed on each state space matrix element throughout U_x to compute the coefficients of $\gamma_1 U_x^{-3} + \gamma_2 U_x^{-2} + \gamma_3 U_x^{-1} + \gamma_4$ (the fit quality is better when using negative powers rather than positive powers, which is a result of the appearance of U_x^{-1} in the plant model). Therefore, the computed state space matrices are an affine function of U_x^{-3}, U_x^{-2}, U_x^{-1}, and 1.

The result of this nonlinear polynomial regression for the $A(U_x)$ matrix is shown in Figure 5.18. For all state space matrices, this results in a near perfect fit. The errors for all of the matrices are shown in Figure 5.19.

This regression-based approximation effectively approximates the state space model of the LPV controller as an affine state space model:

$$\begin{aligned}
A(U_x) &\approx \hat{A} = \hat{A}_1 U_x^{-3} + \hat{A}_2 U_x^{-2} + \hat{A}_3 U_x^{-1} + \hat{A}_4 \\
B(U_x) &\approx \hat{B} = \hat{B}_1 U_x^{-3} + \hat{B}_2 U_x^{-2} + \hat{B}_3 U_x^{-1} + \hat{B}_4 \\
C(U_x) &\approx \hat{C} = \hat{C}_1 U_x^{-3} + \hat{C}_2 U_x^{-2} + \hat{C}_3 U_x^{-1} + \hat{C}_4 \\
D(U_x) &\approx \hat{D} = \hat{D}_1 U_x^{-3} + \hat{D}_2 U_x^{-2} + \hat{D}_3 U_x^{-1} + \hat{D}_4
\end{aligned} \tag{5.14}$$

where $\hat{}$ indicates the matrix of coefficients. This results in a nearly exact, continuous (in U_x), discrete-time parameter-varying controller in a state space form. Computing a control command y_k and a new state x_{k+1} is simply done by first computing $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ and then carrying out the discrete-time state space equations: $x_{k+1} = \hat{A}x_k + \hat{B}u_k$ and $y_k = \hat{C}x_k + \hat{D}u_k$. This can all be implement in a very computationally efficient way by leveraging the NDarray object in the Python 2 package Numpy. The storage requirements are four 3-D arrays need to be stored of sizes $(4\times4\times4)$, $(4\times1\times4)$, $(1\times4\times4)$, and $(1\times1\times4)$ for $\hat{A}_i$, $\hat{B}_i$, $\hat{C}_i$, and $\hat{D}_i$, respectively. Compared to traditionally gain-scheduled controllers, in which dozens of state space

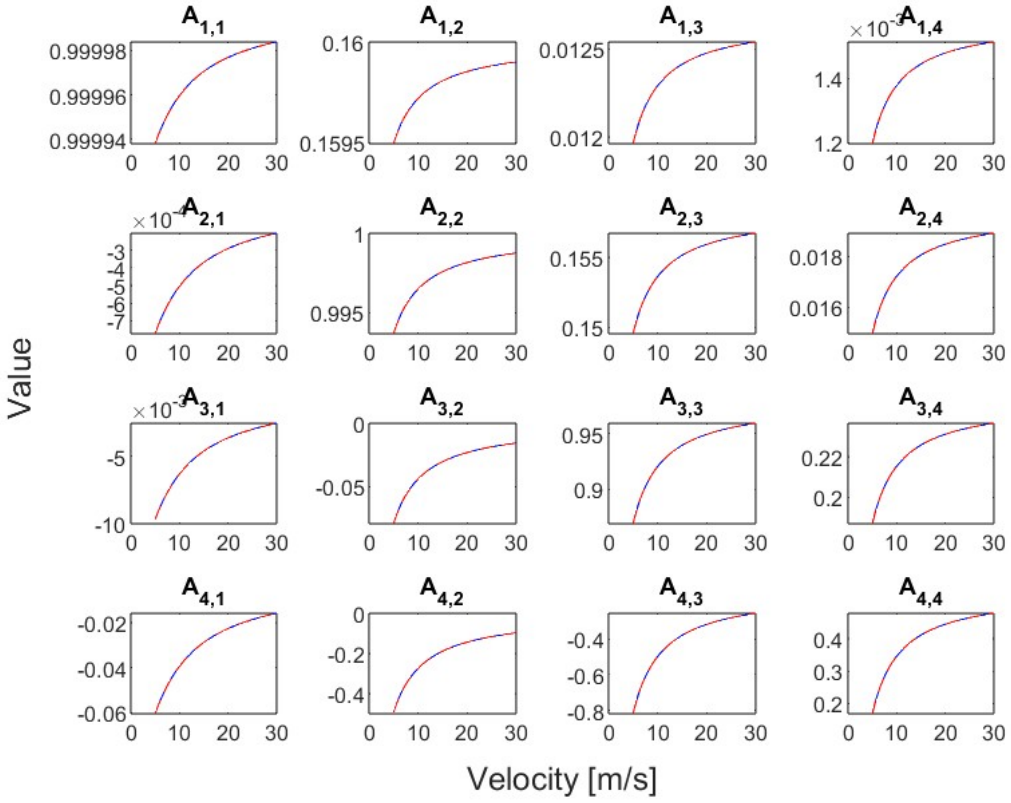


Figure 5.18: Regression fit for $A(U_x)$ elements. The blue line is the true value and the red line is the approximated value.

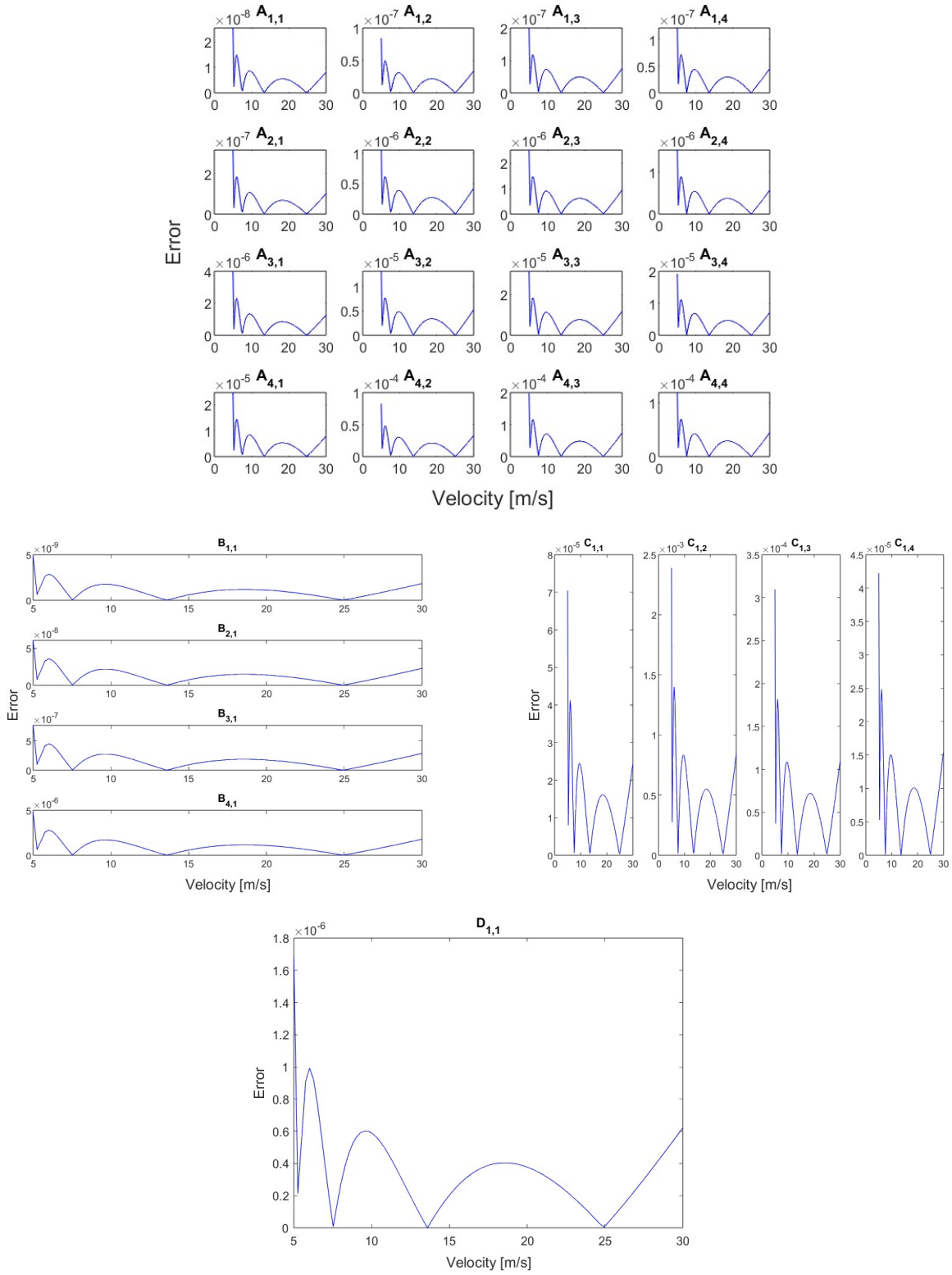


Figure 5.19: Regression fit errors for all state space matrix elements

models are stored and linearly interpolated, this approximation is very computationally and space-efficient. What would be more space efficient is to store the symbolic equations themselves as numerical functions. This may be possible with a combination of Matlab's Symbolic Math Toolbox and Matlab Coder (which can be used to generate C or C++ code).

5.5.3 Experimental Results

Three different on-vehicle results are presented to show the closed loop control performance of the previously described lateral motion controller. These maneuvers were performed on a retired runway near College Station, Texas. The first maneuver is a series of sine waves tracked at 5 m/s. The results in Figure 5.20. This maneuver uses most of the space on the available test track to test the tracking performance at low velocities while undergoing lateral accelerations and yaw rates that are reasonable for everyday driving. The lateral error and yaw error stays below 20 cm and 3 degrees, respectively.

The second maneuver shown in Figure 5.21 is very similar to the first, but is conducted at 25 m/s. Only one wave was possible due to limited testing space. The vehicle starts from rest and reachs 25 m/s at approximately 15 seconds. The lateral error and yaw error stay below 40 cm and 0.6 degrees, respectively. The change in maximum lateral and yaw errors between 5 and 25 m/s suggests that the lookahead distance should be made a nonlinear function of U_x such that the lateral error is weighted more at higher velocities and the yaw error is weighted more at lower velocities. Despite this, these two results show sufficient tracking performance that is comfortable for the vehicle's passengers (as determined by a qualitative assessment from the engineers inside the vehicle when conducting the tests).

The final maneuver tested, shown in Figure 5.22, is a very aggressive overtaking designed to represent a sudden collision avoidance maneuver. It is very similar to the ISO double lane change test. The lateral acceleration reaches -8 m/s^2 and the yaw rate almost reaches -30 deg/s. During the peak lateral accelerations and yaw rates the Jeep's Electronic Stability Control activated. This indicates that this maneuver is very aggressive and pushes the vehicle to its limits of handling. The lateral error remains below 20 cm until after the most aggressive part of the maneuver when the lateral error grows to 30 cm and then begins to decrease again.

These three maneuvers show the performance of the feedforward and feedback control design. The LPV feedback controller helps maintain stability and tracking errors less than 40 cm throughout the vehicle's velocity range and even for very aggressive maneuvers.

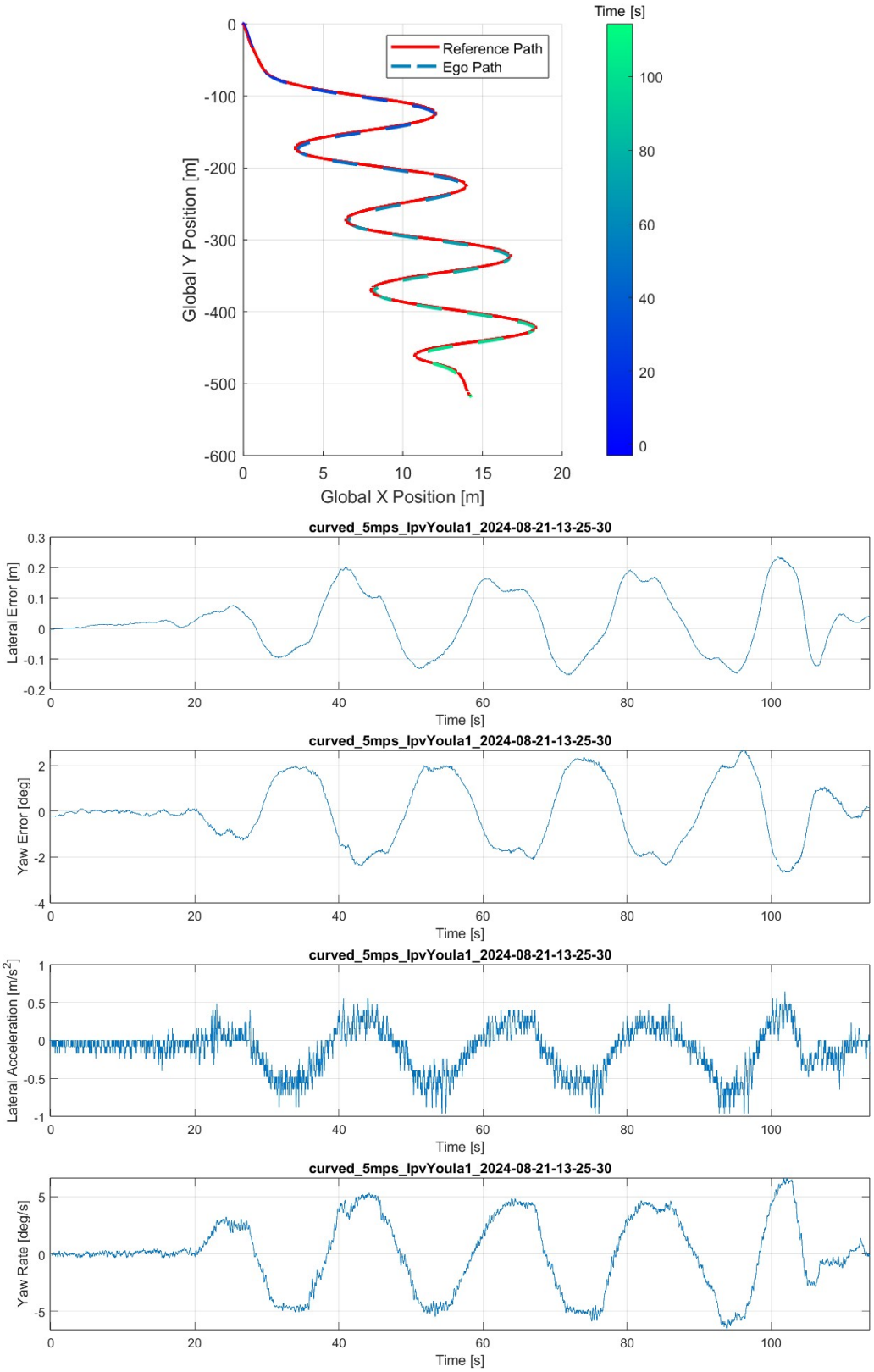


Figure 5.20: LPV lateral motion control performance for curved maneuver at 5 m/s.

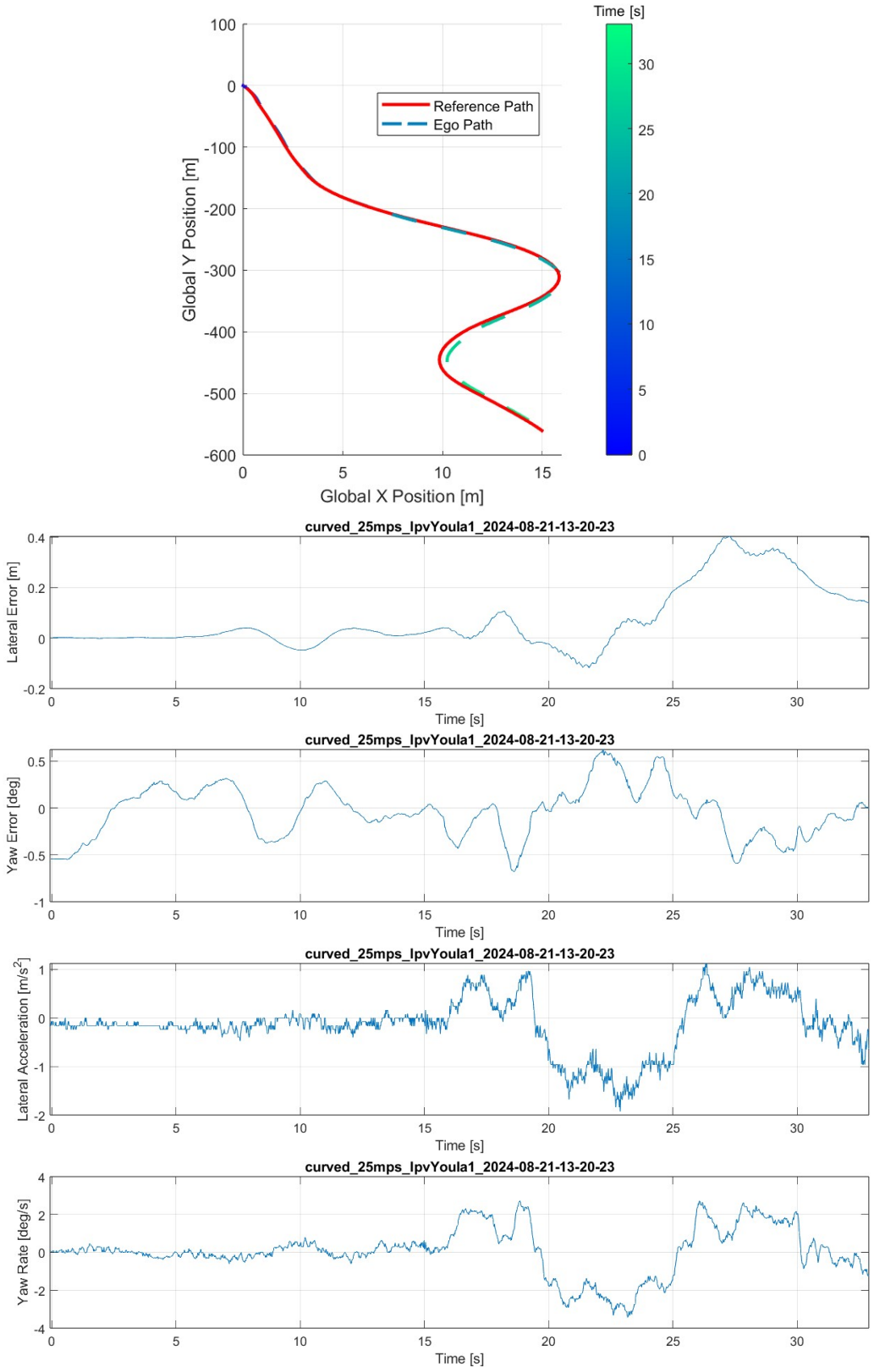


Figure 5.21: LPV lateral motion control performance for curved maneuver at 25 m/s.

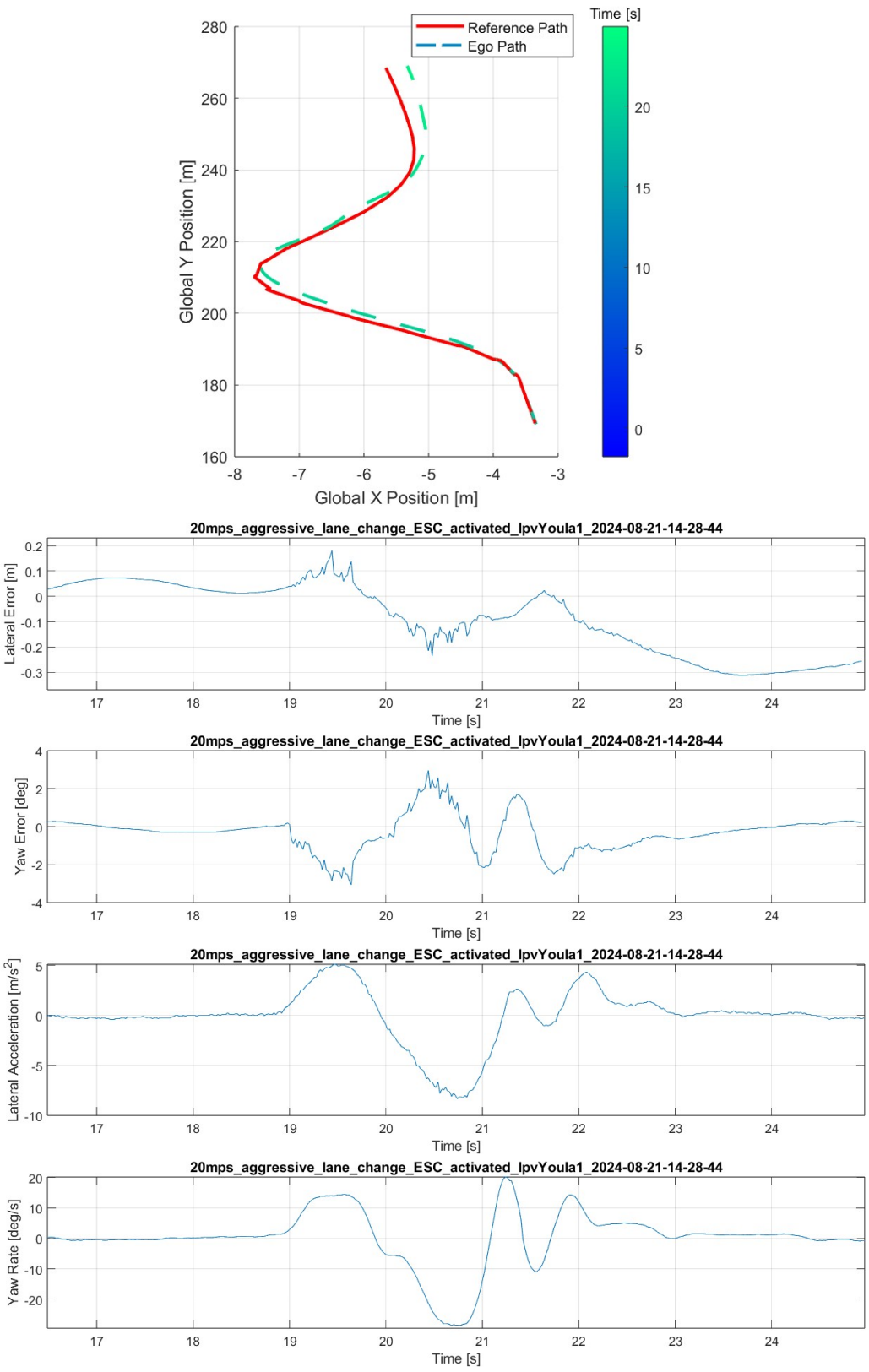


Figure 5.22: LPV lateral motion control performance for aggressive lane change at 20 m/s.

Further tests conducted on a gravel road are used to test the robustness of the gain-scheduled controller. Two controllers are shown in these sets of results. The first controller is the same LPV controller shown in Figures 5.20, 5.21, and 5.22. The results shown in Figure 5.23 are collected when the vehicle accelerates to 5 m/s and tracks a curved path. Figure 5.24 is collected when that same path is tracked at 15 m/s. The LPV controller tracks the path well at low and high speeds. 15 m/s is considered high for this gravel road since it is beyond the speed limit and the engineers conducting the tests noted that the vehicle started to slide and lose traction at times. The second controller is an LTI version of the LPV controller that is designed when the plant's $U_x = 25$ m/s. The results shown in Figure 5.25 shows that the controller is unable to drive the lateral error to zero well enough to complete the maneuver. The results shown in Figure 5.26 are collected on the same maneuver but at 10 m/s. Again, the LTI controller is unable to keep the lateral error within a safe margin and the test engineers abort the test. OVerall, this LTI controller performs very poorly and is unable to safely track the full route.

The results in Figures 5.23, 5.24, 5.25, and 5.26 present empirical evidence for the benefits of gain scheduling on the vehicle's longitudinal velocity to improve robust performance across a wide range of velocities, maneuvers, and road surfaces. This relates back to the divide-and-conquer strategy to address uncertainties in Section 5.4.1. The results shown in this section corroborate the analysis results shown in Section 5.4.1. In fact, the parameter variation is very similar: 5-30 m/s and friction coefficients between 1 and 0.5. The friction coefficient of gravel is unknown and is likely highly uncertain. The gravel road in these results is largely composed of a hard-packed dirt with a layer of loose aggregate on top. As a result, it is not uncommon to experience some sliding at unpredictable times. Despite this, the LPV lateral controller performs well even when sliding is experienced. This change in road friction dramatically changes the vehicle's dynamics.

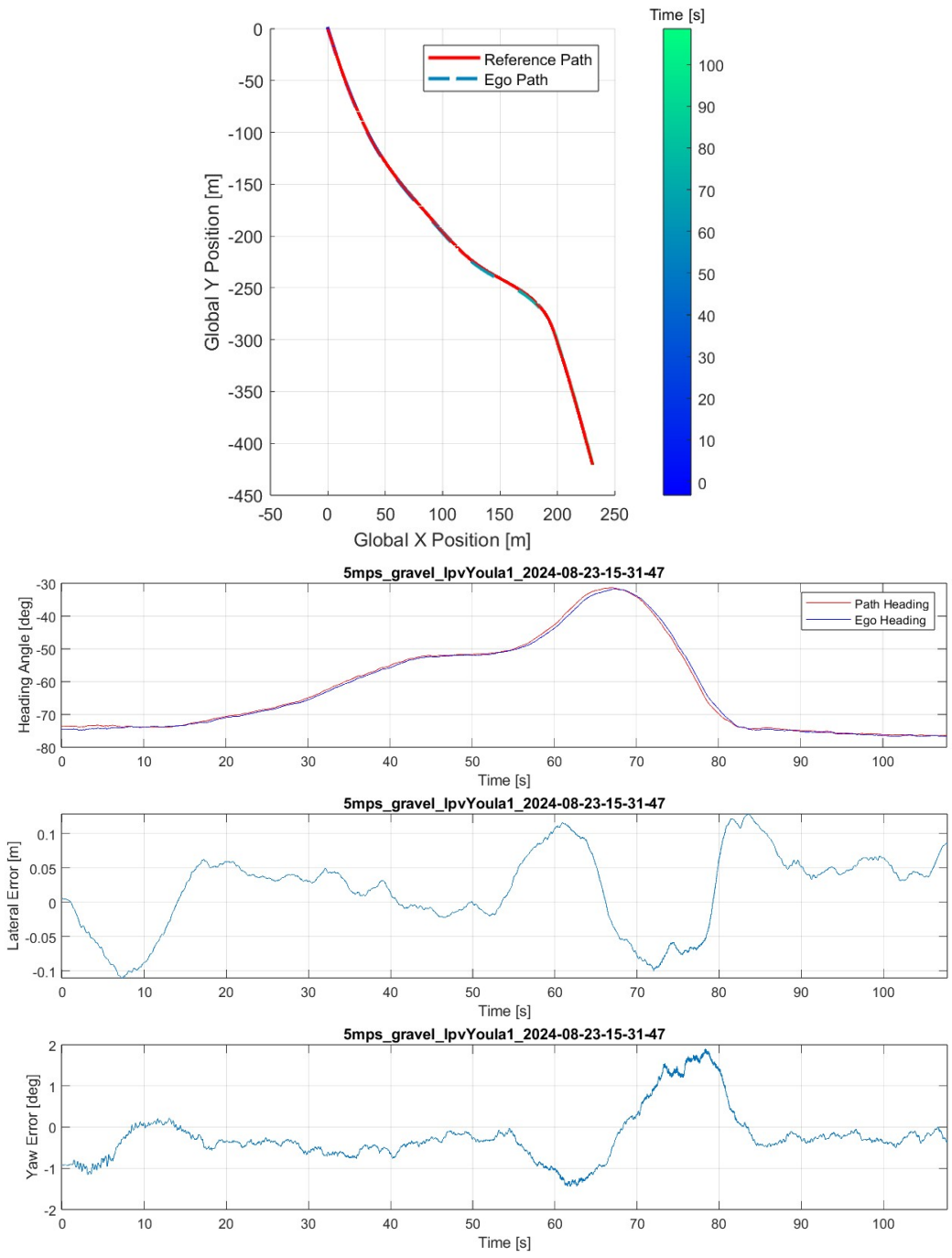


Figure 5.23: LPV lateral motion control performance for curved maneuver on a gravel road when driving at 5 m/s

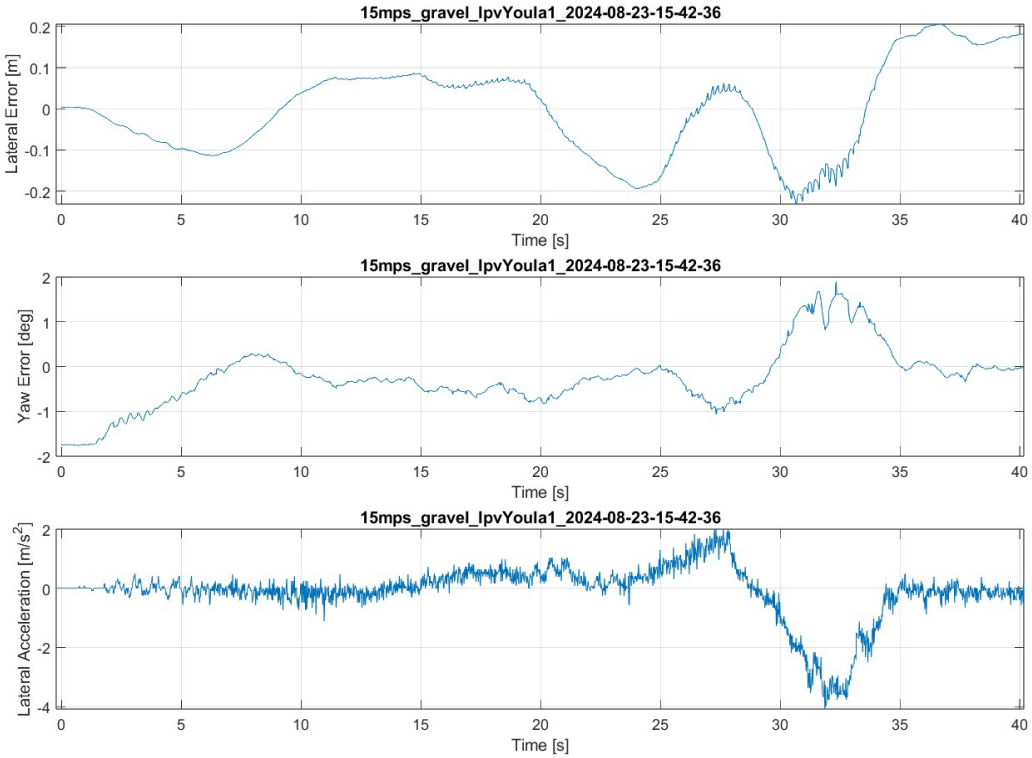


Figure 5.24: LPV lateral motion control performance for curved maneuver on a gravel road when driving at 15 m/s

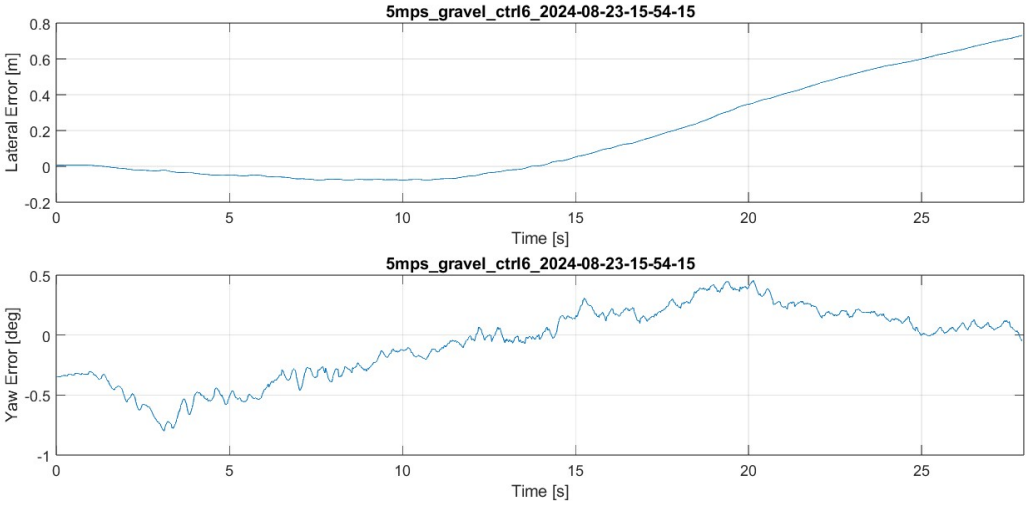


Figure 5.25: LTI lateral motion control performance for curved maneuver on a gravel road when driving at 5 m/s

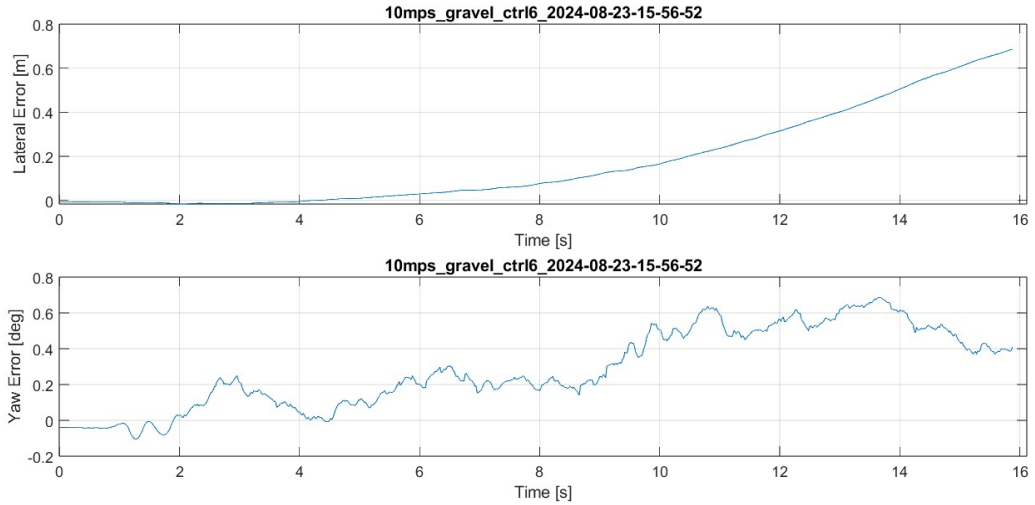


Figure 5.26: LTI lateral motion control performance for curved maneuver on a gravel road when driving at 10 m/s

5.6 Conclusion

This chapter has shown that the Interpolation Conditions originally developed for LTI systems can also be used for LPV systems. The presented example in Section 5.4 is a SISO path-tracking problem. Future work will explore the application of this technique to MIMO control problems. Furthermore, without a mathematical proof that Interpolation Conditions guarantee LPV stability (such as quadratic stability), more investigation is needed to study this method's limitations.

Additional research with this method might concern anti-windup, which is a category of algorithms that handles the windup issue in linear controllers. The windup issue occurs when the error cannot be reduced because of some nonlinear plant phenomena. Typically this is the actuator saturating at some value. However, this can occur in other ways. For instance, a lateral motion controller deployed on a conventional vehicle at rest will not be able to move laterally until it begins to move forward. This is the nonholonomic constraint that is commonly observed in vehicle control. Because the error cannot be reduced, if there is some integral action in the controller, the controller states may grow unbounded. The resulting actuator command can also grow unbounded. Despite this increasing actuator command, the lateral error will not change. This is not a saturation in the steering actuator, but rather a phenomenon presented by the nonholonomic nature of an automobile. This issue is typically observed at the beginning of an experiment when the test begins with the vehicle at rest, but with a non-zero lateral and/or yaw error. However, in

public-road deployments, there are several reasons why a vehicle will need to come to a stop (ie. stop sign, red light, yielding, etc.). Therefore, dealing with the issue of windup when the vehicle is at rest is of significant practical importance.

These saturations, in addition to the uncertainties introduced by the velocity variation of the vehicle, must be appropriately handled to ensure robust performance and stability. Therefore, gain scheduling presents a promising technique to divide and conquer uncertainties. By gain-scheduling on longitudinal velocity, the robust control requirements become more focused on cornering stiffness uncertainties. The design presented in this chapter is shown in Section 5.4 to be sufficiently robust against low friction and external disturbances. However, it may not be robust against complete rear or front tire saturation. This is where the extension developed in [198] might be a useful addition to the YK parameterization and Interpolation Conditions.

This divide and conquer approach to uncertainties is the motivation for increasing the controller's complexity. This is best illustrated in Section 5.5 where the design and implementation are explained in detail. However, this increase in complexity is well-supported by the control performances presented in Section 5.5.3, where the vehicle is tested within the bicycle model's validity and throughout a reasonable velocity range. Furthermore, the combined feedforward and feedback control is shown to be very robust against significant model discrepancies when the vehicle is pushed far beyond the model's valid region and into the vehicle's handling limits. These tests show the promise of this LPV technique based on Interpolation Conditions. It also shows the benefits of addressing a wide range of uncertainties by gain scheduling on one set of uncertainties (vehicle speed in this case).

Chapter 6

Comparison of Cheap Front Steering Controllers

The study of lateral steering control for Automated Driving Systems identifies new control solutions more often than new control problems. This is likely due to the maturity of the field. To prevent repeating efforts toward solving already solved problems, what is needed is a cohesive way of evaluating all developed controllers under a wide variety of environmental conditions. This chapter serves as a step in this direction. Four controllers are tested on five maneuvers representing highways and collision avoidance trajectories. Each controller and maneuver combination is repeated on five sets of environmental conditions or Operational Design Domains (ODDs). The design of these ODDs ensures the translation of these experimental results to real-world applications. A high-fidelity simulator that models the environment, vehicle dynamics, sensor dynamics, and state estimation performance is used to achieve highly repeatable and realistic evaluations of each controller. The results of this work demonstrate that many of the combinations of maneuvers and ODDs have existing cheap controllers that achieve satisfactorily safe performance. Therefore, this field's research efforts should be directed at the ODDs that do not have such controllers.

6.1 Introduction

The design of an automotive front steering controller that tracks some path or trajectory has been a heavily researched topic since the 1950s [73]. Before the late 1990s, most controllers focused on speeds lower than typical highway driving [83]. One notable exception is [64] which developed a steering controller for a 1965 Plymouth Sedan. Various controllers were tested in wet and dry road conditions between 50 and 80 mph. However, their success was mostly limited to low-curvature roads. Also around that time, the US Department of Transportation's Intelligent Transportation Systems program began researching Automated

Highway Systems. However, such efforts were not limited to just the United States. An overview of the advances during this time is found in [107].

Due to the maturity of this research area, it is not surprising that consumers have access to automated steering systems. In 2021, 50% of new vehicle models offer sustained Lane Keeping Assistance, which commonly uses some form of lateral steering control [187]. Insight into the control techniques used by these systems is not publicly available, except for Comma Ai's open-source OpenPilot [49]. This system uses a Deep Neural Network to extract a future waypoint directly from a forward-facing camera. This waypoint is used in separate Longitudinal and Lateral Model Predictive Control-based local planners to generate a reference path. A steering controller then tracks the reference path using a PID feedback control law with a feed-forward controller.

With knowledge of only one commercially deployed controller, very little can be said about these Lane Keeping Assistance systems. In particular it is difficult to investigate their safety. However, their existence proves that many companies feel they can achieve safe automated steering at highway speeds. The natural question to ask is what are their limitations? This question is close to the heart of this work: determining under which conditions lateral front steering controllers fail to meet pre-specified performance requirements.

To understand the limitations of the commercially available steering controllers (such as Lane Keep Assist), we can use the terms and definitions from [50, 51] to discuss their limitations. We begin by pointing out that the systems analyzed in [187] are examples of Active Safety Systems and not Automated Driving Systems (ADS). The Lane Keep Assist investigated in [187] requires an attentive driver capable of interrupting the controller quickly after any issue occurs. This requirement significantly increases the level of acceptable risk in the design of a lateral steering controller. This suggests that the lateral steering controller can fail in some Operational Design Domains (ODDs). If true, this suggests that the problem of designing a sufficiently safe lateral controller for operation in some ODDs is still unsolved. However, it is not possible to say under which ODDs, if any, these controllers will fail. This conclusion is further qualified by the observation that there is not enough information to show that the front steering controller is the cause of the limited ODD. For instance, the perception system (ie. a forward-looking camera), and not the steering controller, may cause the limited ODD. Therefore, there must be efforts dedicated to evaluating these Active Safety Systems in a wide variety of ODDs to understand their limitations and the cause of those limitations. After that, research and development efforts can be more efficiently allocated.

Active Safety Systems are not the only types of systems that would benefit from such an effort. When

the steering controller is designed for an ADS, the system cannot be designed with the assumption that a human can take over. To simplify this problem, constraints are often imposed on the ODD. This typically takes the form of geofencing regions where the ADS is allowed to operate. Robo-taxi companies Cruise and Waymo use this strategy. Both companies have deployed ADS in cities such as Las Vegas, Nevada; Phoenix, Arizona; Houston, Texas; and San Francisco, California. The lack of cities that experience adverse weather throughout the year suggests that modern ADS cannot yet be safely deployed in adverse weather or adverse road conditions. While this is due, in part, to the degraded sensor performances in adverse weather conditions [201], another reason for this is the difficulties encountered in controlling the vehicle under adverse weather conditions.

Despite decades of research on lateral steering controllers, the question of how well steering controllers perform in various ODDs remains insufficiently answered. A major cost of not sufficiently answering this question is that only a subset of the US population will be able to experience the safety and economic benefits of ADS. The current deployment of robo-taxis focuses on the most profitable, and easiest ODDs. These ODDs consist of densely populated areas with good weather throughout the year. Examples are found in the deployment locations of Cruise and Waymo in 2023. Populations that live outside these domains will not benefit from the technology until safe operation can be sustained for enough of the year to provide a profitable business. Therefore, it is of paramount importance to focus attention on studying the performance of steering controllers in a wide variety of ODDs, not just the ones that are most easily driven.

At a high level, this chapter seeks to answer the question of what are the limitations of modern lateral steering controllers. This question is motivated by the desire to have a single lateral steering control architecture that is capable of achieving useful and pre-specified performance requirements across a variety of ODDs. The framing of this question requires the answer to be dependent on the ODD, which will help the steering control research community to focus more directly on ODDs that are difficult to achieve sufficient performance.

Through a proper choice of steering controllers and ODDs, experimental analysis will indicate combinations of maneuvers and ODDs where steering controllers fail to meet requirements (and should be improved upon) and where they already meet requirements. The steering control literature is vast, so a small sample of steering control algorithms is selected as will be explained in Section 6.2.1. The controllers are chosen in an attempt to provide information on what will be the most successful future direction of research. To compare these controllers, several maneuvers and environmental conditions are developed in Chapter 3. Together,

they constitute samples of existing ODDs that need to be solved to achieve SAE Level 5 driving [51]. To compare each controller on the same ODDs in a highly repeatable manner, the extensive simulator developed in Chapter 3 is used. This simulation realistically captures the most significant phenomena in highway driving. It combines models of the environment and its associated disturbances with a high-fidelity vehicle model that captures tire-road interactions and powertrain dynamics. To compare each controller objectively, several metrics are proposed in Section 6.2.3. Finally, the results are presented and discussed in Section 6.3.

The main contribution of this work is the systematic investigation of controllers in realistic ODDs. Another contribution of this chapter is showing that cheap linear and nonlinear controllers can achieve sufficient performance in some specific ODDs. In this work, a controller that does not rely on a high number of online computation steps is referred to as a cheap controller. In this sense, model predictive control is not considered a cheap controller, while a linear controller, implemented as a digital filter, is considered cheap. A secondary contribution is showing the extent to which collision avoidance maneuvers can be used to predict performance on highway trajectories. For instance, the performance of a steering controller in a modified version of ISO 3888's double-lane change (commonly known as the Moose Test) will be compared to the performance of that same controller on an existing highway. Altogether, these contributions are preliminary work toward a standardized method of comparing steering controllers and the development of a robust steering controller that meets performance requirements in all ODDs.

6.2 Methods

6.2.1 Controller Selection

The previous section briefly discussed the history and size of the literature on steering controllers. It is common in survey papers to group controllers based on the model that they use and the type of control methodology such as kinematic, dynamic, nonlinear, linear, and online-optimal [46, 164, 27, 37]. Most geometric controllers are insufficient for high-speed operation, where inertial effects are significant [46, 164, 37].

A similar conclusion is drawn for kinematic controllers, but not without controversy [130, 60]. Controllers that use kinematic models have found success when used in Model Predictive Control techniques, often with additional algebraic terms to compensate for tire dynamics or inertial effects [130, 60, 143, 182]. The drawback of using Model Predictive Control, or any online optimization-based method, is the high

computation cost. Most applications mitigate this with simplified models, hence the kinematic model's popularity. However, even when using simplified models, their computational effort is often greater than controllers implemented using filters. This computational cost may not be necessary to achieve robust performance on highways. To show this, we constrain the controllers considered in this investigation to "cheap" controllers. "Cheap" refers to controllers that do not use online optimization or large amounts of online iterations. This constraint poses a central hypothesis of this investigation: satisfactory control performance is achievable without online optimal techniques for some practical ODDs.

To achieve highway driving, the steering controller must ensure stability and safe performance at high speeds. Therefore, controllers that have been demonstrated at high velocities are preferred over controllers that have been tested at low velocities. It is well known that the vehicle's dynamics vary significantly with longitudinal velocity [121, 64, 83]. This high variation motivates focusing on controllers that are parameter varying or have easy extensions to become parameter varying. However, this is not without controversy as several papers have proposed robust controllers instead of parameter varying controllers [3, 82, 2]. To investigate if an LTI controller is capable of achieving similar performance as LPV controllers, an LTI controller will also be considered in this comparison.

Most nonlinear controllers are designed to operate at various speeds even if their tuning parameters need to be scheduled with velocity [121, 180, 164]. Some linear controllers, such as PID and LQR, are easily extended to parameter-varying control at the cost of additional computing and complexity.

The following three controllers are selected from the controllers surveyed in [46, 164, 27, 37]. Each is chosen because (1) they do not require online optimization, (2) they have been demonstrated on full-scale vehicles at high speeds, and (3) they are either already scheduled with velocity or are easily extended to schedule with velocity.

1. Nonlinear Feedback and Feed-forward control (FDBK+FFW) [121]
2. Target and Control Driver Model (T&C) [180]
3. Discrete-Time Infinite-Horizon Linear Quadratic Regulator (LQR) [1] [160]

A fourth controller will also be included in this comparison, which is a parameter-varying extension of the Youla Parameterization technique with Interpolation Conditions [61]. This Interpolation technique

[1] Formally, this acronym should be DTLQR or dLQR, but since there is no other LQR formulation to differentiate from, LQR is used instead.

(without the LPV extension) has shown success in varying automotive applications: [12, 67, 79].

At this point it is worth addressing the fallacies in characterizing the lateral motion control literature's progress with just a handful of controllers. Practically, it is impossible to evaluate all possible lateral motion controllers. Variations of lateral motion control might be characterized by different plant models, different inputs and outputs, different control theories, and even different system architecture. Therefore, the conclusions from this study are greatly limited. For instance, we have selected an LTI controller to evaluate if it is possible to use an LTI controller to achieve safe tracking performance. A conclusion such as "no LTI controller is capable of achieving safe tracking performance" can be made because only one approach is studied.

For this reason, this Chapter is primarily focused on investigating if it is possible to use a "Cheap" controller to achieve safe lateral motion control. The results in the following subsections demonstrate the degree to which this is possible. This Chapter's results should be considered with other similar control comparison research work such as [46, 164, 27, 37, 18]. The number of controllers investigated in all these studies indicates that (1) there is a wide variety of control techniques that can be applied to lateral motion control and (2) there is a desire to find the current best-performing controller in the literature as well as discover promising approaches for new research. The machine vision research community has a similar goal, which is addressed by developing open-source benchmarking datasets [184]. Such benchmarks crowdsource the task of finding the best machine vision algorithm (and tuning method). A similar approach should be taken for the task of lateral motion control. However, there is no open-source, high-fidelity simulator currently available to establish a lateral motion control benchmark. The simulator developed in Chapter 3 can be considered as a small step towards this goal.

Nonlinear Feedback and Feed-forward Control

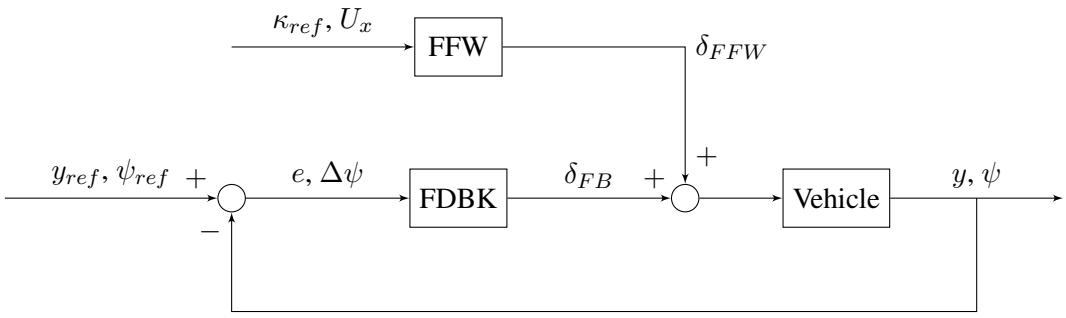


Figure 6.1: FDBK+FFW Control Diagram

The first controller is an example of a nonlinear controller shown in Figure 6.1. It uses a nonlinear feed-forward controller with proportional feedback. This controller is experimentally validated on an Audi TTS in racing conditions [121]. Several other papers are closely related to this work that extend the methodology to drifting maneuvers and a wide variety of road surfaces [122, 76, 176]. The feed-forward utilizes a Fiala tire model to predict the required steering angle, δ_{FFW}, and steady-state body-slip angle, β_{ss}, from the reference curvature, κ_{ref}. The feedback law is given by Equation 6.1, where k_p is the feedback gain, e is the lateral error at c.g., x_{LA} is the look-ahead distance, and $\Delta\Psi$ is the yaw error at c.g.

$$\delta_{FB} = -k_p(e + x_{LA}(\Delta\Psi + \beta_{ss})) \tag{6.1}$$

There are two tuning parameters: the look-ahead distance and the controller gain. The closed-loop pole locations are given by the eigenvalues of Equation 6.2.

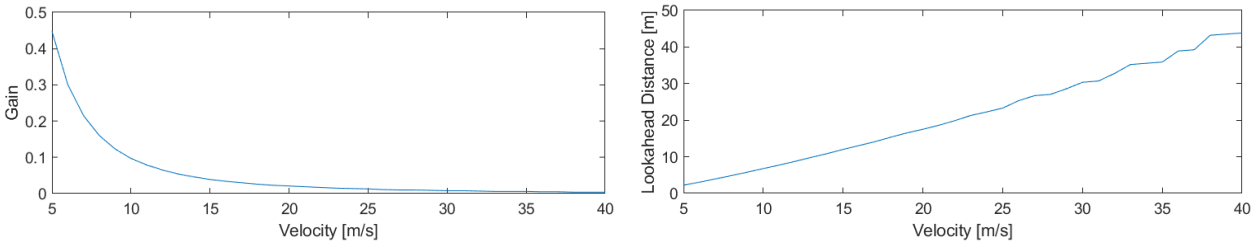
$$A = \begin{bmatrix} 0 & U_x & 0 & U_x \\ 0 & 0 & 1 & 0 \\ \frac{-ak_pC_f}{I_z} & \frac{-ak_px_{LA}C_f}{I_z} & \frac{-(a^2C_f+b^2C_r)}{U_xI_z} & \frac{bC_r-aC_f}{I_z} \\ \frac{-k_pC_f}{mU_x} & \frac{-k_px_{LA}C_f}{mU_x} & \frac{bC_r-aC_f}{mU_x^2} - 1 & \frac{-(C_f+C_r)}{mU_x} \end{bmatrix} \tag{6.2}$$

The original form of this controller (using constant gains) was found to be insufficient across the wide range of velocities at which the vehicle can operate. Therefore, the gains are scheduled with longitudinal velocity to achieve improved performance across all velocities. It was observed that the dominant poles of the closed-loop system move with respect to both control parameters. With this knowledge, an optimization routine, Algorithm 1, is created that achieves desirable natural frequencies, ω_{thresh}, and damping ratios, ζ_{thresh}, of the most dominant poles.

Algorithm 1 FDBK+FFW Gain Optimization Routine

for each U_x **do**
  $\zeta \leftarrow 0$
  $x_{LA} \leftarrow 0$
  while $min(\zeta) \leq \zeta_{thresh}$ **do**
    $x_{LA} \leftarrow x_{LA} + \delta_{x_{LA}}$
    $k_p \leftarrow 0$
    while $min(\omega_n) \leq \omega_{thresh}$ **do**
      $k_p \leftarrow k_p + \delta_{k_p}$
      Compute $min(\zeta)$
      Compute $min(\omega_n)$
    end while
  end while
end for

To keep the gains low, the routine's hyperparameters: ω_{thresh} and ζ_{thresh} are scheduled with longitudinal velocity, U_x. To tune the hyperparameters, simulations are run on the single and double-lane change trajectories under nominal and realistic conditions. These conditions are explained in Chapter 3. The resulting gains are shown in Figure 6.2.



(a) FDBK+FFW k_p

(b) FDBK+FFW x_{LA}

Figure 6.2: FDBK+FFW Gains with respect to Velocity

Target and Control Driver Model

The second controller is an example of driver model-based controllers [181]. This particular controller is one of the most mature and experimentally validated controllers in the literature. It has been tested extensively on an articulated bus, and extended with various fault tolerant systems [180, 103]. It also has been experimentally validated on a sedan at moderate speeds [135]. However, the original formulation is modified by replacing the integrator in the feedback controller from [180, 103] with a Youla-Kucera parameterized controller [135]. This enabled the controller to be adapted online to the type of maneuver the vehicle was performing (either lane change or lane keeping). Such extensions show promise in improving

the base performance of this controller. If this controller shows good results, there would be good motivation to focus research on controllers of this form.

The original controller can be implemented in several ways. The version implemented in this study is given in Equation 6.3 [181].

$$\begin{aligned} \delta_t &= k_p \int_0^t (\psi_{ref} - \psi_{veh}) dt \\ \psi_{ref} &= \text{atan}(\frac{Y_{ref} - Y_{veh}}{X_{ref} - X_{veh}}) + \text{asin}(\frac{x_{LA}}{2R}) \\ R &= \frac{U_x}{\dot{\psi}_{veh}} \\ x_{LA} &= k_{LA} U_x \end{aligned} \tag{6.3}$$

where the variables are defined in Table 6.1.

Variable	Name	Unit
δ_t	Steering angle at time t	radian
ψ_{veh}	Vehicle yaw angle	radian
ψ_{ref}	Reference yaw angle	radian
Y_{ref}	Global Y coordinate of reference point	meter
Y_{veh}	Global Y coordinate of the vehicle's c.g.	meter
X_{ref}	Global X coordinate of the reference point	meter
X_{veh}	Global X coordinate of the vehicle's c.g.	meter
U_x	Vehicle's longitudinal velocity	m/s
$\dot{\psi}_{veh}$	Vehicle's yaw rate	rad/s
x_{LA}	Lookahead distance	m
R	Vehicle's instantaneous radius of curvature	m
k_{LA}	Look-ahead gain	
k_p	Controller gain	

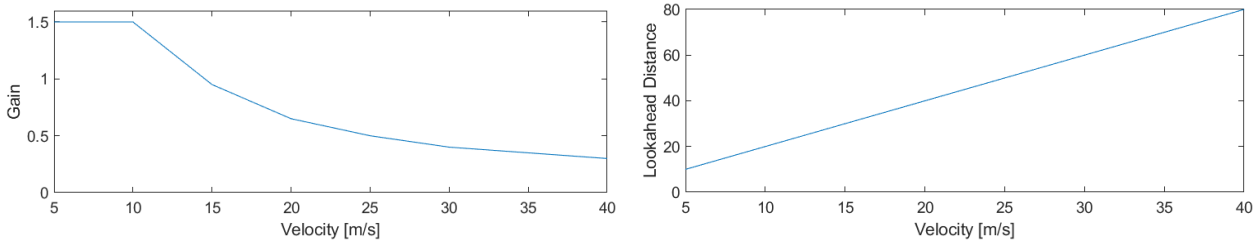
Table 6.1: T & C Control Law Variables

The T & C controller has only two gains to tune: the look-ahead gain and the controller gain, which must be scheduled with longitudinal velocity to ensure stability across the vehicle's velocity range. Similar to the FDBK+FFW controller, these gains are tuned using an optimization technique. However, this optimization computes the closed-loop poles and corresponding disk margin [168](a more robust version of traditional gain and phase margins since they consider the full frequency range of the system) over a grid of gains. Then using a cost function, the optimal gains are selected. This approach is covered in Algorithm 2. The resulting gains are shown in Figure 6.3.

Algorithm 2 T & C Gain Optimization Routine

for each U_x **do**
  Compute $\zeta \forall k_p \in (0, 2)$ and $\forall k_{LA} \in (0, 4)$
  Remove all k_p and k_{LA} where $\zeta \leq \zeta_{thresh}$
  $c_{k_p} \leftarrow \frac{k_p - \min(k_p)}{\max(k_p) - \min(k_p)}$
  $c_{k_{LA}} \leftarrow \frac{k_{LA} - \min(k_{LA})}{\max(k_{LA}) - \min(k_{LA})}$
  $c_{DM} \leftarrow$ compute disk margin
  $(k_p, k_{LA}) \leftarrow \text{argmin}(W_{k_p} c_{k_p} + W_{k_{LA}} c_{k_{LA}} + W_{DM} c_{DM})$
end for

The hyperparameters, W_{k_p}, $W_{k_{LA}}$, W_{DM} are weights on the controller gain, look-ahead gain, and disk margin, respectively. These hyperparameters are tuned in the same way as the Algorithm 1's hyperparameters.



(a) T & C k_p (b) T & C x_{LA}

Figure 6.3: T & C Gains with respect to Velocity

The trajectory of the optimal controller gain with respect to velocity closely resembles the one computed in [134], indicating that this routine is similar to the one developed by the authors of the T & C Driver Model.

Linear Quadratic Regulator

The third controller is the well-known Linear Quadratic Regulator (LQR), which is used to represent LTI controllers. The controller is developed using the error state model from [160], which is reproduced for completeness in equation 6.5.

$$\dot{x} = Ax + B_1\delta + B_2\dot{\psi}_{des} \tag{6.4}$$

$$\frac{d}{dt}\begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_f+C_r}{mU_x} & \frac{C_f+C_r}{m} & \frac{-aC_f+bC_r}{mU_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{aC_f-bC_r}{I_zU_x} & \frac{aC_f-bC_r}{I_z} & -\frac{a^2C_f+b^2C_r}{I_zU_x} \end{bmatrix}\begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{aC_f}{I_z} \end{bmatrix}\delta + \begin{bmatrix} 0 \\ -\frac{aC_f-bC_r}{mU_x} - U_x \\ 0 \\ -\frac{a^2C_f+b^2C_r}{I_zU_x} \end{bmatrix}\dot{\psi}_{des} \tag{6.5}$$

$$\dot{\psi_{ref}} = \frac{U_x}{R_{ref}}$$

Where the variables are defined the same as in previous sections. This model considers two inputs to the bicycle car: front steer angle and desired yaw rate. In [160] the desired yaw rate is considered a measurable disturbance that is rejected with a feed-forward controller.

The design is well explained in [160] except that this study uses the discrete-time version since the controller is sampled during simulation. The LQR cost function provides five tuning parameters: a weighting on each state and the steering angle. The weight on the actuator effort, R, is initially set at $R = B_1^T B_1 \approx 8,500$. Due to this large value, the system's performance is not very sensitive to the weighting matrix on the states, Q, so it is simply set as the identity matrix. Then, the R gain is tuned iteratively through simulation on the single and double lane change maneuvers. As R decreases, the steering controller becomes more aggressive, reducing the error relative to the path. However, if decreased too much, the controller can begin oscillating and potentially saturate and destabilize the closed-loop simulation. Therefore, R is tuned until the steering controller does not oscillate and destabilize while still achieving good performance. The result is $R = 500$.

A final tuning option is the choice of constant velocity when computing the state feedback gains. Through further simulations, it was found that better performance is achieved when the set velocity is equal to or greater than the testing velocity. Therefore, the longitudinal velocity is set as 30 m/s. While this set velocity is higher than any velocity used in this study, it is chosen because it is the reasonable upper bound of the vehicle's operating range and provides good performance at all velocities.

This choice of tuning is further motivated by an analysis of the symmetric disk margin when the system operates at velocities other than 30 m/s. This analysis, shown in Figure 6.4, shows that the symmetric

disk margin increases as the system's velocity decreases. Therefore, the closed-loop system becomes more robustly stable at lower velocities, motivating the decision to tune the controller at the highest feasible velocity rather than some other lower velocity. However, the tradeoff is improved robustness with decreased performance at lower velocities.

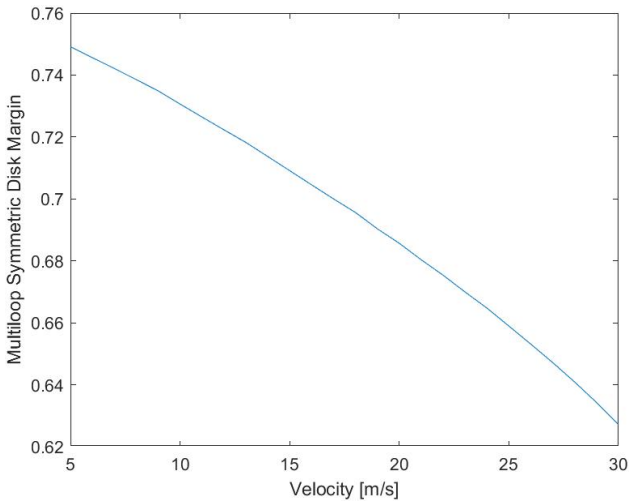


Figure 6.4: Multiloop Symmetric Disk Margin for LQR Controller Across Velocity Range

Linear Parameter Varying Youla Controller

One significant control technique not represented by these three is the frequency-based approach. To address this, we use a control methodology that is a Linear Parameter-Varying (LPV) extension of the Youla Parameterization Interpolation methods from [61]. This controller uses the error state model from [160], where the longitudinal velocity is the scheduling parameter, and a look-ahead distance is used to compute the model's output. This look-ahead distance has been the subject of research for decades and is well known to be very beneficial in steering control at high speeds because it compensates for the system's phase lag [64, 73, 83, 81]. This technique is similar to the LPV Coprime factorization used in [135]. Except where [135] uses Coprime factorization, this technique uses Interpolation Conditions to ensure internal stability.

To overcome this, the controller is computed from a desired closed-loop transfer function that takes the form of Equation 6.6. This is similar to the model-matching problem [42, 202, 144]. The difference is that in the model matching problem the reference model transfer function is unconstrained, whereas in this technique the reference model transfer function is constrained by the Youla Interpolation conditions.

$$
\begin{aligned}
T(s) &= \frac{k\ (s\,\tau_1 + 1)}{(s\,\tau_2 + 1)\,D_2(s)D_1(s)} \\
D_1(s) &= \omega_{\mathrm{n}2}^2 + 2\,\zeta_2\,\omega_{\mathrm{n}2}\,s + s^2 \\
D_2(s) &= \omega_{\mathrm{n}1}^2 + 2\,\zeta_1\,\omega_{\mathrm{n}1}\,s + s^2
\end{aligned}
\tag{6.6}
$$

where Equation 6.7 guarantees internal stability.

$$
\begin{aligned}
k &= \omega_{n1}^2\omega_{n2}^2 \\
\tau_1 &= \tau_2 + 2\frac{\zeta_1}{\omega_{n1}} + 2\frac{\zeta_2}{\omega_{n2}}
\end{aligned}
\tag{6.7}
$$

Therefore, the closed loop system is fully determined by ω_{n1}, ζ_1, ω_{n2}, ζ_2, and τ_2. The challenge is to set these parameters to achieve robust performance and limited actuator effort. Typically, the poles are placed in a Butterworth pattern, which results in desirable low-pass filter characteristics. However, these desirable characteristics require the absence of zeros in the system, and Internal Stability requires at least one zero. It is common practice to place the zero far away from the dominant poles, but the relationship between the zero and the poles (Equation 6.7) makes this impossible since all parameters must be strictly positive. The choice of these values is therefore not trivial.

To overcome this problem, the parameters are tuned to approximate an analogous H-infinity loop-shaping control problem. This analogous problem is solved for the LTI plant at velocity intervals of 5 m/s for 5-40 m/s. The process is explained in Algorithm 3. The nomenclature used is as follows: $Q(s, U_x)$ is the Youla Parameter transfer function from reference signal r to actuator input u, $P(s, U_x)$ is the plant transfer function, and $T(s, U_x)$ is the complementary sensitivity transfer function from r to output y. Each transfer function is also a function of U_x. T_{H_∞}, in particular, is a function of U_x because the weighting filters used in the loop-shaping problem were also parameterized by U_x to allow the closed-looped bandwidth to increase with increasing velocity (in the same way as was done in Chapter 5). This flexibility allows for an improved balance of the performance-robustness trade-off throughout the system's velocity range.

Algorithm 3 LPV Youla Tuning Routine

Require: $|Q(s, U_x)|_\infty = |\frac{u}{r}|_\infty < -10$ dB
  for each U_x **do**
    Compute $P(s, U_x)$
    Compute $T_{H_\infty}(s, U_x)$ as solution to H-infinity loop shaping problem
    $(\omega_{n1}, \zeta_1, \omega_{n2}, \zeta_2, \tau_2) \leftarrow T(s) \approx T_{H_\infty}(s, U_x)$
  end for

The last step in the algorithm is to approximate $T_{H_\infty}(s, U_x)$ with $T(s)$ by setting the poles of $T(s)$ close to the dominant poles of $T_{H_\infty}(s, U_x)$. This achieves a low system order while balancing sensitivity requirements and actuator constraints. In effect, it is a low order approximation of the H-infinity solution.

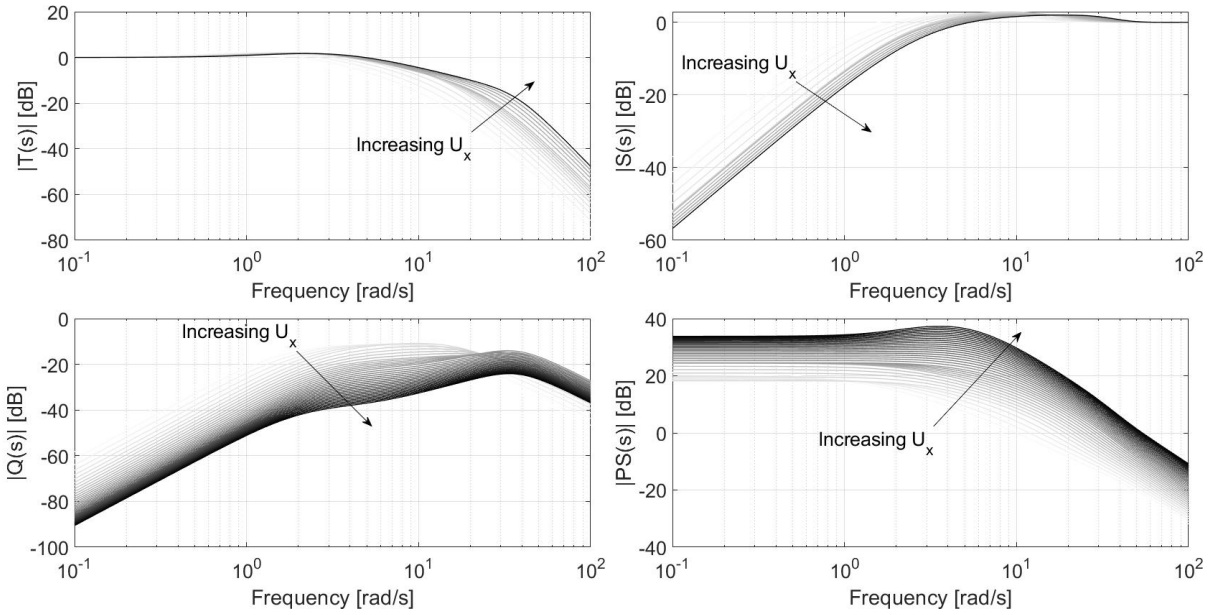


Figure 6.5: LPV Youla Gang of Four

The result of this tuning routine is that at velocities above 20 m/s, the parameters varied very little. To simplify tuning, $T(s)$ is held constant for velocities above 20 m/s. However, even though the reference model transfer function does not change when the velocity is above 20 m/s, the computed controller still varies because the plant varies. The resulting bode plots of the gang of four: $T(s)$, $S(s)$, $Q(s)$, and $PS(s)$ are shown in Figure 6.5. Here, $S(s)$ is the sensitivity transfer function, and $PS(s)$ is the product of $S(s)$ with $P(s)$. Figure 6.5 shows the frequency response of each transfer function as velocity increases in the range 5-40 m/s, represented as the transition from white to black colored lines.

Several observations are worth noting.

- The bandwidth of $T(s)$ increases as velocity increases.
- $|S(s)|_\infty$ remains less than 5 dB.
- $|Q(s)|_\infty$ remains close to -10 dB until higher velocities are reached ($\approx$ 20 m/s). After that, as velocity increases, $|Q(s)|_\infty$ decreases. This is a direct result of setting $T(s)$ constant for velocities above 20 m/s.
- $PS(s)$ has a rather large DC gain that increases with velocity.
- $|PS(s)|_\infty$ increases as velocity increases.

These observations suggest that this controller will be robust to parameter variations, require less actuator effort as velocity increases, but suffer from actuator disturbances such as bump steer and unmodeled steering actuator effects. Furthermore, it is worth noting that the look-ahead error response to curvature is captured by $S(s)$. Therefore, this closed-loop system should reject the reference path's curvature when its frequency content is below 2-3 rad/s. However, when this frequency content is high, the system cannot reject this.

6.2.2 Simulation

The simulation described in Section 3 is used to evaluate each controller's performance in a variety of ODDs and routes.

6.2.3 Metrics

The motivating question behind this study is under which conditions do these controllers perform insufficiently? This is most simply answered by setting a maximum value of acceptable lateral error. However, since the position is estimated with an Inertial Navigation System, there are two possible definitions of lateral error. Both definitions are visualized in Figure 6.6. The estimated lateral error is the projection of the distance between the estimated vehicle pose (shown in grey) and the estimated closest point onto a unit vector that is both normal to the reference path and intersects the estimated vehicle's position. The estimated closest point and the true closest point are not the same because the estimated closest point must be computed online with the estimated vehicle pose. The true lateral error is defined the same as the estimated lateral error, except it uses the true vehicle pose and the true closest point. To date, we are unaware of any

previous work from the literature that simulates INS errors to investigate their impact on the path following controller performance.

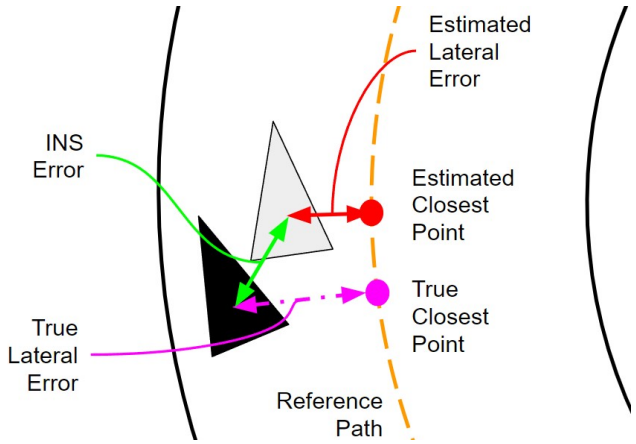


Figure 6.6: Visualization of True and Estimated Lateral Error Definitions

As the INS error decreases, the distance between the estimated and true closest points decreases (by definition of the INS error shown in Figure 6.6). It is also worth noting, that these differences are independent of the vehicle's yaw angle.

Having specified the two forms of lateral error, we can now provide a maximum value of acceptable true lateral error based on the lane and vehicle's widths. However, such a deterministic requirement oversimplifies the uncertainty in everyday driving. To mitigate this, we reframe the question as a probabilistic one: how often do the controllers' true lateral errors exceed some deterministic limit (recall that this is done in simulation so these errors are known)? Therefore, we propose a new metric:

$$P_f = \begin{cases} \dfrac{\sum_{n=1}^{N}(|err_{lat,n}| > \epsilon_{lat})}{N}, & \text{if } |err_{lat,n}| \leq err_{thresh} \\ 1, & \text{otherwise} \end{cases} \tag{6.8}$$

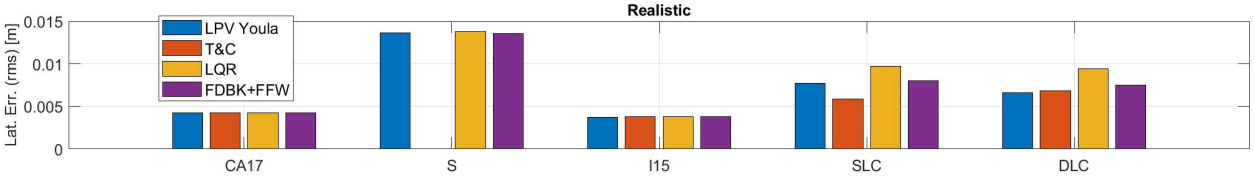
where P_f is the probability of failure, ϵ_{lat} is the maximum acceptable lateral error, err_{lat} are the samples of the true lateral error, N is the number of samples, $(|err_{lat,n}| > \epsilon_{lat})$ denotes a translation between the logical output to an integer (True maps to 1, False maps to 0), and err_{thresh} is a threshold of lateral error, beyond which, the maneuver is considered incomplete.

$\epsilon_{lat} = (w_{lane} - w_{veh})/2$, where w_{lane} is the width of a lane (3.6 m), and w_{veh} is the width of the vehicle (1.9 m). The conditional separation in Equation 6.8 is used to acknowledge the fact that after some large lateral error, the controller must be considered as failing to complete the full maneuver. In this case, 2 m is

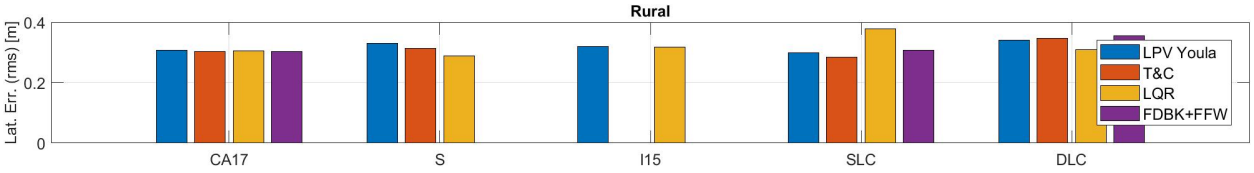
set as that threshold.

True lateral error is used in this definition instead of estimated lateral error because it includes the INS performance. A reasonable question to ask is if the INS performance changes between each controller. The simulation results in Figure 6.7 show that the INS performance is relatively consistent across controllers. The reader will notice that the collision avoidance maneuvers have more variance. This is because they are much shorter durations and so the variance is higher. The absent bars in Figure 6.7 are the result of the controller achieving a probability of failure of 1 during some point in the maneuver. This is discussed further in Section 6.3.

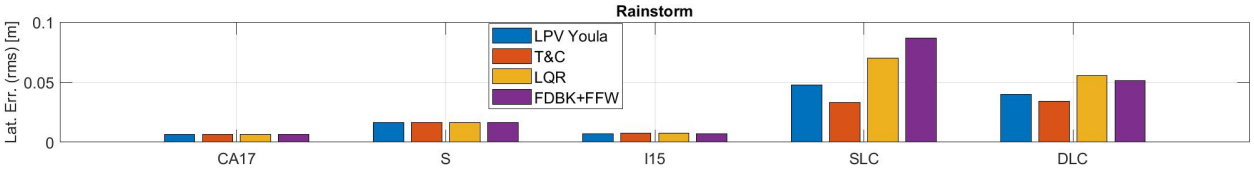
In addition to quantifying the level of sufficient performance, there is still the need to quantify each controller's relative performance and relative robustness. For relative performance, we employ scalar metrics: maximum absolute value (L1 norm) and root mean square (L2 norm) to capture the worst-case and average performance of each controller, respectively. Finally, to quantify robustness we may compare the change in these scalar metrics between each ODD.



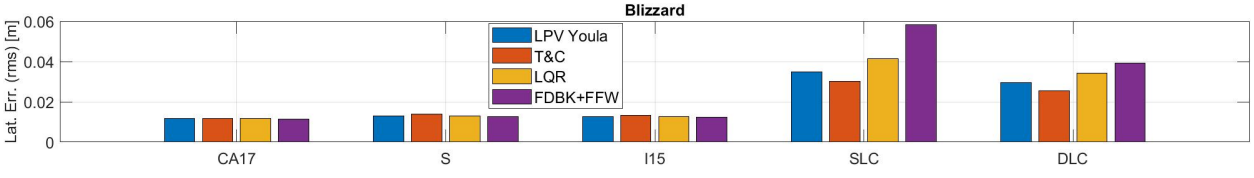
(a) INS Performance in the Realistic ODD



(b) INS Performance in the Rural ODD



(c) INS Performance in the Rainstorm ODD



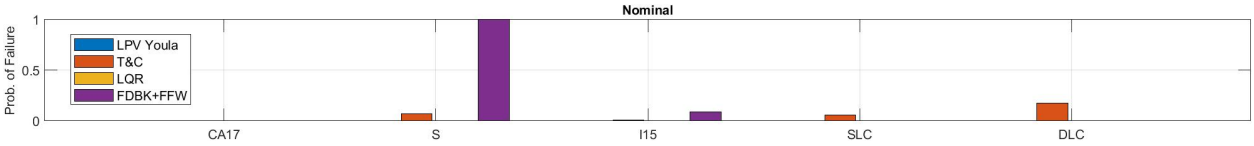
(d) INS Performance in the Blizzard ODD

Figure 6.7: INS Performance Across All ODDs

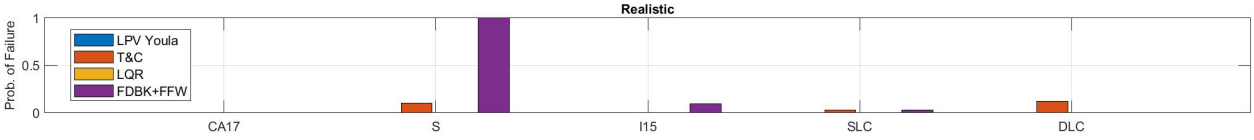
6.3 Results and Discussion

The first set of results to consider are the probabilities of failure, P_f, for each controller, which is shown in Figure 6.8. We begin with these results because they directly answer the core question of this work: which Operational Design Domain (ODD) has an existing control solution? For an ODD to be considered solved, there must exist at least one controller that achieves a $P_f = 0$ for all maneuvers.

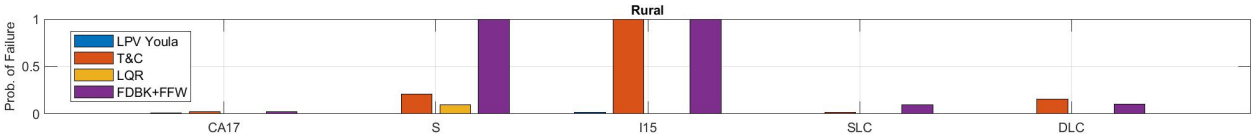
Figure 6.8 shows that all of the ODDs have at least one controller that achieves a $P_f = 0$ for all maneuvers. This suggests that there are controllers that provide a solution to all ODDs. The ODD with the most frequent nonzero probability of failure is the Rural ODD. The reason this ODD is so difficult is because precise control is required to achieve $P_f = 0$, and this is not achievable when the INS uses DGPS. The DGPS configuration of the simulation almost doubles the lateral INS error. This results in a smaller budget of acceptable error for the lateral controller. Also, as the INS error increases, the dynamics of the INS play a more significant role.



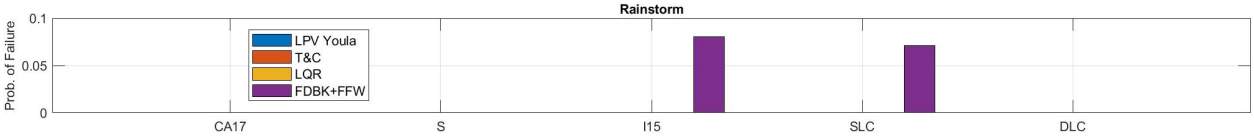
(a) Probability of Failure in the Nominal ODD



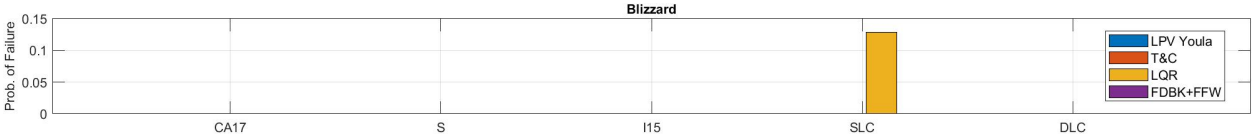
(b) Probability of Failure in the Realistic ODD



(c) Probability of Failure in the Rural ODD



(d) Probability of Failure in the Rainstorm ODD



(e) Probability of Failure in the Blizzard ODD

Figure 6.8: Probability of Failure Across All ODDs

Before proceeding to discuss Figures 6.9 and 6.10 it is necessary to explain the cause of the controllers that achieve $P_f = 1$. This is summarized in Table 6.2.

Table 6.2: Cause of Failures in Experiments

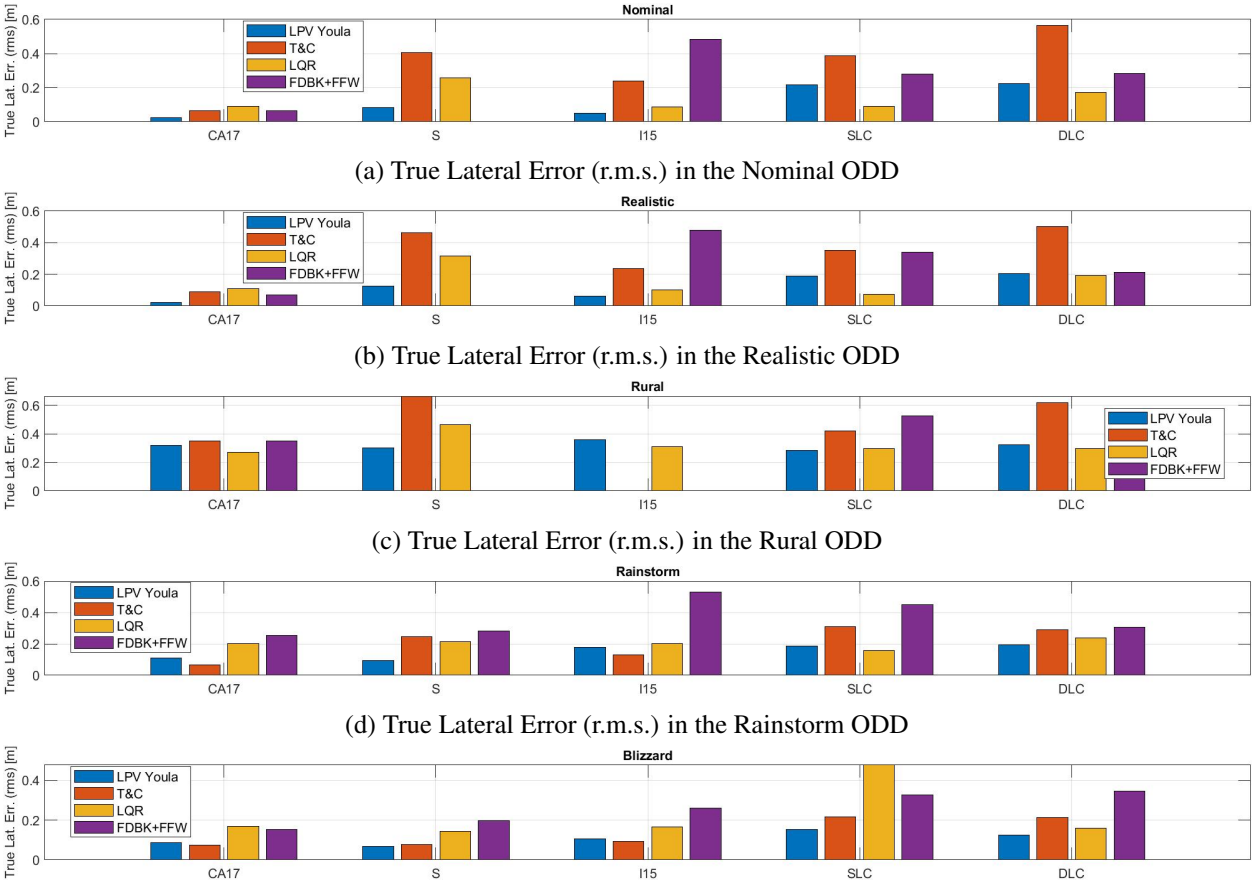
Controller(s)	Route(s)	ODD(s)	Cause
T & C	I15	Rural	Feedback noise causes excessive oscillations that exceed threshold
FDBK+FFW	S	Nominal, Realistic, Rural	Lateral error exceeds threshold
FDBK+FFW	I15	Rural	Lateral error exceeds threshold

Most controllers have zero or low P_f in the Rainstorm and Blizzard ODDs. Since this result is contrary to the typical belief that controlling a car is more difficult in rainy and snow conditions than in dry conditions, it suggests that the method of modifying the trajectories to ensure dynamic feasibility is overly conservative. In other words, the assumed relationship between μ and U_x results in improving the dynamic feasibility so much so that achievable performance is also improved. These results show there are likely additional variables that affect the controller performance. Such a conclusion has been demonstrated for real-world vehicles [26].

Furthermore, the ODD specifications in Table 3.9 would benefit from more environmental variables. One oversimplification is that the INS performance is the same for Realistic, Rainstorm, and Blizzard ODDs. In reality, INS performance can deteriorate in severe weather conditions. Assuming that the accuracy of the RTK-INS degrades to that of DGPS is difficult to validate. Further work is needed to determine how to degrade the INS performance to better simulate adverse weather conditions.

The results of this work demonstrate that it is feasible to find one controller capable of achieving safe performance in a wide variety of ODDs for a wide variety of driving tasks. The methods used in this work represent another step towards such a set of benchmarks with which all steering controllers can be evaluated. An earlier step in this direction was made in [37]. However, while [37] considers more controllers (and different ones) than those considered in this work, the introduction of ODDs, application of high-fidelity simulation, and a wider variety of maneuvers constitute a significant extension of their work.

6.3.1 Relative Performance



(a) True Lateral Error (r.m.s.) in the Nominal ODD

(b) True Lateral Error (r.m.s.) in the Realistic ODD

(c) True Lateral Error (r.m.s.) in the Rural ODD

(d) True Lateral Error (r.m.s.) in the Rainstorm ODD

(e) True Lateral Error (r.m.s.) in the Blizzard ODD

Figure 6.9: True Lateral Error (r.m.s.) Across All ODDs

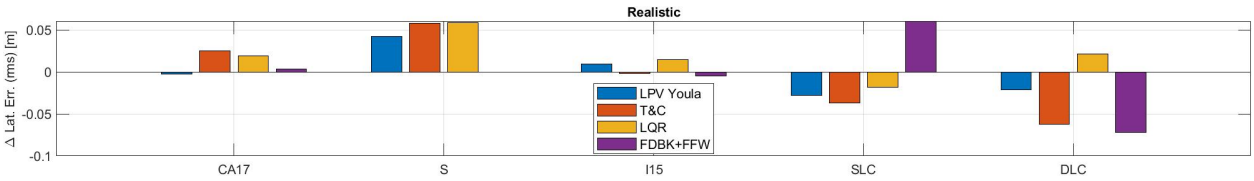
Figure 6.9 shows the r.m.s. value of the true lateral error for all controllers, maneuvers, and ODDs. Again, the absent bars correspond to the controllers that achieved $P_f = 1$. The controller that tends to have the lowest average lateral error across all maneuvers and ODDs is the LPV Youla controller. The LQR controller tends to be the second lowest lateral error. Finally, the FDBK+FFW and T & C controllers tend to be the worst performing controllers.

The difference in performances of the same controller between the collision avoidance maneuvers (SLC and DLC) and the highway maneuvers suggests that using collision avoidance to validate a controller does not guarantee the controller's generalizability to other tasks. If a controller is developed for both highway lane-keeping tasks and collision avoidance tasks, it should be tested on both types of paths.

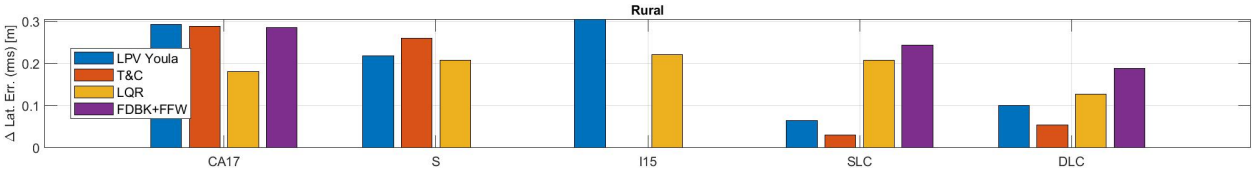
The maximum absolute true lateral error of each controller is not shown because the trends between

controllers, maneuvers, and ODDs are the same. Furthermore, Figure 6.8 also presents information related to the maximum absolute performance of each controller. So its inclusion does not add any additional information to the reader.

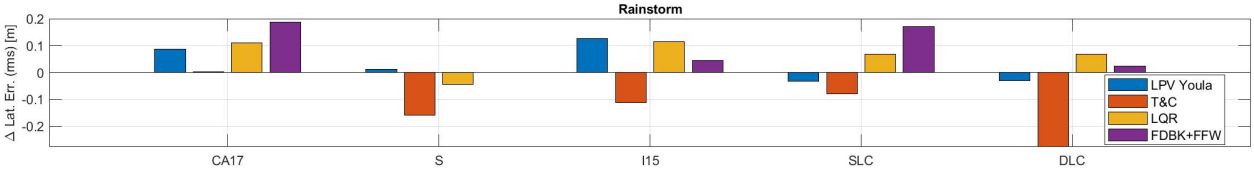
6.3.2 Relative Robustness



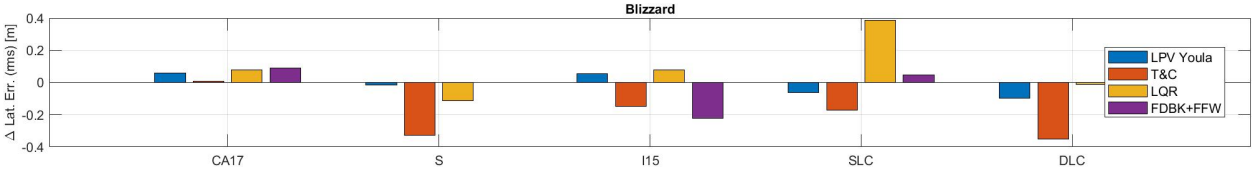
(a) Realistic ODD Change in True Lateral Error (r.m.s.) with respect to Nominal



(b) Rural ODD Change in True Lateral Error (r.m.s.) with respect to Nominal



(c) Rainstorm ODD Change in True Lateral Error (r.m.s.) with respect to Nominal



(d) Blizzard ODD Change in True Lateral Error (r.m.s.) with respect to Nominal

Figure 6.10: Change in True Lateral Error (r.m.s.) with respect to Nominal

Figure 6.10 shows the change in True Lateral Error r.m.s. for each ODD with respect to the Nominal ODD. While it is expected that the changes should be positive (the performance deteriorates), Figure 6.10 shows that some controllers improve in average performance with respect to the Nominal ODD. This is because of the previously mentioned velocity modifications to each maneuver for each ODD. This figure more clearly shows that the strategy used to modify the maneuver's velocity is overly conservative.

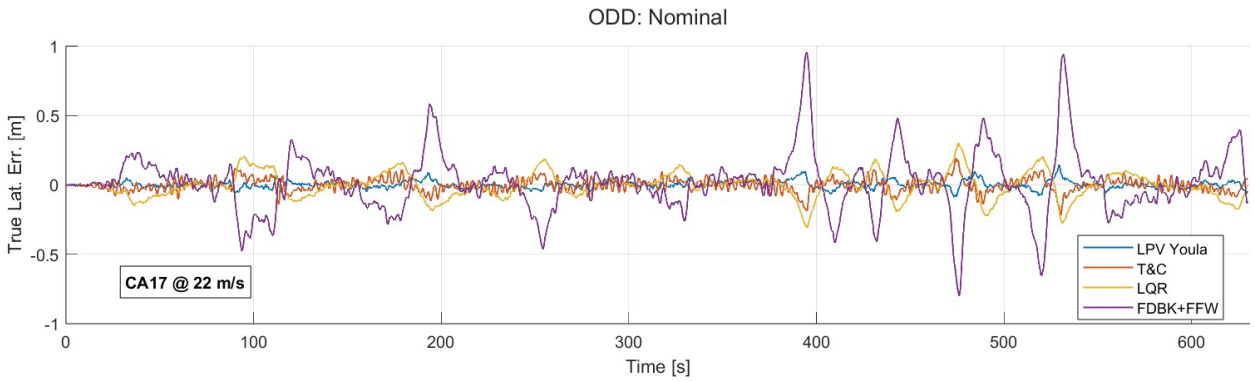
Figure 6.10 most clearly shows each controller's robustness with respect to the environmental conditions. The small changes observed in the Realistic ODD results show that all controllers are relatively robust against RTK feedback errors, system delays, and small wind gusts. The general increase in lateral error

for the Rural ODD show that all controllers suffer in performance when the feedback error is increased, and when the road surface quality is decreased. For highway maneuvers, it appears that the LQR controller changes the least. However, the results differ for the collision avoidance maneuvers, where the T & C controller degrades the least in performance. FDBK+FFW suffers the most in the Rural ODD. This is likely because this controller uses only a proportional feedback gain and relies heavily on its feedforward controller. The closed loop performance is therefore more fragile than the other more intricate feedback controllers.

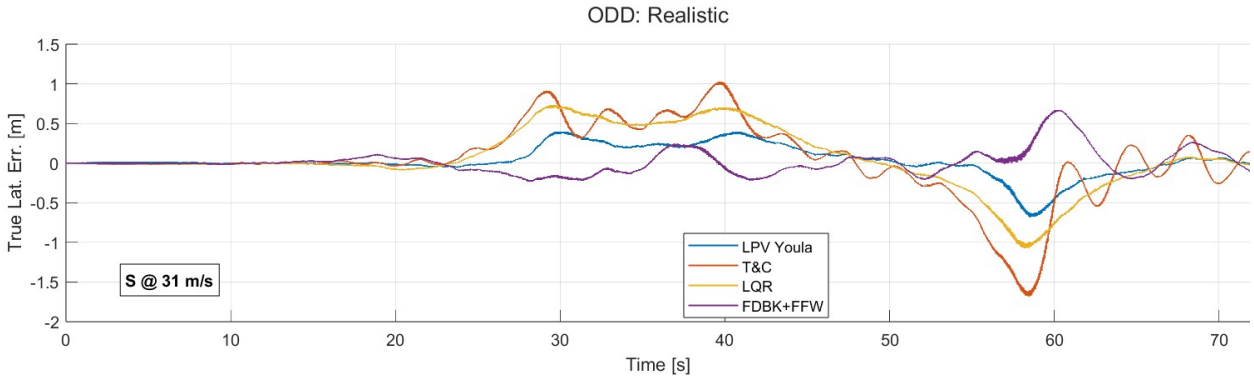
The results for the Rainstorm and Blizzard ODD show that most controllers can reject high winds and diminished road friction. The T & C controller improves the most in the Rainstorm and Rural ODDs.

6.3.3 Time Series Results

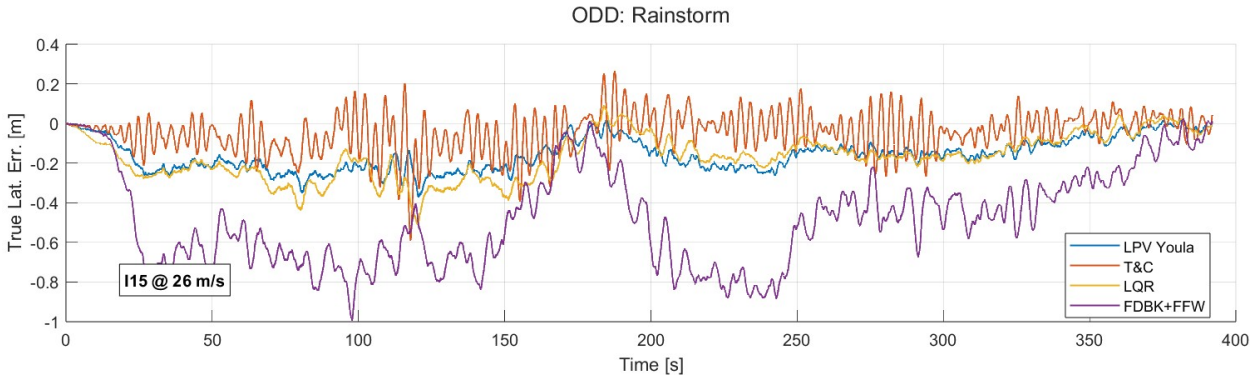
The metrics shown in the previous two subsection aggregate data across time and therefore hide the temporal information. Due to the large number of tests performed, these aggregate metrics are well-suited for describing relative performance and robustness. Furthermore the probability of failure metric (equation 6.8) helps further show that the maximum tracking errors do not exceed very large values. Despite these metrics, there is still value in seeing the temporal behavior of each controller's lateral error signal. Figure 6.11 will present a sampling of the many maneuver-ODD combinations.



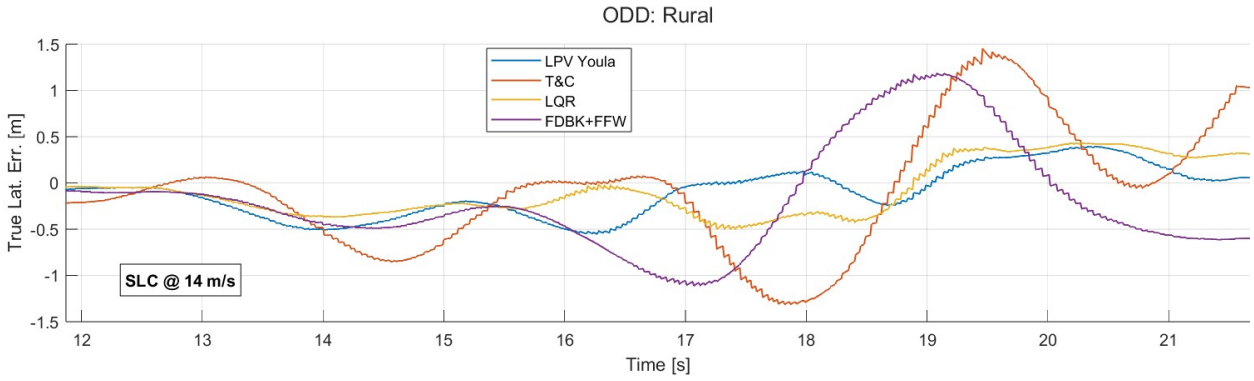
(a) True Lateral Error for Controllers on CA-17 maneuver in Nominal ODD



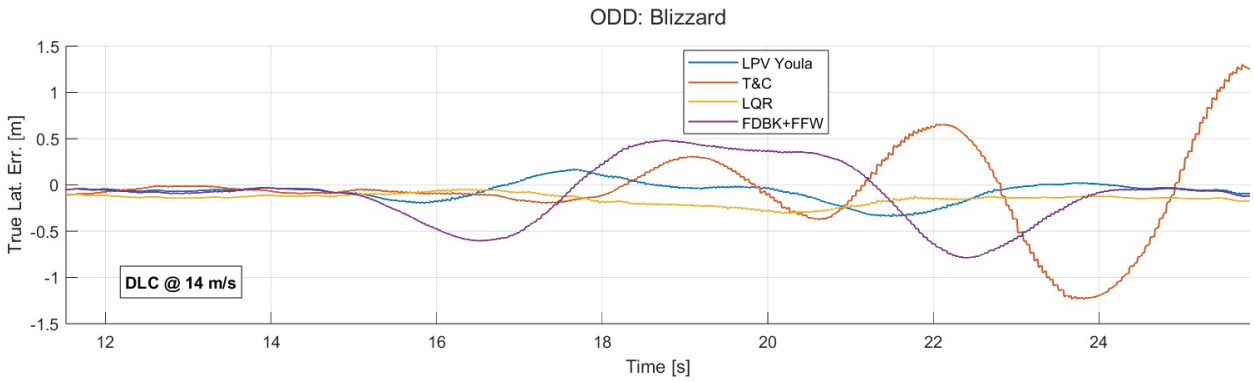
(b) True Lateral Error for Controllers on S maneuver in Realistic ODD



(c) True Lateral Error for controllers on I-15 maneuver in Rainstorm ODD



(d) True Lateral Error for controllers on Single Lane Change maneuver in Rural ODD



(e) True Lateral Error for controllers on Double Lane Change in Blizzard ODD

Figure 6.11: Controller true lateral errors for a selection of maneuver/ODD combinations

From these time series figures several observations can be made:

- The T&C Controller suffers from oscillations of about 0.2 Hz on the highway like maneuvers.
- The T&C controller appears to be unstable at the end of the Single Lane Change (SLC) and Double Lane Change (DLC) maneuvers.
- The FDBK+FFW controller suffers from large curvatures (large peaks in error exhibited in the CA-17 maneuver correspond with large curvatures).
- The LQR and LPV Youla controllers have similar error trajectories.
- The FDBK+FFW controller suffers the most from the environmental disturbances in the Rainstorm ODD (largest lateral error).

6.4 Conclusion

This work uses a custom, high-fidelity simulator to evaluate four controllers on five maneuvers in five ODDs. By framing this comparison with ODDs, environmental disturbances can be grouped to form simulations of realistic scenarios. Through the combination of all five maneuvers and the five ODDs, 25 unique scenarios are used to evaluate if a controller can achieve pre-defined performance requirements. The reference trajectories are designed to represent a wide range of possible maneuvers, including lane-keeping, overtaking on a highly curved road, and abrupt collision avoidance maneuvers.

The results in Figures 6.8, 6.9, and 6.10 show that all of the five ODDs considered have controllers capable of achieving a pre-defined performance requirement. More significantly, all of these controllers require incredibly low computational power. This suggests that more computationally heavy controllers, such as Model Predictive Control, are not needed to solve these ODDs.

The claims in this work are rather significant, so before concluding, it is necessary to qualify the results. A large number of assumptions were made in this work that simplified these ODDs into simpler (and solvable) problems. The assumptions that may have the most significant influence on these results are:

1. Stochastic Disturbances: One simulation was performed per controller, route, and ODD combination. The underlying assumption is that this simulation represents the worst case in that condition, which is a rather significant oversimplification. In the future, the simulator's real-time factor can be improved and a Monte Carlo experiment can be used to evaluate the worst-case, average, and best-case performance of each controller. In this sense, it makes sense to redefine the Probability of Failure accordingly.

2. Longitudinal Control: The longitudinal control performance is not considered in this work. However, the longitudinal controller must also meet pre-defined performance requirements.

3. Simplified road disturbances: The vertical road disturbances are modeled as colored noise according to ISO 8608 [115]. This neglects potholes and other sudden road jumps. These types of disturbances might present destabilizing conditions that are not captured in this work.

4. Adverse weather conditions assumed to be uniform: The adverse weather conditions (Rainstorm and Blizzard ODDs) assume the road friction is uniform (with the addition of colored noise). In reality, puddles and ice patches can exist that cause large and sudden changes in friction. Furthermore, the simulation is incapable of simulating hydroplaning. All of these disturbances greatly complicate simulating adverse weather conditions.

The outcome of this work is to suggest that many of the control problems in steering an Automated Driving System (ADS) with only a front steering actuator are solved. Further work in this area should be directed away from developing new, more complex controllers for lane-keeping in ideal weather conditions. Instead, more focus should be given to understanding and modeling the significant disturbances that make driving in adverse road conditions challenging and dangerous. Additionally, more effort should be spent on

developing longitudinal and lateral control architectures capable of achieving specified joint performances derived from safety requirements.

Chapter 7

Investigating Torque-Based Steering Control

Path-following (or lateral motion) controllers are commonly designed assuming that the interface to the vehicle requires commanding a steering angle. This angle may be a steering wheel angle or a road wheel angle. However, an alternative interface may exist in that a steering torque is commanded to the vehicle. These two interfaces propose different control actuation and may have a significant impact on lateral motion control safety and comfort. This chapter proposes studying different modeling and control architectures for a 2019 Jeep Grand Cherokee by using a CarSim model that is similar to the Jeep. After investigating different modeling and control options, the most suitable combination is applied to the Jeep. This methodology is validated using closed-course and public road testing.

7.1 Introduction

This chapter will investigate the different ways lateral motion controllers can use torque requests instead of steering angle requests. This is highly motivated by the real-world example of controlling the 2019 Jeep Grand Cherokee. However, rather than test various control architectures on the real vehicle, a virtual vehicle in CarSim will be used to study the performances in different environmental conditions. The primary reason for using a virtual vehicle is safety. Testing prototype control architectures on real vehicles in varying weather conditions is dangerous and requires extensive resources to complete successfully. Furthermore, using a virtual vehicle improves reproducibility and can make for a more fair comparison between architectures. However, due to the large amount of resources needed to create a valid model of the 2019 Jeep Grand Cherokee in CarSim, the virtual vehicle will be a generic full-size SUV that comes with CarSim 2020. Since

the Jeep Grand Cherokee is also a full-size SUV, the model is not all that different, and it is assumed that much of what is learned from the CarSim model can be applied to the Jeep Grand Cherokee.

Two different control architectures will be studied by simulating their behavior on two maneuvers in two environmental conditions. The first maneuver, a double lane change, is designed to represent collision avoidance scenarios. The second maneuver, the S road, is designed to represent an aggressive overtaking maneuver on a freeway. The two environmental conditions will model realistic environmental conditions found on a dry day and during a blizzard.

7.1.1 Motivation

The reason to investigate lateral motion control with a steering torque input is highly practical. Most vehicles today that support an Advanced Driving Assist System (ADAS) provide a CAN signal for commanding a torque request to the steering actuator [49]. This means that lateral motion controllers can be implemented on most vehicles, but the interface is based on torque requests. In contrast, many lateral motion control methods are based on steering angle requests [135, 18, 19, 46, 123, 121, 125]. Therefore, some modification to the control architecture is required to make the controllers more widely applicable.

One notable work in the literature is [55] which develops a Linear Parameter Varying (LPV) controller that considers the nonlinear behavior of the steering system. Their task is to design a lateral motion controller given a low-level actuator controller that tracks a reference steering wheel angle. Similar to this chapter, it is assumed not much is known about the actuator. They identify an actuator model using grey-box identification techniques. The model consists of a PID controller, linear steering column dynamics, and Stribeck friction. They argue that the lateral motion controller cannot meet performance requirements without considering this identified actuator model. They propose that the oscillations caused by the steering actuator nonlinearities are not easily eliminated. They argue that this presents a fundamental tradeoff between comfort and tracking performance. An increase in tracking performance (closed-loop bandwidth) results in greater steering oscillation. Their solution is to provide an additional scheduling parameter that acts to raise or lower the lateral motion controller's bandwidth. At low bandwidth, the steering oscillations are diminished, but so is tracking performance. At high bandwidth, the steering oscillations are amplified, but the tracking performance is greatly improved.

The same research group explored an alternative path-tracking formulation [165] that is tasked with driving the derivative of the look-ahead error to 0. They develop a controller using the H-infinity mixed

synthesis technique. By using the derivative of the look-ahead error rather than the look-ahead error as the system output, the plant has one less integrator. Having fewer poles tends to reduce the control design's fundamental trade-off between performance and robustness [61]. This work also considers the steering actuator as a black-box, closed-loop system and finds that a low-order transfer function model with transport delay is sufficient for modeling the system.

Steering actuators are commonly designed to assist the human driver to improve the feel of controlling the vehicle. There is a wide variety of steering system configurations. For simplicity, this chapter will focus on using Electric Power-Assisted Steering (EPAS), of which several varieties are used in passenger vehicles. There are two common mechanical systems: rack-and-pinion and recirculating ball (or pitman arm). There are three common locations for an electric motor in a rack-and-pinion system: on the rack, on the column, or on the pinion [108]. For example, the 2019 Jeep Grand Cherokee uses a rack-and-pinion mechanical system with a rack-mounted electric motor. The full-size SUV in CarSim 2020 uses a rack-and-pinion system with a rack-mounted electric motor. However, CarSim allows the user to choose a manual (not power-assisted) rack-and-pinion system as well.

For rack assist EPAS, the power assist functionality first senses the torque applied by the human driver. Then, an electric motor applies an assistive torque to the rack. The design of this system is highly oriented towards ensuring a safe and comfortable feeling for the human driver when controlling the vehicle. There is commonly a boost curve used to shape the steering feel of the car. This boost curve can be linear, piece-wise linear, or some nonlinear curve [108]. It is also commonly scheduled on vehicle velocity such that more torque is applied by the rack motor at low velocities than at high velocities.

Physics-based modeling drives the majority of control development for EPAS [108, 147, 200, 128, 23]. This is because it is often assumed that the control engineer tasked with designing the EPAS has complete access to the design of the mechanical system. However, when the vehicle has been purchased to be retrofitted for automated driving, little is known about the physical system without having to reverse engineer the system. This is the situation encountered with a 2019 Jeep Grand Cherokee purchased from AutonomouStuff. Therefore, in addition to studying how to use a rack-EPAS, we will also assume nothing else is known except for two aspects: (1) we can measure the steering wheel angle, and (2) we command a torque that is normalized by the motor's maximum torque. This maximum torque is also unknown.

Further complicating this task is that it is not known what modifications are made to the torque request by the control unit on the EPAS. For instance, boost curves may be applied to the requested torque.

Furthermore, an additional motor may be mounted on the column to provide redundancy and improve the reliability of steering-based ADAS features. This motivates studying black-box modeling of three different configurations of rack-EPAS:

1. Direct application of torque from the rack-mounted motor.
2. Torque applied to the steering column with torque boost.
3. Torque applied to the steering column without torque boost.

In Section 7.2, black-box models will be compared against the frequency response of the 2019 Jeep Grand Cherokee's EPAS. Several types of data-driven models can be used in this task [163, 34]. However, we only consider Linear Time Invariant and Linear Parameter Varying models in the form of transfer functions and state space models. The model that most closely matches the Jeep Grand Cherokee EPAS will be used to study torque-based lateral motion control development.

Following the identification of an actuator model for control development, a lateral motion controller will be developed to command a torque (the specific application will be determined after selecting which of the three configurations most closely match the Jeep Grand Cherokee) to the vehicle. Two possible architectures may be favorable for control development:

1. Model Inversion: Keep the lateral motion controller as one that commands the steering wheel angle and invert the actuator model to convert the steering wheel angle to a steering wheel torque. This model inversion may be dynamic or static, but no feedback is performed around the steering wheel angle.
2. Cascade Control: Keep the lateral motion controller as one that commands the steering wheel angle. Design a feedback controller that tracks the commanded steering wheel angle by commanding steering wheel torque.

The control design for each of these architectures is detailed in Section 7.3.

Following the control design, each method will be simulated on two maneuvers and in two environmental conditions. The simulation results will be discussed in Section 7.4.

7.2 Actuator Modeling

The 2019 Jeep Grand Cherokee is equipped with a New Eagle Drive-by-wire device that provides its controller to track a reference steering wheel angle by commanding steering torque and measuring the resulting steering wheel angle. This is an example of an inner loop controller in a cascade control architecture. However, through experimental testing, it was found that this inner loop controller resulted in a resonance at 1.7 Hz and was deemed unsuitable. Therefore, this low-level controller will be bypassed in the car, and steering torque will be commanded directly. The specific configuration of the steering actuator is not known and the software on the Jeep Grand Cherokee normalizes this input with an unknown scalar. Therefore, for the remainder of this Section, the input will be called steering torque, which is intentionally vague because the exact type of input is unknown.

The input and output variables that are most natural are steer torque and steering wheel angle because they can be directly measured. The steering torque is left intentionally vague because of the previously discussed uncertainty about the vehicle's EPAS configuration. Selecting the steering wheel angle is more natural than selecting the road wheel angle because road wheel angles cannot be directly measured. Furthermore, most lateral motion controllers command road wheel angle because they are based on some version of the bicycle car model [121, 103, 164, 125, 46]. The bicycle model ignores the differences between left and right road wheel angles and treats them as a single road wheel angle. This makes it natural to assume a constant ratio between the steering wheel angle and the combined road wheel angle. Therefore, the steering ratio will convert the output of the steering actuator model to the input of the bicycle car model.

Because it is not known how the control module on the Jeep Grand Cherokee's EPAS modifies the requested steer torque, three different EPAS configurations will be modeled. All models will be based on black-box techniques that identify a transfer function to capture the input-output behavior of the system. The input will be steer torque (N-m) and the output will be steering wheel angle (rad).

The first assumption in modeling the steering system is that its dynamics can be modeled independently of the vehicle's dynamics, but must consider velocity. Specifically, consider the vehicle transfer function as being $Y(s, U_x) = G_{veh}(s, U_x)\delta_m(s, U_x)$, where $Y(s, U_x)$ is lateral velocity and yaw rate, and $\delta_m(s, U_x)$ is the measured steering road wheel angle. The U_x term is included to indicate the model's dependence on longitudinal velocity U_x. Let the steering actuator dynamics be represented by the transfer function: $G_{act}(s, U_x) = \frac{\delta_m(s,U_x)}{\tau_r(s,U_x)}$, where $\tau_r(s, U_x)$ is the requested steering torque. This first assump-

tion therefore assumes that G_{veh} and G_{act} are independent and parameterized by U_x. In other words, $G_p(s, U_x) = \frac{Y(s,U_x)}{\tau_r(s,U_x)} \approx G_{veh}(s, U_x)G_{act}(s, U_x)$. This assumption will be validated in Section 7.5.1 after having identified G_{act} and then combining G_{act} and G_{veh} and comparing them to simulation experiments.

To collect data that properly excites the steering system, a chirp signal is used. The frequency linearly increases from 0.001 Hz (Simulink requires a strictly positive lower bound) to 5 Hz (highest frequency of interest). These inputs will be commanded after the vehicle reaches a steady-state longitudinal velocity of 5, 10, 15, 25, and 35 m/s. The amplitude of the chirp signal is tuned at each velocity so that the maximum lateral acceleration does not exceed 0.3 g's. Beyond this acceleration, highly nonlinear effects may break the previously stated assumption. To evaluate the nonlinearity of the system, three different amplitudes are tested at each velocity. The first is intended to achieve a maximum of 0.3 g's, the second is intended to achieve approximately 0.1 g's, and the third should be somewhere in between.

Before analyzing the data and identifying a model, it is worth pausing and applying knowledge of vehicle and tire dynamics to predict trends that will be observed in the data. This will provide an intuition for how the system should behave and will improve understanding of the identified black-box model. A key aspect to note is that this Chapter treats the steering actuator as including the tire-road interactions, alternative formulations may treat the tire-road interactions as an input to the steering actuator. This perspective is taken because tests were not performed to identify the steering actuator independent of the tire-road interface (which can be done by lifting the front tires off the ground). To develop this intuition consider the following thought experiment:

> Suppose the driving task is to apply an appropriate torque on the steering wheel such that the steering wheel angle is a constant value. Now let the vehicle accelerate from rest. A tire generates lateral forces by deforming [75]. The maximum deformation commonly occurs behind the center of the contact patch. The distance between the center of the contact patch and the point of maximum lateral tire deformation is called the pneumatic trail. The lateral force that is generated is offset from the center of the contact patch by this pneumatic trail.
>
> Furthermore, the rotation axis rarely passes through the center of the contact patch and instead is at some point in front (along the vehicle's axis) of the tire's contact patch. The angle between the vertical axis passing through the tire's contact patch and the rotation axis is called the caster angle (the distance between the two points is called the mechanical trail)[75].

The combined effects of the pneumatic trail, caster angle, and lateral force create a torque that always acts to align the tire in its direction of travel (aligning torque). Experiments show that this aligning torque is a nonlinear function of the tire's lateral slip angle, but at low slip angles, the relationship is linear such that as the slip angle increases, the aligning torque increases. To hold a constant steering angle, the mechanical steering system must apply a torque that counteracts this aligning torque. Under steady-state conditions, the torque that the mechanical steering system applies to the tire is proportional to the torque applied to the steering wheel. Using these relationships we can now form a prediction of how much torque must be applied to the steering wheel as the vehicle's velocity increases.

As the vehicle's velocity increases in a constant steering maneuver, the lateral acceleration will also increase. This acceleration is generated by increasing lateral tire forces caused by increasing slip angles. Because the slip angles are increasing, the aligning torque is also increasing. Therefore, more torque must be applied to the steering wheel to overcome this increasing aligning torque. In short, as the vehicle's velocity increases, more torque is required on the steering wheel to maintain a constant steering wheel angle. In dynamic model terms: as the velocity increases the low-frequency gain of the steering actuator will decrease.

7.2.1 Rack Motor Torque

The first EPAS configuration to test is when the steering torque is directly passed to the motor on the rack without any modification. CarSim 2020 simulates the output torque of the rack motor as a linear force directly applied to the rack. For simplicity, we assume the column pinion radius to be the same as the rack-mounted pinion radius. Therefore, to convert the linear force to a torque, the pinion radius will be used. In CarSim, the default C-factor for the steering column is 40 mm/rev. The C-factor is a typical characteristic of a rack-and-pinion system that specifies that the rack translates 40 mm for every revolution of the pinion. The resulting pinion radius is 6.366 mm.

The frequency response of the steering wheel angle to rack motor torque is shown in Figure 7.1. The legend indicates the velocity of the vehicle during the testing and the amplitude of the chirp signal.

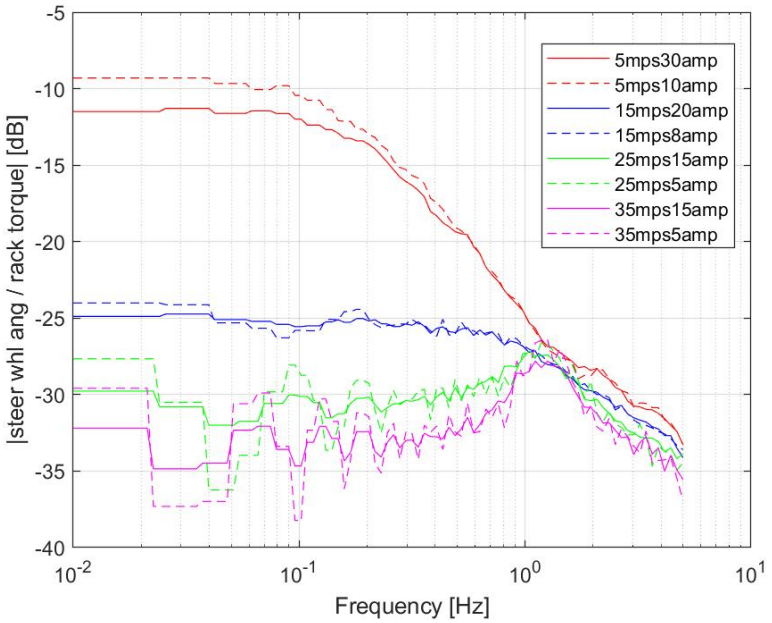


Figure 7.1: CarSim SUV frequency response of the steering wheel angle [rad] to rack torque [N-m]

7.2.2 Column Motor with Rack Assist

The second EPAS configuration to test is when the steering torque is passed to the motor on the column and a motor on the rack is used to assist the column motor. This is identical to modeling a human-applied torque input on the steering wheel with a rack EPAS. The frequency response of the steering wheel angle to the rack-assisted column torque is shown in Figure 7.2.

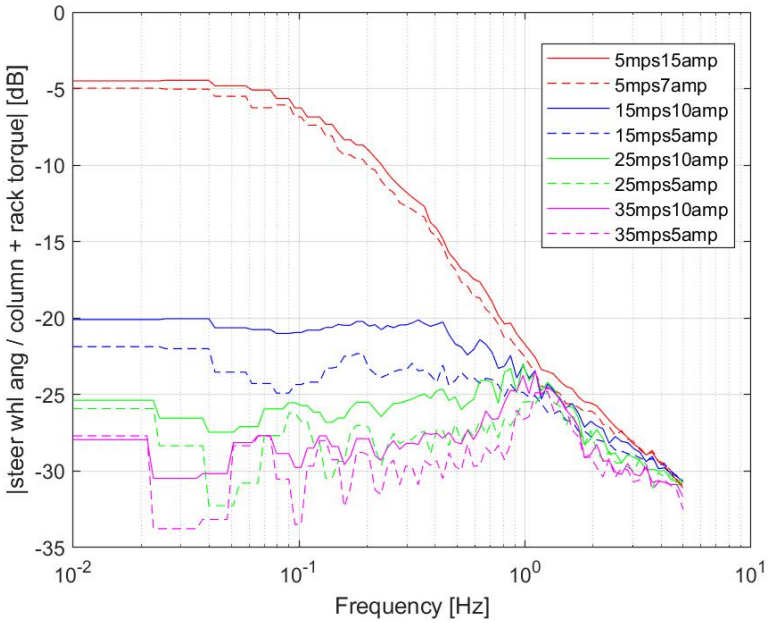


Figure 7.2: CarSim SUV frequency response of the steering wheel angle [rad] to column torque [N-m] with rack assist

7.2.3 Column Motor without Rack Assist

The third EPAS configuration to test is when the steering torque is passed, unmodified, to the motor on the column. No additional torque is applied to the rack by the rack EPAS. The frequency response of the steering wheel angle to the column torque without rack assist is shown in Figure 7.3.

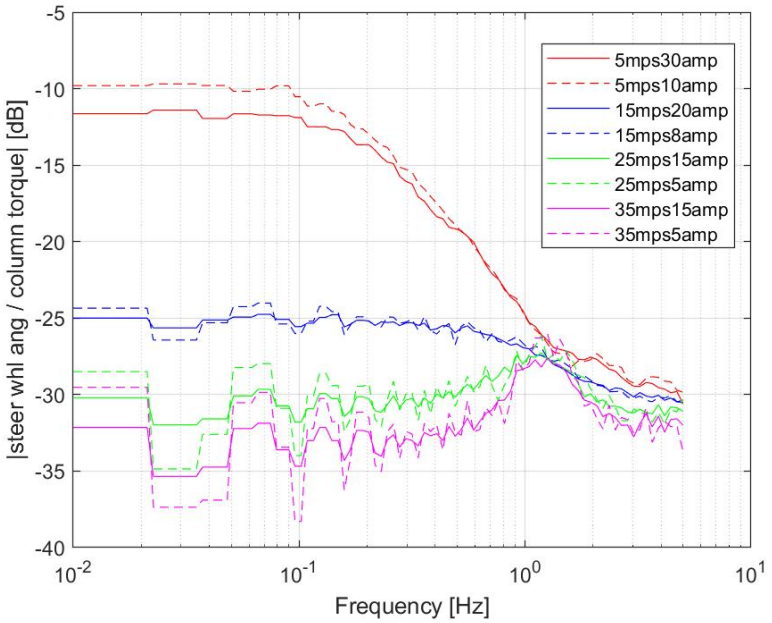


Figure 7.3: CarSim SUV frequency response of the steering wheel angle [rad] to column torque [N-m] without rack assist

7.2.4 Steering Configuration Selection

Figures 7.1, 7.2, and 7.3 shows the frequency response of CarSim's three steering actuator configurations: rack-motor, column-motor with rack-motor-assist, and column-motor without rack-motor-assist, respectively. Comparing these Figures shows that they all exhibit very similar frequency responses. They all have a low-frequency gain that decreases with increasing velocity. They all have a drop-off of approximately -20 dB per decade at around 1 Hz. Figures 7.3 and 7.1, in particular, are nearly identical. This indicates that CarSim's steering system does not change behavior very much if the system is actuated by a motor on the column or on the rack. However, Figure 7.2 differs from the other two frequency responses by having a higher low-frequency gain. This is most likely the result of having the additional motor on the rack assist the torque applied to the column.

The frequency response of each EPAS configuration can now be compared to the Jeep Grand Cherokee's EPAS frequency response. One key difference between the CarSim and Jeep Grand Cherokee interfaces is that the Jeep interface uses a normalized torque request. It is not known how this normalization is done so the frequency response must be done with respect to this normalized steer torque request. The frequency response of the Jeep's steering wheel angle to the normalized steer torque request is shown in Figure 7.4.

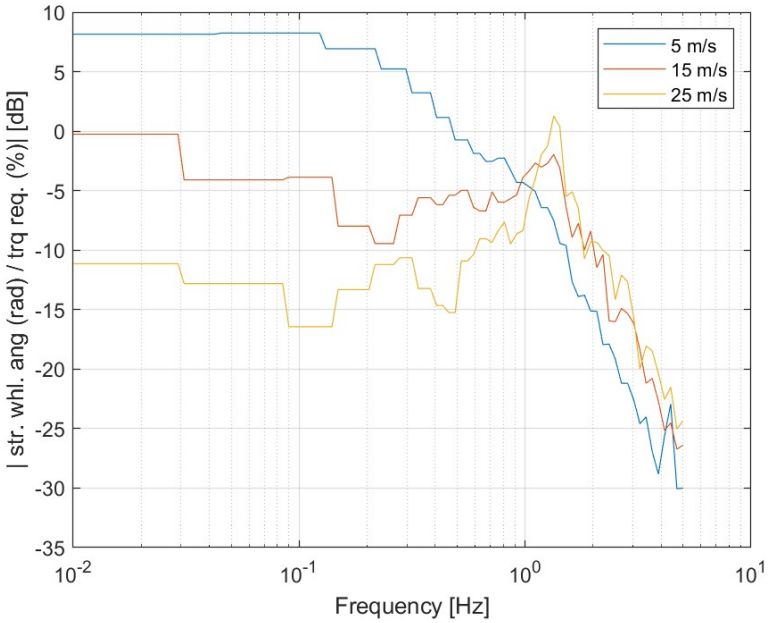


Figure 7.4: Frequency response of the 2019 Jeep Grand Cherokee steering wheel angle [rad] to steer torque request [%]

No testing was done at 35 m/s because there was not enough space on the test track to safely conduct the data collection maneuver. By comparison, the frequency response of the CarSim steering systems and the Jeep steering systems look similar. Figures 7.1, 7.2, 7.3, and 7.4 all have a low-frequency gain that decreases with increasing velocity. This observation corroborates the thought experiment proposed earlier. Between 5 and 25 m/s, the system's gain decreases by approximately 20 dB. The difference between the gains at different velocities decreases as the frequency approaches 1 Hz.

The most obvious difference between the Jeep and CarSim's frequency response is the low-frequency gains. The Jeep's low-frequency gains are between 10 and 20 dB higher than the CarSim responses. This is most likely attributed to the normalization of the torque request. Furthermore, the Jeep's frequency response above 1 Hz appears to decrease at a higher rate than the CarSim steering system. This might be a result of not having an electric motor model in the CarSim steering system and instead using a direct force or torque on the rack or column.

The result is that all of the CarSim actuator configurations are similar to the Jeep. Because of this inconclusive result, additional information is necessary. From repair manuals and after-market components, it is known that there is an electric motor on the rack for the Jeep Grand Cherokee. However, it is not known if there is an additional motor on the column. Having such a motor likely increases the costs of the steering

system, and since the Jeep is a mass-produced vehicle that is designed with manufacturing cost in mind, it is assumed that there is not an additional motor on the column. Therefore, the most reasonable configuration to assume is the rack EPAS. For the rest of this chapter, only the rack motor is used as the steering actuator.

7.2.5 Model Selection

Having selected the CarSim steering configuration, the next step is to choose an appropriate model to capture the system's dynamics. Recall that we are assuming that there is not enough information to develop a physics-based model. Instead, we consider the steering system a black box and attempt to identify models that are useful for control purposes. The types of models that are preferred are Linear Time Invariant (LTI) models. These can be represented as either state-space models or transfer functions.

A major tradeoff in using a black box model (a model whose parameters have no direct physical interpretation) is that the resulting states of the model do not have a physical interpretation. Therefore, the states cannot be directly measured, preventing the application of state feedback control without state observers.

The velocity dependence shown in Figure 7.1 makes finding a single LTI transfer function that accurately models G_{act} challenging. Because of this dependence on a system parameter, a Linear Parameter Varying (LPV) model seems like a natural choice.

We begin searching for an LPV model that can appropriately capture the system's response. Here is when another assumption becomes important to consider: parameter dependence. LPV plants can exhibit behaviors that are not well explained by their LTI counterparts when the dynamics of the scheduled parameter are fast relative to the dynamics of the system [169]. We therefore assume that the dynamics of U_x have negligible effect. In other words, we assume that the experiments performed at fixed velocities generalize well to experiments performed at varying velocities. This is a reasonable assumption because the dynamics of U_x are much slower than the steering dynamics. The limitations of this assumption will be checked when the selected model is validated.

We will begin by trying to fit a transfer function to each frozen-parameter frequency response. To enable LPV model construction, the transfer function should have the same structure for all velocities. It was found that a model structure having two poles and one zero achieves an acceptable balance between model complexity and fit for all velocities. The frequency response of the experiments and the frozen-parameter models is shown in Figure 7.5.

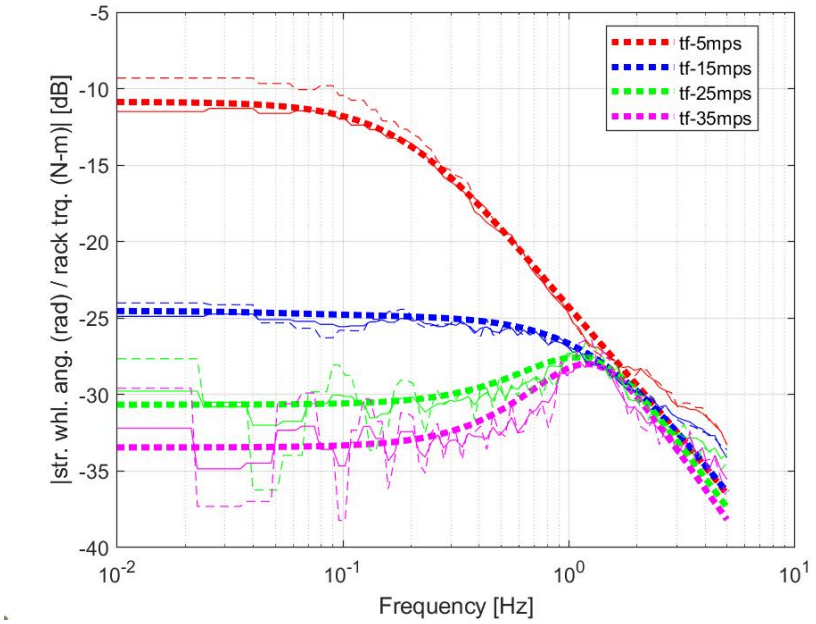


Figure 7.5: Steering Actuator Model Frequency Response at Varying Velocity

By identifying a transfer function at each velocity, independent of each other, this method has generated four different transfer function models. The coefficients of the transfer function: $G_{act} = \frac{A_1 s + A_2}{s^2 + B_1 s + B_2}$ are shown with respect to U_x in Figure 7.6. Interpreting these parameters as those of an LPV model requires assuming the parameter's behavior between the set velocities. A linear, piece-wise assumption is shown in Figure 7.6. It is important to note that the resulting trends in parameters are not linear with U_x. The result is that the LPV model cannot be written as a single model whose parameters form an affine (or linear) relationship with U_x. This prevents using polytopic LPV control methods and therefore reduces the applicable types of LPV control techniques.

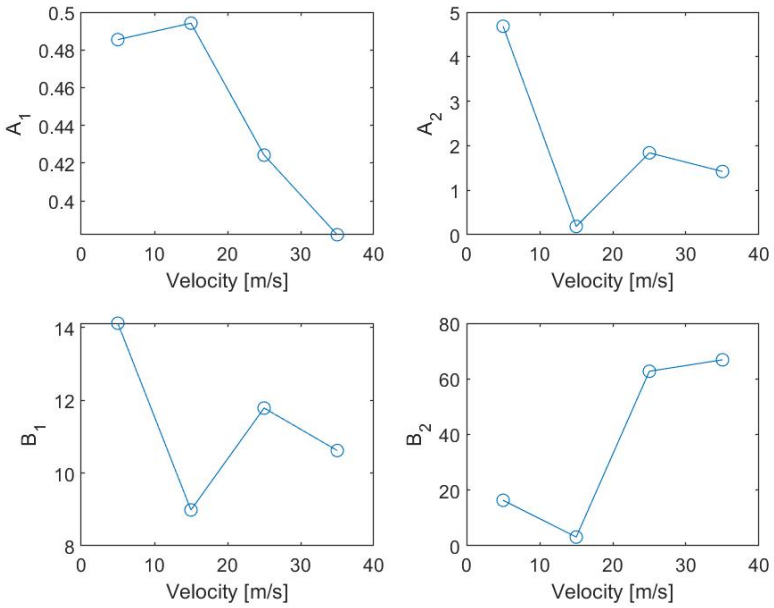


Figure 7.6: Coefficients for actuator models of form: $G_{act} = \frac{A_1 s + A_2}{s^2 + B_1 s + B_2}$

Rather than constrain the control design, we may explore state space models whose parameters are affine with respect to U_x. The approach is to estimate the parameters of the state space realization: $G_{act} = \begin{bmatrix} A(U_x) & B(U_x) \\ C(U_x) & 0 \end{bmatrix}$. The goal now is to identify $A(U_x)$, $B(U_x)$, and $C(U_x)$ such that each matrix A, B, and C is affine with U_x and best fits the experimental data for all velocities. To do this we employ Matlab's nonlinear grey-box identification tools from the System Identification Toolbox. This tooling enables the estimation of parameters for nonlinear models from experimental data.

There are a wide variety of state space structures that we may attempt to identify. The choice of structure must balance the quality of fit, the number of parameters, and its usefulness for control development. Each nonzero element in the state matrices produces two parameters to estimate: (1) the coefficient that multiplies U_x, and (2) the bias term. For instance, $A_{11} = \alpha_{11} U_x + a_{11}$, requires both α_{11} and a_{11} to be estimated. Therefore, it is ideal to make each state matrix sparse to reduce the number of parameters needed to estimate. We may begin with a controllable canonical form:

$$
\begin{aligned}
A(U_x) &= \begin{bmatrix} 0 & 1 & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & 1 \\ \alpha_1 U_x + a_1 & \alpha_2 U_x + a_2 & ... & \alpha_n U_x + a_n \end{bmatrix} \\
B(U_x) &= \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \\
C(U_x) &= \begin{bmatrix} \gamma_1 U_x + c_1 & \gamma_2 U_x + c_2 & ... & \gamma_{1n} U_x + c_n \end{bmatrix}
\end{aligned}
\tag{7.1}
$$

If the order of the model is n then there will be $4n$ parameters to estimate. Therefore, a balance must be made between model complexity (in terms of free parameters) and the model's fit quality. Through a comparison of a second-order and third-order model's fit, it was determined that the fit qualities improvement is not worth the additional four free parameters of the model. Furthermore, reducing the model order improves the simplicity of the control design. Figure 7.7 shows the frequency response of the identified affine LPV model with the experimental frequency responses.

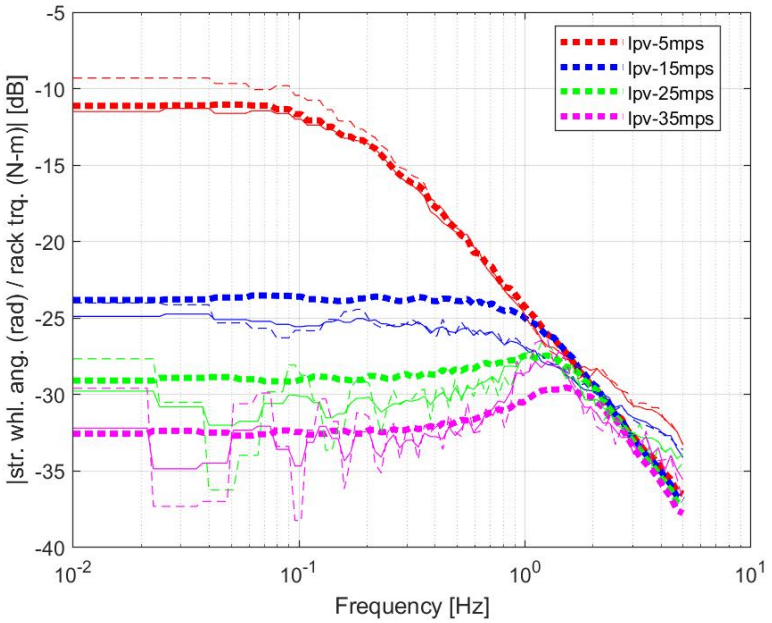


Figure 7.7: Frequency response of the affine LPV model with CarSim SUV experimental frequency responses. The thick, dashed lines are the model's response. The thin, dashed lines are the CarSim's steering response. The colors correspond to longitudinal velocities shown in the legend.

Comparing Figure 7.5 with Figure 7.7 shows that the affine model has a very similar frequency response to that of the piece-wise LPV model. The most obvious difference is that the affine LPV model does not fit the 15 m/s experiment and the 35 m/s as well as the piece-wise model.

To validate this affine LPV model we must check the model against data collected at velocities other than those used to identify the model. It is also important to excite the longitudinal dynamics to check for limitations in the frozen-parameter experimental design. To achieve these goals, a velocity profile will be created that is a series of step inputs. The first step input commands the velocity from 0 to 10 m/s. The second brings the car to 20 m/s. The third: 30 m/s. The fourth, and final step, brings the car back to 0 m/s. Each step is spaced 10 seconds from each other. The steering torque command is a square wave that oscillates between +/- 8 N-m every 10 seconds. The result is a maneuver that accelerates and abruptly steers every 10 seconds. At the end of the maneuver, the car is aggressively decelerated and steered.

Figure 7.8 shows the experimental results with the Affine LPV's predictions. The y-axis is the measured steering wheel angle. Before 5 seconds, the vehicle is accelerating from rest. During this low velocity and high longitudinal acceleration (approximately 0.6 g's), the predicted steering wheel angle is smaller than the measured steering wheel. This amount of model discrepancy is acceptable.

At 30 seconds, the vehicle applies full brake and comes to a stop within 4 seconds. The longitudinal acceleration during this time is between -0.6 and -1.0 g's. At this same time, steering generates a large lateral acceleration of approximately 8 m/s^2. Furthermore, the Electronic Stability Control (ESC) activates and modifies the applied brake pressure to maintain directional stability. This model discrepancy shows that the model will not accurately predict the steering wheel angle during high lateral and longitudinal accelerations. For now, this discrepancy will be deemed acceptable. However, this motivates additional research on lateral motion control under extremely hard braking during cornering with ESC interaction.

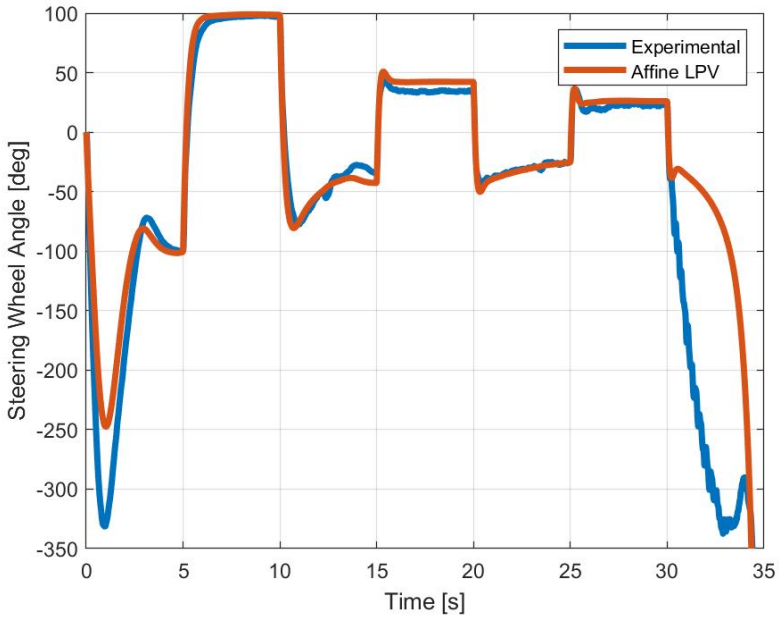


Figure 7.8: Time response of the CarSim SUV steering system and the affine LPV model

7.3 Control Design

This section will cover the design of two different control architectures. The first control architecture will use an inversion of the piece-wise LPV model identified in the previous section. This model inversion will convert the road wheel angle signal to a steering-rack torque in an open-loop fashion. The second control architecture will be a cascade controller (also called an inner and outer loop controller). The inner loop controller will track a desired road wheel angle by commanding steering torque and measuring the steering wheel angle. The outer loop will be the same lateral motion controller used in the first architecture so that a fair comparison can be made. This lateral motion controller is presented before either of these two

architectures since it is used in both architectures.

7.3.1 Angle-Based lateral motion Control

The design of a lateral motion controller that commands a steering wheel angle has been the subject of this thesis. There are a wide variety of methods to use here, each with its benefits and drawbacks. However, the goal of this chapter is to investigate torque-based lateral motion controllers. While there are just as many possible control designs for steering torque, it may be desirable to extend angle-based lateral motion control since so much development has already been done.

To study the methods of extending angle-based lateral motion control with simulation experiments, it is important to isolate the effects of the extension from the angle-based lateral motion controller. To achieve this, the angle-based lateral motion controller will be designed without consideration of the steering actuator or the extension method, and the same angle-based lateral motion controller will be used in both control architectures. Therefore, any differences between the results of the two architectures are solely from the method of extension. Furthermore, since this section is to study the effects of two methods of extension, the lateral controller is selected to be simple, but still reasonable. To this end, a Linear Time-Invariant (LTI) dynamic output feedback controller is designed with the mixed sensitivity H-infinity method. This balances robustness and performance.

However, it is worth noting that there is interaction between angle-based lateral motion controllers and whatever method is used to extend it to be torque-based. For instance, in the cascade control architecture, the lateral motion controller should not have a higher bandwidth than the steering controller to avoid inner and outer loop interactions that can cause instability or oscillations. Therefore, the lateral motion controller is designed with a bandwidth of approximately 1 Hz. Note that is the design for the CarSim vehicle, and a higher outer loop bandwidth is made possible by the high inner-loop bandwidth. The closed loop frequency response is shown in Figure 7.9.

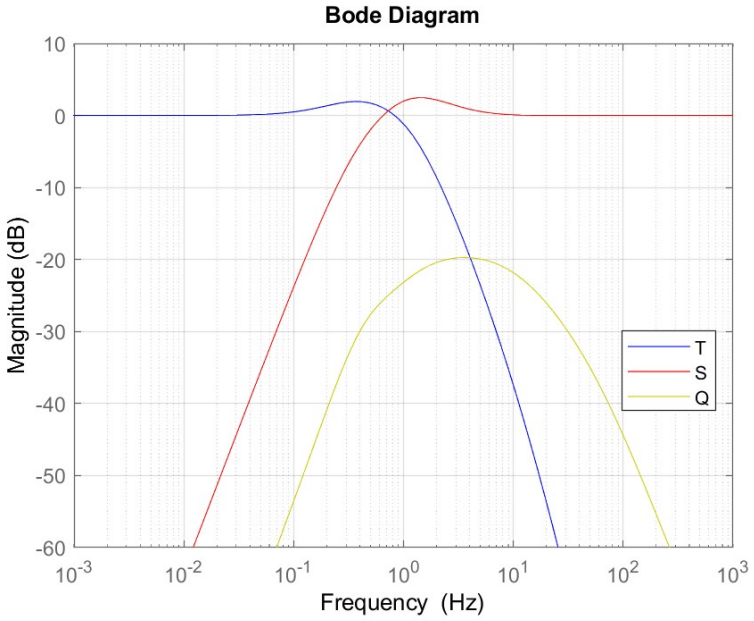


Figure 7.9: Frequency response of the LTI angle-based lateral motion controller for the CarSim SUV

7.3.2 Model Inversion

The model inversion control architecture is shown in Figure 7.10.

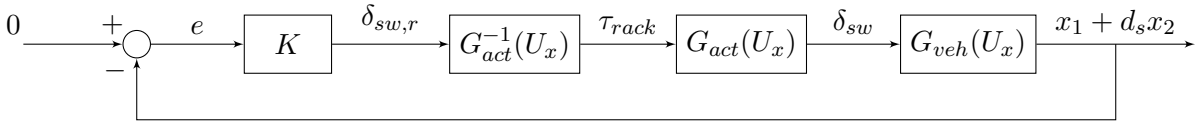


Figure 7.10: Model inversion control architecture

The piece-wise LPV model, whose parameters are shown in Figure 7.6, is a single-input single-output transfer function with one zero and two poles. The inverse of this model is improper. We therefore choose to add additional poles to achieve a proper transfer function and reduce high-frequency sensitivity. The model is $G_{inv}(s, U_x) = G_{act}(s, U_x)^{-1} G_1(s)$ where $G_1(s)$ has two high frequency poles. The first pole is to ensure G_{inv} is proper. The second pole is to bring G_{inv}'s gain down in high frequencies, thereby reducing sensitivity to high frequencies.

During the first attempt at selecting these poles, the double poles are set to achieve a cutoff frequency of 5 Hz. However, the simulation of the resulting system suffered from steering oscillations. These oscillations diminish when the cutoff frequency of these poles is increased. A possible explanation for this is that as the

cutoff frequency of the poles is increased, the inverted model becomes more accurate. This model accuracy reduces the amount of model discrepancy that the lateral motion controller has to compensate for. The final tuning of these poles placed them at a cutoff of 15 Hz (time constant of 0.0106 s). The frequency response of the inverted transfer function models is shown in Figure 7.11.

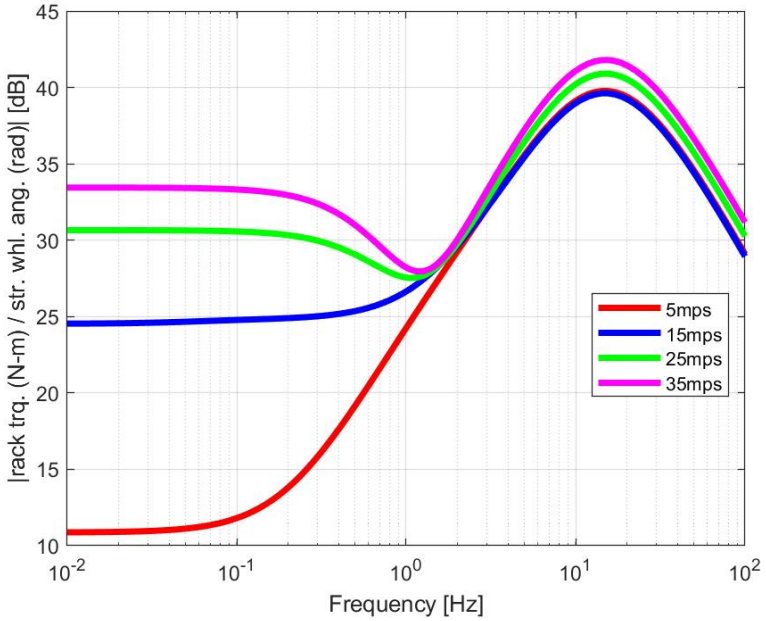


Figure 7.11: Frequency response of the inverted actuator model

These four transfer functions are then discretized using Tustin's method. The resulting discrete-time transfer functions can be combined into an LPV transfer function model whose coefficients are piecewise linear. When implementing this in Simulink, a discrete-time IIR filter is implemented in direct form II, and the coefficients are computed using lookup tables.

7.3.3 Cascade Control

The cascade control architecture is shown in Figure 7.12.

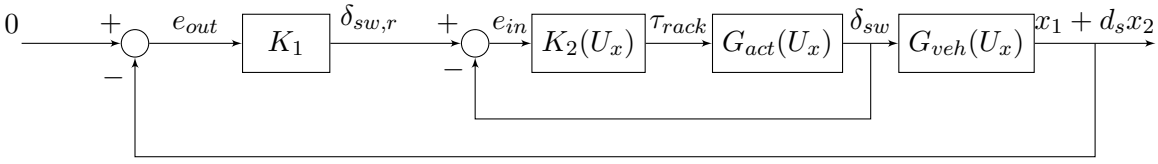


Figure 7.12: Cascade control architecture

The cascade control architecture is made possible because steering actuators that support ADAS functionalities typically have a steering wheel angle sensor [108]. There is a wide array of control techniques that can be used for the inner loop controller. State feedback cannot be used (without a state observer) because the model states do not have physical interpretations. However, output feedback and observer-based design can be used. Furthermore, it has already been shown that the actuator dynamics are dependent on velocity, and an affine LPV model can accurately capture the system's behavior. Therefore, it is natural to consider polytopic LPV dynamic output feedback control techniques such as [10].

To set our desired requirements for the system we will primarily be concerned with achieving low steady-state error, high bandwidth, and reasonably low actuation effort. The low steady-state error should be less than 1%. Therefore, the low-frequency gain of the sensitivity transfer function $S(s)$ should be less than -40 dB. The need for high bandwidth is driven by the cascade architecture. The inner control loop's bandwidth should be greater than the outer loop's bandwidth to simplify tuning. The goal is then to achieve an inner loop bandwidth that is greater than 2 Hz. Since this desired inner loop bandwidth is greater than the steering actuator's bandwidth, high actuator effort is expected. This high actuator effort is tolerable because simulation results show the commanded torques are reasonable for real electric motors. With this in mind, the corner frequency of $S(s)$ should be greater than 2 Hz.

To determine the maximum allowable actuator effort, we return to the actuator's interface. Because the rack-and-pinion is designed to make it so that a human can control the vehicle, it is reasonable to set the allowable actuator effort according to what a human can achieve. In reality, an electric motor would be applying a torque, most of which can outperform a human in delivering torque. While there are a wide variety of variables that influence the maximum torque humans can achieve, the average female can apply 62 N-m, and the average male can apply 133 N-m to the steering wheel [145]. Therefore, an assumption of 50 N-m is a conservative maximum (this is also used to simulate the electric motor's maximum torque).

This maximum torque should not be exceeded while driving. Driving consists of a wide variety of maneuvers, most of which can be performed with very low steering angles or slow dynamics. For instance, parking is a maneuver at low velocities that typically requires large steering wheel angles, but the steering dynamics do not need to be fast. Furthermore, typical highway driving is done at high speeds and with very small steering angles. One maneuver that requires fast steering dynamics and large steering wheel angles is a collision avoidance maneuver. Therefore, we will consider a steering request signal typical of a double-lane change maneuver. When conducting a double lane change maneuver, the typical driver does not use more

than 5 degrees of tire angle to pass at 25 m/s (although some more aggressive drivers can use a little more) [181]. For the CarSim vehicle (steer ratio of approximately 16), this maps to a steering wheel angle of 80 degrees.

In summary, we desire an inner loop controller that achieves (1) a low-frequency gain of $S(s)$ of less than -40 dB, (2) a closed-loop bandwidth greater than 2 Hz, and (3) a time response to a double lane change using less than 50 N-m.

To achieve these requirements, we will use a mixed synthesis H-infinity control design. The frequency weights on the complementary sensitivity transfer function $T(s)$ and the sensitivity transfer function $S(s)$ are shaped to require low gains and a bandwidth greater than 2 Hz. The frequency weight on $Q(s)$ is tuned to ensure the first two requirements are met.

There are several ways in which the H-infinity problem may be solved. Using the LMI methods allows for both the LTI and LPV problems to be solved using the same framework [71, 43, 72, 8, 9, 10, 7, 11]. The LPV problem is solved using Matlab's hinfgs function. The resulting LPV system's (frozen-parameter) frequency responses are shown in Figure 7.13.

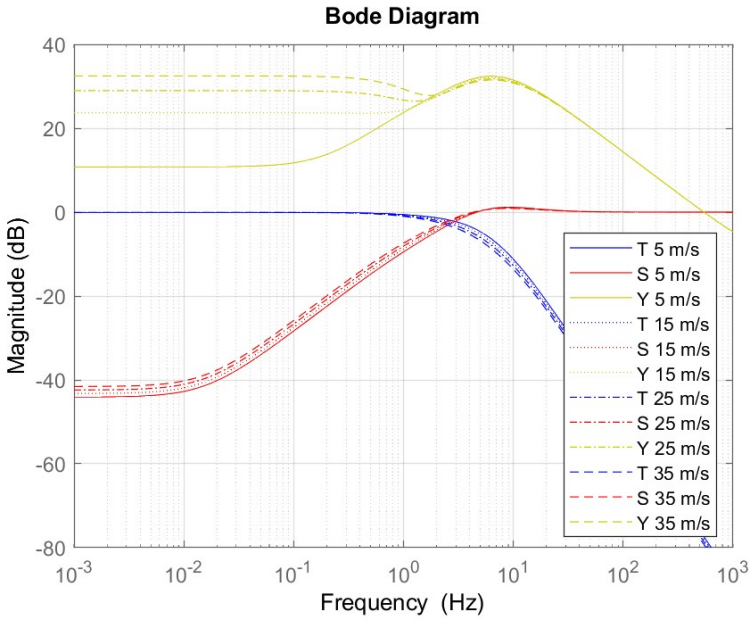


Figure 7.13: Frequency response of the CarSim SUV LPV H-infinity Inner Loop Controller

7.4 Simulation Results

Figure 7.14 shows that the Torque Inverse controller achieves a smaller maximum absolute lateral position error in the S maneuver under the Realistic ODD. Under all other combinations, the Cascade controller achieves a lower maximum absolute lateral error. Figures 7.15, 7.16, and 7.17 show that the Torque Inverse controller commands the larger steering angles, steering torques, and front lateral slip angles in all combinations. However, the Cascade controller's steering angles and steering torques tend to have more oscillations than the Torque Inverse controller (except for the very large periods of peaks).

What is rather remarkable is that the Torque Inverse controller achieves very large front lateral slip angles. These are so large that they are well into the tire saturation. Yet the lateral error remains somewhat reasonable. This indicates that the Torque Inverse controller is quite robustly stable against tire force saturation. However, these results probably exceed the validity of the CarSim tire model and should not be relied upon as reasonable predictions of real-world behavior.

The results for the Cascade controller, by contrast, are far more reasonable. The steering angle and steering torque exhibit more oscillations than what might be considered comfortable for a passenger. However, this is likely the result of setting the outer loop controller's bandwidth too close to the inner loop's bandwidth. Further tuning should be capable of improving this oscillation, as will be shown later in Section 7.5. Overall, the Cascade controller's performance is preferred over the Torque Inverse controller's performance.

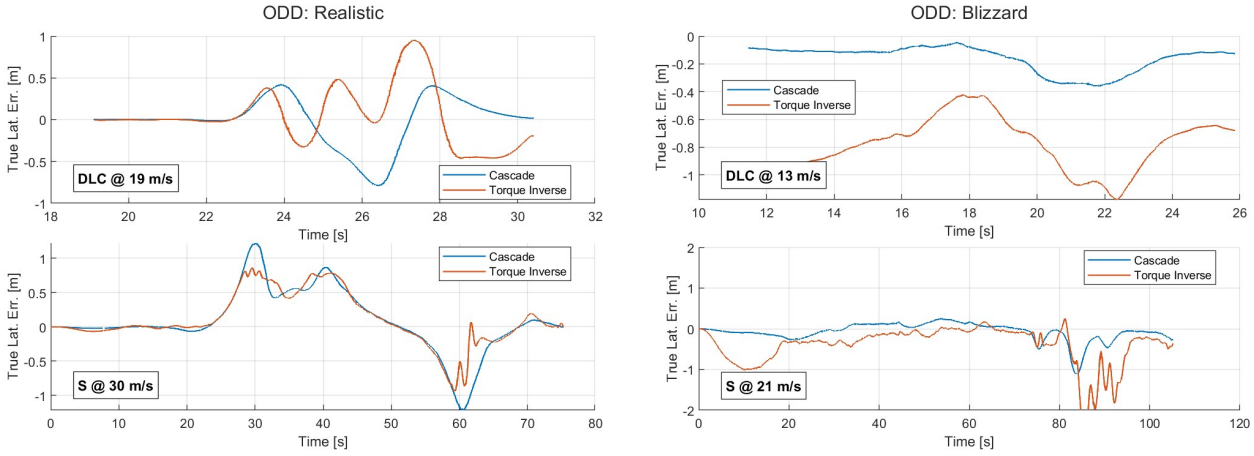


Figure 7.14: True Lateral Error of the Model Inverse and Cascade Architectures in the Realistic and Blizzard ODD

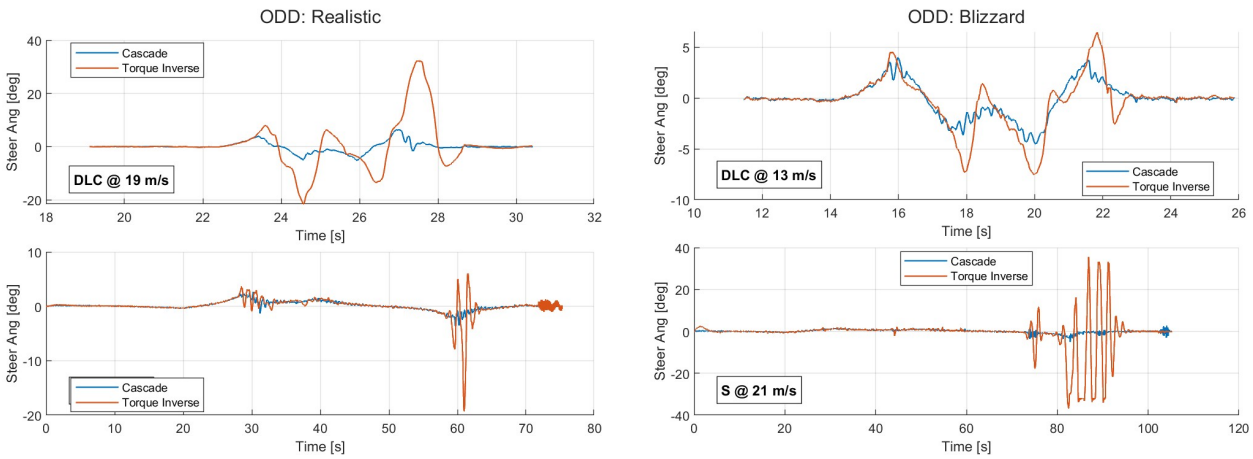


Figure 7.15: Measured Road Wheel Angle (“Steer Ang" in the axis label)
of the Model Inverse and Cascade Architectures in the Realistic and Blizzard ODD

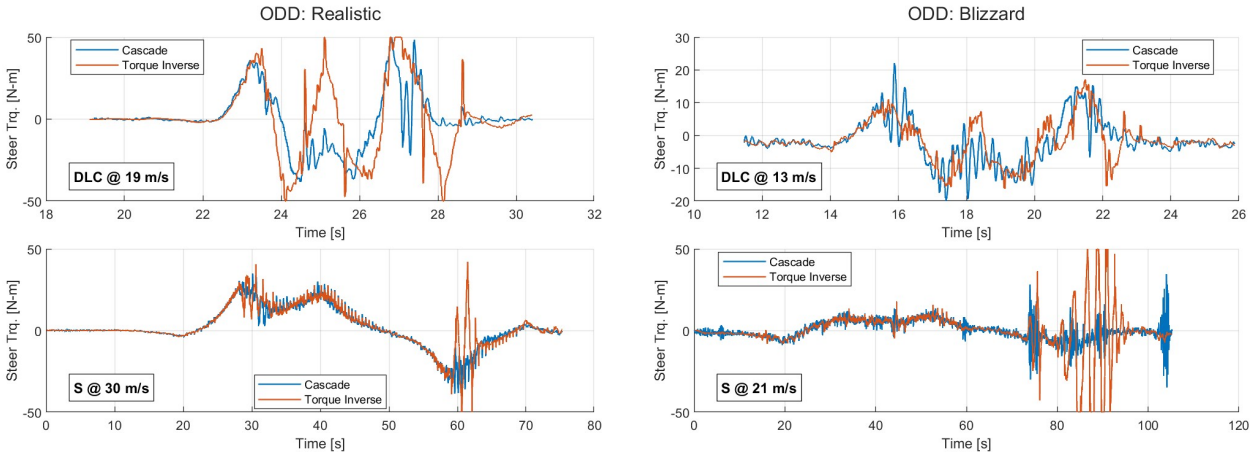


Figure 7.16: Commanded Rack Torque (“Steer Trq." in the axis labels)
of the Model Inverse and Cascade Architectures in the Realistic and Blizzard ODD

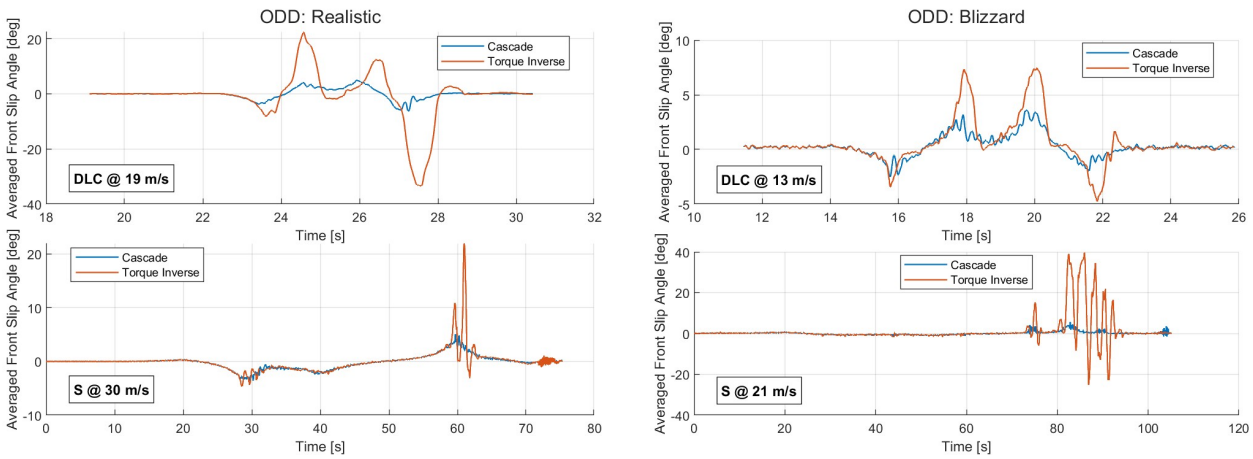


Figure 7.17: Averaged Front Lateral Slip Angles for the Model Inverse and Cascade Architectures in the Realistic and Blizzard ODD

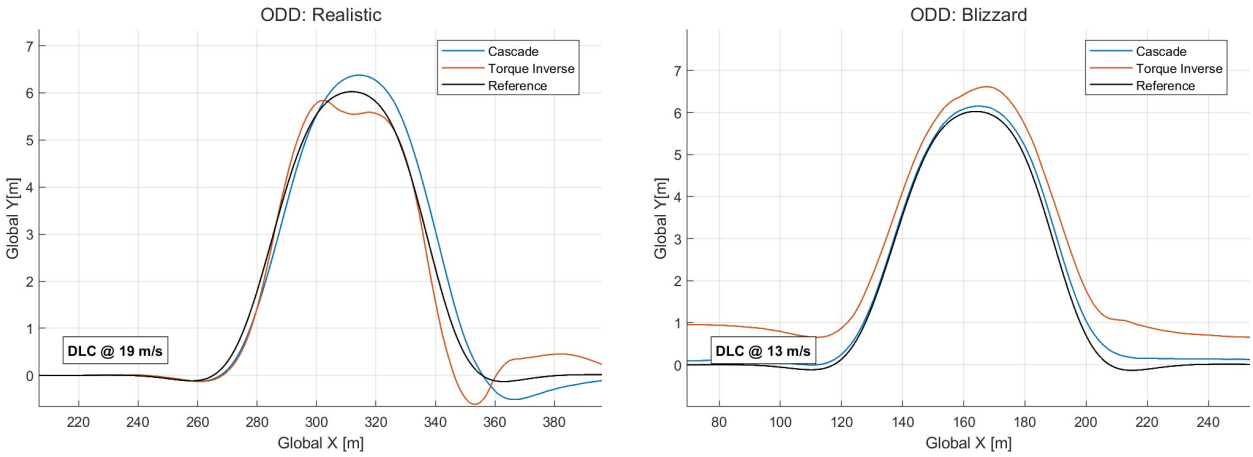


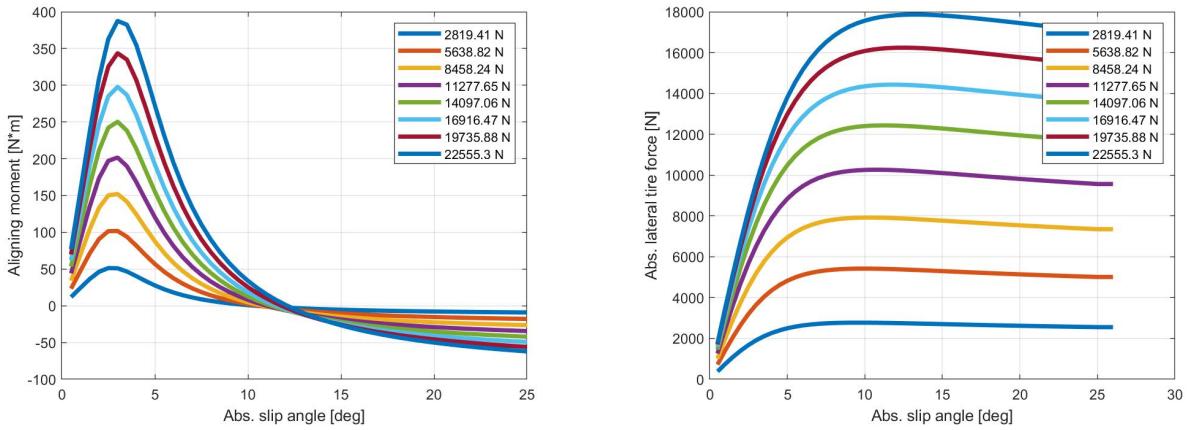
Figure 7.18: Top Down View of the Model Inverse and Cascade Architectures for the Double Lane Change in the Realistic and Blizzard ODD

To explain the Torque Inverse controller's extremely large lateral slip angles, we may consider the relationship between lateral slip angles and the applied rack torques. The EPAS system is stable (and well-modeled by a second-order system) if the applied rack torque does not greatly exceed the opposing external torques. A major contributor to the opposing torques is the aligning moment from the front tires. These aligning moments are nonlinear functions with lateral slip angles. The tire model used in CarSim 2020, as shown in Figure 7.19a, provides the maximum aligning moment at an absolute slip angle of 3 degrees. Beyond this, the aligning moment quickly diminishes with increasing slip angle. At very high slip angles, the aligning moment even changes sign.

The reason this is significant is that for a constant torque input on the rack if the lateral slip angle

increases beyond 3 degrees, the steering angle will begin accelerating because the opposing moment from the front tires is now decreasing. This acceleration of the steering angle will cause an increase in lateral slip angle, which will in turn cause a greater decrease in opposing forces on the rack. The result will be a quickly destabilizing steering system.

Furthermore, to address this without performing feedback control on the steering system, one may impose a design requirement that the lateral slip angle must not exceed 3 degrees. However, according to the tire model shown in Figure 7.19b, the maximum lateral tire force occurs at a much greater lateral slip angle (between 8 and 14 degrees depending on vertical loads). Therefore, this constraint will effectively limit the maximum lateral force achievable by the vehicle and therefore restrict the maneuvers that it can perform.



(a) Aligning moment

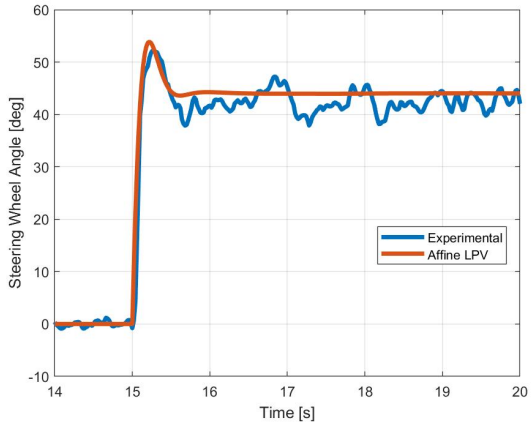
(b) Lateral force

Figure 7.19: CarSim 2020 Tire Model for 265/70 R17

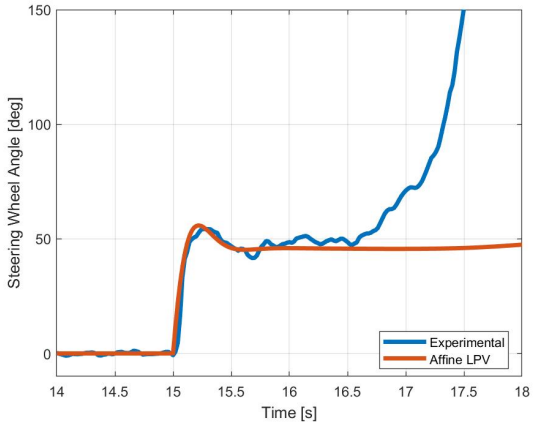
One way to show this destabilizing effect is to use a physics-based model and analyze the system's stability. A phase portrait would be a sufficient tool to show the region of attraction and where the system diverges. However, we previously assumed that there is not enough known about the physical system to develop such a physics-based model. Instead, we developed an LPV model using black-box techniques.

Therefore, to show this we will instead perform two simulation experiments. The first consists of a rack torque step input of 27 N-m when the vehicle is driving at 30 m/s. The magnitude of the step input is tuned to achieve a lateral slip angle just below 5 degrees. The CarSim-simulated response and the LPV affine model's prediction are shown in Figure 7.20a. The second experiment uses a larger step input of 28 N-m (tuned such that the lateral slip angle exceeds 5 degrees). The rack torque is just barely enough to exceed the aligning moment. It takes approximately 1 second after reaching a steady state for the lateral slip angle

to exceed 5 degrees and the resulting steering wheel angle begins to accelerate away from 50 degrees. The LPV affine model is unable to predict this phenomenon.



(a) CarSim's simulated (labeled "Experimental") and Affine LPV model's response to step input of 27 N-m



(b) CarSim's simulated (labeled "Experimental") and Affine LPV model's response to step input of 28 N-m

Figure 7.20: CarSim 2020 steering actuator step response when $U_x = 30$ m/s

The Torque Inverse control architecture relies on the forward model being accurate. However, at large slip angles, this model becomes increasingly inaccurate and the Torque Inverse controller is not robust against this form of model discrepancy. The cascade architecture is more robust against this as evidenced by the controller's ability to keep the lateral slip angles small. If this model discrepancy is considered a disturbance to the inner control loop, this result is somewhat unsurprising. This is because the cascade control architecture is well-known for its disturbance rejection capabilities [69], especially when that disturbance is in the inner control loop.

7.5 Experimental Results

The previous sections investigated modeling and control techniques using CarSim to eventually translate the best performing approaches to a 2019 Jeep Grand Cherokee. To summarize, because it was not known if the Jeep rack-and-pinion system used a motor on the column, the rack, or both, different rack-and-pinion configurations available in CarSim were studied and compared to the dynamics observed on the Jeep Grand Cherokee. After selecting the motor-on-rack configuration, the CarSim simulation was used to explore the validity of modeling the steering actuator as an LPV second-order system. However, a remaining question is whether it is valid to identify the dynamics of the steering system independent of the vehicle dynamics. This

will be addressed in Subsection 7.5.1 using experimentally collected data from the Jeep Grand Cherokee. After identifying the steering and vehicle model parameters, the cascade control architecture (developed in Subsection 7.3.3) is designed and experimentally tested on a closed-course test site. Test results in Subsection 7.5.2 show the tradeoff between tracking and comfort by investigating the balance between comfort and performance. Because comfort is valued more on highway lane-keeping maneuvers the high-performing controller from 7.5.2 is detuned and tested on public roads in rainy conditions. These experimental results are shown in Subsection 7.5.3.

7.5.1 Model Identification and Validation

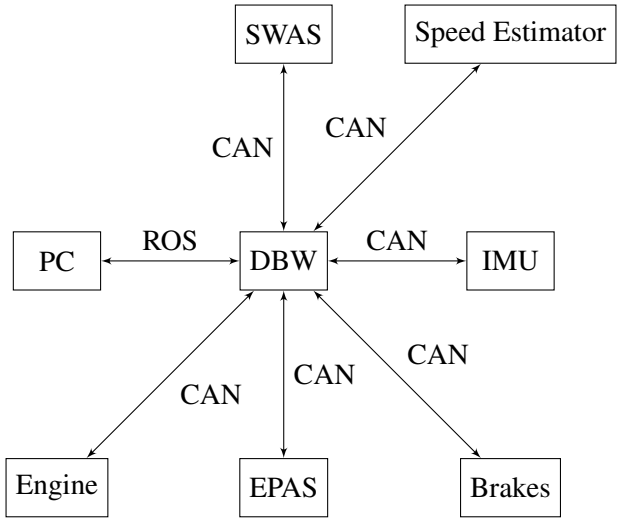


Figure 7.21: 2019 Jeep Grand Cherokee System Architecture

A major difference between the vehicle studied in CarSim and the 2019 Jeep Grand Cherokee is the system architecture. A high-level overview of the system architecture is depicted in Figure 7.21, where SWAS refers to a Steering Wheel Angle Sensor. The personal computer that is running the controllers written in Python 2 is called “PC". There is a drive-by-wire device (shown as “DBW" in Figure 7.21) developed by New Eagle [150] that provides a ROS interface to the PC. This device has a large number of parameters and software of its own. Much of which are unknown or have been pre-configured. The DBW device connects to the vehicle over the Controller Area Network (CAN). Downstream of the DBW, the exact architecture is unknown. It is guessed that there exists an IMU, a SWAS, a speed estimator (the output of which is shown on the vehicle dashboard), an engine control unit, a brakes control unit, and an EPAS control unit. With these available sensors and their signals, it is possible to identify the parameters of the steering actuator and vehicle bicycle

car models. However, it is unknown how these signals are processed before they are received. Furthermore, the communication delay throughout this system architecture is also unknown.

In ROS it is common practice to provide messages with timestamps (the ROS time at which the signal is sent). Because the systems downstream of the DBW device are not implemented in ROS they cannot provide such a timestamp. Instead, the DBW device provides a timestamp. Because of this, it is then possible to compare the current time with the timestamp of the messages received in the PC system. This will provide an estimate of the communication delay between the PC and DBW system. However, it is not possible to measure the communication delay between systems over the CAN with the current system architecture. This delay must be considered in the control design. Therefore, it will be estimated with the parameters of each model.

Identifying Bike Model Parameters

Several tests are performed on the retired Bryan Airforce Base runway (now called Texas A & M University Rellis Campus) in College Station, Texas. One set of tests conducted is called constant steering wheel angle tests. These are designed to allow accurate estimation of the vehicle's understeer gradient. The test procedure, as outlined in SAE J266, consists of the driver holding the steering wheel at a fixed angle while the vehicle is slowly accelerated until a desired lateral acceleration or yaw rate is reached. This test is conducted several times with different steering wheel angles. The path curvature traversed by the vehicle is computed as $\kappa = \frac{1}{R} = \frac{\dot{\psi}}{V_x}$. The results of each test are shown in Figure 7.22. The thick dashed lines are models estimated with linear regression to data ranging 0.1 and 0.5 g's of lateral acceleration. The bottom of this range is selected to eliminate transient accelerations at the start of some tests. The top of this range is to eliminate nonlinearities observed under high lateral accelerations. The gradient of these lines is then the slope of these linear models, which can be used to compute the understeer gradient as follows: $K = -L\frac{\Delta(\dot{\psi}/V_x)}{\Delta(a_y)}$ [52] (where Δ denotes the change in, and L is the vehicle's wheelbase).

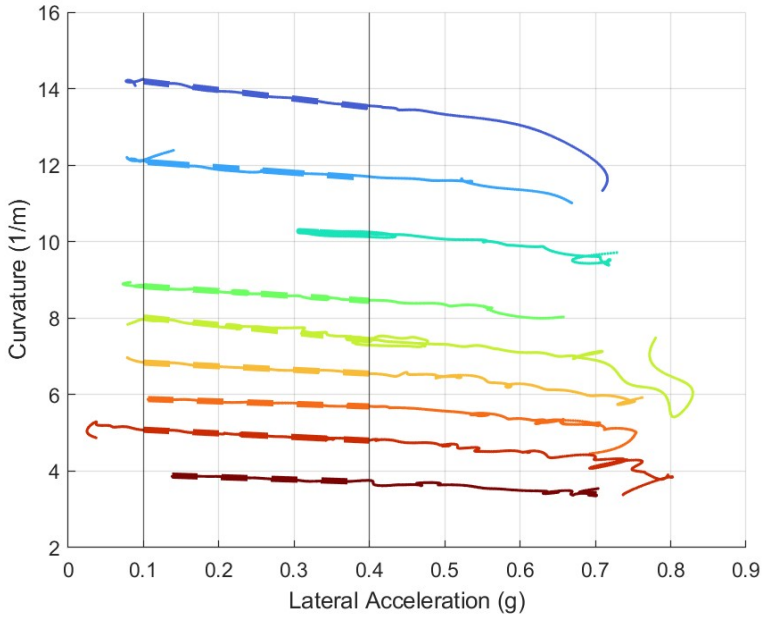


Figure 7.22: Jeep Experimental Results for Constant Steering Wheel Tests

Each test result is then used to compute an understeer gradient. Figure 7.23 shows the results of each test. The two most likely ranges (as determined by experiment counts falling within these ranges) are (1.5-2.0) and (2.5-3.0). The different results show that this test method is highly sensitive to uncertainties. SAE J266 mentions that this can occur “due to differences in road-load, throttle, aerodynamics, tire slip, and steer and inclination angles at different steering-wheel angles, etc." [52].

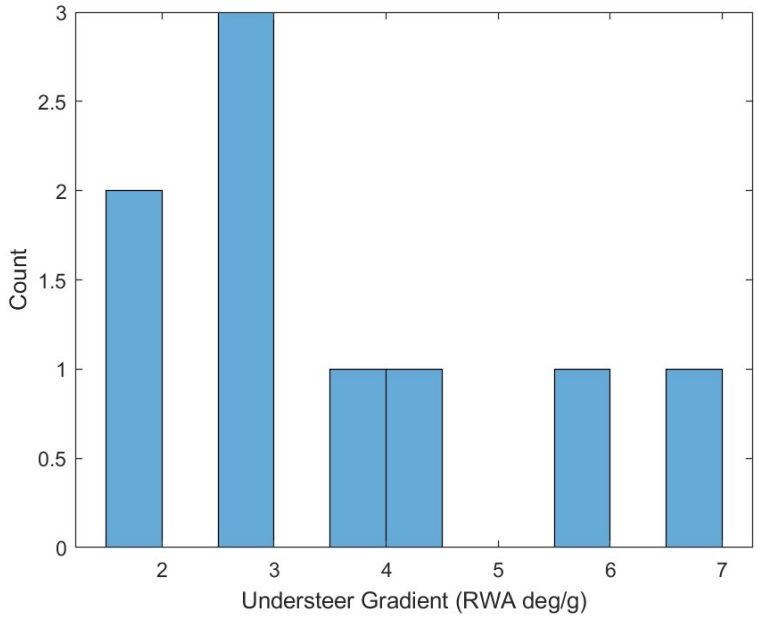


Figure 7.23: Histogram of Jeep Experimentally Measured Understeer Gradient

The next set of tests performed on the closed-course runway is chirp steer inputs which are used to excite frequencies between 0 and 4 Hz. To perform these a chirp signal that spans 0 to 4 Hz is input as a requested steering torque. The measured steering wheel angle, lateral acceleration, and yaw rate are recorded. In addition to these signals, it is known that there is a dependence on velocity, so a test is conducted at 5, 10, and 15 m/s. The longitudinal controller to do this did not achieve perfect tracking so the longitudinal velocity is also recorded and used as an additional input when estimating the model parameters (as opposed to assuming a fixed velocity).

In addition to the chirp tests, sine waves are input to the steering torque to collect more data. Table 7.1 provides a summary of the tests collected and used to identify the parameters.

Velocity (m/s)	Torque Amplitude (%)	Frequency (Hz)
25	10	0-4
15	15	0-4
5	40	0-4
5	15	2-4
5	15	0.5
5	15	1
15	15	1.5
15	15	1
15	15	0.5
15	15	0.1
25	15	1.5
25	15	1
25	15	0.5

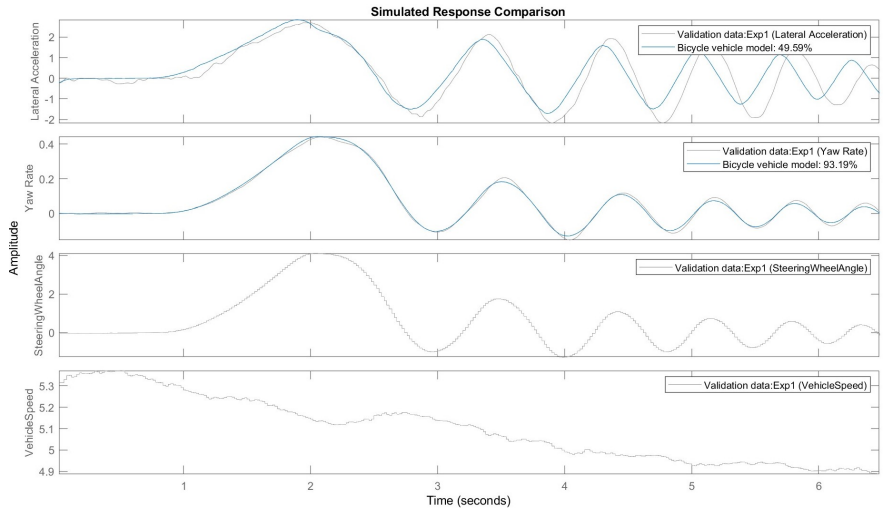
Table 7.1: Summary of Tests for Jeep Parameter Identification

Three (a, L, m) of the six bicycle car model parameters are obtained from National Highway Transportation Safety Administration (NHTSA) data [199]. A seventh model parameter is the steering ratio which maps the measured steering wheel angle to the road wheel angle. This is obtained from online specifications of the Jeep Grand Cherokee. It can also be estimated (if done so, it is approximately found to be close to online specifications). Table 7.2 presents the parameters that best fit the chirp data.

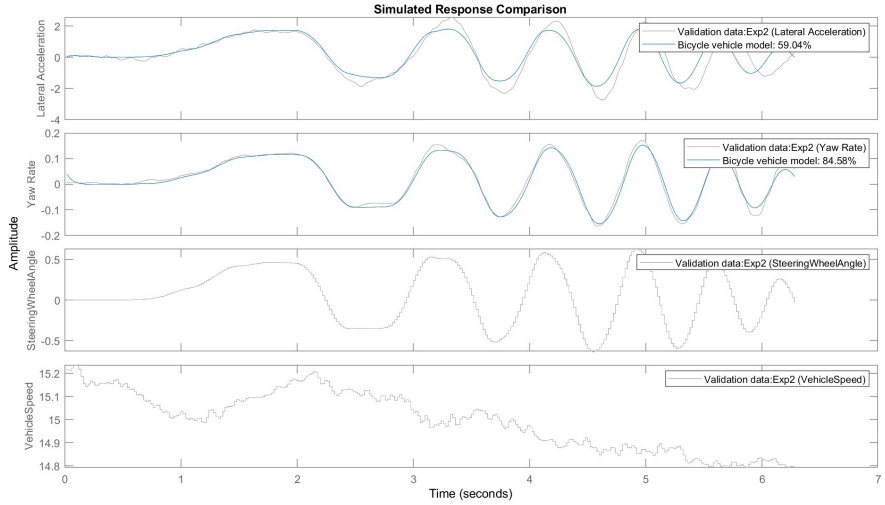
The fit quality is shown in Figure 7.24. Note that frequencies higher than 2 Hz are removed from the data because it was observed that some form of processing is done to the lateral acceleration that distorts these frequencies. It is suspected that a low pass filter is implemented in a low-level Jeep ECU that attempts to smooth the lateral acceleration measurements from the IMU. The evidence for this is the distortion at high frequencies and the phase lag observed in Figure 7.24 between the measured yaw rate and lateral acceleration. For this reason, more weight is placed on fitting the yaw rate instead of the lateral acceleration. As a verification of this model, the computed understeer gradient is 2.1 deg/g, which is corroborated by two constant steer angle tests as shown in Figures 7.23 and 7.22.

Parameter Name	Symbol	Value	Units
Mass	m	2300	kg
Front axle to CG	a	1.4025	m
Wheelbase	L	2.908	m
Front Cornering Stiffness	C_f	1.09×10^5	N/rad
Rear Cornering Stiffness	C_r	1.53×10^5	N/rad
Yaw rotational inertia	J	1,869	kg-m^2
Steering Ratio	K_s	16	1

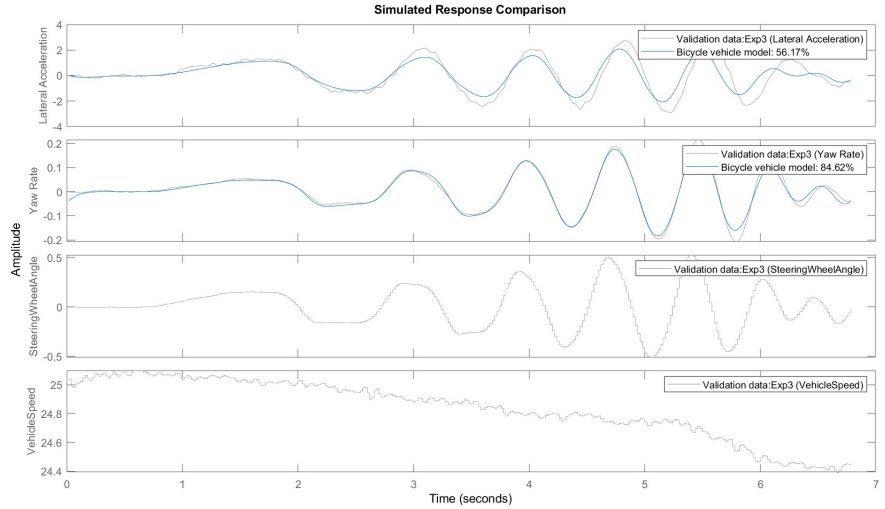
Table 7.2: 2019 Jeep Grand Cherokee Parameters



(a) Fit for 5 m/s



(b) Fit for 15 m/s



(c) Fit for 25 m/s

Figure 7.24: Bicycle car model fit quality for 2019 Jeep Grand Cherokee. Grey lines are experimental measurements and blue are model predictions.

In addition to identifying the parameters of the bicycle car model parameters, a delay is also identified. To do this, an iterative process is performed that consists of four steps:

1. Assume a delay as an integer multiple of the sample rate;
2. Remove the delay by shifting the inputs forward in time by the number of samples;
3. Identify the parameters that best fit the data;
4. Quantify the fit quality with the Normalized Root Mean Squared Error (NRMSE) of the Yaw rate averaged across the experiments.

The NRMSE averaged across all experiments is given by:

$$\frac{1}{N}\sum_{n=1}^{N}\text{NRMSE}_n = \frac{1}{N}\sum_{n=1}^{N}100\left(1-\frac{||y_n-\hat{y}_n||_2}{||y_n-\bar{y}_n||_2}\right)$$

where N is the number of experiments, y_n is the experimental data for experiment n, $\hat{y}_n$ is the model's output, and $\bar{y}_n$ is the mean of y_n. A value of 100 would indicate a perfect fit across all experiments. Furthermore, it should be noted that only Yaw Rate is used because of the previously mentioned unknown corruption of the lateral acceleration signal. Through this iterative process, it was found that if the input is shifted forward by 2 samples the averaged NRMSE is maximized at a value of 85.35. Since the sample rate is 50 Hz, this indicates a pure time delay of 0.04 seconds. This delay is measured as the difference between the timestamps of the steering wheel angle measurements and the timestamps of the IMU measurements. In other words, it is the delay between the Steering Wheel Sensor measurements and the IMU measurements. This is an important distinction to make because it is possible to use the reference steering wheel command instead of the measured. However, the measured steering wheel angle will not include any communication delay between the command and when the corresponding vehicle's system receives the command. The speed is assumed constant enough such that its delay (if it exists) has a negligible effect. The source of the 0.04-second delay can only be guessed at, but some good guesses might be:

1. Variable communication delay on the CAN
2. Jeep ECU processing time
3. DBW processing time

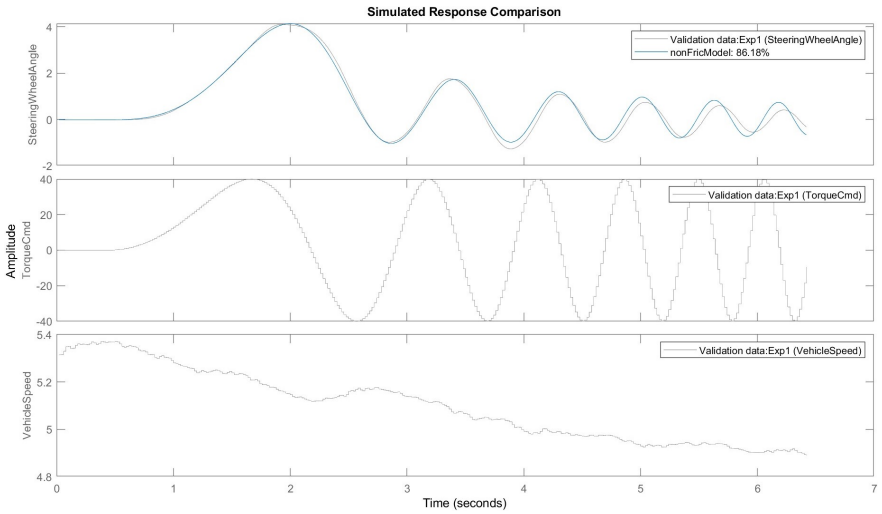
If the first guess accounts for most of the delay, then it is possible that this delay changes and can even become negative (the IMU measurements are received before the steering sensor measurements are received).

Identifying Steering Actator Model Parameters

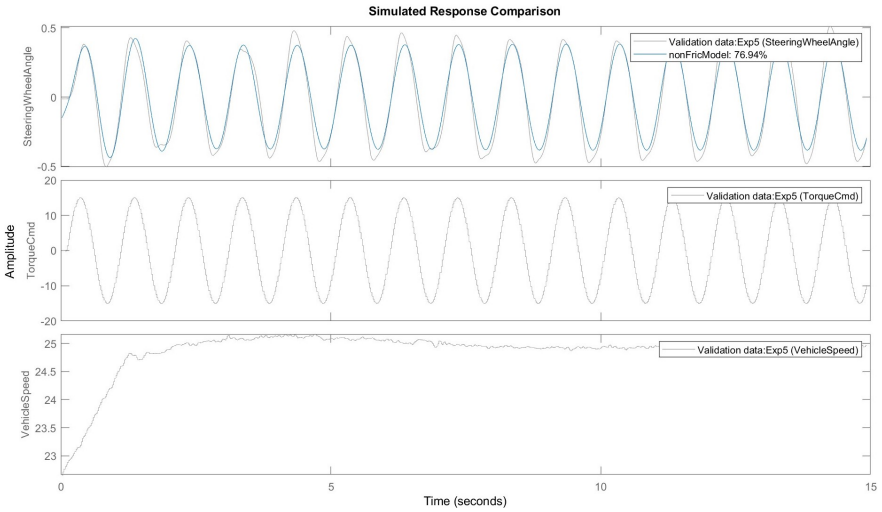
The same method previously described for the CarSim model is applied to the Jeep Grand Cherokee Model. The only modification made to the methodology is that the same technique for estimating delay previously described is also used with the Steering Actuator Model. It was found that a shift of 5 samples resulted in the best fit of the LPV affine model mapping steering wheel torque to steering wheel angle. Table 7.3 summarized the parameters that best fit the experimental data. The quality of fit is shown in Figure 7.25 using the time response data. The fit quality shown in the frequency domain (assuming frozen parameters) is shown in Figure 7.26.

Symbol	Value
α_1	-3.0913
α_2	0.4309
a_1	4.7437
a_2	-12.4578
γ_1	-0.0696
γ_2	-0.0059
c_1	2.3762
c_2	0.2155

Table 7.3: Jeep LPV Affine Steering Actuator Model Parameters



(a) Fit for 5 m/s



(b) Fit for 25 m/s

Figure 7.25: LPV affine steer model fit for 2019 Jeep Grand Cherokee. Grey lines are experimental measurements and blue are model predictions.

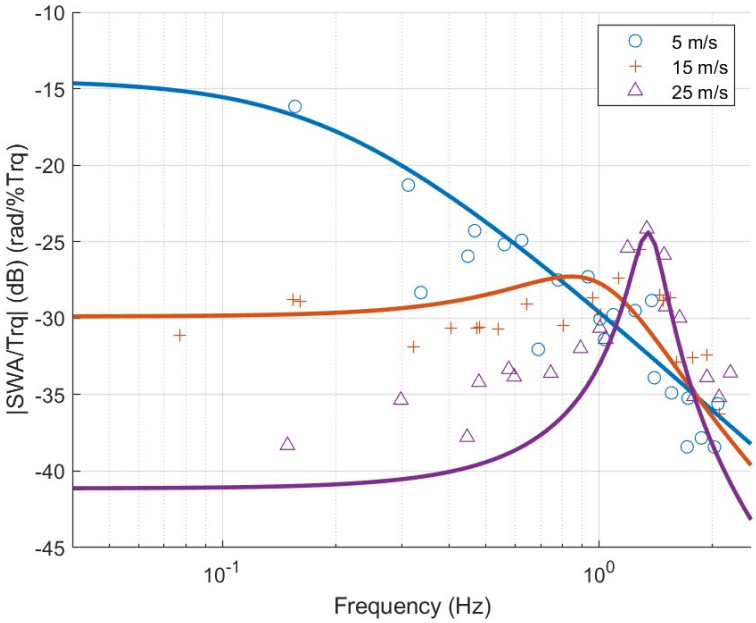
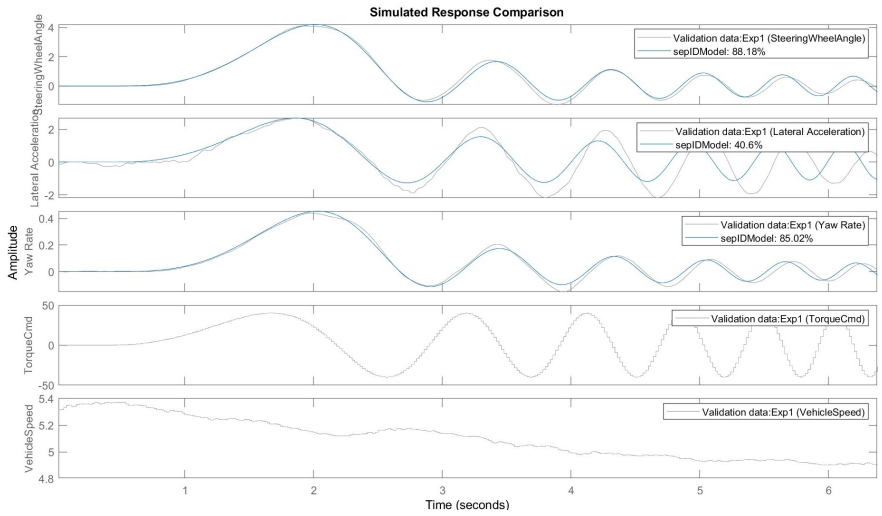


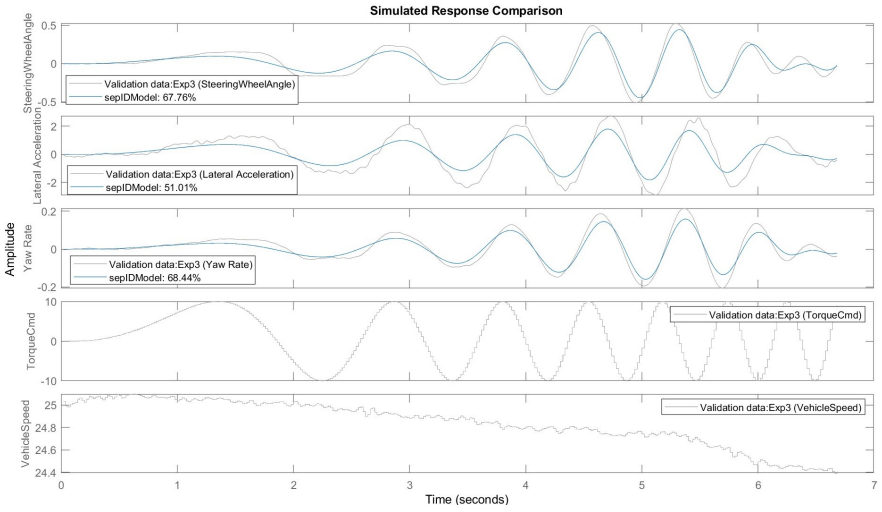
Figure 7.26: Frequency Response for LPV affine steer model fit for 2019 Jeep Grand Cherokee. Lines are the frequency response of the model at fixed velocity and the points are experimental data.

Combining Steer and Bike Models

Having identified the parameters for the steering actuator and bike models, we can now combine them by placing them in series. The output of the steering actuator model is the input of the bicycle car model. The bicycle car model previously identified that there is a delay of 2 samples (0.04 seconds) between the steering wheel sensor and the IMU sensor. The steering actuator identified a delay of 5 samples (0.1 seconds) between the PC and the steering wheel sensor. Combining these, there is a total delay of 7 samples (0.14 seconds) between the PC and the IMU sensor. The fit quality can be visually assessed in Figure 7.27 for two different experiments. The fit for all model outputs (predicted steering wheel angle, lateral acceleration, and yaw rate) are quantified in Figure 7.28. The first three experiments are the chirp experiments. The remaining are sine input experiments. All experiments are summarized in Table 7.1.



(a) Fit for 5 m/s



(b) Fit for 25 m/s

Figure 7.27: Combined steer and bicycle car model fit quality when models are independently estimated. Grey lines are experimental measurements and blue are model predictions.

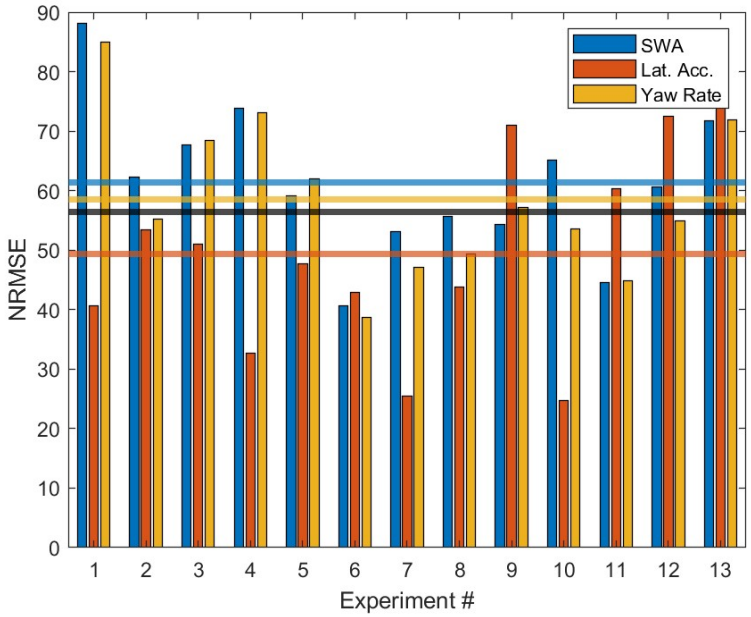


Figure 7.28: Bar chart of combined steer and bicycle car model fit quality when models are independently estimated. The horizontal lines show the average of the NRMSE over all experiments for the signal with matching color. The black horizontal line is the average over all experiments and signals. SWA stands for Steering Wheel Angle.

Overall, Figures 7.27 and 7.28 show that the steering actuator model does a good job of predicting the trends of the steering actuator (assessed by the SWA NRMSE). Some experiments show a better fit of the lateral acceleration signals than others. Most experiments show that the steering actuator model fits the yaw rate better than the lateral acceleration. The lower-quality fits of the lateral acceleration are likely due to the unknown processing done to the signal before the PC records it.

One choice made early on in this system identification process was that it is better to identify the steering actuator model and bike model parameters separately than to identify both model parameter sets jointly. There are several reasons for this choice:

1. If the steering actuator model is not very accurate, then the inputs to the bicycle car model (the predicted outputs of the steering actuator model) will not be very accurate. This then results in performing parameter identification of the bicycle car parameters with an inaccurate input signal.

2. The system is a network with unknown delays between each system. Independently identifying the models enables easier delay identification. For example, using the iterative process previously described, the number of samples to adjust the data is increased for each model independently until the

fit quality (measured by averaged NRMSE) stops increasing and decreases. For the steering actuator model, iterations stopped after 6 steps (since 5 steps was the maximum). For the bike model, iterations stopped after 3 steps (since 2 steps was the maximum). If this were performed jointly, these delays would have to be iterated in a combinatorial fashion because the effect of each assumed delay on the fit quality cannot be independently determined. For instance, the process would be to start with a fixed sample delay of 0 for the bike model and increase the delay for the steering actuator model. Then repeat this for a delay of 1 sample in the bike model. Then repeat for a delay of 2 samples in the bike model. This would continue until a grid of delays is evaluated and then the best-fit model would be selected.

The logic behind this choice can be tested by jointly estimating both steering and bike model parameters and comparing the fit quality to the independently estimated model parameters. The fit results are summarized in Figure 7.29. Comparing Figure 7.28 to Figure 7.29, it appears that when the models are jointly estimated a better quality fit can be found (as judged by the higher horizontal lines indicating the averaged NRMSE for SWA and yaw rate). Because of the previously mentioned combinatorial issue of estimating delay for the different communication channels, it is assumed that both the SWA and IMU sensors have the same communication time delay. The iterative procedure employed earlier then arrives at a value of 8 samples (160 ms) of pure time delay. This is similar to 7 total samples (140 ms) of time delay found in the independently estimated models. This indicates a general agreement of significant pure time delay in the system architecture.

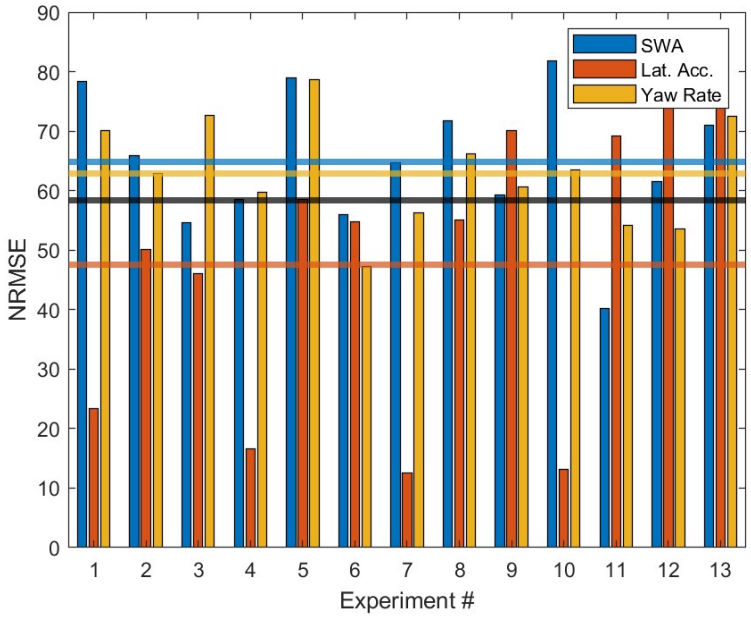


Figure 7.29: Bar chart of combined steer and bicycle car model fit quality when models are jointly estimated. The horizontal lines show the average of the NRMSE over all experiments for the signal with matching color. The black horizontal line is the average over all experiments and signals. SWA stands for Steering Wheel Angle.

A comparison of the model parameters when each model is separately identified and when they are jointly identified is presented in Table 7.4.

Symbol	**Separate Value**	**Joint Value**
m	2300	2300
a	1.4025	1.4025
L	2.908	2.908
C_f	1.09×10^5	1.01×10^5
C_r	1.53×10^5	1.42×10^5
J	1,869	2,260
K_s	16	16
α_1	-3.0913	-3.4336
α_2	0.4309	0.5062
a_1	4.7437	6.2685
a_2	-12.4578	-15.4283
γ_1	-0.0696	-0.0496
γ_2	-0.0059	-0.01376
c_1	2.3762	2.1625
c_2	0.2155	0.4640

Table 7.4: Parameters of steering actuator and bicycle car models when jointly and separately estimated.

Because the data is better fit by the Jointly estimated parameters that model will be used for control design. The reasoning behind the separately identified models still stands but there are perhaps better reasons for jointly estimating the parameters:

1. While the principle of linearity is applied to separate the steering actuator model from the bicycle car model, the fact that both models are parameter-varying may violate this principle. Therefore, separating the two models may not be appropriate.

2. The additional steering wheel angle measurement used to identify the joint model parameters likely contributes significantly to the model fit. This follows from the assumption that more sensors will result in better parameter estimation. Further research is warranted to investigate this.

3. The underlying dynamics of the steering actuator are heavily dependent on tire dynamics (as is the bicycle car model). Tire dynamics are deeply connected to the accelerations and rotational velocities of the vehicle body. Therefore, by jointly estimating the parameters these nonlinearities can be better approximated with the parameter-varying model.

The last point is rather important to emphasize. The relationship between the steering actuator and the vehicle has been studied in [138] in which bond graphs are used to develop a model between a large number of vehicle actuators and sensors. The proposed model of the steering actuator is primarily focused on electric power assist, so it includes human inputs with electric motor inputs. An additional input to the steering actuator is the lateral force, which is converted to a torque on the steering system by the mechanical trail. This provides intuition on how lateral force is related to the steering torque, and lateral force is connected to the vehicle chassis model. However, this model includes steering actuator electronics and gear ratios, which are not necessary for steering control design when the interface takes a commanded motor torque instead of voltages or currents. What future research could focus on is connecting an electric power steering (without human inputs) to the bicycle car model. This is a white-box version of the model identified in this Chapter and with a nonlinear tire model, can be used to study the various observations covered in this chapter more theoretically.

Delay Identification

The previous sections jointly estimated pure time delay and the model parameters. The procedure was an iterative process under which an assumed pure time delay is removed from the data and the fit quality is

used to assess the overall model parameters. This methodology is dependent on a wide number of variables such as those of the models, the optimization solver settings (Levenberg-Marquardt), and the method used to quantify the fit. In this subsection, the delay identified (0.160 seconds from the joint model and 0.120 seconds from the independently-identified model) is estimated using a different methodology and different data.

The experiments used to identify the steering and vehicle dynamics were mostly based on chirp signals that linearly swept between 0 and 4 Hz. This experimental design is recommended by industry standards [111] (in which it is called the random steer tests). However, when estimating delay with this data, a step input may be preferable since the initial response of the dynamic system is more easily discernable. Unfortunately, when conducting this step input data the IMU measurements were not recorded so they cannot be used to identify bicycle car model parameters. However, what can be done is to estimate the delay between the torque command (sent from the PC) and the measured steering wheel angle (sent from the steering wheel angle sensor over CAN). Like previously, this is a delay between the timestamps of each message. Later, the delay that can be measured online will be analyzed.

The method used to estimate the pure time delay between the torque and steering wheel angle is based on a voting system. Three different model structures will be used to identify a family of models from the data. The family of models will consist of different orders and different assumed pure time delays. The model that best fits the Jeep's experimentally-collected data will then be selected from the family of models. Models without explicit delay parameters will be used to predict the impulse response. The time it takes for the system to reach more than 3 times the output's standard deviation will be used as the pure time delay. The three model structures are:

1. Autoregressive and Exogenous (ARX) model. (Matlab's arxstruc function)
2. Finite Impulse Response (FIR) model (also known as impulse response model). (Matlab's impulseest function)
3. State Space (SS) model. (Matlab's n4sid function).

Each of these three models can be identified from data using the associated Matlab commands and the uncertainties are retained in the Matlab objects. When predicting the impulse response the standard deviation of the output can then be computed.

The methodology is as follows:

1. For each experiment

   (a) Select ARX model that minimizes Akaike's Information Criterion (AIC) [133] for a family of models having input orders between 2 and 5, output orders between 2 and 5, and delay samples between 0 and 20. (arxstruc and selstruc)

   (b) Identify FIR model. (impulseest)

   (c) Predict FIR impulse response and identify the time it takes for the response to exceed 3 times the output's standard deviation. (impulse)

   (d) Identify 10th order SS model (10th order is chosen as practical maximum of the model order to be identified) and use Hankel Singular Values to select the best balance between fit and model complexity (as quantified by number of states). Note that this selection is automated with Matlab's n4sid function.

   (e) Predict SS impulse response and identify the time it takes for the response to exceed 3 times the output's standard deviation. (impulse)

2. Consider each model's estimated delay for each experiment as a single "vote" and select the most voted delay as the most likely value of pure time delay.

The results are presented in Figures 7.30 and 7.31. Figure 7.30 shows that each experiment can be estimated as having different amounts of delay. The ARX model predicts the most delay variation across all experiments. The modes for all three models agree that 0.100 seconds is the most likely delay (the SS model results are tied between 0.1 and 0.08 seconds). Figure 7.31 presents the same data in a way that makes it more clear that the most likely delay is 0.100 seconds.

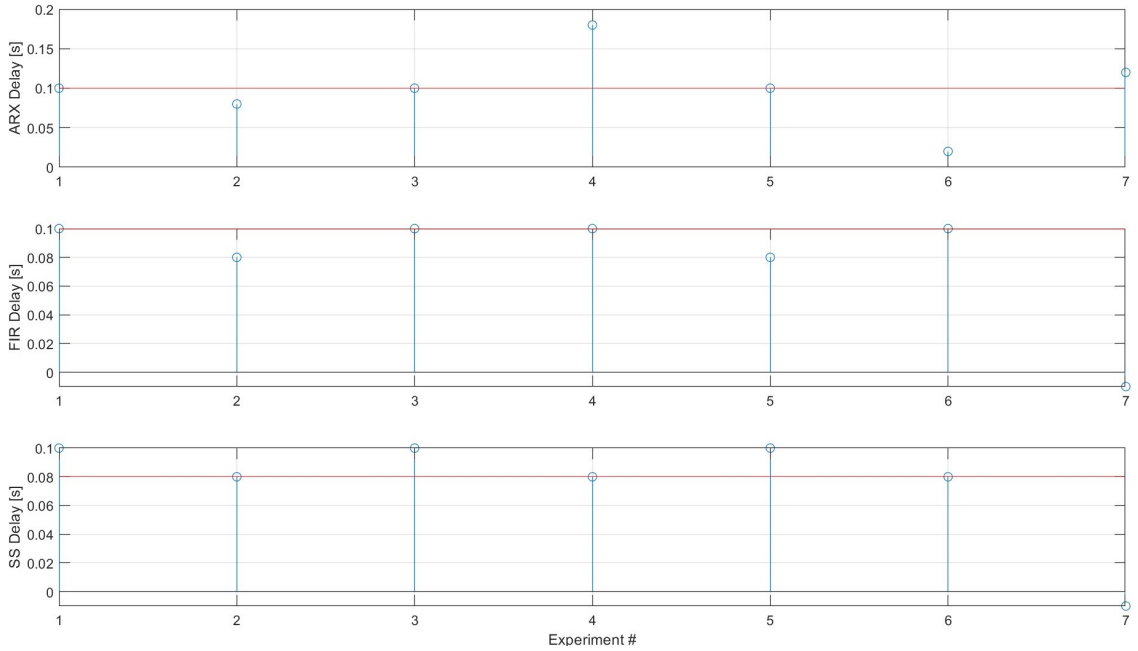


Figure 7.30: Delay estimations from each model identified with each experiment on the Jeep Grand Cherokee between normalized rack torque and steering wheel angle measurement timestamps. The red horizontal lines indicate the mode of each model across all experiments. Negative values (in the experiment 7 results for the FIR and SS models) indicate when impulse responses never exceed the response's uncertainty (which indicates that the results are not reliable)

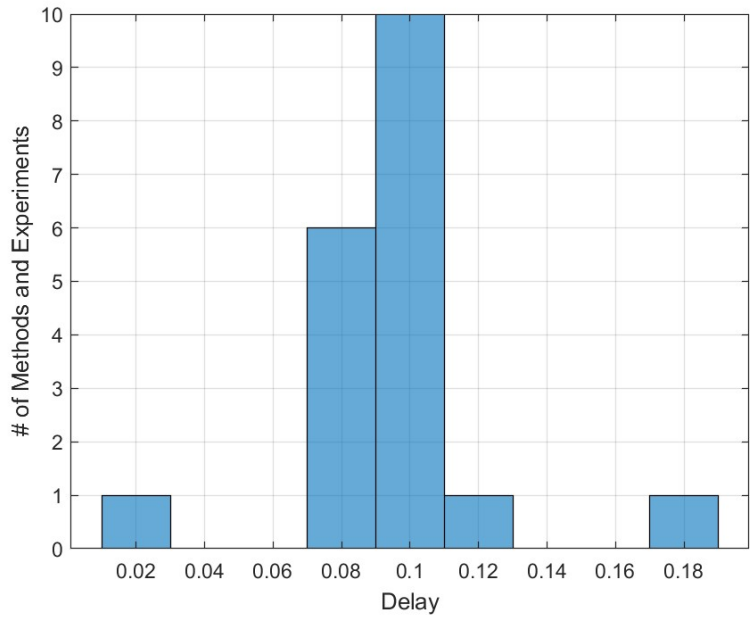


Figure 7.31: Histogram of all delay estimations from each model identified with each experiment on the Jeep Grand Cherokee between steering torque and steering wheel angle measurement timestamps.

This pure time delay estimation differs from the delay results of the previous subsubsections. However, these results are trusted more than the iterative procedure so the jointly estimated steering and bicycle car models are estimated to account for a delay of 0.100 seconds. The new parameters are compared to the parameters estimated when the delay is 0.16 seconds in Table 7.5.

Symbol	Delay = 0.16 Value	Delay = 0.1 Value
m	2300	2300
a	1.4025	1.4025
L	2.908	2.908
C_f	1.01×10^5	9.82×10^4
C_r	1.42×10^5	1.44×10^5
J	2,260	2,453
K_s	16	16
α_1	-3.4336	-3.5858
α_2	0.5062	0.3591
a_1	6.2685	8.9837
a_2	-15.4283	-11.3181
γ_1	-0.0496	-0.0225
γ_2	-0.01376	-0.0092
c_1	2.1625	1.7480
c_2	0.4640	0.2876

Table 7.5: Parameters of steering and bicycle car models when jointly estimated for different amounts of pure time delay.

Steering Control Design

An LPV H-infinity steering controller is designed using the same procedure performed in Section 7.3.3. A pade approximation of the 0.1 seconds of pure time delay is put in series with the plant. The resulting frequency response of closed-loop transfer functions is shown in Figure 7.32. By comparing the design shown in Figure 7.32 with the design shown in Figure 7.13, the influence of the delay is visible. The steering controller designed for the Jeep has more variation in its closed-loop frequency responses as velocity increases. The bandwidth of the complementary sensitivity transfer function when the vehicle velocity is 5 m/s is approximately 2 Hz. When the velocity is 25 m/s, the bandwidth is approximately 1 Hz. This lower bandwidth is necessary to lower the larger peak in the sensitivity transfer function across all velocities (a result of the non-minimum phase zero introduced by the Pade approximation of the delay).

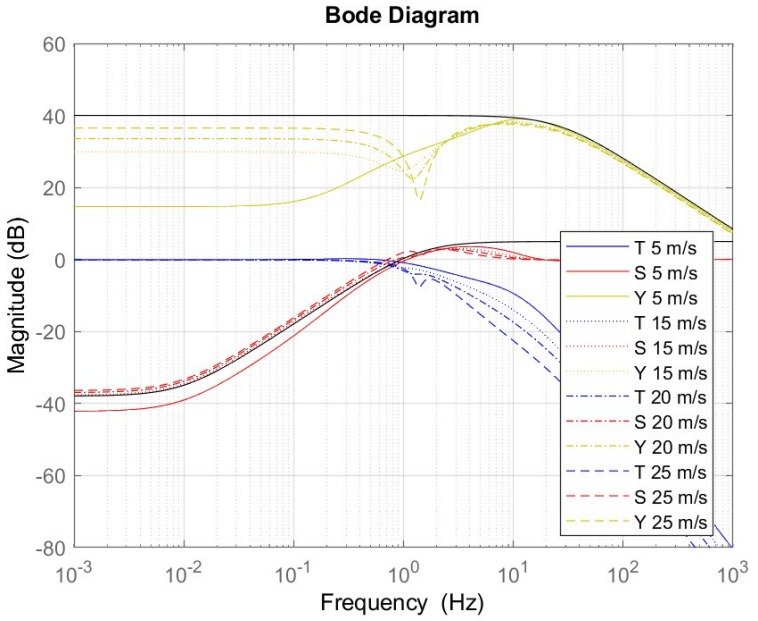


Figure 7.32: Magnitudes of the closed-loop transfer function frequency responses for the LPV H-infinity controller at different velocities.

Because the steering controller has a bandwidth between 2 and 1 Hz as velocity increases, the outer loop controller (lateral motion controller) needs to be much lower in bandwidth than 1 Hz. If the lateral motion controller's bandwidth becomes too close to the bandwidth of the steering controller, oscillations may be induced. These oscillations cause passenger discomfort. This presents a challenging tradeoff between performance (as determined by bandwidth) and comfort. A feedforward and feedback lateral motion controller is designed to achieve an acceptable balance. The feedforward is given as $\frac{1+KV^2/L}{V/L}\kappa V$, where κ is the reference path's curvature, L is the vehicle's wheelbase, K is the vehicle's understeer gradient, and κV is the reference path's yaw rate. This expression is the inverse of the steady-state gain from road wheel angle to yaw rate [75]. The feedback controller is computed by solving the Interpolation Conditions for the bicycle car model when the velocity is 25 m/s. Figure 7.33 shows the frequency responses of the closed-loop transfer functions. The response shows that the complementary sensitivity transfer function T has a bandwidth of 0.35 Hz. At this low frequency, human passengers do not perceive the oscillations caused by the interaction between the steering and lateral controllers. However, the joint feedback and feedforward controllers track highway-like paths with less than 30 cm of lateral position error.

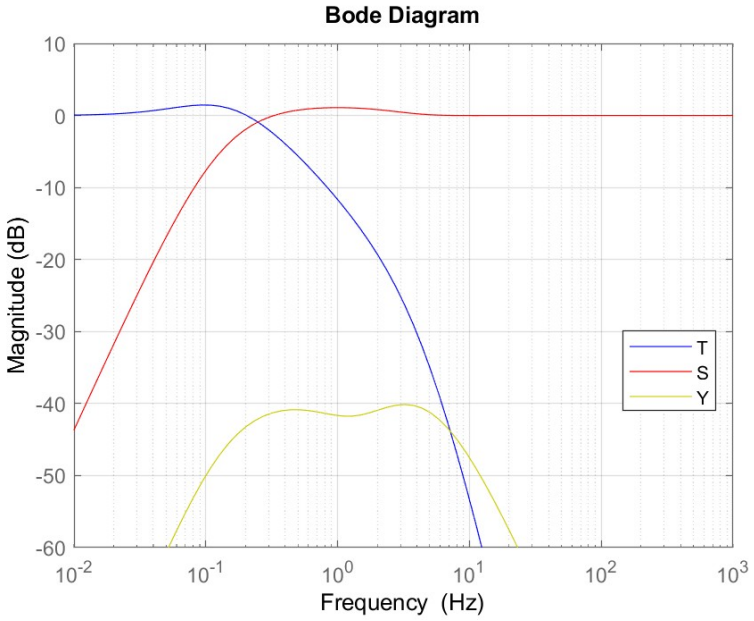


Figure 7.33: Magnitude of the frequency response for the lateral motion controller's closed-loop transfer functions when the longitudinal velocity is 25 m/s.

7.5.2 Closed Course Testing

The closed-course testing is performed at the previously mentioned Rellis test location. This is a concrete runway large enough to perform highway speed maneuvers. However, there is not enough space to perform highway speed cornering maneuvers for long durations. So a slalom or sine wave maneuver is performed. The experimental results are shown in Figures 7.34 and 7.35. Figure 7.34 shows a top-down or a birds-eye view of the maneuver. Note that it is slightly asymmetrical, which causes the asymmetry in the lateral error shown in Figure 7.35. Figure 7.35 shows various signals plotted against time. The longitudinal velocity accelerates to 30 m/s (5 m/s greater than the design velocity). The vehicle enters the first curve just after 10 seconds. Before then, there is a small lateral error from the controller's response to the 0.75-degree initial yaw error. The first curve generates a lateral acceleration of about 1 m/s^2 at about 20 m/s. After the peak of this curve, the lateral error grows to about 10 cm. The largest lateral error occurs during the peak of the third curve (between 20 and 25 seconds). This curve generates just under 4 m/s^2. Overall, this tracking perform is deemed comfortable and sufficiently safe for highway-like driving.

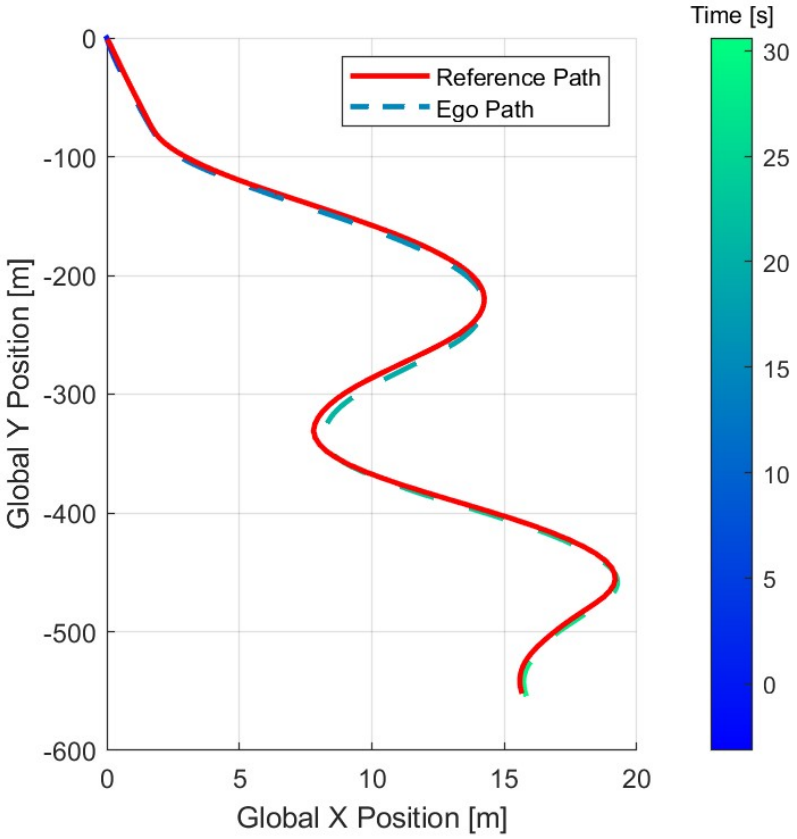


Figure 7.34: Top down view of the slalom maneuver performed at Rellis. The dashed line's color indicates the time throughout the maneuver.

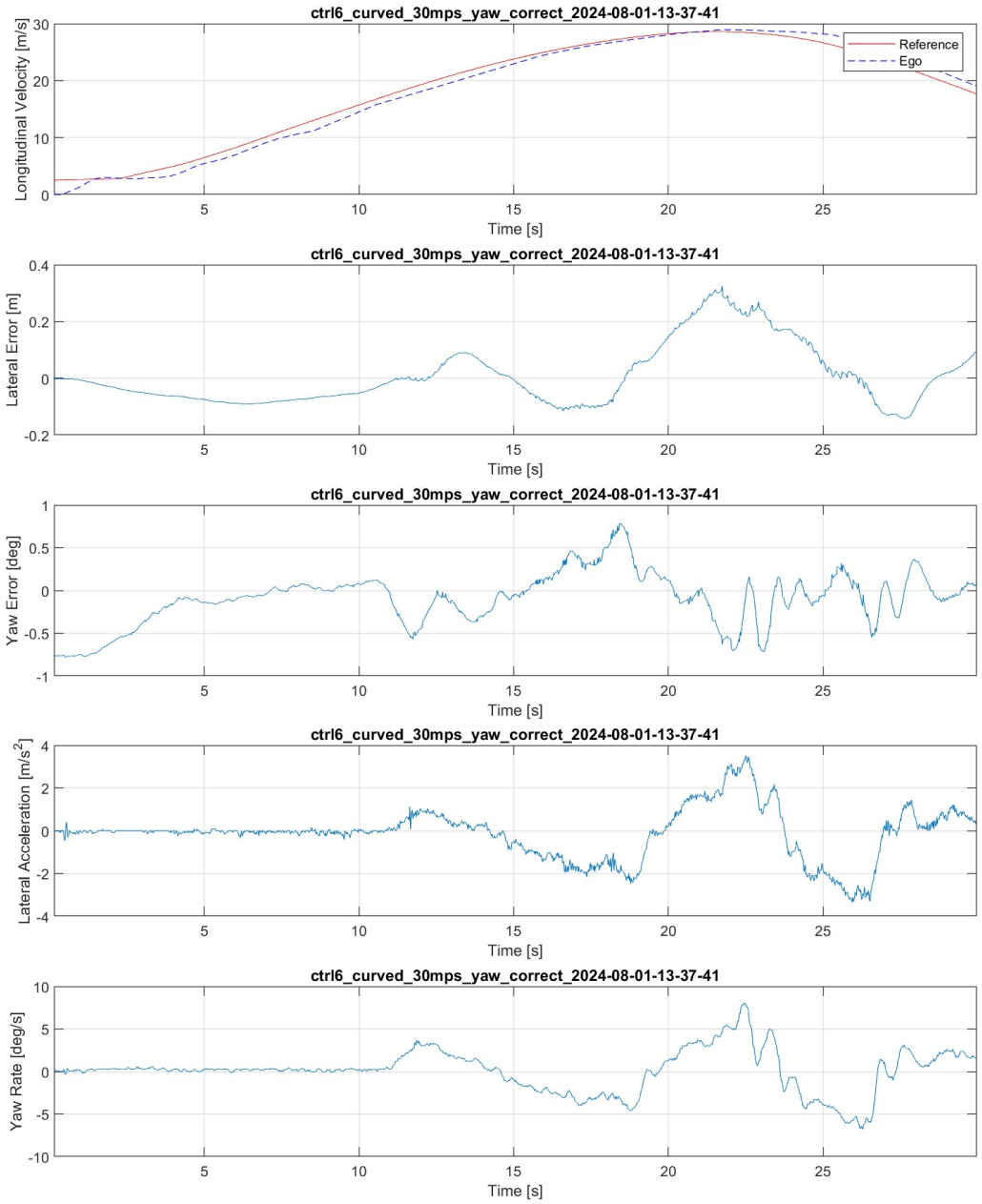


Figure 7.35: Timeseries of experimental testing for slalom maneuver performed at Rellis.

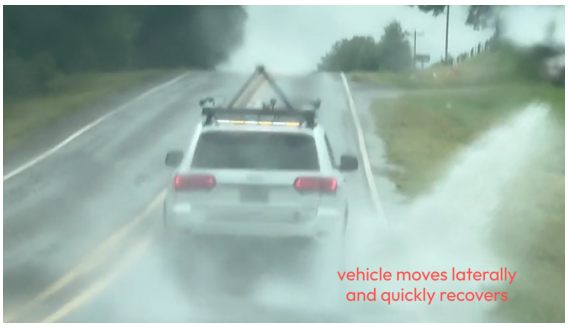
7.5.3 Public Road Testing

With successful results from closed-course testing, the next round of tests to evaluate the controller is performed on public roads. A multi-layered safety approach is taken to reduce the increased risk assumed on public roads. The approach consists of implementing a series of safety procedures and software checks. Each safety procedure and software check is considered as a layer that will catch potential failures. It is

assumed that each layer cannot catch all possible failures. However, after stacking enough layers, there is a low probability of a failure getting through all layers. This is the so-called “swiss-cheese" model. The layers are as follows:

1. Restrict autonomous public-road operations so that no vehicle is ever in the lane adjacent to the ego vehicle (the autonomous vehicle).
2. A trained, professional driver will monitor the ego vehicle in the driver’s seat.
3. Check the age (as defined by the difference between when the message was timestamped and when the message is read by the controller) of each message that the controller receives. If the age exceeds some threshold, disable autonomy.
4. Using the methodology presented in Chapter 2 monitor the computed longitudinal, lateral, and yaw error. Using pre-designed alert limits, assume that the yaw protection level is exactly the current yaw error and compute the resulting longitudinal and lateral protection levels. If the longitudinal or lateral errors exceed these protection levels, disable autonomy.
5. Verify that the path provided to the vehicle is feasible by using simple kinematic bicycle car model equations.
6. Verify that the computed control actions (steering wheel angle and steering torque) are within the vehicle’s linear regime (do not cause excessive lateral accelerations or yaw rates).

If any of the layers catch a failure, the autonomous system is disabled. Three days of public road testing were conducted at the end of July 2024 and no failure got past all layers and caused either a near miss or a collision. During these three days, a storm blew into the area and caused heavy rain during some of the public road testing. During testing, there were some large puddles on the route. Figure 7.36 shows two photographs taken from dashcams during public road testing. In the moment captured here, the vehicle drifts slightly to the left and the lateral motion controller quickly corrects.



(a) Exterior view of the Jeep driving through a water puddle on a public road.

(b) Interior view of the Jeep driving just before entering a water puddle on a public road. The arrow indicates the location of the puddle.

Figure 7.36: Dashcam photos of the Jeep during public road testing in rainy conditions.

Figure 7.37 shows a top-down view of the public road route. The gaps indicate where the vehicle was disabled as part of the previously described set of safety layers. Figure 7.38 shows the velocity profile for the references and the measured velocity. There are a total of six interruptions to the full route because of the safety layers. For the majority of the route the velocity is constant at 15 m/s (approximately 35 mph) except for the last segment when the vehicle operates at 20 m/s (approximately 45 mph). This reduced speed is result of a combination of considerations.

1. Due to heavy rainfall, safe highway operation is conducted at lower speeds. This reduced speed is similar to the speed modifications made to the maneuvers in Chapter 6.

2. The vehicle's perception sensors (lidar, camera, and radar) do not provide sufficient accuracy at large distances. Therefore, the vision system cannot detect obstacles or other road users with enough reliability and enough distance to provide the vehicle with enough time to safely respond. This limits the maximum speed that the vehicle can safely operate.

3. The heavy rainfall further deteriorates the quality and field of view of the perception sensors. This further restricts the maximum velocity that can be safely performed on public roads.

The lateral errors and yaw errors are shown in Figure 7.39. There is bias in both. This is the result of a slightly incorrect calibration of the relative orientation between the GPS and IMU sensors. The reason this bias appears in both lateral and yaw errors is that the error that the controller attempts to bring to zero is a linear combination of lateral error and yaw error: $(e_1 + d_s e_2)$ where d_s is a look-ahead distance. However,

since the planner that provides the controller with a reference path cannot provide a preview of sufficient look-ahead distance, the controller is implemented as a look-down controller as explained in Chapter 4.

Addressing this calibration issue is simple because the steady-state bias in the yaw error is exactly the sensor bias. A simple straight line maneuver is sufficient to reveal the magnitude of the calibration misalignment, which can then be used to modify the calibration parameters. The results presented in Figures 7.35 do not display this offset because these results are obtained after the public road testing and the calibration offset was addressed.

For the public road testing, the lateral error never exceeded 50 cm (a handful of samples that are not displayed in Figure 7.39) or the protection levels. Figure 7.39 shows that the vast majority of the lateral error remains within 10 cm of the bias (which is approximately -6 cm). More specifically, the root mean square values for lateral error and yaw error during this run are 9 cm and .45 degrees, respectively. The standard deviations for the lateral error and yaw error are 8 cm and 0.31 degrees, respectively.

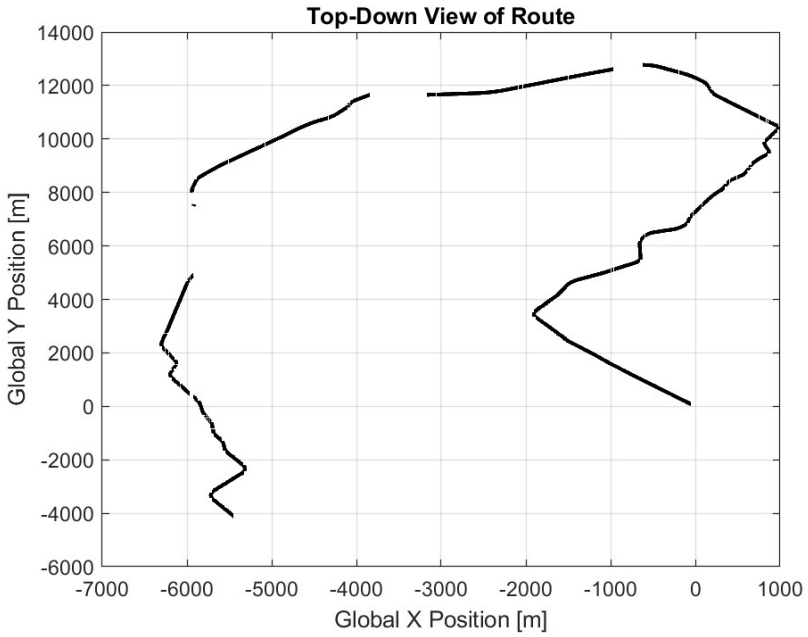


Figure 7.37: Top down view of the public road maneuver performed on 07/25/2024. The gaps in the path indicate when the autonomous system was disabled.

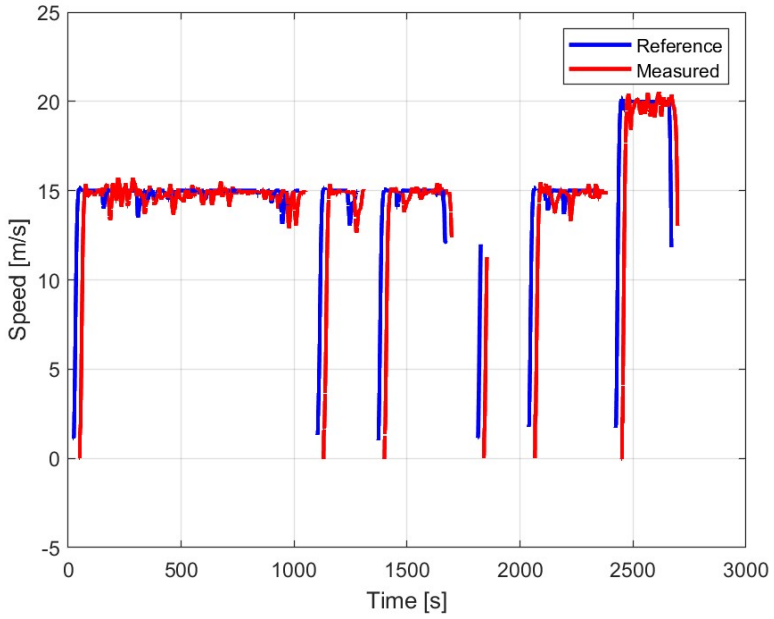
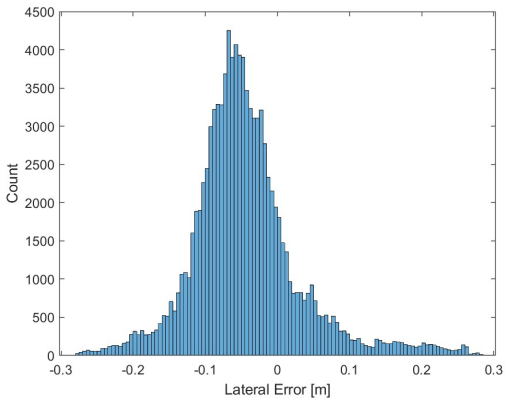
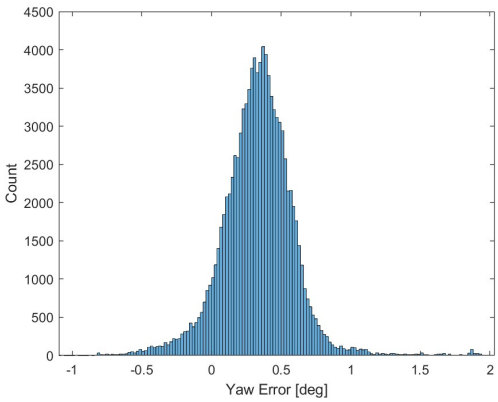


Figure 7.38: Velocity profile of the public road testing during rainy conditions on 07/25/2024.



(a) Histogram of lateral errors during public road testing in rainy conditions.



(b) Histogram of yaw errors during public road testing in rainy conditions.

Figure 7.39: Histogram of control errors recorded throughout the first run on public roads during rainy conditions on 07/25/2024.

7.6 Conclusion

This chapter documents the solution to safe, comfortable path tracking on rural highways. At a high level, this Chapter proposes using a high-fidelity simulation to study and explore different modeling and control solutions. However, while such a strategy is typical in computer-aided engineering, constructing a high-

fidelity virtual vehicle of the Jeep is prohibitively difficult. One solution is to use a lower-fidelity vehicle model to predict the real vehicle's response. However, the Jeep system architecture is unknown since it is proprietary. So instead, a vehicle model similar to the Jeep is selected in CarSim and studied as a proxy for the Jeep. This significantly reduces the required amounts of physical testing on the Jeep. The underlying assumption is that the results obtained for the CarSim vehicle are translatable to the Jeep Grand Cherokee. The successful results shown on closed-course and public road testing validate this assumption.

One challenge with using the CarSim vehicle as a proxy for the Jeep is that the Jeep's control interface exposes a steering torque and not a steering wheel angle as the signal to command. This is easily addressed by reconfiguring CarSim to use a steering torque command. However, there are several configurations for the power-assisted rack-and-pinion and the exact configuration for the Jeep is unknown. Ultimately, the distinction between each configuration is shown to be inconsequential since they all exhibit similar frequency-domain shapes. The major difference is in the DC gain for the CarSim steering system. This difference is also unimportant because the Jeep commands a normalized steering torque, and the denominator is not known. This normalization greatly changes the DC gain of the frequency response of the Jeep steering system, but it does not change the overall shape, which closely matches the CarSim actuator's shape. In particular, both vehicles (CarSim SUV and Jeep) have a resonant peak just above 1 Hz that increases in magnitude with increasing vehicle velocity. Both vehicles have a DC gain that decreases with increasing velocity. Therefore, it is concluded that the CarSim model is reasonably representative of the Jeep and is suitable for studying the problem of controlling the Jeep.

Because the Jeep steering architecture is unknown, a more data-driven steering actuator model is well-motivated. Furthermore, there are well-known LPV control techniques that require a polytopic plant model. Combining these two motivations, a unique LPV state space model is proposed that models the CarSim SUV and Jeep steering systems well enough to enable performant steering control design. The LPV control technique is used because the large variations of the steering actuator's dynamics with longitudinal velocity can be well compensated for with gain scheduling. One shortcoming of the LPV affine steering actuator model is that it does not correctly capture the steering response when there is aggressive braking. The steering actuator model developed in [138] provides some insight into steering actuator control design because it connects the steering actuator model to the rest of the vehicle through the lateral tire forces. Therefore, through the tire model, the vehicle longitudinal velocity (and other dynamics) are incorporated into the steering system. With this understanding, the LPV affine steering actuator model that is identified in this

chapter indirectly identifies the tire model. Using a simplified tire model: $F_y = C_f\alpha$, where α is the tire's sideslip angle, the steering system can be assumed to depend on the tire's sideslip angle. Since the sideslip angle is a function of lateral velocity, longitudinal velocity, and yaw rate, there is good reason to propose gain scheduling on lateral velocity and yaw rate in addition to longitudinal velocity. However, scheduling on lateral velocity may be challenging in practice since lateral velocity is not as easily measured or estimated as longitudinal velocity.

Furthermore, a tire's lateral forces are dependent on the normal force, which differs from the static values when the vehicle is accelerating. Therefore, there is further motivation for also gain-scheduling on longitudinal acceleration. However, this increase in gain scheduling parameters significantly increases the model's and the controller's complexity. In the end, this increase in modeling complexity needs to be motivated by an observed deficiency in control performance. Such a deficiency is not observed in closed-course or public road testing.

It is worth pointing out that the vehicle developed and tested in this Chapter has a Dynamic Driving Task (DDT) [51] constrained to lane-keeping tasks and lane changes (not emergency, collision-avoidance maneuvers). So it is reasonable to argue that the control deficiencies would not be observed under this constrained DDT. Should the DDT extend to more aggressive maneuvers, further work might be to consider a white box modeling approach of the actuator or to schedule the LPV model on longitudinal acceleration.

Despite these motivations for scheduling on more signals, the LPV affine steering actuator model is scheduled only on longitudinal velocity. This achieves a good balance between fit and complexity. The next step is to explore methods of controlling the bike model and the new steering actuator model. Two control architectures are proposed and implemented using the CarSim vehicle. The results for the Cascade control architecture are preferred over those of the Torque Inverse control.

This architecture is then applied to the Jeep Grand Cherokee. However, there are some notable differences between the CarSim system and the Jeep. The most important difference is the presence of significant pure time delay. This is identified along with the parameters for the bicycle car model and LPV affine steering actuator model. There are two approaches to identifying these parameters: estimating them separately or jointly. It is shown that jointly estimating them provides a slightly better fit. These parameters are then used to design a Cascade control architecture that is experimentally tested on the Jeep. The results collected on the runways at Rellis show a balance of comfort and tracking performance. While there was no opportunity to test in snow or a blizzard, there was an opportunity to test in heavy rain. This test was conducted on public

roads with some additional safety features implemented in the controller. The results show comfortable and robust tracking.

Chapter 8

Conclusions

8.1 High-Level Summary

This Thesis is a composition of several investigations that are tied together and motivated by the Lateral Motion Control problem. There are two overarching motivations: (1) this problem needs to be solved for rural environments and (2) the solution needs to be cheap (as defined in Chapter 6). The first motivation presents the challenges of different road conditions, degraded sensor feedback, and adverse weather. The second motivation constrains the solution space to simple control techniques. The approach taken in this Thesis is to start with the simplest solution and incrementally grow more complex as requirements become more challenging to meet.

The main argument of this Thesis is that there exists a single, cheap controller that can achieve these challenging requirements. Chapter 6 supports this argument through many simulation experiments of cheap controllers. This Chapter presents one the main results of this Thesis which shows that various cheap controllers can achieve suitably safe control performance in a variety of maneuvers throughout a variety of Operational Design Domains. Further evidence for this argument is presented in Chapters 5 and 7. Chapter 7 proposes a cascade control architecture in which the inner controller commands steering rack torque and the outer loop control commands steering wheel angle. Chapter 7 presents a control design that achieves a sufficient balance between passenger comfort and tracking accuracy on a 2019 Jeep Grand Cherokee. These results are collected on a closed-course runway in ideal weather conditions. Further results are collected on public roads in ideal weather and in heavy rain conditions. Chapter 5 presents an LPV variant of this controller that shows improved performance across the operational velocity of rural highways as well as on

gravel roads.

Several other arguments are presented in this Thesis. Chapter 2 argues that requirements for the Pose, Planner, and Control modules should be designed together. Chapter 3 presents evidence supporting the largely agreed-upon argument that the dynamic bicycle car model better models a real-world vehicle's dynamics throughout its operational velocity range. Chapter 4 presents several arguments. Its results show that small improvements in control performance can be made by making some nuanced implementations. The most significant impact is the use of a nonlinear modification of the model's output when using the error state model. Another important result is that little difference is observed in control performance when the lateral motion controller is implemented as a path-following or a trajectory-following controller. While the results for severely degraded longitudinal performance indicate that path-following better decouples lateral control from longitudinal control, this example is admittedly exaggerated. It is largely for illustration purposes and other considerations not treated in this Thesis may make either path-following or trajectory-following more suitable for a particular system.

Another contribution that this Thesis makes is the extension of the Interpolation Conditions presented in Chapter 5. These Interpolation Conditions are originally formulated for Linear Time-Invariant control design. This Chapter exploits these conditions to compute a Linear Parameter Varying controller that can then be implemented in numerous ways. The Interpolation Conditions help solve the problem of internal stability and help make the control design more applicable to a wide variety of plants. The lateral motion control problem is a nice example to explore this technique with since its plant has two poles at the origin. These two poles present interpolation conditions that impose limitations on the achievable performance. The Interpolation Conditions make this observation clear and relatively easy to address.

8.2 Future Work

A major oversight of this Thesis is the assumption that the lateral position needs to be tracked to ensure the lateral motion controller guarantees asymptotic tracking of a path or trajectory. This assumption is better thought of as an assumption on system structure. The lateral motion controller is given the task of asymptotically tracking position. However, this need not be the case. For instance, the lateral motion controller could be giving the task of asymptotically tracking orientation, the derivative of orientation, the derivative of position, or some combination of these tasks. In the end, the vehicle must stay within the lane.

Or, to use the terms in Chapter 2, the controller must guarantee the vehicle stays within a virtual corridor. This task was interpreted in this thesis as requiring position tracking. However, it is also possible to give the task of position tracking to a local planner and the task of velocity or orientation tracking to the lateral motion controller. The task of guaranteeing the vehicle remains inside the virtual corridor is then shared between the local planner and the lateral motion controller.

This may present better performance because it separates lateral motion control into controlling velocities and positions. Velocity tracking is achieved in the Control module and position tracking is achieved in the Planner module. Requiring the Planner to track positions is beneficial because it also has to avoid collisions. The work proposed in this thesis assumes a static environment and therefore a static trajectory is sufficient. However, because of this choice, the Control module must track position regardless of the length or duration of the path or trajectory.

This shortcoming gets at the higher-level question that is not addressed in this Thesis: what is the most appropriate system architecture for Automated Driving Systems? This is a question with a very large scope, but it is one that is at the center of the race to develop Automated Driving Systems that can impact lives every day. Future work should be devoted to investigating the Planner-Control interface. This interface needs to be verifiably safe, sufficiently robust for all Operational Design Domains, and reasonably comfortable for passengers and other road users.

Bibliography

[1] AAA. *Evaluation of Active Driving Assistance Systems*. Aug. 2020. URL: https://newsroom.aaa.com/2020/08/aaa-finds-active-driving-assistance-systems-do-less-to-assist-drivers-and-more-to-interfere/ (visited on 10/01/2023).

[2] J. Ackermann et al. "Linear and nonlinear controller design for robust automatic steering". In: *IEEE Transactions on Control Systems Technology* 3.1 (1995), pp. 132–143. DOI: 10.1109/87.370719.

[3] Juergen Ackermann and Wolfgang Sienel. "Robust Control for Automatic Steering". In: *1990 American Control Conference*. 1990, pp. 795–800. DOI: 10.23919/ACC.1990.4790841.

[4] *Adams Car*. URL: https://hexagon.com/products/adams-car (visited on 09/19/2023).

[5] Brian DO Anderson. "From Youla–Kucera to identification, adaptive and nonlinear control". In: *Automatica* 34.12 (1998), pp. 1485–1506.

[6] Brian DO Anderson and John B Moore. *Optimal control: linear quadratic methods*. 1990.

[7] Pierre Apkarian and Richard J Adams. "Advanced gain-scheduling techniques for uncertain systems". In: *Advances in linear matrix inequality methods in control*. SIAM, 2000, pp. 209–228.

[8] Pierre Apkarian, Jean-Marc Biannic, and Pascal Gahinet. "Self-scheduled H-infinity control of missile via linear matrix inequalities". In: *Journal of Guidance, Control, and Dynamics* 18.3 (1995), pp. 532–538.

[9] Pierre Apkarian and Pascal Gahinet. "A convex characterization of gain-scheduled h-infinity controllers". In: *IEEE Transactions on Automatic Control* 40.5 (1995), pp. 853–864.

[10] Pierre Apkarian, Pascal Gahinet, and Greg Becker. "Self-scheduled H-infinity control of linear parameter-varying systems: a design example". In: *Automatica* 31.9 (1995), pp. 1251–1261.

[11] Pierre Apkarian and Hoang Duong Tuan. “Parameterized LMIs in control theory”. In: *SIAM journal on control and optimization* 38.4 (2000), pp. 1241–1264.

[12] Ehsan Arasteh and Francis Assadian. “A Comparative Analysis of Brake-by-Wire Smart Actuators Using Optimization Strategies”. In: *Energies* 15.2 (2022). ISSN: 1996-1073. DOI: 10.3390/en15020634. URL: https://www.mdpi.com/1996-1073/15/2/634.

[13] Francis Assadian, Alex K. Beckerman, and Jose Velazquez Alcantar. “Estimation Design Using Youla Parametrization With Automotive Applications”. In: *Journal of Dynamic Systems, Measurement, and Control* 140.8 (Mar. 2018), p. 081015. ISSN: 0022-0434. DOI: 10.1115/1.4039157. eprint: https://asmedigitalcollection.asme.org/dynamicsystems/article-pdf/140/8/081015/6127181/ds_140_08_081015.pdf. URL: https://doi.org/10.1115/1.4039157.

[14] *AsteRx-i3 D Pro+ Datasheet*. URL: https://www.septentrio.com/en/products/gnss-receivers/gps-gnss-ins-oem-boards/asterx-i3-d-pro-plus (visited on 06/30/2024).

[15] Karl Johan Astrom and Richard M. Murray. “Frequency Domain Design”. In: 2nd ed. Princeton University Press, 2020. Chap. 12.

[16] Karl Johan Åström and Björn Wittenmark. *Computer-controlled systems: theory and design*. Prentice-Hall, 1997.

[17] Hussam Atoui et al. “Advanced LPV-YK Control Design with Experimental Validation on Autonomous Vehicles”. working paper or preprint. Apr. 2022. URL: https://hal.univ-grenoble-alpes.fr/hal-03648497.

[18] Hussam Atoui et al. “LPV-Based Autonomous Vehicle Lateral Controllers: A Comparative Analysis”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.8 (2022), pp. 13570–13581. DOI: 10.1109/TITS.2021.3125771.

[19] Hussam Atoui et al. “Multi-variable and multi-objective gain-scheduled control based on Youla-Kucera parameterization: Application to autonomous vehicles”. In: *International Journal of Robust and Nonlinear Control* (2024).

[20] Hussam Atoui et al. “Toward switching/interpolating LPV control: A review”. In: *Annual Reviews in Control* 54 (2022), pp. 49–67.

[21] *AutonomousStuff*. URL: https://autonomoustuff.com/ (visited on 06/30/2024).

[22] *AVA: Automated Vehicles for All*. URL: https://www.transportation.gov/av/grants (visited on 06/30/2024).

[23] Aly Badawy et al. *Modeling and analysis of an electric power steering system*. Tech. rep. SAE Technical Paper, 1999.

[24] D. Barros, S. Fekri, and M. Athans. “Robust Mixed-Mu Synthesis Performance for Mass-Spring System with Stiffness Uncertainty”. In: *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.* 2005, pp. 743–748. DOI: 10.1109/.2005.1467107.

[25] Johannes Betz et al. “A software architecture for an autonomous racecar”. In: *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE. 2019, pp. 1–6.

[26] Johannes Betz et al. “Autonomous Driving—A Crash Explained in Detail”. In: *Applied Sciences* 9.23 (2019). ISSN: 2076-3417. DOI: 10.3390/app9235126. URL: https://www.mdpi.com/2076-3417/9/23/5126.

[27] Johannes Betz et al. “Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing”. In: *IEEE Open Journal of Intelligent Transportation Systems* 3 (2022), pp. 458–488. DOI: 10.1109/OJITS.2022.3181510.

[28] Mario Bijelic et al. “Seeing Through Fog Without Seeing Fog: Deep Multimodal Sensor Fusion in Unseen Adverse Weather”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[29] S. Brennan and A. Alleyne. “The Illinois Roadway Simulator: a mechatronic testbed for vehicle dynamics and control”. In: *IEEE/ASME Transactions on Mechatronics* 5.4 (2000), pp. 349–359. DOI: 10.1109/3516.891046.

[30] S. Brennan and A. Alleyne. “Using a scale testbed: Controller design and evaluation”. In: *IEEE Control Systems Magazine* 21.3 (2001), pp. 15–26. DOI: 10.1109/37.924794.

[31] Sean Brennan. “Modeling and Control Issues Associated with Scaled Vehicles”. MA thesis. New Mexico State University, 1997.

[32] Sean Brennan and Andrew Alleyne. “Robust scalable vehicle control via non-dimensional vehicle dynamics”. In: *Vehicle System Dynamics* 36.4-5 (2001), pp. 255–277.

[33] Corentin Briat. *Linear parameter-varying and time-delay systems*. Springer, 2014.

[34] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[35] Jeffrey A. Butterworth, Lucy Y. Pao, and Daniel Y. Abramovitch. “The effect of nonminimum-phase zero locations on the performance of feedforward model-inverse control techniques in discrete-time systems”. In: *2008 American Control Conference*. 2008, pp. 2696–2702. DOI: 10.1109/ACC.2008.4586900.

[36] L. Dugard C. Poussot-Vassal O. Sename and S. M. Savaresi. “Vehicle dynamic stability improvements through gain-scheduled steering and braking control”. In: *Vehicle System Dynamics* 49.10 (2011), pp. 1597–1621. DOI: 10.1080/00423114.2010.527995. eprint: https://doi.org/10.1080/00423114.2010.527995. URL: https://doi.org/10.1080/00423114.2010.527995.

[37] Davide Calzolari, Bastian Schürmann, and Matthias Althoff. “Comparison of trajectory tracking controllers for autonomous vehicles”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. 2017, pp. 1–8. DOI: 10.1109/ITSC.2017.8317800.

[38] *Car Sim*. URL: https://www.carsim.com/products/carsim/index.php (visited on 09/19/2023).

[39] *CarMaker*. URL: https://www.ipg-automotive.com/en/products-solutions/software/carmaker/ (visited on 06/30/2024).

[40] Ching-Yao Chan et al. “Characterization of magnetic tape and magnetic markers as a position sensing system for vehicle guidance and control”. In: *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*. Vol. 1. 2000, 95–99 vol.1. DOI: 10.1109/ACC.2000.878780.

[41] Christoforos Chatzikomis et al. “Comparison of Path Tracking and Torque-Vectoring Controllers for Autonomous Electric Vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 3.4 (2018), pp. 559–570. DOI: 10.1109/TIV.2018.2874529.

[42] Chi-Tsong Chen. *Linear system theory and design*. Saunders college publishing, 1984.

[43] Mahmoud Chilali and Pascal Gahinet. “H-infinity design with pole placement constraints: an lmi approach”. In: *IEEE Transactions on automatic control* 41.3 (1996), pp. 358–367.

[44] Jessica B. Cicchino. “Effects of lane departure warning on police-reported crash rates”. In: *Journal of Safety Research* 66 (2018), pp. 61–70. ISSN: 0022-4375. DOI: https://doi.org/10.1016/j.jsr.2018.05.006. URL: https://www.sciencedirect.com/science/article/pii/S002243751730556X.

[45] Laurène Claussmann et al. “A Review of Motion Planning for Highway Autonomous Driving”. In: *IEEE Transactions on Intelligent Transportation Systems* 21.5 (2020), pp. 1826–1848. DOI: 10.1109/TITS.2019.2913998.

[46] Jarrod M. Snider. *Automatic Steering Methods for Autonomous Automobile Path Tracking*. Tech. rep. Robotics Institute, Carnegie Mellon University, 2009.

[47] Omead Amidi. *Integrated Mobile Robot Control*. Tech. rep. The Robotics Institute, Carnegie Mellon University, 1990.

[48] Craig Coulter. *Implementation of the Pure Pursuit Tracking Algorithm*. Tech. rep. The Robotics Institute, Carnegie Mellon University, 1992.

[49] *Comma Ai openpilot 0.9.4 Release*. Version 0.9.4. 2023. URL: https://github.com/commaai/openpilot/releases/tag/v0.9.4.

[50] Active Safety Systems Standards Committee. *Active Safety Systems Terms and Definitions*. Mar. 2021. DOI: 10.4271/J3063_202103.

[51] On-Road Automated Driving (ORAD) Committee. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. Apr. 2021. DOI: 10.4271/J3016_202104.

[52] Vehicle Dynamics Standards Committee. *Steady-State Directional Control Test Procedures for Passenger Cars and Light Trucks*. Nov. 2018. DOI: https://doi.org/10.4271/J266_201811. URL: https://doi.org/10.4271/J266_201811.

[53] OpenStreetMap contributors. *Planet dump retrieved from https://planet.osm.org*. https://www.openstreetmap.org. 2023.

[54] W.H. Cormier and R.E. Fenton. "On the steering of automated vehicles—A velocity-adaptive controller". In: *IEEE Transactions on Vehicular Technology* 29.4 (1980), pp. 375–385. DOI: 10.1109/T-VT.1980.23869.

[55] Matteo Corno et al. "An LPV Approach to Autonomous Vehicle Path Tracking in the Presence of Steering Actuation Nonlinearities". In: *IEEE Transactions on Control Systems Technology* 29.4 (2021), pp. 1766–1774. DOI: 10.1109/TCST.2020.3006123.

[56] Cruise. *Cruise Safety Report*. Tech. rep. Cruise, 2022. URL: https://getcruise.com/safety/ (visited on 08/16/2023).

[57] *Cubic Smoothing Spline - MATLAB*. https://www.mathworks.com/help/curvefit/csaps.html. Accessed: 2023-12-29.

[58] J. Doyle, A. Packard, and K. Zhou. "Review of LFTs, LMIs, and mu". In: *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*. 1991, 1227–1232 vol.2. DOI: 10.1109/CDC.1991.261572.

[59] Ellen Edmonds. *Rained Out: Vehicle Safety Systems Struggle to "See" in Bad Weather*. 2021. URL: https://newsroom.aaa.com/2021/10/rained-out-vehicle-safety-systems-struggle-to-see-in-bad-weather/ (visited on 10/01/2023).

[60] Paolo Falcone et al. "Predictive Active Steering Control for Autonomous Vehicle Systems". In: *IEEE Transactions on Control Systems Technology* 15.3 (2007), pp. 566–580. DOI: 10.1109/TCST.2007.894653.

[61] Kevin R. Mallon Farhad Assadian. *Robust Control: Youla Parameterization Approach*. John Wiley and Sons, Inc., 2022.

[62] Francesca Favaro et al. *Building a Credible Case for Safety: Waymo's Approach for the Determination of Absence of Unreasonable Risk*. 2023. arXiv: 2306.01917 [cs.CY].

[63] Yanming Feng, Charles Wang, and Charles Karl. “Determination of Required Positioning Integrity Parameters for Design of Vehicle Safety Applications”. In: *Proceedings of the 2018 International Technical Meeting of The Institute of Navigation*. Reston, Virginia, Jan. 2018, pp. 129–141.

[64] R. Fenton, G. Melocik, and K. Olson. “On the steering of automated vehicles: Theory and experiment”. In: *IEEE Transactions on Automatic Control* 21.3 (1976), pp. 306–315. DOI: 10.1109/TAC.1976.1101230.

[65] R.E. Fenton and R.J. Mayhan. “Automated highway studies at the Ohio State University-an overview”. In: *IEEE Transactions on Vehicular Technology* 40.1 (1991), pp. 100–113. DOI: 10.1109/25.69978.

[66] R.E. Fenton and I. Selim. “On the optimal design of an automotive lateral controller”. In: *IEEE Transactions on Vehicular Technology* 37.2 (1988), pp. 108–113. DOI: 10.1109/25.9890.

[67] Louis Filipozzi and Francis Assadian. “Control of Over-Actuated Systems — From Practical to Theoretical Concepts with Application in Hybrid Powertrain Speed Control Development”. In: *2022 IEEE Vehicle Power and Propulsion Conference (VPPC)*. 2022, pp. 1–6. DOI: 10.1109/VPPC55846.2022.10003327.

[68] Michael S. Floater. “On the deviation of a parametric cubic spline interpolant from its data polygon”. In: *Computer Aided Geometric Design* 25.3 (2008), pp. 148–156. ISSN: 0167-8396. DOI: https://doi.org/10.1016/j.cagd.2007.08.001. URL: https://www.sciencedirect.com/science/article/pii/S0167839607000908.

[69] RG Franks and CW Worley. “Quantitative analysis of cascade control”. In: *Industrial & Engineering Chemistry* 48.6 (1956), pp. 1074–1079.

[70] Cheryl Thole, Alasdair Cain, and Jennifer Flynn. *The EmX Franklin Corridor BRT Project Evaluation*. Tech. rep. National Bus Rapid Transit Institute, Apr. 2009.

[71] Pascal Gahinet. “Explicit controller formulas for LMI-based H-infinity synthesis”. In: *Automatica* 32.7 (1996), pp. 1007–1014.

[72] Pascal Gahinet and Pierre Apkarian. “A linear matrix inequality approach to H-infinity control”. In: *International journal of robust and nonlinear control* 4.4 (1994), pp. 421–448.

[73] K. Gardels. *Automatic car controls for electronic highways*. Tech. rep. General Motors Research Labs, 1960.

[74] Erwin de Gelder et al. "Risk Quantification for Automated Driving Systems in Real-World Driving Scenarios". In: *IEEE Access* 9 (2021), pp. 168953–168970. DOI: 10.1109/ACCESS.2021.3136585.

[75] Thomas D. Gillespie. *Fundamentals of Vehicle Dynamics - Revised Edition*. SAE International, 2021. ISBN: 978-1-4686-0356-9.

[76] Jonathan Y. Goh, Tushar Goel, and J. Christian Gerdes. "Toward Automated Vehicle Control Beyond the Stability Limits: Drifting Along a General Path". In: *Journal of Dynamic Systems, Measurement, and Control* 142.2 (Nov. 2019), p. 021004. ISSN: 0022-0434. DOI: 10.1115/1.4045320. eprint: https://asmedigitalcollection.asme.org/dynamicsystems/article-pdf/142/2/021004/6472363/ds_142_02_021004.pdf. URL: https://doi.org/10.1115/1.4045320.

[77] David González et al. "A Review of Motion Planning Techniques for Automated Vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 17.4 (2016), pp. 1135–1145. DOI: 10.1109/TITS.2015.2498841.

[78] *GPS Simulation Model*. URL: https://www.mathworks.com/help/nav/ref/gps.html (visited on 06/30/2024).

[79] Sergio Guillen and Francis Assadian. "Unique approaches to integrating semi-active suspension and active anti-roll bar control systems". In: *International Journal of Vehicle Design* 90.1-4 (2022), pp. 98–115. DOI: 10.1504/IJVD.2022.129167. eprint: https://www.inderscienceonline.com/doi/pdf/10.1504/IJVD.2022.129167. URL: https://www.inderscienceonline.com/doi/abs/10.1504/IJVD.2022.129167.

[80] Sergio Guillen and Francis Assadian. "Unique approaches to integrating semi-active suspension and active anti-roll bar control systems". In: *International Journal of Vehicle Design* (2023).

[81] J. Guldner, Han-Shue Tan, and S. Patwardhan. "Study of design directions for lateral vehicle control". In: *Proceedings of the 36th IEEE Conference on Decision and Control*. Vol. 5. 1997, 4732–4737 vol.5. DOI: 10.1109/CDC.1997.649756.

[82] J. Guldner, V.I. Utkin, and J. Ackermann. “A sliding mode control approach to automatic car steering”. In: *Proceedings of 1994 American Control Conference - ACC ’94*. Vol. 2. 1994, 1969–1973 vol.2. DOI: 10.1109/ACC.1994.752420.

[83] JÜRGEN GULDNER, HAN-SHUE TAN, and SATYAJIT PATWARDHAN. “Analysis of Automatic Steering Control for Highway Vehicles with Look-down Lateral Reference Systems”. In: *Vehicle System Dynamics* 26.4 (1996), pp. 243–269. DOI: 10.1080/00423119608969311. eprint: https://doi.org/10.1080/00423119608969311. URL: https://doi.org/10.1080/00423119608969311.

[84] H. Hailu and S. Brennan. “Use of dimensional analysis to reduce the parametric space for gain-scheduling”. In: *Proceedings of the 2005, American Control Conference, 2005.* 2005, 598–603 vol. 1. DOI: 10.1109/ACC.2005.1470022.

[85] Haftay Hailu. *Dimensional transformation: A novel method for gain-scheduling and robust control*. The Pennsylvania State University, 2006.

[86] Haftay Hailu and Sean Brennan. “Reduction in the Number of Gain-Scheduling Parameters Using Dimensional Transformation”. In: *Journal of Dynamic Systems, Measurement, and Control* 130 (2008), pp. 034505–1.

[87] F. Hamano and G. Marro. “Using preaction to eliminate tracking error in feedback control of non-minimum phase systems”. In: *Proceedings of 35th IEEE Conference on Decision and Control*. Vol. 4. 1996, 4549–4551 vol.4. DOI: 10.1109/CDC.1996.577583.

[88] C. Hatipoglu, U. Ozguner, and K.A. Redmill. “Automated lane change controller design”. In: *IEEE Transactions on Intelligent Transportation Systems* 4.1 (2003), pp. 13–22. DOI: 10.1109/TITS.2003.811644.

[89] J.K. Hedrick, M. Tomizuka, and P. Varaiya. “Control issues in automated highway systems”. In: *IEEE Control Systems Magazine* 14.6 (1994), pp. 21–32. DOI: 10.1109/37.334412.

[90] Alexander Heilmeier et al. “Minimum curvature trajectory planning and control for an autonomous race car”. In: *Vehicle System Dynamics* (2019).

[91] J William Helton and Orlando Merino. *Classical Control Using H-Infinity Methods: An Introduction to Design*. SIAM, 1998.

[92] Leonhard Hermansdorfer, Johannes Betz, and Markus Lienkamp. "Benchmarking of a software stack for autonomous racing against a professional human race driver". In: *2020 Fifteenth international conference on ecological vehicles and renewable energies (EVER)*. IEEE. 2020, pp. 1–8.

[93] João P. Hespanha. *Linear Systems Theory*. 2nd ed. Princeton University Press, 2018. DOI: 9780691179575.

[94] Joao P Hespanha and A Stephen Morse. "Switching between stabilizing controllers". In: *Automatica* 38.11 (2002), pp. 1905–1917.

[95] T. Hessburg and M. Tomizuka. "Fuzzy logic control for lateral vehicle guidance". In: *IEEE Control Systems Magazine* 14.4 (1994), pp. 55–63. DOI: 10.1109/37.295971.

[96] Rami Y Hindiyeh and J Christian Gerdes. "A controller framework for autonomous drifting: Design, stability, and experimental validation". In: *Journal of Dynamic Systems, Measurement, and Control* 136.5 (2014), p. 051015.

[97] P. Hoblet, R.T. O'Brien, and J.A. Piepmeier. "Scale-model vehicle analysis for the design of a steering controller". In: *Proceedings of the 35th Southeastern Symposium on System Theory, 2003.* 2003, pp. 201–205. DOI: 10.1109/SSST.2003.1194558.

[98] Christian Hoffmann and Herbert Werner. "A survey of linear parameter-varying control applications validated by experiments or high-fidelity simulations". In: *IEEE Transactions on Control Systems Technology* 23.2 (2014), pp. 416–433.

[99] Gabriel M. Hoffmann et al. "Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing". In: *2007 American Control Conference*. 2007, pp. 2296–2301. DOI: 10.1109/ACC.2007.4282788.

[100] Gabriel M. Hoffmann et al. "Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing". In: *2007 American Control Conference*. 2007, pp. 2296–2301. DOI: 10.1109/ACC.2007.4282788.

[101] Chuan Hu et al. "Robust H-Infinity output-feedback control for path following of autonomous ground vehicles". In: *Mechanical Systems and Signal Processing* 70-71 (2016), pp. 414–427. ISSN: 0888-3270. DOI: https://doi.org/10.1016/j.ymssp.2015.09.017. URL: https://www.sciencedirect.com/science/article/pii/S0888327015004124.

[102] Jihua Huang and Han-Shue Tan. “Control System Design of an Automated Bus in Revenue Service”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.10 (2016), pp. 2868–2878. DOI: 10.1109/TITS.2016.2530760.

[103] Jihua Huang and Han-Shue Tan. “Control System Design of an Automated Bus in Revenue Service”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.10 (2016), pp. 2868–2878. DOI: 10.1109/TITS.2016.2530760.

[104] Jihua Huang and Han-Shue Tan. “Development and Validation of an Automated Steering Control System for Bus Revenue Service”. In: *IEEE Transactions on Automation Science and Engineering* 13.1 (2016), pp. 227–237. DOI: 10.1109/TASE.2015.2497256.

[105] *IMU Simulation Model*. URL: https://www.mathworks.com/help/nav/ref/imu.html (visited on 06/30/2024).

[106] National Bus Rapid Transit Institute. *EmX Map and Schedule*. National Bus Rapid Transit Institute, 2023. URL: https://www.ltd.org/system-map/route%5C_103/ (visited on 08/17/2023).

[107] Petros A. Ioannou, ed. *Automated Highway Systems*. Springer New York, 1997.

[108] Abubakar Isah, Ameer Mohammed, and Aminu Hamza. “Electric Power-Assisted Steering: A Review”. In: *2019 2nd International Conference of the IEEE Nigeria Computer Chapter (NigeriaComputConf)*. 2019, pp. 1–6. DOI: 10.1109/NigeriaComputConf45974.2019.8949620.

[109] Rolf Isermann. *Digital Control Systems*. 1981.

[110] International Organization for Standardization. *Passenger cars - Vehicle dynamic simulation and validation - Steady-state circular driving behaviour*. Standard. Geneva, CH: International Organization for Standardization, 2016.

[111] International Organization for Standardization. *Passenger cars — Validation of vehicle dynamics simulation — Lateral transient response test methods*. Standard. Geneva, CH: International Organization for Standardization, 2021.

[112] International Organization for Standardization. *Road vehicles — Functional safety*. Standard. Geneva, CH: International Organization for Standardization, 2018.

[113] International Organization for Standardization. *Passenger cars — Test track for a severe lane-change manoeuvre — Part 1: Double lane-change*. Standard. Geneva, CH: International Organization for Standardization, 2018.

[114] International Organization for Standardization. *Road vehicles - Lateral transient response test methods - Open-loop test methods*. Standard. Geneva, CH: International Organization for Standardization, 2011.

[115] International Organization for Standardization. *Mechanical vibration — Road surface profiles — Reporting of measured data*. Standard. Geneva, CH: International Organization for Standardization, 2016.

[116] International Organization for Standardization. *Road Vehicles—Safety of the Intended Functionality*. Standard. Geneva, CH: International Organization for Standardization, 2019.

[117] Seungmin Jeon et al. "Path Tracking Control of Autonomous Vehicles Using Augmented LQG with Curvature Disturbance Model". In: *2019 19th International Conference on Control, Automation and Systems (ICCAS)*. 2019, pp. 1543–1548. DOI: 10.23919/ICCAS47443.2019.8971654.

[118] Yugong Luo Jinghua Guo and Keqiang Li. "Robust gain-scheduling automatic steering control of unmanned ground vehicles under velocity-varying motion". In: *Vehicle System Dynamics* 57.4 (2019), pp. 595–616. DOI: 10.1080/00423114.2018.1475677. eprint: https://doi.org/10.1080/00423114.2018.1475677. URL: https://doi.org/10.1080/00423114.2018.1475677.

[119] Allen Tannenbaum John Doyle Bruce Francis. *Feedback Control Theory*. Macmillan Publishing Co., 1990.

[120] J. C. Kaimal et al. "Spectral characteristics of surface-layer turbulence". In: *Quarterly Journal of the Royal Meteorological Society* 98.417 (1972), pp. 563–589. DOI: https://doi.org/10.1002/qj.49709841707. eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/qj.49709841707. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.49709841707.

[121] Nitin R. Kapania and J. Christian Gerdes. "Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling". In: *Vehicle System Dynamics*

53.12 (2015), pp. 1687–1704. DOI: 10.1080/00423114.2015.1055279. eprint: https://doi.org/10.1080/00423114.2015.1055279. URL: https://doi.org/10.1080/00423114.2015.1055279.

[122] Nitin R. Kapania and J. Christian Gerdes. "Learning at the Racetrack: Data-Driven Methods to Improve Racing Performance Over Multiple Laps". In: *IEEE Transactions on Vehicular Technology* 69.8 (2020), pp. 8232–8242. DOI: 10.1109/TVT.2020.2998065.

[123] Dimitrios Kapsalis et al. "A reduced LPV polytopic look-ahead steering controller for autonomous vehicles". In: *Control Engineering Practice* 129 (2022), p. 105360.

[124] Dimitrios Kapsalis et al. "Gain-scheduled steering control for autonomous vehicles". In: *IET Control Theory & Applications* 14.20 (2020), pp. 3451–3460. DOI: https://doi.org/10.1049/iet-cta.2020.0698. eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-cta.2020.0698. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-cta.2020.0698.

[125] Yassine Kebbati et al. "Lateral control for autonomous wheeled vehicles: A technical review". In: *Asian Journal of Control* 25.4 (2023), pp. 2539–2563. DOI: https://doi.org/10.1002/asjc.2980. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asjc.2980. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/asjc.2980.

[126] Siddartha Khastgir et al. "Systems Approach to Creating Test Scenarios for Automated Driving Systems". In: *Reliability Engineering & System Safety* 215 (2021), p. 107610. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.107610. URL: https://www.sciencedirect.com/science/article/pii/S0951832021001551.

[127] Olivier N. Kigotho and Jason H. Rife. "Comparison of Rectangular and Elliptical Alert Limits for Lane-Keeping Applications". In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)*. St. Louis, Missouri, Sept. 2021, pp. 93–104.

[128] Ji-Hoon Kim and Jae-Bok Song. "Control logic for an electric power steering system using assist motor". In: *Mechatronics* 12.3 (2002), pp. 447–459.

[129] Ilya Kolmanovsky, Emanuele Garone, and Stefano Di Cairano. “Reference and command governors: A tutorial on their theory and automotive applications”. In: *2014 American Control Conference*. IEEE. 2014, pp. 226–241.

[130] Jason Kong et al. “Kinematic and dynamic vehicle models for autonomous driving control design”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. 2015, pp. 1094–1099. DOI: 10.1109/IVS.2015.7225830.

[131] Vladimír Kučera. “Stability of discrete linear feedback systems”. In: *IFAC Proceedings Volumes* 8.1 (1975), pp. 573–578.

[132] Vincent A. Laurense, Jonathan Y. Goh, and J. Christian Gerdes. “Path-tracking for autonomous vehicles at the limit of friction”. In: *2017 American Control Conference (ACC)*. 2017, pp. 5586–5591. DOI: 10.23919/ACC.2017.7963824.

[133] Lennart Ljung. *System Identification: Theory for the User*. 2nd ed. Prentice Hall, 1999.

[134] Bi-Cheng Luan et al. “Design and Field Testing of a Lane Following Control System with a Camera Based on T&C Driver Model”. In: *SAE 2016 World Congress and Exhibition*. SAE International, Apr. 2016. DOI: https://doi.org/10.4271/2016-01-0117. URL: https://doi.org/10.4271/2016-01-0117.

[135] I. Mahtout et al. “Youla-Kucera Based Lateral Controller for Autonomous Vehicle”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018, pp. 3281–3286. DOI: 10.1109/ITSC.2018.8569779.

[136] Imane Mahtout et al. “Advances in Youla-Kucera parametrization: A Review”. In: *Annual Reviews in Control* 49 (2020), pp. 81–94. ISSN: 1367-5788. DOI: https://doi.org/10.1016/j.arcontrol.2020.04.015. URL: https://www.sciencedirect.com/science/article/pii/S1367578820300249.

[137] Lorenzo Marconi, Giovanni Marro, and Claudio Melchiorri. “A solution technique for almost perfect tracking of non-minimum-phase, discrete-time linear systems”. In: *International Journal of Control* 74.5 (2001), pp. 496–506.

[138] Donald Margolis and Taehyun Shim. “A bond graph model incorporating sensors, actuators, and vehicle dynamics for developing controllers for vehicle safety”. In: *Journal of the Franklin Institute* 338.1 (2001), pp. 21–34.

[139] G Marro and L Marconi. “Using the diophantine equation in the design of a digital perfect tracking compensator”. In: *1997 European Control Conference (ECC)*. IEEE. 1997, pp. 1037–1042.

[140] Helmut Martin et al. “Functional Safety of Automated Driving Systems: Does ISO 26262 Meet the Challenges?” In: *Automated Driving: Safer and More Efficient Future Driving*. Cham: Springer International Publishing, 2017, pp. 387–416. ISBN: 978-3-319-31895-0. DOI: 10.1007/978-3-319-31895-0_16. URL: https://doi.org/10.1007/978-3-319-31895-0%5C_16.

[141] Sanjiv Singh Martin Buehler Karl Iagnemma, ed. *The 2005 DARPA Grand Challenge - The Great Robot Race*. Springer Science and Business Media, 2007.

[142] Sanjiv Singh Martin Buehler Karl Iagnemma, ed. *The DARPA Urban Challenge - Autonomous Vehicles in City Traffic*. Springer Science and Business Media, 2009.

[143] Jose A. Matute et al. “Experimental Validation of a Kinematic Bicycle Model Predictive Control with Lateral Acceleration Consideration”. In: *IFAC-PapersOnLine* 52.8 (2019). 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019, pp. 289–294. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2019.08.085. URL: https://www.sciencedirect.com/science/article/pii/S2405896319304185.

[144] Duncan C McFarlane and Keith Glover. *Robust controller design using normalized coprime factor plant descriptions*. Springer, 1990.

[145] Mark L McMulkin and Jeffrey C Woldstad. “Effects of wheel design on the torques applied to large hand wheels”. In: *International Journal of Industrial Ergonomics* 15.3 (1995), pp. 205–213.

[146] Alexandre Megretski. *Lecture 11: The Tustin Transformation*. MIT OpenCourseWare. Cambridge, MA: Massachusetts Institute of Technology, 2004. URL: https://ocw.mit.edu/courses/6-245-multivariable-control-systems-spring-2004/resources/lec11_6245_2004/.

[147] Naser Mehrabi. “Dynamics and model-based control of electric power steering systems”. In: (2014).

[148] Dan Negrut Michael Taylor Radu Serban. *Basic Comparison of Chrono::Vehicle and ADAMS/Car*. Tech. rep. University of Wisconsin-Madison Simulation-Based Engineering Lab, 2016. URL: https://projectchrono.org/validation/ (visited on 09/19/2023).

[149] George Verghese Mohammed Dahleh Munther A. Dahleh. *Chapter 26: Balanced Realization*. MIT OpenCourseWare. Cambridge, MA: Massachusetts Institute of Technology, 2011. URL: https://mitocw.ups.edu.ec/courses/electrical-engineering-and-computer-science/6-241j-dynamic-systems-and-control-spring-2011/lecture-notes/MIT6_241JS11_lec22.pdf.

[150] *New Eagle*. URL: https://neweagle.net/ (visited on 06/30/2024).

[151] Jun Ni, Jibin Hu, and Changle Xiang. "Robust Path Following Control at Driving/Handling Limits of an Autonomous Electric Racecar". In: *IEEE Transactions on Vehicular Technology* 68.6 (2019), pp. 5518–5526. DOI: 10.1109/TVT.2019.2911862.

[152] Nuro. *Delivering Safety*. 2021. URL: https://www.nuro.ai/safety (visited on 10/01/2023).

[153] Metropolitan Planning Organization. *All Motor Vehicle Crashes 2007 - 2021*. Metropolitan Planning Organization, 2023. URL: https://www.lcog.org/thempo/page/advanced-user-data (visited on 08/17/2023).

[154] Hans B. Pacejka, ed. *Tire and Vehicle Dynamics (Third Edition)*. Third Edition. Oxford: Butterworth-Heinemann, 2012. ISBN: 978-0-08-097016-5. DOI: https://doi.org/10.1016/B978-0-08-097016-5.01001-9. URL: https://www.sciencedirect.com/science/article/pii/B9780080970165010019.

[155] BN PARLETT. "Contribution II/11 Balancing a Matrix for Calculation of Eigenvalues and Eigenvectors* by BN PARLETT and C. REINSCH". In: *Handbook for Automatic Computation: Volume II: Linear Algebra* 186 (2012), p. 315.

[156] S. Patwardhan, H.-S. Tan, and M. Tomizuka. "Experimental results of a tire-burst controller for AHS". In: *Control Engineering Practice* 5.11 (1997), pp. 1615–1622. ISSN: 0967-0661. DOI: https://doi.org/10.1016/S0967-0661(97)10017-X. URL: https://www.sciencedirect.com/science/article/pii/S096706619710017X.

[157] Matthew Peet. *Lecture 13: LMIs for Optimal Control and Quadratic Stability with Affine Polytopic and Interval Uncertainty*. 2023. URL: https://control.asu.edu/Classes/MAE509/509Lecture13.pdf.

[158] Huei Peng and Masayoshi Tomizuka. "Preview Control for Vehicle Lateral Guidance in Highway Automation". In: *1991 American Control Conference*. 1991, pp. 3090–3095. DOI: 10.23919/ACC.1991.4791977.

[159] Hung Anh Pham. "Combined Lateral and Longitudinal Control of Vehicles for the Automated Highway System". PhD thesis. University of California at Berkeley, 1996.

[160] Rajesh Rajamani. *Vehicle Dynamics and Control*. Springer Science and Business Media, 2006. ISBN: 0-387-28823-6.

[161] Carlo Ravizolli. "Identification and control of an RC car for drifting purposes". MA thesis. Polytechnic University of Milan, 2017.

[162] Tyler G.R. Reid et al. "Localization Requirements for Autonomous Vehicles". In: *SAE International Journal of Connected and Automated Vehicles* 2.3 (Sept. 2019), pp. 173–190. ISSN: 2574-0741. DOI: https://doi.org/10.4271/12-02-03-0012. URL: https://doi.org/10.4271/12-02-03-0012.

[163] Gábor Rödönyi et al. "Identification of the nonlinear steering dynamics of an autonomous vehicle". In: *IFAC-PapersOnLine* 54.7 (2021), pp. 708–713.

[164] Mohammad Rokonuzzaman et al. "Review and performance evaluation of path tracking controllers of autonomous vehicles". In: *IET Intelligent Transport Systems* 15.5 (2021), pp. 646–670. DOI: https://doi.org/10.1049/itr2.12051. eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/itr2.12051. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/itr2.12051.

[165] F. Roselli et al. "H-infinity control with look-ahead for lane keeping in autonomous vehicles". In: *2017 IEEE Conference on Control Technology and Applications (CCTA)*. 2017, pp. 2220–2225. DOI: 10.1109/CCTA.2017.8062781.

[166] Wilson J Rugh and Jeff S Shamma. "Research on gain scheduling". In: *Automatica* 36.10 (2000), pp. 1401–1425.

[167] Harald Schafer et al. *A Commute in Data: The comma2k19 Dataset*. 2018. eprint: arXiv:1812.05752.

[168] Peter Seiler, Andrew Packard, and Pascal Gahinet. "An Introduction to Disk Margins [Lecture Notes]". In: *IEEE Control Systems Magazine* 40.5 (2020), pp. 78–95. DOI: 10.1109/MCS.2020.3005277.

[169] Jeff S Shamma. "Analysis and design of gain scheduled control systems". PhD thesis. Massachusetts Institute of Technology, 1988.

[170] S E Shladover. "Automated vehicles for highway operations (automated highway systems)". In: *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 219.1 (2005), pp. 53–75. DOI: 10.1243/095440705X9407. eprint: https://doi.org/10.1243/095440705X9407. URL: https://doi.org/10.1243/095440705X9407.

[171] STEVEN E. SHLADOVER. "Review of the State of Development of Advanced Vehicle Control Systems (AVCS)". In: *Vehicle System Dynamics* 24.6-7 (1995), pp. 551–595. DOI: 10.1080/00423119508969108. eprint: https://doi.org/10.1080/00423119508969108. URL: https://doi.org/10.1080/00423119508969108.

[172] Steven E. Shladover. "PATH at 20—History and Major Milestones". In: *IEEE Transactions on Intelligent Transportation Systems* 8.4 (2007), pp. 584–592. DOI: 10.1109/TITS.2007.903052.

[173] Christopher Smith. *20 Best-Selling Cars In The US In 2023*. https://www.motor1.com/features/703891/best-selling-cars-2023/. Accessed: 2023-04-26.

[174] DIRK E. SMITH and JOHN M. STARKEY. "Effects of Model Complexity on the Performance of Automated Vehicle Steering Controllers: Controller Development and Evaluation". In: *Vehicle System Dynamics* 23.1 (1994), pp. 627–645. DOI: 10.1080/00423119408969078. eprint: https://doi.org/10.1080/00423119408969078. URL: https://doi.org/10.1080/00423119408969078.

[175] DIRK E. SMITH and JOHN M. STARKEY. "Effects of Model Complexity on the Performance of Automated Vehicle Steering Controllers: Model Development, Validation and Comparison". In: *Vehicle System Dynamics* 24.2 (1995), pp. 163–181. DOI: 10.1080/00423119508969086.

eprint: https://doi.org/10.1080/00423119508969086. URL: https://doi.org/10.1080/00423119508969086.

[176] Nathan A. Spielberg et al. "Neural network vehicle models for high-performance automated driving". In: *Science Robotics* 4.28 (2019), eaaw1975. DOI: 10.1126/scirobotics.aaw1975. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.aaw1975. URL: https://www.science.org/doi/abs/10.1126/scirobotics.aaw1975.

[177] Gilles Tagne, Reine Talj, and Ali Charara. "Design and Comparison of Robust Nonlinear Controllers for the Lateral Dynamics of Intelligent Vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 17.3 (2016), pp. 796–809. DOI: 10.1109/TITS.2015.2486815.

[178] Kirstin LR Talvala and J Christian Gerdes. "Lanekeeping at the limits of handling: Stability via Lyapunov functions and a comparison with stability control". In: *Dynamic Systems and Control Conference*. Vol. 43352. 2008, pp. 361–368.

[179] Kirstin LR Talvala, Krisada Kritayakirana, and J Christian Gerdes. "Pushing the limits: From lanekeeping to autonomous racing". In: *Annual Reviews in Control* 35.1 (2011), pp. 137–148.

[180] Han-Shue Tan and Jihua Huang. "Design of a High-Performance Automatic Steering Controller for Bus Revenue Service Based on How Drivers Steer". In: *IEEE Transactions on Robotics* 30.5 (2014), pp. 1137–1147. DOI: 10.1109/TRO.2014.2331092.

[181] Han-Shue Tan and Jihua Huang. "Experimental Development of a New Target and Control Driver Steering Model Based on DLC Test Data". In: *IEEE Transactions on Intelligent Transportation Systems* 13.1 (2012), pp. 375–384. DOI: 10.1109/TITS.2011.2174785.

[182] Luqi Tang et al. "An Improved Kinematic Model Predictive Control for High-Speed Path Tracking of Autonomous Vehicles". In: *IEEE Access* 8 (2020), pp. 51400–51413. DOI: 10.1109/ACCESS.2020.2980188.

[183] Siyu Teng et al. "Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives". In: *IEEE Transactions on Intelligent Vehicles* 8.6 (2023), pp. 3692–3711. DOI: 10.1109/TIV.2023.3274536.

[184] Jeyan Thiyagalingam et al. "Scientific machine learning benchmarks". In: *Nature Reviews Physics* 4.6 (2022), pp. 413–420.

[185] Masayoshi Tomizuka. "Zero Phase Error Tracking Algorithm for Digital Control". In: *Journal of Dynamic Systems, Measurement, and Control* 109.1 (Mar. 1987), pp. 65–68. ISSN: 0022-0434. DOI: 10.1115/1.3143822. eprint: https://asmedigitalcollection.asme.org/dynamicsystems/article-pdf/109/1/65/5778350/65_1.pdf. URL: https://doi.org/10.1115/1.3143822.

[186] Han-Shue Tan Ching-Yao Chan. *Evaluation of Magnetic Markers as a Position Reference System for Ground Vehicle Guidance and Control*. Tech. rep. California Path Program, Institute of Transportation Studies, Mar. 2003.

[187] *Understanding the Current State of Vehicle Automation*. Tech. rep. Consumer Reports, 2021.

[188] M. Vidyasagar. *Control Systems Synthesis: A Factorization Approach*. Signal Processing, Optimization, and Control. MIT Press, 1988. ISBN: 9780262720120.

[189] J. Christian Gerdes Vivian Patterson Sarah Thornton. "Tire Modeling to Enable Model Predictive Control of Automated Vehicles From Standstill to the Limits of Handling". In: *14th International Symposium on Advanced Vehicle Control*. 2018.

[190] A Wischnewski et al. "Tube model predictive control for an autonomous race car". In: *Vehicle System Dynamics* 60.9 (2022), pp. 3151–3173.

[191] Alexander Wischnewski, Johannes Betz, and Boris Lohmann. "A Model-Free Algorithm to Safely Approach the Handling Limit of an Autonomous Racecar". In: *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*. 2019, pp. 1–6. DOI: 10.1109/ICCVE45908.2019.8965218.

[192] Alexander Wischnewski et al. "Indy autonomous challenge-autonomous race cars at the handling limits". In: *12th International Munich Chassis Symposium 2021: chassis. tech plus*. Springer. 2022, pp. 163–182.

[193] Alexander Wischnewski et al. "Vehicle Dynamics State Estimation and Localization for High Performance Race Cars". In: *IFAC-PapersOnLine* 52.8 (2019). 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019, pp. 154–161. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2019.08.064. URL: https://www.sciencedirect.com/science/article/pii/S2405896319303957.

[194] Wei Xie and Toshio Eisaka. “Design of LPV control systems based on Youla parameterisation”. In: *IEE Proceedings-Control Theory and Applications* 151.4 (2004), pp. 465–472.

[195] D. Youla, J. Bongiorno, and H. Jabr. “Modern Wiener–Hopf design of optimal controllers Part I: The single-input-output case”. In: *IEEE Transactions on Automatic Control* 21.1 (1976), pp. 3–13. DOI: 10.1109/TAC.1976.1101139.

[196] D. Youla, H. Jabr, and J. Bongiorno. “Modern Wiener-Hopf design of optimal controllers–Part II: The multivariable case”. In: *IEEE Transactions on Automatic Control* 21.3 (1976), pp. 319–338. DOI: 10.1109/TAC.1976.1101223.

[197] Dante C Youla and M Saito. “Interpolation with positive real functions”. In: *Journal of the Franklin Institute* 284.2 (1967), pp. 77–108.

[198] Luca Zaccarian and Andrew R Teel. “A common framework for anti-windup, bumpless transfer and reliable designs”. In: *Automatica* 38.10 (2002), pp. 1735–1744.

[199] Scott Zagorski et al. “Measured Vehicle Inertial Parameters - NHTSA’s Data through August 2020”. In: *SAE International Journal of Advances and Current Practices in Mobility* 3.4 (Apr. 2021), pp. 1535–1557. DOI: https://doi.org/10.4271/2021-01-0970. URL: https://doi.org/10.4271/2021-01-0970.

[200] A.T. Zaremba, M.K. Liubakka, and R.M. Stuntz. “Control and steering feel issues in the design of an electric power steering system”. In: *Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No.98CH36207)*. Vol. 1. 1998, 36–40 vol.1. DOI: 10.1109/ACC.1998.694623.

[201] Yuxiao Zhang et al. “Perception and sensing for autonomous vehicles under adverse weather conditions: A survey”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 196 (2023), pp. 146–177. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2022.12.021. URL: https://www.sciencedirect.com/science/article/pii/S0924271622003367.

[202] Kemin Zhou and John Comstock Doyle. *Essentials of robust control*. Vol. 104. Prentice hall Upper Saddle River, NJ, 1998.