Staying Inside the Lines: Vehicle Agnostic Path Following Using Cascaded Adaptive Control

by

William Bryan

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 12, 2020

Keywords: adaptive control, path following, lane keeping, model predictive control, vehicle dynamics

This thesis is dedicated to my wife, Rachel.

Abstract

This thesis presents a vehicle agnostic steering controller for path following. Many active safety systems, such as collision avoidance and lane centering, as well as all SAE Level 2+ autonomous vehicles, rely on a lateral controller to follow a desired path. The vehicle agnostic path following controller presented in this thesis is comprised of two adaptive controllers in a cascaded architecture, with the outer loop controlling the path dynamics and the inner loop controlling the vehicle dynamics. First, some commonly used tire models and lateral vehicle models are introduced. Next, a sensitivity analysis is performed on a variety of lateral controllers under the influence of model uncertainties. This analysis is used to develop the vehicle agnostic path following controller, which is tested in simulation at multiple velocities on a sedan, an SUV, a pickup truck, and a minivan. In simulation, the controller is able to adapt to each platform and achieve good lane keeping performance around a curvy track at a wide range of speeds. The controller is then implemented and validated in real-time on a Lincoln MKZ, a Class-8 Peterbilt 579 cab, and a Peterbilt 579 with a loaded trailer. During a double lane change maneuver, the vehicle agnostic path following controller maintains maximum path tracking errors of approximately half a meter with all three of these setups. The controller is also implemented on a $1/10^{th}$ scale RC car using a vision-based lane centering system to generate the reference path. Even on this scaled platform, the controller is able to follow the lane lines at multiple speeds, with maximum lookahead errors of 15 cm. Overall, the controller is shown to perform well on four simulated platforms and four experimental platforms at a range of longitudinal speeds, demonstrating the flexibility of the path following controller.

Acknowledgments

Many people have had a hand in my success as a graduate student in the Auburn GAVLAB, but none has had a greater impact than Dr. David Bevly. While his Systems students typically view him as a product of Mordor that they must defeat in order to graduate, I know this is not the case. Whether he is traveling seemingly every week to bring in contracts to support his students, lining students up with job interviews, or refusing to lower his standard because that would lessen Auburn's reputation, everything he does is to help his students succeed in their lives after school. He is our biggest advocate and it does not go unnoticed. I also owe a huge thanks to Dr. Scott Martin, who has always provided guidance when needed and teaches his students new ways to spell. Some of my favorite memories in the lab have been skiing in Utah with Dr. Martin and Dr. Bevly or talking about skiing in Utah over Friday lab lunches. Howard (Dr. Chen to some) has also been an invaluable resource in my time here. His class was excellent, but most importantly he was always down for a round of ping pong when I needed a mental break. All of these men have created a great learning environment with every tool at our disposal to perform cutting edge research.

I would like to thank Dr. George Flowers for his passion for teaching, helpfulness in his office hours, and willingness to be on my committee.

Top notch faculty is certainly important, but the students in the GAVLAB are what have made the last two years so great. I would like to thank Matt Boler for always being willing to take on crazy side projects with me and helping with the computer vision portion of this thesis. While experiments with Matt usually go horribly wrong, somehow everything always works out in the end. I would like to thank Jake Ward for his help in testing my controller on the trucks after "the accident" and his constant positive attitude. We all know some blind optimism is always needed when working on the trucks. I cannot leave out the fourth founding father of Autonomous Tiger Racing, Brendan Schretter. He is the go-to resource for sensor questions, Indy simulator setup, and chess puzzles. I would like to thank Amy Strong for editing my thesis, no matter how painful it was to read. I owe Amy infinite free meals. Robert Brothers provided me all sorts of controls and programming knowledge that helped me immensely in my research.

John David Sprunger was always willing to help me improve my code without being as mean as Matt. Scott Burchfield has been the comedic relief in my life for the last two years. I could always convince him to do something stupid on a snowboard or mountain bike with painful results. He is the hardest worker in the GAVLAB and should probably start sleeping more than three hours per night. Connor Jones is a great office mate and was happy to stay in a freezing cabin in Idaho at an abandoned drive in theater with Scott and I. Those sub-zero nights in a hot tub looking up at the Milky Way will not soon be forgotten. Josh Wood is always eager to talk about cars and trucks with me and was always happy to share his notes when I (occasionally) missed a class. Anderson Givhan is always ready to talk some Auburn basketball and is too smart for his own good. Drew Jennings never missed an opportunity to order Chipotle and jumped at the opportunity to adopt a stray cat I caught in my backyard.

These fellow GAVLABers are some of the kindest and most intelligent people I have ever met. All of them have helped me in some way with my research, but more importantly, they have become some of my closest friends. They are always down for a Friday board game night or random deep discussions about politics or the universe. I love all of you and know these friendships will last a lifetime.

There are also many friends outside of the lab that mean the world to me, but they will never read this so I am not going to list them here.

By far, my family has had the largest impact on my life. My brother, David, is my best friend, is always down for adventure, and I am proud to be your brother every day. Katie makes David a better person and I am so glad to finally have a sister. My mom and dad have always put us first, and I can never repay everything you have done for me. You always make me feel loved and have supported me at every step of my life. Making you proud is one of the strongest motivators in my life. Naomi, I know you cannot read because you are a dog, but you always welcomed me home with a wagging tail and are the best dog in the world. Lastly but most importantly, I must thank my wife, Rachel. You convinced me to go to grad school and put our plans on hold, and none of this would have been possible without you. Hopefully it was worth it. There are no words for how amazing and beautiful you are, but you give my life happiness and mean more to me than anything in this world.

"For I know the plans I have for you, declares the Lord, plans for welfare and not for evil, to give you a future and a hope." - Jeremiah 29:11

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ABS	Anti-lock Braking System
ACC	Adaptive Cruise Control
ADAS	Advanced Driver-Assistance Systems
AV	Autonomous Vehicle
CACC	Cooperative Adaptive Cruise Control
CAN	Controller Area Network
CG	Center of Gravity
CNN	Convolutional Neural Network
DARPA	Defense Advanced Research Projects Agency
DOF	Degrees of Freedom
DRTK	Dynamic base Real Time Kinematic
DSRC	Dedicated Short Range Communication
ECEF	Earth-Centered, Earth-Fixed
ECU	Electronic Control Unit
ESC	Electronic Stability Control
FOV	Field of View

GNSS	Global Navigation Satellite System
GNSS/INS	Global Navigation Satellite System/Inertial Navigation System
GPS	Global Positioning System
IAC	Indy Autonomous Challenge
IMU	Inertial Measurement Unit
ISO	International Organization for Standardization
LiDAR	Light Detection And Ranging
LKS	Lane Keeping System
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
NCAT	National Center for Asphalt Technology
NED	North-East-Down
NMPC	Nonlinear Model Predictive Control
NN	Neural Network
OEM	Original Equipment Manufacturer
PCHIP	Piecewise Cubic Hermite Polynomial
PID	Proportional Integral Derivative (Control)
RANSAC	Random Sample Consensus
RMSE	Root Mean Square Error
ROS	Robotic Operating System
RTK	Real Time Kinematic

SAE	Society of Automotive Engineers
SIL	Software In the Loop
TDCP	Time-Differenced Carrier Phase
V2V	Vehicle-to-Vehicle
WGS84	World Geodetic System

Chapter 1

Introduction

1.1 Motivation

Automobiles are an integral part of everyday life for most Americans. In the United States, automobiles outnumber the number of driving age adults [1, 2]. As beneficial and indispensable as motor vehicles are, they also have brought a significant cost of life. Over the last 20 years, an average of over 38,000 people per year have died from motor vehicle crashes in the United States [3, 4]. Multiple studies have associated over 90% of these crashes as the result of human driving error [5].

Advances in vehicle autonomy promise a safer future on the roads. Passive safety systems such as airbags, seat belts, and crumple zones have worked to save countless lives but do nothing to prevent the accident. Active safety systems, on the other hand, attempt to prevent accidents from ever occurring. Advanced Driver-Assistance Systems (ADAS) include features on vehicles such as adaptive cruise control (ACC), lane keeping systems (LKS), electronic stability control (ESC), anti-lock brake systems (ABS), and more. These systems have been estimated to have saved over 600,000 lives from the year 1960 to 2012 [6]. Electronic stability control alone has been able to greatly reduce the number of fatal single car accidents [7]. The result of vehicle safety systems can be seen in Figure 1.1, where the number of deaths per mile driven has decreased greatly with the addition of passive and active safety systems [8].

Many of these ADAS systems qualify as low levels of vehicle autonomy and provide the foundation for full autonomous driving. The increasing availability of cheaper sensors, actuators, and powerful embedded computing platforms over the last ten years has lead to the

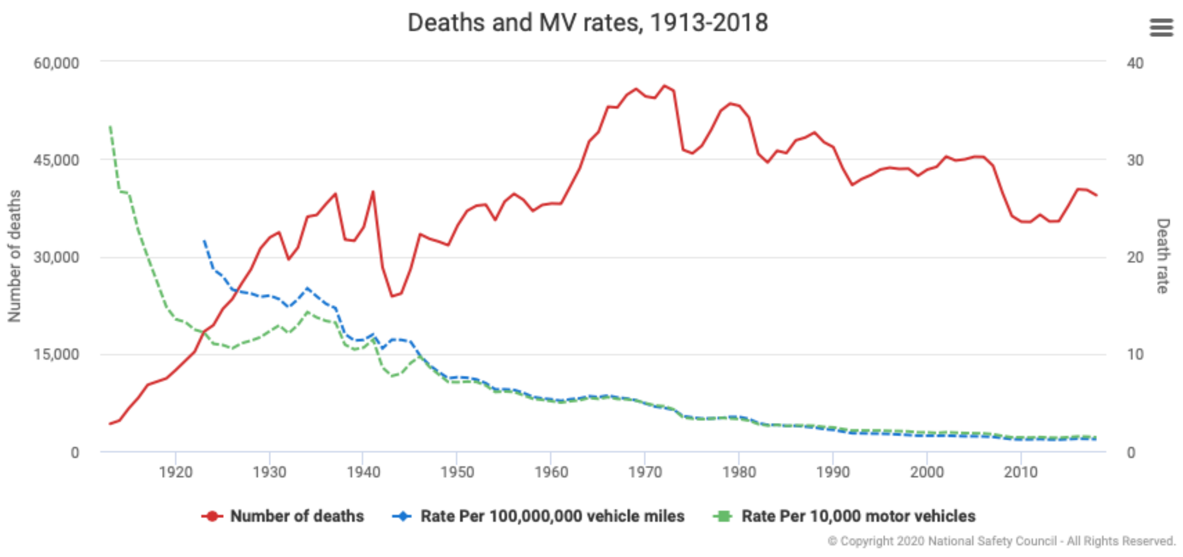


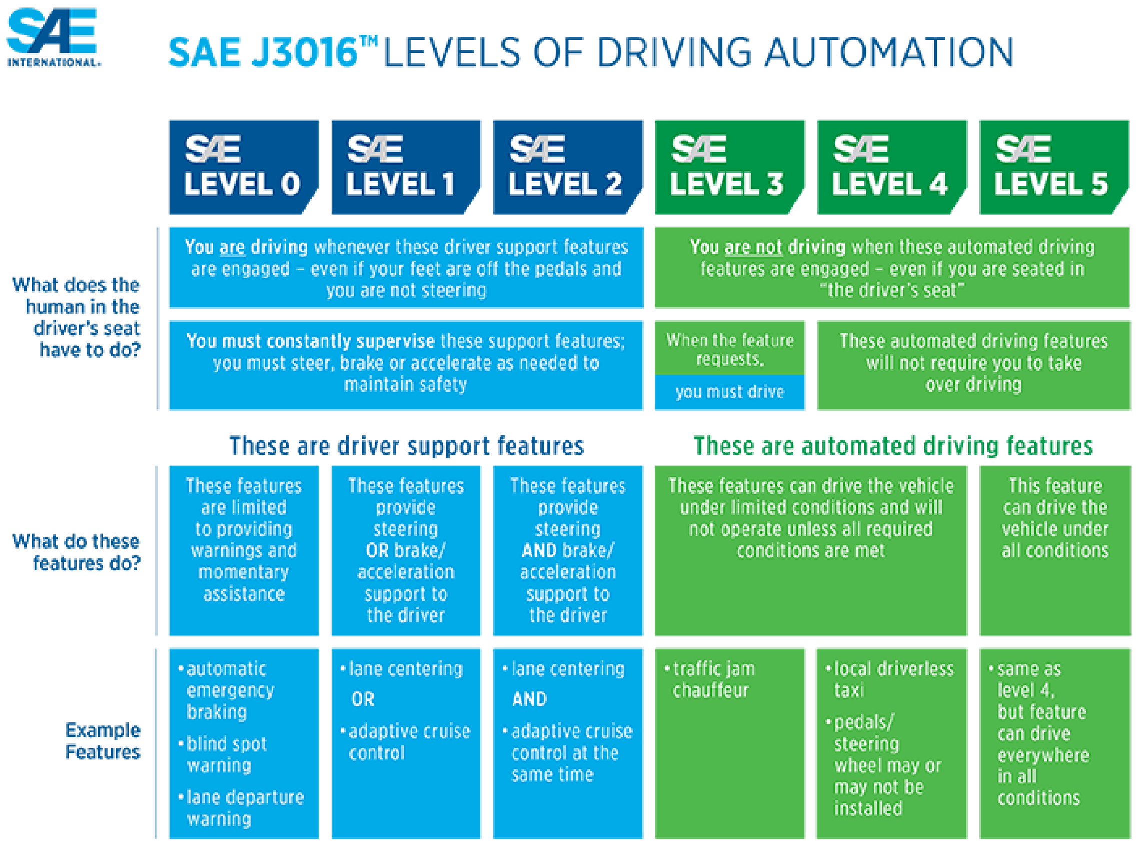
Figure 1.1: United States Motor Vehicle Fatalities.
Used with permission from the NSC [8].

adoption of automated driving features in almost every new car sold today. Furthermore, every major car manufacturer has put a huge emphasis and investment into achieving high levels of autonomy in the near future [9–16].

While safety is the strongest motivator for vehicle autonomy, there are numerous secondary benefits. Autonomous vehicles can increase the efficiency and reduce the costs for the trucking industry. They have the ability to reduce traffic greatly, allow people to work during their commutes, improve parking, and many more benefits [17]. The combination of these and the safety benefits has sparked the autonomous vehicle (AV) industry in the last 15 years, including many fast-growing startups.

The Society of Automotive Engineers (SAE) defines six levels of vehicle autonomy [18]. Level 0 is where the human driver is performing almost all driving tasks. Systems that provide warnings or emergency braking fall into this category. Most of the convenience and safety features found on new cars today fall into Level 1 and 2. These systems require the driver to maintain focus on driving, but the vehicle handles some parts of the driving task itself, such as lane centering or adaptive cruise control. Level 3 takes this a step further to where the driver is fully disengaged from the operation of the vehicle but must be ready to drive when requested.

Lastly, levels 4 and 5 are what most people consider driverless vehicles, where the human is just a passenger. These levels are explained in more detail in Figure 1.2.


SAE J3016™ LEVELS OF DRIVING AUTOMATION

	SAE LEVEL 0	SAE LEVEL 1	SAE LEVEL 2	SAE LEVEL 3	SAE LEVEL 4	SAE LEVEL 5
What does the human in the driver's seat have to do?	You are driving whenever these driver support features are engaged – even if your feet are off the pedals and you are not steering			You are not driving when these automated driving features are engaged – even if you are seated in "the driver's seat"		
	You must constantly supervise these support features; you must steer, brake or accelerate as needed to maintain safety			When the feature requests, you must drive	These automated driving features will not require you to take over driving	
	These are driver support features			These are automated driving features		
What do these features do?	These features are limited to providing warnings and momentary assistance	These features provide steering OR brake/ acceleration support to the driver	These features provide steering AND brake/ acceleration support to the driver	These features can drive the vehicle under limited conditions and will not operate unless all required conditions are met		This feature can drive the vehicle under all conditions
Example Features	• automatic emergency braking • blind spot warning • lane departure warning	• lane centering OR • adaptive cruise control	• lane centering AND • adaptive cruise control at the same time	• traffic jam chauffeur	• local driverless taxi • pedals/ steering wheel may or may not be installed	• same as level 4, but feature can drive everywhere in all conditions

Figure 1.2: SAE Levels of Automation [18].

This thesis focuses on robust path following for an autonomous driving. Path following is the ability of a vehicle to follow a given reference path, such as the center of a lane or a specific route between two locations. Path following control is an integral component of any level 2 or higher autonomous system. The path following controllers are also known as steering controllers because they calculate the required steering commands for a vehicle to follow the reference path. In a lane keeping system, the steering controller uses inputs from a camera or other sensors to make the vehicle follow the center of the lane. Steering controllers are also needed for any waypoint path following, such as in Level 2 class-8 truck platooning [19] or automated farming [20]. For this work, the driver is assumed to be completely hands-off and the steering done solely by the controller such as in a Level 2 or higher system.

1.2 Related Work

A wide range of architectures for steering control have been used for lane keeping and path following. Simple open loop controllers relying only on the vehicle kinematics have been used successfully [21], but do not incorporate any state feedback. Using a kinematic based steering controller, Stanford won the 2005 DARPA grand challenge (Figure 1.3) [22]. Proportional integral derivative (PID) controllers include feedback on the desired states, but require time intensive tuning [23, 24]. Dynamic model inversion has also been used to create a higher performance controller without as much tuning, but this requires thorough knowledge of the vehicle model [25].

Figure 1.3: Stanley, the winner of the 2005 DARPA Grand Challenge [22].

More complex controllers exist that achieve better tracking performance through using lookahead error for the feedback, instead of the error at the vehicle body [26]. The lookahead distance provides damping to the system and is one of the most commonly used methods. However, the lookahead distance must be selected carefully or unwanted behavior including oscillations can occur [27]. This is dependent on the longitudinal velocity as well as other parameters.

Performance can also be improved by using model predictive control (MPC). MPC predicts the vehicle response into the future and optimizes the inputs over a time horizon to achieve the desired output [28, 29]. The first input command is sent and the process is then repeated at each time step. MPC can incorporate hard and soft constraints from the actual system [30]. The complexity in solving the optimization relegated MPC to industrial applications until recently, when more powerful computing platforms became available. Nonlinear model predictive control (NMPC) can increase the fidelity of the prediction even more, but at the cost of computation time for the optimization [31, 32].

The above methods all require an accurate vehicle model, but often the model parameters are extremely difficult to estimate. In addition, unmodeled disturbances from road bank, wind, and other external forces can cause performance and stability guarantees to be violated. One proposed way of guaranteeing performance in the face of model inaccuracy is to design the control law around the ”center of percussion” where the rear lateral forces are eliminated [33]. This was shown to perform well even under very dynamic maneuvers. Another robust method is to incorporate the quadratic stabilization technique and input constraints. This allows a four wheel steer vehicle to limit wheel slip and have good lane keeping performance even under the disturbances from air drag and road curvature [34].

A different approach for control at the limit of handling was to prescribe a desired velocity vector at the center of gravity at each time step. The vector field created is then analyzed in the context of limited friction and acceleration in order to determine a constrained vector. This is then used as the reference to the controller. This method was able to show asymptotic stability even at the limits of tire grip [35].

All of these robust methods were able to maintain performance and stability during limit handling and while under the influence of unmodeled disturbances. However, they still rely on a fairly accurate vehicle model without too much uncertainty. On the other hand, adaptive controllers are designed to handle very large amounts of model uncertainty while still maintaining stability. This uncertainty can be from many sources such as nonlinearities, time-varying models, coupled dynamics, unknown disturbances, and measurement noise. Adaptive controllers typically change the controller gains based on system variations or parameter estimation algorithms. They have been used extensively in aerospace applications, especially where the system dynamics can change rapidly and there is constant excitation.

A very simple adaptive controller, where the lookahead distance is adapted based on the vehicle model and longitudinal speed, was able to show improved path tracking abilities under a wide range of speeds [36]. While this method could handle model variation due to velocity differences, it still assumed that the other vehicle parameters were known well. An adaptive approach using a self tuning regulator was shown to work well when the vehicle parameters were unknown but assumed to belong to a compact known set [37]. This works as long as one can bound the vehicle parameters with certainty, but breaks down otherwise.

Most adaptive control also requires persistent excitation in order for the controller to converge asymptotically. Some classical modifications that are done to adaptive control in order to improve its robustness such as σ-modification [38], e-modification [39], and a projection operator [40]. These ensure the weight matrix used to calculate the adaptive gains is bounded even when there is unstructured uncertainty. Another way of guaranteeing stability with unmodeled dynamics is using model reference adaptive control with added robustness terms as described in [41]. These controllers add stability in the presence of unmodeled dynamics and lack of excitation, but in doing so relax the performance.

An alternative method to handle the lack of persistent excitation is by using concurrent-learning adaptive control, which restricts the weight updates so that the response to current data is not outweighed by stored data [42]. If the uncertainty can be modeled as a combination of nonlinear bases, radial basis function neural networks can be effectively used as the adaptive elements. This method is able to guarantee exponential convergence of the neural network parameters to values very close to the ideal values, without requiring persistent excitation [43].

Even the adaptive controllers that handle model uncertainty rely on an estimate of the dynamic model. This is feasible if the vehicle type is not changed drastically, but this thesis proposes using the principles of adaptive control to design a steering controller that requires no vehicle model information. This controller will also be vehicle agnostic, able to handle a wide range of ground vehicle types without re-tuning. This will be a significant contribution considering the time and expense associated with tuning lateral controllers.

1.3 Research Contributions

A list of the contributions of this thesis are shown below.

- A survey of commonly used vehicle and tire models
- A sensitivity analysis on a range of lateral vehicle controllers to determine which methods are more affected by model uncertainty
- A new control architecture that is vehicle independent with good path tracking performance
- A real-time experimental implementation of the vehicle agnostic path following controller and results from RTK-GPS path following and vision-based lane keeping on a wide range of vehicle types

1.4 Thesis Outline

This chapter provided the motivation and background of this research. Chapter 2 goes through the coordinate frames, tire models, and vehicle models used in this thesis. Chapter 3 conducts a sensitivity analysis of five lateral controller types by measuring each of their performance under model parameter uncertainty. Chapter 5 presents the vehicle agnostic path following controller formulation and architecture. Chapter 6 discusses the simulation and experimental setup. It also describes the path generation methods for the various experiments. Then, the results of the simulation and real-world experiments are given. Lastly, chapter 6 provides some conclusions and insights that were made from this research.

Chapter 2

Ground Vehicle and Tire Modeling

For safe design and control of autonomous vehicles, most of the initial testing and proof of concept must be done in simulation. Simulation environments reflect aspects of the real-world environment without the safety, financial, and legal risks involved in experimental testing. A simulation requires a model of the vehicle, whether formulated empirically or derived theoretically based on the vehicle's physical parameters. Furthermore, model-based controllers typically need an accurate model to guarantee performance and stability.

Generally, two types of models are used for most path following control: a lateral vehicle model and a tire model. The lateral model represents the motion of the vehicle due to the external forces. A tire model is another integral part of formulating the vehicle dynamics. As the only interface between the ground and the vehicle, the majority of the lateral and longitudinal forces are transferred through the tires. The following section will discuss the coordinate frames used to define the vehicle and its environment. The next section introduces a few commonly employed tire models. Lastly, models of the vehicle itself will be presented in an order of increasing complexity.

2.1 Coordinate Frames

Autonomous vehicles require a frame of reference to understand their motion within the environment around them. This section defines a set of coordinate frames and associated notation that will be used throughout the remainder of this thesis. All of these coordinate systems are three-dimensional Cartesian systems using the right-handed convention. Note, however, many non-Cartesian frames are also used in practice, such as the geodetic system (latitude-longitude-altitude). The primary coordinate systems used in this thesis are the vehicle frame, the tire frame, the navigation frame, the global frame, and the camera frame.

2.1.1 Vehicle Frame

The vehicle frame used follows the standards established by SAE J670 [44] and ISO 8855 [45]. This system is body-fixed with the origin at the vehicle center of gravity (CG), the x-axis extends out the front of the vehicle, the y-axis comes out the passenger side, and the z-axis down out the floor pan. This can be seen in Figure 2.1 with the x, y, and z axes in blue, green, and red respectively.

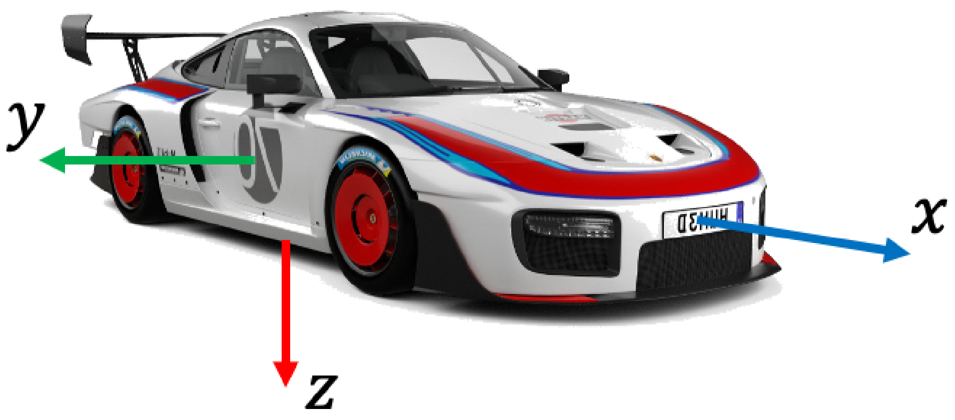


Figure 2.1: Vehicle-Fixed Coordinate System.

Vehicle model from [46].

Rotation angles are also defined based on these axes and are defined as positive when in the counterclockwise direction according to the right hand rule. A rotation about the x-axis is referred to as roll and is denoted by ϕ. A rotation about the y-axis is referred to as pitch and is denoted by θ. A rotation about the z-axis is referred to as yaw or heading, is denoted by ψ, and its derivative is denoted as $\dot{\psi}$ or r. Note that this follows the commonly used Euler angle notation.

The vehicle frame is used to derive the vehicle models presented in Section 2.3. The terms longitudinal and lateral refer to the x and y vehicle axes, respectively. Additionally, reference path inputs for control systems are frequently expressed in the vehicle frame, making the control formulations simpler.

2.1.2 Tire Frame

A tire coordinate system is also instrumental to the derivation of a vehicle model, tire model, and controller of a vehicle. The origin of the tire frame is fixed to the center of the wheel. The x-axis extends forward out the front of the tire parallel to the ground, and the y-axis extends to the right out of the wheel center. The front wheels are rotated around the tire z-axis as the vehicle is steered. The steering angle is the angle between the vehicle x-axis and the tire x-axis and will be referred to as δ in the following sections. Figure 2.2 shows how the tire frame is oriented relative to the vehicle and the navigation frame.

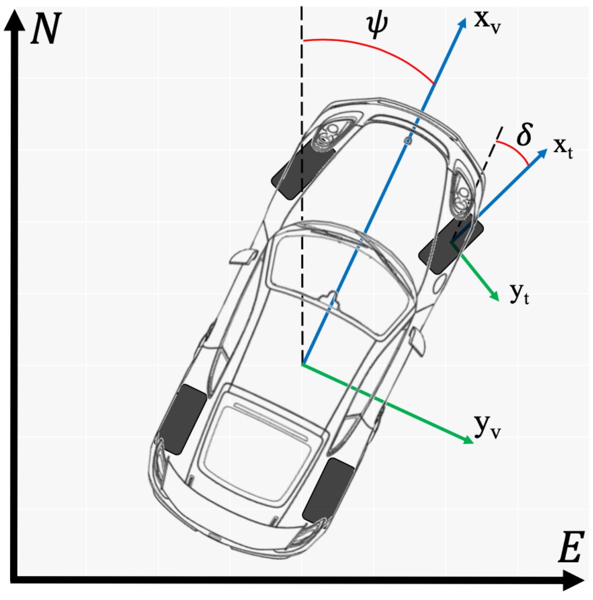


Figure 2.2: Vehicle and Tire Coordinate Frame Relative to NED.

2.1.3 Navigation Frame

The vehicle's motion and position is usually measured within a local navigation frame. This thesis employs the North-East-Down (NED) convention traditionally used in aerospace and control applications. The NED coordinate system defines a tangent plane along the surface of the earth aligned with the North and East directions. Figure 2.2 shows the vehicle within the NED frame. Heading is defined as the angle between the navigation frame N-axis and the

x-axis of the vehicle. Often the origin of the NED frame is aligned with the initial position of the vehicle and the $North$-axis is aligned with the initial heading instead of true north.

2.1.4 Global Frame

When the vehicle is travelling a long distance, the NED frame is insufficient to measure the path of the vehicle due to the curvature of the earth. For these applications and for systems relying on GPS measurements, the Earth-Centered, Earth-Fixed (ECEF) coordinate system is used [47]. This is officially defined by WGS84 as the origin at the earth's center of gravity, the Z-axis out the North Pole, the X-axis out the intersection of the Prime Meridian and Equator, and the Y-axis 90° East also along the Equator. This is depicted in Figure 2.3 as well as a NED frame relative to the ECEF frame. By knowing just the origin of the NED frame in ECEF coordinates and the initial heading, simple transformations between the two frames can be made. This is particularly useful in converting GPS positions and velocities into the navigation frame.

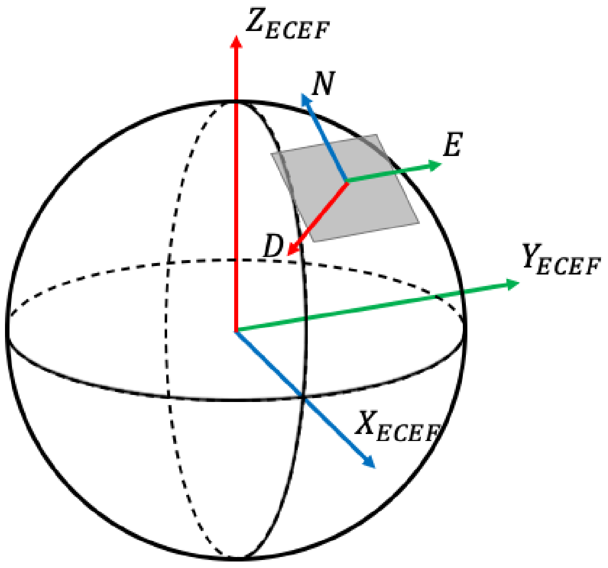


Figure 2.3: North-East-Down and Earth-Centered, Earth-Fixed Coordinate Systems.

2.1.5 Camera Frame

Chapter 5 presents a camera-based path generation system. This system employs a camera that is rigidly fixed to the roof of the car and used the modern camera coordinate system shown in

Figure 2.4. This coordinate system defines the Z-axis as forward out the center of the camera, the Y-axis as down, and the X-axis to the right. To transform between the camera frame and the vehicle frame, the lever arm and orientation between the two must be known.

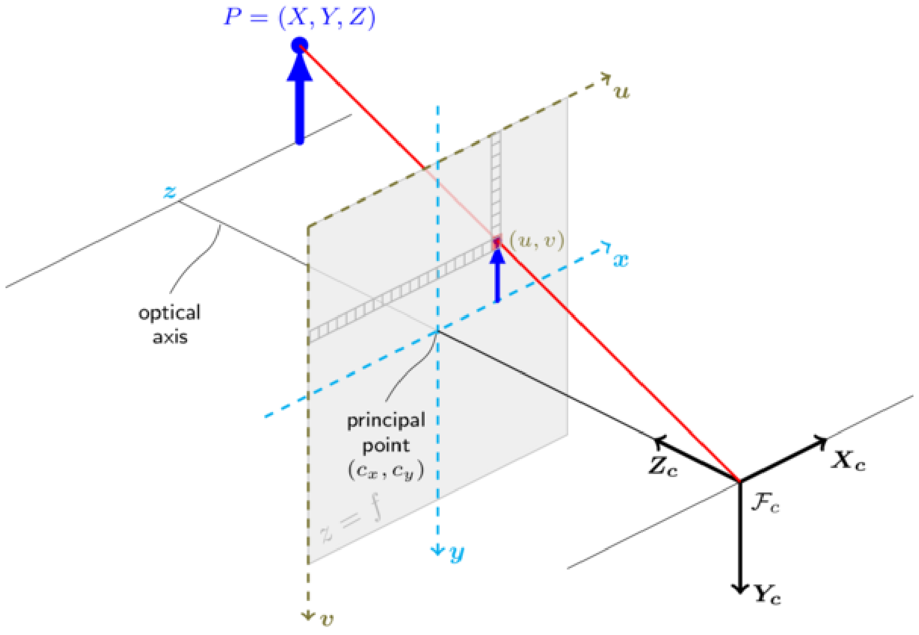


Figure 2.4: Camera Coordinate System.

Figure from OpenCV [48].

2.2 Tire Models

Tires are one of the most important components of a vehicle because they transmit all major forces to the vehicle except air drag and the force due to gravity. Figure 2.5 shows the external forces transmitted to the vehicle through the tires. Note that the smaller friction forces are lumped together here into the rolling resistance term, F_{rr}. F_x controls the longitudinal motion of the car along the driving surface and is comprised of the thrust force to the driven tires and the breaking force to all tires. The lateral forces to all tires, F_y, determine the path the vehicle follows and is therefore the most important to the following chapters. The normal force to each tire, F_N, determines the ride quality and also affects the available traction for F_x and F_y. Additionally, each tire has only a very small contact area to transmit these interdependent forces. Therefore, an accurate tire model is instrumental to useful simulation and higher dynamic control.

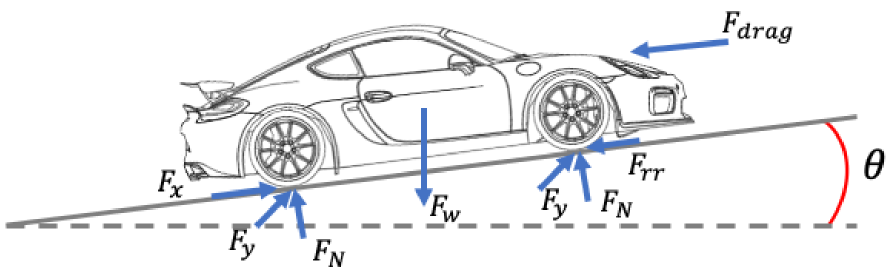


Figure 2.5: External Forces Acting on a Car.

Tires generate their lateral and longitudinal tire forces through slip. Slip is the relative motion between the tire contact patch and the driving surface. The amount of force that is generated by a tire is a function of its slip, which changes with a given load, pressure, and other tire parameters. Since this thesis focuses on the path following of a vehicle, the tire side-slip angle (α) will be the most important for tire modeling as it is needed to accurately depict lateral force acting on the vehicle. Side slip angle is defined as the angle between the tire's direction of motion and its x-axis as defined in Section 2.1.2. The tire models discussed in the next few sections are not analytical models, but common semi-empirical fits based on testing data.

2.2.1 Linear Tire Model

For low slip situations, a linear tire model is an extremely simple but accurate tire representation. The linear tire model defines the lateral force of the tire as the tire cornering stiffness parameter (denoted C_α) multiplied by the tire slip angle (α) as shown in Equation (2.1).

$$F_y(\alpha) = -C_\alpha \cdot \alpha \tag{2.1}$$

Tire cornering stiffness is affected by a variety of factors such as tire size, construction, width, tread pattern, inflation pressure, and load. Of these, load and inflation pressure have the most dramatic effect [49].

The more complex models described below also contain this linear region, however the linear model becomes inaccurate a higher degrees of slip because the tire eventually reaches a peak force threshold at a limited slip value. This can still be approximated simply by adding a saturation limit to the linear model as shown in Equation (2.2).

$$F_y(\alpha) = \begin{cases} -C_\alpha * \alpha, & \text{if } \alpha < \alpha_{max\,force} \\ F_{y\,max}, & \text{otherwise} \end{cases} \tag{2.2}$$

The linear model and the linear model with saturation limits are shown below in Figure 2.6 with tire force on the y-axis and slip angle on the x-axis.

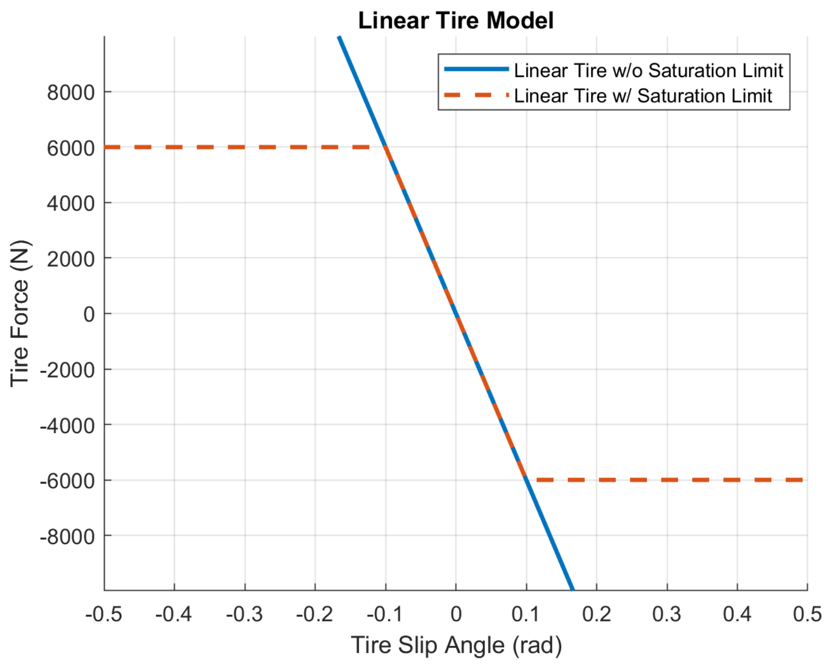


Figure 2.6: Linear Tire Model.

2.2.2 Brush-Fiala Model

A non-linear tire model can increase the fidelity of the overall vehicle model. The Brush-Fiala tire model is a simple non-linear model that was analytically derived but is easy to fit to tire system identification data because it still only relies on two parameters, C_α and μ [50]. C_α is the same tire cornering stiffness as the linear tire model and μ is a friction parameter that can be tweaked to match the peak force. Equation (2.3) shows the mathematical representation of the Brush-Fiala model.

$$F_y(\alpha) = \begin{cases} -C_\alpha \tan(\alpha) + \frac{C_\alpha^2}{3\mu F_z}|\tan(\alpha)|\tan(\alpha) & \\ -\frac{C_\alpha^3}{27\mu F_z^2}\tan^3(\alpha), & \text{if } |\alpha| < \arctan\left(\frac{2\mu F_z}{C_{alpha}}\right) \\ -\mu F_z sgn(\alpha), & \text{otherwise} \end{cases} \tag{2.3}$$

An example Brush-Fiala tire curve can be seen in Figure 2.7. Chapter 3 uses the Brush-Fiala model as the nonlinear tire model for the controller sensitivity analysis.

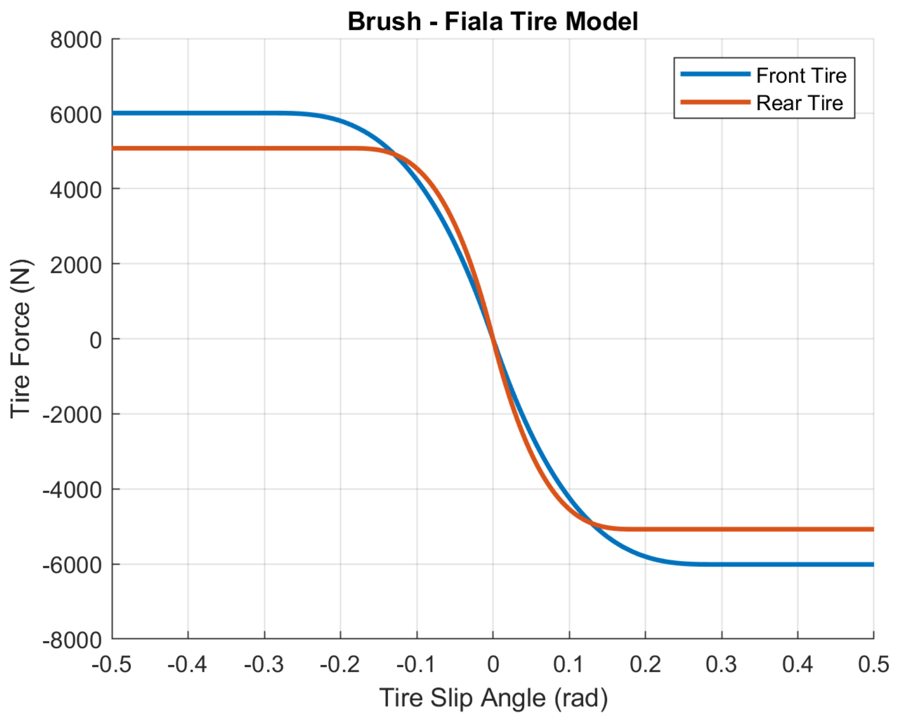


Figure 2.7: Brush-Fiala Tire Model.

2.2.3 Pacejka Magic Model

The ”Magic Formula” tire model is the name for several versions of a semi-empirical tire model created by Pacejka and Bakker [51–53]. Since its inception, it has been considered the gold standard for vehicle dynamicists, race teams, video game developers, and automotive manufacturers. Equations (2.4-2.6) below show the general formulation of this model [49],

$$y = D\sin\left[C\arctan\left(Bx - E\left(Bx - \arctan Bx\right)\right)\right] \tag{2.4}$$

$$Y(X) = y(x) + S_V \tag{2.5}$$

$$x = X + S_H \tag{2.6}$$

where

Y :	output variable F_x or F_y
X :	input variable $tan(\alpha)$ or κ

and

B :	stiffness factor
C :	shape factor
D :	peak value
E :	curvature factor
S_H :	horizontal shift
S_V :	vertical shift.

Figure 2.8 shows the Pacejka tire model plotted for various load conditions but identical tire parameters. While this model is considered to be the most accurate of those discussed, its complexity is the primary drawback. The number of parameters and the difficulty in intuitively estimating them makes achieving a good empirical fit more challenging. Because of this, the Pacejka Magic Model was not used in this thesis.

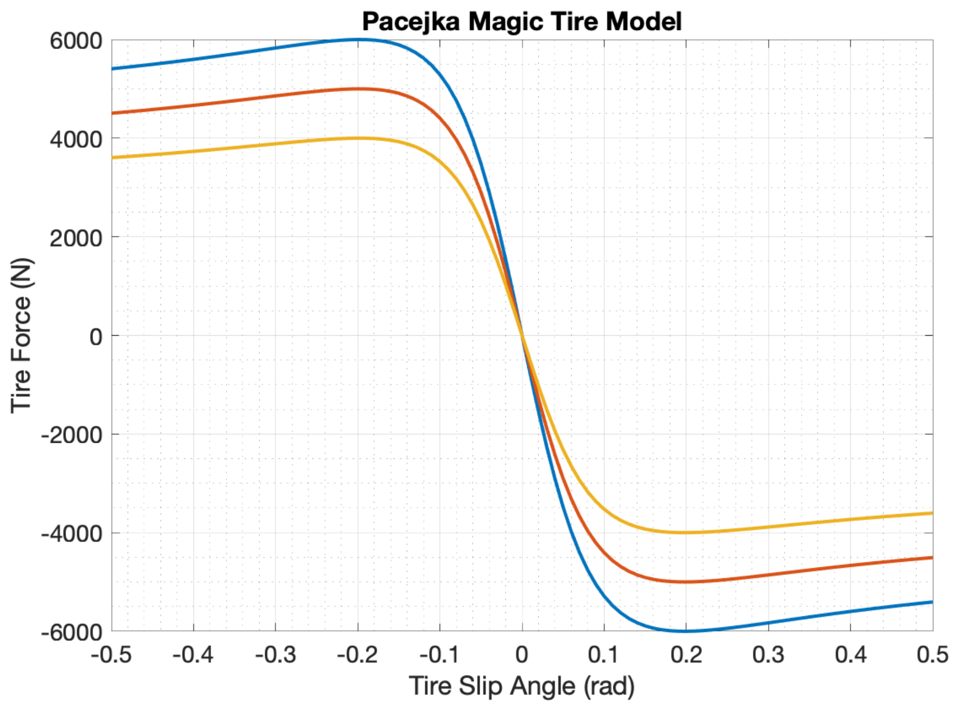


Figure 2.8: Pacejka Magic Tire Model.

2.3 Ground Vehicle Models

This section contains some commonly used vehicle models, increasing in complexity from a simple kinematic model up to dynamic models with many degrees of freedom (DOF). While the more complex models are great for producing high fidelity simulations and controllers, the simpler models are often employed for control and other instances when computation time is a major factor such as real-time implementations. A mixture of both kinematic and dynamic vehicle models will be used in the following chapters, depending on the needs of each application. Note that all of the employed models constrain the vehicle to planar motion, disregarding the effects of pitch, roll, and other weight transfer.

2.3.1 Kinematic Bicycle Model

The most commonly used model accumulates the forces on each axle into a single tire representing the left and right tire, neglecting the track width of the vehicle. This is known as the bicycle model and is the basis for the derivation of the kinematic model in this section, the understeer gradient model in the next section, and the dynamic bicycle model in Section 2.3.3. The bicycle model is shown in Figure 2.9.

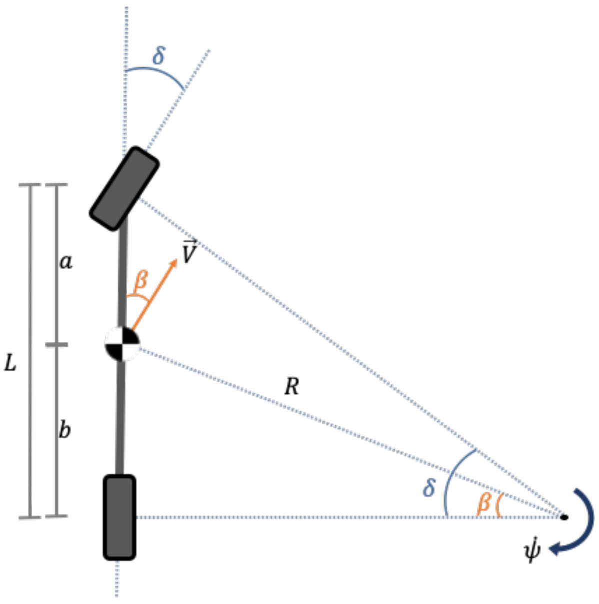


Figure 2.9: Kinematic Bicycle Model.

When the four wheel vehicle model is reduced into the bicycle model, the steering angle is assumed to be the average of the left and right steer angles. Due to rigid body assumptions, the lateral velocities at each axle are unaffected by using the bicycle model. In order to derive the models in the next three sections, some additional variables must be defined. The wheelbase length (L) is the sum of the distance from the CG to the front axle (a) and the distance from the CG to the rear axle (b). If R is the radius from the CG to the point that the vehicle is driving around, the geometry shows that at steady state, the yaw rate is equal to the magnitude of the velocity divided by R as shown in Equation (2.7).

$$\dot{\psi} = \frac{|\vec{V}|}{R} \tag{2.7}$$

Since the sideslip angle can be related to R in Equation (2.8) as well as steering angle (δ) and wheelbase (L) in Equation (2.9), the turning radius can be eliminated from Equation (2.7) [54].

$$\sin(\beta) = \frac{b}{R} \tag{2.8}$$

$$\beta = \arctan\left(\frac{b\tan(\delta)}{L}\right) \tag{2.9}$$

$$\dot{\psi} = \frac{|\vec{V}|\sin(\beta)}{b} \tag{2.10}$$

$$\dot{\psi} = \frac{|\vec{V}|\sin\left(\arctan\left(\frac{b\tan(\delta)}{L}\right)\right)}{b} \tag{2.11}$$

This model can be used in this form or simplified even further if the small angle approximation is applied to the steering angle shown in Figure 2.9. The small angle approximation states that $\sin\theta \approx \tan\theta \approx \theta$ and $\cos\theta \approx 1$ for $\theta < 20°$. This a fair assumption considering the steering angle rarely exceeds $20°$ when driving above extremely slow speeds, resulting in the following relationship for yaw rate (2.12).

$$\dot{\psi} = \frac{|\vec{V}|}{L}\tan(\delta) \qquad \Longrightarrow \qquad \dot{\psi} = \frac{|\vec{V}|}{L}\delta \tag{2.12}$$

When driving at higher speeds or under transient maneuvers, the kinematic bicycle model is less accurate than the models discussed throughout the rest of the chapter. Nevertheless, the simplicity and ease of determining wheelbase length makes the kinematic model a strong candidate for many real-world implementations.

2.3.2 Understeer Gradient Model

In order to derive a more accurate steady state model, individual slip angles, α, are defined for each axle as shown in Figure 2.10. This better matches real vehicles because it allows each axle to slip different amounts and tires must have slip to generate force as shown in the tire models section previously.

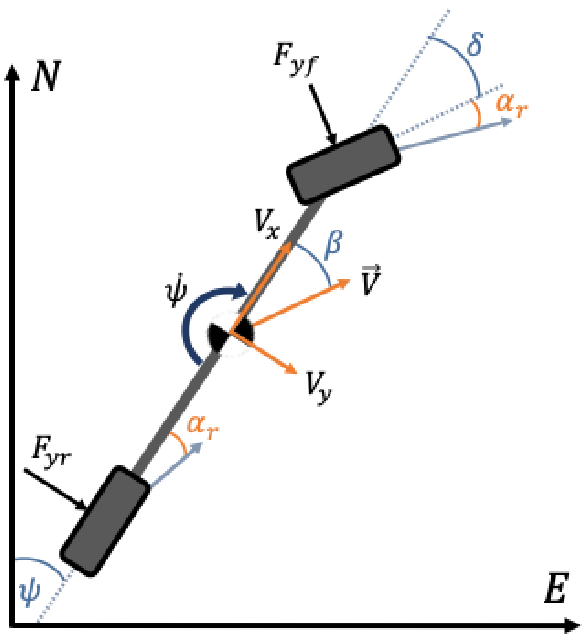


Figure 2.10: Dynamic Bicycle Model.

The forces and moments can be summed about the vehicle center of gravity. These equations can then be simplified through the assumption of steady state cornering ($\ddot{\psi} = \dot{V}_y = 0$).

$$\sum F_y = m\left(\frac{V_x^2}{R} + \dot{V}_y\right) \implies F_{yf}\cos\delta + F_{yr} = m\left(\frac{V_x^2}{R} + \dot{V}_y\right) = m\left(\frac{V_x^2}{R}\right) \tag{2.13}$$

$$\sum M_z = I_{zz}\ddot{\psi} \implies aF_{yf}\cos\delta - bF_{yr} = I_{zz}\ddot{\psi} = 0 \tag{2.14}$$

Using the vehicle's geometry, expressions for the slip angles can be formulated as shown in Equations (2.15-2.16).

$$\alpha_f = \arctan\left(\frac{V_y + a\dot{\psi}}{V_x}\right) - \delta \tag{2.15}$$

$$\alpha_r = \arctan\left(\frac{V_y - b\dot{\psi}}{V_x}\right) \tag{2.16}$$

Again, a small angle approximation will simplify the model without greatly reducing its accuracy because slip angles typically stay very small in most driving scenarios.

$$\alpha_f = \left(\frac{V_y + a\dot{\psi}}{V_x}\right) - \delta \tag{2.17}$$

$$\alpha_r = \left(\frac{V_y - b\dot{\psi}}{V_x}\right) \tag{2.18}$$

Equations (2.17-2.18) can be solved for V_y and set equal to each other to yield a single equation below.

$$\delta + \alpha_f - \frac{a\dot{\psi}}{V_x} = \alpha_r + \frac{b\dot{\psi}}{V_x} \tag{2.19}$$

$$\delta = \frac{L\dot{\psi}}{V_x} + \alpha_r - \alpha_f \tag{2.20}$$

Finally, since $V_x \approx R\dot{\psi}$, the steer angle equation can be reduced even further.

$$\delta = \frac{L}{R} + \alpha_r - \alpha_f \tag{2.21}$$

This model (2.21) can be used directly with a nonlinear tire model. However, using the linear tire model, steering angle can be directly related to lateral acceleration as shown in Equation (2.22).

$$\delta = \frac{L}{R} + \left(\frac{mbV_x^2}{LC_{\alpha f}R} - \frac{maV_x^2}{LC_{\alpha r}R}\right) \tag{2.22}$$

Since steady state cornering was assumed ($\dot{V}_x = \dot{V}_y = 0$), the only lateral acceleration is from the centripetal component which simplifies the expression further. First, W_f is defined as weight on the front axle, W_r as weight on the rear axle, and a_y as the lateral acceleration.

$$\frac{mb}{L} = W_f, \qquad \frac{ma}{L} = W_r \qquad \frac{V_x^2}{R} = a_y \tag{2.23}$$

These can be substituted into Equation (2.22) to get Equation (2.24).

$$\delta = \frac{L}{R} + \left(\frac{W_f}{C_{\alpha f}} - \frac{W_r}{C_{\alpha r}}\right) a_y \tag{2.24}$$

The term understeer gradient, K_{us}, is defined as the ratio of the weight on each axle to the tire cornering stiffness.

$$K_{us} = \left(\frac{W_f}{C_{\alpha f}} - \frac{W_r}{C_{\alpha r}}\right) \tag{2.25}$$

The understeer gradient is often defined in the units of $\frac{deg}{g_{accel}}$ or $\frac{rad}{g_{accel}}$. Equation (2.24) can be simplified further using the understeer gradient.

$$\delta = \frac{L}{R} + K_{us} a_y \tag{2.26}$$

Using Equation (2.7), Equation (2.26) can be solved for yaw rate which yields the following equation.

$$\dot{\psi} = \frac{V}{L + K_{us} V^2} \delta \tag{2.27}$$

A positive understeer gradient ($K_{us} > 0$) implies that a vehicle will have more front tire slip in a corner and will have to apply increasingly more steering than the simple kinematic model shows as the lateral accelerations increase. This is referred to as an understeer vehicle. Inversely, a negative understeer gradient ($K_{us} < 0$) implies that a vehicle will have more rear tire slip in a corner and will have to apply less steering than predicted in the kinematic model as the lateral accelerations increase. This is called an oversteer vehicle.

While this model is more complex than the kinematic model, it still makes many assumptions such as steady state cornering, using the linear tire model, and the small angle approximation. However, constant radius or constant velocity tests can be run to empirically determine the understeer gradient of a vehicle without needing to know the values of tire cornering stiffness and CG location, making it a desirable choice of model for many applications.

2.3.3 Dynamic Bicycle Model

The linear dynamic bicycle model removes the assumption of steady state cornering and is one of the most widely used models [55]. It is very accurate under most circumstances including transient maneuvers when the vehicle parameters are well known. Again, the formulation begins with the summing of the forces and moments about the z-axis in Figure 2.10.

$$aF_{yf}\cos\delta - bF_{yr} = I_{zz}\ddot{\psi} \tag{2.28}$$

$$F_{yf}\cos\delta + F_{yr} = m\left(\frac{V_x^2}{R} + \dot{V}_y\right) \tag{2.29}$$

The linear tire model (2.1) allows the forces to be converted into slip angles.

$$I_{zz}\ddot{\psi} = -aC_{\alpha f}\alpha_f\cos\delta + bC_{\alpha r}\alpha_r \tag{2.30}$$

$$m\left(\frac{V_x^2}{R} + \dot{V}_y\right) = -C_{\alpha f}\alpha_f\cos\delta + -C_{\alpha r}\alpha_r \tag{2.31}$$

Equations (2.17-2.18) are then used so that the model is in terms of body velocities and rotation rate. The small angle approximation asserts that $\cos\delta \approx 1$ for angles under $20°$ resulting in the following equations.

$$I_{zz}\ddot{\psi} = -aC_{\alpha f}\left(\frac{V_y + a\dot{\psi}}{V_x} - \delta\right) + bC_{\alpha f}\left(\frac{V_y + b\dot{\psi}}{V_x}\right) \tag{2.32}$$

$$m\left(\frac{V_x^2}{R} + \dot{V}_y\right) = -C_{\alpha f}\left(\frac{V_y + a\dot{\psi}}{V_x} - \delta\right) - C_{\alpha f}\left(\frac{V_y + b\dot{\psi}}{V_x}\right) \tag{2.33}$$

These equations are then solved for yaw acceleration and lateral acceleration and rearranged to be functions of yaw rate, lateral velocity, and steering angle.

$$\ddot{\psi} = \frac{-(a^2C_{\alpha f} + b^2C_{\alpha r})}{I_{zz}V_x}\dot{\psi} + \frac{-(aC_{\alpha f} - bC_{\alpha r})}{I_{zz}V_x}V_y + \frac{aC_{\alpha f}}{I_{zz}}\delta \tag{2.34}$$

$$\dot{V}_y = \left(\frac{-(aC_{\alpha f} - bC_{\alpha r})}{mV_x} - V_x\right)\dot{\psi} + \frac{-(C_{\alpha f} + C_{\alpha r})}{mV_x}V_y + \frac{C_{\alpha}}{m}\delta \tag{2.35}$$

The dynamic bicycle model can be used in this form, converted to a transfer function, or put into a state space representation. The state variable formulation is shown in Equation (2.36) and will be used in later chapters.

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} \frac{-C_2}{I_{zz}V_x} & \frac{-C_1}{I_{zz}V_x} \\ \frac{-C_1}{mV_x} - V_x & \frac{-C_0}{mV_x} \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \end{bmatrix} + \begin{bmatrix} \frac{aC_{\alpha f}}{I_{zz}} \\ \frac{C_{\alpha f}}{m} \end{bmatrix} \delta \tag{2.36}$$

where

$$C_0 = C_{\alpha f} + C_{\alpha r} \tag{2.37}$$

$$C_1 = aC_{\alpha f} - bC_{\alpha r} \tag{2.38}$$

$$C_2 = a^2 C_{\alpha f} + b^2 C_{\alpha r} \tag{2.39}$$

This dynamic bicycle model still assumes a linear tire model and steady state longitudinal speed. However, under most typical circumstances, these assumptions are not violated so this is a valid model for most scenarios. The dynamic bicycle model is also more accurate than the prior models if the model parameters are close to the true values. Vehicle mass and CG location are straightforward to measure, but tire cornering stiffness and vehicle inertia can be very difficult to estimate. Thus, in practice, the dynamic bicycle is not the optimal choice of vehicle models when these parameters are unknown. The trade-off between model complexity and parameter accuracy is explored in Chapter 3.

2.3.4 Higher Order Models

Many higher fidelity models exist and are used extensively in simulation and other applications. The nonlinear dynamic bicycle model can be easy implemented and provides a good alternative to the linear bicycle model, if one has knowledge of the vehicle parameters and a nonlinear tire model. Four-wheel models that do not make the planar motion assumption and include weight transfer to provide even more accurate dynamics, but the roll and pitch stiffnesses can be even more difficult to estimate than the tire cornering stiffnesses [56]. Many visual simulation software packages include very high fidelity vehicle dynamics modeling, such as Gazebo [57], CarSim [58], AirSim [59], and CarMaker [60]. These simulation environments use physics engines to model the system as different bodies of mass and the interactions between them [61]. Chapter 5 of this thesis uses Gazebo as the simulation environment due to the higher fidelity dynamics and Software-In-the-Loop (SIL) capabilities.

Chapter 3

Controller Sensitivity Analysis

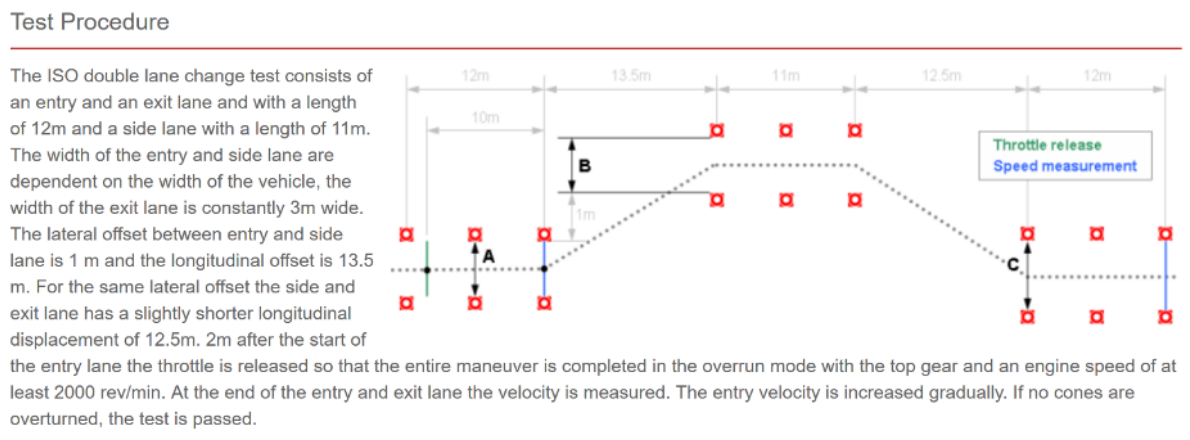
This chapter investigates the sensitivity of a wide variety of lateral vehicle controllers to dynamic model parameter uncertainties. The following sections will describe the sensitivity analysis setup, derive each controller and present its results, and formulate conclusions and takeaways from the sensitivity analysis.

3.1 Introduction

A steering controller is needed to keep an autonomous vehicle following a reference path. These controllers range from simple kinematic controllers to more sophisticated nonlinear model predictive controllers. However, complexity is not always necessary for successful operation. For example, the steering controllers on the vehicle that won the 2005 DARPA Grand Challenge (Stanford) and the vehicle that won the 2007 DARPA Urban Challenge (CMU) both utilized simple kinematic models [62]. Since then, the improvement in computing power has fueled research interests into more complex optimal control methods.

To explore the benefits and drawbacks of different steering controllers when the vehicle dynamics contain structured uncertainties, a sensitivity analysis was performed in MATLAB [63]. Structured uncertainties are when the internal parameters of the plant model are uncertain causing correlated perturbations in the response. This sensitivity analysis was designed to evaluate the strengths and weaknesses of various controller types and to ultimately inform the design of a high performance vehicle independent path following controller. The controllers were evaluated based on their mean lateral errors, max lateral errors, and stability. The types of controllers tested include kinematic control, classical model-based pole placement control, feedforward-feedback lookahead error control, model predictive control, and model reference adaptive control.

Each controller type in this chapter was tested against a double lane change path. The double lane change was chosen because it is repeatable, designed to excite the dynamics of the vehicle, and produces high slip values. It follows the standard set by International Standard ISO 3888-2 and shown below in Figure 3.1.


Test Procedure

The ISO double lane change test consists of an entry and an exit lane and with a length of 12m and a side lane with a length of 11m. The width of the entry and side lane are dependent on the width of the vehicle, the width of the exit lane is constantly 3m wide. The lateral offset between entry and side lane is 1 m and the longitudinal offset is 13.5 m. For the same lateral offset the side and exit lane has a slightly shorter longitudinal displacement of 12.5m. 2m after the start of the entry lane the throttle is released so that the entire maneuver is completed in the overrun mode with the top gear and an engine speed of at least 2000 rev/min. At the end of the entry and exit lane the velocity is measured. The entry velocity is increased gradually. If no cones are overturned, the test is passed.

Figure 3.1: ISO Double Lane Change Maneuver.

Figure directly from [64].

3.1.1 Reference Path Generation

Two path interpolation methods were investigated to create the reference path for the double lane change. First, Piecewise Cubic Hermite Polynomial (PCHIP) interpolation [65] was used to create the desired path shown in Figure 3.2. This was originally chosen because PCHIP interpolation creates a trajectory with minimal overshoot and oscillation. However, using PCHIP created a path that contained points where the curvature increased immediately from zero to a large value, causing the derivative of curvature to be infinite at these points. This can be seen in the right subplot where the curvature is plotted along the path. As a result, all of the controllers were less smooth because the vehicle could not react to the instantaneous change in curvature. This ”jerky” behavior would not be desirable as a passenger of a vehicle.

In order to alleviate these issues, a smoothing spline was implemented [66] to create the desired path. A smoothing spline is another path interpolation method that guarantees smooth

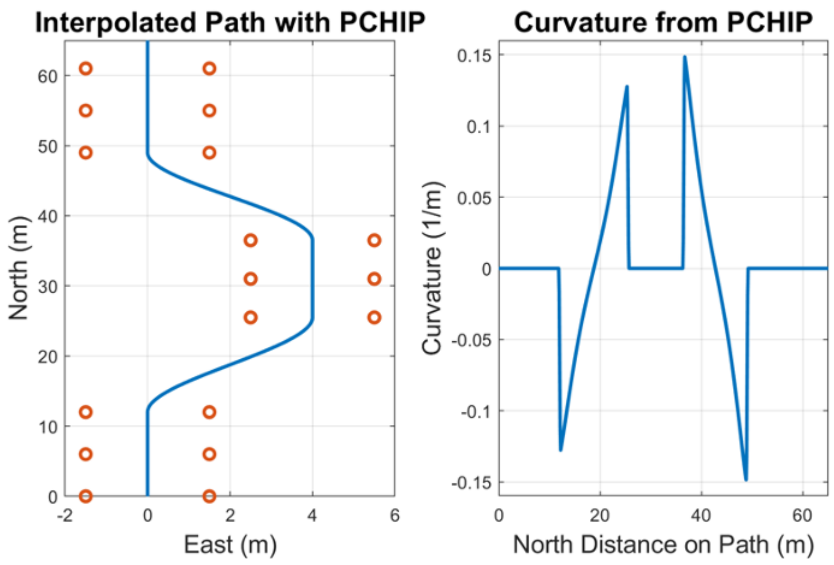


Figure 3.2: Piecewise Cubic Hermite Polynomial (PCHIP) Path Generation.

transitions in curvature values. The smoother curvature translates to smoother steering actuation which better matches the driving behaviors of humans in the real-world [67]. Different parameters for the path creation were tested until the desired path shape was achieved. As shown below on the right in Figure 3.3, the curvature no longer increases from zero to a high curvature instantaneously. Instead, the curvature is always following a linear increase or decrease. This has the added benefit of having much lower maximum curvature values. This allows the vehicle to follow the path with smoother steering inputs and lower yaw rates.

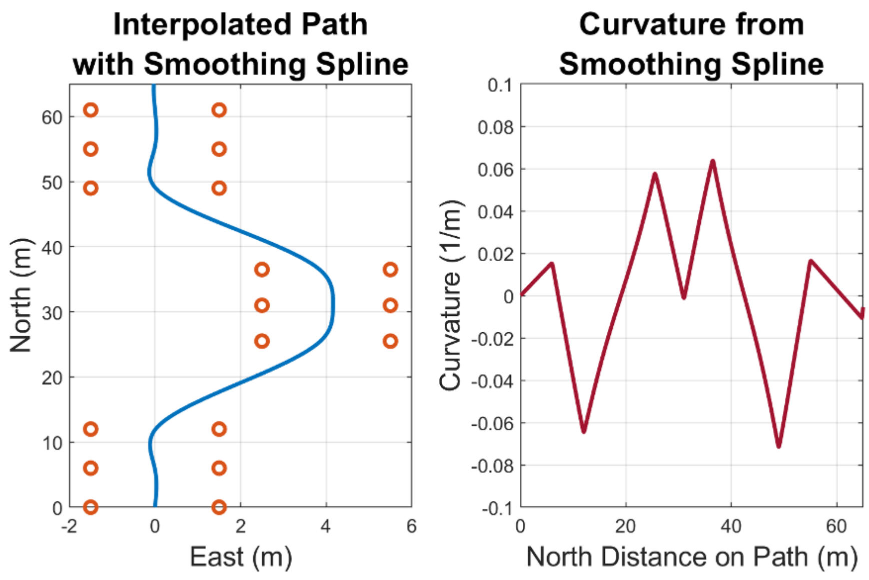


Figure 3.3: Smoothing Spline Path Generation.

As expected, the lower curvatures of the smoothing spline result in lower overall steering angles and smoother vehicle yaw rates. This can be seen in Figure 3.4, which shows a comparison of the yaw rates and heading during the double lane change with the lookahead error controller described later in Section 3.4. The left shows the PCHIP method of path generation and the right side shows the smoothing spline method. Based on these results, the smoothing spline was used as the reference path for the remainder of Chapter 3. The lower steering angles and decreased steer angle rates help to stay below saturation limits present on an experimental vehicle. Additionally, having smoother and more gradual changes in yaw rate makes for a more comfortable vehicle to ride in and has a lower chance of upsetting the vehicle's balance.

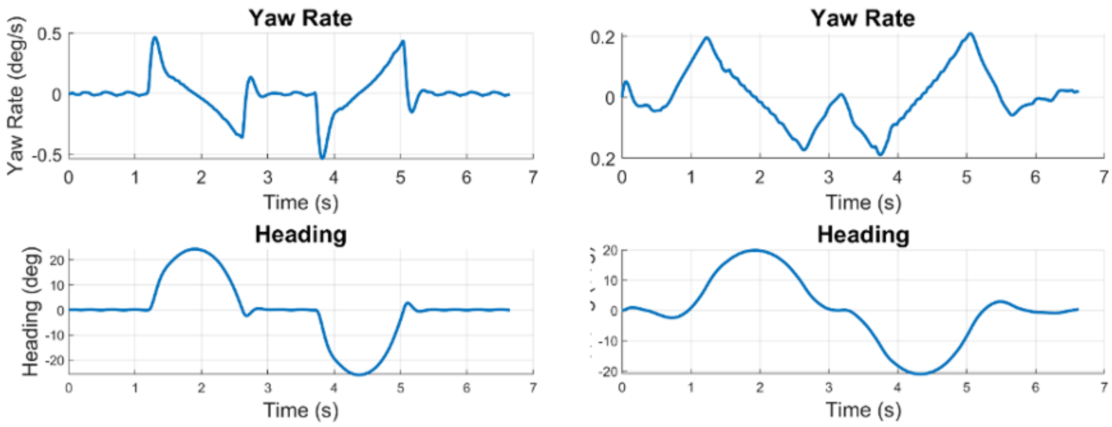


Figure 3.4: Vehicle Response to PCHIP vs. Smoothing Spline.

3.1.2 Simulation Setup

To perform the sensitivity analysis, each controller follows the double lane change reference path. The controller calculates the input steering angle desired based on a nominal vehicle model. Then the steering input calculated from the controller is simulated through the dynamic bicycle model with a nonlinear Brush-Fiala tire model. The simulation model has uncertainties ranging from -50% to +50% of the nominal value for each parameter. The nominal values of the bicycle model parameters and the modified values used in the sensitivity analysis are shown below in Table 3.1. Each run only has a single parameter changed to clearly see which parameters have the most effect on the performance of the controller. The results shown are from simulations with a longitudinal speed of 10 m/s.

Table 3.1: Parameters used for sensitivity analysis.

Dynamic Bicycle Model Parameters					
Percent from Nominal	-50%	-25%	Nominal	+25%	+50%
Front Tire Cornering Stiffness (N/rad)	60000	90000	120000	150000	180000
Rear Tire Cornering Stiffness (N/rad)	92300	138450	184600	230750	276900
Distance from CG to Front Axle (m)	0.6285	0.9427	1.2570	1.5712	1.8855
Distance from CG to Rear Axle (m)	0.7965	1.1947	1.593	1.9912	2.3895
Vehicle Mass (kg)	928.5	1392.7	1857	2321.3	2785.5
Vehicle Inertia $(Kg \cdot m^2)$	2146	3219	4292	5365	6438
*Tire cornering stiffnesses are per axle					

The following sections contain the formulation, performance, and analysis of each controller type. The last section in this chapter provides the conclusions that were drawn from this sensitivity analysis.

3.2 Kinematic Control

Kinematic controllers have been used widely in robotics and driving applications. Kinematic control uses the kinematic equation of the system, inverting it to calculate the input needed to achieve the desired state. In the case of autonomous driving, the input is steering angle and the desired state is typically a yaw rate command. This type of control is commonly used when the dynamics are negligible such as low speed driving. In low speed situations there is very little tire slip, and inverse kinematic control is an effective solution. It is also popular compared to more advanced controllers because only wheelbase length and current speed need to be known to calculate the steering angle for a desired yaw rate. The following two sections will derive the kinematic control law and present the sensitivity analysis results.

3.2.1 Control Formulation

To formulate the kinematic controller, Equation (2.12) from Chapter 2 is used to calculate a steering angle (δ) for a given a yaw rate command ($\dot{\psi}_{cmd}$).

$$\frac{|\vec{V}|}{L}\tan(\delta) = \dot{\psi}_{cmd} \tag{3.1}$$

$$\tan\delta = \frac{L \cdot \dot{\psi}_{cmd}}{|\vec{V}|} \tag{3.2}$$

$$\delta = \arctan\frac{L \cdot \dot{\psi}_{cmd}}{|\vec{V}|} \tag{3.3}$$

If the vehicle sideslip angle is assumed to be very small, the velocity can be approximated as longitudinal velocity.

$$|\vec{V}| \approx V_x \qquad \Longrightarrow \qquad \delta = \arctan\frac{L \cdot \dot{\psi}_{cmd}}{V_x} \tag{3.4}$$

To determine the yaw rate command that should be input to the kinematic controller, the path dynamics controller derived later in Chapter 4 was used. This is shown in Equation (3.5), but it is not the focus of this section.

$$\dot{\psi}_{cmd} = -k_p \cdot e_{LA} + \kappa \cdot V_x \tag{3.5}$$

3.2.2 Results

The kinematic controller had a max error of 0.2585 meters and a mean error of 0.1271 meters when simulated with the nominal values. The lateral errors and the steering angle input from the controller during this run can be seen in Figure 3.5.

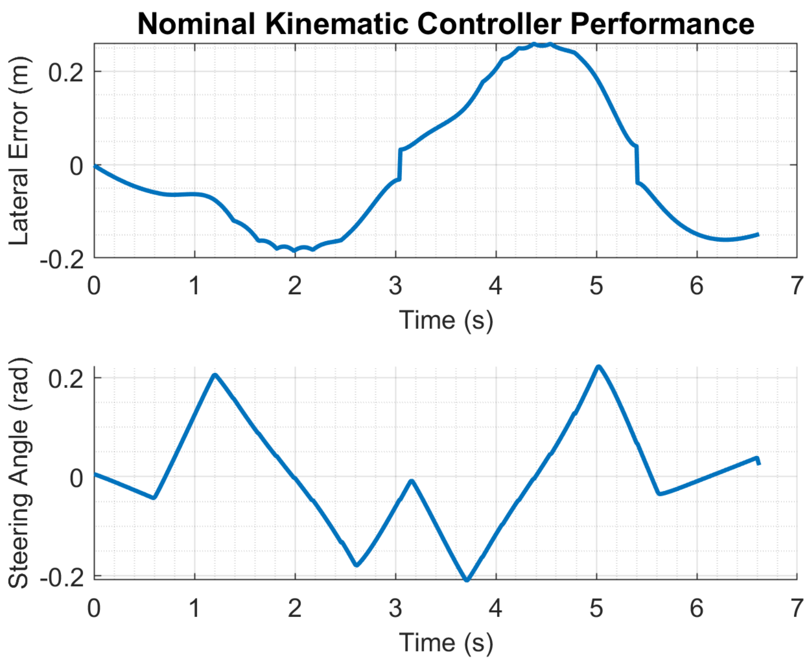


Figure 3.5: Kinematic Controller Nominal Model Performance.

Figure 3.6 below shows the sensitivity of the kinematic controller to inaccuracies of each of the dynamic bicycle model parameters. Each solid line represents the maximum lateral error compared to the reference path when performing the double lane change. The dotted lines represent the mean lateral error from the reference path during the same maneuver. The individual line colors represent the various parameters that were changed in the simulation model. The horizontal axis represents the change to each parameter, and the vertical axis represents the error due to the inaccuracy in the given parameter.

The kinematic controller's max and a mean errors (0.2585 meters and 0.1271 meters) when simulated with the nominal values can be seen where the change to vehicle parameter is 0% along the x-axis. The darker blue line shows the vehicle response to inaccurate front tire cornering stiffness $C_{\alpha r}$. Front tire cornering stiffness greatly increased error when it was

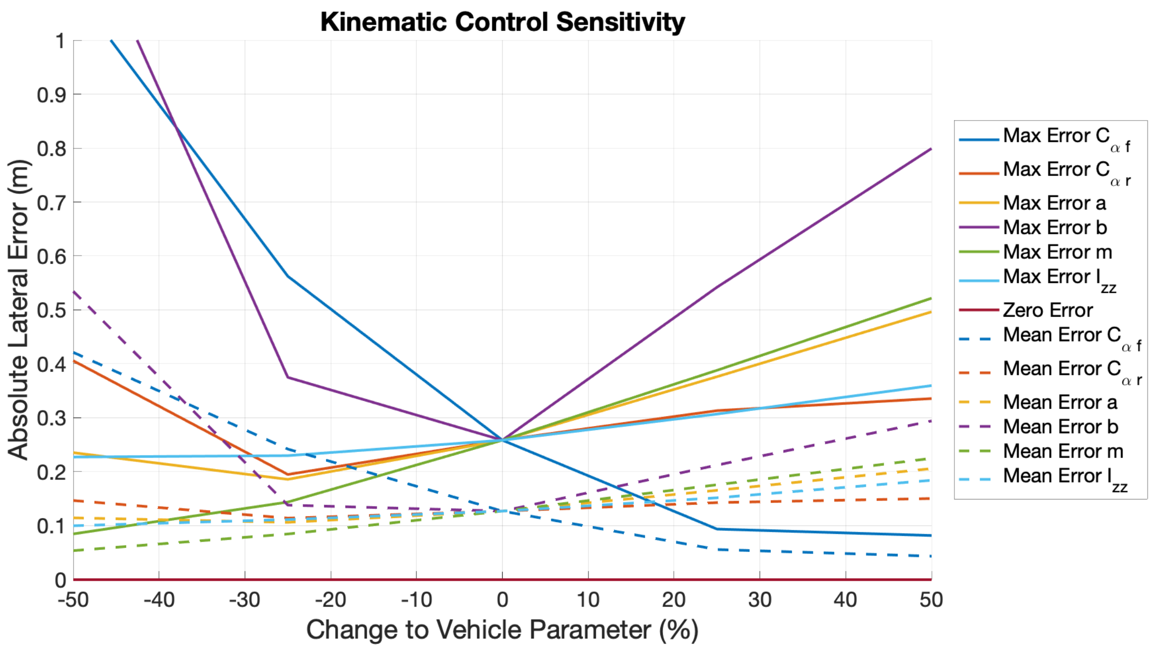


Figure 3.6: Kinematic Controller Sensitivity to Model Inaccuracy.

decreased, probably because the tires could not generate the required force to make the maneuver. When increased, however, the error actually decreased, due to the front tires producing more force than expected by the controller, which caused the vehicle to respond quicker to the reference signal. The rear tire cornering stiffness $C_{\alpha r}$ (orange) did not greatly affect the lateral error, but this could be different in an oversteer vehicle.

The distance from the CG to the front axle a shown by the yellow line did not have much effect on the lateral error when overestimated, but did increase error significantly when underestimated. The most significant degradation in controller performance was seen when the distance from the CG to the rear axle b was inaccurate. This can be seen by the purple line which increases dramatically when b was overestimated or underestimated. This resultis explained through Equation (2.12) because the kinematic model only relies on the wheelbase length L, which is greatly affected by inaccuracies in a and b. The observation that b caused a larger increase in error was at least partially due to the nominal value being larger meaning that any percentage change was larger than the equivalent change to a.

The vehicle mass m (green) decreased error when decreased below nominal and increased error when it was increased. Vehicle inertia about the z-axis (light blue) had a similar effect as mass, but to a much lesser extent. This is logical since both of these vehicle characteristics act to decrease response when they are larger. Conversely, when mass and inertia are decreased they increase the nimbleness of the vehicle.

Overall, $C_{\alpha f}$, a, and b had the largest effect on controller performance. The individual simulation runs and detailed results can be found in the appendices in Section A.1.

3.3 Dynamic Bicycle Model Pole Placement

The dynamic bicycle model controller is a feedback controller on the yaw rate designed with classical control methods. The second-order linear dynamic bicycle model derived in Chapter 2 is employed and used to calculate the controller gains for desired closed loop eigenvalues (poles). This method is also commonly used, but takes more of the vehicle dynamics into consideration than the kinematic controller.

The formulation of this controller and the results of the sensitivity analysis are shown in the following sections.

3.3.1 Control Formulation

The controller was formulated in state space. First, the dynamic bicycle model from Chapter 2 and the nominal values from Table 3.1 were used to form the state matrices as shown in Equations (3.63.7) below.

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} \frac{-(a^2C_{\alpha f}+b^2C_{\alpha r})}{I_{zz}V_x} & \frac{-(aC_{\alpha f}-bC_{\alpha r})}{I_{zz}V_x} \\ \frac{-(aC_{\alpha f}-bC_{\alpha r})}{mV_x} - V_x & \frac{-(C_{\alpha f}+C_{\alpha r})}{mV_x} \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \end{bmatrix} + \begin{bmatrix} \frac{aC_{\alpha f}}{I_{zz}} \\ \frac{C_{\alpha f}}{m} \end{bmatrix} \delta \tag{3.6}$$

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} -15.3322 & 3.3371 \\ -2.2871 & -16.4028 \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \end{bmatrix} + \begin{bmatrix} 35.1445 \\ 64.6204 \end{bmatrix} \delta \tag{3.7}$$

The desired system response was a settle time of 0.5 seconds and a damping ratio of 0.707, because these parameters gave the best performance when compared to other simulation runs with different values for settle time and damping. The closed loop eigenvalues for these response characteristics were then calculated as follows.

$$\omega_n = \frac{4.6}{0.707 \cdot 0.5} \tag{3.8}$$

$$\omega_d = \omega_n \cdot \sqrt{1 - 0.707^2} \tag{3.9}$$

$$s_{des} = -.707 \cdot \omega_n \pm \omega_d \cdot i \tag{3.10}$$

$$s_{des} = -9.2000 \pm 9.2028i \tag{3.11}$$

Because the simulation is run in the discrete domain at 100 Hz, both the vehicle model and the desired eigenvalues were converted to the discrete domain as shown below in Equations (3.12-3.14).

$$\begin{bmatrix} \dot{\psi}_{k+1} \\ \dot{V}_{y_{k+1}} \end{bmatrix} = \begin{bmatrix} 0.8575 & 0.0285 \\ -0.0195 & 0.8484 \end{bmatrix} \begin{bmatrix} \dot{\psi}_k \\ V_{y_k} \end{bmatrix} + \begin{bmatrix} 0.3355 \\ 0.5923 \end{bmatrix} \delta_k \tag{3.12}$$

$$z_{des} = e^{s_{des} \cdot Ts} \tag{3.13}$$

$$z_{des} = 0.9082 \pm 0.0838i \tag{3.14}$$

To determine the controller gains K, the characteristic equation is solved for the desired eigenvalues. This can be done manually by solving for K in (3.15) or in MATLAB with the `place` function.

$$|zI - A_d + B_d \cdot K| = 0 \tag{3.15}$$

To ensure the closed loop system has a DC gain of one, the non-zero reference scaling values are calculated. A DC gain of one ensures that at steady state the output yaw rate would match the yaw rate command. Note, $C_d = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D_d = 0.$

$$\begin{bmatrix} N_x \\ N_u \end{bmatrix} = \begin{bmatrix} A_d - I & B_d \\ C_d & D_d \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.16}$$

The final equation for commanded steering angle is formulated in Equation (3.17).

$$\delta_k = N_u \cdot \dot{\psi}_{cmdk} - K \cdot \left(\begin{bmatrix} \dot{\psi}_k \\ V_{y_k} \end{bmatrix} - N_x \cdot \dot{\psi}_{cmdk} \right) \tag{3.17}$$

The same outer loop from the kinematic controller was used to determine the yaw rate command that should be input into the pole placement controller, Equation (3.5), shown below for easy reference.

$$\dot{\psi}_{cmd} = -k_p \cdot e_{LA} + \kappa \cdot V_x \tag{3.18}$$

3.3.2 Results

The pole placement controller had a max error of 0.1396 meters and a mean error of 0.0698 meters when simulated with the nominal values. The lateral errors and the steering angle input from the controller during this run can be seen in Figure 3.7.

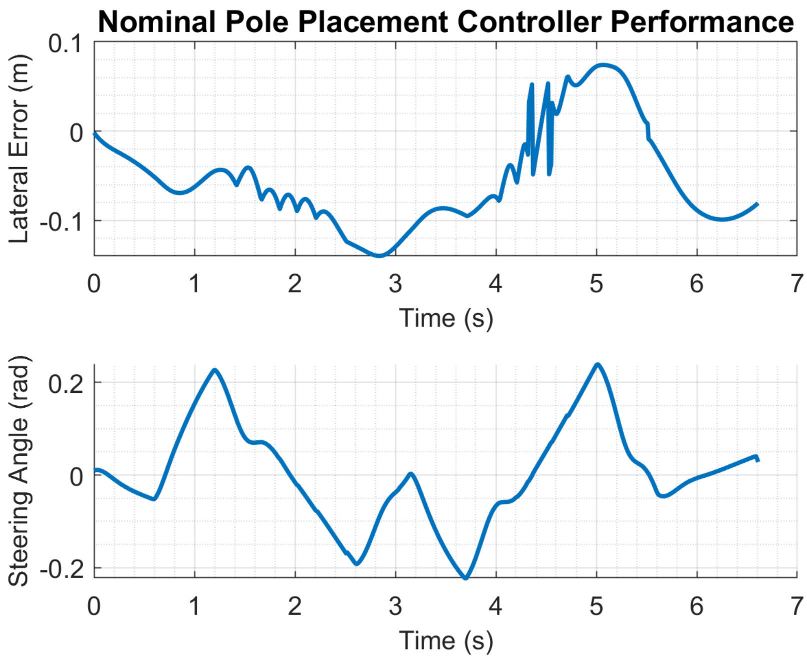


Figure 3.7: Dynamic Bicycle Model Pole Placement Controller Nominal Model Performance.

The pole placement controller designed with the second order dynamic bicycle model was expected to have lower errors than the kinematic controller, since the model is higher fidelity. Figure 3.8 below shows the sensitivity of the dynamic bicycle model pole placement controller to inaccuracies of each of the dynamic bicycle model parameters. For an explanation of the plot, refer to Section 3.2.2.

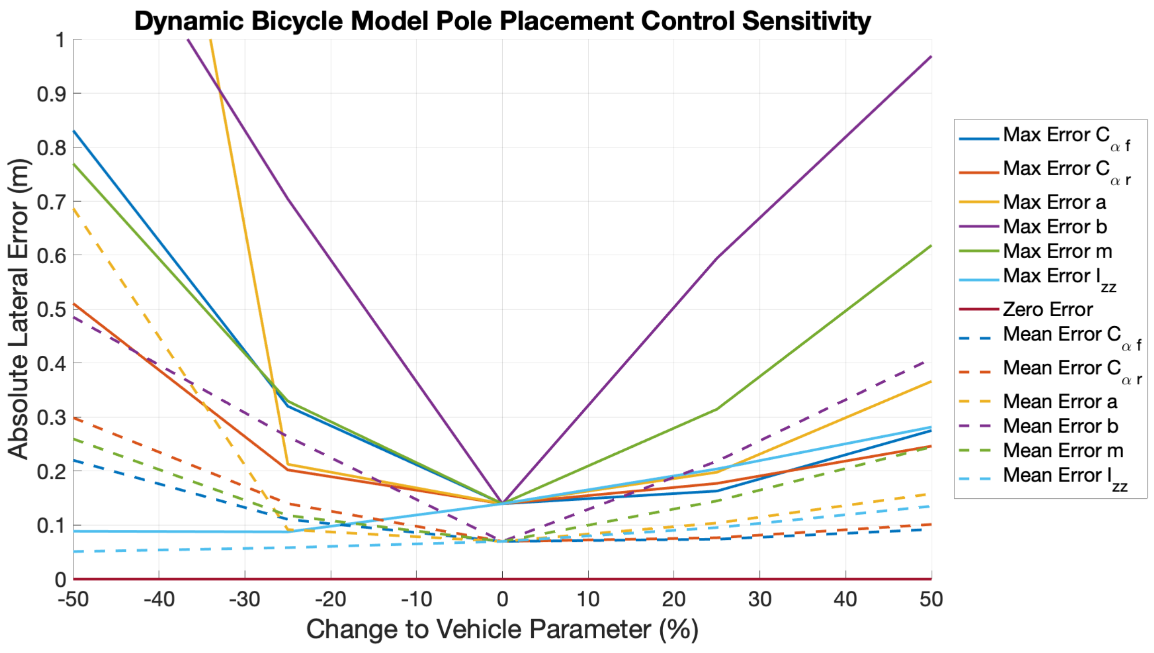


Figure 3.8: Dynamic Bicycle Model Pole Placement Controller Sensitivity to Model Inaccuracy.

The pole placement controller's max error of 0.1396 meters and mean error of 0.0698 meters when simulated with the nominal values is shown where the change in vehicle parameter is 0% along the x-axis. The expectation that the errors would be lower than those of the kinematic controller was true with the nominal parameter values. When the model was known, the max and mean errors were half those of the kinematic controller. When the parameters were uncertain, however, the improvements in performance disappeared.

Front tire cornering stiffness $C_{\alpha r}$ (darker blue) greatly increased error when it was overestimated, but only slightly increased the error when it was underestimated. As can be seen by the orange line, the rear tire cornering stiffness $C_{\alpha r}$ had a similar effect but to a lesser degree. The increase in error when overestimating tire cornering stiffness was likely because the tires could not generate the lateral force needed.

The distance from the CG to the front axle a (yellow) had a significant effect on the lateral error when inaccurate, especially when it was overestimated. Again, inaccuracies in the distance from the CG to the rear axle b produced the most substantial errors. This can be seen by the purple line which shows large errors when b was overestimated or underestimated. These large sensitivities of the dynamic bicycle model controller to a and b are likely due to the large difference in understeer gradient between the nominal model and the simulated model. This changes the DC gain between steering and yaw rate for a given speed and shows the importance of these parameters if using this type of controller.

The vehicle mass m (green) increased error significantly when it was either overestimated or underestimated. Vehicle inertia I_{zz} about the z-axis (light blue) had much less effect on performance than any of the other parameters.

Even though the pole placement controller had very low errors when used with the nominal model, it was extremely sensitive to inaccuracies in $C_{\alpha f}$, a, b, and m. Overall, it was more sensitive than the kinematic controller and produced higher errors when the model was inaccurate. The individual simulation runs and detailed results can be found in the appendices in Section A.2.

3.4 Lookahead Error Feedforward-Feedback Controller

The lookahead error feedforward-feedback controller is a control architecture based on autonomous racing research done by Stanford University's Dynamic Design Lab (DDL) [68]. This controller was designed to have low lateral and heading errors, even when the car is in high dynamic situations such as racing or aggressive driving. It is composed of two parts: a feedforward portion that calculates the required steering angle to follow the path profile, and a feedback component that corrects any lateral or heading deviation. The block diagram of this controller can be seen in Figure 3.9. The following two sections will go through the derivation of the lookahead error feedforward-feedback control law and present the sensitivity analysis results.

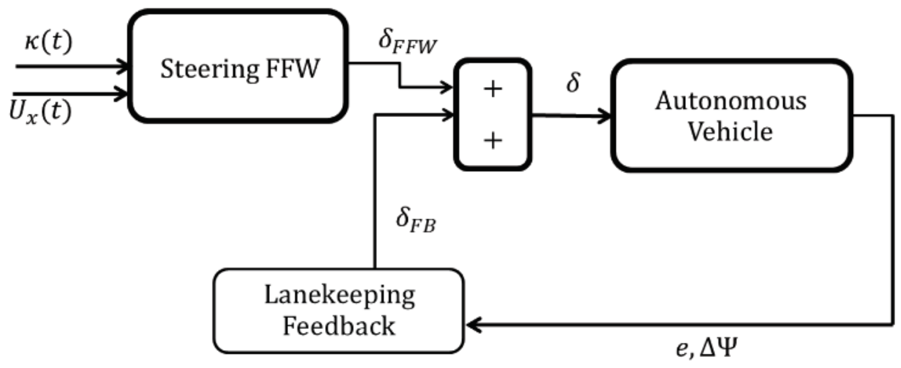


Figure 3.9: Feedforward-Feedback Controller Block Diagram.

3.4.1 Control Formulation

The first component of the controller is the feedforward term that determines the steering angle needed to hold the current path profile. To determine this, the desired lateral tire forces in order to hold the curvature (κ) of the road must be calculated. Curvature κ is the inverse of the radius of the path. The lateral forces are calculated based on steady state cornering, shown below in Equations (3.19-3.20).

$$F_{yf} = \frac{m \cdot b}{L} \cdot V_x^2 \cdot \kappa \tag{3.19}$$

$$F_{yr} = \frac{m \cdot a}{L} \cdot V_x^2 \cdot \kappa \tag{3.20}$$

Then an inverted tire model is used to convert the desired lateral tire forces into desired tire slip angles. This can be done with either a linear tire model or nonlinear tire model, as shown below.

Linear Tire Model: Nonlinear Tire Model:

$$\alpha_f = \frac{F_{yf}}{C_{\alpha f}} \qquad \alpha_f = f_{tire}^{-1}(F_{yf}) \tag{3.21}$$

$$\alpha_r = \frac{F_{yr}}{C_{\alpha r}} \qquad \alpha_r = f_{tire}^{-1}(F_{yr}) \tag{3.22}$$

The desired tire slip angles to hold the road profile are then used to calculate the feedforward steering angle with Equation (3.23).

$$\delta_{FFW} = L \cdot \kappa - \alpha_f + \alpha_r \tag{3.23}$$

To determine the feedback steering angle, the lateral error e and heading error $\Delta\psi$ are first defined by Figure 3.10. The lookahead distance x_{la} is a distance projected out in front of the vehicle that is used to calculate the lateral lookahead error. Lookahead distance is a tuning parameter that acts similar to damping in the system, keeping it from becoming oscillatory. However, if it is set too high, it can cause the vehicle to cut corners, leading to more error.

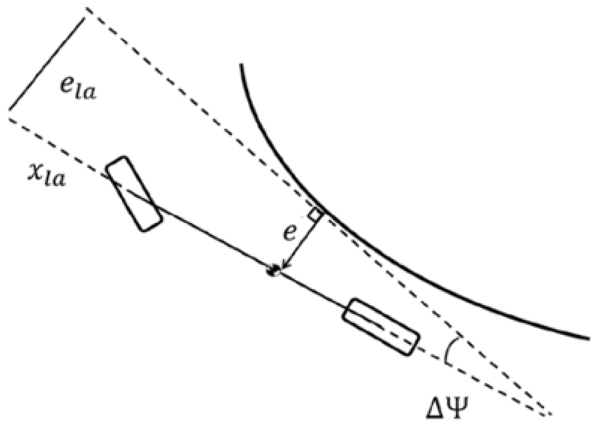


Figure 3.10: Schematic of Error States
Adapted from [69].

The error states e and $\Delta\psi$ are used to calculate the lookahead error e_{la}. This value is the lateral error between the lookahead point to the tangent of the path.

$$e_{LA} = e + x_{LA} \cdot \Delta\psi \tag{3.24}$$

Lookahead error is then multiplied by a proportional gain k_p to determine the steering command that will decrease the lookahead error.

$$\delta_{FB} = -k_p \cdot e_{LA} \tag{3.25}$$

The total steering input is then calculated by summing the feedback and feedforward steering terms.

$$\delta_{Total} = \delta_{FFW} + \delta_{FB} \tag{3.26}$$

In order to improve path tracking when larger sideslip angles are occuring, such as in high dynamic maneuvers, a steady state sideslip angle can be added to the controller formulation. This is shown below in Equations (3.27-3.28).

$$\beta_{SS} = \alpha_r^{FFW} + b * \kappa \tag{3.27}$$

$$\delta_{FB} = -k_p \left(e_{LA} + x_{LA} \left(\Delta\Psi + \beta_{SS}\right)\right) \tag{3.28}$$

This modification did not make any significant difference in the simulations that were run in this thesis, so it was left out for simplicity. However, under more extreme scenarios or higher speeds, adding this term may be necessary.

3.4.2 Results

The lookahead error feedforward-feedback controller had a max error of 0.1836 meters and a mean error of 0.0900 meters when simulated with the nominal values. The lateral errors and the steering angle input from the controller during this run can be seen in Figure 3.11.

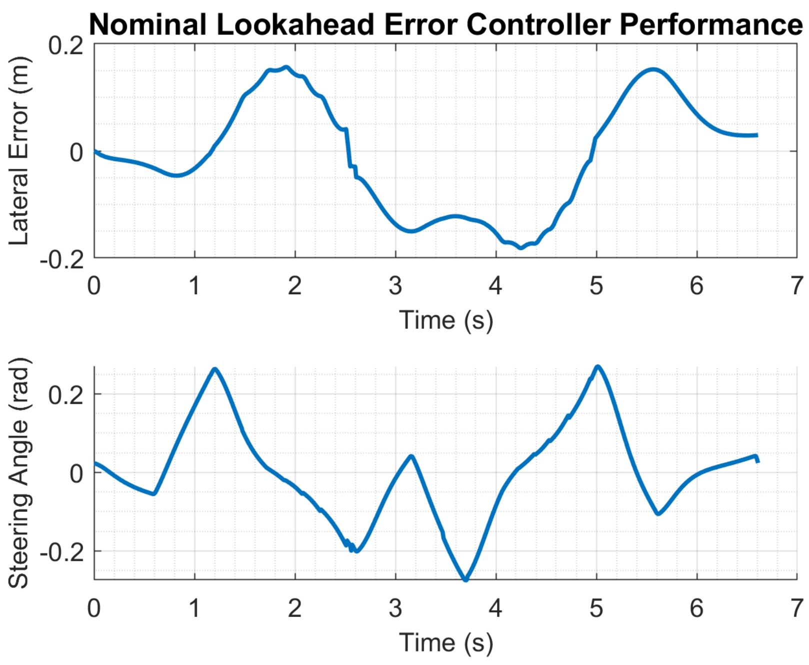


Figure 3.11: Lookahead Error Controller Nominal Model Performance.

The lookahead error feedforward-feedback controller was expected to have much better path following performance than the prior two controllers because it feeds forward the curvature and looks at the path ahead. This allows it to steer proactively, instead of only reacting to the errors. Additionally, it relies on the vehicle model only for the feedforward portion. Figure 3.12 below shows the sensitivity of the lookahead error feedback-feedforward controller to inaccuracies of each of the dynamic bicycle model parameters. For an explanation of the plot, refer to Section 3.2.2.

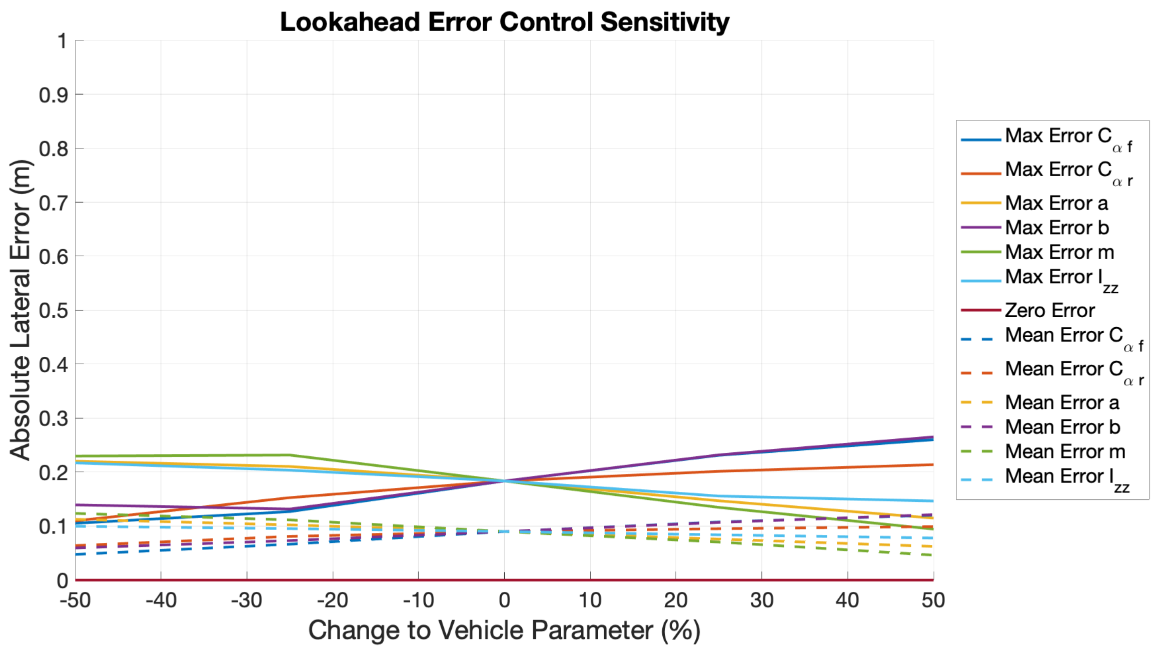


Figure 3.12: Lookahead Error Controller Sensitivity to Model Inaccuracy.

The lookahead error feedforward-feedback controller had a max and mean errors can be seen where the x-axis is zero. These mean and max errors are in between the kinematic controller and the pole placement controller. However, unlike the first two controllers, this controller was very robust to parameter inaccuracy. This is likely due to only the feedforward term relying on the model parameters.

Shown with the darker blue line and orange line, the front tire cornering stiffness $C_{\alpha r}$ and rear tire cornering stiffness $C_{\alpha r}$ both slightly increased error when underestimated and slightly decreased error when overestimated. These error changes are very small compared to the previous two controllers.

The distance from the CG to the rear axle b followed the same trend as the tire cornering stiffnesses and can be seen by the purple line. While it did have the largest effect of the vehicle parameters, it was still very small compared to the other controllers. The distance from the CG to the front axle a had an even smaller impact on the controller performance. The error was slightly increased when a was overestimated and slightly decreased when underestimated. This can be seen by the yellow line.

The vehicle mass m (green) increased error slightly when it was overestimated and decreased it when underestimated. Vehicle inertia I_{zz} about the z-axis (light blue) acted in the same manner, but again made the least difference overall.

The lookahead error feedforward-feedback controller was extremely resilient to model uncertainty and performed well under all conditions presented. Using the error states for the feedback and the required steady state steering to follow the curvature for the feedforward lessened the reliance on the dynamic model. At the same time it maintained great path tracking capabilities. Even with the most extreme inaccuracies, the lateral error remained below 0.3 m. This would be more than acceptable for most real-world driving applications. The feedforward-feedback architecture helped inform the design of the vehicle agnostic controller in Chapter 4. The individual simulation runs of the sensitivity analysis and detailed results can be found in the appendices in Section A.3.

3.5 Model Predictive Control

Model Predictive Control (MPC) is a form of control that determines the optimum control input to follow a reference signal over a period of time. This differs from a Linear Quadratic Regulator (LQR), another form of optimal control, because it performs the optimization over a finite time period instead of an infinite time period. The time period is referred to as the time horizon H and is always shifting forward over time. Because of this property, MPC is also referred to as receding horizon control.

MPC uses a dynamic model of the system to predict the response of the system in the future to optimize the control input sequence based off a cost function. The control horizon N_c determines how many sequences of control inputs in the future the MPC calculates. The prediction horizon N_p refers to how far forward the controller predicts the system response [70]. Model predictive control can also be done with nonlinear dynamics (NMPC) [32].

The optimization is typically done by minimizing a cost function that includes penalties on control effort, output error, and any constraints to the system. Only the first input of the control sequence determined by the optimization is used and then the optimization routine is repeated. The complexity of the optimization problem as well as the fact that must be iterated through every time step has traditionally restricted the use of Model Predictive Control to slower systems such as chemical plants and factory environments. However, the massive increase in computing power available and efficient optimization libraries over recent years has increased the interest in using MPC for faster dynamic systems, such as autonomous vehicles and aerial platforms. The next few sections will go through the derivation of the bicycle model in path coordinates, the formulation of an unconstrained model predictive controller based on this dynamic model, and the results of the sensitivity analysis using the MPC algorithm.

3.5.1 Model Derivation in Path Coordinates

In order to have the model predictive controller optimally control the vehicle to the reference path, the linearized dynamic bicycle model was transformed into the navigation frame. The states chosen to be penalized by the MPC's cost function J are the lateral error and the heading error. These states and their derivatives are used as the new state vector.

First, the yaw rate needed to follow the path during steady-state cornering r_{ss} is calculated in Equation (3.29) where $r = \dot{\psi}$.

$$r_{ss} = \kappa \cdot V_x \tag{3.29}$$

The variable κ represents the reference path curvature and is defined as the change in angle over a unit arc length. This is also equivalent to one divided by the radius of the turn.

$$\kappa = \frac{d\theta}{ds} = \frac{1}{R} \tag{3.30}$$

Lateral acceleration during steady state can then be shown to be equal to the yaw rate multiplied by the longitudinal velocity.

$$\dot{V}_y = r_{ss} \cdot V_x \tag{3.31}$$

With e as the lateral error of the vehicle's CG to the path, the second derivative of error is then found.

$$\ddot{e} = \dot{V}_y + V_x(r - r_{ss}) \tag{3.32}$$

The path yaw rate subtracted from the current vehicle yaw rate would be equal to the rate of change of the heading error $\dot{\theta}_e$.

$$\dot{\theta}_e = r - r_{ss} \tag{3.33}$$

$$\ddot{\theta}_e = \dot{r} - \dot{r}_{ss} \tag{3.34}$$

Equation (3.33) can be plugged into Equation (3.32) to put the lateral error in terms of heading error.

$$\ddot{e} = \dot{V}_y + V_x\dot{\theta}_e \tag{3.35}$$

$$\dot{e} = V_y + V_x \sin\theta_e \tag{3.36}$$

Now the linear dynamic bicycle model equations derived in Chapter 2 are inserted into Equations (3.34-3.35) and rearranged.

$$\dot{V}_y = \ddot{e} - V_x * \dot{\theta}_e \tag{3.37}$$

$$= \frac{-(c_{\alpha f} + c_{\alpha r})}{m * V_x}(\dot{e} - V_x * \theta_e) + \left(\frac{b * c_{\alpha r} - a * c_{\alpha f}}{m * V_x} - V_x\right)\left(\dot{\theta}_e + r_{ss}\right) + \frac{c_{\alpha f}}{m}\delta \tag{3.38}$$

$$\ddot{e} = \frac{-(c_{\alpha f} + c_{\alpha r})}{m * V_x}\dot{e} + \frac{(c_{\alpha f} + c_{\alpha r})}{m}\theta_e + \left(\frac{b * c_{\alpha r} - a * c_{\alpha f}}{m * V_x}\right)\dot{\theta}_e + \left(\frac{b * c_{\alpha r} - a * c_{\alpha f}}{m * V_x} - V_x\right) r_{ss} + \frac{c_{\alpha f}}{m}\delta \tag{3.39}$$

$$\dot{r} = \ddot{\theta}_e + \dot{r}_{ss} \tag{3.40}$$

$$= \left(\frac{b * c_{\alpha r} - a * c_{\alpha f}}{I_z * V_x}\right)(\dot{e} - V_x * \theta_e) - \left(\frac{a^2 * c_{\alpha f} + b^2 * c_{\alpha r}}{I_z * V_x}\right)\left(\dot{\theta}_e + r_{ss}\right) + \frac{a * c_{\alpha f}}{m}\delta \tag{3.41}$$

$$\ddot{\theta}_e = \left(\frac{b * c_{\alpha r} - a * c_{\alpha f}}{I_z * V_x}\right)\dot{e} + \left(\frac{a * c_{\alpha f} - b * c_{\alpha r}}{I_z}\right)\theta - \left(\frac{a^2 * c_{\alpha f} + b^2 * c_{\alpha r}}{I_z * V_x}\right)\dot{\theta}_e - \left(\frac{a^2 * c_{\alpha f} + b^2 * c_{\alpha r}}{I_z * V_x}\right) r_{ss} + \frac{a * c_{\alpha f}}{m}\delta - \dot{r}_{ss} \tag{3.42}$$

The above equations are then put into state space for use as the dynamic matrix A for MPC.

$$\begin{bmatrix} \dot{e} \\ \ddot{e} \\ \dot{\theta}_e \\ \ddot{\theta}_e \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(c_{\alpha f}+c_{\alpha r})}{m*V_x} & \frac{c_{\alpha f}+c_{\alpha r}}{m} & \frac{b*c_{\alpha r}-a*c_{\alpha f}}{m*V_x} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{b*c_{\alpha r}-a*c_{\alpha f}}{I_z*V_x} & \frac{a*c_{\alpha f}-b*c_{\alpha r}}{I_z} & \frac{-(a^2*c_{\alpha f}+b^2*c_{\alpha r})}{I_z*V_x} \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \\ \theta_e \\ \dot{\theta}_e \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{c_{\alpha f}}{m} \\ 0 \\ \frac{a*c_{\alpha f}}{I_z} \end{bmatrix} \delta + \begin{bmatrix} 0 \\ \frac{b*c_{\alpha r}-a*c_{\alpha f}}{m*V_x} - V_x \\ 0 \\ \frac{-(a^2*c_{\alpha f}+b^2*c_{\alpha r})}{I_z*V_x} \end{bmatrix} r_{ss} \tag{3.43}$$

3.5.2 Control Formulation

To formulate the model predictive controller, the state equation above must be discretized. The difference equation that represents this system is shown below in Equation (3.45).

$$x_d(k+1) = A_d x_d(k) + B_d u(k) \tag{3.44}$$

$$y(k) = C_d x_d(k) \tag{3.45}$$

This is then reformulated to represent the change in states based on the change in input.

$$\Delta x_d(k+1) = x_d(k+1) - x_d(k) \tag{3.46}$$

$$= A_d \Delta x_d(k) + B_d \Delta u(k) \tag{3.47}$$

An augmented matrix is formed with the measurement y being the lateral offset from the path. This augmentation embeds an integrator in the MPC, which helps to drive the error to zero.

$$\begin{bmatrix} \Delta x_d(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} A_d & 0 \\ C_d A_d & 1 \end{bmatrix} \begin{bmatrix} \Delta x_d(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_d \\ C_d B_d \end{bmatrix} \Delta u(k) \tag{3.48}$$

$$y(k) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x_d(k) \\ y(k) \end{bmatrix} \tag{3.49}$$

As alluded to in the introduction to this section, the prediction horizon N_p is the number of time steps that the MPC predicts the states forward, and the control horizon N_c is the number of time steps the control sequence is optimized for. The values used for the sensitivity analysis are shown below.

$$N_p = 10, \qquad N_c = 5$$

The predicted output over the prediction horizon and the control sequence for the control horizon are denoted by Y and ΔU, respectively. The current time step is k_i.

$$Y = [y(ki+1 \mid ki),\ y(ki+2 \mid ki),\ ...,\ y(ki+N_p \mid ki)]^T \tag{3.50}$$

$$\Delta U = [\Delta u(ki),\ \Delta u(ki+1),\ ...,\ \Delta u(ki+N_c-1)]^T \tag{3.51}$$

This can be refactored into the form:

$$Y = Fx(ki) + \Phi \Delta U \tag{3.52}$$

where,

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ ... \\ CA^{N_p} \end{bmatrix}, \qquad \Phi = \begin{bmatrix} CB & 0 & 0 & ... & 0 \\ CAB & CB & 0 & ... & 0 \\ CA^2B & CAB & CB & ... & 0 \\ ... & ... & ... & ... & ... \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & ... & CA^{N_p-N_c}B \end{bmatrix}. \tag{3.53}$$

The cost function used to minimize error from the reference path and control effort is defined as J.

$$J = (R_S - Y)^T(R_S - Y) + \Delta U^T \bar{R} \Delta U \tag{3.54}$$

$$R_S^T = \begin{bmatrix} 1 & 1 & ... & 1 \end{bmatrix}_{1\times N_p} r(ki), \qquad \bar{R} = r_w I_{N_c\times N_c}, \qquad r_w = \text{weight on input} \tag{3.55}$$

Where unconstrained, as in this sensitivity analysis, the optimum control sequence has the analytical solution shown below in Equation (3.56) [70].

$$\Delta U = (\Phi^T\Phi + \bar{R})^{-1}\Phi^T(R_S - Fx(ki)) \tag{3.56}$$

The first input of the optimal control sequence ΔU is used on every time step as the controller output for the sensitivity analysis. Since it recalculates a new control sequence at each time step, the model predictive controller is better able to account for disturbances in the system and model uncertainties.

3.5.3 Results

The model predictive controller had a max error of 0.1556 meters and a mean error of 0.0601 meters when simulated with the nominal values. The lateral errors and the steering angle input from the controller during this run can be seen in Figure 3.13.

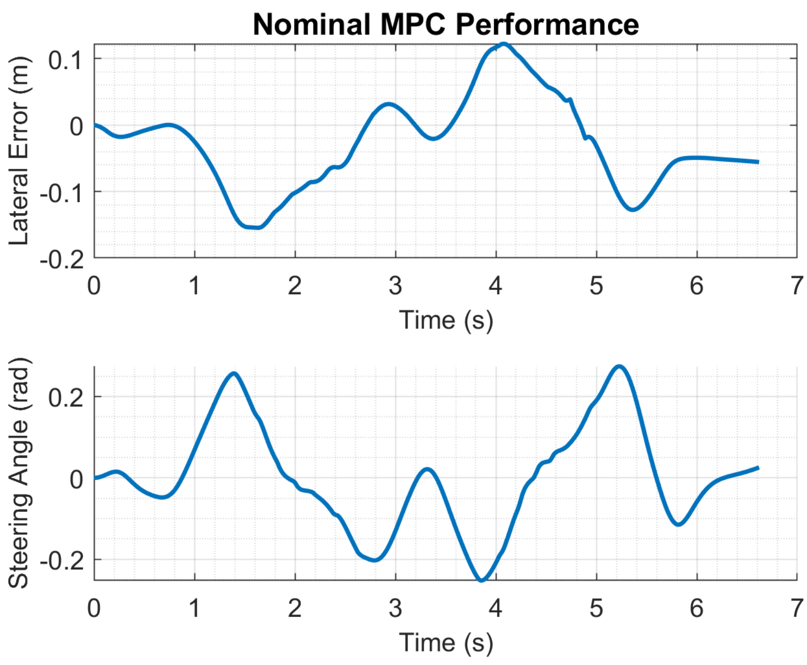


Figure 3.13: Model Predictive Controller Nominal Model Performance.

The model predictive controller was expected to have similar path following performance as the pole placement controller with the nominal model, but much more robust to model inaccuracies. This is because it also uses the same dynamic bicycle model in the prediction, but propagating the states forward when optimizing the steering input should allow it to recover from deviations much quicker. Figure 3.14 below shows the sensitivity of the model predictive controller to inaccuracies of each of the dynamic bicycle model parameters. For an explanation of the plot, refer to Section 3.2.2.

The model predictive controller's errors when simulated with the nominal values can be seen where the x-axis is zero. This was very similar to the pole placement controller and lower than all the other controllers. It performed much better than the pole placement controller and

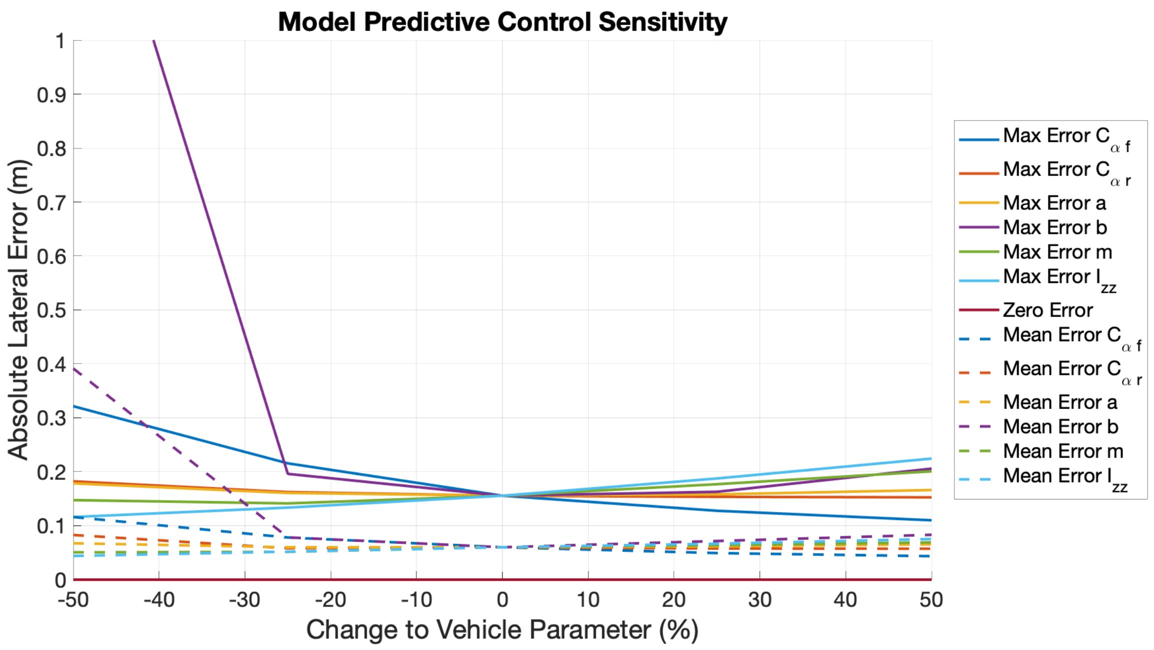


Figure 3.14: Model Predictive Control Sensitivity to Model Inaccuracy.

kinematic controller when the model was perturbed. This is likely because of the prediction step and control optimization, which allows any deviations to be corrected sooner.

The front tire cornering stiffness $C_{\alpha f}$ had a moderate effect on the controller performance when overestimated significantly. This was probably due to the tire not being able to generate enough lateral force through the front tire and understeering. However, even in this case, the vehicle stayed below a third of a meter of lateral error from the path. The rear tire cornering stiffness $C_{\alpha r}$ had almost no effect on the controller performance.

Likewise, uncertainty in the distance from the CG to the front axle a caused no controller performance degradation. In contrast, the distance from the CG to the rear axle caused the vehicle to become unstable when overestimated by a factor of two. When barely overestimated or when underestimated, error in b had very little effect on performance. This was probably due to the car becoming very oversteer when b was much shorter than expected and the model predictive controller prediction step to be unrepresentative of the real dynamics.

The controller was able to handle inaccuracies in the mass m and vehicle inertia about the z-axis I_{zz} very well and very little change in performance were observed due to them. The model predictive controller was resilient to most of the parameter inaccuracies, with the exception of large errors in b. Neglecting that case, the MPC kept the max lateral error less than a third of a meter, which is very good under most circumstances. This placed it as the second most robust controller behind the lookahead error controller. The individual simulation runs and detailed results for the MPC simulation can be found in the appendices in Section A.4.

3.6 Model Reference Adaptive Control

Adaptive control methods are able to adapt the control laws themselves in order to mitigate errors from varying conditions and model uncertainties. They are very common in aerospace applications where the dynamics change significantly as fuel is burned. Additionally, in aerospace there are many high order dynamics and model uncertainties [71]. Adaptive control relies on the Lyapunov stability criterion to derive its update laws.

There are many forms of adaptive controllers but most fall within the categories of indirect and direct adaptive control. Indirect adaptive control relies on a parameter estimation algorithm to learn the dynamic model of the system. As the model of the system is learned, the control law changes because it depends on the model. Direct adaptive control directly modifies the control law without any system identification module [72]. This is the type of adaptive control that is studied in this section.

Model reference adaptive control (MRAC) was the specific form of direct adaptive control implemented. MRAC simulates a reference model and controller that provides the desired closed loop performance. Then, the control inputs determined by this nominal controller are used as the input for the actual uncertain system. The error between the response of the uncertain system and the reference model is used in the adjustment mechanism to adapt the controller to drive the error between the two systems to zero. The high level architecture of model reference adaptive control can be seen in Figure 3.15.

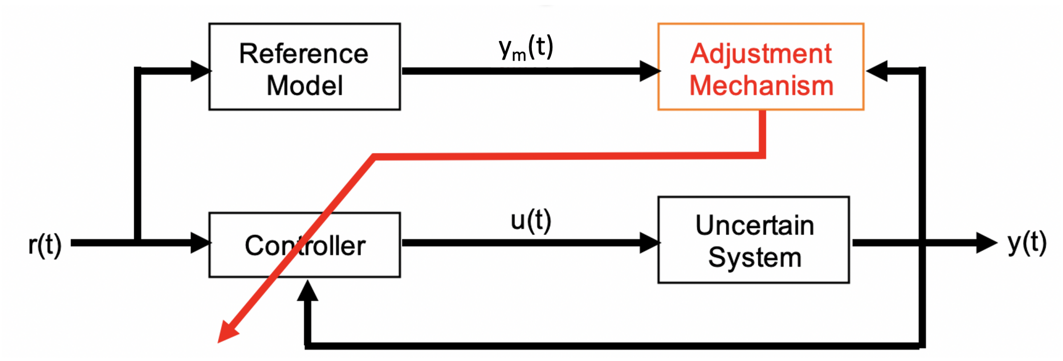


Figure 3.15: Model Reference Adaptive Control.

Model reference adaptive control is advantageous because it can handle high levels of system uncertainty and nonlinearity [73]. Additionally, it does not rely on any parameter estimation algorithm. The MRAC control law can guarantee asymptotic stability and performance, if the assumptions made in the derivation are not violated.

The following sections go through the derivation of the adaptive control law used and present the sensitivity analysis results for the MRAC algorithm.

3.6.1 Control Formulation

To formulate the MRAC controller, first the reference model and nominal controller must be designed. An estimate of the vehicle model (A and B matrices) and pole placement with the desired eigenvalues is used to calculate the nominal controller gains.

$$A_m = A_{est} - B_{est} \cdot K_1 \tag{3.57}$$

$$B_m = B_est \cdot K_0 \tag{3.58}$$

The closed loop response of this system provides the matrices of the reference model. The controller output is given to be $u(t)$, where $R(t)$ is the reference signal (yaw rate command).

$$u(t) = K_0(t) \cdot R(t) - K_1(t) \cdot x(t) \tag{3.59}$$

Next, the error dynamics between the uncertain system and the nominal system are derived.

$$e = x - x_m \tag{3.60}$$

$$\dot{e} = \dot{x} - \dot{x}_m = Ax + Bu - A_m x_m - B_m R \tag{3.61}$$

The nominal controller from above is used to replace the input signal u.

$$\dot{e} = Ax + B(K_0 R - K_1 x) - A_m x_m - B_m R + (A_m x - A_m x) \tag{3.62}$$

$$\dot{e} = A_m(x - x_m) + (BK_0 - B_m)R + (-BK_1 + A - A_m)x \tag{3.63}$$

The following mismatch functions f_1 and f_2 are defined in Equations (3.64-3.65) in order to simplify the expression.

$$f_1 = BK_0 - B_m \tag{3.64}$$

$$f_2 = -BK_1 + A - A_m \tag{3.65}$$

To ensure stability, the error between the uncertain model and reference model must always go to zero. This can be proven for a some example cases. First, if f_1 and $f_2 = 0$, then $\dot{e} = A_m e$ which means e is asymptotically stable, since A_m was designed to be a stable plant. Also, if $f_1 R + f_2 x = 0$, then $\dot{e} = A_m e$. To ensure $e \to 0$, $f_1 \to 0$, and $f_2 \to 0$, Lyapunov's Criterion is applied to find the cases which ensure this stability. For the Lyapunov candidate function, "energy" is chosen:

$$V(e, f_1, f_2) = \frac{1}{2}(e^2 + f_1^2 + f_2^2) \tag{3.66}$$

$$\dot{V} = \frac{dV}{dt} < 0 \Rightarrow \text{energy is always decreasing} \tag{3.67}$$

$$\dot{V} = e\dot{e} + f_1\dot{f}_1 + f_2\dot{f}_2 \tag{3.68}$$

Equation (3.63) is then substituted in for $\dot{e}$.

$$\dot{V} = e(A_m e + f_1 R + f_2 x) + f_1 \dot{f}_1 + f_2 \dot{f}_2 \tag{3.69}$$

$$\dot{V} = A_m e^2 + f_1(\dot{f}_1 + Re) + f_2(\dot{f}_2 + xe) \tag{3.70}$$

If $(\dot{f}_1 + Re)$ and $(\dot{f}_2 + xe)$ are zero, then $\dot{V} = A_m e^2$ which is negative semidefinite. Therefore, the error is bounded. To ensure that the error always goes to zero, Barbalat's Lemma is applied. Barbalat's Lemma states:

- Let $f(t) : R \to R$ be differentiable and has a finite limit as $t \to \inf$. If $\dot{f}(t)$ is uniformly continuous then $\dot{f}(t) \to$ as $t \to \inf$.
- If $\dot{f}(t)$ is bounded then $f(t)$ is uniformly continuous.

Through this Lemma, it can be proven that if $V(t) : R \to R$ is twice differentiable, has a finite limit, and its second derivative is bounded then $\dot{V}(t) \to 0$ as $t \to \inf$. Therefore, in this system, the error will go to zero and f_1 and f_2 are bounded.

To find how to adapt the controller gains, the derivatives of f_1 and f_2 are taken.

$$f_1 = BK_0 - B_m \qquad f_2 = -BK_1 + A - A_m \tag{3.71}$$

$$\dot{f}_1 = B\dot{K}_0 \qquad \dot{f}_2 = -B\dot{K}_1 \tag{3.72}$$

$$\dot{f}_1 + Re = 0 \qquad \dot{f}_2 + xe = 0 \tag{3.73}$$

$$\dot{K}_0 = -\frac{Re}{B} \qquad \dot{K}_1 = \frac{xe}{B} \tag{3.74}$$

Since the system is uncertain, B is unknown, and the adaptation gains are set as a parameter.

$$\dot{K}_0 = -\gamma_0 Re \qquad \dot{K}_1 = \gamma_1 xe \tag{3.75}$$

Finally, to determine the yaw rate command that should be input to the model reference adaptive controller, the same outer loop from the kinematic controller and pole placement controller was used (3.5). It is also shown below for reference.

$$\dot{\psi}_{cmd} = -k_p * e_{LA} + \kappa \cdot V_x \tag{3.76}$$

3.6.2 Results

The model reference adaptive controller had a max error of 0.3995 meters and a mean error of 0.1574 meters when simulated with the nominal values. The lateral errors and the steering angle input from the controller during this run can be seen in Figure 3.16.

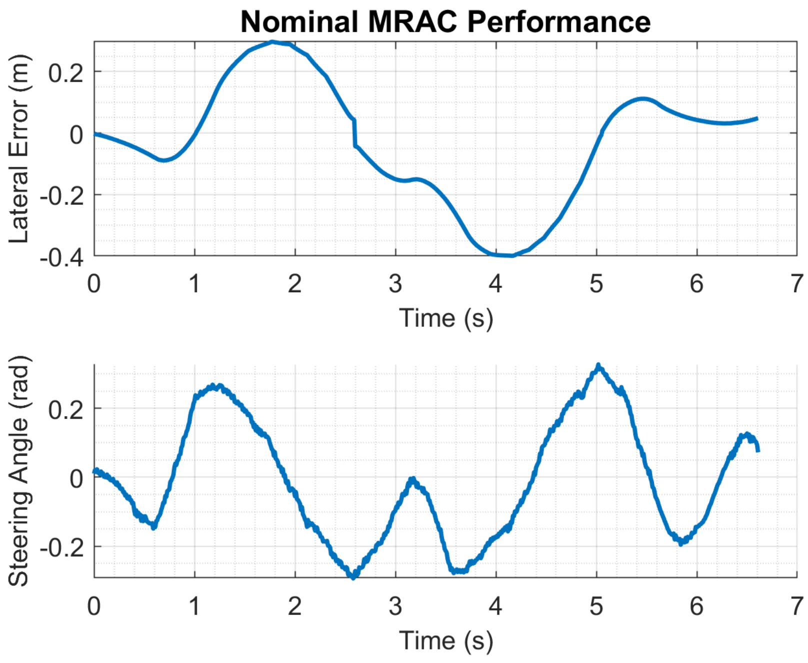


Figure 3.16: Model Reference Adaptive Controller Nominal Model Performance.

The model reference adaptive controller was expected to be stable for all parameter variations, but to have worse path tracking capabilities in the double lane change because the controller requires excitation for the gains to adapt and converge to their steady state values. Figure 3.17 below shows the sensitivity of the model reference adaptive controller to inaccuracies of each of the dynamic bicycle model parameters. For an explanation of the plot, refer to Section 3.2.2.

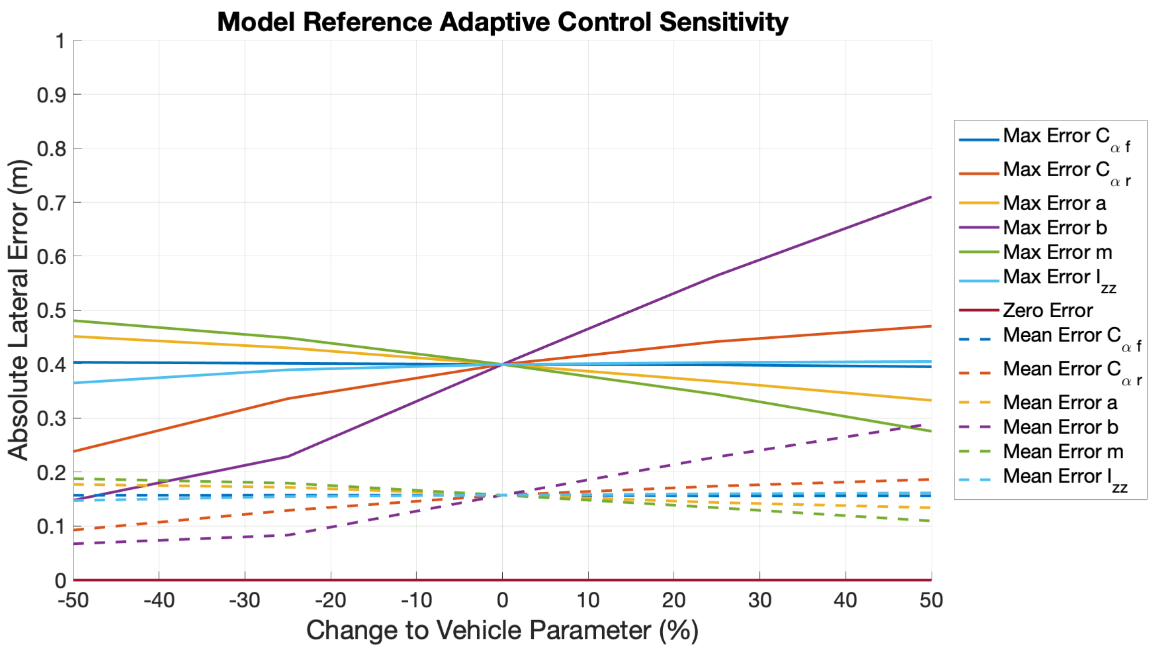


Figure 3.17: Model Reference Adaptive Control Sensitivity to Model Inaccuracy.

The model reference adaptive controller's nominal performance can be seen where the x-axis is zero. This was the highest max and mean error with the nominal parameters out of the tested controllers. The MRAC showed its strength when the parameters were changed, staying stable for each scenario.

The front tire cornering stiffness $C_{\alpha f}$ (darker blue) had almost no effect on the controller performance. The orange line shows the rear tire cornering stiffness $C_{\alpha r}$ which slightly increased the lateral error when underestimated.

The distance from the CG to the front axle a (yellow) had very little effect on the path following performance. The distance from the CG to the rear axle b caused significant error when underestimated as seen by the purple line. However, the error decreased significantly when b was decreased, possibly because an oversteer vehicle produces much more excitation which is necessary for a model reference adaptive controller to work well.

The vehicle mass m (green) and the vehicle inertia about the z-axis (light blue) had very little effect on controller performance. Overall, the only parameter which made a significant difference in controller performance was b, but regardless of the parameter values, the controller always remained stable. The individual simulation runs and detailed results can be found in the appendices in Section A.5.

The lower nominal performance of the model reference adaptive is understandable considering that MRAC is designed to improve over time. As more excitation and time passes, the MRAC is able to refine the controller gains to better make the real system match the reference system. Therefore, a double lane change maneuver may not be enough for the gains to converge to their final values.

In order to give the opportunity for the model reference adaptive controller to converge, another sensitivity analysis was done with the nonlinear bicycle model. For this experiment, a sine wave yaw command input was used instead of a double lane change maneuver. This was chosen because it better simulates a series of excitation so that the controller can fully converge.

Like the prior sensitivity analysis, each vehicle parameter was varied $\pm 50\%$ and the system's performance and stability was analyzed. Under every variation, the controller remained stable and the error converged to zero while under excitation or not. Additionally, since the above sensitivity to b implied that understeer gradient had a large effect, the parameters were adjusted to create a very understeer vehicle and a very oversteer vehicle in order to show the most extreme cases. From the understeer gradient model in Chapter 2, it can be seen that the vehicle will be most understeer when the distance from the CG to the rear axle and the rear tire cornering stiffness is increased, and the distance from the CG to the front axle and the front tire cornering stiffness is decreased. For a vehicle with the most oversteer, the distance from the CG to the rear axle and the rear tire cornering stiffness is decreased, and the distance from the CG to the front axle and the front tire cornering stiffness is increased.

Figure 3.18 below shows the MRAC response to the oversteer vehicle case. In this plot r is the reference yaw rate signal sent to the controller, x_m is the nominal model response, and x is the system response. The controller rapidly drives the error to zero even under such an extreme case. The vehicle was very oversteer in this instance, but the high excitation from the unstable plant quickly drove the controller gains to steady state. The response also converges quickly to the reference model as shown closer in Figure 3.19.

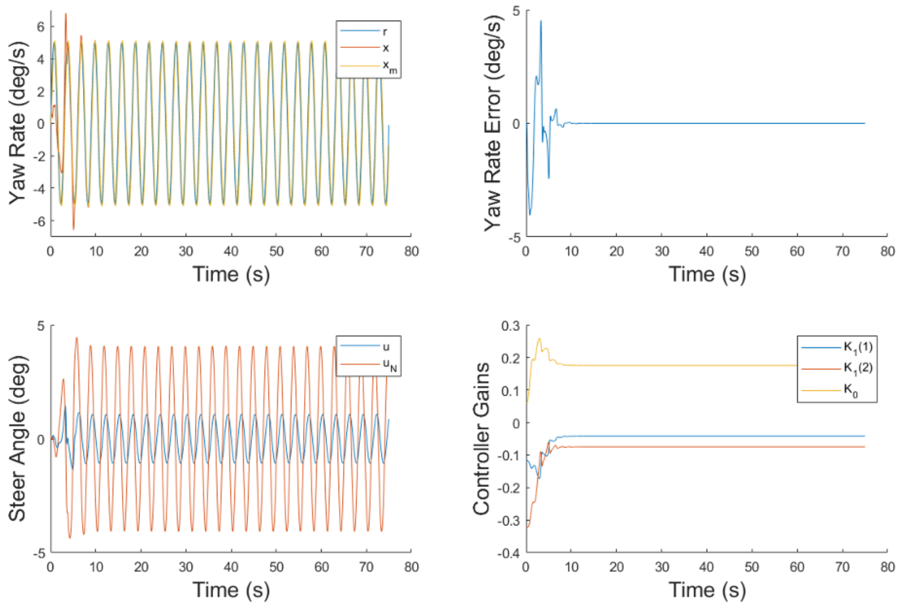


Figure 3.18: MRAC Response to Sine Input for Oversteer Vehicle.

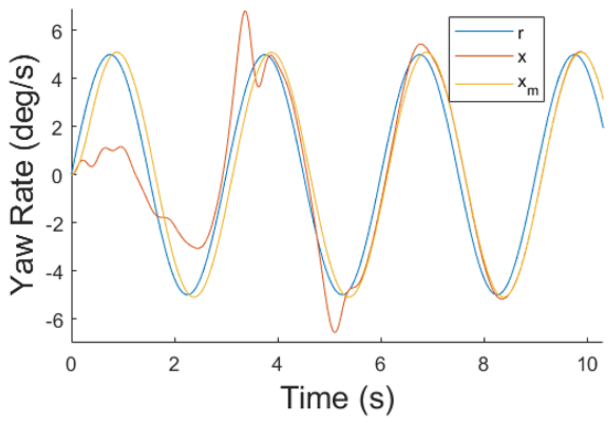


Figure 3.19: MRAC Response to Sine Input for Oversteer Vehicle (Zoomed from Figure 3.18).

Figure 3.20 below shows the MRAC response to understeer vehicle case. The controller eventually drives the error to zero, however, because of the lower excitation from the extremely understeer vehicle, this takes much longer. The controller gains are also much slower to converge. Nevertheless, the response is still driven to the reference model as shown closer in Figure 3.21.

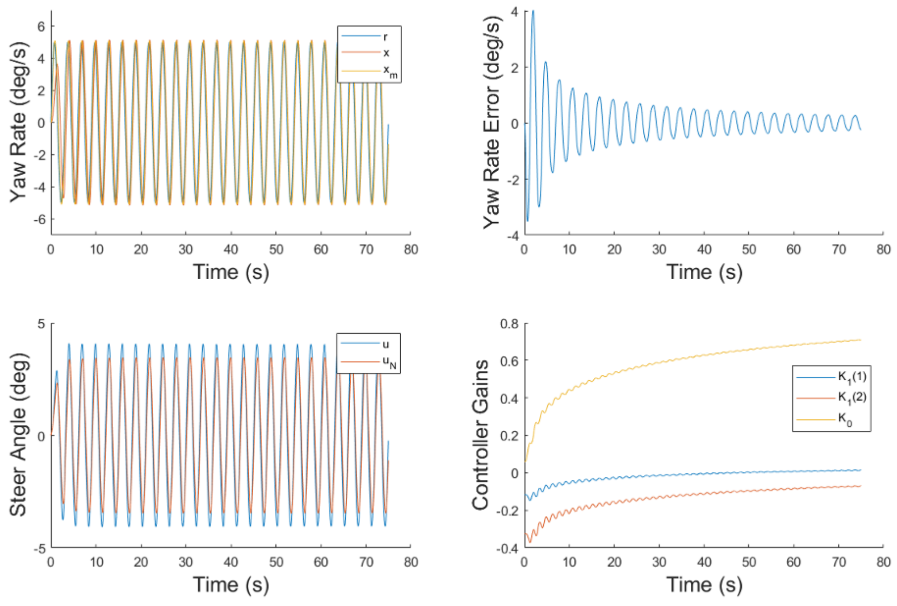


Figure 3.20: MRAC Response to Sine Input for Understeer Vehicle.

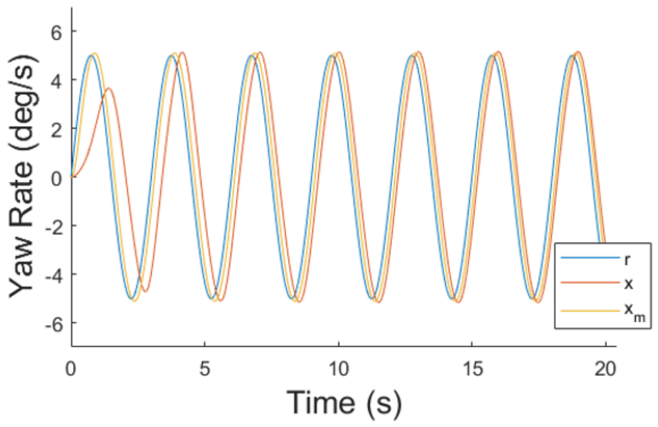


Figure 3.21: MRAC Response to Sine Input for Understeer Vehicle (Zoomed from Figure 3.20).

Overall, the model reference adaptive control worked well. It always maintained stability and was able to drive the error to zero. It performed much better when there is larger and longer excitation as shown in the second portion of the sensitivity analysis. Without strong enough or prolonged excitation, the controller gains and the error do not converge, and the errors are larger as seen in the Figure 3.17.

3.7 Sensitivity Analysis Conclusions

In general, the controllers were most affected by the vehicle parameters that determine the understeer gradient ($a, b, C_{\alpha f}, C_{\alpha r}$). The kinematic and dynamic bicycle model pole placement controllers were very sensitive these parameters as well as mass. The model predictive controller was more robust, but still affected by b and $C_{\alpha f}$. The model reference adaptive controller was robust as well, only being sensitive to b. The lookahead error feedforward-feedback controller performed best overall and was not very sensitive to any parameter.

The kinematic controller remained stable, except when the distance from CG to rear axle b was decreased by 50%. This implies that kinematic control may not work well when controlling an oversteer vehicle in transient maneuvers. The other parameters increased the lateral error, but remained stable and followed the path well. The kinematic controller seemed to achieve the commanded yaw rate well, with a short time delay.

The dynamic bicycle model pole placement controller was better able to account for the transience of the double lane change and provided lower nominal errors, but this came at a cost. The pole placement controller was the most sensitive to parameter uncertainties and had huge spikes in error when the model was incorrect. For acceptable performance with the classical pole placement controller, either an accurate model or a lot of tuning is necessary.

The lookahead error feedforward-feedback controller's robustness was extremely impressive. This makes sense when looking at its formulation, since only the feedforward term was directly calculated using the vehicle model. The feedback term was calculated using the lookahead error, which allows for both the lateral error and heading error to be taken into consideration. The controller was hand tuned using the nominal model, but the performance did not degrade much with the varied models. However, if the vehicle type was changed drastically (e.g. from a passenger car to a tractor trailer), this may not be the case.

With the model predictive controller, the front tire cornering stiffness was only detrimental when overestimated because the tires could not handle the lateral forces required by them. When the vehicle became oversteer from distance b being decreased by 50%, the MPC had the potential to become unstable. The model predictive control was very robust to mass, inertia, distance a, and rear tire cornering stiffness. This is due to the controller recalculating new optimal control inputs at every time step as the vehicle response does not match the prediction, correcting any variation from the path.

The model reference adaptive control had higher nominal errors but was able to maintain stability under every scenario. The only parameter that had much effect was the distance b, but the MRAC was still able to follow the path and showed no signs of instability. In the additional simulation run with the sine input, it was clear that excitation levels have an extremely significant bearing on the performance of a model reference adaptive controller. An unstable plant such as an oversteer vehicle works better than an understeer vehicle because the extra excitation causes the controller gains to converge quicker.

The sensitivity analysis provided in this chapter was able to provide a great deal of insight into the advantages and disadvantages of each type of controller tested. The simulation was a low fidelity environment with a highly idealized situation allowing to quickly test the various control architectures. In the real world, unmodeled dynamics and disturbances are always present, and there is usually a combination of parameter uncertainties. Nevertheless, this data was extremely useful in informing the formulation of the vehicle agnostic path following controller designed in Chapter 4 and tested in Chapter 5.

Chapter 4

Vehicle Agnostic Path Following Controller

The primary goal of this thesis was to design a vehicle path following controller that is able to work on a wide variety of platforms. Some of the controllers in the sensitivity analysis were able to handle small amounts of model inaccuracy, but would not be able to handle vehicles that varied drastically. However, human drivers are able to go from driving a small sports car to a large truck without having any knowledge of the vehicle models. For the formulation of the vehicle agnostic controller presented in this chapter, the driving behaviors of humans was mimicked in the controller logic.

The human driving dynamics can be simplified into two components. First, the controller should only need very limited vehicle information to be able to control a wide variety of vehicles. Second, the controller should have similar path following dynamics and smoothness as a human. The following section discusses the formulation and decision making process behind the vehicle agnostic controller.

4.1 Control Formulation

To achieve the desired human driving characteristics, the path dynamics and the vehicle yaw dynamics of the vehicle were separated. The controller employs a cascaded architecture with the adaptive path dynamic controller on the outer loop and the adaptive yaw dynamic controller on the inner loop. This can be seen in Figure 4.1.

The cascaded architecture isolates the vehicle dynamics from the path dynamics. Since the path dynamics are vehicle independent, they can be set to match the path dynamics of typical human driving. Similar to the lookahead error feedforward-feedback controller in Chapter 3,

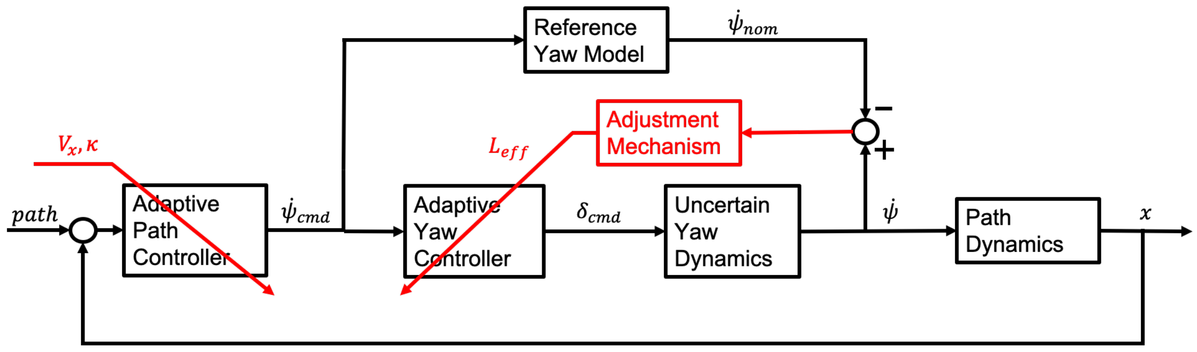


Figure 4.1: Path Following Controller Block Diagram.

the path dynamic controller relies on the lookahead error and the curvature of the reference path. The lookahead error can be defined in terms of lateral error and heading error as shown previously in Equation (3.24) and restated below.

$$e_{LA} = e_{lat} + x_{LA} \cdot \Delta\psi \tag{4.1}$$

If the reference path is rotated into the vehicle frame, the lookahead error is simply the y-component of path point where the $x_{path} = x_{LA}$. The curvature of the path is equal to the change in heading in two path points divided by the arc length between the two points. This can be approximated as shown below.

$$\kappa = \frac{\psi_b - \psi_a}{\sqrt{x_b - x_a + y_b - y_a}} \tag{4.2}$$

In order to mimic human driving behaviors, the lookahead distance x_{LA} is calculated based on the vehicle speed and reference path profile. The lookahead distance acts like damping in the system and is increased as a function of longitudinal velocity. This replicates how a driver on the interstate looks further ahead than they do when driving slowly through a parking lot. This formulation also helps to reduce the lag in vehicle response as the speed increases.

Hingwe and Tomizuka proposed a controller that calculates the optimal lookahead distance as a quadratic function of longitudinal velocity [36]. This holds at higher speeds, but at lower speeds a linear relation achieved better results. Therefore, in this thesis the lookahead distance was calculated as shown in Equation (4.3).

$$x_{LA}(V_x) = \begin{cases} 0.05 \cdot V_x^2, & \text{if} \quad V_x > 15\frac{m}{s} \\ 0.75 \cdot V_x, & \text{otherwise} \end{cases} \tag{4.3}$$

Additionally, the lookahead distance is increased as the curvature of the path decreases. This is similar to how human drivers should look further ahead on a straight road than a tight, twisty one. To replicate this behavior, when the path curvature is below a threshold that is approximately straight, the lookahead distance calculated above is increased by 25%.

The path dynamics controller then calculates the yaw rate needed to follow the curvature of the road under steady state cornering $\dot{\psi}_{FFW}$. This is done by multiplying the curvature at the point nearest to the vehicle with the vehicle's longitudinal velocity V_x.

$$\dot{\psi}_{FFW} = \kappa \cdot V_x \tag{4.4}$$

To correct for deviation from the reference path due to the transience of a maneuver or other disturbances, a corrective feedback yaw rate, $\dot{\psi}_{FB}$, is calculated by multiplying the lookahead error by a proportional gain.

$$\dot{\psi}_{FB} = -K_p \cdot e_{LA} \tag{4.5}$$

The feedforward and feedback terms are added together to determine the total commanded yaw rate, $\dot{\psi}_{cmd}$.

$$\dot{\psi}_{cmd} = \dot{\psi}_{FFW} + \dot{\psi}_{FB} \tag{4.6}$$

This calculated yaw rate is then sent to the inner loop which calculates the steering angle command needed to achieve the desired yaw rate. This yaw rate controller must be able to handle the vehicle dynamics and adapt to completely different vehicle types. Initially, for the inner loop, the model reference adaptive control from Section 3.6 was implemented in ROS/C++ and tested in Gazebo.

However, moving the algorithm to a higher fidelity simulation brought to light some weaknesses. First, without a direct measurement of the lateral velocity, the MRAC reference model and the uncertain vehicle model did not closely match. Also, under normal driving there is not persistent excitation that is necessary for the controller gains to converge quickly. Because of these issues and the inherent unstructured uncertainties, the model reference adaptive controller sometimes became unstable after a certain amount of time.

To combat these issues and the reliance on a dynamic bicycle reference model, an adaptive controller based on a simpler model was designed to control the yaw rate. It uses the same principles as the previous model reference adaptive controller, but employs the kinematic bicycle model instead of the dynamic bicycle model. It also uses indirect adaptive control where a parameter is estimated in order to adapt the controller, instead of direct adaptive control where the controller gains are adapted directly with no estimation algorithm. This was done so that the controller could be validated with some intuition.

The kinematic model only relies on the wheelbase length and steering ratio, instead of the six parameters of the dynamic bicycle model. This greatly reduces the difficulty in getting a good estimate of the model. It is also easy to take an accurate measurement of wheelbase length to initialize the system. The indirect form of MRAC used for the yaw control estimates an effective wheelbase length, L_{eff}, in order to adapt the kinematic controller. This effective wheelbase is used to estimate the DC gain of the uncertain system and is essentially a function of the wheelbase, steering ratio, understeer gradient, and disturbances.

To calculate the steering command for a given yaw command, first, the steering angle from the nominal yaw controller is calculated as shown below in Equation (4.7).

$$\delta_{cmd} = \arctan \frac{L_{eff} \cdot \dot{\psi}_{cmd}}{V_x} \tag{4.7}$$

Next, the predicted nominal response is represented as a first order delay with DC gain of one using the previous commanded yaw rate $\dot{\psi}_{cmd}^{-}$, the previous actual yaw rate $\dot{\psi}_{cmd}^{-}$, and a filter coefficient C_{delay}. This coefficient is a function of the time step T_s and the filter time constant τ as shown below.

$$C_{delay} = \frac{T_s}{\tau} \tag{4.8}$$

The nominal yaw rate at the given time step is then calculated with the difference equation shown below, which represents the delay in the steering actuator and vehicle yaw dynamics.

$$\dot{\psi}_{nom} = \dot{\psi}^{-}_{act} + (\dot{\psi}^{-}_{cmd} - \dot{\psi}^{-}_{act}) \cdot C_{delay} \tag{4.9}$$

$$\dot{\psi}_{nom} = \dot{\psi}^{-}_{act}(1 - C_{delay}) + \dot{\psi}^{-}_{cmd} \cdot C_{delay} \tag{4.10}$$

The error between the actual model and the reference model is represented by $e_{\dot{\psi}}$.

$$e_{\dot{\psi}} = \dot{\psi}_{act} - \dot{\psi}_{nom} \tag{4.11}$$

This error between the models is multiplied by the adaptation gain, γ, which is a tuning parameter that determines how quickly the system adapts the effective wheelbase. This equation, represented below, results in the change in effective wheelbase ΔL_{eff}.

$$\Delta L_{eff} = -\gamma \cdot e_{\dot{\psi}} \tag{4.12}$$

The change in effective wheel base is then added to the previous estimate to achieve the new estimate. The process then starts over by calculating the next steering command with Equation (4.7) using the updated effective wheelbase.

$$L_{eff} = L^{-}_{eff} + \Delta L_{eff} \cdot T_s \tag{4.13}$$

4.2 Software Architecture

This system was implemented using the Robotic Operating System (ROS) in C++ and Python. Figure 4.2 shows the ROS nodes and communication channels.

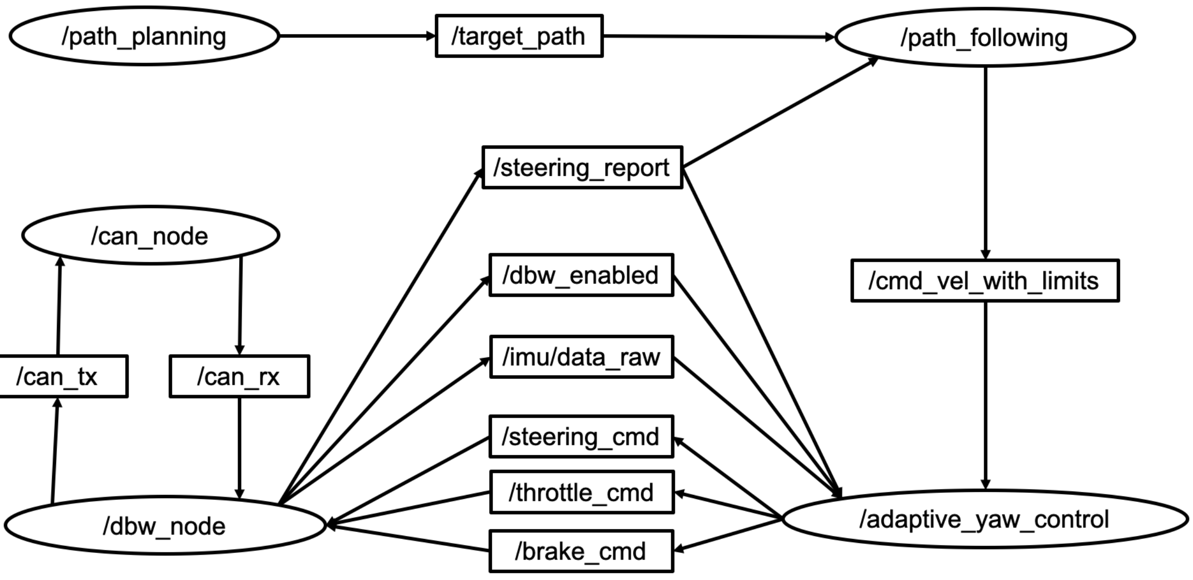


Figure 4.2: Vehicle Agnostic Controller ROS Software Architecture.

First, the reference path in the vehicle frame is sent from the path planning node to the adaptive path following controller node. The path following node calculates the desired yaw rate which is then sent to the adaptive yaw controller node. The adaptive yaw control node calculates a commanded steer angle and sends it to the DBW node.

The DBW node converts the vehicle commands to CAN messages and sends them to the vehicle through the Controller Area Network (CAN) bus, utilizing the original equipment manufacturer steering motor, throttle actuator, and brake actuator. The DBW node also receives CAN messages containing important vehicle state information, which it converts to a more usable form to be used by the control nodes.

This modular architecture allows the vehicle agnostic controller to be used for different applications by only switching out the path planning node. The controller can also be quickly implemented on any vehicle with a ROS interface to the CAN bus with drive-by-wire capabilities and access to vehicle states. In the next chapter, the controller is used on four different vehicles in simulation as well as three different platforms in experimental testing with no changes made to the controller between the applications.

Chapter 5

Simulation and Experimentation

This chapter presents the simulation and experimental results for the vehicle agnostic path following controller presented in Chapter 4. The first section presents the setup and results of the controller when tested on four vehicles in the simulation environment. The next section discusses the experimental setups for the real-time implementation on the drive-by-wire test vehicles. This is followed by analysis of the behavior and error dynamics of typical human driving for comparison to the controller. The fourth section presents the two reference path generation techniques used for the real-world testing. The last section presents the experimental results for both test scenarios.

5.1 Simulation

This section describes the vehicles, environment, and results of the vehicle agnostic path following controller when tested in simulation.

5.1.1 Simulation Vehicles

Software-in-the-loop (SIL) testing was performed in order to ensure the control architecture would perform well before being implemented on the MKZ in real time. Dataspeed provides a Gazebo vehicle model and ROS interface that mimics that of the actual car, allowing for seamless transition between simulation and real-world testing. Figure 5.1 shows the ROS interface used in simulation. This is nearly identical to the one used on the DBW vehicle, allowing the same control stack to be used on both with no modification.

One difference between the two interfaces is the /gazebo node which replaces the /can_node and controls the car based on the CAN messages. The /gazebo node also publishes simulated sensor data and ground truth data. The only other difference is that the received and transmitted CAN messages (/can_bus_dbw/can_rx and /can_bus_dbw/can_tx) are in the /vehicle namespace. This SIL testing allows the majority of software errors to be discovered before being implemented on the vehicle.

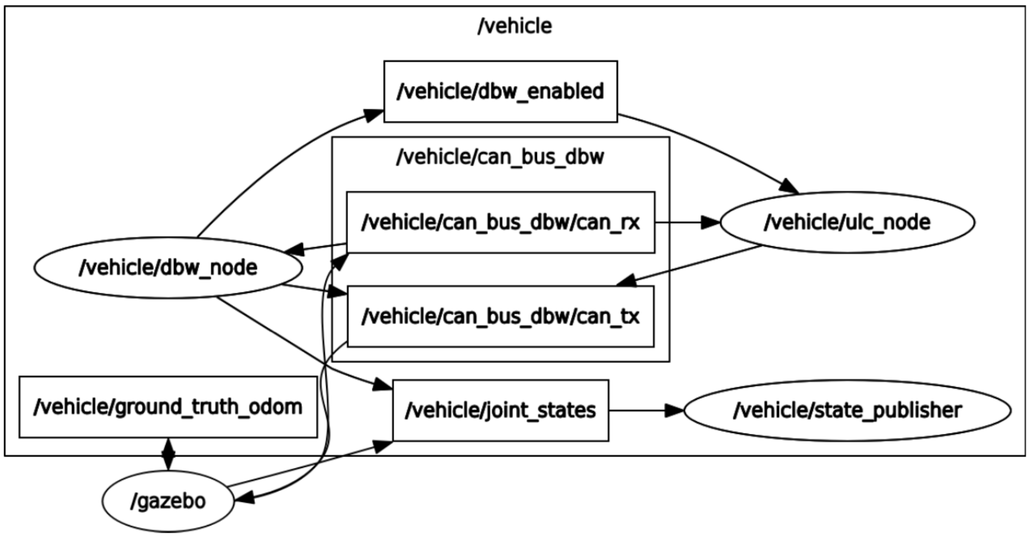
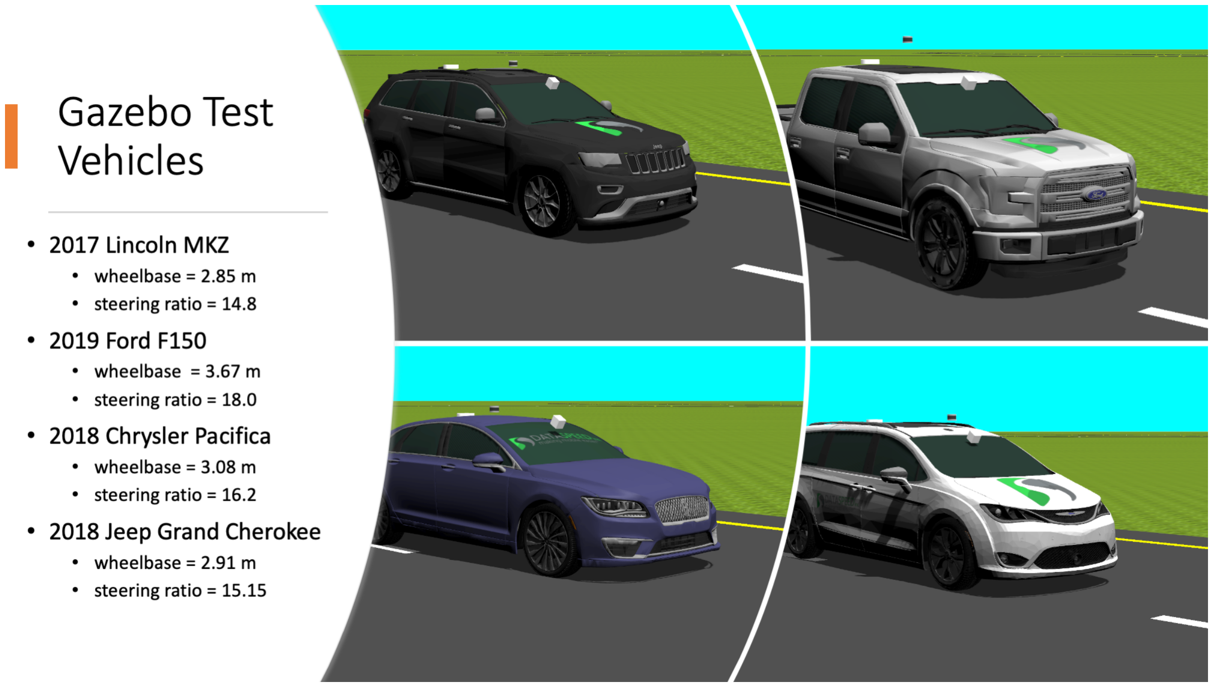


Figure 5.1: Gazebo MKZ DBW ROS node.

To test how the controller would perform on a variety of vehicle platforms, three other Dataspeed Gazebo models were used in addition to the MKZ. These models were the Ford F150, Chrysler Pacifica, and Jeep Grand Cherokee. The Gazebo models and some basic vehicle information can be seen in Figure 5.2. These models were chosen because they represent various types of vehicles: sedans, trucks, minivans, and SUVs.



Gazebo Test Vehicles

- 2017 Lincoln MKZ
  - wheelbase = 2.85 m
  - steering ratio = 14.8
- 2019 Ford F150
  - wheelbase = 3.67 m
  - steering ratio = 18.0
- 2018 Chrysler Pacifica
  - wheelbase = 3.08 m
  - steering ratio = 16.2
- 2018 Jeep Grand Cherokee
  - wheelbase = 2.91 m
  - steering ratio = 15.15

Figure 5.2: Gazebo Models used for SIL Testing.

5.1.2 Simulation Environment

The environment used to test the controller is a Gazebo world with a test track consisting of right and left turns of various radii. The track contains markings for $12 foot (3.66m)$ wide lanes. A section of the track can be seen in Figure 5.3. The reference path for the controller input is generated by the Dataspeed lane detection system that creates a path in the center of the lane and publishes it with the ROS message type `nav_msgs/Path`. This message structure consists of multiple poses with type `/geometry_msgs/Pose`, each corresponding to a specific location along the path. Each pose contains the x, y, z position in the vehicle frame and quaternion orientation. This can be seen from the simulated camera's perspective in Figure 5.4. The red and green lines represent detected lane lines, while the thin pink line is the generated reference path for the control system.

The vehicle can also be visualized relative to the reference path in RViz, a 3D visualization tool in ROS [74]. This can be seen in Figure 5.5 and was useful for the simulation and real world testing in this chapter.

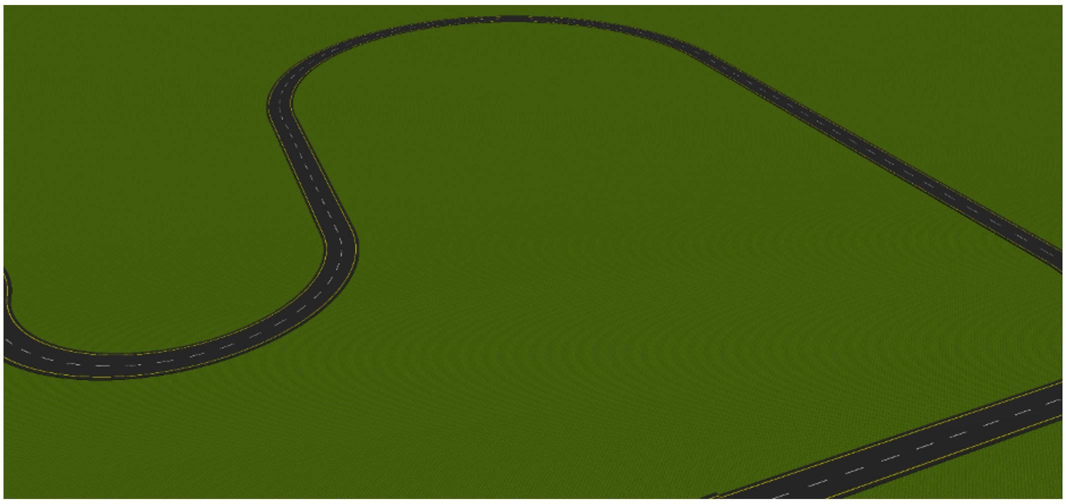


Figure 5.3: Portion of Gazebo Simulation Test Track.

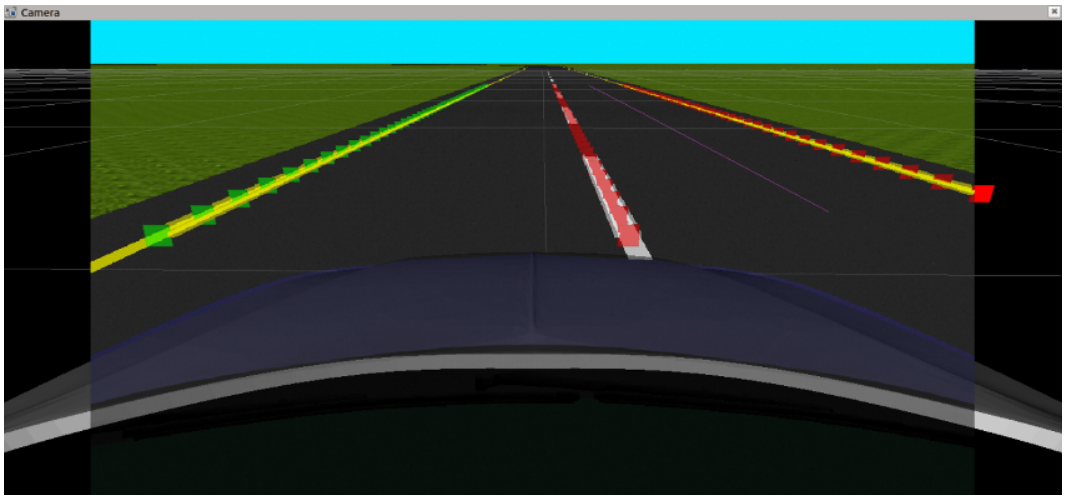


Figure 5.4: Gazebo Reference Path Generation.

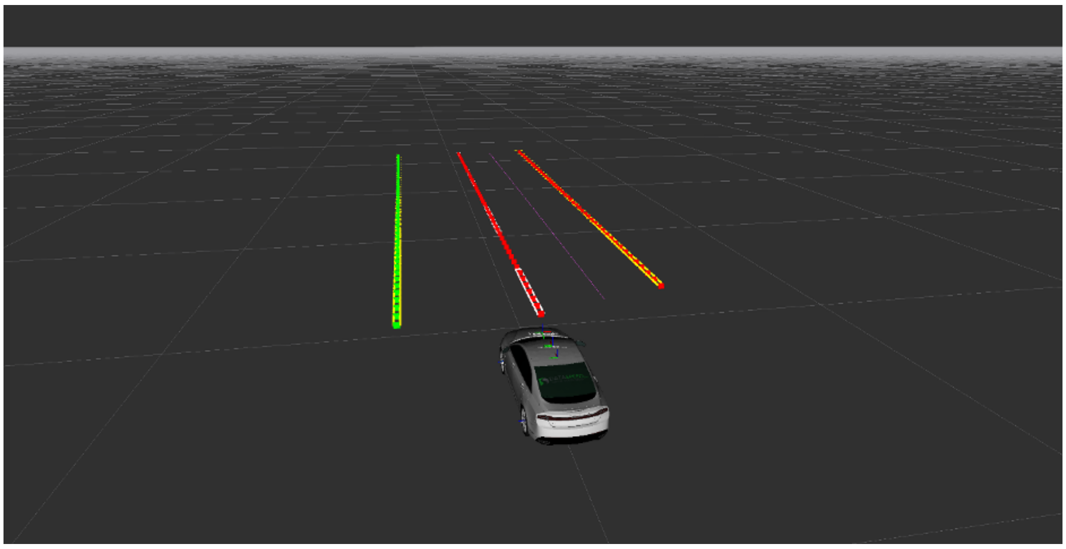


Figure 5.5: RViz Visualization of Vehicle with Path.

5.1.3 Results

The vehicle was run around the entire Gazebo track at constant speeds with runs from 5 m/s to 17.5 m/s. The controller was used throughout the entire run. To determine controller tuning, the proportional gain in the outer loop was tested at a range of values to minimize the lateral error. The final value used for this for this gain was 0.15. The adaptive gain was set to 10, a value high enough so the controller would converge within a couple turns, but low enough that the changes were gradual. This prevented the car balance from being upset mid-corner. The first order delay time constant τ of the yaw dynamics reference model was set to a conservative 0.1 seconds. The adaptive gain can be increased and the time constant decreased if more aggressive control is desired.

The plots below (Figures 5.6-5.10) show the lookahead error at the closest path point in the camera's field of view. The lookahead distance at this point is approximately 8 meters, and therefore, the lateral error shown is much higher than the true lateral error at the CG. Although this value is not the true error, it provides insight into how the vehicle is performing and serves as a baseline to compare directly against the on-road lane keeping tests where there is no truth value available.

The controller was tested with each of the four vehicles at each of the six speeds for five minutes. The effective wheelbase was initialized at 2 meters for each run to determine if the effective wheelbase would be able to converge to the true value for each scenario. The controller assumes a steering ratio of 14.8 (the steering ratio of the MKZ), so the effective wheelbase of the Lincoln MKZ should match the wheelbase, or 2.85 m, if the understeer gradient is neglected. The other models had different steering ratios so the expected effective wheelbase (again neglecting understeer gradient) would be equal to the wheelbase multiplied by the steering ratio and divided by 14.8 as shown in Equation (5.1).

$$L_{eff} = \frac{L \cdot \text{steering ratio}}{14.8} \tag{5.1}$$

The Ford F150 has a wheelbase of $3.67m$ and a steering ratio of 18.0, so the truth effective wheelbase should be $4.463m$. The Chrysler Pacifica has a wheelbase of $3.08m$ and a steering ratio of 16.2, so the truth effective wheelbase should be $3.371m$. Lastly, the Jeep Grand Cherokee has a wheelbase of $2.91m$ and a steering ratio of 15.15, so the truth effective wheelbase should be $2.978m$.

Figure 5.6 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 5 m/s. The bottom subplot shows that the effective wheelbase for each vehicle reached the truth value (dotted lines) within a few turns. The initial plateau from 0 to 40 seconds was due to a lack of excitation along the straight section of the track that the vehicle started on. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1 meter for all vehicles. The small steady state error was likely due to unmodeled dynamics which are treated as disturbances to the controller, but this had little effect on the performance. Qualitatively, the vehicles all stayed inside the lane lines, even before the effective wheelbases converged. Figure 5.7 shows that the adaptive yaw controller was able to follow the commanded yaw rate very closely for the duration of the runs, especially once the effective wheelbase converged.

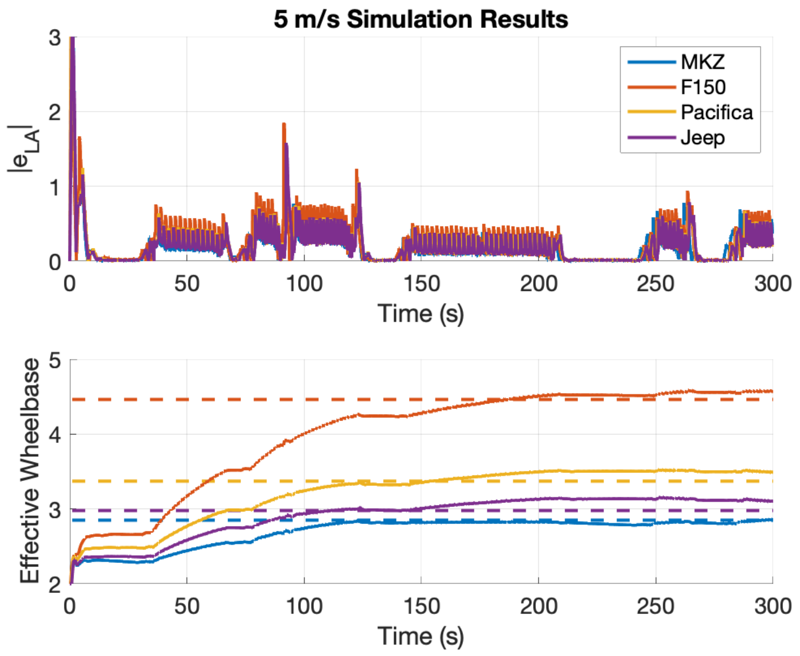


Figure 5.6: Simulation Lookahead Error at 5 m/s.

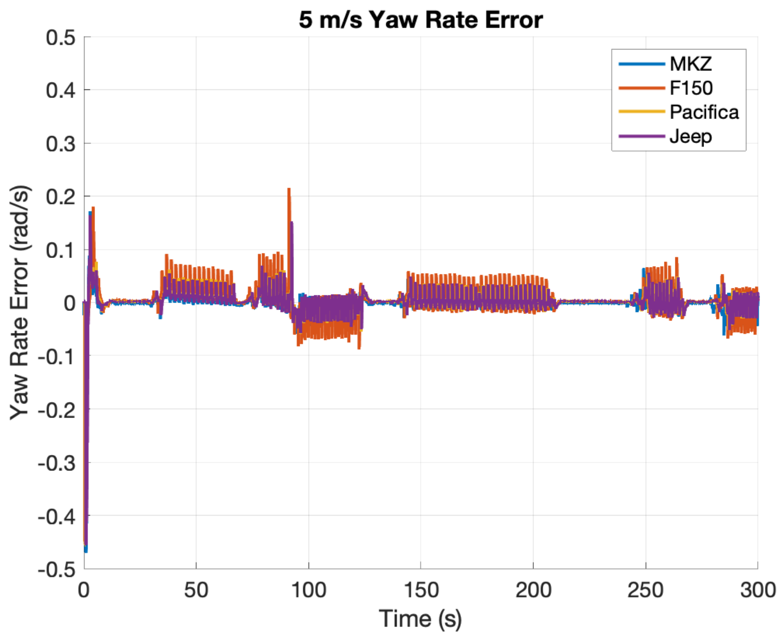


Figure 5.7: Simulation Yaw Rate Error at 5 m/s.

Figure 5.8 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 10 m/s. The effective wheelbase for each vehicle reached the steady state truth value within a few turns. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1.7 meters for all vehicles. Qualitatively, the vehicles all stayed inside the lane lines throughout the runs. Figure 5.9 shows that the adaptive yaw controller was able to follow the commanded yaw rate closely for the duration of the runs.

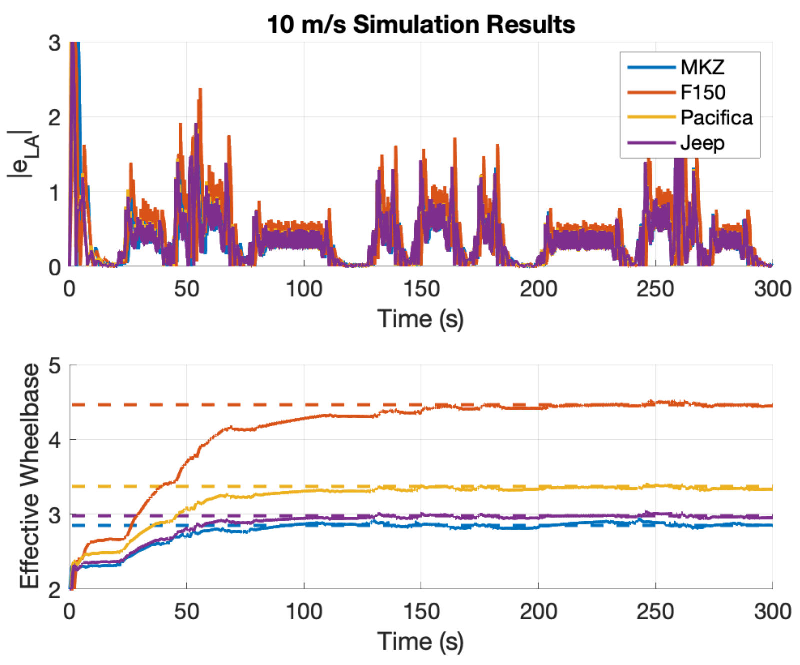


Figure 5.8: Simulation Lookahead Error at 10 m/s.

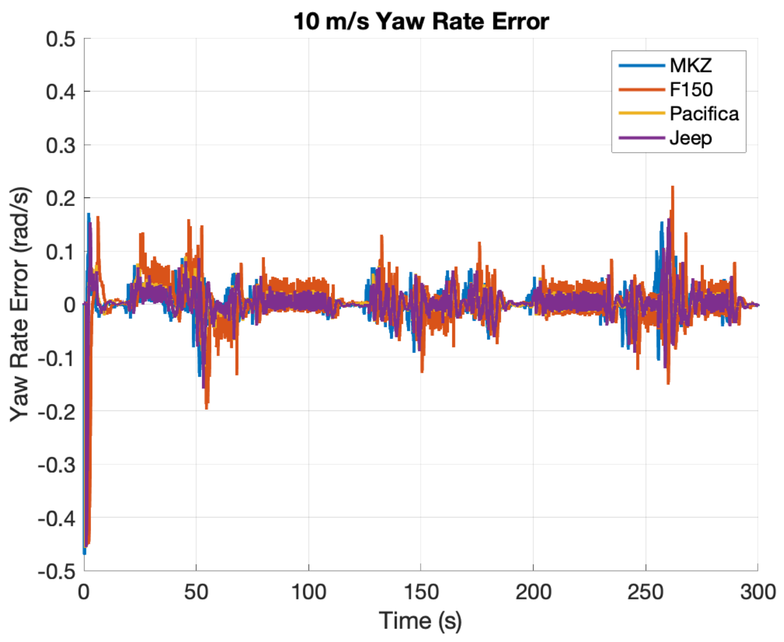


Figure 5.9: Simulation Yaw Rate Error at 10 m/s.

Figure 5.10 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 15 m/s. The bottom subplot shows that the effective wheelbase for each vehicle reached the truth value within a few turns. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1.7 meters for all vehicles. Qualitatively, the vehicles all stayed inside the lane lines, even before the effective wheelbases stabilized. Figure 5.11 shows that the adaptive yaw controller was able to follow the commanded yaw rate closely for the entire runs.

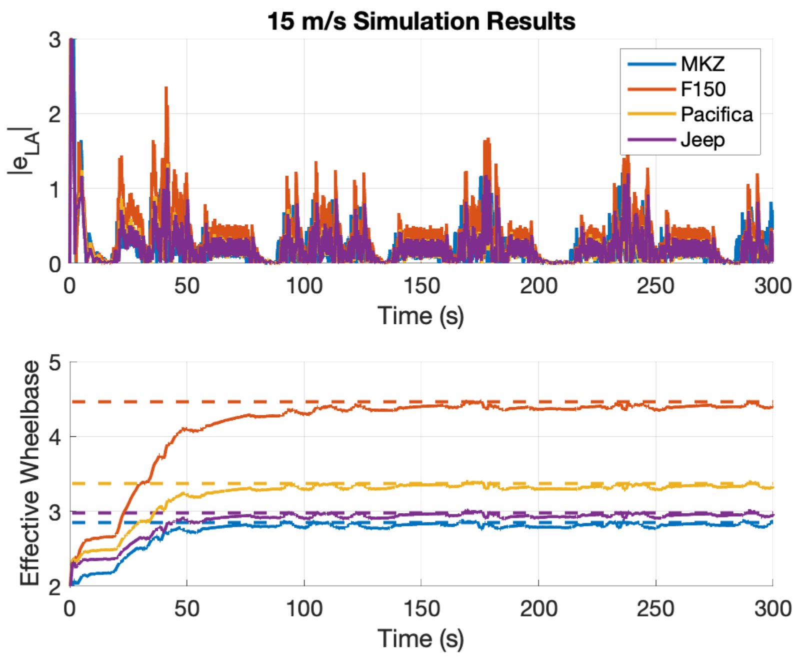


Figure 5.10: Simulation Lookahead Error at 15 m/s.

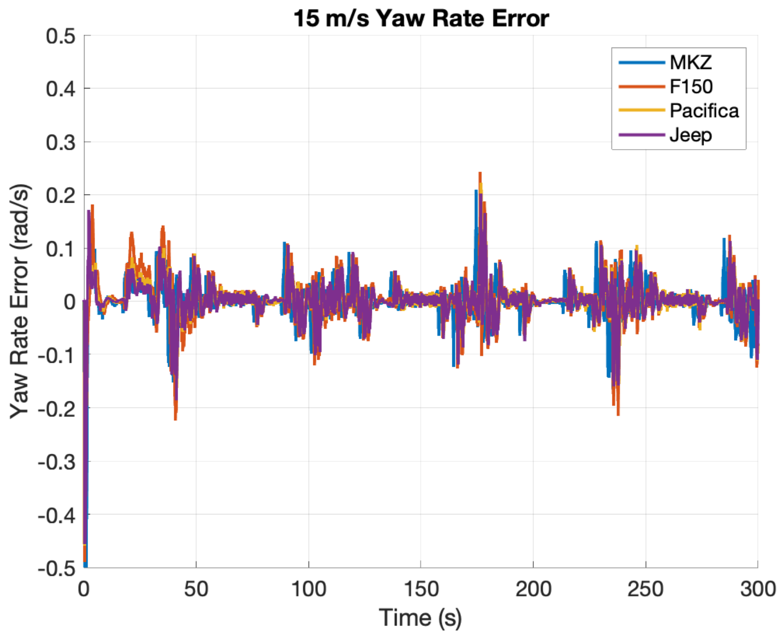


Figure 5.11: Simulation Yaw Rate Error at 15 m/s.

Additional plots and analysis of the simulation runs performed at 7.5 m/s, 12.5 m/s, and 17.5 m/s can be found in Appendix B. The root mean square error (RSME) of the vehicle yaw rate to the commanded yaw rate was also calculated for each vehicle. Figure 5.12 shows that at 5 m/s the RSME yaw error was around 0.025-0.031 rad/s for each of the vehicles. At the other end of the spectrum, at 17.5 m/s the RSME yaw error was between 0.042 and 0.045 rad/s.

These yaw rate errors are good considering the range of vehicles and velocities, and the lookahead errors were comparable to the lookahead errors observed during the human driving analysis. These tests also showed that the controller was able to handle an effective wheelbase initialization that was very far from the truth value. Nevertheless, the closer the initialization was to the true value, the quicker the effective wheelbase reached its steady state value.

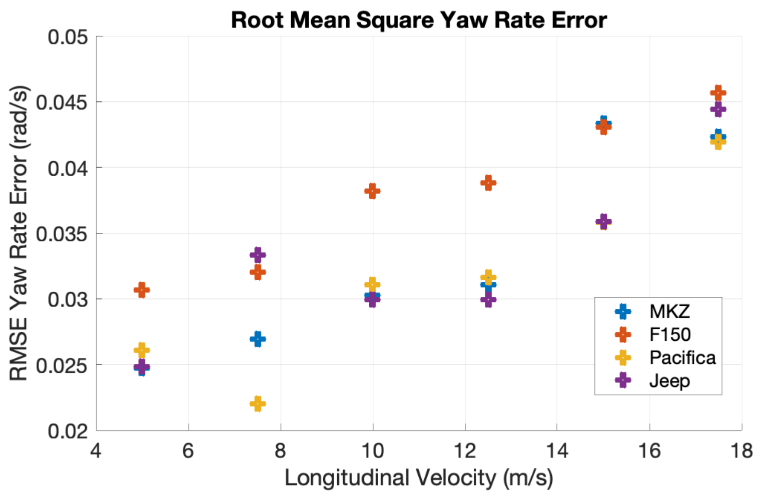


Figure 5.12: Simulation Yaw Rate Root Mean Square Error.

5.2 Experimental Setup

Three drastically different vehicle platforms were used to test the vehicle agnostic path following controller. This was done to validate the flexibility of the controller on vehicles ranging from a small RC car up to a class-8 truck. This section goes through the hardware and software setup of each platform.

5.2.1 Lincoln MKZ

One platform used for the real-world experiments is a Dataspeed 2017 Lincoln MKZ [75] as shown in Figure 5.13. This vehicle is fully drive-by-wire (DBW), meaning the steering, throttle, brake, and gear can all be controlled electronically. Dataspeed also provides a ROS software interface to the vehicle CAN bus. The CAN interface allows vehicle sensor data and states to be read directly from the vehicle engine control unit (ECU). It also allows control signals to be sent to the OEM actuators for steering, throttle, and brake, without having to add any aftermarket actuators.

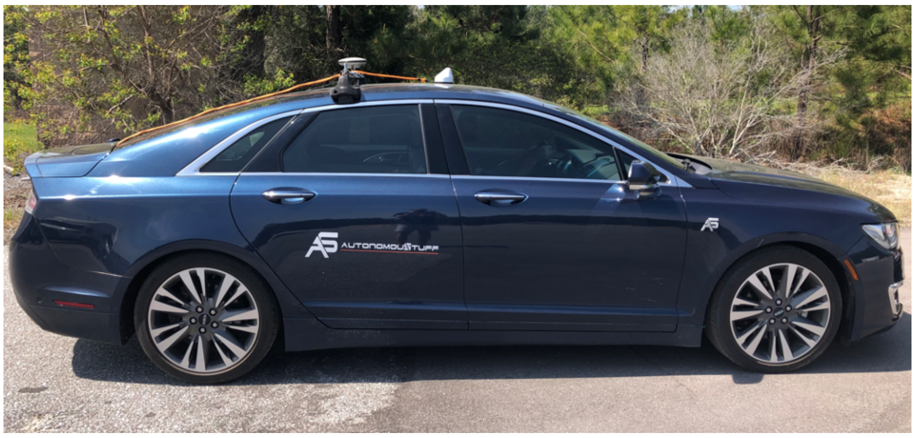


Figure 5.13: Dataspeed Lincoln MKZ.

The ROS graph for the CAN interface can be seen below in Figure 5.14. The gateway to the vehicle is the `/vehicle/dbw_node`, which sends and receives CAN messages. Vehicle sensor data such as wheel speeds, vehicle IMU data, and vehicle GPS data is received through the `/can_bus_dbw/can_rx` topic and is converted to easy to use forms in the /vehicle/dbw_node. Other information from the MKZ actuators, such as steering angle, steering velocity, throttle position, brake position, and gear selection, are handled in the same way. Commands published from the control software stack are sent to the vehicle/dbw_node and converted to CAN messages. These are then published on the /can_bus_dbw/can_tx topic. These CAN topics are sent and received from the /can_node which represents the physical hardware connection with the vehicle CAN port.

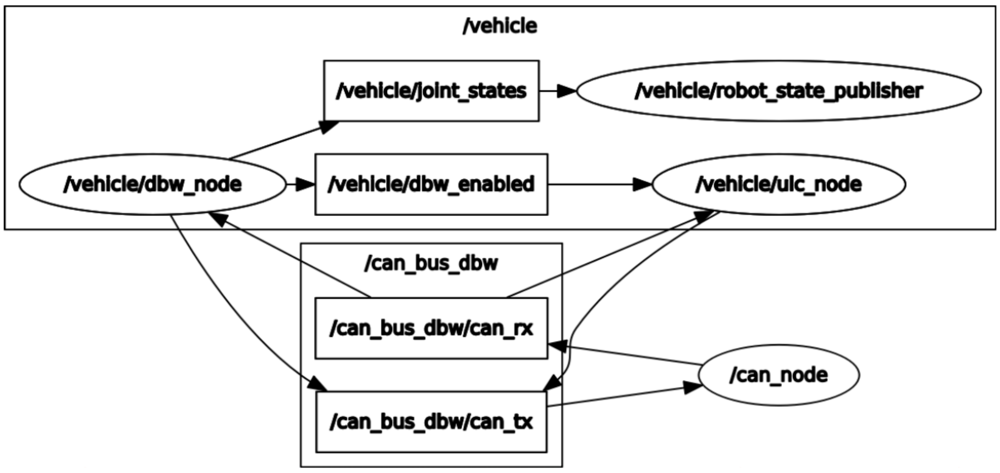


Figure 5.14: MKZ DBW ROS Interface.

The /vehicle/ulc_node is unused and deactivated since all the control commands are sent directly from the controller. Any time the drive-by-wire capabilities are being used for longitudinal or lateral control, the ROS topic /vehicle/dbw_enabled must be set to true so that the /vehicle/dbw_node will send commands.

The sensor suite used for the following tests with the Lincoln MKZ includes a Memsense 3020 IMU, NovAtel GPS receiver, and Intel RealSense D435 camera. Each of these is connected to a central Linux computer in the trunk of the MKZ. The CAN gateway is also attached to the main computer. The RealSense Camera is mounted on the roof of the car along the vehicle x-axis. The NovAtel receiver is in the trunk of the vehicle, while the antenna is mounted to the vehicle's roof. This setup can be seen in Figure 5.15. For vehicle following, a Cohda Wireless DSRC radio can be used for vehicle-to-vehicle (V2V) communication. The computer and all of the sensors are powered by the power distribution system that outputs the required

DC voltage for each system. The MKZ has a measured wheelbase of 2.85 m and a steering ratio of 14.8.

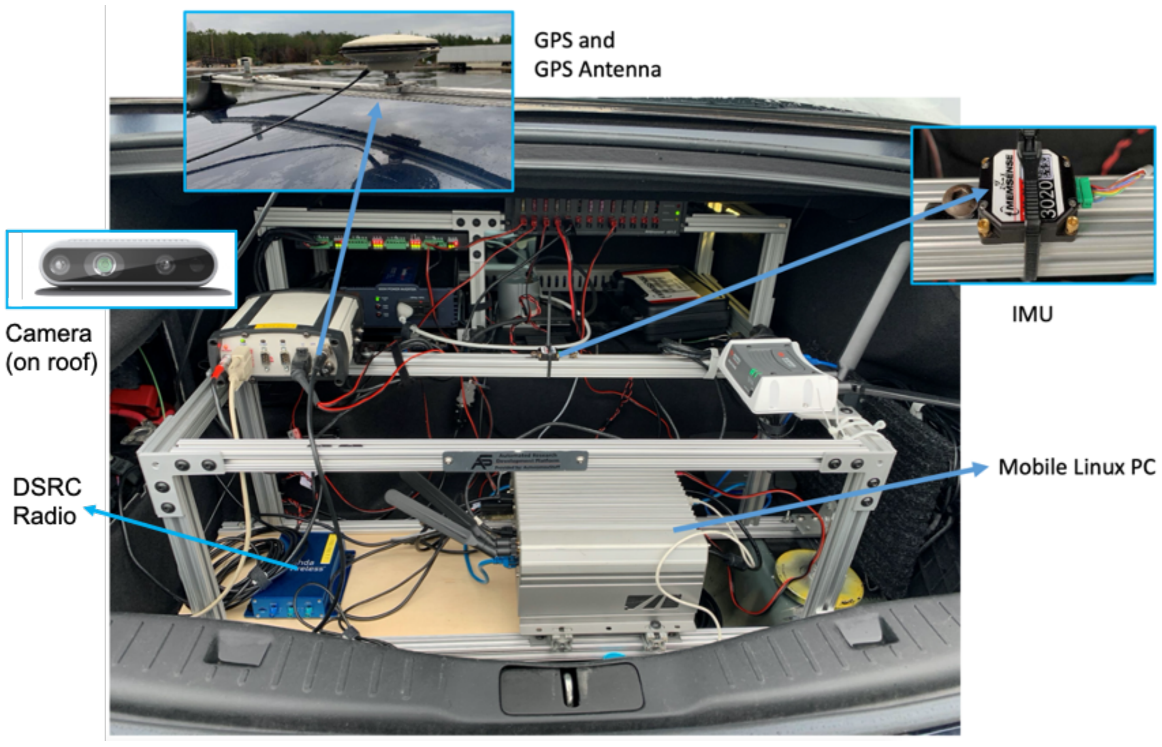


Figure 5.15: MKZ Hardware Setup [31].

5.2.2 Peterbilt 579

The second experimental platform used in this chapter was a Peterbilt 579. The 579 is a class-8 tractor trailer for long haul trucking and can be seen in Figure 5.16. The hardware suite on the truck includes a Memsense 3020 IMU, NovAtel GPS, Codha DSRC radio, Delphi RADAR, and a Linux computer. The computer calculates the controller outputs and sends them to the truck ECU through the CAN bus. All of the hardware is powered through an inverter connected to the truck main power. The 579 has a measured wheelbase of 6.1 m from the front axle to halfway between the rear axles and a steering ratio of 18.2.

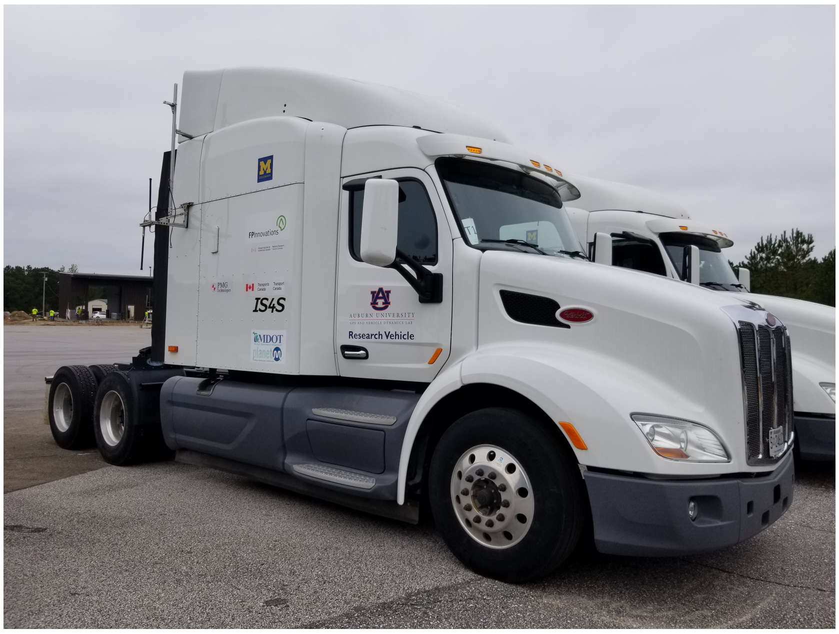


Figure 5.16: Peterbilt 579.

5.2.3 F1TENTH Car

An F1TENTH autonomous racing car was used as a third experimental platform for the vehicle agnostic path following controller [76]. The chassis, motor, and steering servo are from a $1/10^{th}$ scale Traxxas Slash 4x4 RC car as shown below in Figure 5.17. This car was outfitted with a sensor suite consisting of a Sparkfun Razor 9DOF IMU, Rplidar A2 single channel LiDAR, and an Intel RealSense D435 camera. The computation takes place on an on-board NVIDIA Jetson Xavier NX with everything powered by rechargeable NiMH batteries. The throttle and steering commands are sent to the motors through a VESC-6+ MK III motor controller. The F1TENTH car has a measured wheelbase of 0.32 m.

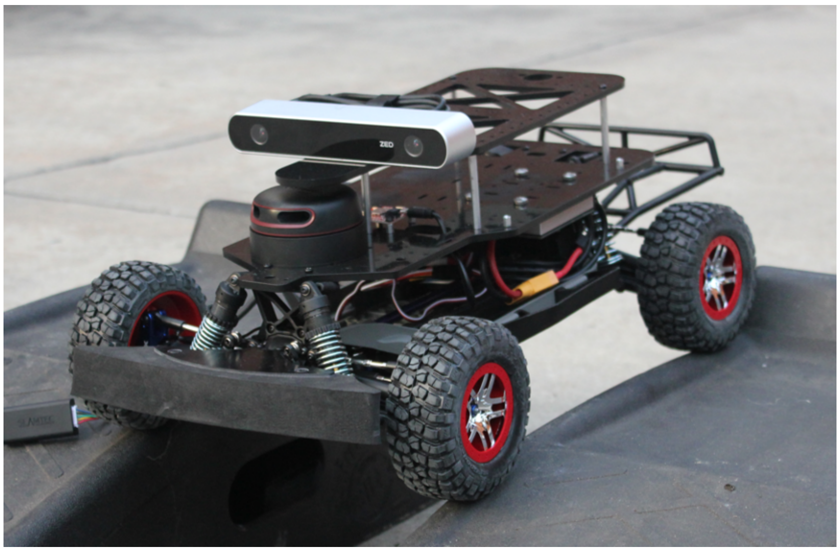


Figure 5.17: F1TENTH RC Car.

5.3 Reference Path Generation

Generating a desired reference path to follow is the first step for real-world testing. This chapter outlines two methods used for this: vision-based lane detection and RTK-GPS waypoint following. The vision-based system uses a camera to detect lane markings to create a center reference path. This was implemented on the vehicles in real-time using the Intel RealSense D435 camera and the Zed camera. The GPS waypoint path generation uses a NovAtel GPS receiver to record the desired reference path and a TDCP-INS algorithm to provide vehicle states. This could be used on or off road to follow a predetermined route or to follow another connected vehicle such as in vehicle platooning. This reference path is transformed into the frame of the controlled vehicle and sent to the path following node.

5.3.1 RTK-GPS Reference Path

GPS points were used to generate the path for the waypoint following system. To do this Real Time Kinematic (RTK) GPS points were taken along the NCAT skid pad in an ISO standard double lane change maneuver. RTK uses the data from a surveyed static GPS base station to remove errors from the data taken on the moving receiver. It does this by differencing the code and carrier measurements to give the position of the moving receiver relative to the base station, providing centimeter level measurement accuracy [77]. This highly accurate path was used as the input to the path following controller.

A Time Differenced Carrier Phase (TDCP) algorithm was fused with inertial measurements from an IMU to enable the vehicle to know its relative location to the path. The TDCP algorithm is similar to the RTK corrections described above, but it uses successive measurements from the same receiver to determine a differential measurement, which gives very accurate position changes because clock errors and atmospheric effects are canceled out. These measurements are combined with the IMU measurements in an EKF to provide a high frequency estimate of the vehicle position relative to the path [78]. The reference path was rotated to the initial orientation of each run, so that there were no variations in initial conditions and all measured errors resulted from the controller alone. An RViz visualization of the vehicle relative to the reference path during a test run can be seen in Figure 5.18

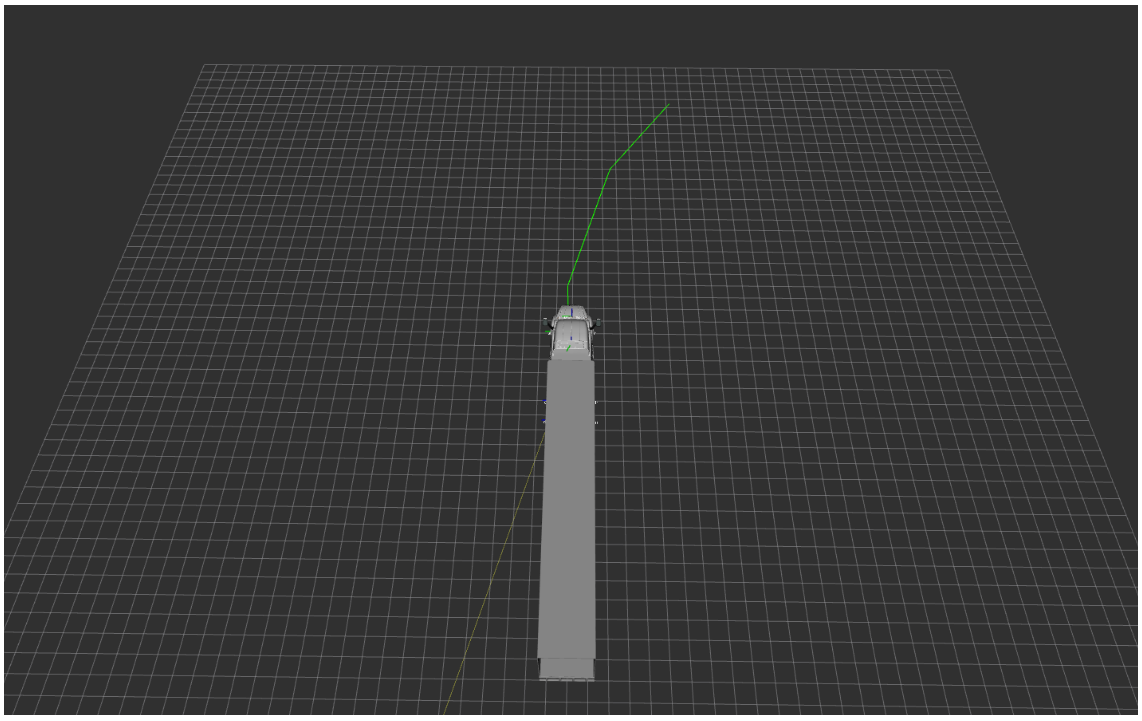


Figure 5.18: Visualization of Path Following on Skidpad.

5.3.2 Vision-Based Lane Keeping

A vision based system was designed to create a reference for a lane keeping system. The system provided the same outputs as lane detection node used in simulation. Two major versions of this system were implemented: a lightweight system using classical image processing methods and a deep neural network.

The classical image processing method reads in raw images from an Intel RealSense D435 camera (Figure 5.19). Next, a color threshold is applied to find white and yellow lane markings. A Sobel filter [79] is applied to find regions with high gradients [80]. These are combined into a binary mask, shown in Figure 5.20. The processed image then had an inverse perspective warp applied to generate a bird's eye view, as seen in Figure 5.21 [81].

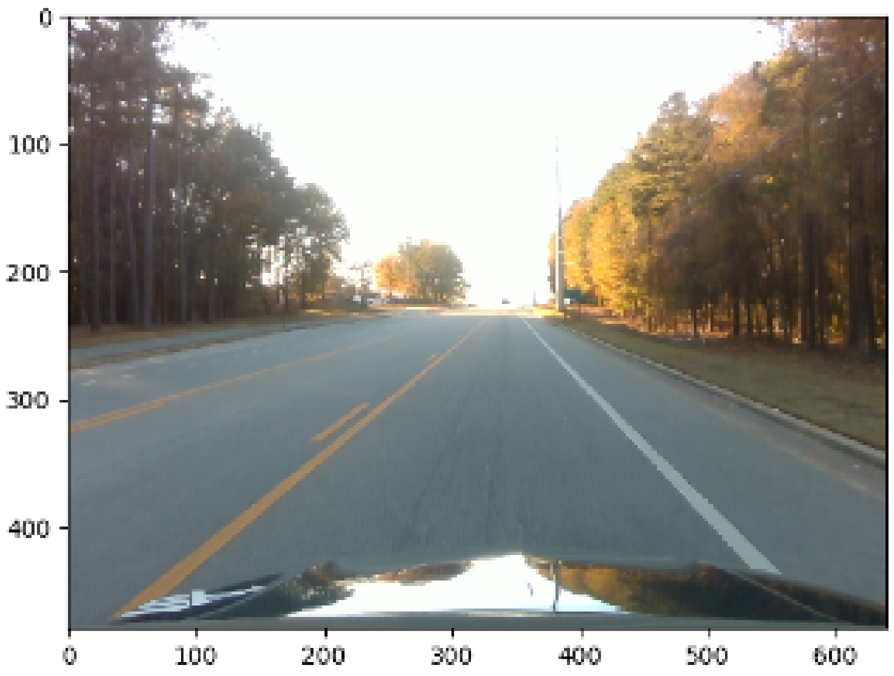


Figure 5.19: Unfiltered Raw Image.

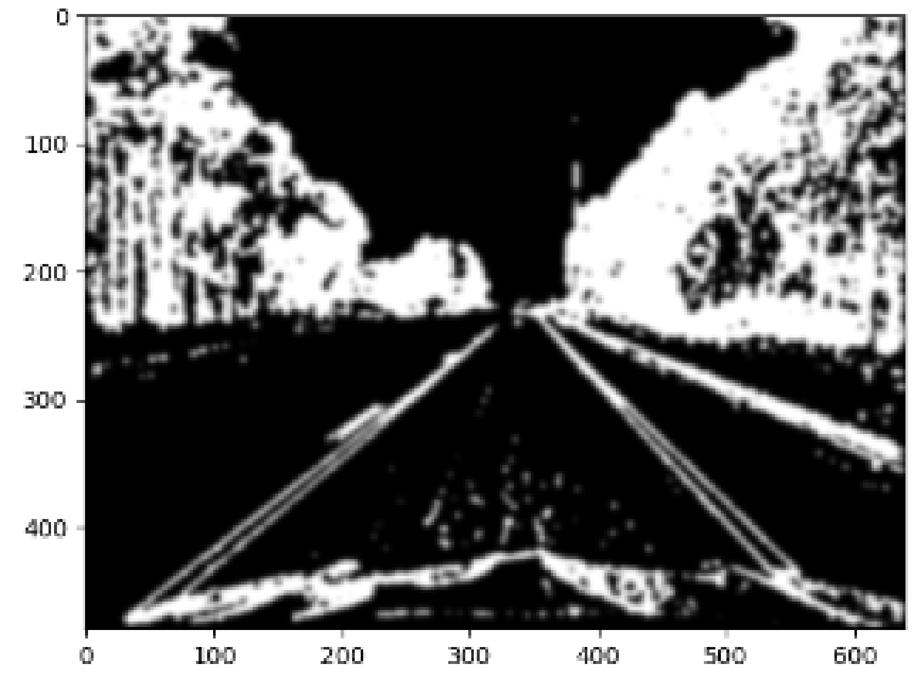


Figure 5.20: Color Threshold and Sobel Operator Applied to Image.

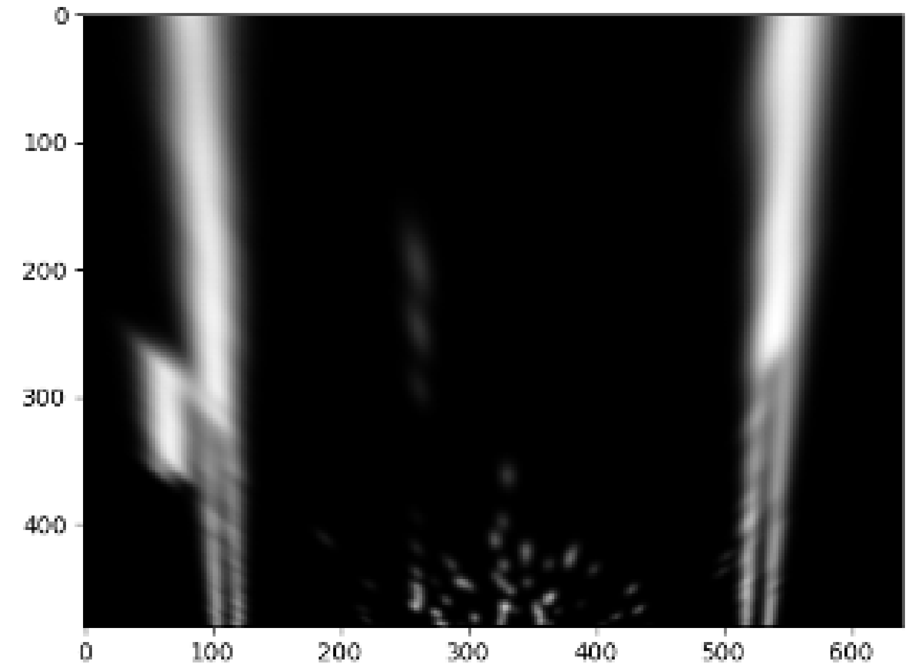


Figure 5.21: Inverse Perspective Warp.

Polynomials were fitted to the detected pixels, creating the lane line estimates. Both second and third order polynomial fits were tested, but there was no visible improvement in performance so a second-order polynomial was used in the final implementation. The lines were interpolated to form a center reference line which was used as the controller input. Figure 5.22 shows the polynomials transformed and overlaid on the input image, creating a visualization of the detected lane [82].

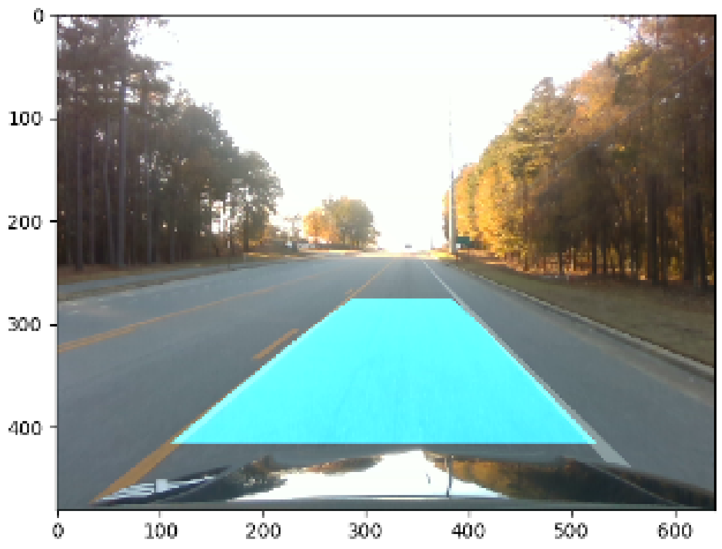


Figure 5.22: Classical Lane Detection Output.

The classical lane detection method was mostly accurate in detecting lanes but contained noise in the estimate as it was lighting dependent. However, this noise was filtered by the controller and had minimal effect on the lane keeping performance. The system lost the lane on tight corners, even when higher order polynomials were used. This was due to limited field of view (FOV) on the camera and the lost data when performing a perspective warp. For more details on the image processing refer to [83–85].

To improve the robustness of the lane detection system, a convolutional neural network (CNN) was designed to replace the classical lane detection system. This system was based on a simplified version of VGG-16 [86]. The image from the camera is input into the encoder, which takes an input and converts it to a tensor. The output tensor is fed into two decoders, which converts a tensor into a desired output. Both the encoder and decoder were CNNs. The first decoder performs binary segmentation on each pixel in the image. Each pixel that corresponds to a lane line is given a value of one. The second decoder performs instance segmentation, differentiating multiple lane lines. This architecture can be seen in Figure 5.23.

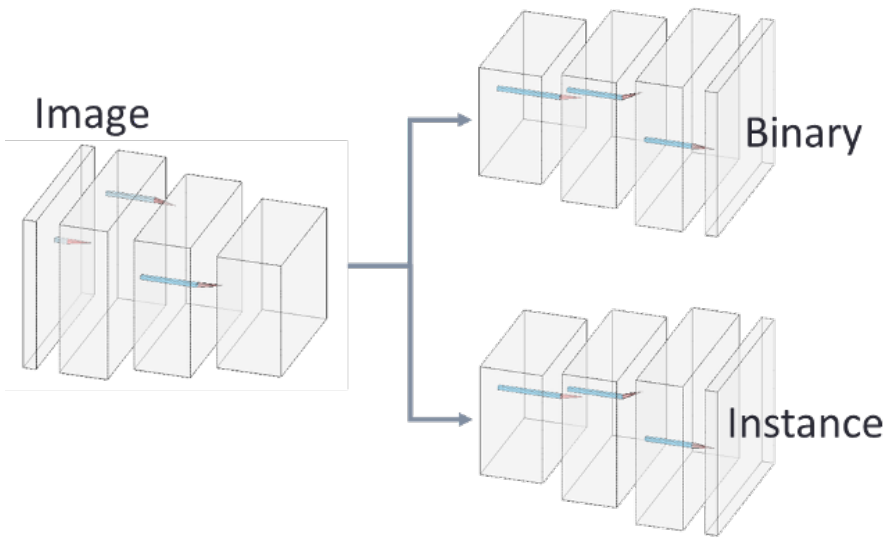


Figure 5.23: Lane Detection Network Architecture [87].

The network was trained on a combination of the TuSimple dataset [88] and some hand-labeled data recorded in the Auburn area. The TuSimple dataset contains 7000 one-second-long video clips, each with 20 frames. These were all recorded in daytime with good or medium weather conditions. The data was recorded along 2-lane, 3-lane, and 4-lane highways. Figure 5.24 provides examples of the TuSimple dataset. This provided the majority of the training data but lacked rainy conditions and yellow lane lines. To alleviate this, data was collected in the Auburn area that included yellow lane lines and rainy conditions. The images were hand-labeled to augment the TuSimple data.

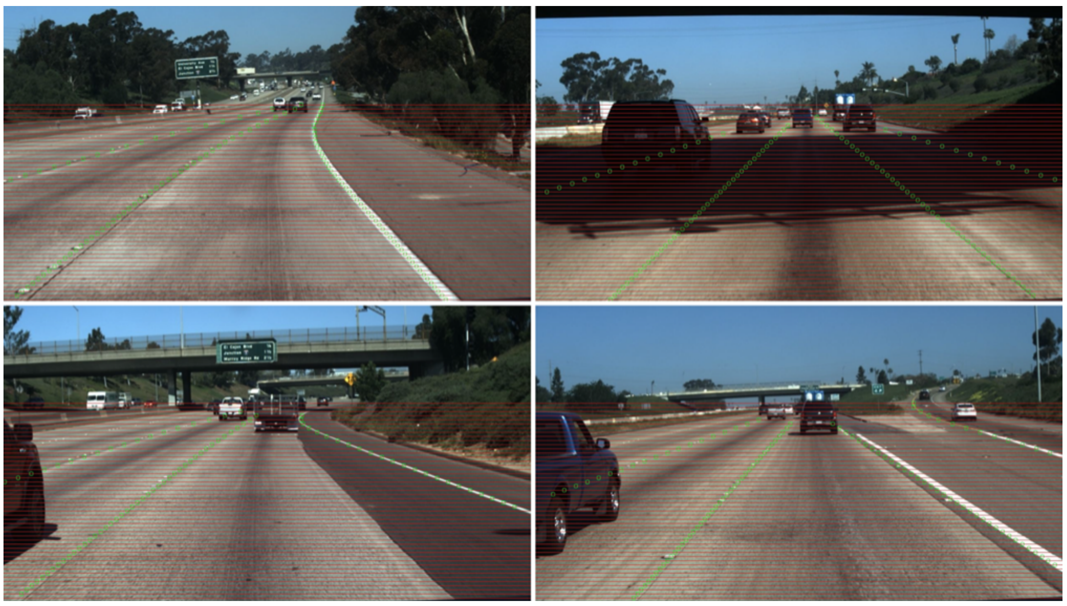


Figure 5.24: Images from the TuSimple dataset [88].

This system performed well and was able to reliably extract the lane lines. Figure 5.25 shows the output of the lane detection system overlaid on the input image.

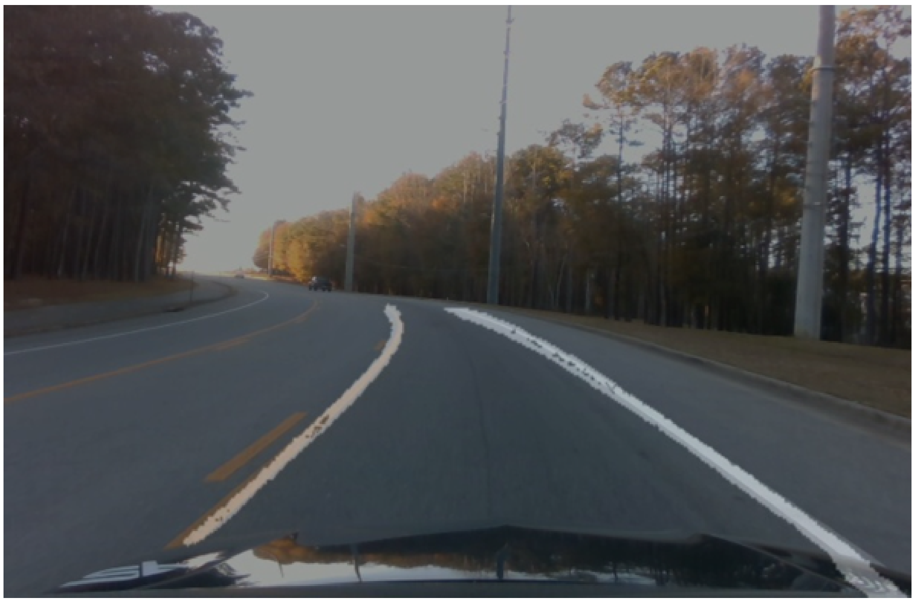


Figure 5.25: Lane Detection Output [87].

The neural network was put into a ROS node to communicate results to the rest of the vehicle. The output was sent to a path planning node which applied random sample consensus (RANSAC) [89] to reject outliers. Then, a second order polynomial was fit to each filtered lane line. An inverse perspective warp mapped samples of the polynomials to the navigation frame. These two lane lines are interpolated to find the center line path points and orientations, which was then used as the input to the controller. If one lane line is missing, the path planning node places the center line at an offset to the right or left of the detected lane line, depending on which lane marker was lost. Figure 5.26 shows the a visualization of the output from the lane detection and path planning node. The blue points represent the left lane, the red represents the right lane, and the green points are the interpolated center line.

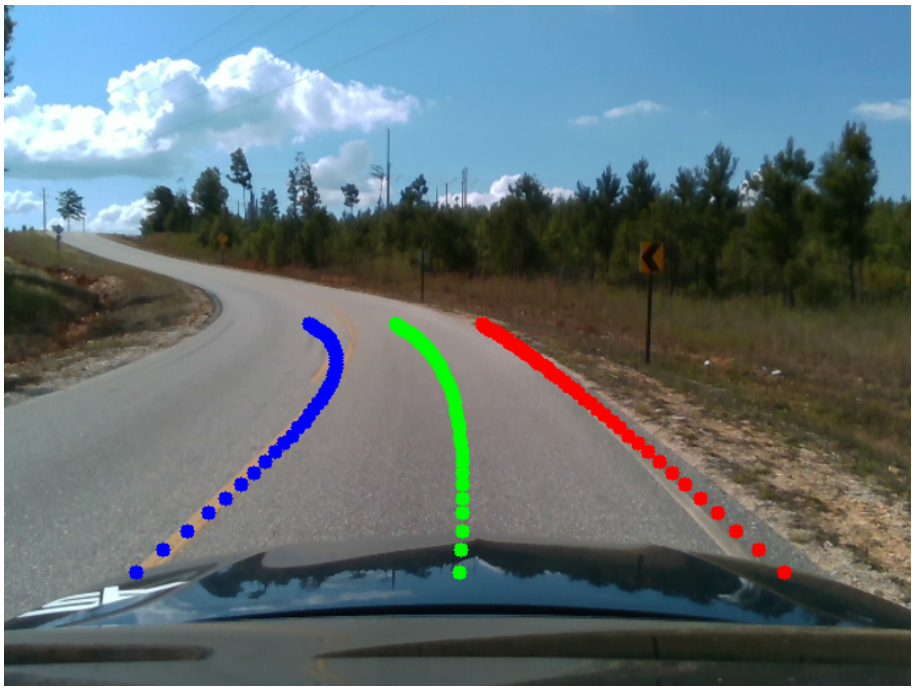


Figure 5.26: Lane Keeping Path Planning Output [87].

The CNN based lane detection system qualitatively performed well, but was extremely computationally expensive. In order to run in real time, a high quality GPU is needed. It also lost the lane lines in tight corners when the lanes leave the camera field of view (FOV). This issue was present in both lane detection systems but can be mitigated with a wider FOV camera or by fusing the inputs from multiple cameras. The CNN based lane detection system was also used to test the controller, but because the classical image processing system had sufficient accuracy and ran in real time even without GPU usage, it was used to produce the results in the following section.

5.4 Human Driving Analysis

In order to establish a performance baseline for the vehicle agnostic path following controller, human driving data was recorded. This data was collected for multiple drivers performing a double lane change as well as typical back-road driving. The next two sections will present the results of the human driving data.

5.4.1 Double Lane Change

Cones for a double lane change maneuver were set up along the RTK-GPS path described in Section 5.3.1. Four drivers manually drove the Peterbuilt 579 truck through the cones at approximately 5 m/s. Each driver performed a minimum of three runs. The lateral errors measured during these tests by the TDCP-INS system as well as the steering inputs from each driver can be seen in Figure 5.27.

All four drivers had fairly consistent errors and steering profiles between their own runs, but somewhat differed from driver to driver. The first driver had maximum lateral errors in the 1.0-1.5 m range for each run. The second driver had maximum lateral errors of about 1.0-2.0 m. The third driver was the most consistent, with lateral errors for each run falling between 1.1 m and 1.4 m. The fourth driver had errors ranging from 1.0-1.6 m.

While this data only represents one maneuver for one vehicle, some basic information can still be extracted from it. First, each driver had a max lateral error of approximately 1.0 m for their best run at 5 m/s. For the vehicle agnostic path following controller to successfully mimic human driving, it should have similar or less errors at the same speed. Second, each human driver had mostly smooth and consistent steering input. The vehicle agnostic path following controller should therefore also provide smooth and consistent steering.

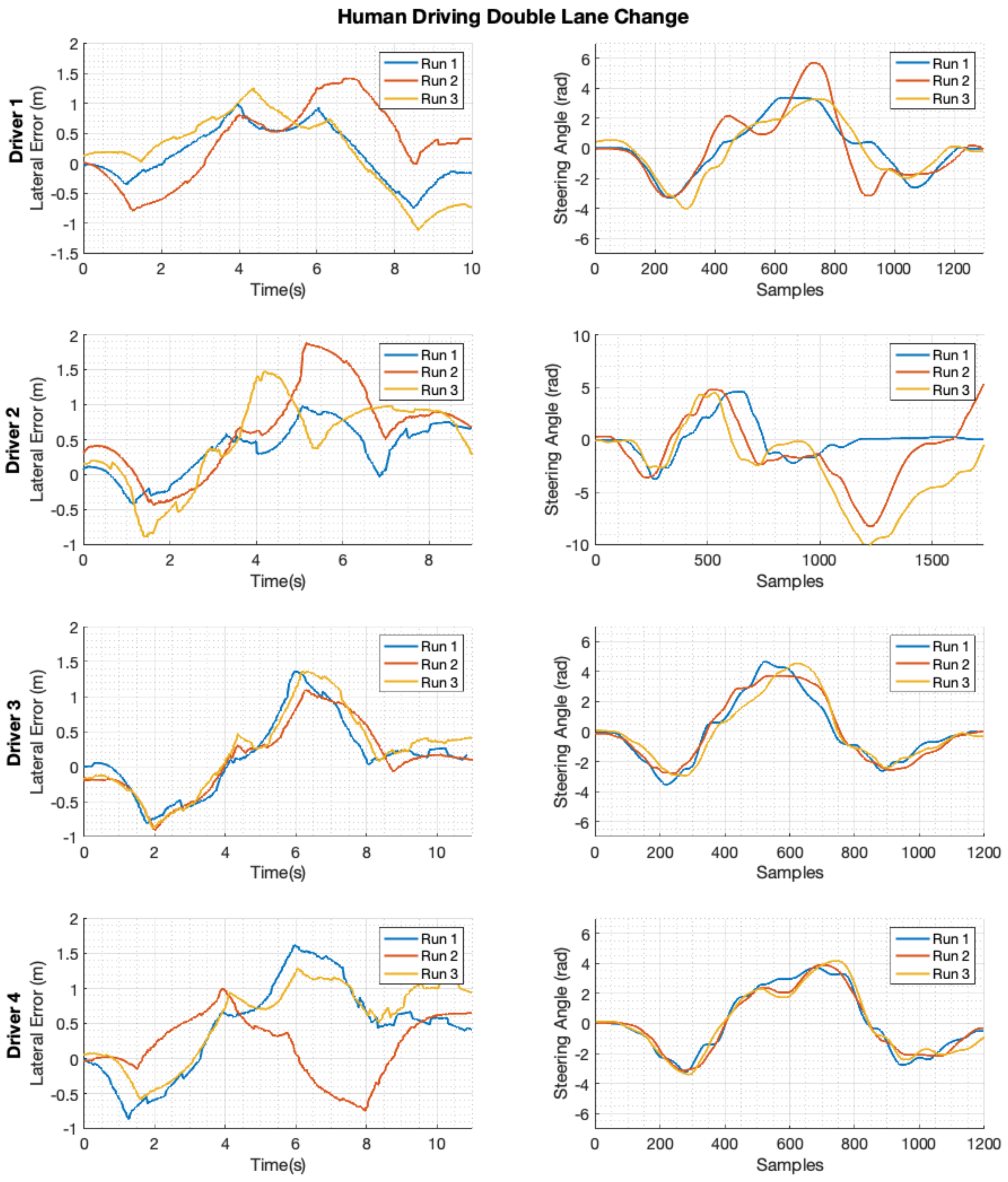


Figure 5.27: Human Driving Double Lane Change.

5.4.2 Lane Keeping

For the lane keeping system presented in this chapter, the reference path is determined by detecting lane lines and interpolating between them for the center line. This method only provides lookahead errors and no truth error at the vehicle CG. In order to compare the errors of the vehicle agnostic controller to those of a human driver, data was collected for multiple drivers while driving down the same stretch of curvy road. The drivers were instructed to drive normally at speeds well below the limit of handling and each driver drove in both directions down the section of road.

Figures 5.28 and 5.29 show that even with different drivers, the steering angle inputs between drivers were extremely consistent along the same route. It also shows the max lookahead errors were around 1.5 meters with a lookahead distance of 8 meters. The mean lookahead error was approximately 0.5 meters. These values were used as the baseline in which to compare the vehicle agnostic lane keeping system analyzed later in this chapter. While this is not enough data to fully characterize human driving, the goal was to achieve a ballpark number for typical lookahead error and steering response, and use it to compare controller errors along similar sections of curvy road when using the lane keeping system.

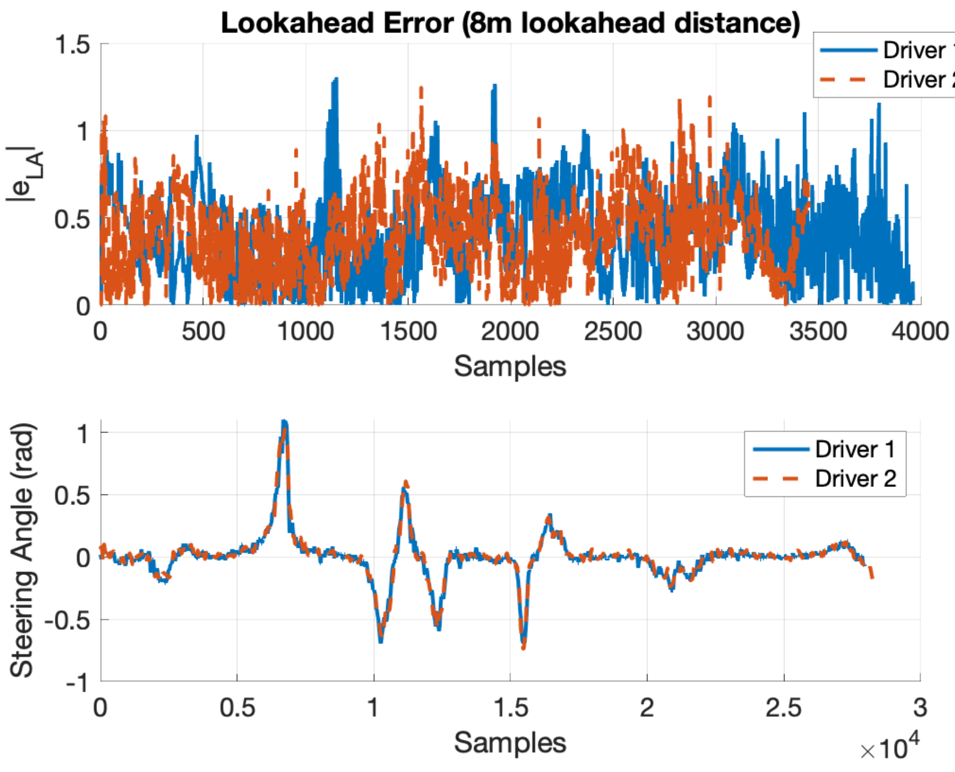


Figure 5.28: Human Driving Lookahead Error Northbound.

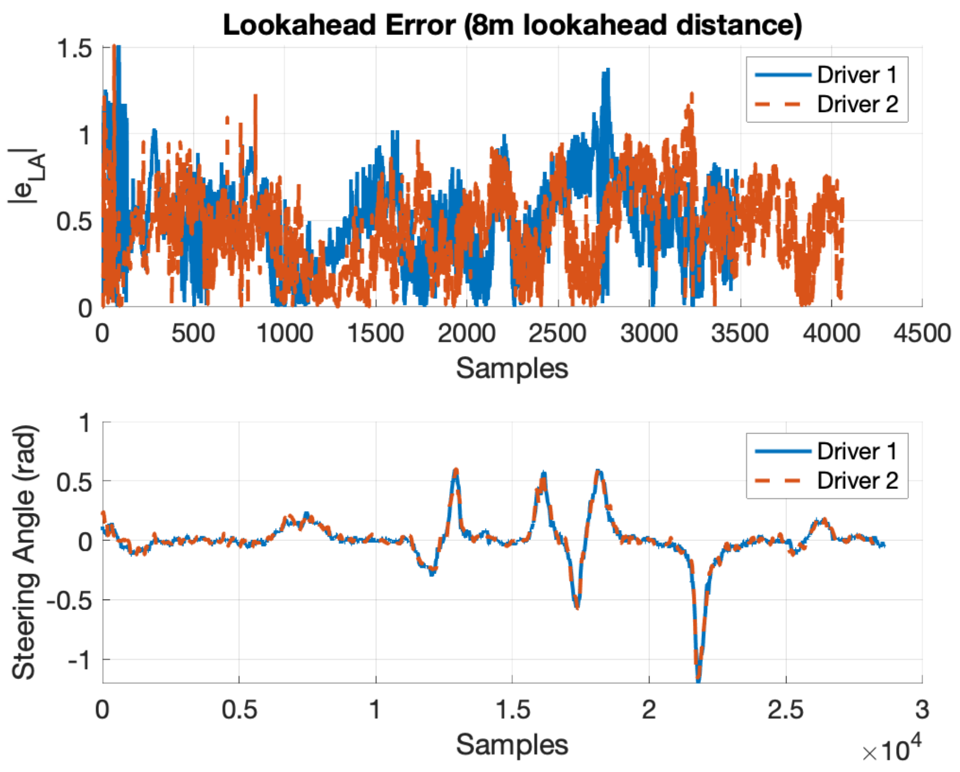


Figure 5.29: Human Driving Lookahead Southbound.

5.5 Experimental Results

This section presents the experimental results obtained by testing the vehicle agnostic path following controller in real-time on the Dataspeed Lincoln MKZ, Peterbilt 579 class-8 truck, and F1TENTH car. For each test, the experimental procedure will be described followed by the results and a brief description of the results.

5.5.1 RTK-GPS Reference Path Following

A double lane change was used as the reference path to test the path following abilities of the controller. This path was chosen because it is standardized, and it excites the dynamics of the vehicle. A full description of this maneuver was provided previously in Figure 3.1. The double lane change was run on the NCAT skid pad at multiple velocities ranging from 5 m/s to 10 m/s. Longitudinal velocities above this were not considered due to space constraints. The tests were performed on the Dataspeed MKZ, the Peterbilt 579 cab, and the Peterbilt 579 with a trailer. The MKZ on the NCAT skid pad as well as the reference path can be seen in Figure 5.30. The longitudinal velocities were controlled using a PID controller on the MKZ and manually controlled on the Peterbilt.

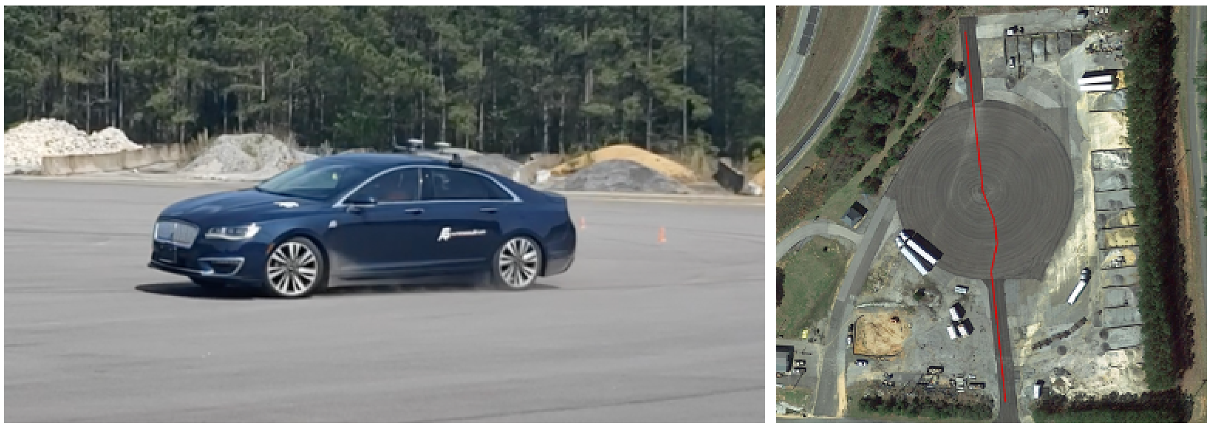


Figure 5.30: Double Lane Change Maneuver on NCAT Skidpad.

The lateral control system takes in a set of RTK-GPS waypoints corresponding to the double lane change. It uses the TDCP-INS estimation algorithm described earlier in this chapter to locate the vehicle relative in global coordinates. This position and orientation is used to transform the reference path into the vehicle frame to be sent to the controller. The tuning of the controller was the same as the tuning completed in simulation runs in Gazebo, as described in Section 5.1.3. Figure 5.31 displays the position output from the TDCP-INS node during one of the double lane change runs.

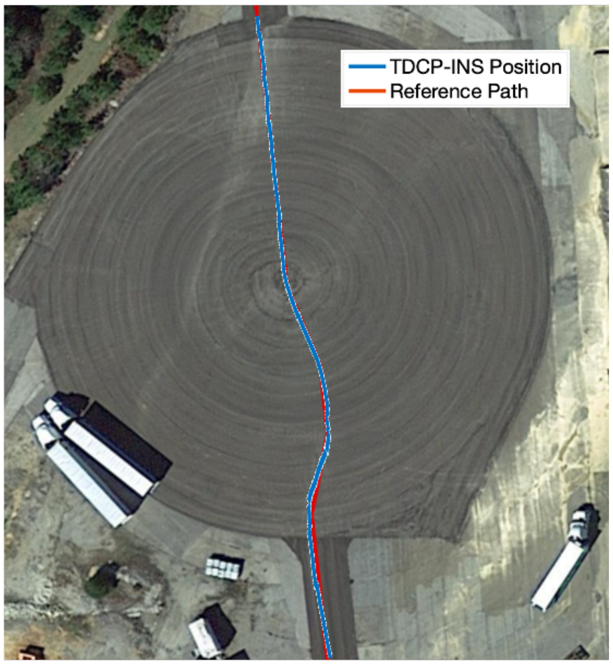


Figure 5.31: Vehicle Path During Double Lane Change.

A minimum of three runs were completed at each velocity. The plots below have three runs at the stated velocities, overlaid on each other. The top subplot displays the lateral error in meters between the TDCP-INS estimator position and the RTK-GPS reference path. The bottom subplot shows the steering command calculated by the controller.

The first set of runs were completed using the Lincoln MKZ. The effective wheelbase was initialized as the measured wheelbase (2.85 m). Figure 5.32 shows that at 5 m/s, the lateral error is less than half a meter throughout the maneuver. The steering input from the controller was also extremely smooth. All three runs were very consistent as well, with any differences resulting from variations in initial position when the test was initiated. The effective wheelbase stayed about the same, ending at 2.82 m. At 10 m/s the max lateral error remained around half a meter for all three runs as shown in Figure 5.33. Again, the steering was exceptionally smooth and consistent between runs. The effective wheelbase barely changed, ending up at 2.87 m.

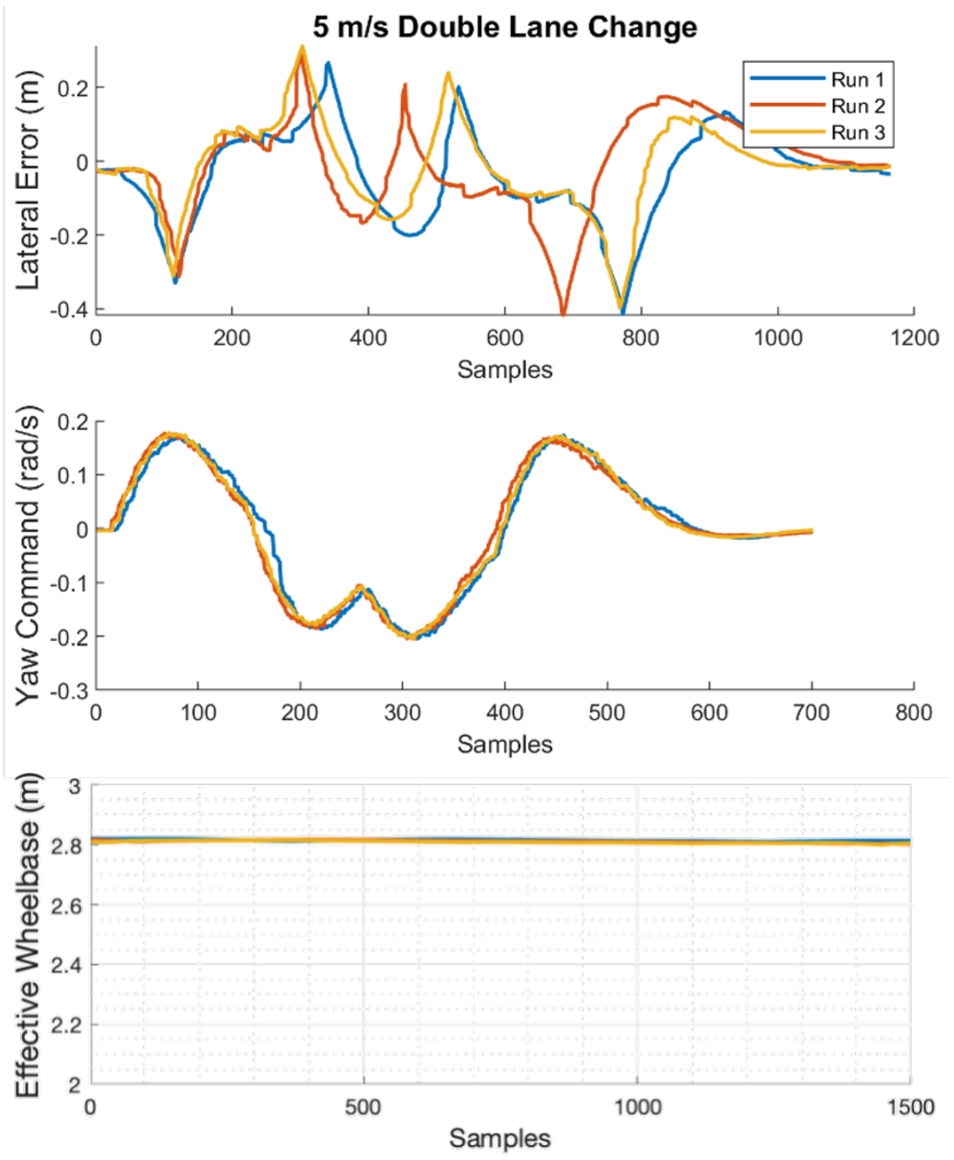


Figure 5.32: MKZ Controller Performance for 5 m/s Double Lane Change.

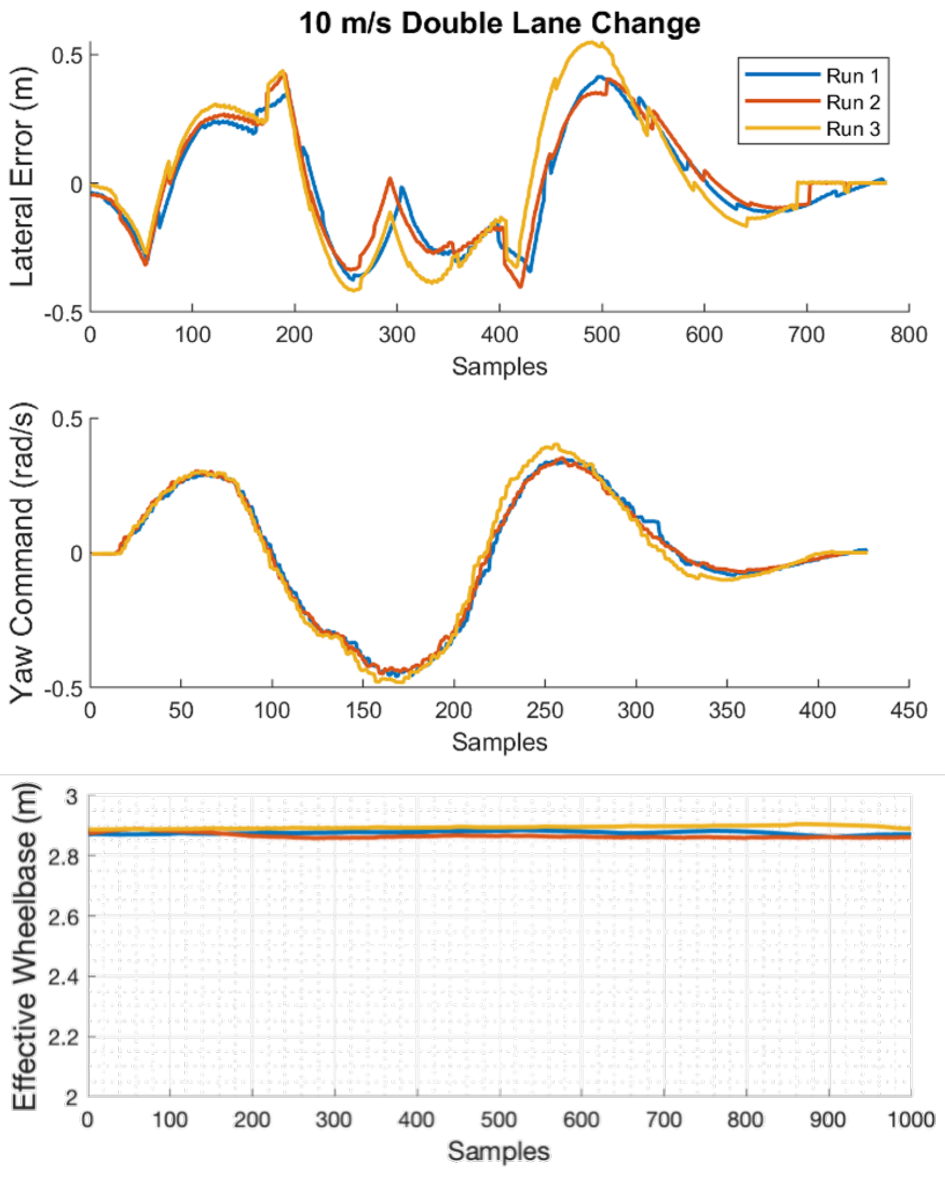


Figure 5.33: MKZ Controller Performance for 10 m/s Double Lane Change.

The next set of runs were completed using the Peterbilt 579. The first two sets of tests were completed with the truck cab only, and the third was with a trailer attached as shown in Figure 5.34. Since the truck has two rear axles, the effective wheelbase was initialized as the average of the distance from the front axle to each rear axle (6.1 m). Note: the Peterbuilt had opposite sign conventions on the steering motor than the MKZ.

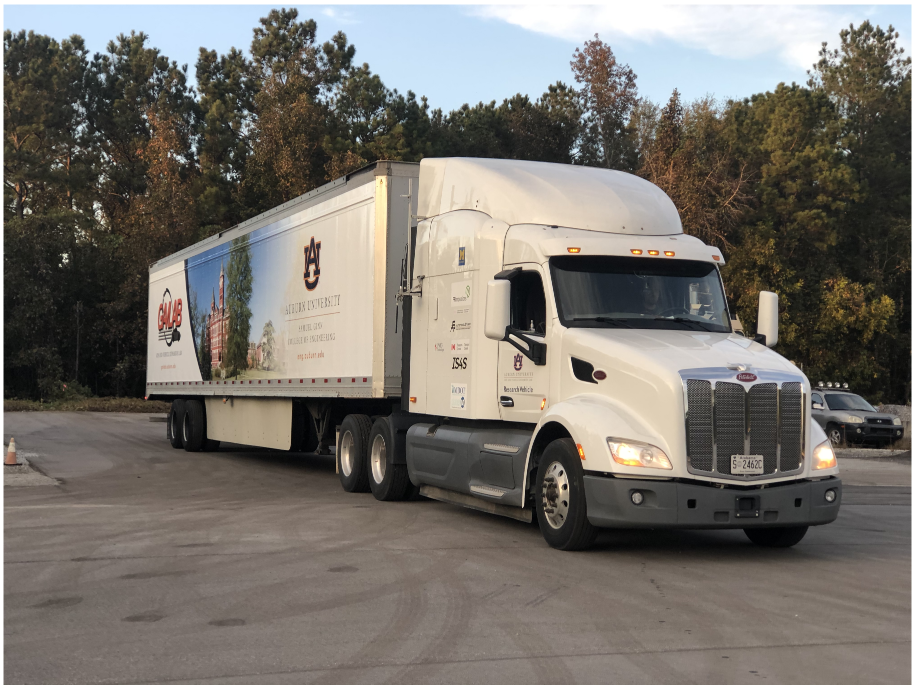


Figure 5.34: Peterbilt 579 with Trailer on Skidpad.

When the truck was tested at 5 m/s, the max lateral error was approximately 0.5 meters, which is comparable to the MKZ double lane change at this speed. The steering was still smooth and consistent as seen in Figure 5.35. For the initial run, the effective wheelbase increased throughout the maneuver significantly. Because the double lane change did not provide enough persistent excitation for the effective wheelbase to converge in a single run, the maneuver was run multiple times until the effective wheelbase reached a steady state value at this speed. The blue lines show the first run, before the effective wheelbase had converged. The red and yellow lines show two runs after the effective wheelbase converged. The final effective wheelbase value was about 8.78 m. However, even before convergence, the controller had similar performance to the final runs and successfully handled the maneuver.

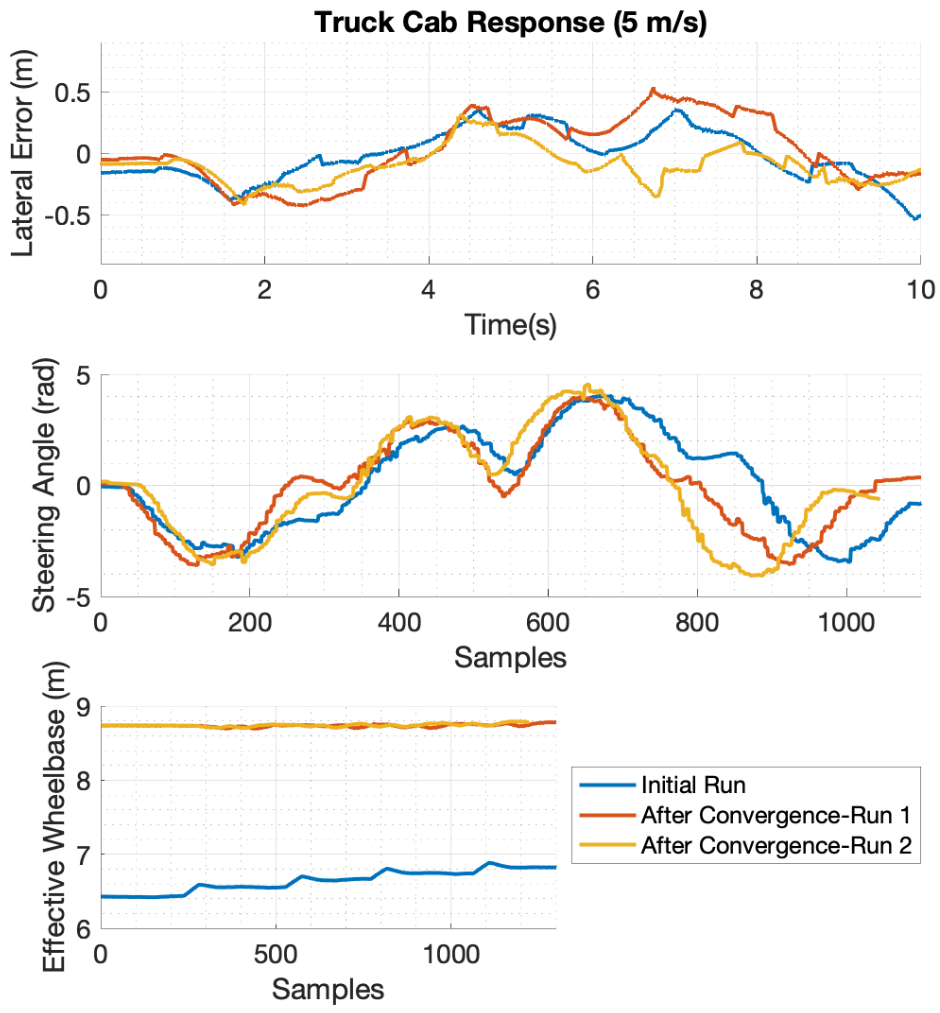


Figure 5.35: Peterbilt Controller Performance for 5 m/s Double Lane Change.

Next, the same test was run with the Peterbilt at 8 m/s. This speed was as high as the truck could accelerate to in the given space. The same procedure as previously for initializing the effective wheelbase and letting it converge was performed. With the increased longitudinal velocity, the initial run had a max lateral error of about 1.5 meters, as shown in Figure 5.36. After the effective wheelbase converged to 8.75 m, the max lateral error dropped to about 0.5 meters. This was comparable to the MKZ at this velocity and impressive given the aggressiveness of the maneuver. At this velocity, the truck's dynamics were stressed and it was nearing the limits of the tires. The steering remained smooth for these runs.

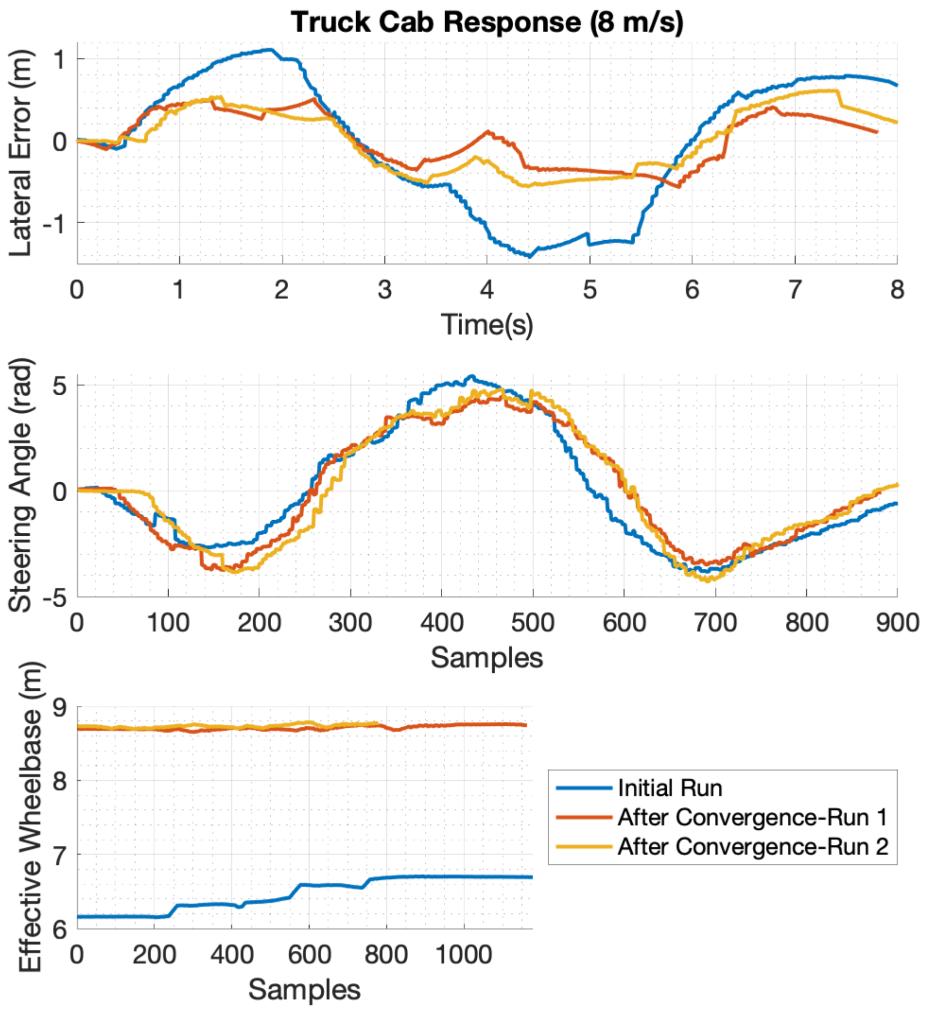


Figure 5.36: Peterbilt Controller Performance for 8 m/s Double Lane Change.

The last plot in this section shows the truck response with a trailer attached at 5 m/s. The process of initializing the effective wheelbase at 6.1 m and completing runs until it converged was repeated. With the trailer, the max lateral error was just over 0.5 meters both before and after effective wheelbase convergence, as shown in Figure 5.37. This was equivalent to the error of the cab with no trailer and the MKZ at this speed. The effective wheelbase converged to 8.82 m, and the steering was smooth and consistent.

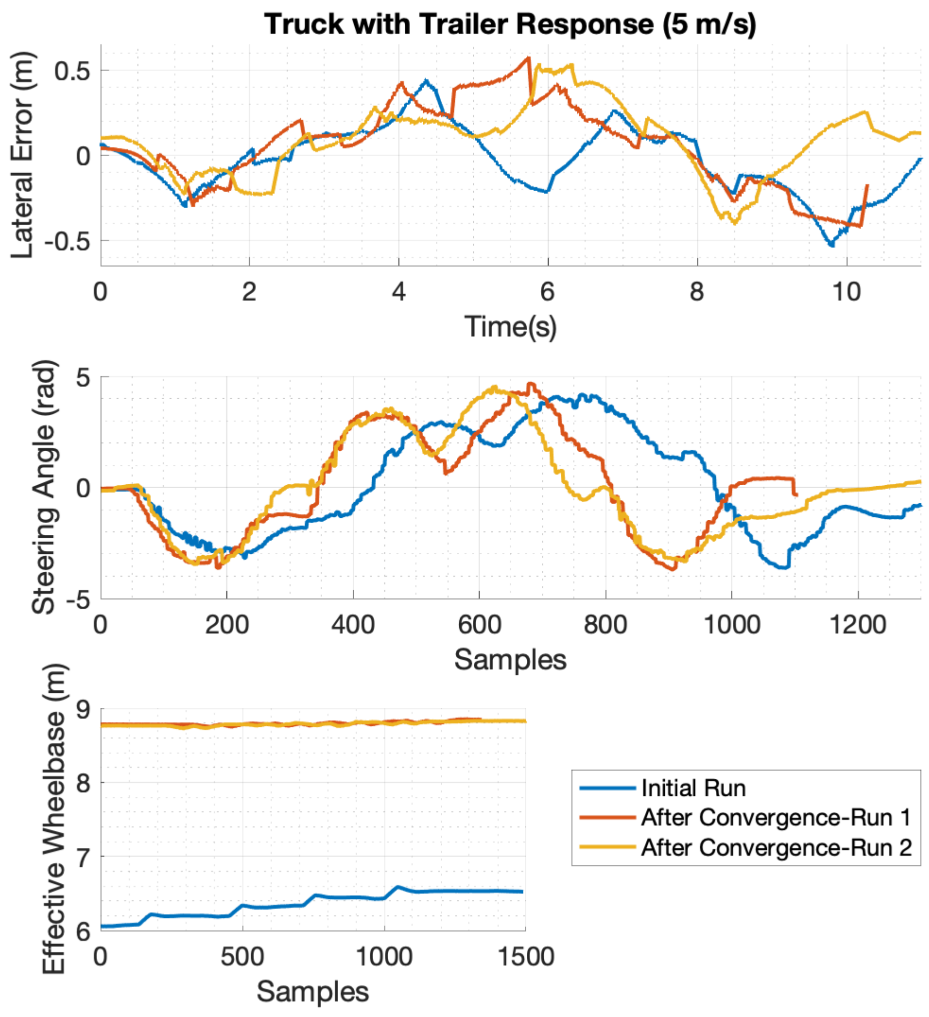


Figure 5.37: Peterbilt with Trailer Controller Performance for 5 m/s Double Lane Change.

Overall, the vehicle agnostic path following controller was successful at maintaining low lateral errors at a variety of longitudinal velocities, while also mimicking the path following behavior of a human driver. During the double lane change maneuvers, human drivers had max lateral errors of over a meter. The controller produced errors around half a meter for each vehicle platform during the same maneuver, even at higher speeds. The controller was able to handle a passenger sedan, a Class-8 truck cab, and the Class-8 truck with trailer. The flexibility to work across platforms combined with the smoothness of steering actuation validates the vehicle agnostic path following controller.

5.5.2 Vision-Based Lane Keeping

Finally, the vehicle agnostic path following controller was tested as part of a lane keeping system, using the vision-based lane detection to generate the reference path. The tests were performed using the F1TENTH platform. The lanes were created indoors using tape on the floor, with a variety of curves to produce dynamic maneuvers. The lane keeping system was tested on a $1/10^{th}$ scale car to verify the flexibility of the controller on a wide variety of platforms.

Figure 5.38 shows the ROS software architecture of the lane keeping system. The camera nodes publish the raw color image which is used by the lane detection node, which detects the lanes and plans a path in the center for the vehicle to follow. The method used for path planning is the same as the classical lane detection described previously in detail in Section 5.3.2. The controller takes in the target path and sends steering, throttle, and brake commands to the drive-by-wire node. On a road vehicle, this node converts these into CAN messages and sends them to the vehicle. In the case of the F1TENTH car, the drive-by-wire node represents the motor controller, which converts the commands to voltages for the drive motor and steering servo.

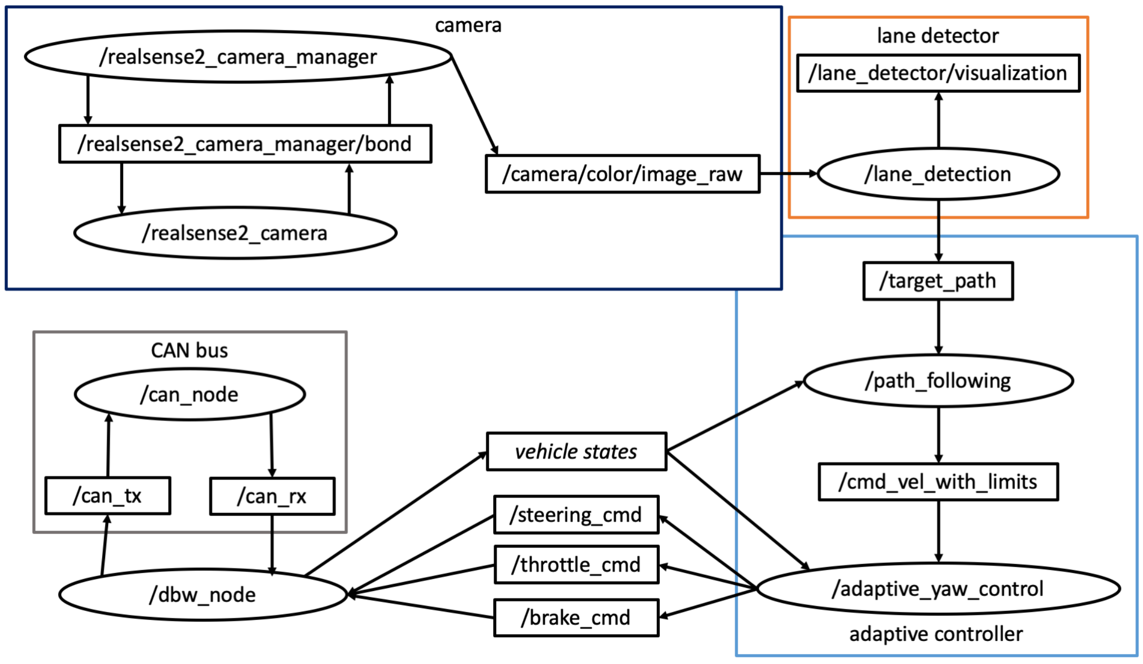


Figure 5.38: Lane Keeping System ROS Software Architecture.

The lane keeping system was tested at speeds ranging from 0.5 m/s to 1.5 m/s. The following results show the lookahead error at the closest path point in the camera's field of view. The lookahead distance at this point is approximately 30 centimeters, so the true lateral error at the CG is even lower. The vehicle can be seen during the lane keeping test in Figure 5.39.

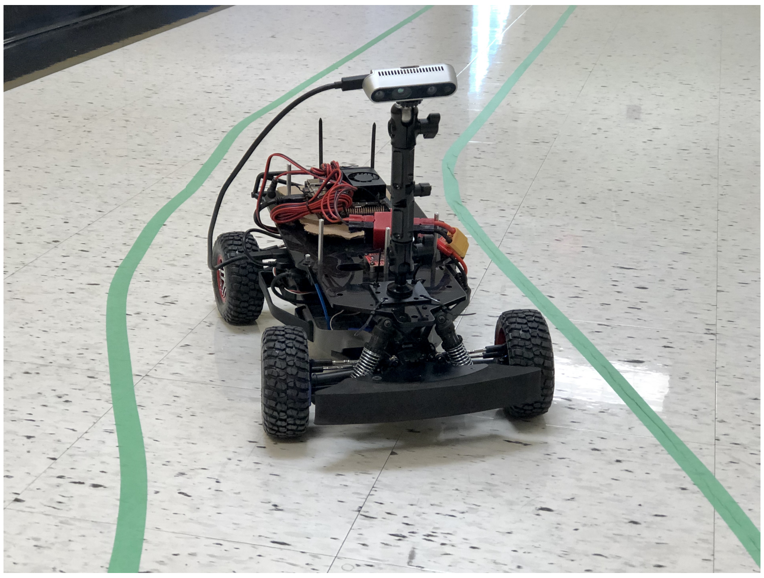


Figure 5.39: F1TENTH Car During Lane Keeping.

Figure 5.40 shows the performance at 0.5 m/s. At this speed the lookahead error remained below 10 cm and the vehicle remained close to the reference center line at all times. The steering input from the controller was consistent between runs.

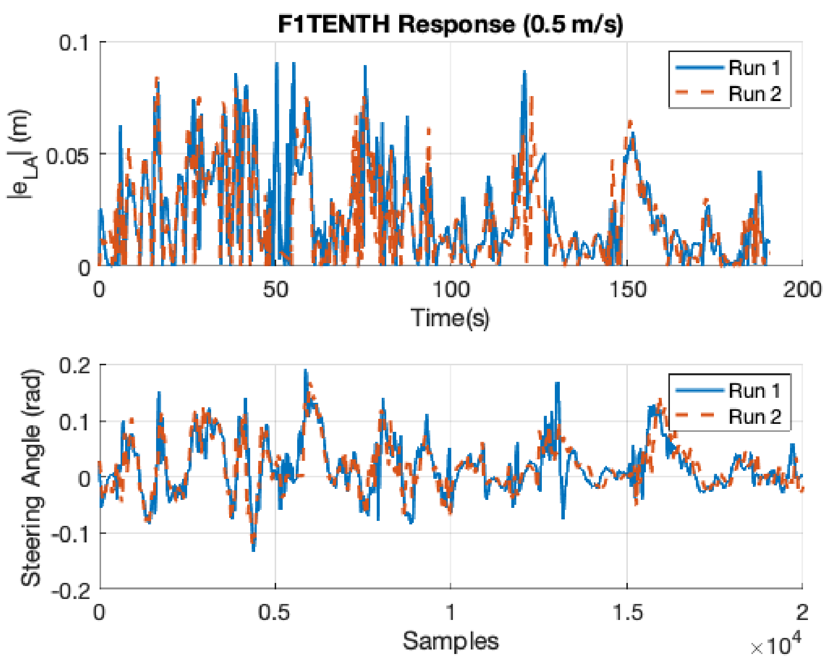


Figure 5.40: F1TENTH LKS Performance at 0.5 m/s.

At 1.0 m/s the max lateral error was approximately 14 cm as shown in Figure 5.41. For the most part, the vehicle stayed near the reference path with the spikes occurring when the lane detection lost the path for short periods of the experiment.

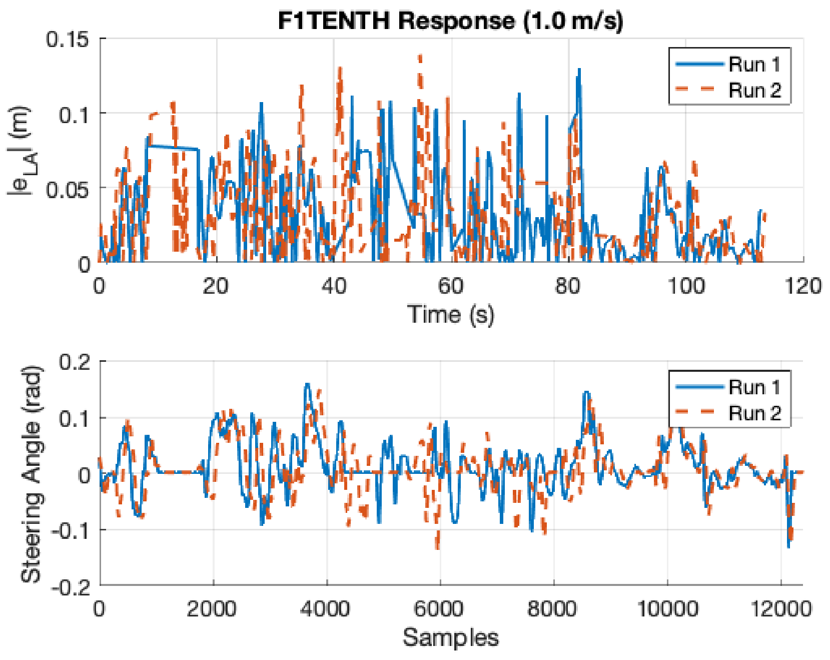


Figure 5.41: F1TENTH LKS Performance at 1.0 m/s.

When tested at 1.5 m/s, the max lateral error still remained below 15 cm, but the runs were less consistent. At this speed, the lane detection had more instances of bad reference generation which lead to spikes in error. This could be alleviated by further tuning the lane detection node for indoor use. Nevertheless, when the path generation was good, the vehicle was able to stay on the path as seen in Figure 5.42.

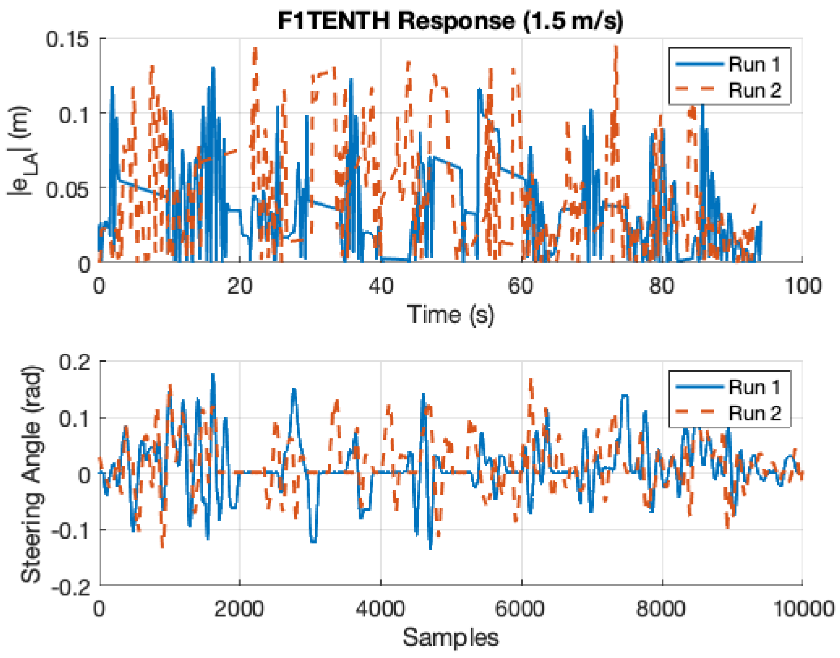


Figure 5.42: F1TENTH LKS Performance at 1.5 m/s.

Without fully modelling the vehicle dynamic parameters, the scaled equivalent speed cannot be determined [90]. However, the lateral path tracking errors of the $1/10^{th}$ scale car can be compared to the human drivers in a full size car by converting the errors to a dimensionless value by diving the lookahead error by the wheelbase length [91, 92].

$$e^*_{max} = \frac{e_{max}}{L} \tag{5.2}$$

$$e^*_{max} = \frac{1.4m}{2.85m} = 0.4912\,[unitless] \tag{5.3}$$

The same is done with the F1TENTH car for each speed.

$$\text{0.5 m/s:} \qquad e^*_{max} = \frac{10cm}{32cm} = 0.3125\,[unitless] \tag{5.4}$$
$$\text{1.0 m/s:} \qquad e^*_{max} = \frac{14cm}{32cm} = 0.4375\,[unitless] \tag{5.5}$$
$$\text{1.5 m/s:} \qquad e^*_{max} = \frac{15cm}{32cm} = 0.4687\,[unitless] \tag{5.6}$$

When put into dimensionless form, the lane keeping system on the F1TENTH car was able to provide equivalent or better performance than human drivers. Additionally, the steering was smooth and mostly consistent with the variations coming from the lane detection node. This was also sufficient enough to keep the vehicle inside the lane lines at all times.

Overall, the vehicle agnostic path following controller was successful on both the lane keeping system and during the double lane change tests. In both scenarios, the controller maintained path tracking errors less than those seen in human driving. The vehicle agnostic path following controller also worked on a wide range of simulated and experimental vehicle platforms without changing the tuning while providing smooth steering inputs.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis presented a vehicle agnostic path following controller that was able to mimic human driving characteristics. The models for vehicle and tire dynamics used in the thesis were described in Chapter 2. Chapter 3 provided a sensitivity analysis of many common path following controller architectures to model uncertainty to determine the strengths and weaknesses of each one. The results of the sensitivity analysis were used to inform the design of the new vehicle agnostic path following controller presented in Chapter 4, the primary contribution of this work. Chapter 5 validated the controller in simulation and real-time on a wide range of vehicles.

The vehicle agnostic path following controller was shown to be able to control a sedan, minivan, SUV, and pickup truck in a lane keeping scenario in simulation. This test was run with each vehicle at a wide variety of speeds and with no controller tuning between scenarios. The controller was intentionally initialized poorly in order to stress the system, but the controller was robust to these initialization errors. In addition to being capable of keeping all the vehicles inside their lanes, the controller was able to effectively estimate each vehicles effective wheelbase. The effective wheelbase was a value corresponding to the DC gain of the yaw dynamics, comprised of the wheelbase length and the steering ratio. The effective wheelbase was the primary parameter used to adapt the inner loop control law.

To test the controller on experimental platforms, two different reference path generation methods were used. First, a RTK-GPS reference path generation model was presented, which could be used in a variety of applications such as truck platooning. In this method, a TDCP-INS estimator is used to determine the vehicle state and rotate the global reference path into the vehicle coordinate frame. Second, a vision-based lane keeping system was introduced. This system uses either classical image processing or a deep neural network to detect the lane lines in an image. These detected lines then have a polynomial fit to each one, and a path in the center of the lane is then calculated by interpolating the two lane polynomials. This center path is used as the reference input to the controller for a lane keeping system.

The vehicle agnostic path following controller was then implemented on three experimental platforms, a small $1/10^{th}$ scale RC car, a typical passenger sedan, and a Class-8 tractor trailer. This was done to test the flexibility of the controller to vehicles with drastically different dynamics. The controller was tested on the sedan and semi truck during a double lane change maneuver. The controller performed well at all speeds with both vehicles. Additionally, tests were run with the Class-8 truck while pulling a loaded trailer. The controller was robust to each of these perturbations. Lastly, the controller was tested on the $1/10^{th}$ scale car using the lane keeping system. Even with this substantially smaller system, the controller performed well, always keeping the car inside the lines at a range of velocities.

Based on the simulation and experimental results, the vehicle agnostic path following controller was successful at achieving the objectives listed below.

- The system closely matched the performance and smoothness of human driving
- The system was vehicle independent
- The system required no tuning between platforms
- The system was robust to disturbances

6.2 Future Work

Although the control system was able to meet the design requirements, there is plenty of room for improvement and related future research. First, the path generation can be improved by using clothoids or other methods that create a continuous reference path. This would provide a few benefits over the current paths which consist of discrete points. A continuous path would allow the adaptive lookahead distance to be used more precisely when finding the lookahead error instead of searching for the nearest point. Having guaranteed continuity in the curvature of the path would increase the smoothness of the feedforward portion of the controller. These would provide for even smoother steering actuation and likely lower lateral errors.

Much more time could be spent finding optimal controller parameters. The method used to adapt the lookahead error based on speed could be further investigated and fine tuned. Also, the threshold and manner of varying lookahead distance based on road profile would benefit from further investigation. The proportional gain, the adaptation rate, and the time constant of the nominal model could be further optimized with more testing to increase path tracking performance and robustness.

Some of the most interesting future work is in testing different types of controllers and models in the inner loop of the controller. Replacing the simple kinematic based controller with a model predictive controller has the potential to decrease the lateral path errors while increasing robustness. Model predictive control showed greater disturbance rejection in the sensitivity analysis than the kinematic controller. Implementing a simple MPC could be done fairly easily while still using the effective wheelbase as the adaptive parameter. Higher order nominal models or other controller types for the inner adaptive piece could also possibly increase the performance, especially at the limit of handling.

This thesis only discussed the design and implementation of the lateral control portion of path following, while a simple PID controller was used for the longitudinal control. However, an implementation of the cascaded adaptive controller for longitudinal control could provide similar performance and stability benefits as the lateral controller. This combined with the lateral steering controller would be an excellent candidate for applications with changing dynamics, such as a vehicle being loaded and unloaded.

Lastly, combining the vehicle agnostic path following controller with a perception and path planning software stack would provide a modular and vehicle independent Level 3 or higher autonomous system. This would add obstacle avoidance, allowing the system to be used for hands-off highway driving or other applications.

References

[1] USDAT Bureau of Transporation Statistics. *Number of U.S. aircraft, vehicles, vessels, and other conveyances.* URL: https://www.bts.gov/content/number-us-aircraft-vehicles-vessels-and-other-conveyances. (accessed: 09.18.2020).

[2] United States Census Bureau. *2010 Census briefs: Age and sex composition.* URL: https://www.census.gov/prod/cen2010/briefs/c2010br-03.pdf. (accessed: 09.18.2020).

[3] Insurance Institute for Highway Safety (IIHS). *Fatality facts 2018.* URL: https://www.iihs.org/topics/fatality-statistics/detail/state-by-state%5C#yearly-snapshot. (accessed: 09.16.2020).

[4] National Highway Traffic Safety Administration (NHTSA). *FARS encyclopedia.* URL: https://www-fars.nhtsa.dot.gov/Main/index.aspx. (accessed: 09.16.2020).

[5] Bryant Walker Smith. *Human error as a cause of vehicle crashes.* URL: http://cyberlaw.stanford.edu/blog/2013/12/human-error-cause-vehicle-crashes. (accessed: 09.16.2020).

[6] C.J. Kahane. "Lives saved by vehicle safety technologies and associated federal motor vehicle safety standards". In: *DOT HS Report* No. 812 069 (2015).

[7] National Highway Traffic Safety Administration (NHTSA). "Statistical analysis of the effectiveness of electronic stability control (ESC) systems – Final report". In: *DOT HS Report* 810 794 (2007).

[8] National Safety Council- Injury Facts. *Car crash deaths and rates*. URL: https://injuryfacts.nsc.org/motor-vehicle/historical-fatality-trends/deaths-and-rates/. (accessed: 09.18.2020).

[9] Alex Davies. *GM has aggressive plans for self-driving cars*. URL: http://www.wired.com/2015/10/gm-has-aggressive-plans-for-self-driving-cars/. (accessed: 09.16.2020).

[10] Curtis Franklin Jr. *Ford's autonomous car: Under the hood*. URL: http://www.informationweek.com/iot/fords-autonomous-car-under-the-hood/a/d-id/1323920. (accessed: 09.16.2020).

[11] Heather Kelly. *Toyota to invest $1 billion into artificial intelligence*. URL: http://money.cnn.com/2015/11/05/technology/toyota-ai-research/. (accessed: 09.16.2020).

[12] GM Corporate Newsroom. *Honda joins with Cruise and General Motors to build new autonomous vehicle*. URL: https://media.gm.com/media/us/en/gm/home.detail.html/content/Pages/news/us/en/2018/oct/1003-gm.html. (accessed: 09.16.2020).

[13] Philip Iglauer. *Hyundai to deveop fully autonomous cars by 2030*. URL: http://www.zdnet.com/article/hyundai-to-develop-fully-autonomous-cars-by-2030/. (accessed: 09.16.2020).

[14] Mike Ramsey. *Renault-Nissan gives details on autonomous vehicle plan*. URL: http://www.wsj.com/articles/renault-nissan-announces-autonomous-vehicle-plan-1452194640. (accessed: 09.16.2020).

[15] Andrew Hawkins. *VW predicts its standalone self-driving unit will be 'the world's best-funded start-up'*. URL: https://www.theverge.com/2019/10/28/20936114/vw-self-driving-startup-spinoff-argo-announce. (accessed: 09.16.2020).

[16] Michael Wayland. *Fiat Chrysler and Waymo sign exclusive deal on self-driving commercial vehicles*. URL: https://www.cnbc.com/2020/07/22/fiat-

chrysler-and-waymo-sign-deal-on-self-driving-commercial-vehicles.html. (accessed: 09.16.2020).

[17] McKinsey & Company. *Ten ways autonomous driving could redefine the automotive world*. URL: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/ten-ways-autonomous-driving-could-redefine-the-automotive-world. (accessed: 09.16.2020).

[18] Society of Automotive Engineers. *SAE International releases updated visual chart for its 'Levels of driving automation' standard for self-driving vehicles*. URL: https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles. (accessed: 09.16.2020).

[19] J. Ward, P. Smith, and D. Pierce et al. "Cooperative adaptive cruse control (CACC) in controlled and real-world environments: Testing and results". In: Ground Vehicle Systems Engineering and Technology Symposium, 2019.

[20] David Bevly. *High speed, dead reckoning, and towed implement control for automatically steered farm tractors using GPS*. PhD. Dissertation, Stanford University, 2001.

[21] P. Polack et al. "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" In: IEEE Intelligent Vehicles Symposium, 2017.

[22] Sebastian Thrun et al. "Stanley: The robot that won the DARPA grand challenge". In: *Journal of Field Robotics* 23.9 (2006), pp. 661–692.

[23] K. Åström and T. Hägglund. *PID controllers: Theory design and tuning*. NY: Instrument Society of America, 1995.

[24] A. O'Dwyer. *Handbook of PI and PID controller tuning rules*. Singapore: World Scientific Publishing, 2009.

[25] Carlos Filho and Denis Wolf. “Dynamic inversion-based control for front wheel drive autonomous ground vehicles near the limits of handling”. In: IEEE International Conference on Intelligent Transportation Systems (ITSC), 2014.

[26] 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.

[27] Eric J Rossetter. *A potential field framework for active vehicle lanekeeping assistance*. PhD Dissertation, Stanford University, 2003.

[28] F. Borrelli, P. Falcone, and T. Keviczky. “MPC-based approach to active steering for autonomous vehicle systems”. In: *International Journal of Vehicle Automation Systems* Vol. 3.2 (2005), pp. 265–291.

[29] J. B. Rawlings and D. Q. Mayne. *Model predictive control: Theory, computation, and design, Vol. II*. Madison, WI: Nob Hill Publishing, 2017.

[30] D. Q. Mayne et al. “Constrained model predictive control: Stability and optimality”. In: *Automatica* Vol. 36.6 (2000), pp. 789–814.

[31] Robert Brothers. *Path following and obstacle avoidance for autonomous ground vehicles using nonlinear model predictive control*. Master’s thesis, Auburn University, 2020.

[32] Lars Grüne and Jürgen Pannek. *Nonlinear model predictive control: Theory and algorithms*. Springer International Publishing Switzerland, 2017. ISBN: 978-3-319-46023-9.

[33] Krisada Kritayakirana and J. Christian Gerdes. “Using the centre of percussion to design a steering controller for an autonomous race car”. In: *Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility* 50.sup1 (2012), pp. 33–51.

[34] Changfang Chen et al. “Lane keeping control for autonomous 4WS4WD vehicles subject to wheel slip constraint”. In: American Control Conference (ACC), 2012.

[35] TJ Gordon, Matt C Best, and PJ Dixon. “An automated driver based on convergent vector fields”. In: *Institute of Mechanical Engineers, Part D: Journal of Automobile Engineering* 216.4 (2002), pp. 329–347.

[36] P Hingwe and M Tomizuka. “A variable look-ahead controller for lateral guidance of four wheeled vehicles”. In: American Control Conference (ACC), 1998.

[37] M.S. Netto, S Chaib, and S Mammar. “Lateral adaptive control for vehicle lane keeping”. In: American Control Conference (ACC), 2004.

[38] P. Ioannou and P. Kokotovic. “Instability analysis and improvement of robustness of adaptive control”. In: *Automatica* 20 (1984), pp. 583–594.

[39] K.S. Narendra and A.M. Annaswamy. “A new adaptive law for robust adaptive without persistent of excitation”. In: *IEEE Transaction on Automatic Control* 32 (1987), pp. 134–145.

[40] J.B. Pomet and L. Praly. “Adaptive nonlinear regulation: Estimation from Lyapunov equation”. In: *IEEE Transaction on Automatic Control* 37 (1992), pp. 729–740.

[41] Dogan et al. “Relaxing the stability limit of adaptive control systems in the presence of unmodeled dynamics”. In: *International Journal of Control* (2017), pp. 1–11.

[42] Girish Chowdhary and Eric Johnson. “Theory and flight-test validation of a concurrent learning adaptive controller”. In: *Journal of Guidance, Control and Dynamics* 34.2 (2011), pp. 592–607.

[43] Girish Chowdhary, Maximillian Müehlegg, and Eric Johnson. “Exponential parameter and tracking error convergence guarantees for adaptive controllers without persistency of excitation”. In: *International Journal of Control* 87.8 (2014), pp. 1583–1603.

[44] Vehicle Dynamics Standards Committee. *Vehicle dynamics terminology, SAE J670*. Warrendale, PA: Society of Automotive Engineers, 2008.

[45] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary. ISO 8855:2011*. Geneva, Switzerland: International Organization for Standardization, 2011.

[46] Hum3D. *Porsche 935 2019 3D model*. URL: `https://hum3d.com/3d-models/porsche-935-2019`. (accessed: 08.17.2020).

[47] Jay A. Ferrell. *Aided navigation: GPS with high rate sensors*. McGraw-Hill Company, 2008. ISBN: 0-07-164266-8.

[48] OpenCV. *Camera calibration and 3D reconstruction*. URL: https://docs.opencv.org/2.4/modules/calib3d/doc/%20camera%5C_calibration%5C_and%5C_3d%5C_reconstruction.html. (accessed: 08.18.2020).

[49] Hans B. Pacejka. *Tyre and vehicle dynamics, Second edition*. Butterworth-Heinemann Publications, 2006. ISBN: 9780750669184.

[50] E. Fiala. "Seitenkraften am rollenden luftreifen (in German)". In: *VDI-Zeitschrift* Vol. 96.29 (Oct. 1954), pp. 973–979.

[51] E. Bakker, L. Nyborg, and H. Pacejka. "Tire modeling for use in vehicle dynamic studies". In: *Society of Automotive Engineers* Paper No. 870421 (1987).

[52] E. Bakker, H. Pacejka, and L. Lidner. "A new tire model with an application in vehicle dynamic studies". In: *Society of Automotive Engineers* Paper No. 890087 (1989).

[53] H.B. Pacejka and E. Bakker. "The magic tyre model". In: Proceedings of 1st Colloquium on Tyre Models for Vehicle Analysis, Delft, ed. H.B. Pacejka, Suppl. Vehicle System Dynamics, 21, 1993, 1993.

[54] J. Kong et al. "Kinematic and dynamic vehicle models for autonomous driving control design". In: IEEE Intelligent Vehicles Symposium, Seoul, 2015.

[55] Thomas D. Gillespie. *Fundamentals of vehicle dynamics*. Society of Automotive Engineers, Inc, 1992. ISBN: 978-1-56091-199-9.

[56] Lowell Brown. *Determination of the allowable parametric uncertainty in the closed loop via a skew-μ based technique with an application to the yaw-roll vehicle model*. PhD. Dissertation, Auburn University, 2017.

[57] Open Source Robotics Foundation. *Gazebo*. URL: http://gazebosim.org. (accessed: 08.21.2020).

[58] Mechanical Simulation Corporation. *CarSim*. URL: https://www.carsim.com. (accessed: 08.21. 2020).

[59] Microsoft. *Microsoft AirSim: Open source simulator for autonomous vehicles*. URL: https://github.com/microsoft/AirSim. (accessed: 08.21.2020).

[60] IPG Automotive. *CarMaker simulation software*. URL: https://ipg-automotive.com/products-services/simulation-software/carmaker/. (accessed: 08.21.2020).

[61] N. Koenig and A. Howard. “Design and use paradigms for Gazebo, an open-source multi-robot simulator”. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sept. 2004, pp. 2149–2154.

[62] Jerrod M. Snider. *Automatic steering methods for autonomous automobile path tracking*. Robotics Institute, Carnegie Mellon University, 2009.

[63] MathWorks. *MATLAB Simulink*. URL: https://www.mathworks.com/products/matlab.html. (accessed: 08.25.2020).

[64] VEHICO. *ISO lane change test*. URL: https://www.vehico.com/index.php/en/applications/iso-lane-change-test. (accessed: 08.20.2020).

[65] F. N. Fritsch and R. E. Carlson. “Monotone piecewise cubic interpolation”. In: *SIAM Journal on Numerical Analysis* Vol. 17 (1980), pp. 238–246.

[66] P. J. Green and B. W. Silverman. *Nonparametric regression and generalized linear models: A roughness penalty approach*. Chapman and Hall, 1994.

[67] Matthew C. Best. “Real time characterisation of driver steering behaviour”. In: *Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility* 57.1 (2019), pp. 64–85.

[68] 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: International Journal of Vehicle Mechanics and Mobility* 53.12 (2015), pp. 1687–1704.

[69] N. R. Kapania and J. C. Gerdes. “An autonomous lanekeeping system for vehicle path tracking and stability at the limits of handling”. In: International Symposium on Advanced Vehicle Control, Tokyo, 2014.

[70] Liuping Wang. *Model predictive control system design and implementation using MATLAB*. Advances in Industrial Control. Springer-Verlag London Limited, 2009. ISBN: 978-1-84882-330-3.

[71] Eugene Lavretsky. “Robust and adaptive control methods for aerial vehicles”. In: *Handbook of Unmanned Aerial Vehicles* (2015), pp. 675–710.

[72] Gang Tao. “Multivariable adaptive control: A survey”. In: *Automatica* 50.11 (2014), pp. 2737–2764.

[73] Tansel Yucelen. *Model reference adaptive control*. Wiley Encyclopedia of Electrical and Electronics Engineering, 2019.

[74] Open Source Robotics Foundation. *RViz*. URL: `http://wiki.ros.org/rviz`. (accessed: 09.09.2020).

[75] Dataspeed Inc. *Drive-by-wire kit, an autonomous vehicle by-wire solution*. URL: `https://www.dataspeedinc.com/adas-by-wire-system/`. (accessed: 09.08.2020).

[76] RACECAR/J. *RACECAR/J Build Instructions*. URL: `https://racecarj.com/build`. (accessed: 10.28.2020).

[77] W. Travis et al. “Non-line-of-sight automated vehicle following using a dynamic base RTK system”. In: *NAVIGATION, Journal of the Institute of Navigation* 58.11 (2011), pp. 241–255.

[78] Scott Martin. *Closely coupled GPS/INS relative positioning for automated vehicle convoys*. Master’s thesis, Auburn University, 2011.

[79] Irwin Sobel. “History and definition of the so-called ’Sobel Operator’, more appropriately named the Sobel-Feldman Operator”. In: *a public letter* (February 2, 2014, Updated June 14, 2015).

[80] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. USA: Prentice-Hall, Inc., 2006. ISBN: 013168728X.

[81] Tom Bruls et al. “The Right (Angled) Perspective: Improving the Understanding of Road Scenes using Boosted Inverse Perspective Mapping”. In: *CoRR* abs/1812.00913 (2018). URL: http://arxiv.org/abs/1812.00913.

[82] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2nd ed. Cambridge University Press, 2004.

[83] J. C. McCall and M. M. Trivedi. “Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation”. In: *IEEE Transactions on Intelligent Transportation Systems* 7.1 (2006), pp. 20–37.

[84] M. P. Batista et al. “Lane Detection and Estimation using Perspective Image”. In: *2014 Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol*. 2014, pp. 25–30.

[85] OpenCV. *Open Source Computer Vision Library*. 2020.

[86] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large scale image recognition”. In: International Conference on Learning Respresentations (ICLR), 2015.

[87] William Bryan and Matthew Boler et al. “A vision-based lane keeping system using a cascaded adaptive controller”. In: American Control Conference, 2020.

[88] TuSimple. *TuSimple dataset*. URL: https://github.com/TuSimple/tusimple-benchmark/issues/3. (accessed: 09.15.2020).

[89] M.A. Fischler and R.C. Bolles. “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[90] Sean Brennan. *Modelling and control issues associated with scaled vehicles*. Master’s Thesis, University of Illinois at Urbana-Champaign, 1999.

[91] Sean Brennan. *On size and control: The use of dimensional analysis in controller design*. PhD Dissertation, University of Illinois at Urbana-Champaign, 2002.

[92] Sigurd Skogestad and Ian Postlethwaite. *Multivariable feedback control: Analysis and design*. Wiley, 2005.

Appendices

Appendix A

Additional Sensitivity Analysis Plots

This Appendix provides additional plots generated during the sensitivity analysis presented in Chapter 3. Each section of this appendix provides the results from each controller tested. This includes a table with the percent differences in max and mean lateral errors when compared to the nominal case and a plot showing the position of the vehicle through the double lane change maneuver and the errors for each run. These tables and plots are provided for each dynamic bicycle model parameter and each controller.

A.1 Kinematic Control

Table A.1: Kinematic Controller Sensitivity to Front Tire Cornering Stiffness.

	Front Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	322.7256%	117.5685%	0%	-63.7185%	-68.3264%
Lateral Mean Error	231.5222%	90.4433%	0%	-56.0324%	-65.7493%

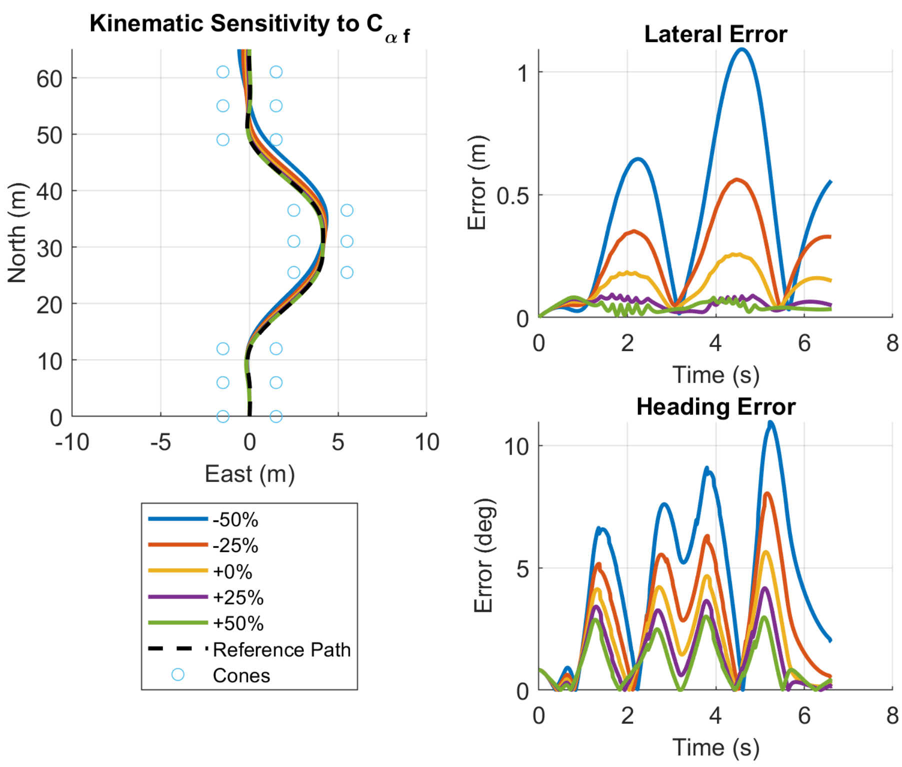


Figure A.1: Kinematic Controller Sensitivity to Front Tire Cornering Stiffness.

Table A.2: Kinematic Controller Sensitivity to Rear Tire Cornering Stiffness.

	Rear Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	56.9976%	-24.5684%	0%	21.2026%	29.8571%
Lateral Mean Error	15.4529%	-10.2711%	0%	12.4080%	18.3590%

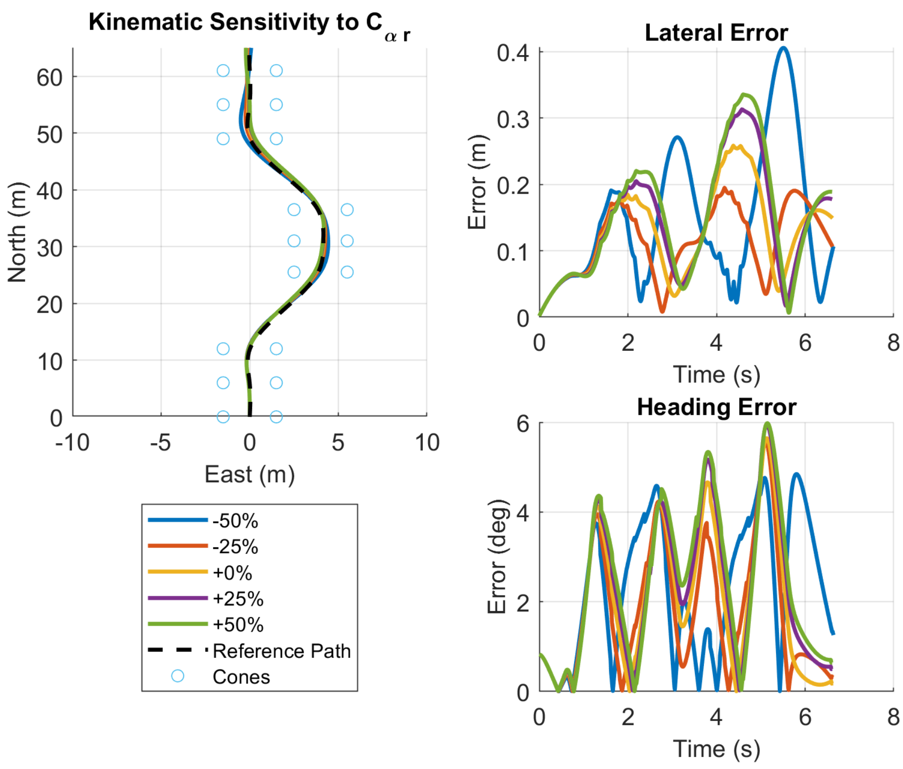


Figure A.2: Kinematic Controller Sensitivity to Rear Tire Cornering Stiffness.

Table A.3: Kinematic Controller Sensitivity to Distance from CG to Front Axle.

	Distance from Front Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-8.9266%	-28.0759%	0%	45.5338%	92.0615%
Lateral Mean Error	-9.8690%	-16.4082%	0%	30.3197%	61.9525%

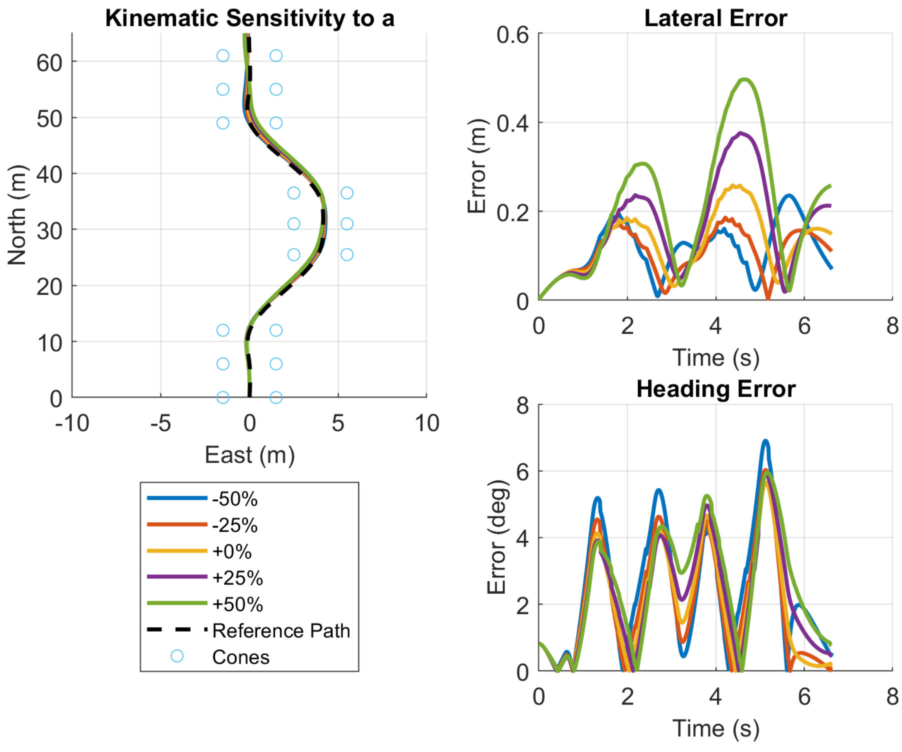


Figure A.3: Kinematic Controller Sensitivity to Distance from CG to Front Axle.

Table A.4: Kinematic Controller Sensitivity to Distance from CG to Rear Axle.

	Distance from Rear Axle to CG				
	-50%*	-25%	±0%	+25%	+50%
Lateral Max Error	389.3254%	45.0572%	0%	109.6969%	209.1717%
Lateral Mean Error	320.4830%	8.7196%	0%	67.0709%	131.4614%
* went unstable					

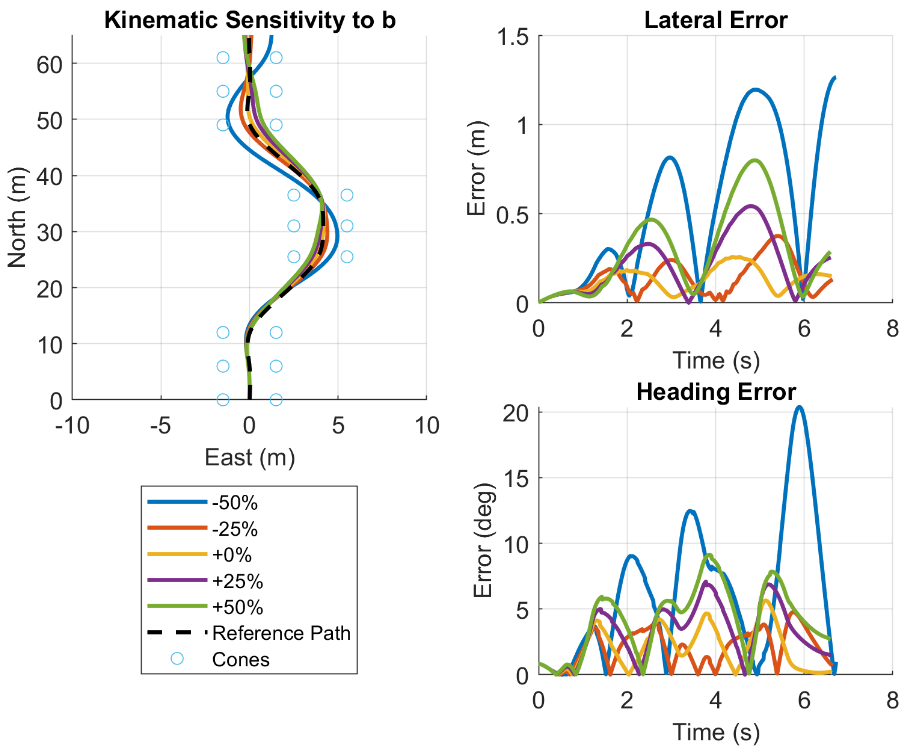


Figure A.4: Kinematic Controller Sensitivity to Distance from CG to Front Axle.

Table A.5: Kinematic Controller Sensitivity to Vehicle Mass.

	Vehicle Mass				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-67.1839%	-44.3518%	0%	50.1214%	101.8426%
Lateral Mean Error	-57.6658%	-33.3741%	0%	38.3969%	76.9420%

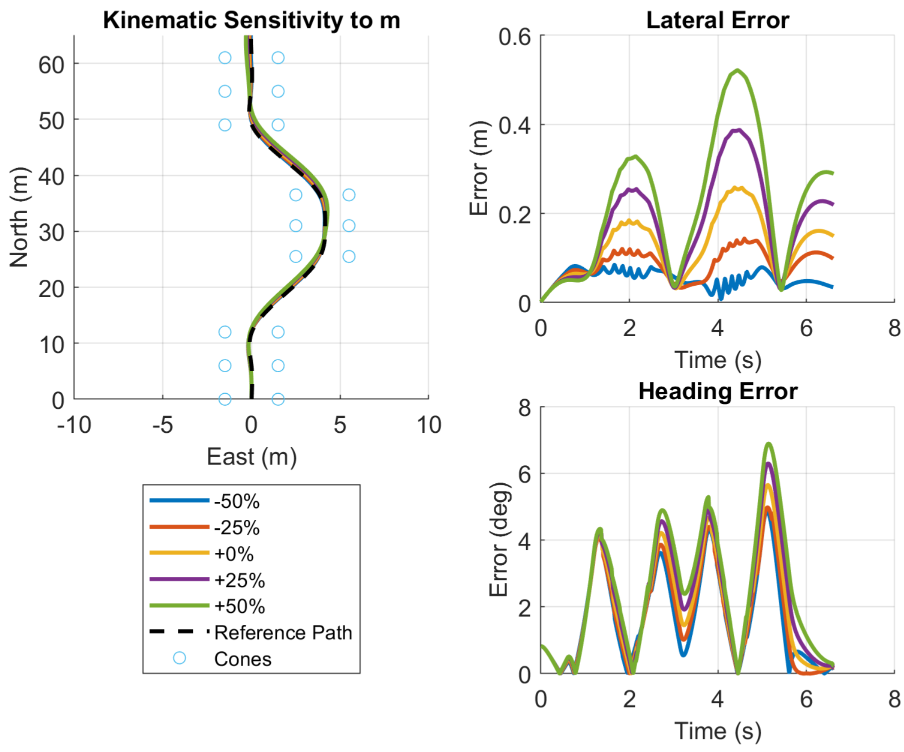


Figure A.5: Kinematic Controller Sensitivity to Vehicle Mass.

Table A.6: Kinematic Controller Sensitivity to Vehicle Inertia.

	Vehicle Inertia about the Vertical Axis				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-12.0534%	-11.0627%	0%	18.7419%	39.1621%
Lateral Mean Error	-21.3320%	-12.3113%	0%	19.3750%	44.9370%

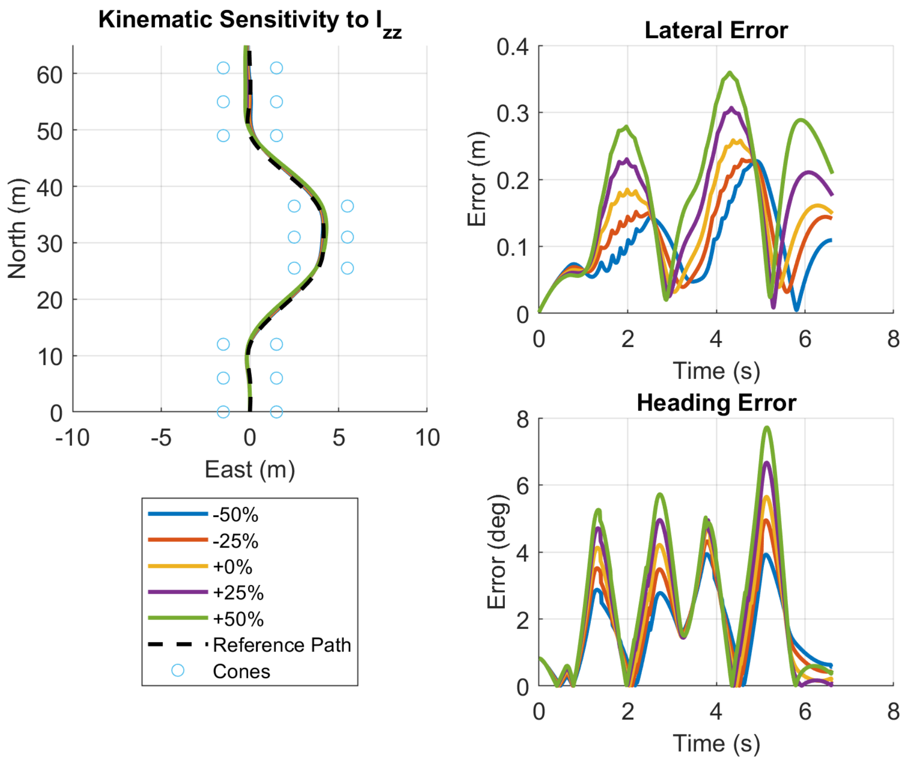


Figure A.6: Kinematic Controller Sensitivity to Vehicle Inertia.

A.2 Dynamic Bicycle Model Pole Placement

Table A.7: Dynamic Bicycle Model Pole Placement Sensitivity to Front Tire Cornering Stiffness.

	Front Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	265.6480%	44.8797%	0%	26.9988%	76.4405%
Lateral Mean Error	215.3706%	58.6238%	0%	5.9065%	31.8951%

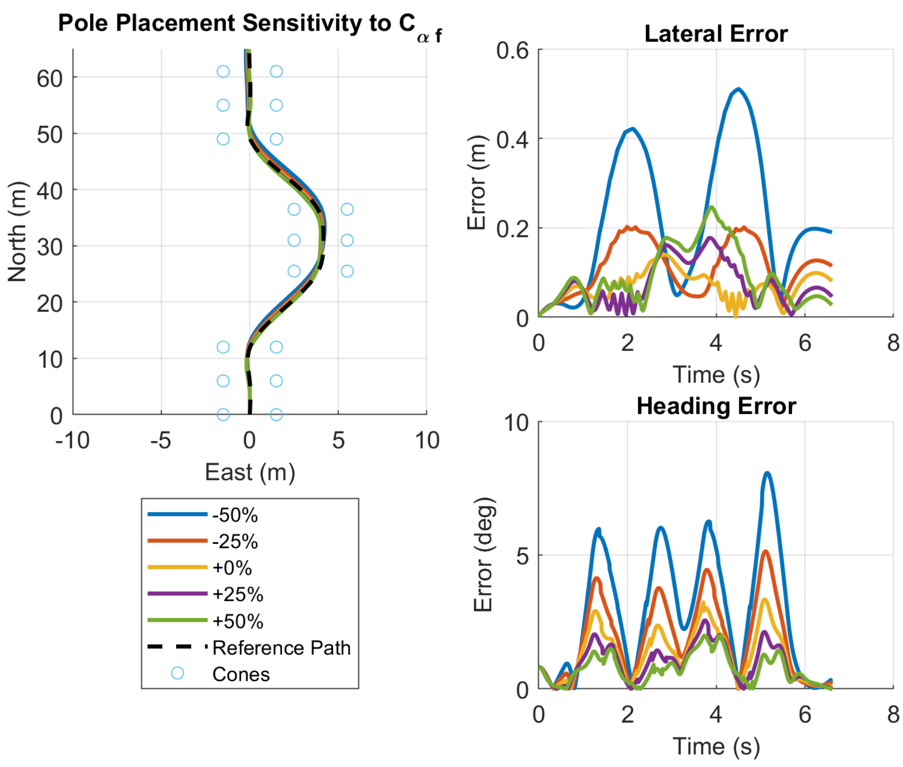


Figure A.7: Dynamic Bicycle Model Pole Placement Sensitivity to Front Tire Cornering Stiffness.

Table A.8: Dynamic Bicycle Model Pole Placement Sensitivity to Rear Tire Cornering Stiffness.

	Rear Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	495.1428%	129.3531%	0%	16.8321%	97.1306%
Lateral Mean Error	328.0971%	100.2609%	0%	9.7209%	45.1294%

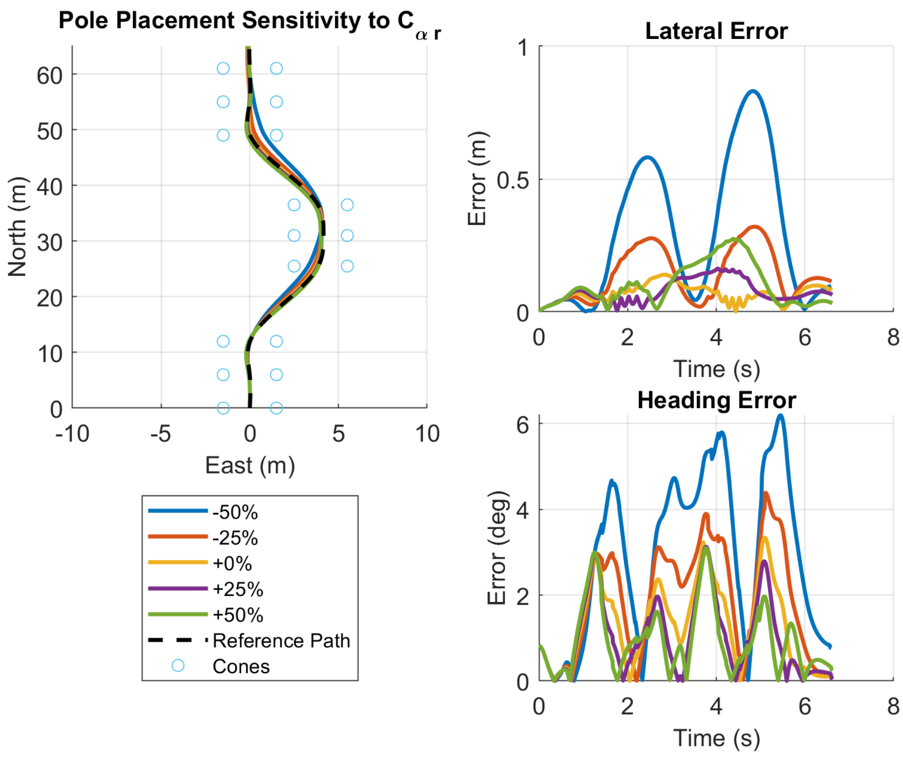


Figure A.8: Dynamic Bicycle Model Pole Placement Sensitivity to Rear Tire Cornering Stiffness.

Table A.9: Dynamic Bicycle Model Pole Placement Sensitivity to Distance from CG to Front Axle.

	Distance from Front Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	1615.3959%	52.3724%	0%	41.7685%	162.4251%
Lateral Mean Error	883.7134%	31.3123%	0%	48.8857%	126.3656%

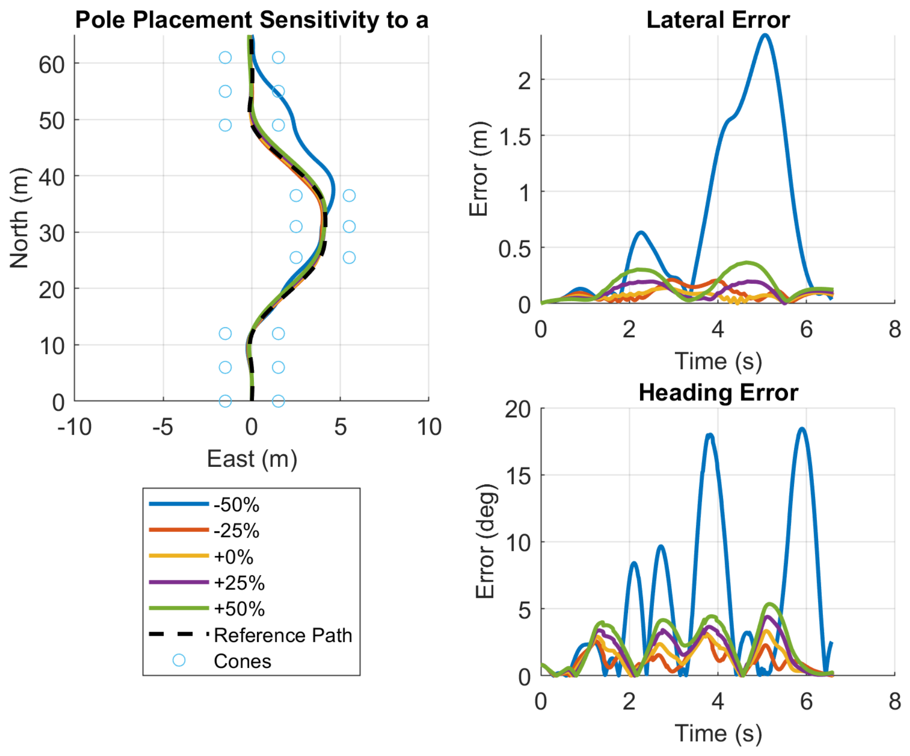


Figure A.9: Dynamic Bicycle Model Pole Placement Sensitivity to Distance from CG to Front Axle.

Table A.10: Dynamic Bicycle Model Pole Placement Sensitivity to Distance from CG to Rear Axle.

	Distance from Rear Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	858.2235%	404.3638%	0%	325.7266%	594.0291%
Lateral Mean Error	595.2393%	277.9231%	0%	212.8811%	484.8787%

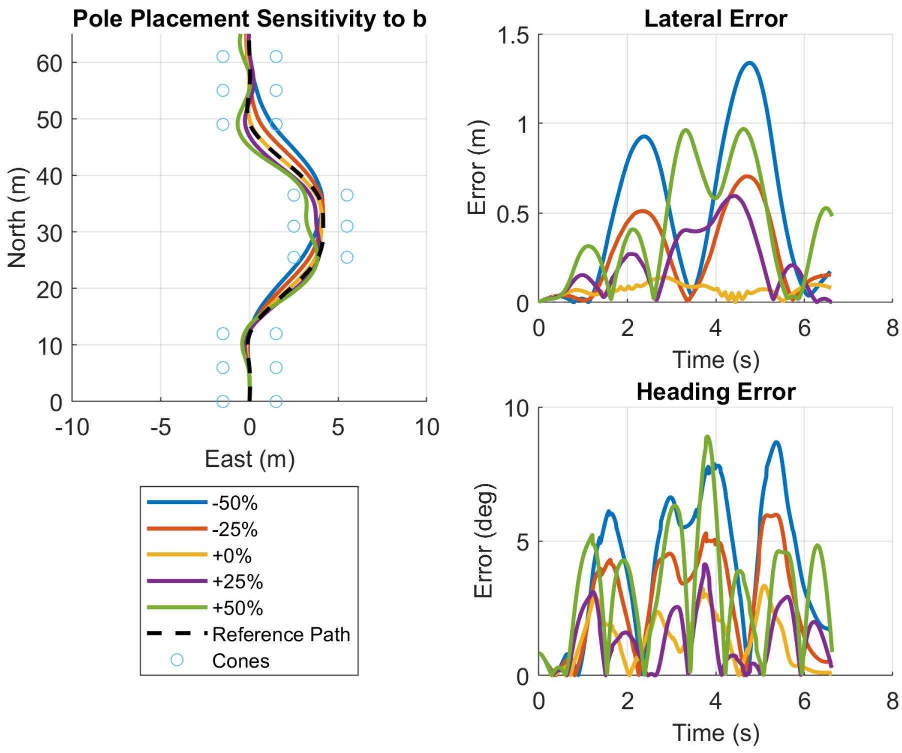


Figure A.10: Dynamic Bicycle Model Pole Placement Sensitivity to Distance from CG to Front Axle.

Table A.11: Dynamic Bicycle Model Pole Placement Sensitivity to Vehicle Mass.

	Vehicle Mass				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	451.1247%	136.1228%	0%	125.4351%	342.9053%
Lateral Mean Error	271.9236%	68.8859%	0%	107.5359%	250.8236%

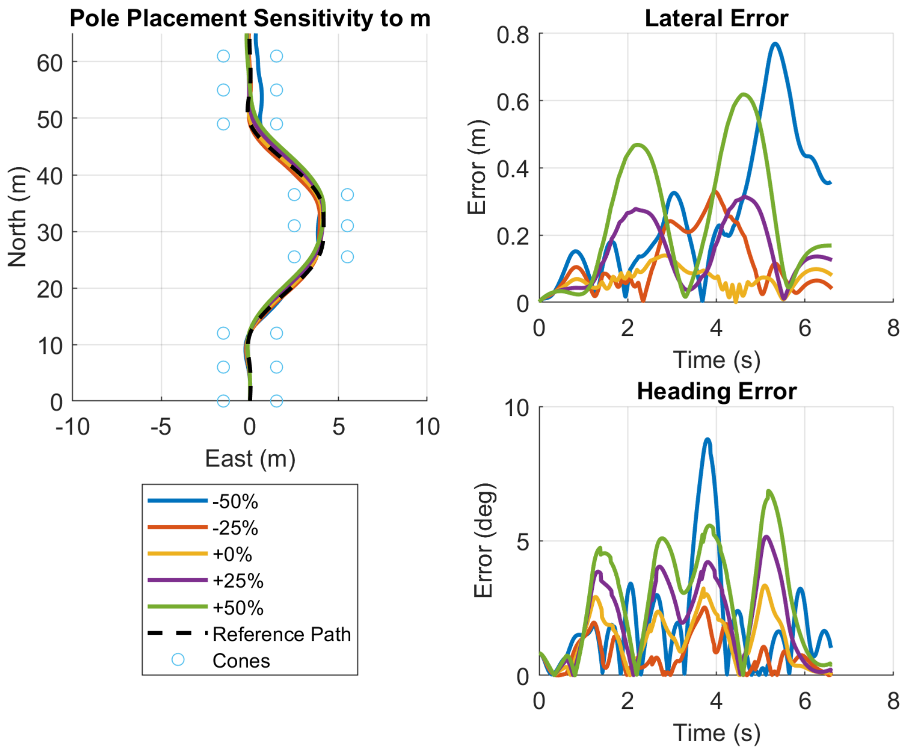


Figure A.11: Dynamic Bicycle Model Pole Placement Sensitivity to Vehicle Mass.

Table A.12: Dynamic Bicycle Model Pole Placement Sensitivity to Vehicle Inertia.

	Vehicle Inertia about the Vertical Axis				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-36.5308%	-37.3478%	0%	46.1689%	101.7763%
Lateral Mean Error	-26.9263%	-16.5193%	0%	36.7092%	93.4664%

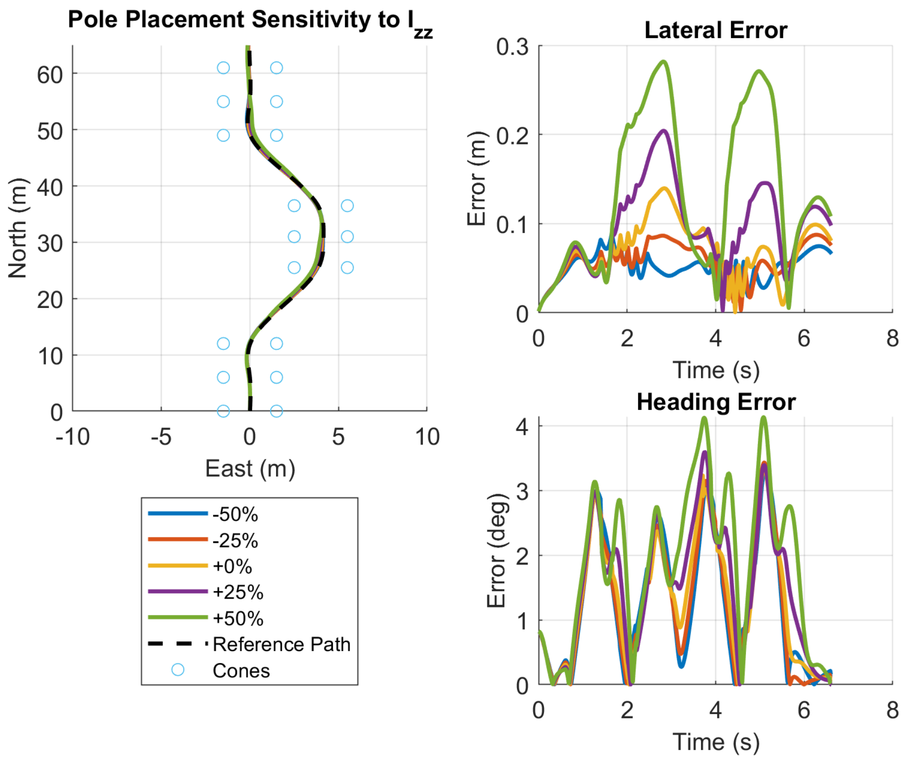


Figure A.12: Dynamic Bicycle Model Pole Placement Sensitivity to Vehicle Inertia.

A.3 Lookahead Error Feedback-Feedforward Control

Table A.13: Lookahead Error Controller Sensitivity to Front Tire Cornering Stiffness.

	Front Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-42.6600%	-30.8900%	0%	25.9147%	41.6449%
Lateral Mean Error	-47.1048%	-26.0449%	0%	19.5667%	33.6581%

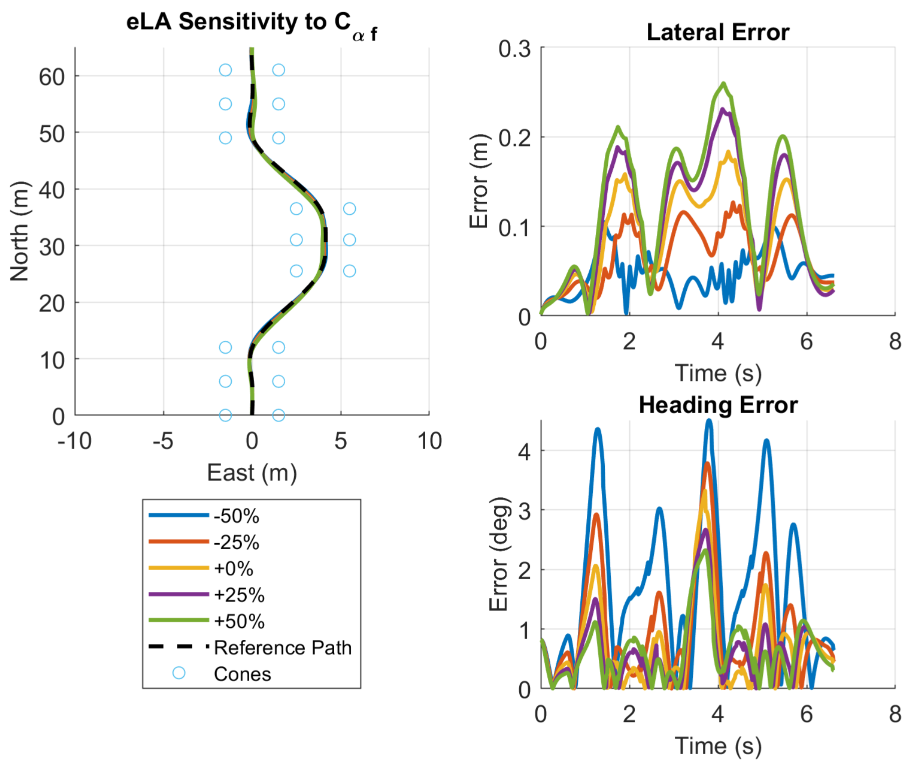


Figure A.13: Lookahead Error Controller Sensitivity to Front Tire Cornering Stiffness.

Table A.14: Lookahead Error Controller Sensitivity to Rear Tire Cornering Stiffness.

	Rear Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-40.2367%	-16.8213%	0%	9.7621%	16.4556%
Lateral Mean Error	-28.9954%	-10.0443%	0%	5.7898%	10.1955%

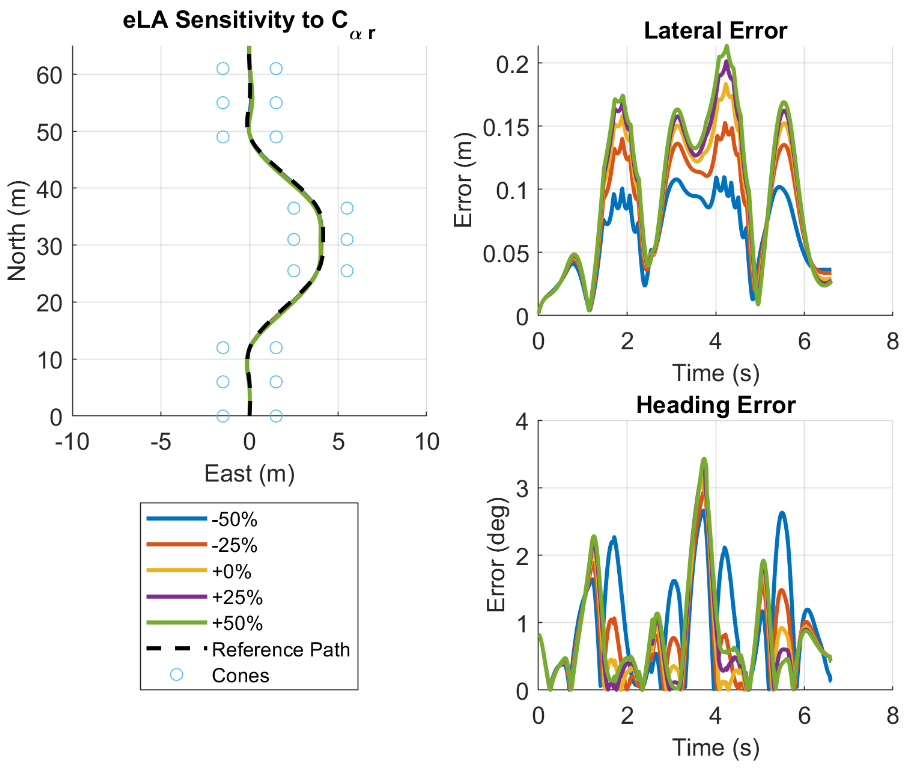


Figure A.14: Lookahead Error Controller Sensitivity to Rear Tire Cornering Stiffness.

Table A.15: Lookahead Error Controller Sensitivity to Distance from CG to Front Axle.

	Distance from Front Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	19.9366%	14.6252%	0%	-19.9928%	-37.4552%
Lateral Mean Error	25.1147%	13.6619%	0%	-15.6155%	-30.6208%

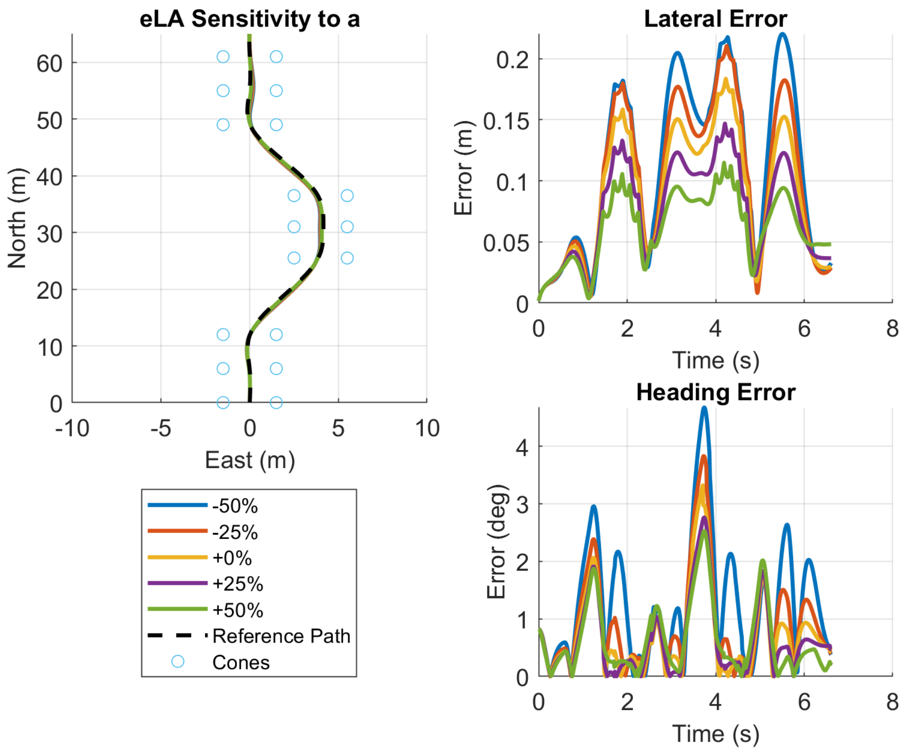


Figure A.15: Lookahead Error Controller Sensitivity to Distance from CG to Front Axle.

Table A.16: Lookahead Error Controller Sensitivity to Distance from CG to Rear Axle.

	Distance from Rear Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-24.0996%	-28.2905%	0%	26.3384%	44.4573%
Lateral Mean Error	-33.8382%	-18.6184%	0%	18.6995%	34.7928%

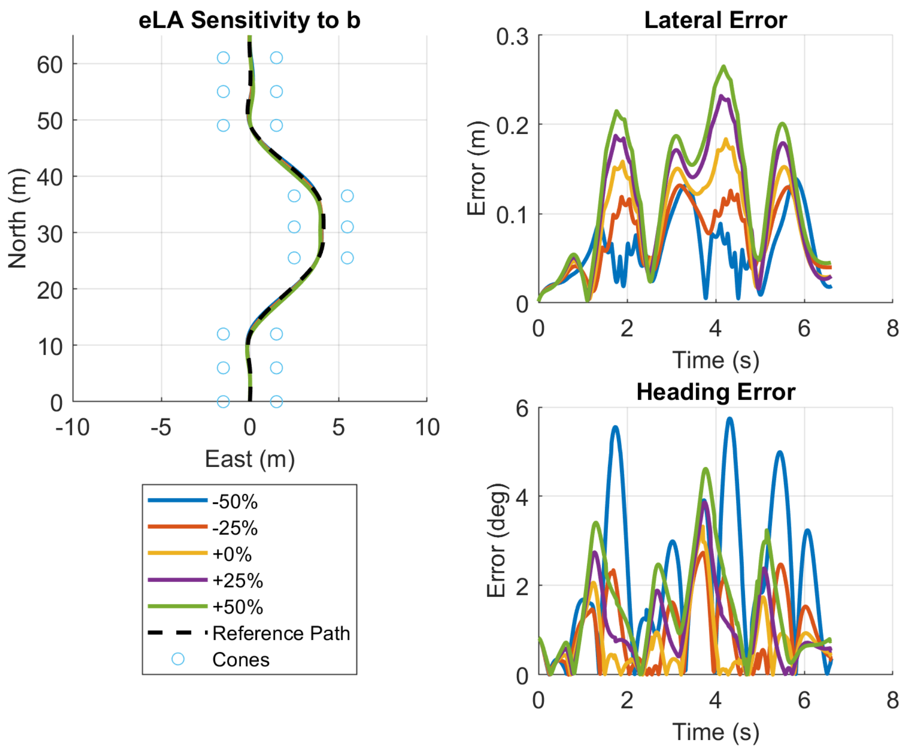


Figure A.16: Lookahead Error Controller Sensitivity to Distance from CG to Front Axle.

Table A.17: Lookahead Error Controller Sensitivity to Vehicle Mass.

	Vehicle Mass				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	25.1487%	26.1422%	0%	-26.6354%	-48.5362%
Lateral Mean Error	37.3091%	23.8943%	0%	-21.6194%	-48.5637%

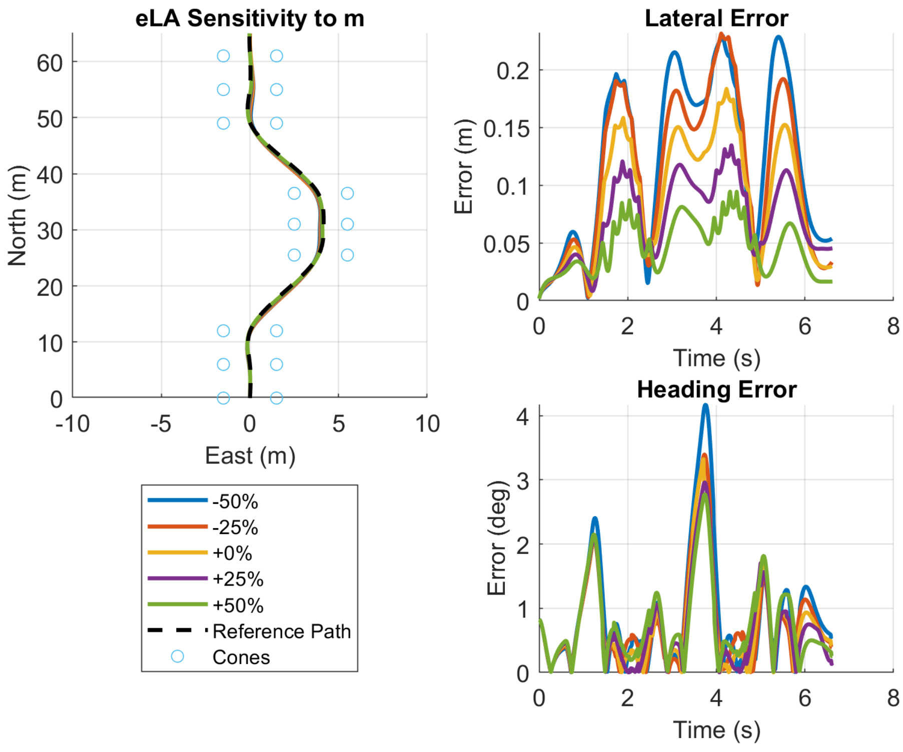


Figure A.17: Lookahead Error Controller Sensitivity to Vehicle Mass.

Table A.18: Lookahead Error Controller Sensitivity to Vehicle Inertia.

	Vehicle Inertia about the Vertical Axis				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	18.2489%	10.9044%	0%	-15.2157%	-20.1874%
Lateral Mean Error	10.8415%	5.9751%	0%	-6.7691%	-13.3448%

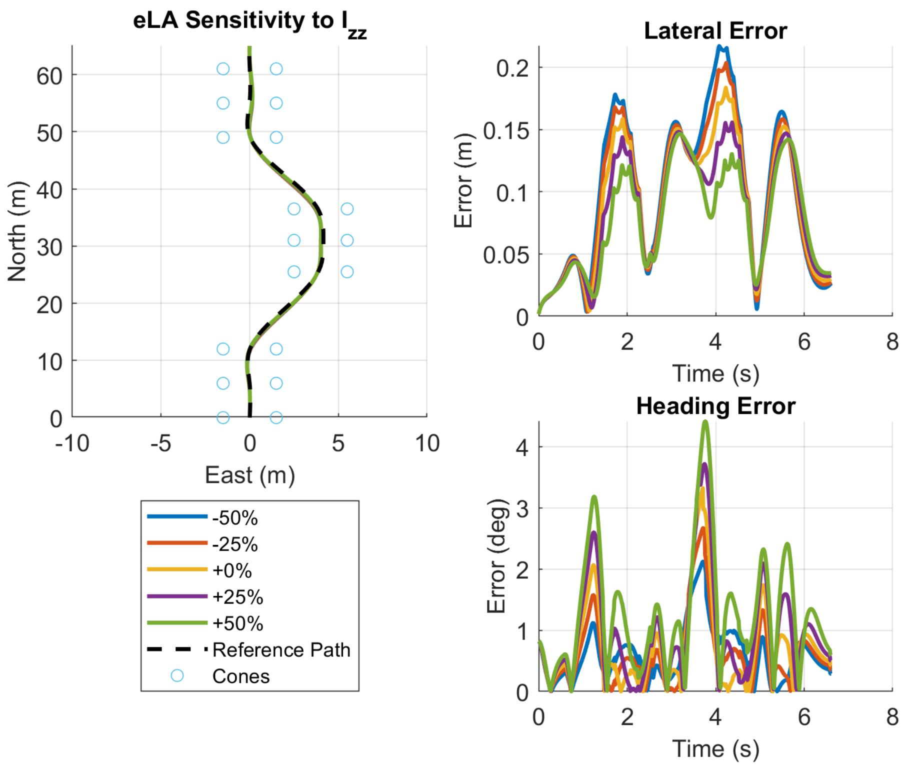


Figure A.18: Lookahead Error Controller Sensitivity to Vehicle Inertia.

A.4 Model Predictive Control

Table A.19: MPC Sensitivity to Front Tire Cornering Stiffness.

	Front Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	106.6341%	38.5803%	0%	-17.9141%	-29.2562%
Lateral Mean Error	93.2763%	30.1062%	0%	-17.9598%	-27.5641%

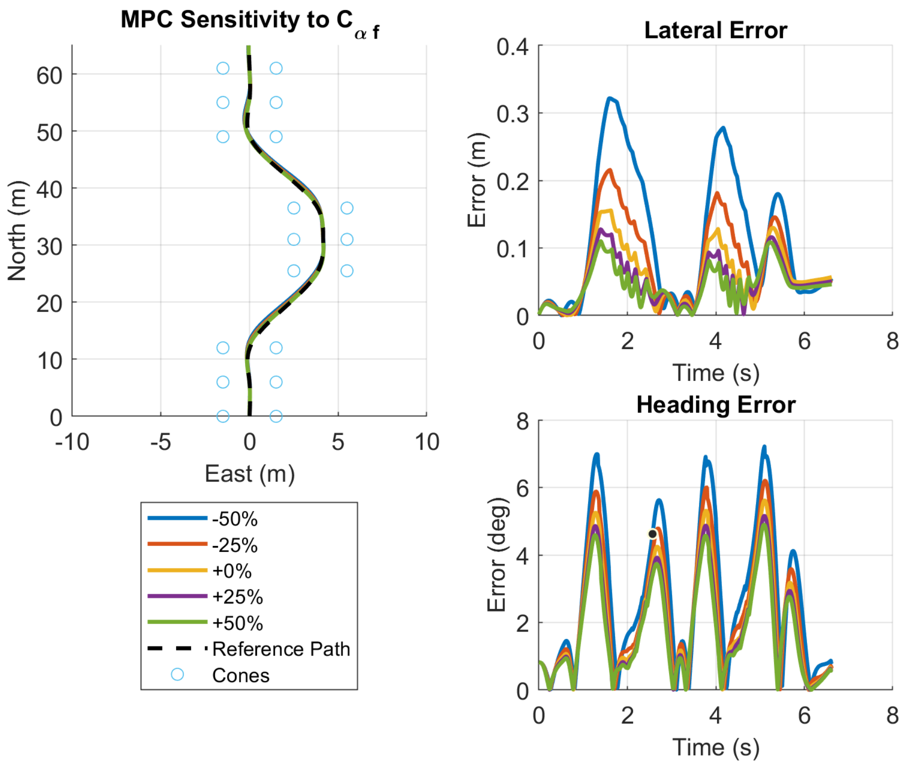


Figure A.19: MPC Sensitivity to Front Tire Cornering Stiffness.

Table A.20: MPC Sensitivity to Rear Tire Cornering Stiffness.

	Rear Tire Cornering Stiffness				
	-50%*	-25%	±0%	+25%	+50%
Lateral Max Error	17.1317%	4.3575%	0%	-1.3302%	-1.9239%
Lateral Mean Error	37.8536%	-3.2005%	0%	-4.0159%	-4.5649%

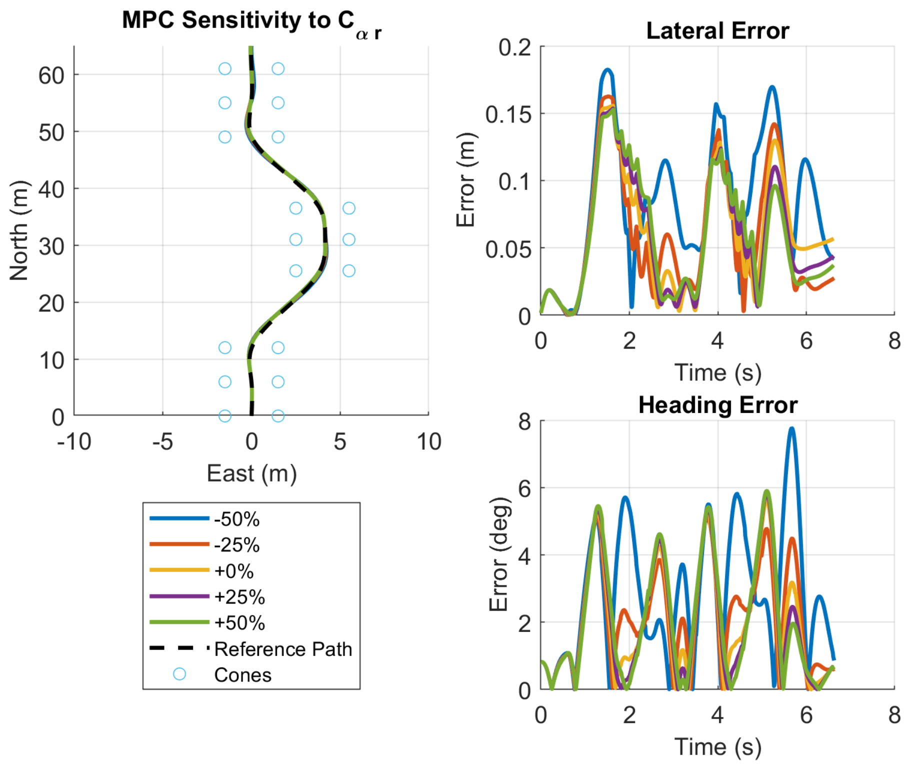


Figure A.20: MPC Sensitivity to Rear Tire Cornering Stiffness.

Table A.21: MPC Sensitivity to Distance from CG to Front Axle.

	Distance from Front Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	14.7139%	3.3050%	0%	1.6186%	6.7938%
Lateral Mean Error	12.2618%	0.0312%	0%	2.5210%	8.8823%

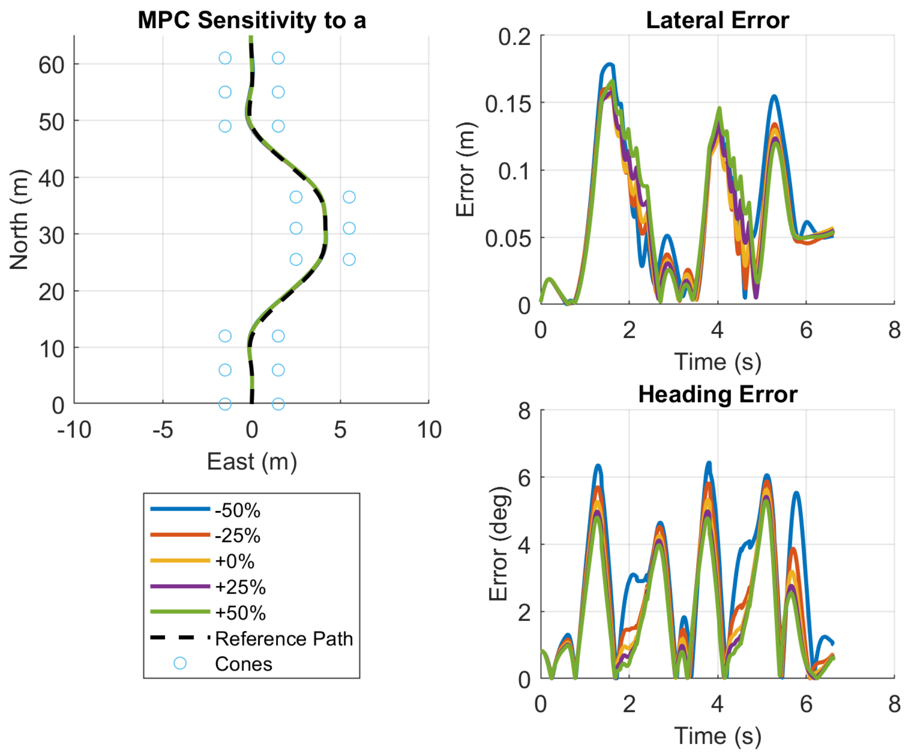


Figure A.21: MPC Sensitivity to Distance from CG to Front Axle.

Table A.22: MPC Sensitivity to Distance from CG to Rear Axle.

	Distance from Rear Axle to CG				
	-50%*	-25%	±0%	+25%	+50%
Lateral Max Error	850.9560%	26.0444%	0%	4.6793%	32.2816%
Lateral Mean Error	551.4695%	30.4607%	0%	19.2355%	38.4103%
* went unstable					

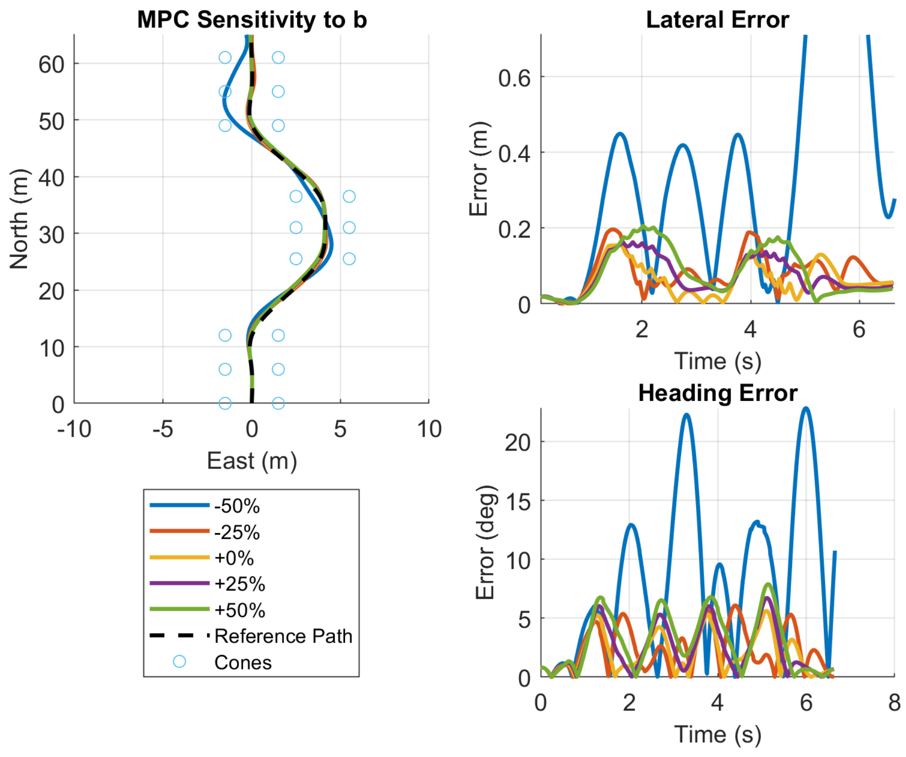


Figure A.22: MPC Sensitivity to Distance from CG to Rear Axle.

Table A.23: MPC Sensitivity to Vehicle Mass.

	Vehicle Mass				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-5.2265%	-9.1503%	0%	13.7197%	29.0769%
Lateral Mean Error	-15.6694%	-14.1902%	0%	6.4247%	15.0244%

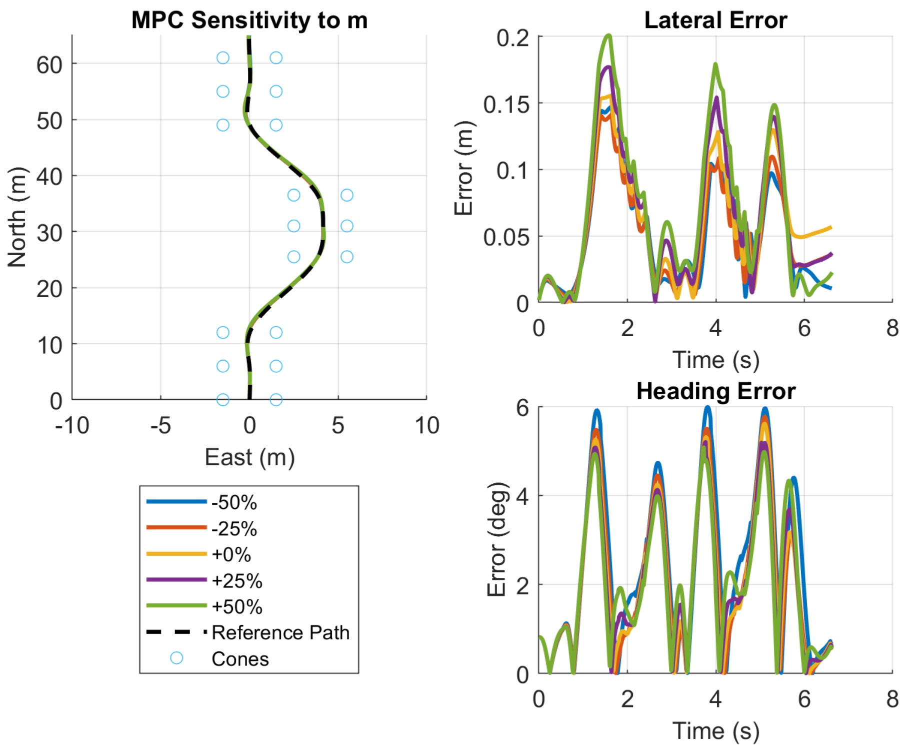


Figure A.23: MPC Sensitivity to Vehicle Mass.

Table A.24: MPC Sensitivity to Vehicle Inertia.

	Vehicle Inertia about the Vertical Axis				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-25.3448%	-14.1412%	0%	20.5596%	44.2323%
Lateral Mean Error	-26.8291%	-13.6050%	0%	10.9468%	25.3582%

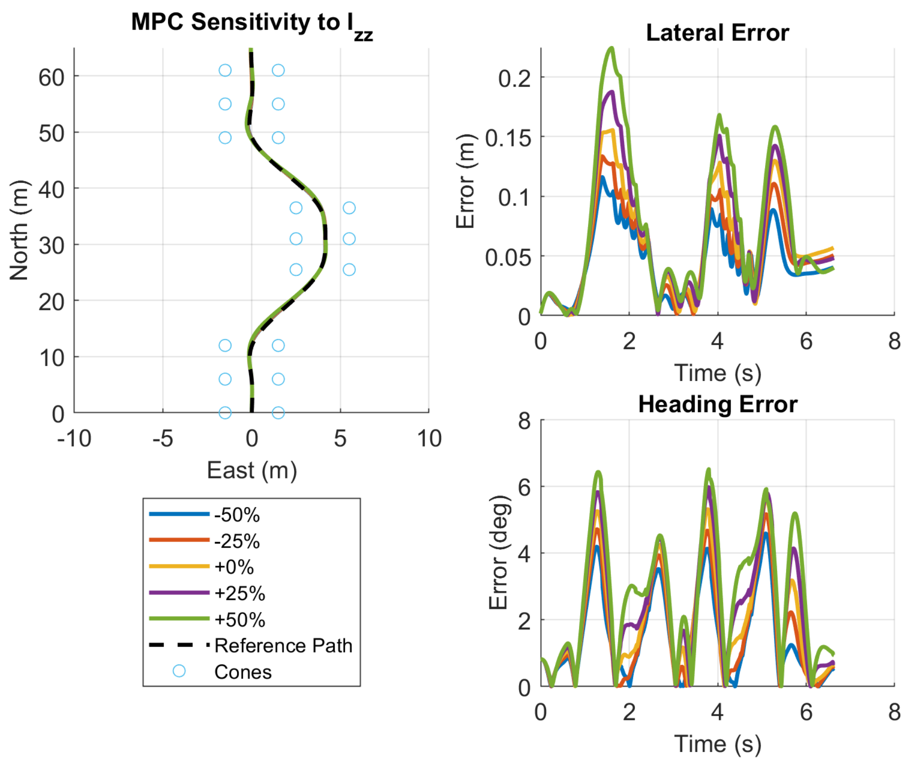


Figure A.24: MPC Sensitivity to Vehicle Inertia.

A.5 Model Reference Adaptive Control

Table A.25: MRAC Sensitivity to Front Tire Cornering Stiffness.

	Front Tire Cornering Stiffness				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	0.9961%	0.4855%	0%	-0.2167%	-1.0677%
Lateral Mean Error	-0.1376%	-0.0443%	0%	-0.9665%	-0.9332%

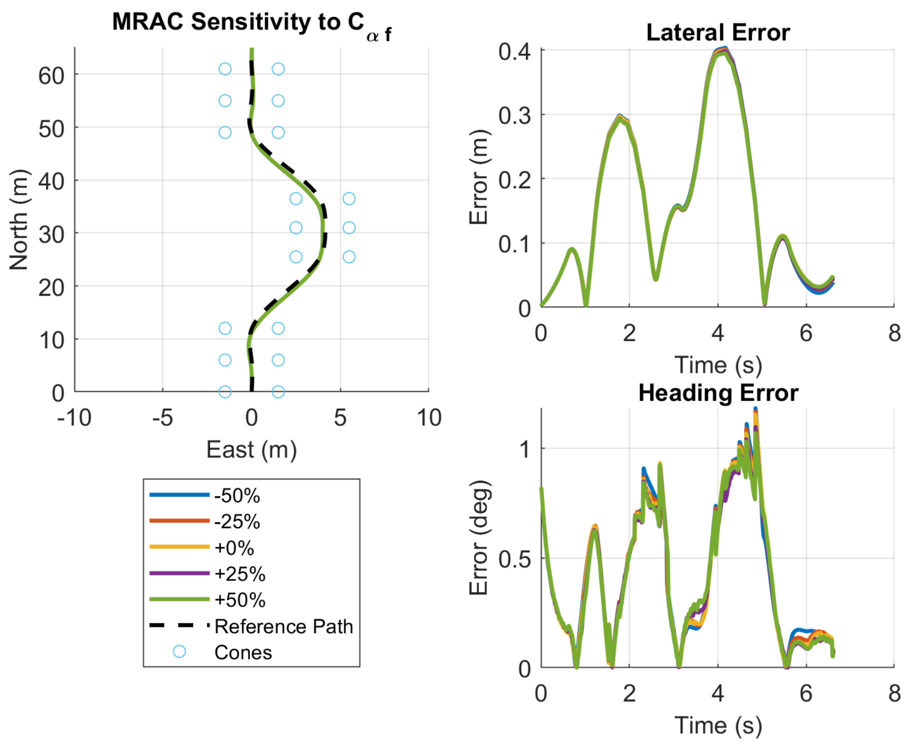


Figure A.25: MRAC Sensitivity to Front Tire Cornering Stiffness.

Table A.26: MRAC Sensitivity to Rear Tire Cornering Stiffness.

	Rear Tire Cornering Stiffness				
	-50%*	-25%	±0%	+25%	+50%
Lateral Max Error	-40.3929%	-15.8747%	0%	10.6058%	17.7489%
Lateral Mean Error	-41.1114%	-17.9962%	0%	10.6246%	18.4777%

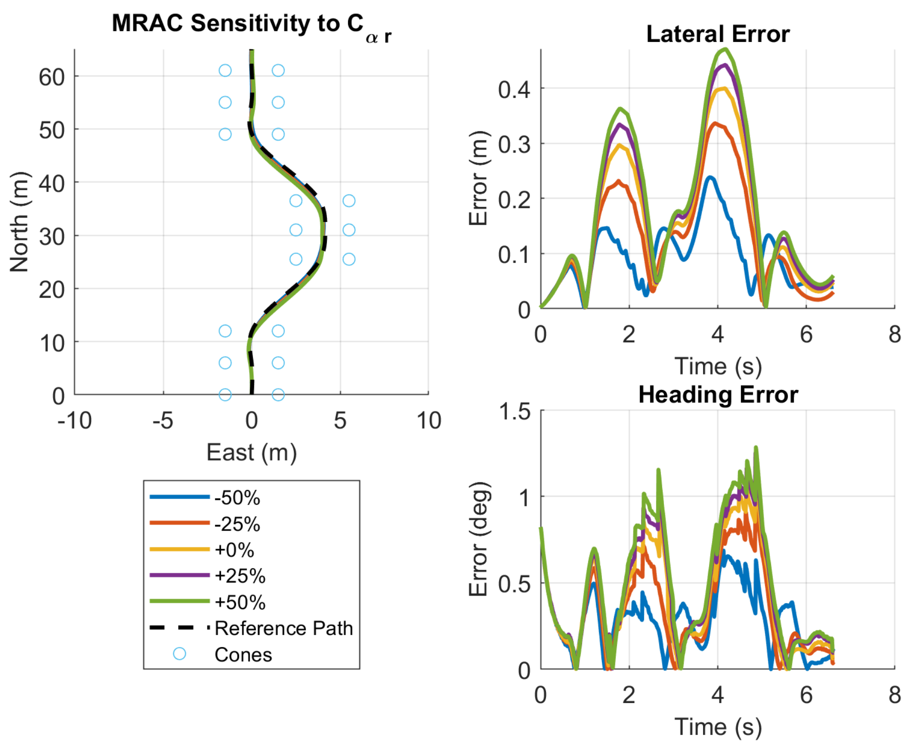


Figure A.26: MRAC Sensitivity to Rear Tire Cornering Stiffness.

Table A.27: MRAC Sensitivity to Distance from CG to Front Axle.

	Distance from Front Axle to CG				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	12.9998%	7.6768%	0%	-7.9420%	-16.6665%
Lateral Mean Error	12.6069%	9.2510%	0%	-8.6830%	-14.8298%

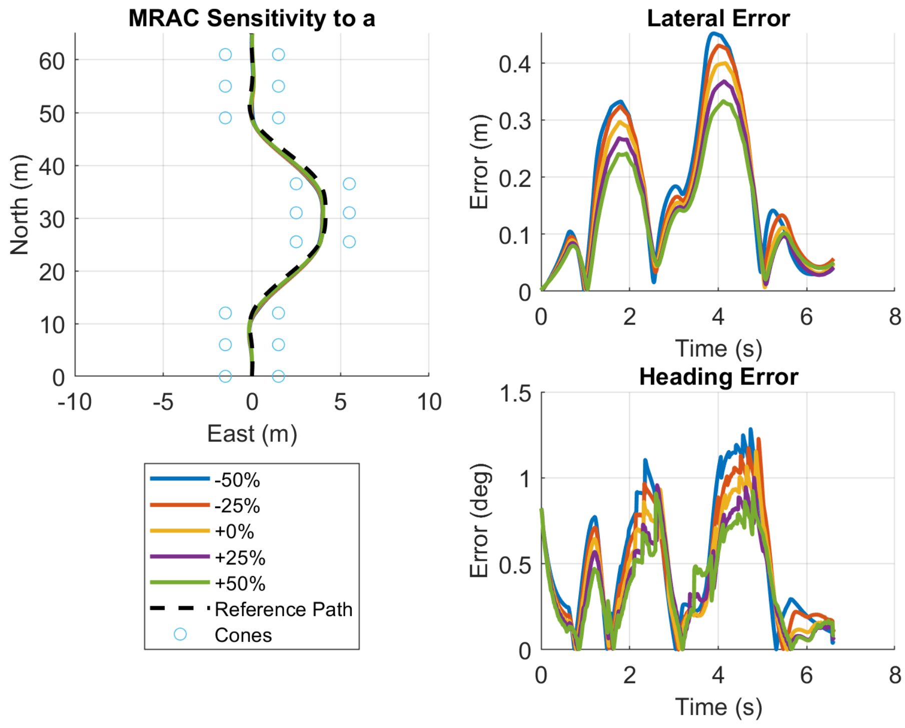


Figure A.27: MRAC Sensitivity to Distance from CG to Front Axle.

Table A.28: MRAC Sensitivity to Distance from CG to Rear Axle.

	Distance from Rear Axle to CG				
	-50%*	-25%	±0%	+25%	+50%
Lateral Max Error	-62.9670%	-42.7566%	0%	41.2211%	77.7106%
Lateral Mean Error	-57.2614%	-47.0936%	0%	44.9714%	84.1349%

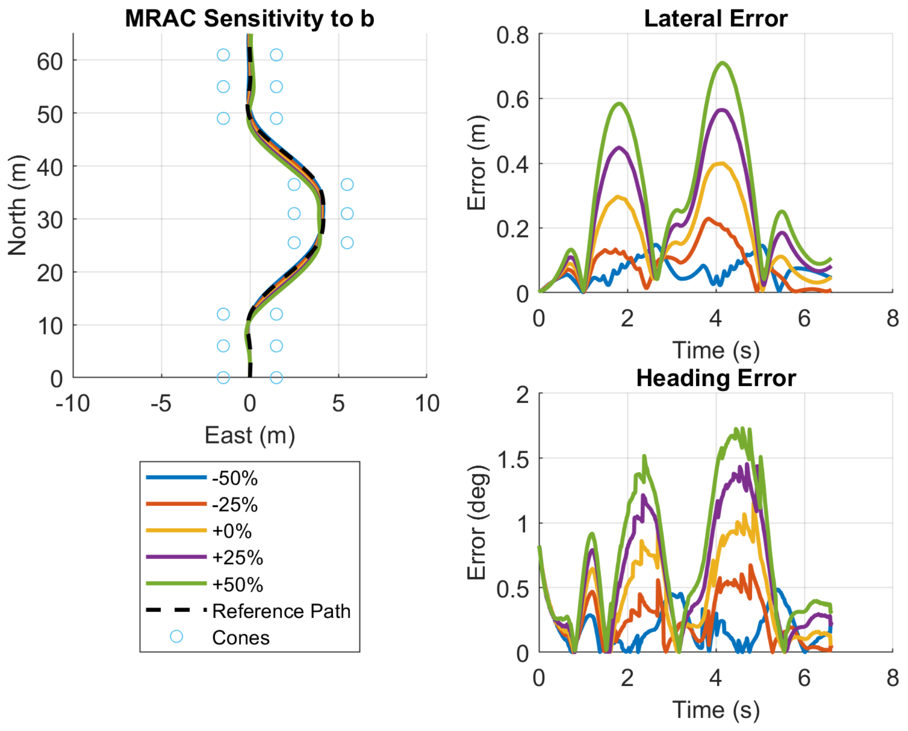


Figure A.28: MRAC Sensitivity to Distance from CG to Rear Axle.

Table A.29: MRAC Sensitivity to Vehicle Mass.

	Vehicle Mass				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	20.3097%	12.2888%	0%	-13.9490%	-30.9866%
Lateral Mean Error	19.4097%	14.0720%	0%	-14.8172%	-30.4124%

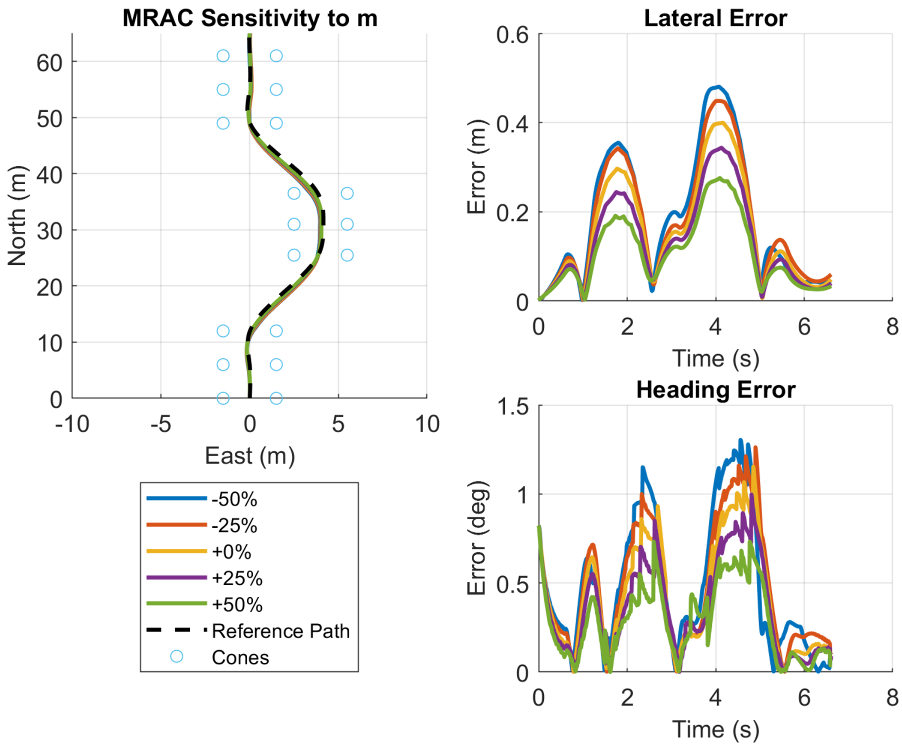


Figure A.29: MRAC Sensitivity to Vehicle Mass.

Table A.30: MRAC Sensitivity to Vehicle Inertia.

	Vehicle Inertia about the Vertical Axis				
	-50%	-25%	±0%	+25%	+50%
Lateral Max Error	-8.5953%	-2.5102%	0%	0.8928%	1.3697%
Lateral Mean Error	-6.4141%	-1.6891%	0%	1.5117%	2.6278%

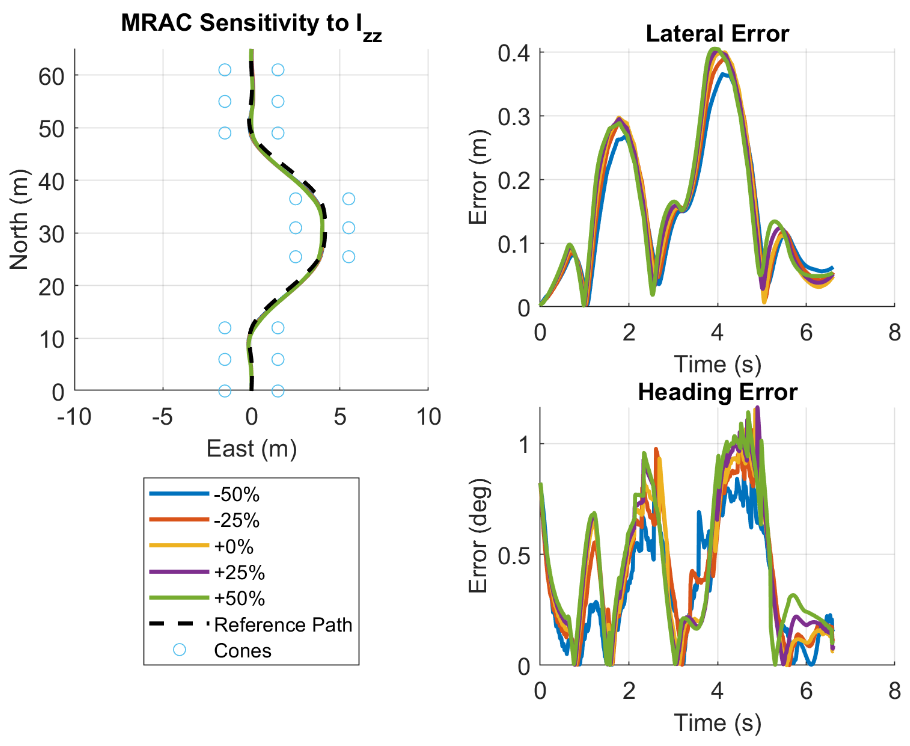


Figure A.30: MRAC Sensitivity to Vehicle Inertia.

Appendix B

Additional Simulation Results

This appendix contains additional results of the vehicle agnostic path following controller when tested in simulation at 7.5, 12.5, and 17.5 m/s. These tests were performed on Gazebo models of a Lincoln MKZ, Ford F150, Chrysler Pacifica, and Jeep Grand Cherokee following the testing procedure described in Section 5.1.

Figure B.1 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 7.5 m/s. Again, the effective wheelbase for each vehicle reached the truth value within a few turns. As the speed was increased, the effective wheelbase reached the steady state value quicker due to more excitation earlier in the run. This trend continued throughout the rest of the speeds. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1.3 meters for all vehicles. Qualitatively, the vehicles all stayed inside the lane lines for the entire runs. Figure B.2 shows that the adaptive yaw controller was able to follow the commanded yaw rate very closely for the duration of the runs, especially once the effective wheelbase converged.

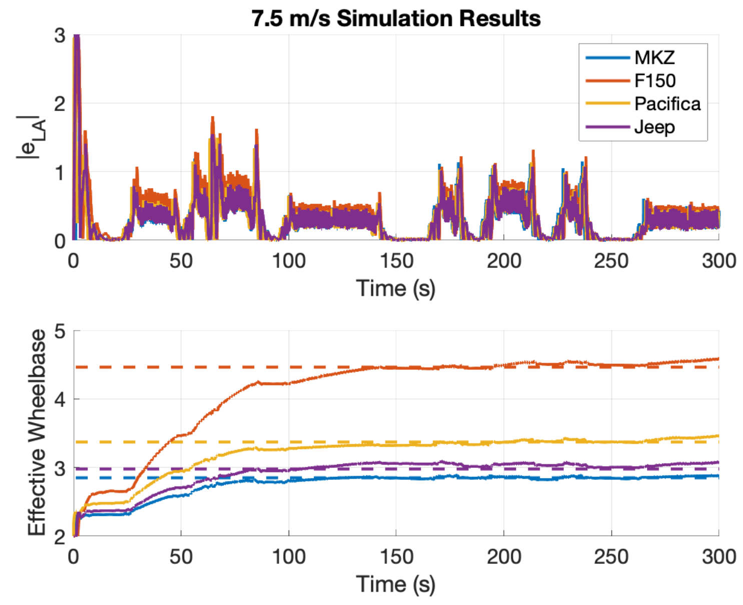


Figure B.1: Simulation Lookahead Error at 7.5 m/s.

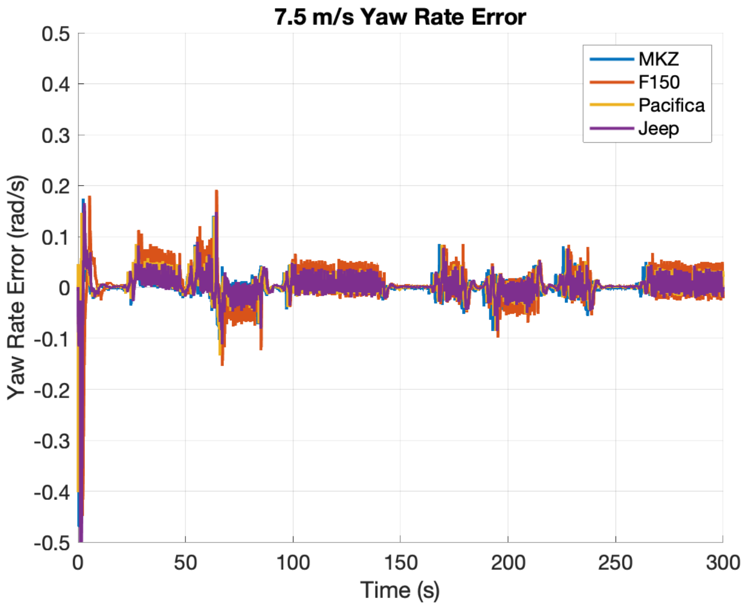


Figure B.2: Simulation Yaw Rate Error at 7.5 m/s.

Figure B.3 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 12.5 m/s. The bottom subplot shows that the effective wheelbase for each vehicle reached the truth value within a few turns. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1.7 meters for all vehicles. Qualitatively, the vehicles all stayed inside the lane lines during the entire runs. Figure B.4

shows that the adaptive yaw controller was able to follow the commanded yaw rate closely for the duration of the runs.

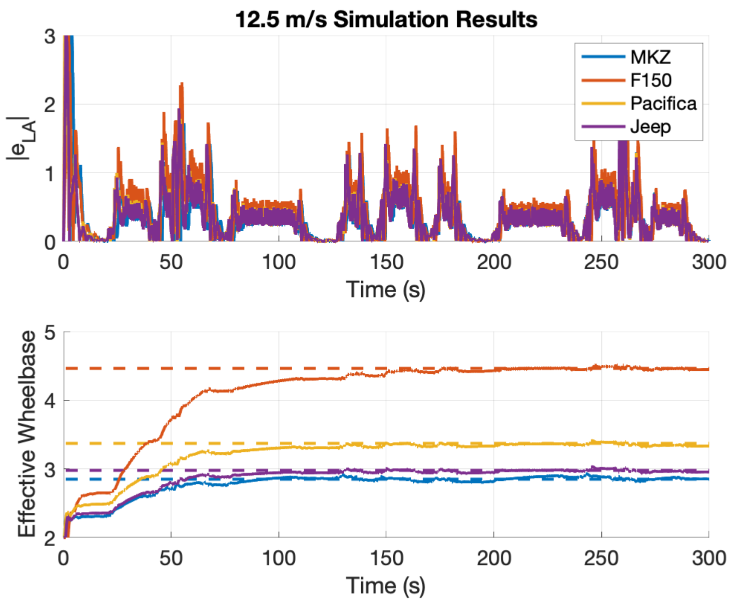


Figure B.3: Simulation Lookahead Error at 12.5 m/s.

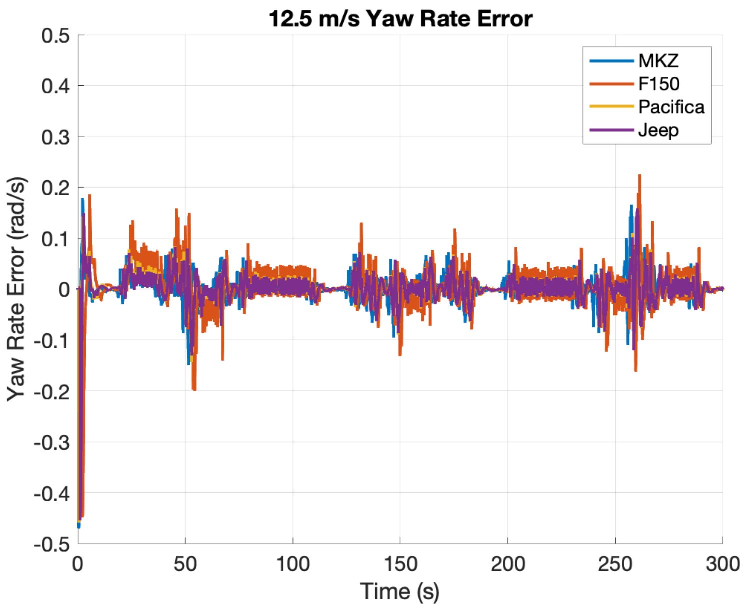


Figure B.4: Simulation Yaw Rate Error at 12.5 m/s.

Figure B.5 below displays the lookahead errors and effective wheelbase lengths during the runs with a longitudinal velocity of 17.5 m/s. The bottom subplot shows that the effective wheelbase for each vehicle reached the truth value within a few turns. Once the effective wheelbases reached steady state, the lookahead errors stayed below 1.8 meters for all vehicles. Qualitatively, the vehicles all stayed inside the lane lines, even before the effective wheelbases converged. Figure B.6 shows that the adaptive yaw controller was able to follow the commanded yaw rate closely for the duration of the runs.

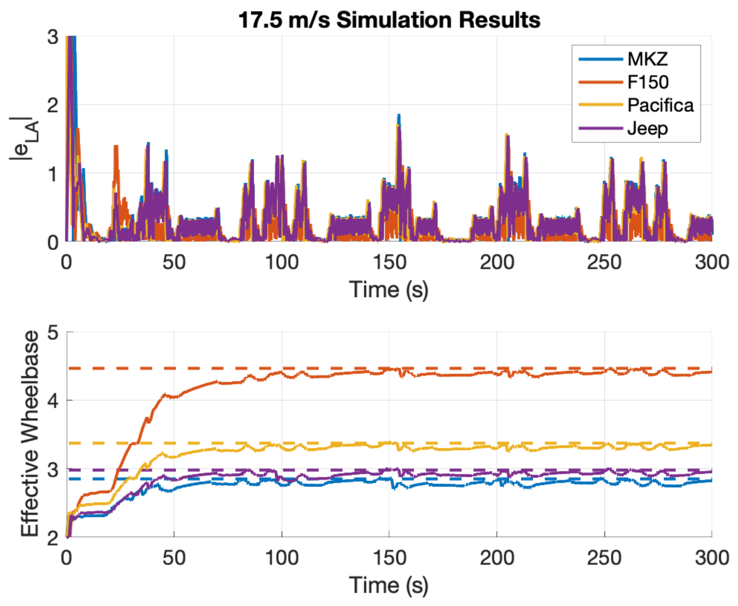


Figure B.5: Simulation Lookahead Error at 17.5 m/s.

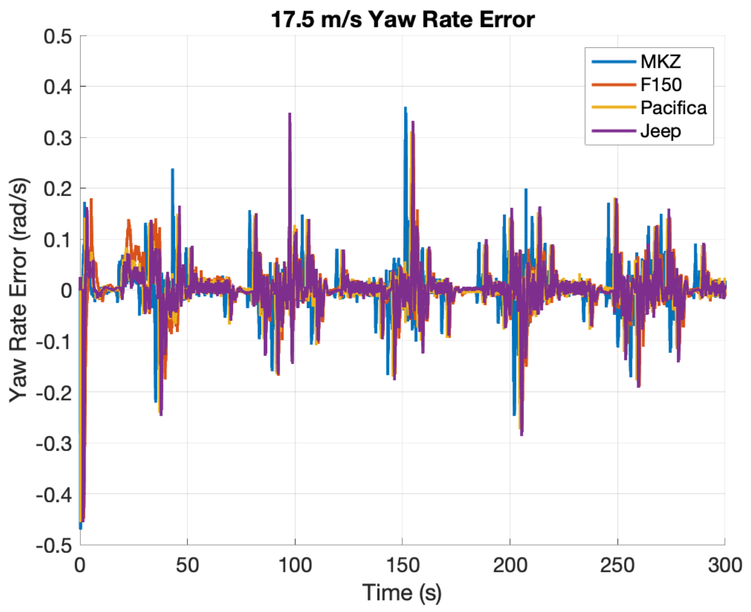


Figure B.6: Simulation Yaw Rate Error at 17.5 m/s.