Adaptive Steering Actuator Delay Compensation for a Vehicle Lateral Control System

by

William Kennedy

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
May 6, 2023

Keywords: Vehicle Controls, Time Delay Compensation, Steering Control

Approved by

David Bevly, Chair, Bill and Lana McNair Distinguished Professor
Scott Martin, Assistant Professor of Mechanical Engineering
Brendon Allen, Assistant Professor of Mechanical Engineering

Abstract

This thesis presents an adaptive control algorithm for steering actuator delay compensation in a ground vehicle lateral control system. Unknown and time-varying actuator delay values are considered. Many active safety systems and all autonomous vehicles rely on a lateral control system in order to follow a desired path. Lateral control systems designed without considering actuator delay may not achieve desired path following performance or may exhibit an undesirable system response such as steering oscillation, resulting in an uncomfortable ride or even instability or collisions. The delay compensating controller presented in this thesis is implemented at a low level within the lateral control system. It attempts to mitigate the effects of the time delay without using a vehicle model or altering the high level path following controller. The estimation algorithm used in this thesis utilizes knowledge of the control input and measured steer angle to estimate both the communication delay and the actuator dynamics present in the control system. This allows the compensating controller to adapt to changing or unknown actuator delay. The performance of the inner-loop compensation algorithm is evaluated with multiple path following controllers. The controller is then tested in a lane-keeping simulation, and finally applied to a real-world path following experiment. The proposed compensation algorithm is shown in both simulation and real-time tests to improve steering response and overall path following performance.

Acknowledgments

I would first like to thank my parents, my sister, and the rest of my family for all of their love and support over the years. This would not have been possible without them. Thank you to my girlfriend, Maci, for being there for me and for her constant encouragement.

I would next like to thank my advisor, Dr. Bevly, for encouraging me to pursue a Master's degree and providing a great lab to work in. I also want to acknowledge Dr. Martin and Dr. Chen for their help throughout my time in the lab. Thank you as well to Dr. Allen for agreeing to be a member of my committee.

Finally, I would like to thank my friends and colleagues in the GAVLab. A special thanks to Sam McDougal, Gabe, and Ben for always being up for a round of golf and to Elizabeth and Kathleen for providing the daily crossword puzzle. Thanks as well to the rest of the Wednesday night trivia team.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Automobiles are by far the most common form of day-to-day transportation used in America, where over 70% of commuters either drove alone or carpooled to work in 2021 [1]. In 2020 there were 276 million registered vehicles in the U.S., compared to only 228 million licensed drivers [2]. Driving does pose a safety risk, however, with motor vehicle crashes accounting for over 38,000 deaths in 2020 alone [3]. A study published by the National Highway Traffic Safety Administration (NHSTA) in 2018 attributes the "critical reason" for 94% of car crashes to the driver [4].

Recent advances in vehicle autonomy have led to many valuable safety systems, available in most new cars today. These systems attempt to aid in avoiding collisions and performing normal driving tasks when the driver makes mistakes or becomes distracted. These include adaptive cruise control, lane keeping assist, electronic stability control and automatic emergency braking, among others. These Advanced Driver-Assistance Systems (ADAS) attempt to prevent crashes from occurring. Many of these features are classified as low levels of autonomy according to the Society of Automotive Engineers (SAE). The six levels of vehicle autonomy defined by SAE are shown in Figure 1.1. Most new vehicles are capable of level 2 autonomy, and companies have invested hundreds of millions of dollars towards reaching higher levels of autonomy.

Safe and effective path following control is critical to vehicle autonomy for any system classified level 2 or higher. The vehicle must be able to adhere to a reference path with minimal

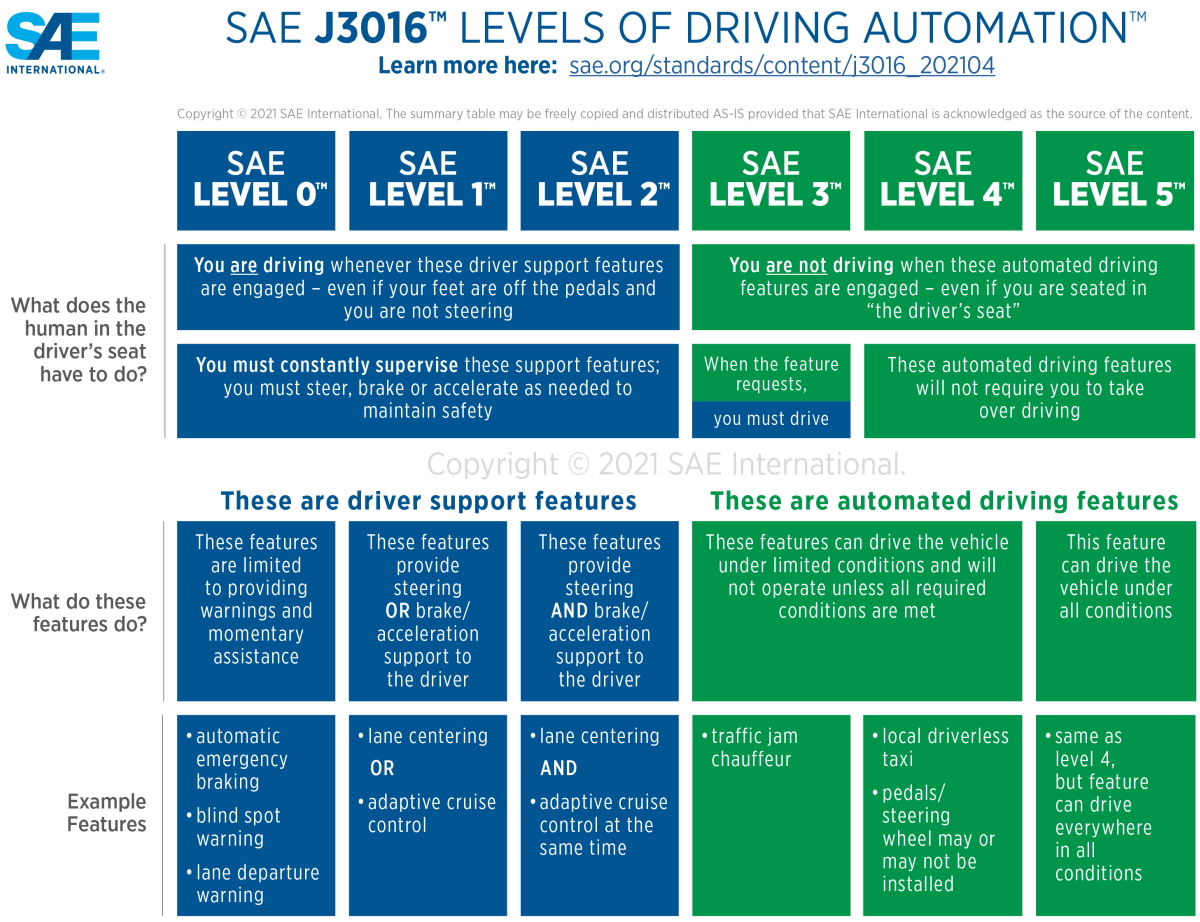


Figure 1.1: SAE Levels of Vehicle Automation [5]

error in order to maintain lane position and avoid collisions. Path tracking controllers, also commonly referred to as steering or lateral controllers, calculate the necessary steering input to follow the desired path. In every vehicle there will exist a communication delay between the path tracking (outer-loop) controller and the steering actuator. Additionally, every steering actuator will have inherent dynamics, and will not instantaneously produce the desired steer angle.

Communication delay and steering actuator dynamics can degrade path following controller performance, often causing oscillation, reduced path tracking bandwidth, and in severe cases, instability. Therefore, in order to meet performance specifications, actuator delays must be accounted for in the control design. For smaller delays, it may be possible to achieve desired performance by simply tuning the existing controller. Gain values can be scaled down to maintain stability margins in spite of delays. For larger delay values, however, this strategy may not

be sufficient, as reducing gain values leads to slower performance, less accuracy and reduced bandwidth. In this case, a more active compensation strategy is necessary.

Vehicles that have been retrofitted with drive-by-wire (DBW) hardware are especially susceptible to communication delays. The desired steer angle must pass through multiple levels of hardware and software to get from the controller to the steering system. outer-loop delay compensation methods have been shown to be effective at mitigating the effects of steering delay, however many of these methods make assumptions that may not be true. One of the main drawbacks to many outer-loop delay compensation strategies is the algorithms' reliance on an accurate vehicle model to predict future states. An accurate model is not always available. Additionally, vehicle parameters may change over the course of operation. For example, millitary and commercial trucks operate under a wide range of loading conditions, leading to significant changes in vehicle parameters. Even passenger vehicles can be susceptible to model inaccuracies due to changing road conditions, tire wear, loading, etc. Additional drawbacks of outer-loop compensation methods can include increased complexity, increased computation time, and the necessity of an outer-loop controller redesign.

For many systems, an outer-loop compensation method may not be the best option. An alternative approach is to consider an inner-loop compensation method. The benefits of compensating for delay at a lower level are:

- Compensation can be achieved without the use of a vehicle model, meaning an accurate model is not necessary.
- The negative effects of delay can be mitigated without a complete redesign of the outer-loop controller, giving the designer more freedom in outer-loop control design.
- Inner-loop methods are often less complex and less computationally expensive than many outer-loop methods.

Another drawback of many delay compensation methods is the assumption of known, constant communication delay and actuator dynamics, which may not be known accurately.

Additionally, steering dynamics can change based on tire wear and road conditions, and communication delay can vary due to a number of factors [6]. Assuming known, constant delay and dynamics can lead to prediction errors and degraded controller performance.

This thesis presents a low-level adaptive delay compensating controller designed to be inserted between the path following controller and the steering system. The goal will be to target the delay at the source, providing compensation without changing the outer-loop controller. The inner-loop controller estimates and compensates for the inherent delays between the computer and steering system as well as the steering actuator dynamics. This allows for the control system to maintain stability, reduce steering oscillations, and maintain near-desired path tracking bandwidth, without the use of a vehicle model.

1.2 Related Work

1.2.1 Path Following Control

Path following control for autonomous vehicles has been explored extensively [7]. Solutions range from simple kinematic controllers to complex control schemes such as model predictive control (MPC). Kinematic controllers offer simple control laws based on kinematic vehicle models and require no feedback [8,9]. Classical control techniques such as PID feedback loops have also been applied to path following [10]. Adaptive controllers have been shown to yield good performance as well [11–13]. Techniques such as MPC attempt to apply optimal control to the path following problem [14, 15]. The majority of path following control designs ignore actuator dynamics or try to account for them by tuning controller gains or look-ahead distance. This may not be sufficient for slower actuators or for higher dynamic maneuvers.

1.2.2 Time Delay Compensation and Estimation

Time delays and actuator dynamics are present in nearly all control systems, and the topic of delay compensation has been widely researched for many years. One of the earliest solutions, and still one of the most commonly used, is the Smith predictor [16], proposed in 1957 for factory and chemical process control where long time delays are common. The Smith predictor works by predicting and feeding back future states to mitigate the effects of time delay. Since

its conception, it has been adapted to a wide variety of control systems with varying delay times and dynamics. Methods exist to make the Smith predictor adaptive [17–23], to increase its robustness [24–27], and to adapt it to unstable or non-minimum phase systems [28–31]. Additional methods such as MPC [32–34], step-ahead control [35], gain scheduling [36], and autotuning [37] have been investigated to compensate for both pure delays and actuator dynamics.

The topic of time delay estimation has been well researched, and has applications to multiple fields including system identification for control design and signal processing, among others. Numerous methods have been proposed to estimate delays. These can be divided into two main categories [38]. The first includes methods that approximate the time delay in a system model where the delay itself is not an explicit parameter [39]. The second category includes methods that attempt to estimate the delay as an explicit parameter [40, 41]. Neural networks [17] and sliding mode estimation [42] have also been applied to the time delay estimation problem.

1.2.3 Delay Compensation in Autonomous Vehicle Systems

Actuator delay compensation for autonomous vehicle applications has been well researched, with a multitude of resources describing the design and implementation of delay compensation algorithms. One application where delay compensation has proven to be valuable is in longitudinal control systems such as adaptive cruise control (ACC) and cooperative adaptive cruise control (CACC) systems. The authors of [34] examine the effect of actuator and sensor delay on the string stability of ACC systems where an MPC control architecture, augmented with actuator and sensor models, is proposed to compensate for the delays. The delays are assumed to be known exactly, however. String stability of a CACC system with delays is examined in [43]. A Smith Predictor control scheme is utilized with a proportional-derivative (PD) controller to compensate for the delays. The delays are, again, considered to be known and constant. Longitudinal control of commercial heavy vehicles (CHV) with actuator delays is accomplished using a nonlinear backstepping control scheme in [44].

Delay compensation has also been applied to numerous other autonomous vehicle subsystems. A compensation method for environmental sensor delays is presented in [45], but constant delay values are again assumed. The authors of [32] present an MPC design to compensate for actuator delay in a vehicle stability control system. In [46], an adaptive delay compensating controller is applied to a vehicle active suspension system. The authors of [47] also address delays in an active suspension system, presenting a sliding mode control based solution. A networked predictive control strategy is developed to compensate for delays in an electro-hydraulic actuated wet clutch system in [48]. An MPC compensation strategy was applied to air-to-fuel ratio control in [49]. Each of these works, however, consider delay and actuator dynamics to be known and constant.

Steering delay compensation methods have been proposed to improve lateral control system performance. The authors of [50] present an MPC based lane keeping assist system with Smith Predictor based delay compensation. Both the delays inherent to Global Navigation Satellite System (GNSS) measurements and the steering actuator delay are considered. A process-model control based approach is taken to compensate for steering actuator delays and saturation in [51]. Performance improvements are shown, however the delay value is once again assumed to be known and constant.

In [52], a preview path tracking controller with steering delay compensation is designed, implemented, and compared to a predictive compensation method. Path tracking improvements are shown over the uncompensated performance for both compensation methods, however the steering delay and dynamics are assumed to be known and constant and an accurate vehicle model is assumed. The authors of [33] propose a robust, delay-aware model predictive control architecture for lateral path tracking using Robust Tube MPC. They also implement an adaptive Kalman Filter to estimate the computation time of the MPC controller. This method focuses mainly on the delay introduced by the computation time of the model predictive controller rather than the communication delay between controller and actuator. Constant, known actuator dynamics were also assumed.

1.3 Research Contributions

The majority of delay compensation methods that have been proposed for vehicle steering control employ a high-level approach, often attempting to predict future vehicle states or augmenting an MPC model with actuator dynamics. These methods have shown promising results, however there are drawbacks. Designing a predictive path tracking controller requires a relatively accurate vehicle model and often results in a very specialized controller that may be difficult to implement on different vehicles. Additionally, many proposed methods assume known, constant delay and steering dynamics. This is not always a safe assumption, as these values may be modeled incorrectly or may vary over time. This thesis presents a low level compensation scheme that adapts to changes in the steering response.

The main contribution of this work is an adaptive inner-loop steer angle controller that compensates for delay between the control computer and the steering system and for the dynamics present in the actuator. The compensation does not require a vehicle model and is able to adapt to varying or incorrectly modeled delay and steering dynamics. The algorithm will be shown both in simulation and in real-time experimental data to maintain stability, reduce steering oscillations, and improve path tracking performance in the face of steering delays. It will also be shown to adapt well to different steering actuators on different vehicles. Some of the results from this research were presented in [53].

Chapter 2

Technical Background

2.1 Notation

Throughout this thesis, scalars, matrices and vectors will be represented as follows. Scalars will be represented by lowercase, non-bolded variables such as a. Matrices will be represented by capital, non-bolded letters as shown in Equation (2.1)

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} \\ m_{2,1} & m_{2,2} \end{bmatrix} \tag{2.1}$$

where M represents a 2x2 matrix of scalars. Vectors will be represented by bolded, lower case letters or symbols such as $\mathbf{v}$. Accents over variables will be used to represent different types of values. A "hat" over a variable such as $\hat{\mathbf{x}}$ implies that it represents an estimated value of that variable. A tilde over a variable such as $\tilde{y}$ will be used to represent a measured value of that variable.

Throughout this thesis, dynamic systems will be represented using both state variable formulation and transfer functions. Basic knowledge of differential equations, control systems, frequency response, and state space and transfer function notation is assumed. An introduction to these topics can be found in the prior work [54, 55].

2.2 Time Delay

This section will discuss the modeling, approximation, and effect of time delay in control systems. Two types of delay will be discussed in this thesis. The first type will be referred to as

"pure delay." Pure delays are also commonly referred to as transport delays, non-distorting delays, or latency. Pure delay encompasses communication delay, computation time and sampling delays. Pure delays do not distort signals, they only shift the signals in time.

Time delays are inherent to all control systems and can occur for a variety of reasons. Actuators, sensors, and other hardware components all introduce varying degrees of delay. Software design can also introduce delay into the system due to computation time. All time delays have a negative effect on control system performance, leading to reduced gain margin, reduced phase margin, oscillation, and in severe cases, instability. This work will focus on delays caused by the steering actuator of a vehicle.

The second type of delay discussed will be referred to as "dynamic delay." This describes the actuator dynamics caused by the mechanical properties of the actuator itself and the low level control design within the actuation system. Both types of delay are inherent to all mechanical actuators in varying degrees, and both have a negative effect on control performance.

2.2.1 Mathematical Representation of Actuator Delay

A block diagram of an arbitrary control system with actuator delay and dynamics is shown in Figure 2.1. The pure delay and actuator dynamics are shown as two seperate components. The plant and actuator dynamics are assumed to be linear and time-invariant for this representation. The dynamics of the plant and the actuator are represented in Equation (2.2) and Equation (2.3), respectively.

$$\dot{\mathbf{x}} = A\mathbf{x} + Bu(t) \tag{2.2}$$

$$\dot{u} = -a_u u + b_u d(t - \tau) \tag{2.3}$$

A and a_u represent the state matrix of the system and the eigenvalue of the actuator dynamics, respectively. B and b_u represent the input matrix of the system and the input scale value for the actuator. The variables d and u represent the desired input and the actual input provided by the actuator, respectively. For this model, the actuator dynamics are assumed to be first order.

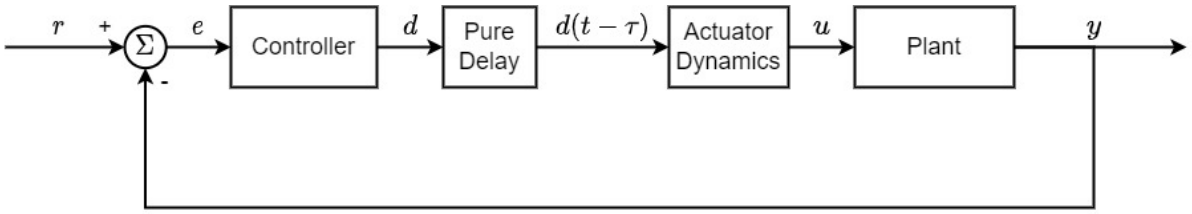


Figure 2.1: Block Diagram of Control System With Actuator Lag

The actuator dynamics are assumed to have both a pure delay portion and a dynamic portion. The pure delay value in seconds is represented by τ. In order to represent the original system and actuator dynamics in a single system, the state and input matrices can be augmented as shown in Equation (2.4).

$$A_n = \begin{bmatrix} A & B \\ 0 & a_u \end{bmatrix} \qquad B_n = \begin{bmatrix} 0 \\ b_u \end{bmatrix} \tag{2.4}$$

This augmentation assumes that the actuator dynamics are independent of the state dynamics. However the augmented matrix could easily be altered to represent a linear relationship between the two. The full augmented system is shown in Equation (2.5).

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{u} \end{bmatrix} = A_n \begin{bmatrix} \mathbf{x} \\ u \end{bmatrix} + B_n d(t-\tau) \tag{2.5}$$

The delayed desired input d has replaced the actuator output as the input to the system and the actuator dynamics are included in the augmented state matrix. Due to the pure delay present in the system, the desired input d is shifted in time by τ seconds. Linear time delay systems can also be represented using continuous or discrete transfer functions. The laplace domain and z-domain representation of a pure delay of τ are shown in Equations (2.6) and (2.7), respectively.

$$e^{-\tau s} \tag{2.6}$$

$$z^{\frac{-\tau}{T}} \tag{2.7}$$

In order to represent delay accurately in the discrete domain the delay value must be divisible by the sample rate T. If the delay is not divisible by the sample rate then it must be approximated by rounding to the closest number of samples. Therefore if the time delay is significantly smaller than the sample period of the system it is impractical to approximate the delay in the discrete domain for control design and analysis.

2.2.2 Approximation of Time Delay Systems

There are numerous methods to approximate time delay systems for control design and analysis. One of the most common is the Padé Approximation, which models pure time delay as a rational transfer function. This is necessary for some design methods such as root locus which require the system to be represented as a rational transfer function. An example of a first order Padé approximation of pure delay is shown in Equation (2.8).

$$e^{-\tau s} \approx \frac{-\frac{\tau}{2}s + 1}{\frac{\tau}{2}s + 1} \tag{2.8}$$

The approximation mimics the frequency response characteristics of the delay. Note that a zero is added in the right half plane. Higher order Padé approximations are also possible and approximate the delay more accurately than first order models. A simple example of a padé approximation of a first order time delay system is shown in Equation (2.9).

$$G(s) = e^{-0.1s}\frac{5}{s+5} \approx \frac{-s+20}{s+20} * \frac{5}{s+5} \tag{2.9}$$

The step responses of the actual system and the Padé approximation are compared in Figure (2.2). The approximation closely resembles the actual system response.

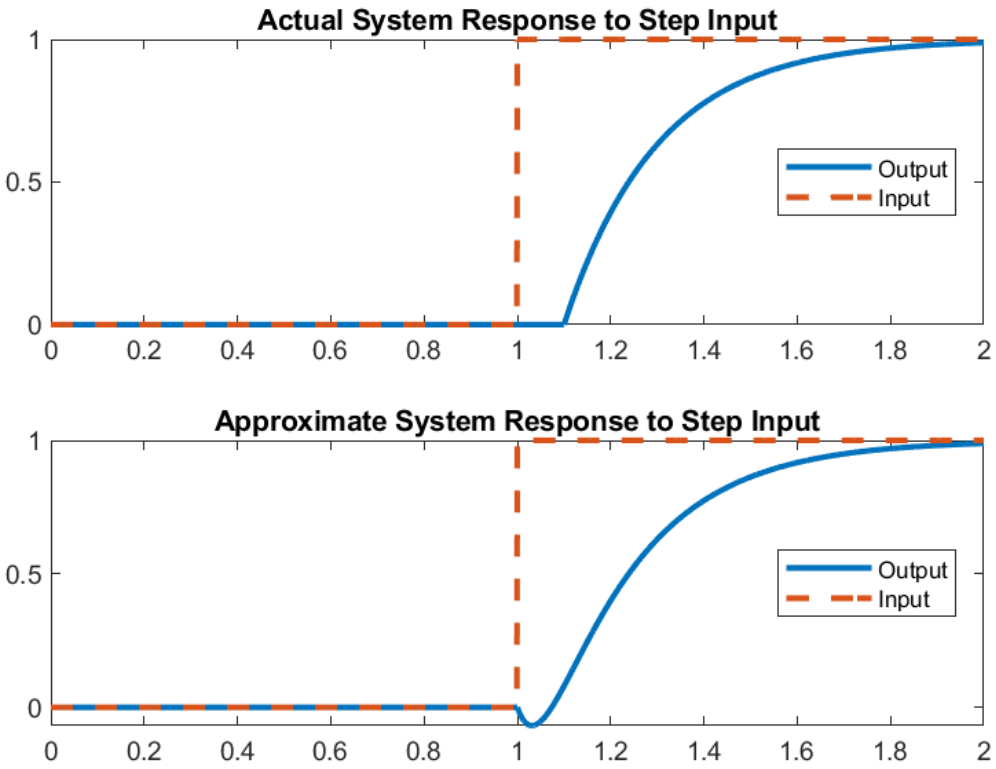


Figure 2.2: First Order Time Delay System Compared to Padé Approximation.

The step response and phase angles of the true and approximate delay responses are compared in Figure 2.3 for a first order Padé approximation of a 0.1 second pure delay. The approximation nearly matches the phase angle of the pure delay up to around 10 rad/s. The step response approximation loosely follows that of the pure delay.

Higher order Padé approximations more accurately match the true system response. A fifth order Padé approximation of a pure delay of 0.1 seconds is compared to the true response in Figure 2.4. The higher order approximation more closely mimics the step response and matches the phase response up to around 100 rad/s. Padé approximations are useful, but are not exact and can lead to instability in some systems when used for control design [56].

2.2.3 Frequency Response Effect of Time Delay

Both pure delay and actuator dynamics have an effect on the frequency response of the system. Pure delay does not affect the gain of the open loop system as it only shifts the input in time and therefore it does not have an effect on the magnitude of the signal. It does, however, have

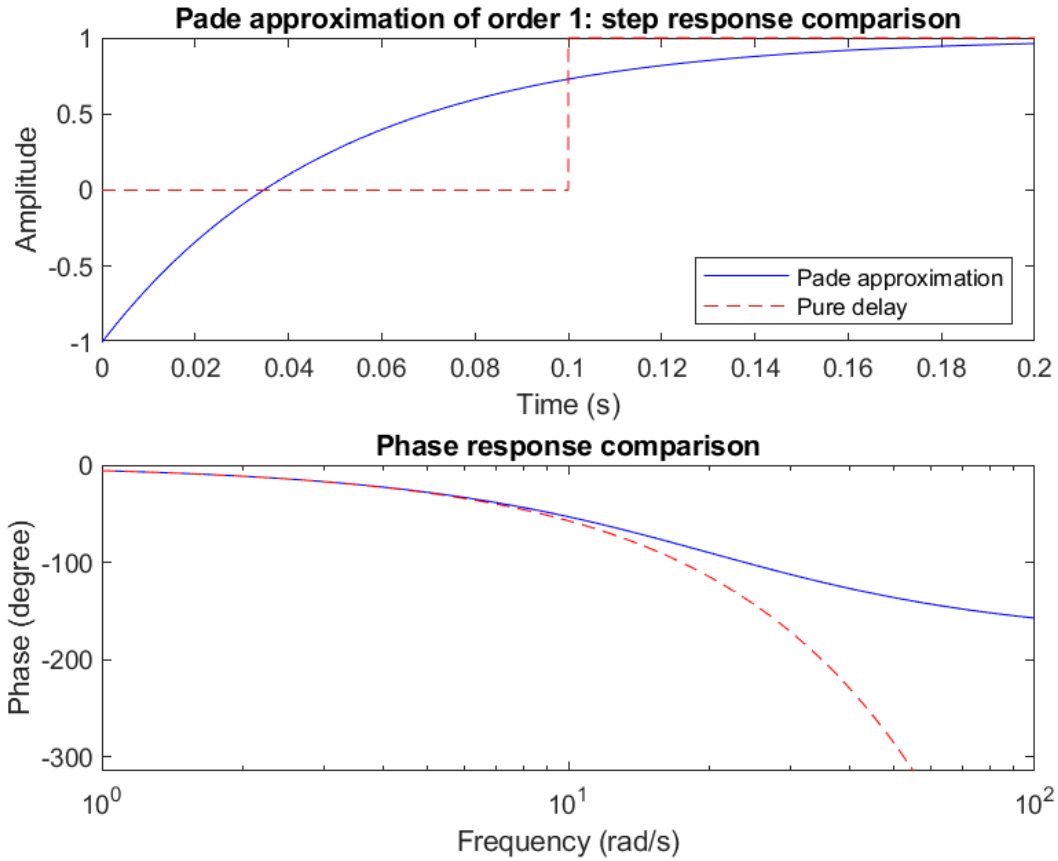


Figure 2.3: First Order Padé Approximation

an effect on the phase angle of the system. The phase of a pure time delay of τ seconds is given in Equation (2.10).

$$\angle\tau(j\omega) = -\omega\tau \tag{2.10}$$

The relationship between time delay, frequency, and phase angle is shown in Figure 2.5. The dramatic drop in phase angle for higher frequency and time delay values greatly limits the achievable closed loop bandwidth of the control system. This highlights the need for compensation if the delay is large enough or if a high closed loop bandwidth is desired.

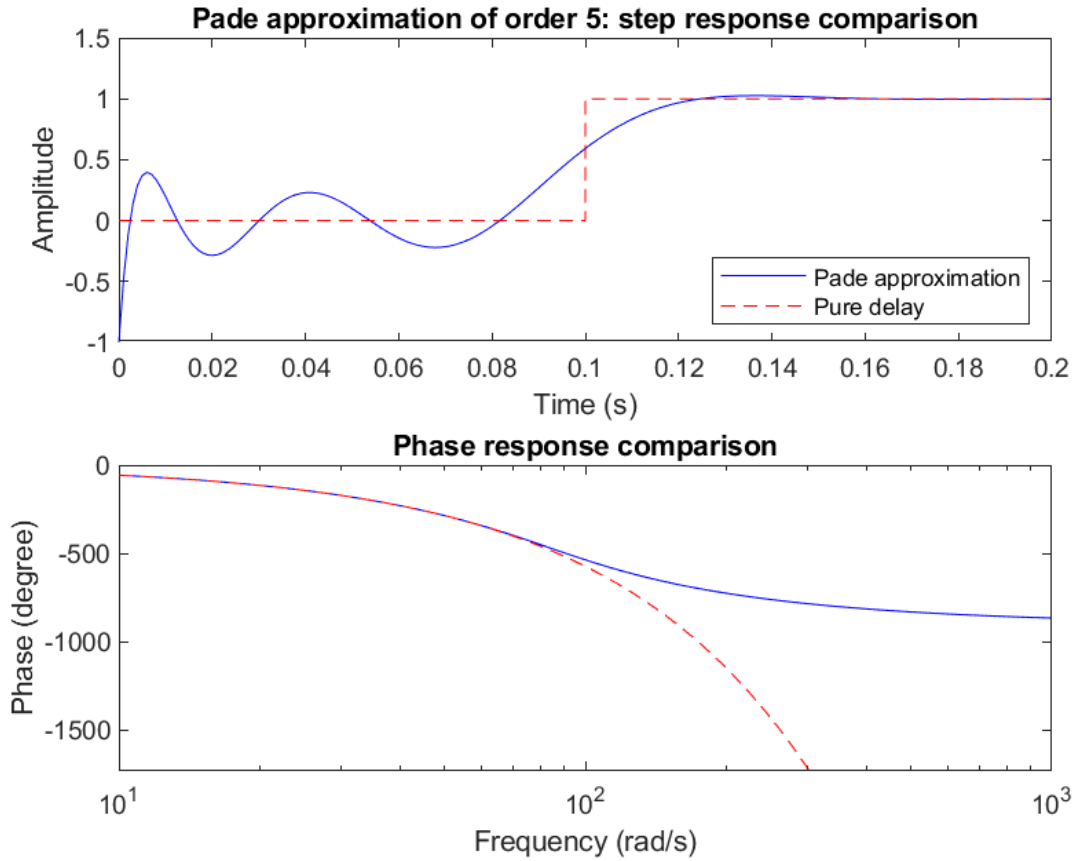


Figure 2.4: Fifth Order Padé Approximation

2.3 System Analysis and Control Design in the Presence of Time Delay

The presence of time delay in a system complicates control design and stability analysis and is detrimental to system performance. If delay is not considered during the design process, the controller will not achieve the desired response and may cause excess oscillation or instability. Various control design methods and their application to time delay systems will be discussed in the following sections.

2.3.1 Root Locus Analysis for Continuous Time Delay Systems

Use of the root locus design method is difficult for continuous time delay systems because of the need to approximate the delay in the form of a rational transfer function. This can be done using a Padé approximation as described previously in Section 2.2.3. Once the delay is approximated, the system can be analysed using root locus, however the approximation will

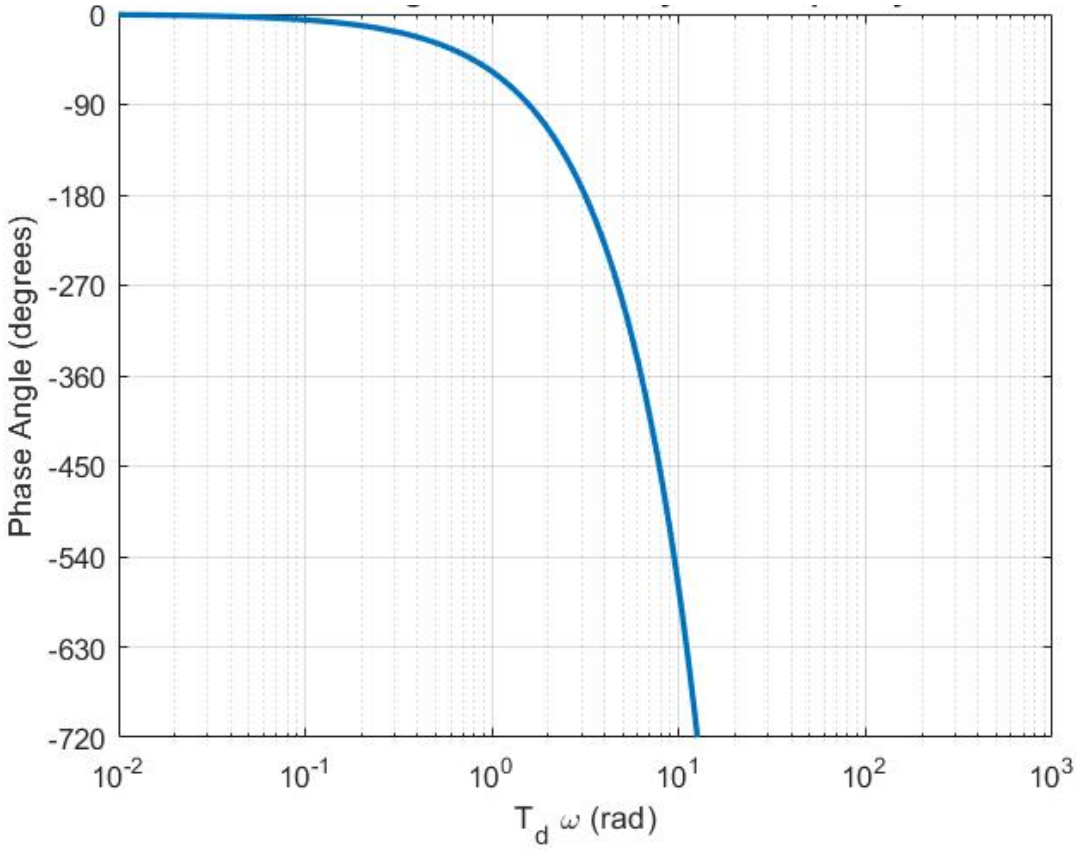


Figure 2.5: Phase Angle Compared to Time Delay and Frequency

not be exact and could lead to errors in the control design. An example of root locus design for a first order time delay system is described below. The example system and its approximation are shown in Equation (2.11).

$$G(s) = e^{-0.1s}\frac{5}{s+5} \approx \frac{-s+20}{s+20} * \frac{5}{s+5} \tag{2.11}$$

The root locus plot of the system without delay is compared to the plot of the approximate system in Figure 2.6. A simple proportional controller is assumed. The importance of considering delay in control design is apparent. When delay is not considered, the system appears to be stable for all gain values. Modifying the model using the Padé approximation to approximate the delay shows that this is not true. The right half plane zero included in the approximation results in the system being marginally stable at a proportional gain value of around 5.

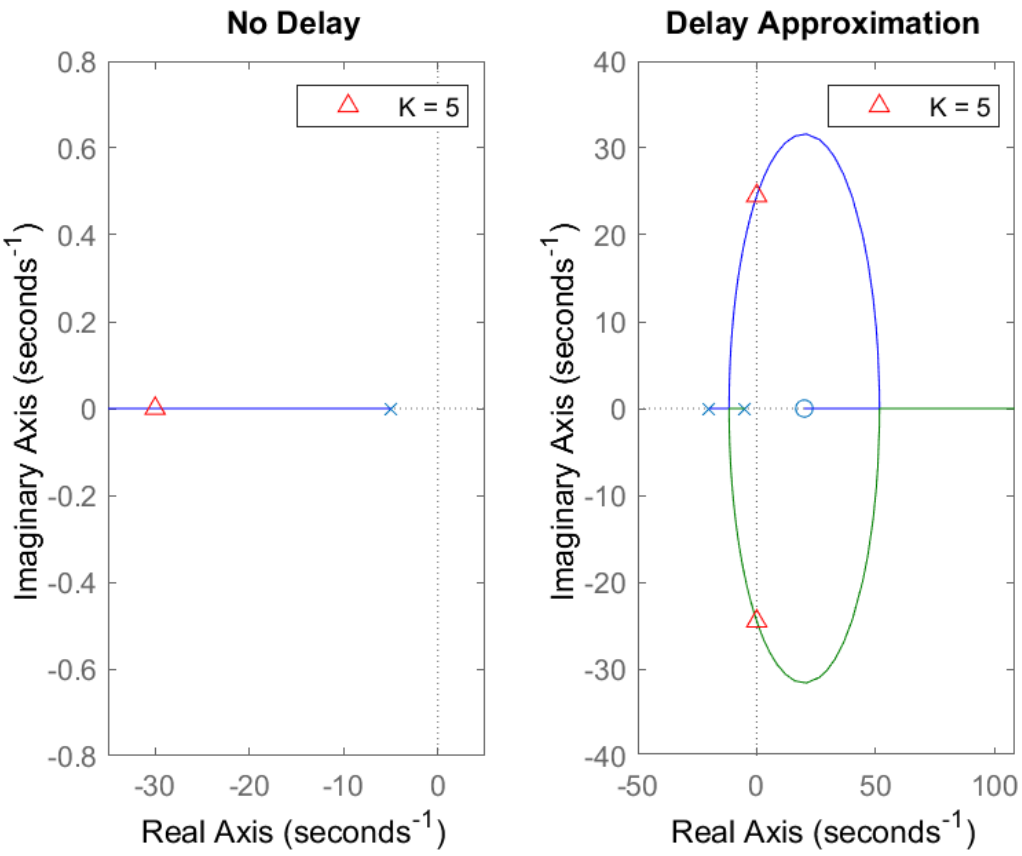


Figure 2.6: Comparison of Root Loci Between Delay Approximation and Non-Delayed System

2.3.2 Root Locus Analysis for Discrete Time Delay Systems

When represented as a discrete system, time delay can be included directly into root loci in the form of poles at the origin. This is still an approximation, as the delay value must be rounded to the nearest whole number of samples in order to be expressed in the discrete domain. An example of discrete root locus design is shown below using the same first order system as shown in the previous section, discretized at 100 Hz. The discrete system model with a delay of 10 samples is shown in Equation (2.12).

$$G(z) = z^{-10}\frac{0.04877}{z - 0.9512} \tag{2.12}$$

The root loci of the system with and without delay with a proportional controller are compared in Figure 2.7. When delay is not considered, the system appears to be stable even at

high gain values. Once the delay is included into the system, however, the root locus shows the system to be marginally stable at a gain value of around 3.5.

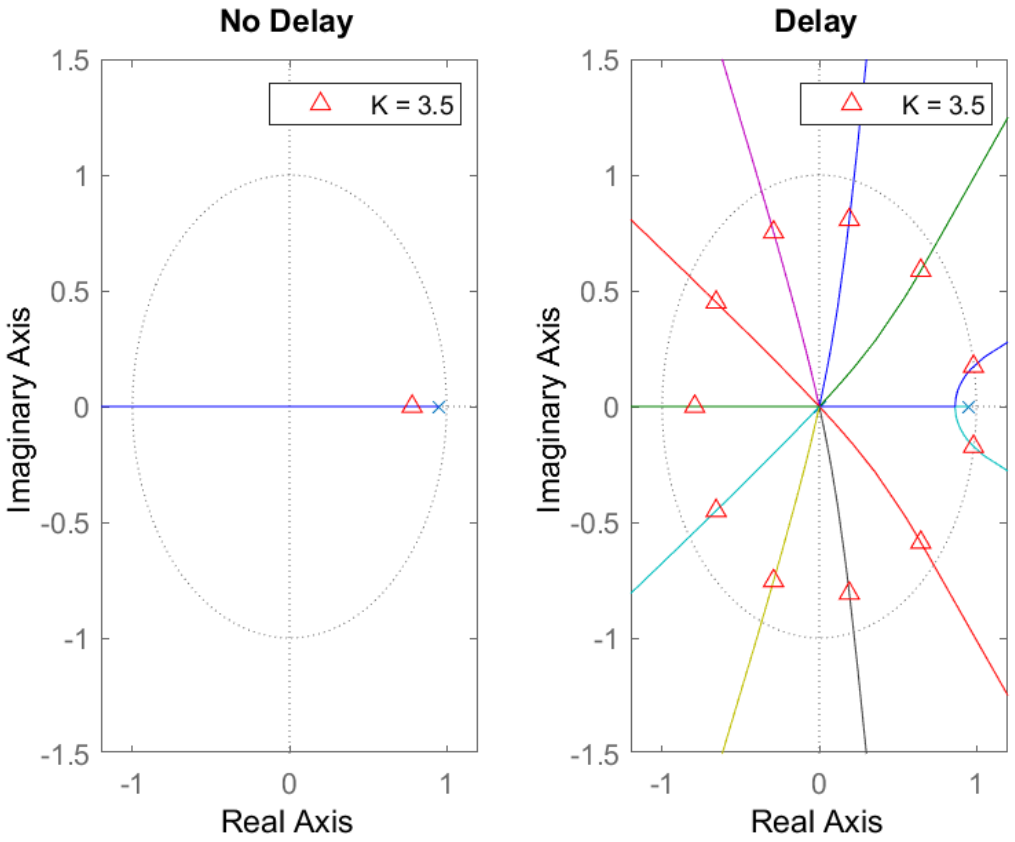


Figure 2.7: Comparison of Discrete Root Loci Between Delayed and Non-Delayed System.

2.3.3 Bode-Based Analysis for Time Delay Systems

Frequency response design methods are often most effective for considering time delay systems because the effect of the pure delay can be expressed directly in the frequency domain [54]. An example of Bode based control design is described below using the first order system previously shown in Equation (2.12).

Assuming a simple proportional controller, the control design can be performed by examining the open loop Bode plot of the system shown in Figure 2.8. The open loop Bode plots of the system with and without delay are shown. The delay has a large effect on the open loop phase of the system but does not affect the gain of the system.

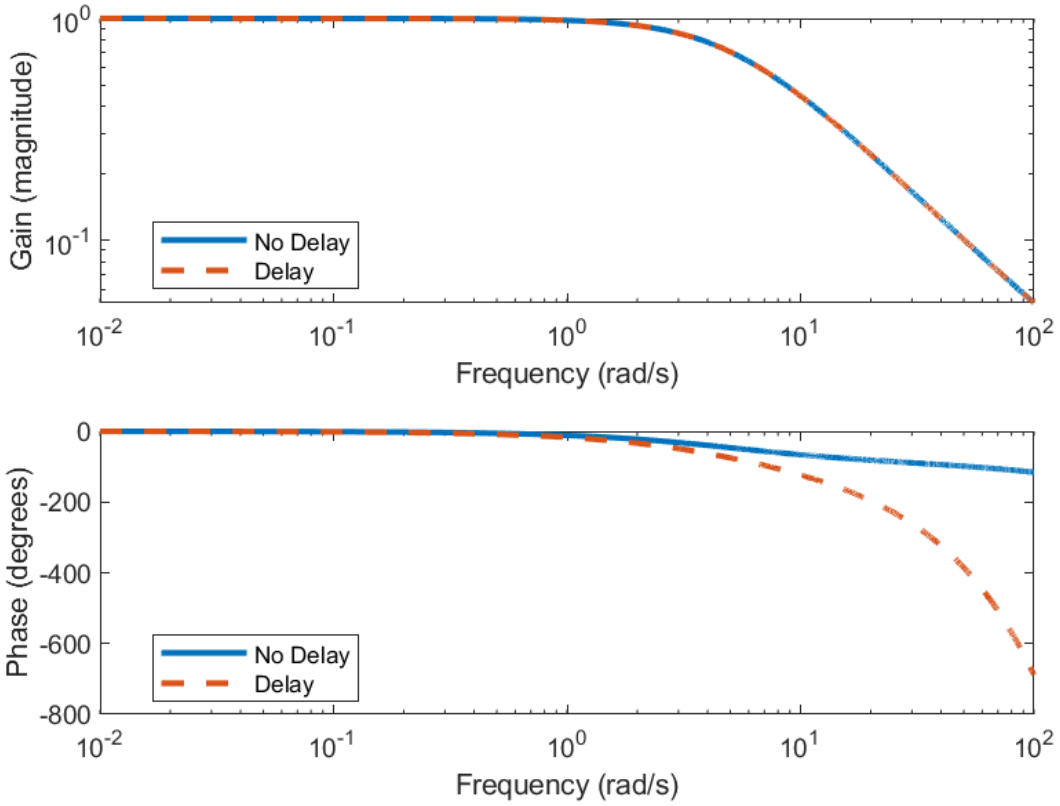


Figure 2.8: Comparison of Open Loop Bode Plots Between Delayed and Non-Delayed System

Stability for linear systems can be observed through the gain and phase margin of the open loop Bode plot. Gain margin is defined as the factor by which the system gain can be increased while maintaining stability. A gain margin greater than 1 indicates stability, while a margin of 1 indicates marginal stability and less than 1 indicates instability. Phase margin is defined as the difference between the open loop system's phase lag and -180 degrees at the crossover frequency, which is defined as the frequency where the open loop gain crosses 0 dB (magnitude of 1). A positive phase margin indicates stability while a negative or zero phase margin indicates instability or marginal stability, respectively. The crossover frequency of the open loop system approximately equals the bandwidth of the closed loop system.

Phase margin provides a metric describing the robustness of the control system to unmodeled dynamics or to errors in the plant model. For example, a system with a phase margin of 90 degrees or greater could maintain stability even with an unmodeled integrator (pole at 0). As phase margin decreases, any unmodeled dynamics must have a higher bandwidth. For example, a system with a phase margin of 30 degrees could maintain stability with an unmodeled

pole at 2 times the open loop crossover frequency. As phase margin approaches zero, the system response becomes more and more oscillatory. Gain margin describes how much the loop gain can be increased before reaching instability, or how aggressive the controller gains can be made.

The effect shown in the previous section can be seen again here. The maximum gain value and achievable closed loop bandwidth for the system with delay is much lower than that of the undelayed system. Because of the rapid drop in phase angle caused by the pure delay, closed loop bandwidths around 10 rad/s or greater may not be achievable. The gain and phase margins are plotted versus crossover frequency for a simple proportional controller in Figure 2.9. The gain and phase margins are shown for the system described in Equation (**??**), with and without delay. The point where the system reaches marginal stability is shown for the delayed system. It can be seen that the delay greatly limits the achievable crossover frequency compared to the undelayed system.

2.4 Coordinate Frames

2.4.1 Vehicle Frame

The vehicle frame used in this thesis is fixed to the body of the vehicle, with the origin at the center of gravity (CG). The X-axis extends through the front of the vehicle, the Y-axis extends from the passenger side, and the Z-axis points downward. This coordinate frame is pictured in Figure 2.10 and is used to define the vehicle model in Section 2.5. Vehicle longitudinal and lateral velocity are defined as velocity in the X and Y axes of the vehicle frame, respectively.

2.4.2 Local Navigation Frame

The local navigation frame used in this thesis is the North-East-Down (NED) frame. Most of the results presented will be in this coordinate frame. The NED frame defines a plane tangential to the surface of the earth, with the axes aligned with the North and East directions. Heading is defined as the angle between the X-axis of the vehicle frame and the North axis of the NED frame, with a positive heading representing a clockwise rotation off of the North-axis. An illustration of the NED frame is shown in Figure 2.11.

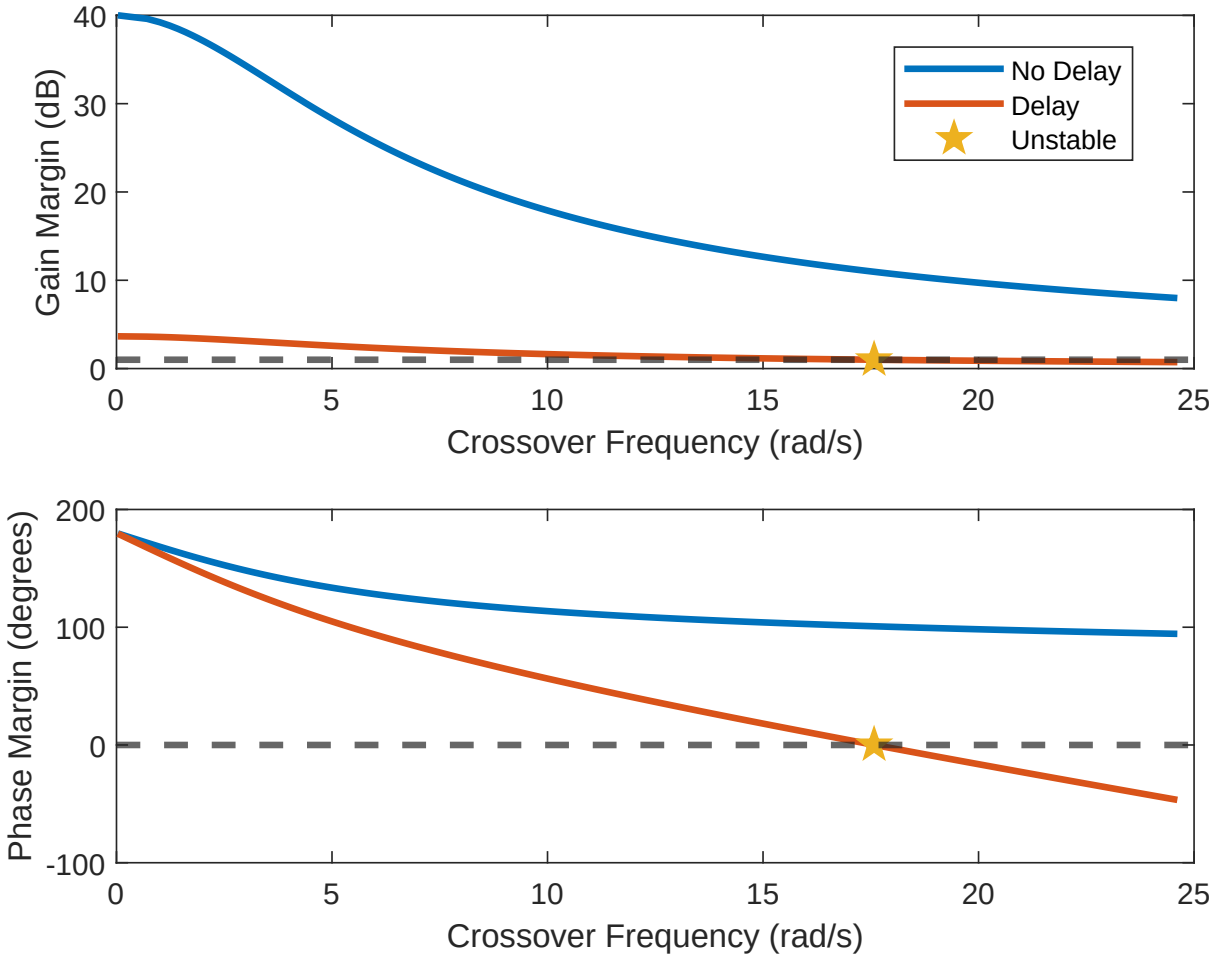


Figure 2.9: Comparison of Gain and Phase Margins Versus Controller Gain Between Delayed and Non-Delayed System

2.4.3 Global Navigation Frame

When vehicles travel long distances, a local frame such as NED is no longer sufficient due to the curvature of the earth. In these scenarios and for systems using GPS measurements, the Earth-Centered, Earth-Fixed (ECEF) coordinate frame is utilized. This frame, defined in WGS84, has its origin at the Earth's CG, its X-axis extending through the intersection of the Equator and Prime Meridian, its Z-axis extending through the North Pole, and the Y-axis completing the right-handed coordinate frame, also along the Equator. Transformations between the local NED frame and the ECEF frame can be made with knowledge of the origin of the NED frame in ECEF coordinates and the vehicle heading in the NED frame. The ECEF frame is illustrated in Figure 2.12.

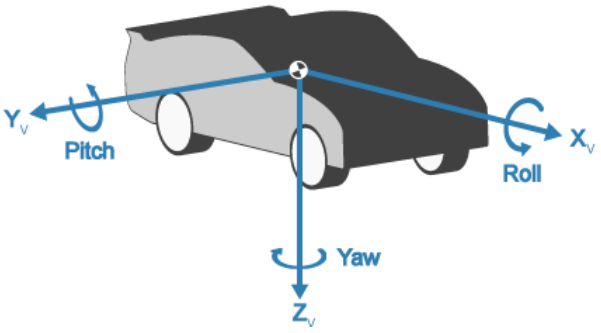


Figure 2.10: Illustration of vehicle frame. [57]

2.5 Vehicle Model

The linear dynamic bicycle model is used to model the lateral dynamics of the vehicle for the purpose of control design and stability analysis. The model is derived by summing the forces and moments about the z axis of the vehicle, shown in Figure 2.13. This model was chosen because it is valid under most driving conditions and is well suited to linear control design and stability analysis.

The dynamic bycicle model in continuous state space format is shown in Equation (2.13)

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \end{bmatrix} = \begin{bmatrix} -\frac{C_2}{I_{zz}V_x} & -\frac{C_1}{I_{zz}V_x} \\ -\frac{C_1}{mV_x} - V_x & -\frac{C_0}{mV_x} \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \end{bmatrix} + \begin{bmatrix} \frac{aC_{\alpha f}}{I_{zz}} \\ \frac{C_{\alpha f}}{m} \end{bmatrix} \delta \tag{2.13}$$

where the terms C_0, C_1, and C_2 are combinations of the distance from the vehicle's center of gravity to the front and rear axles, a and b, and the front and rear tire cornering stiffness, $C_{\alpha,f}$ and $C_{\alpha,r}$, respectively, as shown in Equations (4.30) - (4.32).

$$C_0 = C_{\alpha f} + C_{\alpha r} \tag{2.14}$$

$$C_1 = aC_{\alpha f} + bC_{\alpha r} \tag{2.15}$$

$$C_2 = a^2C_{\alpha f} + b^2C_{\alpha r} \tag{2.16}$$

For the purpose of control design, this model can be discretized at the desired sample rate. It can also be expressed in the form of a transfer function describing the dynamics from steer angle to heading, lateral position, etc. depending on the type of controller being designed.

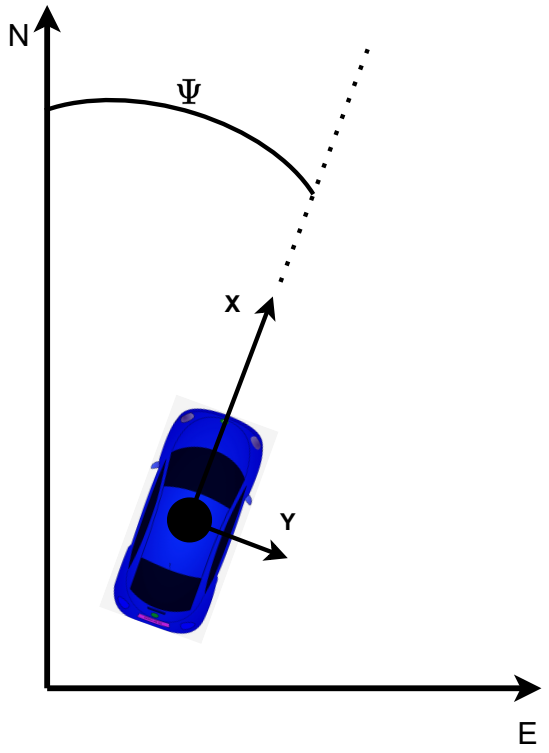


Figure 2.11: Illustration of NED Coordinate Frame. [58]

The dynamic bicycle model asumes steady state longitudinal velocity (V_x) and a linear tire model. These assumptions are valid under most driving conditions. The steering dynamics present in the vehicle are assumed to be described by a first order model with a time delay. The nonlinearities present in the steering system are ignored. This assumption holds as long as the steering system response is reasonably close to that of a first order linear system. For a more detailed model of an electric power steering system, see [6, 59].

The first order model used to describe the steering dynamics is shown in Equation (2.17).

$$\dot{\delta} + \frac{1}{T_c}\delta = d(t - \tau) \tag{2.17}$$

T_c represents the time constant of the steering dynamics. The terms d and τ represent the commanded steer angle and the pure time delay, respectively. This model can also be represented in continuous transfer function format as shown in Equation (2.18).

$$\frac{\delta(s)}{d(s)} = e^{-\tau s}\frac{b}{s + a} \tag{2.18}$$

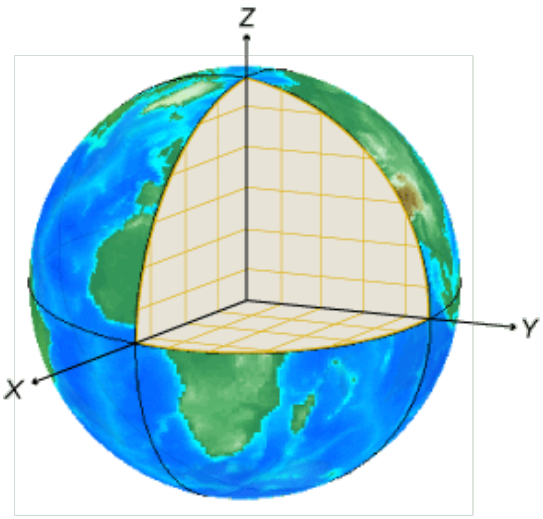


Figure 2.12: Illustration of ECEF Coordinate Frame. [58]

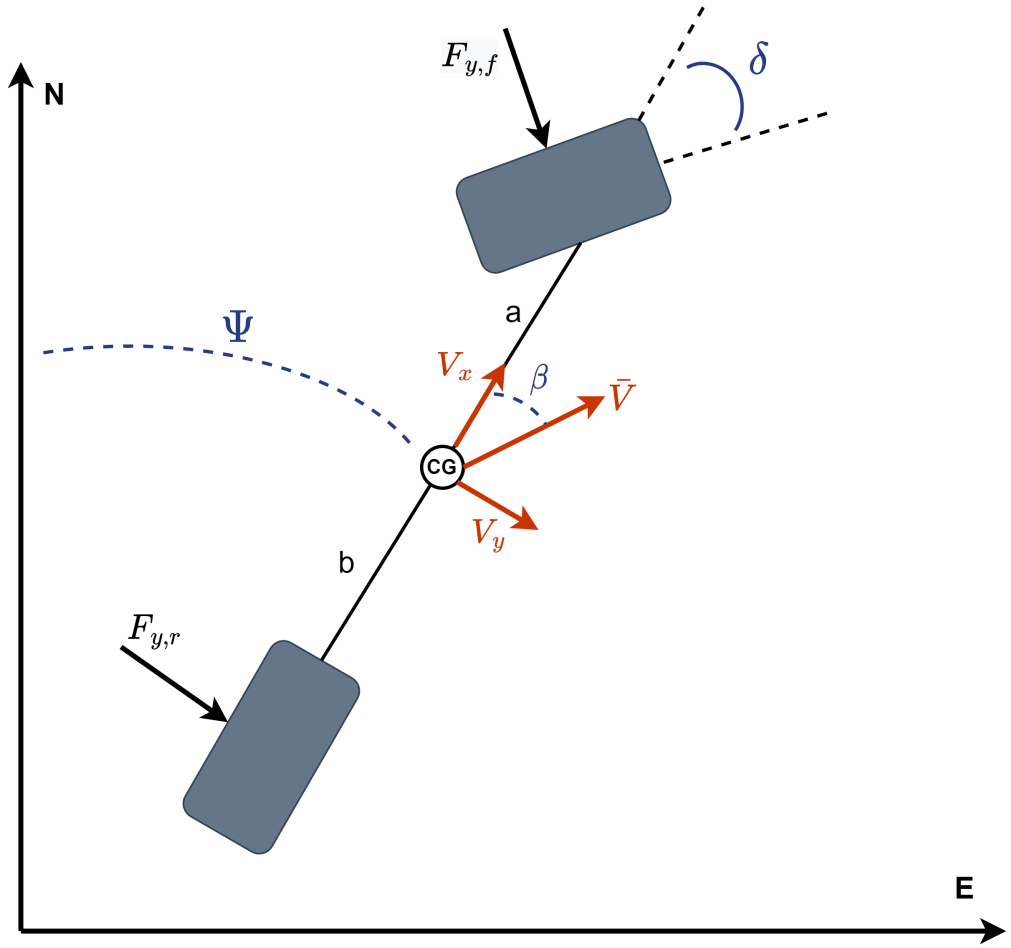


Figure 2.13: Dynamic Bicycle Model.

The model is shown in discrete transfer function format in Equation (2.19).

$$\frac{\delta(z)}{d(z)} = z^{\frac{-\tau}{T}} \frac{b}{z-a} \tag{2.19}$$

T represents the sampling period of the system. In order to represent the model in the discrete domain the pure delay τ is assumed to be some multiple of the sampling period. The dynamic bicycle model can be augmented with the steering dynamics in order to include the full system dynamics in one model as shown in Equation (2.20).

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} -\frac{C_2}{I_{zz}V_x} & -\frac{C_1}{I_{zz}V_x} & \frac{aC_{\alpha f}}{I_{zz}} \\ -\frac{C_1}{mV_x} - V_x & -\frac{C_0}{mV_x} & \frac{C_{\alpha f}}{m} \\ 0 & 0 & -\frac{1}{T_c} \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \\ \delta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} d \tag{2.20}$$

This model assumes that the steering response is independent of the vehicle states. This ignores variations in the steering response due to vehicle velocity, yaw rate, etc. One of the goals of the parameter estimation algorithm discussed in Chapter 3 is to adapt to these changes in steering dynamics.

Chapter 3

Delay Compensation Algorithm

3.1 System Architecture

Figure 3.1 shows how the delay compensation algorithm would fit into a typical autonomous vehicle software structure. The algorithm can be inserted between the high level software (path planning, path tracking controller, etc.) and the low level software and hardware (autonomous steering system, braking system, throttle control, etc.). This allows for delay compensation without alteration of the high level algorithms or low level hardware. The inner-loop takes desired steer angle as an input and uses measured steer angle in the estimation algorithm. No other inputs or sensor measurements are necessary.

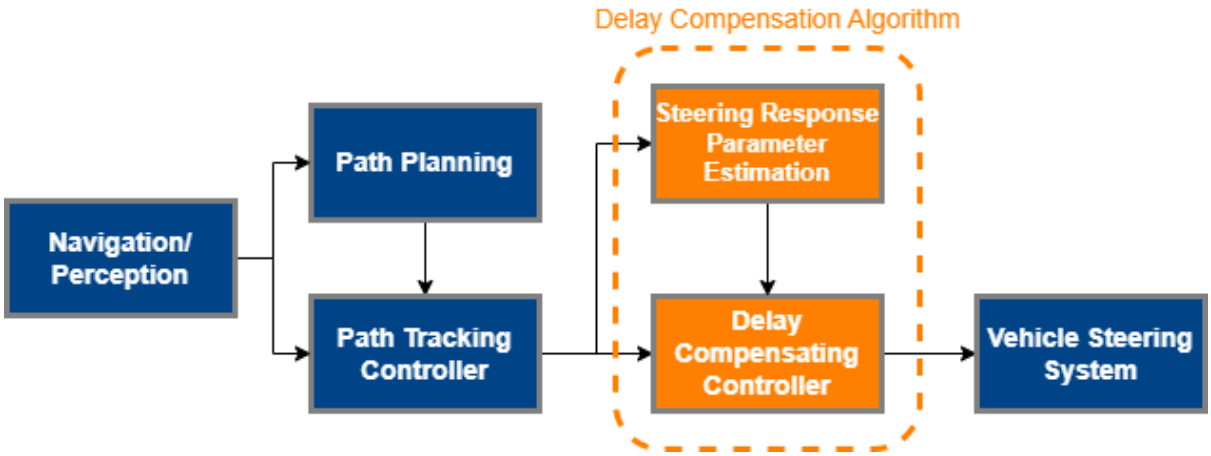


Figure 3.1: System Software Architecture

3.1.1 Inner-loop Control Design

The goals of the inner-loop control design are to quicken the steering response, to increase the gain and phase margins of the closed loop system, and to provide filtering of high frequency

steering commands in order to reduce oscillations. This is accomplished by introducing a low-level controller to regulate the steer angle commands that are passed into the steering system. Because of the communication delay between the control computer and the steering system, implementing an inner-loop controller without delay compensation could lead to additional oscillation and overshoot, possibly worsening path tracking performance. For this reason, a delay compensation method is necessary when implementing an inner-loop steering controller.

The Smith Predictor (SP) is a control algorithm designed to compensate for pure time delays. It was initially developed to compensate for large delays (1 second or more), but it has been applied to a variety of time delay systems with various amounts of delay. The SP feeds back predictions of future states to reduce overshoot and oscillation in the control system. Applying the delay compensation in the inner-loop allows for the steering dynamics to be sped up without introducing extra oscillation or adverse effects.

The block diagram of the inner-loop with SP compensation is shown in Figure 3.2. G_i and $G_{i,m}$ represent the actual and modeled steering dynamics, respectively, excluding the pure delay. T_d and $T_{d,m}$ represent the actual and modeled pure delay. The commanded steer angle from the outer-loop is fed into the inner-loop controller, C_i. The inner-loop controller adjusts the steer angle commands to reduce oscillation and speed up the steering dynamics. A model of the steering dynamics and an estimate of the pure time delay are used to predict future steer angles. The predicted steer angles are fed back, mitigating the effects of the pure time delay.

The closed loop transfer function of the SP system is shown in Equation (3.1).

$$\frac{\delta(s)}{d(s)} = \frac{C_i T_d G_i}{1 + C_i G_{i,m} - C_i G_{i,m} T_{d,m} + C_i G_i T_d} \tag{3.1}$$

When the modeled plant and time delay match the true plant and time delay exactly ($T_{d,m} = T_d$ and $G_{i,m} = G_i$), the effects of the pure delay are removed from the characteristic equation of the closed-loop system. This effect can be seen in Equation (3.2).

$$\frac{\delta(s)}{d(s)} = \frac{C_i T_d G_i}{1 + C_i G_{i,m}} \tag{3.2}$$

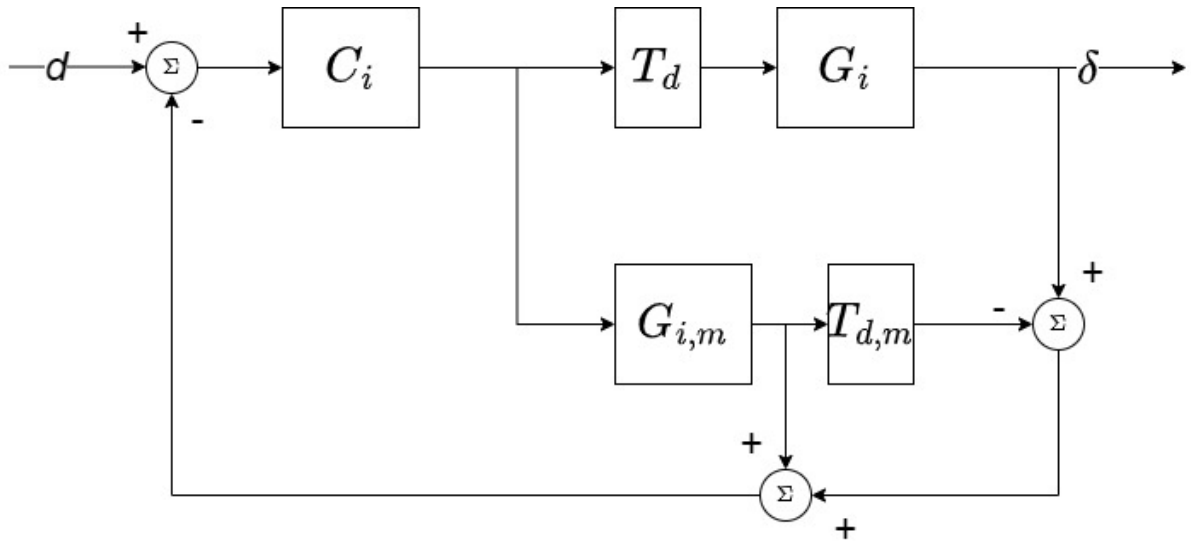


Figure 3.2: Block Diagram of Inner Control Loop and SP.

After the pure delay is removed from the characteristic equation, the inner-loop controller can be designed assuming no delay. This simplifies the control design process and allows for more aggresive controllers to be implemented. The delay will still be present in the numerator of the closed loop transfer function, which means the system will still track reference changes with a delay. For the purpose of control design, the steering dynamics were assumed to be modeled by a first order system and nonlinearities in the steering system were ignored. For some actuators, this assumption may need to be adjusted, but the steering actuators on the test vehicles examined in this thesis exhibit characteristics of first order systems.

Assuming an actuator response similar to that of the Lincoln MKZ test vehicle described later in Chapter 4, the controller was designed to increase the bandwidth of the steering dynamics and to provide filtering at higher input frequencies. The assumed actuator dynamics, found experimentally, are described by a first order system with a time constant of 0.1898 seconds. A discrete compensator with one pole and two zeros was chosen for the inner-loop controller. The transfer function of the controller in the continuous domain is shown in Equation (3.3).

$$C_i(s) = \frac{d_i(s)}{e_\delta(s)} = K_i \frac{s+10}{(s+15)(s+16)} \tag{3.3}$$

The poles and zeros of the controller were chosen in the continuous domain, then transformed into the discrete domain using the Tustin method of discretization at a sampling frequency of 100 Hz. The transfer function of the discrete compensator is shown in Equation (3.4).

$$C_i(z) = \frac{d_i(z)}{e_\delta(z)} = \frac{0.004522z^2 + 0.0004307z - 0.004091}{z^2 - 1.712z + 0.733} \tag{3.4}$$

Pre-reference scaling is used to ensure a DC gain of 1. The closed loop frequency response of the inner-loop is compared to that of the unmodified actuator dynamics in Figure 3.3.

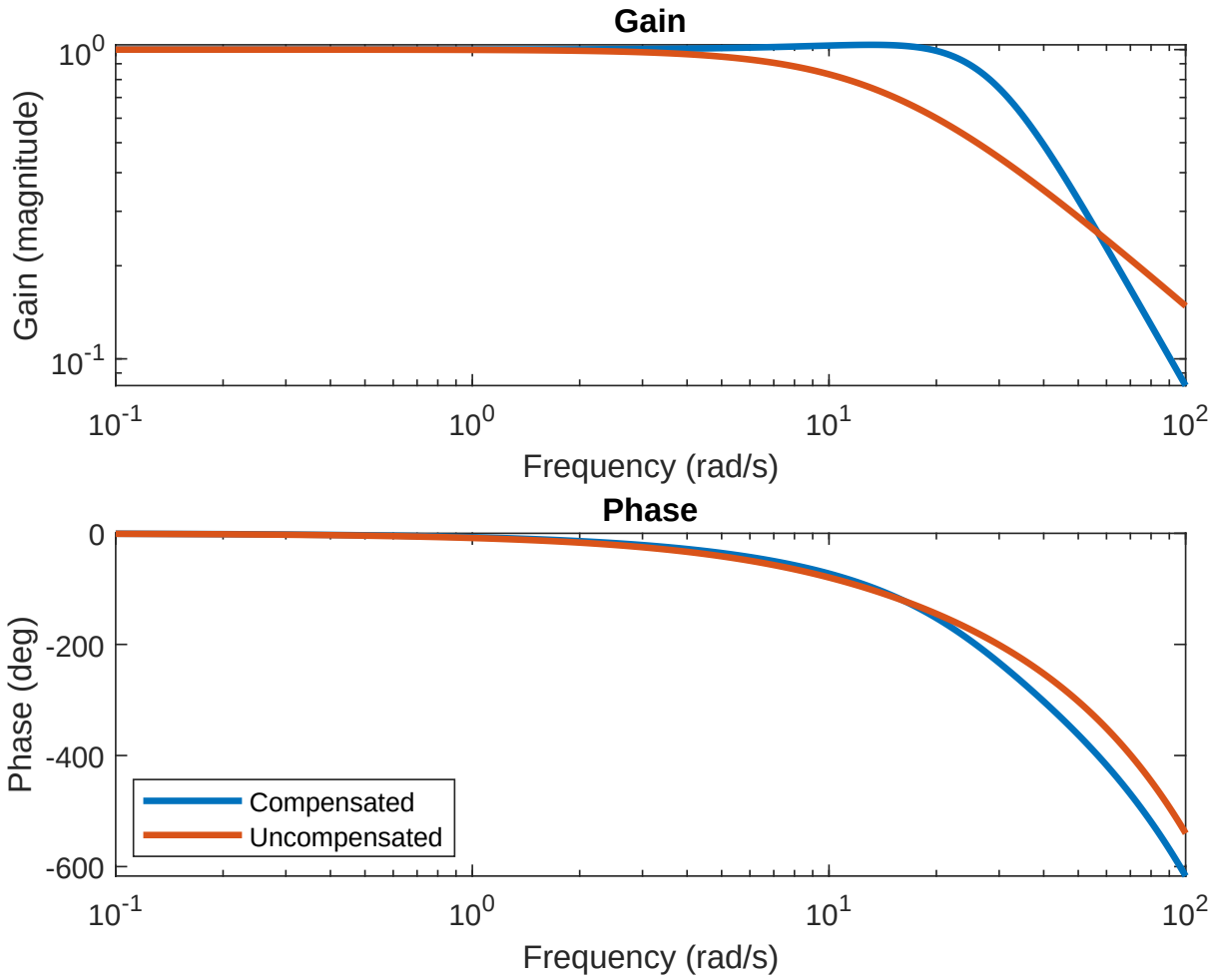


Figure 3.3: Comparison of Inner-Loop Frequency Responses.

The inner-loop controller provides a significant increase in bandwidth over the original actuator response. It also adds a steeper drop in gain at higher frequencies, providing more filtering at all frequencies greater than approximately 50 rad/s. The phase lag of the closed loop system is also lower for frequencies less than approximately 15 rad/s. The higher bandwidth

and increase in phase will provide better tracking of steer inputs and the additional filtering will help reduce steering oscillation. The increase in phase provided by the inner-loop compensation is plotted versus frequency in Figure 3.4.

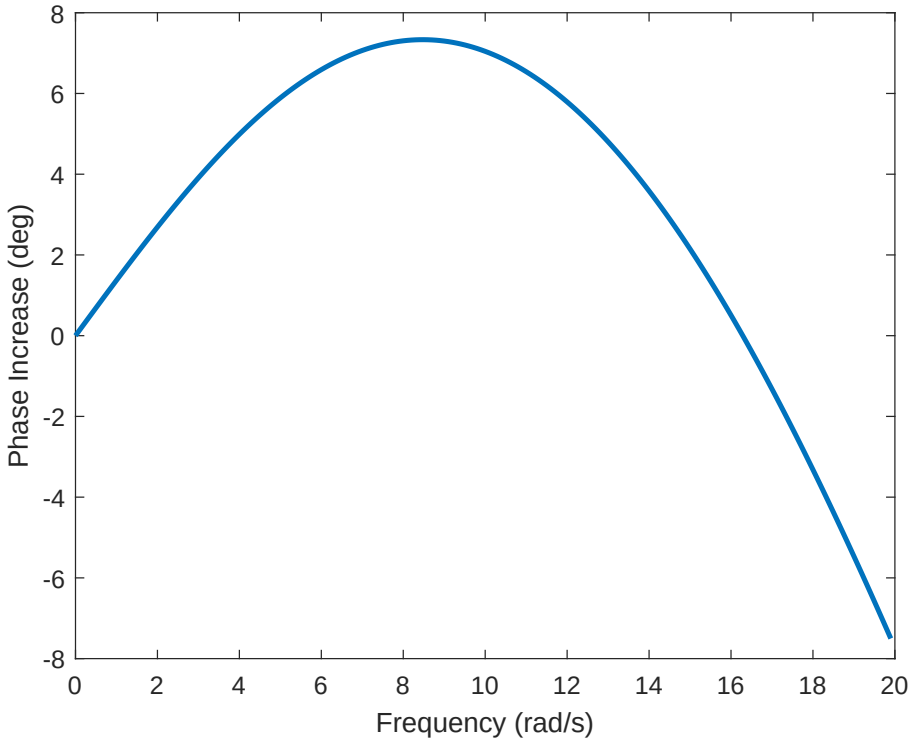


Figure 3.4: Phase Increase Provided by Inner-Loop Controller

The compensating controller is able to provide approximately a 7.5 degree increase in phase at a frequency of around 9 rad/s, and also increases the phase for all frequencies up to 15 rad/s. This phase increase in the inner-loop will lead to an increase in phase margin in the outer-loop. For frequencies from 0 to around 15 rad/s, the compensation increases the phase of the inner-loop. Therefore, if the desired outer-loop crossover frequency (approximate closed loop bandwidth) is within this range, the phase margin of the outer loop will be increased by the value shown previously in Figure 3.4. This analysis is based on the control design discussed previously in this section and assumes a perfect model of both actuator delay and dynamics. If the model does not match perfectly, this analysis becomes an approximation, but improvement is still expected for small model inaccuracies.

For actuators with significantly different dynamics from the MKZ test vehicle, the inner-loop controller may need to be altered slightly to achieve the desired performance. This design

works well for actuators with dynamics close to those of the MKZ. When the SP is assumed to have a perfect model of the plant and time delay, the delay can be considered effectively removed from the system response. However the model will never be identical to the true dynamics, which creates mismatch between the modeled and true dynamics. This mismatch, if large enough, can lead to poor performance and, in some cases, instability.

When model mismatch is present in the SP system, the closed loop transfer function shown in Equation (3.1) can be represented as shown in Equation (3.5).

$$\frac{\delta(s)}{d(s)} = \frac{C_i T_d G_i}{1 + C_i G_{i,m} + C_i(G_i T_d - G_{i,m} T_{d,m})} \tag{3.5}$$

]The mismatch term, $(G_i T_d - G_{i,m} T_{d,m})$, represents the error between the modeled and actual system. Errors in both the dynamic model $G_{i,m}$ and the modeled delay $T_{d,m}$ can lead to poor performance and instability if large enough. A detailed sensitivity analysis with respect to delay mismatch is performed in [60]. Additional analyses of SP robustness to model mismatch can be found in [61, 62]. In general, smaller mismatch leads to smaller prediction errors and better overall controller performance. To maintain low prediction errors even with uncertain or changing dynamics, a parameter estimation algorithm was implemented to estimate the actuator model and adapt the SP.

3.2 Parameter Estimation Algorithm

The steering response of an autonomous vehicle may not be known accurately during the control design process. Additionally, the pure delay value as well as the steering dynamics can change over time. To make the delay compensation algorithm robust to unknown or changing steering response, an estimation algorithm was developed to estimate both the pure delay and the steering dynamics. The following assumptions were made to simplify the estimation process:

1. The steering dynamics are assumed to be a first order linear process.

2. Measurements of current steer angle are assumed to be available at the sampling rate T.

3. The pure delay is assumed to be a multiple of the sampling time, meaning it can be represented as an integer value of samples.

4. The pure delay value is bounded by $\alpha_{min} \leq \alpha \leq \alpha_{max}$.

While these assumptions may not be true on a real system, they are assumed to be sufficiently close to the real response so as not to introduce excess error into the estimates. The pure delay is not an exact multiple of the sampling rate, however knowing the delay more precisely than the sample rate would not be useful as a fractional pure delay cannot be implemented in the SP architecture.

The discrete domain first order model that the steering data will be fit to is shown in Equation (3.6).

$$\frac{\delta(z)}{d(z)} = z^{-\hat{\alpha}} \frac{\hat{b}}{z - \hat{a}} \qquad where \qquad \hat{\alpha} = \frac{\hat{\tau}}{T} \tag{3.6}$$

T represents the sampling rate. Because of the simplicity of the model, only three parameters need to be estimated: the two steering dynamics parameters, a and b, and the pure delay, α (measured in samples). The algorithm used to estimate pure delay is the one presented in [41]. This method was chosen because it is relatively computationally simple, it can be used alongside standard parameter estimation methods, and it estimates the pure delay as a separate parameter which is necessary for the implementation of the SP. The delay estimation is coupled with a Kalman filter to estimate the steering dynamic parameters. A Kalman filter was chosen over other estimation methods, such as recursive least squares, to allow the estimator to better respond to the changing steering dynamics. The dynamic parameters are included in the parameter vector $\hat{\boldsymbol{\theta}}$ defined in Equation (3.7).

$$\hat{\boldsymbol{\theta}} = \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} \tag{3.7}$$

The estimation algorithm consists of two parts, both of which are executed at every time step. The dynamic parameter estimates are first updated using the current estimate of the pure delay value α, in samples. Next the pure delay estimate is updated using the most recent estimate of the dynamic parameters. The derivation of the Kalman filter and steps for updating the dynamic parameter estimates are given below. Both $\hat{a}$ and $\hat{b}$ are modeled as a Gaussian random walk, as shown in Equations (3.8) and (3.9).

$$\hat{a}_k = \hat{a}_{k-1} + w_k \quad where \quad w_{a,k} \sim N(0, Q_a) \tag{3.8}$$

$$\hat{b}_k = \hat{b}_{k-1} + w_k \quad where \quad w_{b,k} \sim N(0, Q_b) \tag{3.9}$$

This allows for the filter to track changes in the parameters without needing a precise model. The process noise covariance values are used to derive the process noise covariance matrix Q of the Kalman filter, shown in Equation (3.10).

$$Q = \begin{bmatrix} Q_a & 0 \\ 0 & Q_b \end{bmatrix} \tag{3.10}$$

Because $\hat{a}$ and $\hat{b}$ are modeled as a random walk, the state transition matrix used in the filter is simply the identity matrix. The time update equations for the parameter predictions and the covariance matrix P are shown in Equations (3.11) - (3.12).

$$\hat{\theta}^-_{k+1} = A_d \hat{\theta}_k \tag{3.11}$$

$$P^-_{k+1} = A_d P_k A_d^T + Q \tag{3.12}$$

A_d represents the state transition matrix. P and Q represent the estimate covariance and process noise covariance matrices, respectively. The measurement matrix, H, is dependent on the current delay estimate, $\hat{\alpha}$. k represents the current sample. The model shown in Equation (3.6)

is converted to a difference equation to obtain the prediction model for the steer angle δ, shown in Equation (3.13).

$$\hat{\delta}_k = \hat{a}\tilde{\delta}_{k-1} + \hat{b}d(k - \hat{\alpha}) \tag{3.13}$$

The measured angle from the last sample ($\tilde{\delta}_{k-1}$) and the commanded steer angle from $\hat{\alpha}$ samples ago ($d(k - \hat{\alpha})$) are used to predict the next steer angle measuremtent ($\hat{\delta}_k$). Since the actuator dynamics are known to have a DC gain value of 1, the relationship $\hat{a} + \hat{b} = 1$ must be satisfied. This and the prediction model defined in Equation (3.13) are used to form the measurement matrix H shown in Equation (3.14).

$$H_k = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} \tilde{\delta}_{k-1} & d(k - \hat{\alpha}) \\ 1 & 1 \end{bmatrix} \tag{3.14}$$

where $\tilde{\delta}_k$ represents the measurement of steer angle at timestep k. The measurement update portion is shown in Equations (3.15) - (3.17).

$$K_k = P_{k+1}^{-} H_k^T (H_k P_{k+1}^{-} H_k^T + R)^{-1} \tag{3.15}$$

$$z_k = \begin{bmatrix} \tilde{\delta}_k \\ 1 \end{bmatrix} \tag{3.16}$$

$$\hat{\theta}_{k+1}^{+} = \hat{\theta}_{k+1}^{-} + K_k(z_k - H_k \hat{\theta}_{k+1}^{-}) \tag{3.17}$$

$$P_{k+1}^{+} = (I - K_k H) P_{k+1}^{-} \tag{3.18}$$

R represents the measurement noise covariance matrix and K represents the Kalman gain matrix.

The matrix R is tuned based on the noise present in the measurements and Q is tuned based on the process noise present in the system, or the uncertainty in the system model. P is initialized based on the uncertainty in the initial estimates of the parameters $\hat{a}$ and $\hat{b}$. Once the

dynamic parameters have been updated, the new estimates are used to update the pure delay estimate by minimizing a cost function J, defined in Equation (3.19)

$$J = \sum_{t}^{i=0} [e_p(t-i)]^2 \tag{3.19}$$

where

$$e_p = \tilde{\delta}_k - \hat{\delta}_k = \tilde{\delta}_k - h_{1,k}\hat{\theta}_{k+1} \tag{3.20}$$

This is accomplished by searching over a predefined range of delay values shown in Equation (3.21).

$$\mathbf{r} = \begin{bmatrix} \alpha_{min} & \alpha_{max} \end{bmatrix} \tag{3.21}$$

The pure delay estimation and dynamic parameter estimation are be done simultaneously and each estimate will be updated at every timestep. The algorithm searches for the value of $\hat{\alpha}$, within the range $\mathbf{r}$ defined in Equation (3.21), that minimizes J. A smaller range $\mathbf{r}$ leads to a quicker computation time, but may cause issues if the true delay value is outside the range. A larger range can handle larger uncertainty in the initial delay estimate at the cost of a higher computational load.

The cost function and delay estimates are updated as shown in Equations (3.22) - (3.23).

$$\hat{\alpha}_{k+1} = argmin J_{k+1}(\hat{\alpha}) \forall \hat{\alpha} \in \mathbf{r} \tag{3.22}$$

$$J_{k+1} = \lambda J_k(\hat{\alpha}_{k+1}) + (\tilde{\delta}_k - h_{1,k}\hat{\boldsymbol{\theta}}_{k+1})^2 \tag{3.23}$$

A forgetting factor λ is used so that the algorithm prioritizes new data and responds better to changes in the true pure delay. The value of λ is between 0 and 1, with lower values putting

more weight on new data. As the value approaches 1 the algorithm is less reliant on new measurements. The cost function J is updated at each time step according to Equation (3.24).

$$J_{k+1}(\hat{\alpha}, \hat{\boldsymbol{\theta}}) = \lambda J_k(\hat{\alpha}, \hat{\boldsymbol{\theta}}) + (\tilde{\delta}_k - h_{1,k}\hat{\boldsymbol{\theta}}_{k+1})^2 \tag{3.24}$$

The current estimates of $\hat{\boldsymbol{\theta}}$ and $\hat{\alpha}$ are then used to calculate the new value of J. The function J is computed for each integer value of α within the range $\mathbf{r}$ and the value that yields the lowest cost is chosen as the updated delay estimate ($\hat{\alpha}$). This is shown in Equation (3.23). Next, the new value of J is updated with the new value of $\hat{\alpha}$, as shown in Equation (3.22).

The equations described above are implemented after the Kalman filter measurement update at every timestep. If the true delay value is within $\mathbf{r}$ and the system is linear and first order, the algorithm should converge near the exact values. The estimated parameters $\hat{a}$, $\hat{b}$ and $\hat{\alpha}$ are used to adapt the SP to ensure accurate predictions of steer angle and to minimize model mismatch. The block diagram of the inner-loop with adaptation is shown in Figure 3.5.

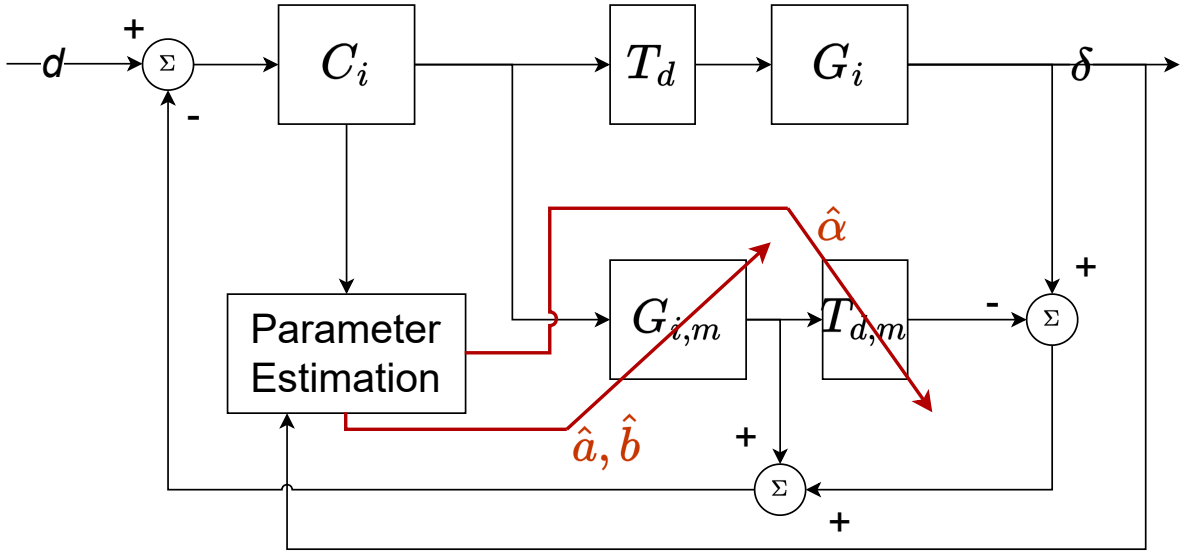


Figure 3.5: Adaptive SP Inner-Loop Block Diagram

Chapter 4

Simulated Performance Analysis with Various outer-loop Controllers

One of the benefits of implementing delay compensation in an inner feedback loop is that the designer has more freedom when choosing the outer-loop path tracking controller. A wide variety of vehicle path tracking control schemes have been designed, implemented, and shown to be effective in both simulation and in real-time experiments. The delay compensation should provide improvements regardless of the outer-loop control design. In this chapter, the performance of the inner-loop compensation will be examined in a path following control architecture with various outer-loop control schemes. This will attempt to show that performance improvement can be achieved using inner-loop compensation for most outer-loop controllers. The following sections will describe the simulation set up, present each outer-loop controller tested, and show the results and conclusions from the various controllers.

4.1 Simulation Model Description

Each simulation was performed in MATLAB. The linear bicycle model, described in Section 2.5, was used to simulate the dynamics of the vehicle. The model parameters of the bicycle model were chosen to match those of the test vehicle used for the real-world experiments shown in a later section. These parameters are given in Table 4.1.

The vehicle model was discretized at 100 Hz for the simulation, resulting in the state space model shown in Equation (4.1).

$$\begin{bmatrix} \psi_{k+1} \\ V_{y,k+1} \end{bmatrix} = \begin{bmatrix} 0.9898 & 0.0022 & 0 \\ -0.0097 & 0.9891 & 0 \end{bmatrix} \begin{bmatrix} \psi_k \\ V_{y,k} \end{bmatrix} + \begin{bmatrix} 0.0350 \\ 0.0641 \end{bmatrix} \delta \tag{4.1}$$

Table 4.1: Bicycle model parameters used for MATLAB simulation.

Parameter	**Symbol**	**Value and Units**
Mass	m	$1856\ Kg$
Yaw moment of inertia	I_{zz}	$4292\ Kg * m^2$
Rear Cornering Stiffness	$C_{\alpha r}$	$120,000\ N/rad$
Front Cornering Stiffness	$C_{\alpha f}$	$184,600\ N/rad$
Distance from c.g. to front/rear axle	a/b	$1.257/1.593\ m$
Longitudinal Speed	V_x	$Kg * m^2$
Lateral Speed	V_y	$Kg * m^2$
Steer Angle	δ	rad

Measurements of position, heading, and steer angle are assumed to be available for the simulation. The steer angle measurements are modeled as an encoder with a resolution of 0.18 degrees. The heading and position measurements were corrupted by zero mean Gaussian noise as shown in Equations (4.2) - (4.4).

$$\tilde{\psi} = \psi + \omega_\psi, \quad \omega_\psi \sim N(0, 0.25^2)\ \ (degrees) \tag{4.2}$$

$$\tilde{N} = N + \omega_N, \quad \omega_N \sim N(0, 0.02^2)\ \ (meters) \tag{4.3}$$

$$\tilde{E} = N + \omega_E, \quad \omega_E \sim N(0, 0.02^2)\ \ (meters) \tag{4.4}$$

The steering delay and dynamics were modeled as a first order linear time delay system. The values for the steering delay and the steering dynamic's time constant were varied on each run to test the ability of the estimation algorithm to converge to the correct parameters. The nominal values for these parameters were chosen from experimental steering step response data taken from the test vehicle. A model was chosen that closely matched the measured response of the vehicle's actuator. The step response of the test vehicle steering actuator along with the simulated response to the same input is shown in Figure 4.1. The simulated response is shown to nearly match the measured response.

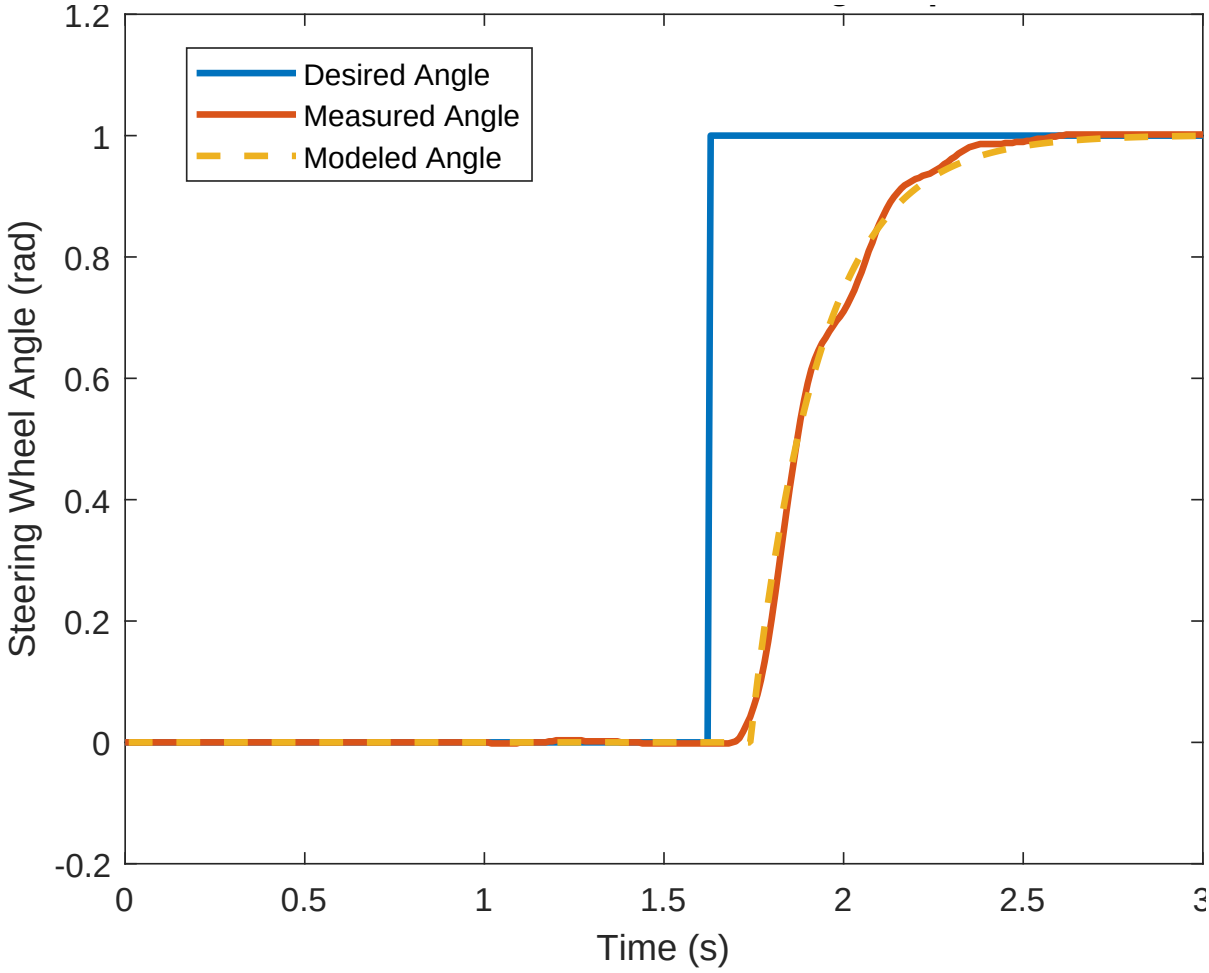


Figure 4.1: MKZ Measured and Modeled Steering Response to Step Input

The nominal model of the steering response, obtained from the data shown in Figure 4.1, is shown in Equation (4.5),

$$G(z) = z^{-10}\frac{0.0513}{z - 0.9487} \tag{4.5}$$

discretized at 100 Hz for the simulation. This model corresponds to a steering time constant of 0.1898 seconds and a pure delay of 0.1 seconds. To simulate modeling inacuracies, both the time constant and the pure delay value were varied at the start of each run as shown in Equations 4.6 and 4.7.

$$T_c = T_{c,nom} + \omega_{tc}, \qquad \omega_{tc} \sim N(0, 0.0025^2) \tag{4.6}$$

$$T_d = T_{d,nom} + \omega_{td}, \qquad \omega_{td} \sim N(0, 0.05^2) \tag{4.7}$$

Gaussian random variables ω_{tc} and ω_{td} are added to the nominal time constant and pure delay values, respectively. The distributions of ω_{tc} and ω_{td} are also shown in (4.6) - (4.7).

4.2 Simulation Setup

The goal of the simulated vehicle was to track a reference path as closely as possible and to accurately estimate the steering response parameters. Four outer-loop controllerers were tested. The simulation was run at speeds of 10 and 15 m/s for each controller. For each combination of outer-loop control scheme and longitudinal speed, the following five tests were run:

- Sim 1 (No Delay): No delay, no compensation.
- Sim 2 (Delay): Delay, no compensation.
- Sim 3 (Compensated): Delay, with non-adaptive inner-loop SP compensation using nominal parameters.
- Sim 4 (Adaptive): Delay, with inner-loop SP compensation and delay estimation. Nominal model used to initialize estimation algorithm.
- Sim 5 (Adaptive Converged): Delay, with fixed inner-loop SP compensation using parameter estimates from Sim 4 for prediction model.

Sim 1 shows the performance that was designed for assuming a perfect actuator. Sim 2 shows the theoretical controller performance in the presence of actuator delay and dynamics, and Sim 3 shows the performance using non-adaptive inner-loop compensation inserted to mitigate the effects of the actuator delay. Sim 4 shows the performance of the full, adaptive inner-loop algorithm, and in Sim 5 the tests are re-run with the model obtained by the estimation algorithm in Sim 4, but the model was not adapted in-run. The goal will be for the results

of Sims 3-5 to be as close as possible to the results of Sim 1. Sim 1 was only performed once because the vehicle and steering model was assumed to be known and constant for this run, however Sims 2-5 were each performed 100 times, varying the actuator delay as described in Section 4.1, and the results were averaged across all runs. Succesful compensation would mean that the results from Sims 2-5 are similar to those of Sim 1.

4.3 Heading Controller

The first outer-loop controller tested in the Monte Carlo simulation was a discrete heading controller. A heading controller is one of the simplest methods of path following. It attempts to track the reference path by simply pointing the vehicle towards target waypoints selected from the path.

4.3.1 Control Formulation

To calculate the heading error to be fed into the discrete heading controller, a point is selected from the path based on the look-ahead distance. The bearing from the current vehicle position to the selected point in the local frame is then calculated. This calculated bearing is then compared to the current vehicle heading relative to the local frame to calculate heading error. This heading error value is fed into the control law. The heading error calculations, illustrated in Figure 4.2, are shown in Equations (4.8) - (4.9).

$$\Psi_{target} = arctan(\frac{E_{ref} - E}{N_{ref} - N}) \tag{4.8}$$

$$e_{\psi} = \Psi_{target} - \Psi \tag{4.9}$$

The calculated heading error, e_{ψ}, is supplied to the discrete heading controller. The controller was designed as a lead compensator, with the transfer function shown in Equation (4.10)

$$\frac{\delta_{OL}(z)}{e_{\psi}(z)} = K\frac{z - 0.7}{z - 0.2} \tag{4.10}$$

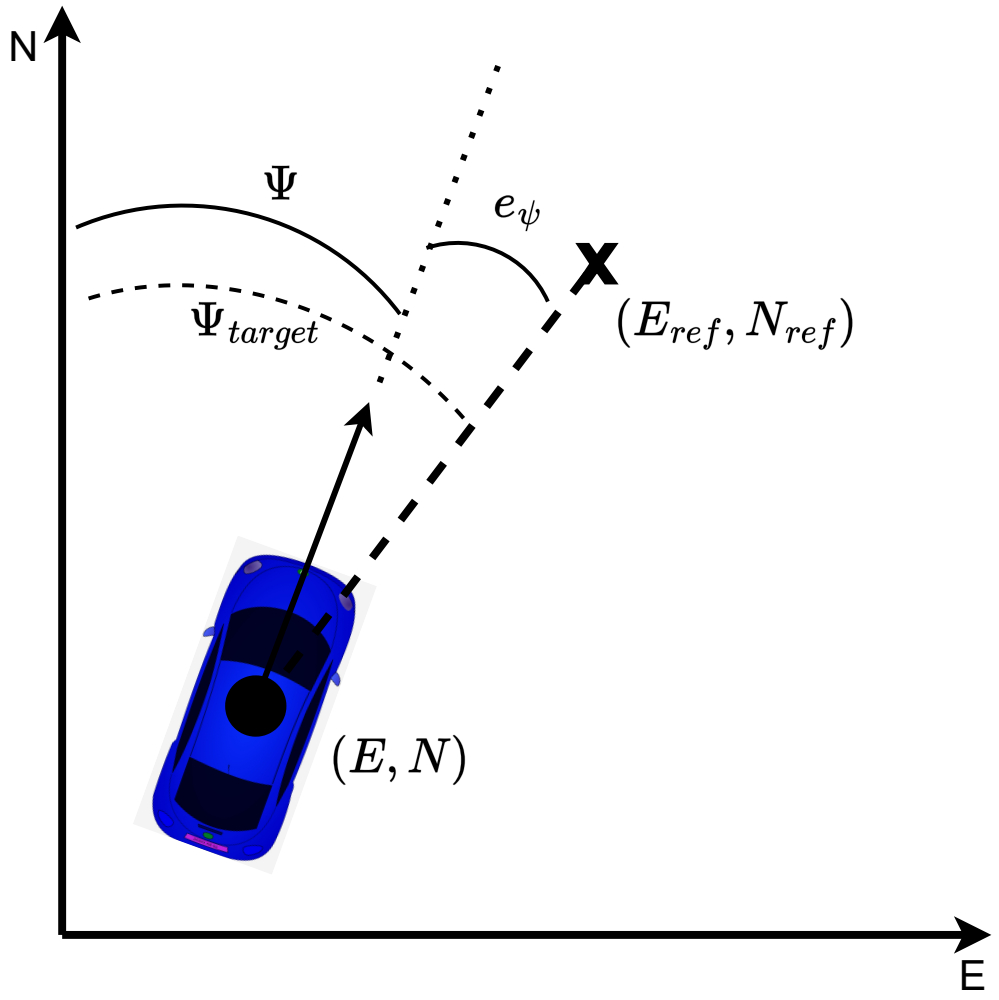


Figure 4.2: Heading Error Calculation Illustration

where K represents the outer-loop gain. The calculated steer angle from the outer-loop controller was then supplied to the inner-loop, or sent directly into the simulated steering system for the uncompensated runs.

4.3.2 Heading Controller Results

The minimum and maximum heading, lateral path, and steer angle error for the heading controller across all 100 Monte Carlo runs are shown in Figure 4.3. The mean errors are also shown with an asterisk. Run 1 is labeled as "no delay," run 2 is labeled as "delay," and runs 3 and 4 are labeled as "compensated" and "adaptive," respectively. Only the data from the 10 m/s run is shown here. The data from the 15 m/s run, as well as additional data from the 10 m/s run can be found in Appendix A. The heading error is calculated as the difference between the vehicles

current heading in the local frame and the heading to the desired waypoint. Lateral error is defined as the y-component of distance in the vehicles body frame from the vehicle's current position to the nearest path point. The steer angle error is defined as the difference between the desired steer angle calculated by the outer loop controller and the actual measured steer angle.

The compensated runs are shown to reduce the maximum and mean heading, steer angle, and lateral position errors. The maximum heading error is reduced by around 10 degrees, and the maximum lateral error is reduced by almost 1 meter. The mean heading and lateral errors for the compensated runs are almost as low as the mean errors for the undelayed run, showing that the compensation is nearly able to accomplish its goal of returning the delayed system performance to that of the undelayed system. The maximum and mean steer angle errors are reduced by both the compensated and Adaptive runs, showing that the compensated actuator is better able to track the desired steer angle than the uncompensated actuator. The mean and maximum heading and steer angle errors are reduced in the Adaptive and "Adaptive Converged" runs. Sim 5 (Adaptive Converged) shows lower errors than the Adaptive sim. This is due to the time that the adaptive algorithm takes to converge, leading to poorer performance during this time. Sim 5 is not adapted but is initialized with a more accurate model than Sim 3. This leads to lower errors for the Adaptive Converged run compared to both the Compensated and Adaptive runs.

Plots of the position, heading error and lateral path error of the simulated vehicle for the heading controller are shown in Figure 4.4, averaged across all 100 runs of each sim. It can be seen that the heading controller is susceptible to the negative effects of steering delay. Once the delay is introduced into the system, the response becomes more oscillatory and takes longer to reach steady state after the maneuver is complete. While the system response may not have become unstable when the delay was introduced without compensation, the response was undesirable.

Both the non-adaptive run and the adaptive run show significant performance improvements over the uncompensated run, as expected. The heading error and lateral path errors of Sims 3 and 4 stay below that of the uncompensated run for most of the maneuver. The inner-loop compensation is also able to significantly reduce oscillation in the lateral position of the

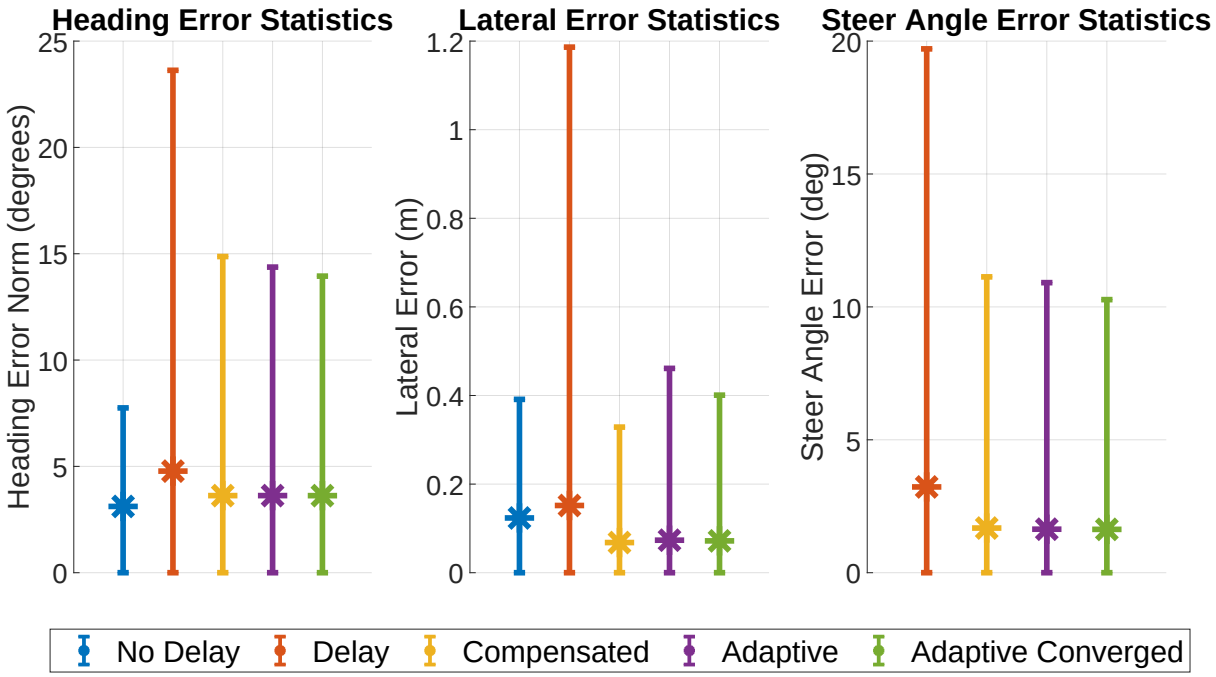


Figure 4.3: Heading Controller Error Statistics, V_x = 10 m/s

vehicle during the last portion of the double lane change. There is very little difference between the path tracking performance of the adaptive and non adaptive algorithms. This was expected as the variance of the steering parameters from run to run was relatively small for this dataset.

The estimation algorithm was able to reduce the prediction error significantly and converge to a more accurate model of the steering response. The norm of the parameter estimation errors over the course of the run are shown in Figure 4.5.

The estimation algorithm is able to reduce the error in the parameter estimates significantly over the course of the run. While the estimates do not always converge to the true values, they do improve from the initial parameter estimates on average. This improvement in the parameter estimates leads to a more accurate prediction of steer angle, which leads to less error in the Smith predictor. The prediction error over the course of the run is plotted for the Sims 3-5 in Figure 4.6. The SP model was adapted at a rate of 1 Hz for every Monte Carlo run. A relatively slow adaptation rate was chosen instead of adapting at every timestep because it was found that adapting at 100 Hz led to a more oscillatory response. The adaptive run is shown to have lower prediction error after the 1 second mark, when the prediction model is first updated with the parameter estimates. For the remainder of the run, the prediction error for the adaptive run stays

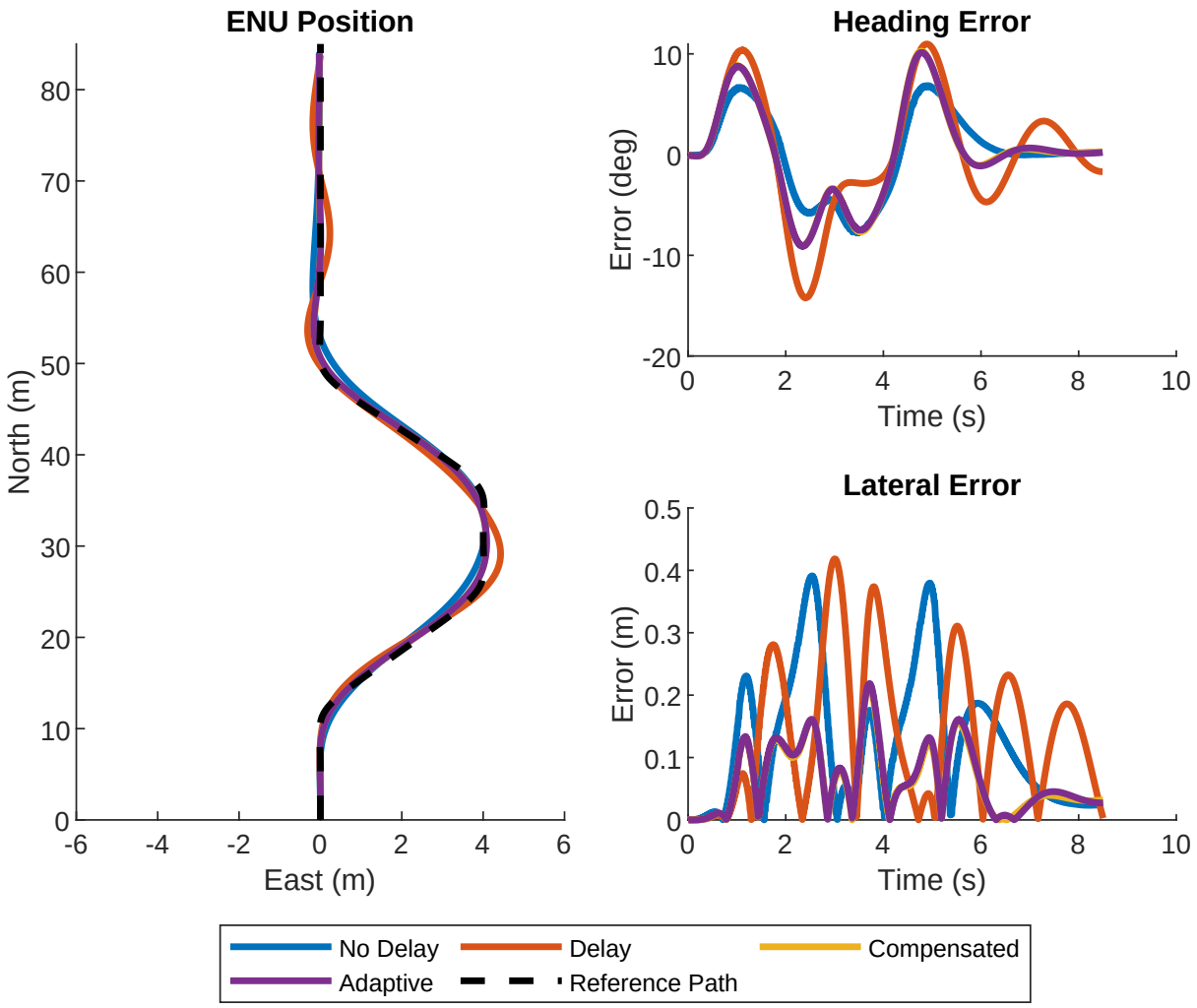


Figure 4.4: Heading Controller Double Lane Change Performance, V_x = 10 m/s

below that of the non-adaptive run. It is also shown to converge nearly to 0 degrees of error by the end of the run. Run 5 is labeled "Adaptive Converged," and is shown to have low prediction error throughout the run. This was expected as it was initialized with the model obtained from the adaptive sim, meaning an accurate model was used for the entire run.

Note that the errors in parameters $\hat{a}$ and $\hat{b}$ appear to exhibit the same response. This is due to the fact that the simulated steering dynamics always have a DC gain of 1. Additionally, the initial model for each Sim has a DC gain of 1. Since the Kalman filter constrains the estimated model to a DC gain of 1, the parameters $\hat{a}$ and $\hat{b}$ are tied to one another and will converge at very similar rates and display similar error characteristics.

As expected the heading controller performance is shown to be negatively affected by the introduction of delay. The oscillation introduced in the uncompensated runs led to undesirable

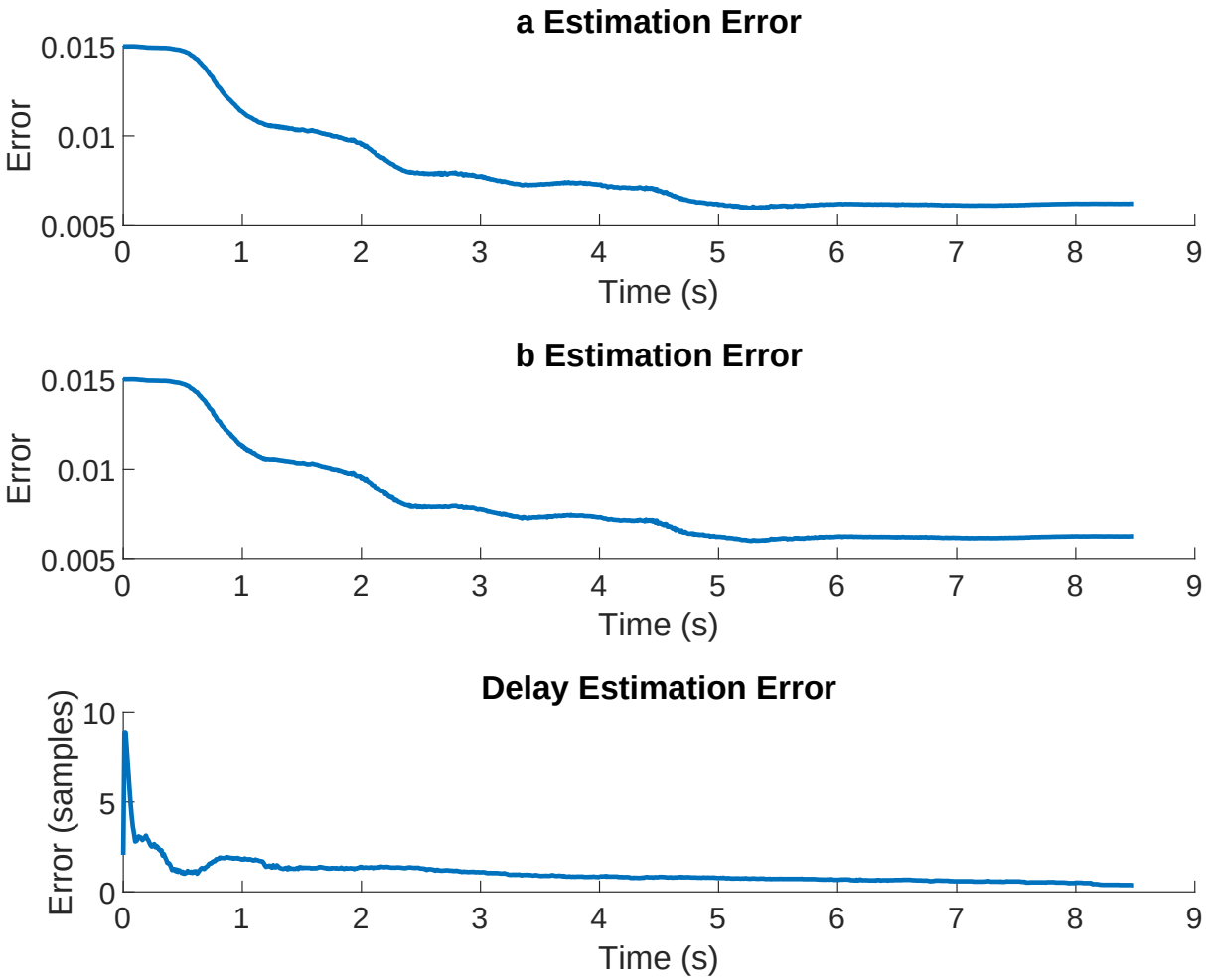


Figure 4.5: Heading Controller Parameter Estimation Errors, V_x = 10 m/s

path tracking performance. The SP compensation was able to mitigate these effects and ensure satisfactory path following performance in the presence of actuator delay. The parameter estimation algorithm was shown to converge nearly to the true model, on average, even though the adaptive algorithm did not provide noticeable improvements to the overall control performance.

4.4 Pure Pursuit with Kinematic Controller

Pure pursuit was one of the earliest proposed solutions to the vehicle path tracking problem, originally proposed in [63], and has been widely used in vehicle and robotics applications. It calculates a yaw rate command based on a selected waypoint position and the current vehicle position and velocity. The pure pursuit control law is applied here with a simple kinematic controller used to calculate desired steer angle from the yaw rate command.

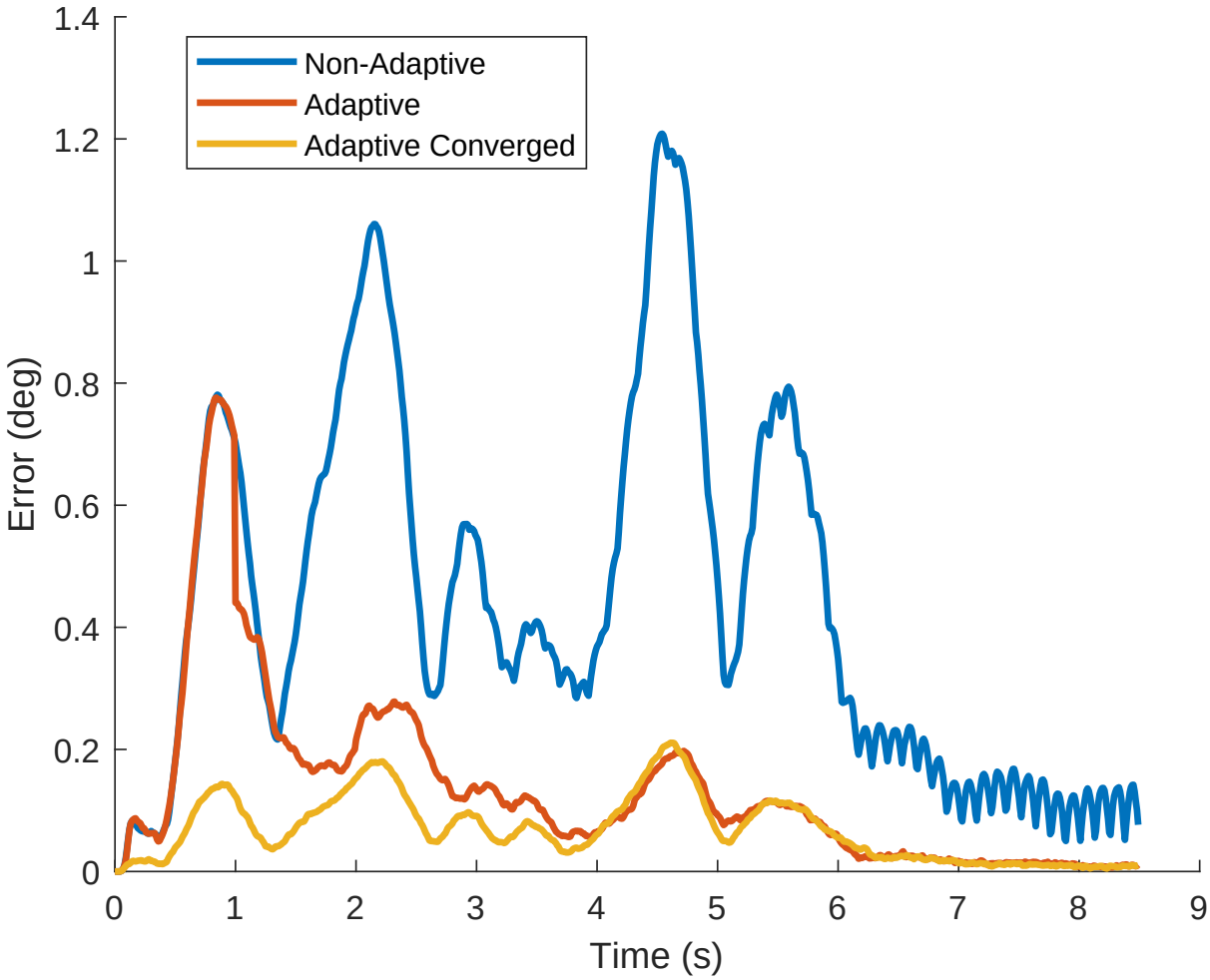


Figure 4.6: Heading Controller Steer Angle Prediction Errors, V_x = 10 m/s

Kinematic controllers work by calculating a steer angle based on a kinematic vehicle model. These controllers have the benefit of being simple to implement and have been shown to be effective in many autonomous driving and robotics applications. Only the wheelbase length and the longitudinal speed of the vehicle must be known to calculate the steer angle input, making this one of the easiest controllers to implement.

4.4.1 Control Formulation

The pure pursuit control law calculates a desired angular velocity by fitting a curve from the vehicle's current position to the position of the target waypoint. An illustration of this calculation is shown in Figure 4.7. The radius of curvature, R, is calcuated based on the angle from the vehicle to the target waypoint (α) and the look-ahead distance (L). The desired vehicle yaw rate is calculated as shown in Equations (4.11) - (4.12).

$$k = R^{-1} = \frac{2sin(\alpha)}{L} \tag{4.11}$$

$$\omega_{cmd} = V_x k = \frac{2V_x sin(\alpha)}{L} \tag{4.12}$$

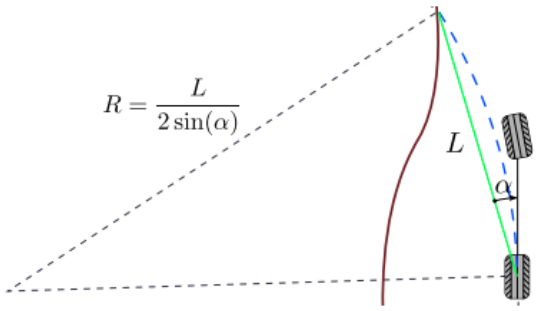


Figure 4.7: Pure Pursuit Illustration [7]

To make the controller more responsive to lateral position errors, an additional term was added to the angular velocity calculation. The full control law is shown in Equation (4.13).

$$\omega_{cmd} = \frac{2V_x sin(\alpha)}{L} + k_p L sin(\alpha) \tag{4.13}$$

The added term is the lateral error at the look-ahead point multiplied by a proportional gain. This ensures that the controller will minimize errors in lateral position. This additional term was found to provide improved tracking performance, especially at higher speeds and when the steering delay was included. To convert the commanded yaw rate into a desired steer angle, a kinematic vehicle model used. For simplicity, the sideslip angle of the vehicle is assumed to be very small, meaning only the longitudinal component of velocity is considered. The kinematic equation relating yaw rate and steer angle is shown in Equation (4.14)

$$\frac{V}{L} tan(\delta) = \omega \tag{4.14}$$

where L represents the wheelbase of the vehicle.This can be solved for δ to obtain the steer angle command that corresponds to the desired angular velocity, shown in Equation (4.15).

$$\delta = arctan(\frac{L\omega}{V_x}) \tag{4.15}$$

4.4.2 Pure Pursuit Results

The statistics of the heading, lateral, and steer angle errors for the pure pursuit Monte Carlo run are shown in Figure 4.8.

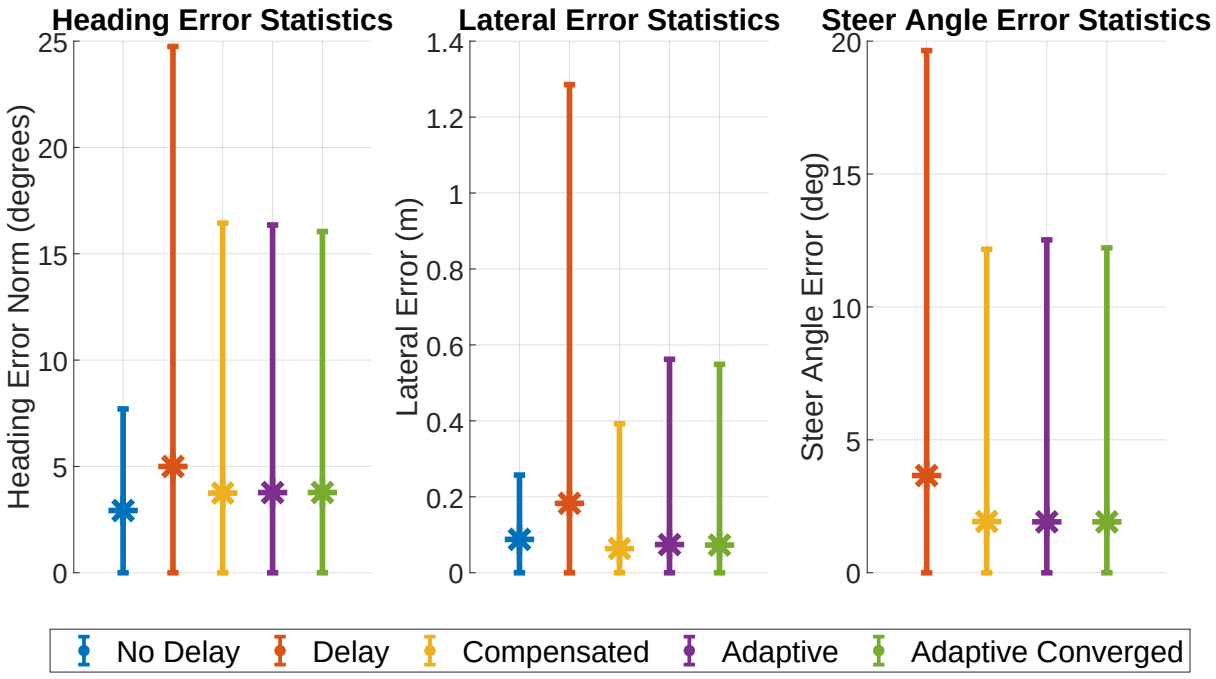


Figure 4.8: Pure Pursuit Kinematic Controller Error Statistics, V_x = 10 m/s

The position, heading error and lateral path error of the simulated vehicle for the pure pursuit controller are shown in Figure 4.9. The description of the plots in Figures 4.9-4.11 are the same as the previous section. The compensation is shown again to succeed at decreasing the heading and lateral errors once delay is introduced. The maximum heading error is reduced by around 10 degrees from the uncompensate run to the compensated and adaptive runs. The mean heading error is also reduced by around 1 degree. A similar trend is shown in the lateral error statistics, where the maximum and mean errors are decreased by around 1 and 0.1 meters, respectively, once the compensation is introduced. The maximum heading error for the adaptive run is slightly below that of the non-adaptive run, while the reverse is true for the lateral error. The mean errors for the adaptive and non-adaptive runs are almost identical. The mean and

maximum steer angle errors are again reduced in the compensated and adaptive runs, indicating that the actual steer angle more closely tracks the desired steer angle for these runs.

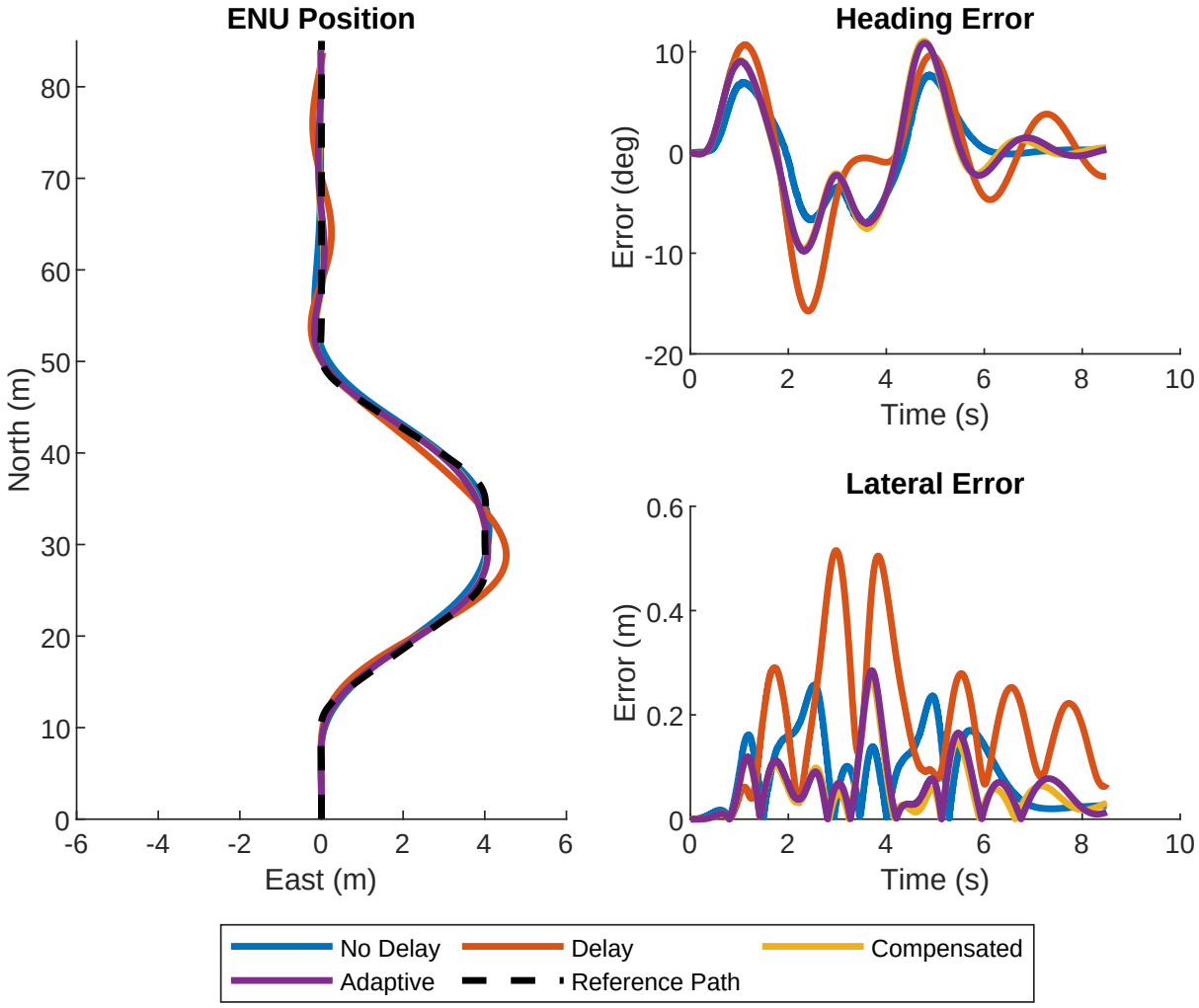


Figure 4.9: Pure Pursuit Kinematic Controller Double Lane Change, V_x = 10 m/s

The pure pursuit controller was more strongly affected by the introduction of delay than the heading controller. The uncompensated run shows oscillations both in lateral position and in heading error, as well as higher maximum and mean errors than the heading controller. This could be due to the lack of feedback in the controller design. Additionally, the only method of tuning the controller is by adjusting the look-ahead distance. There are no control gains to adjust to make the control less or more aggressive.

Both the non-adaptive run and the adaptive run again show improvements over the uncompensated run. The compensated and adaptive runs are again able to keep the heading and lateral position errors below those of the uncompensated run for most of the maneuver. There

is very little difference between the path tracking performance of the adaptive and non adaptive algorithms for this dataset, similar to the data shown for the previous controller. The estimation algorithm was able to reduce the prediction error on average and converge to a more accurate model of the steering response. The norm of the parameter estimation errors over the course of the run are shown in Figure 4.10.

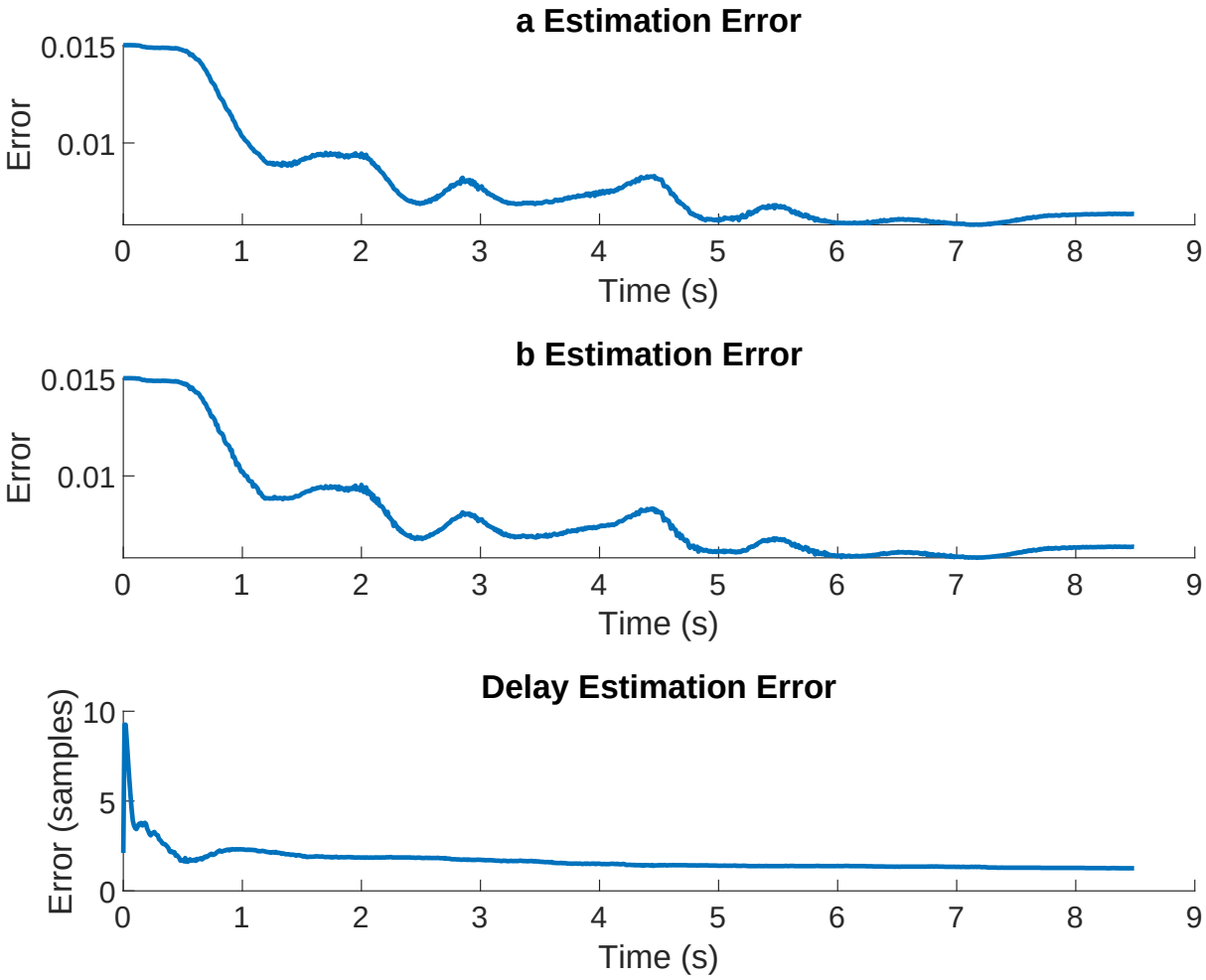


Figure 4.10: Pure Pursuit Kinematic Controller Parameter Estimation Errors, V_x = 10 m/s

The estimation algorithm is able to reduce the error in the parameter estimates, on average. The prediction improvement for the pure pursuit controller is less noticeable than the previous outer-loop controller, most likely due to the slightly different dynamics introduced by the new controller. The estimation still improves the prediction error. The prediction error over the course of the run is plotted for Sims 3-5 in Figure 4.11. Once the prediction model was updated after 1 second, the adaptive prediction error again stays below the non-adaptive error for the rest of the run. The prediction errors for Sim 5 are the lowest of the three runs, as expected.

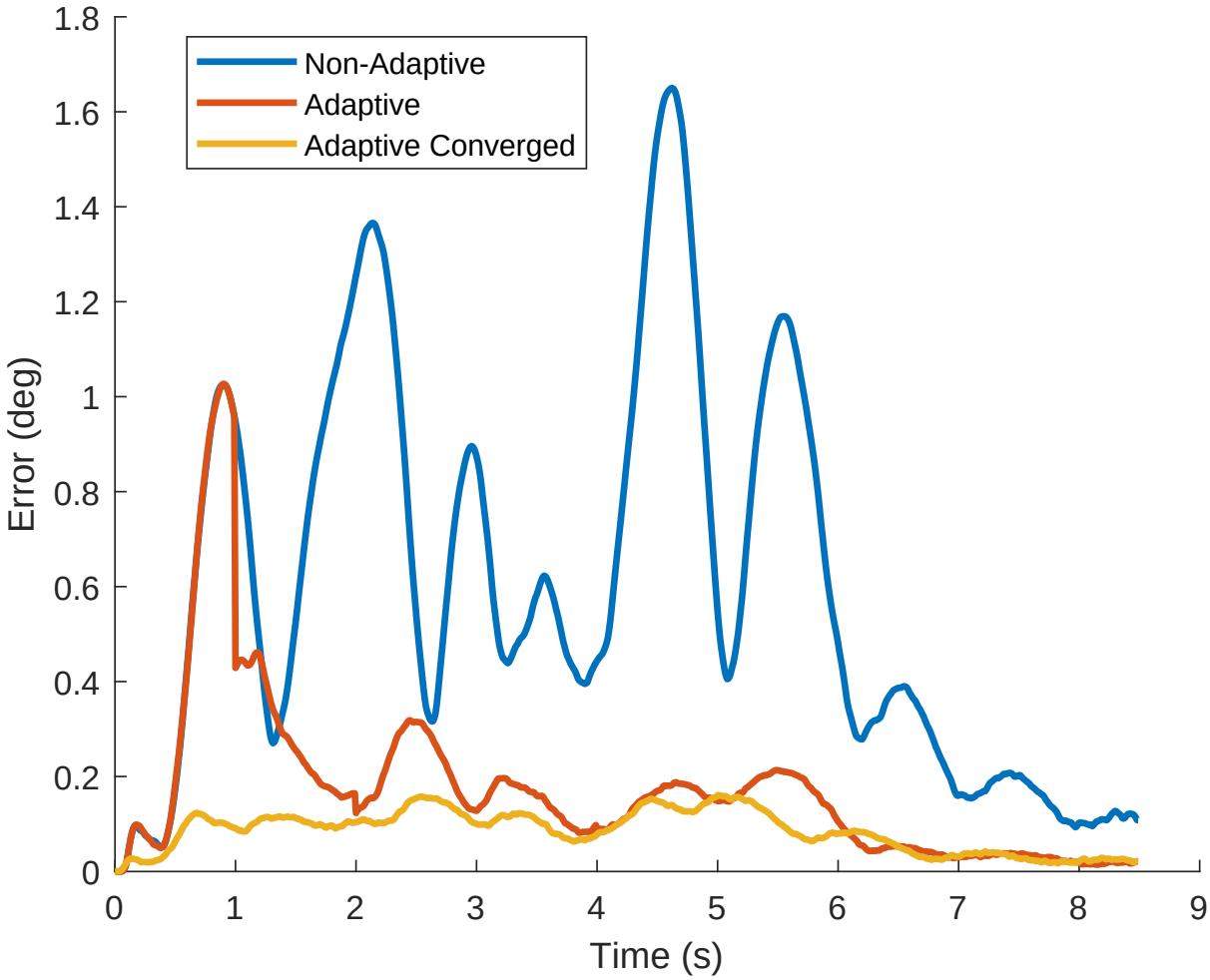


Figure 4.11: Pure Pursuit Kinematic Controller Steer Angle Prediction Errors, V_x = 10 m/s

The pure pursuit kinematic controller was shown to be susceptible to the negative effects of time delay, with results similar to those of the heading controller. The maximum and mean errors for the "delay" run were slightly higher than the previous controller, while the path tracking performance was stable but showed some undesirable characteristics. The compensation was able to improve the system response, reducing oscillation, heading error, and lateral path error. The parameter estimation algorithm converged within around 2 seconds to average errors under 0.005, and the improvements in steer angle prediction are shown. Overall, both the adaptive and the non-adaptive compensation schemes were succesful at improving the path tracking performance.

4.5 State Feedback Controller

4.5.1 Control Formulation

State feedback uses the state space model of the system to calculate control gains based on the desired closed loop eigenvalues of the system. For this controller, the discretized dynamic bicycle model is used to calculate the gain matrix. For control design, the dynamic bicycle model, with heading included as an additional state, was used. The state space model in the continuous domain is shown in Equation (4.16).

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{C_2}{I_{zz}V_x} & -\frac{C_1}{I_{zz}V_x} & 0 \\ -\frac{C_1}{mV_x} - V_x & -\frac{C_0}{mV_x} & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \\ \psi \end{bmatrix} + \begin{bmatrix} \frac{aC_{\alpha f}}{I_{zz}} \\ \frac{C_{\alpha f}}{m} \\ 0 \end{bmatrix} \delta \tag{4.16}$$

The parameter values from Table 4.1 are inserted into the model to obtain Equation (4.17).

$$\begin{bmatrix} \ddot{\psi} \\ \dot{V}_y \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -10.2214 & 2.2247 & 0 \\ -9.8581 & -10.9352 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ V_y \\ \psi \end{bmatrix} + \begin{bmatrix} 35.1445 \\ 64.6204 \\ 0 \end{bmatrix} \delta \tag{4.17}$$

The reference state used was heading to the desired waypoint. Since there is a pure integrator in the dynamics from steer angle to heading, no reference scaling is necessary to guarantee zero steady state error for step inputs. The desired eigenvalues were chosen based on the desired performance and were validated in simulation. Overdamped eigenvalues were found to have better performance than underdamped, so the real pole locations were chosen as shown in Equation (4.18).

$$s_{des} = [-10 \quad -9.9 \quad -9.8] \tag{4.18}$$

Because the simulation is run in discrete time at 100 Hz, the state space model and desired eigenvalues were transformed into the discrete domain. The discrete model and eigenvalues are shown in Equations (4.19) - (4.20).

$$\begin{bmatrix} \psi_{k+2} \\ V_{y,k+1} \\ \psi_{k+1} \end{bmatrix} = \begin{bmatrix} 0.9898 & 0.0022 & 0 \\ -0.0097 & 0.9891 & 0 \\ 0.01 & 0 & 1 \end{bmatrix} \begin{bmatrix} \psi_{k+1} \\ V_{y,k} \\ \psi_k \end{bmatrix} + \begin{bmatrix} 0.0350 \\ 0.0641 \\ 0 \end{bmatrix} \delta \tag{4.19}$$

$$z_{des} = e^{s_{des} T_s} = [0.9048 \;\; 0.9039 \;\; 0.9066] \tag{4.20}$$

To determine the gain matrix K, the "place" command in MATLAB is used to solve the equation shown in Equation (4.21).

$$|zI - A_d + B_d K| = 0 \tag{4.21}$$

The desired heading is calculated with the same method described in Section 4.3. The desired heading is then fed into the state feedback controller and the steer angle is calculated as shown in Equation (4.22).

$$\delta_{des} = \psi_{des} - K \left(\begin{bmatrix} \dot{\psi} \\ V_y \\ \psi \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \psi_{des} \right) \tag{4.22}$$

4.5.2 State Feedback Results

The heading, lateral, and steer angle error statistics for the state feedback controller are shown in Figure 4.12. The compensated and adaptive runs again have lower maximum and mean errors, however the improvement is less drastic compared to the previous outer-loop controllers. The compensated and adaptive runs still have maximum heading errors of over 20 degrees, compared to approximately 27 degrees of maximum heading error for the uncompensated run. This is in contrast to the previous two controllers, where the compensation provided a more noticeable reduction in maximum heading error. The reason for this difference is likely the more aggressive tuning of the state feedback controller compared to the previous controllers. Additionally, the controller gains for the state feedback controller are calculated based on a

model that does not include the steering dynamics. This leads to model inaccuracies and helps to explain the higher maximum and mean errors for this controller.

This controller setup is the first in which the adaptive algorithm performs noticably better than the non-adaptive compensation. For the state feedback controller the adaptive compensation provides improvement in the maximum heading error, lateral error, and steer angle error. This is likely also due to the design of the outer-loop controller. Because of the more aggresive steer angle commands from the outer-loop controller, errors in the steer angle prediction lead to more overshoot and oscillation. Therefore, the reduction in these prediction errors shown in the adaptive run lead to better overall control performance. The "Adaptive Converged" run outperformed both the compensated and adaptive runs in each of the error metrics. This is due to the presence of an accurate model for the entirety of the run. The compensation was again able to track the desired steer angle better than the non-compensated run, showing an improvement in mean and maximum steer angle errors.

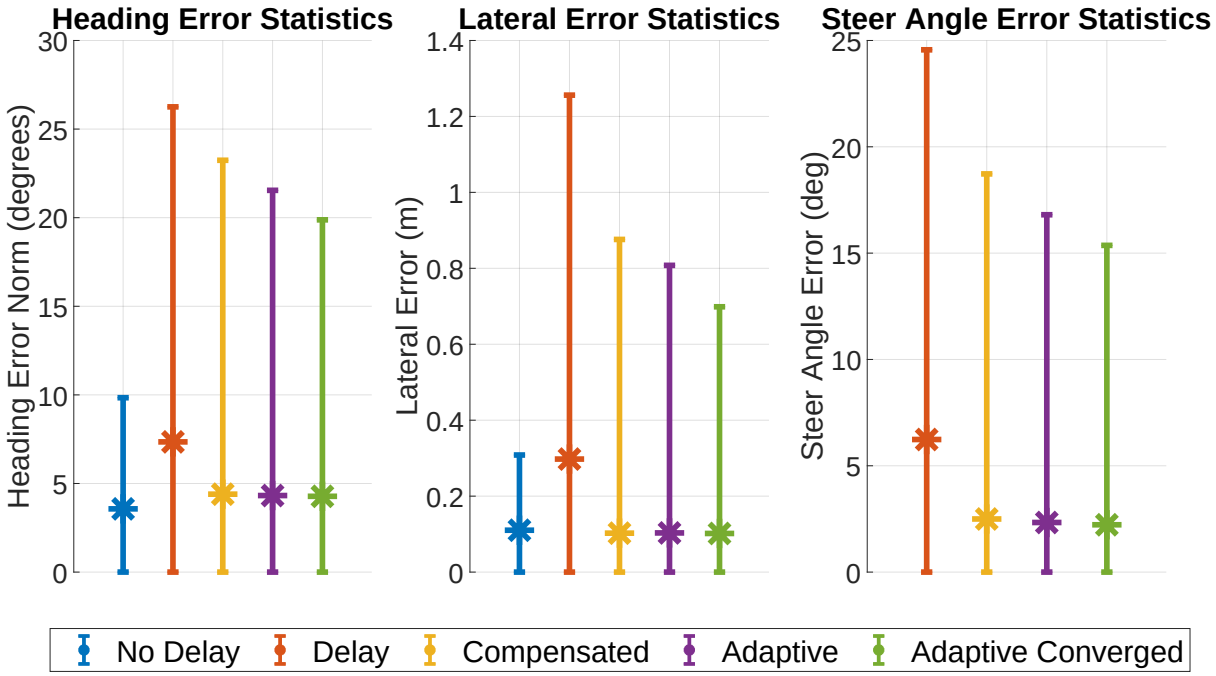


Figure 4.12: State Feedback Controller Error Statistics, V_x = 10 m/s

The position, heading error and lateral path error of the simulated vehicle for the state feedback controller are shown in Figure 4.13. The description of the legends and layout of Figures 4.13-4.15 can be found in Section 4.3.2. The controller performs poorly when delay is present

and there is no compensation, showing larger oscillations than the previous two outer-loop controllers. The average lateral error for all runs stays below 1 meter, and the average heading error stays below 20 degrees for the entire run. The compensated runs show better performance than the uncompensated run when delay is present, with reduced oscillation and faster convergence to the reference path after the maneuver. The lateral error for the compensated runs is also held below that of Sim 2, on average.

The norm of the parameter estimate errors for the state feedback controller are shown in Figure 4.14. The estimation performance is similar to the previous two outer-loop controllers. The dynamic parameter errors converge to an error of under 0.005, on average, within around 1 second. The average delay estimate error converges quickly to around 2 samples of error, and slowly approaches the true value as the run continues. The improvements in prediction error provided by the estimation can be seen in Figure 4.15. The non-adaptive prediction error for the state feedback controller is higher and more oscillatory than the previous two outer-loop controllers. This is likely due to the more aggressive and oscillatory nature of the steer commands calculated by the state feedback controller.

The state feedback controller was found to be more sensitive than the previous controllers to delay, showing worse performance for the uncompensated run. This was seen in Figure 4.12, where the mean and maximum errors for the "delay" run are higher than either of the first two controllers. The inner-loop compensation again provides better performance in terms of mean and maximum errors and reduced oscillation. The state feedback controller is more sensitive to inaccuracies in the SP model. This can be observed in the noticeable improvement in error provided by the adaptive algorithm. The sensitivity to model inaccuracies is also apparent in Figure 4.15 where the non-adaptive prediction is shown to have higher errors than the adapitve prediction. Sim 5 out-performed Sims 3 and 4 in terms of prediction error. This was expected due to the presence of a more accurate model for the whole run, in contrast to the adaptive run which does not converge unitl after 2 seconds.

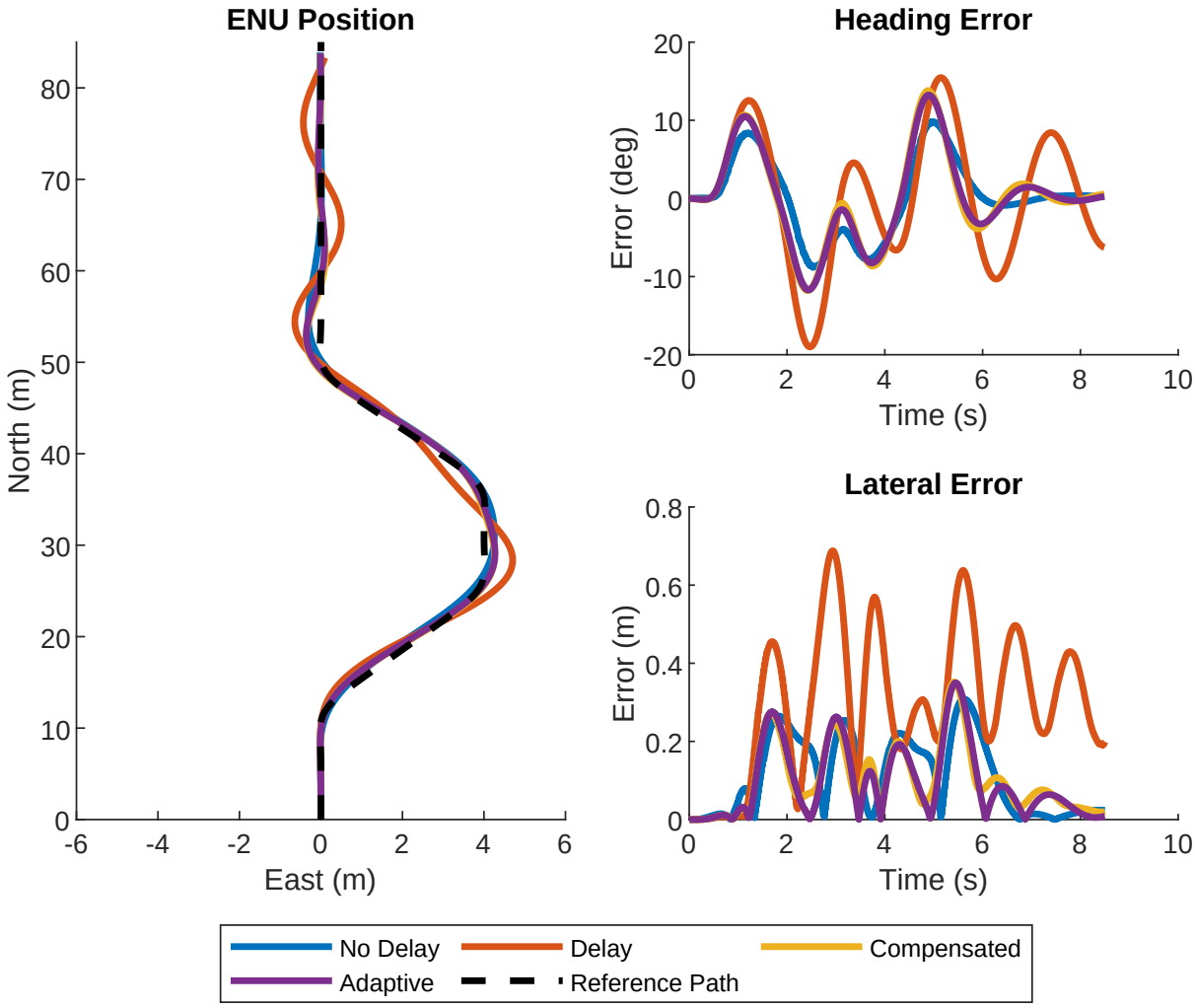


Figure 4.13: State Feedback Controller Double Lane Change Performance, V_x = 10 m/s

4.6 Model Predictive Control (MPC)

The last outer-loop controller tested was MPC, which is a type of predictive control that computes the optimal control input to follow a reference over a finite period of time. A dynamic model of the system, in this case the linear dynamic bicycle model, is used to predict the future system response in order to optimize the control inputs over the prediction horizon. The optimization is performed by minimizing a cost function that includes control effort, error, and in some cases system constraints. In this analysis, constraints will not be considered in the cost function for simplicity. The optimization is performed at every iteration, which can lead to long computation times for some systems. This has historically limited the application of MPC to

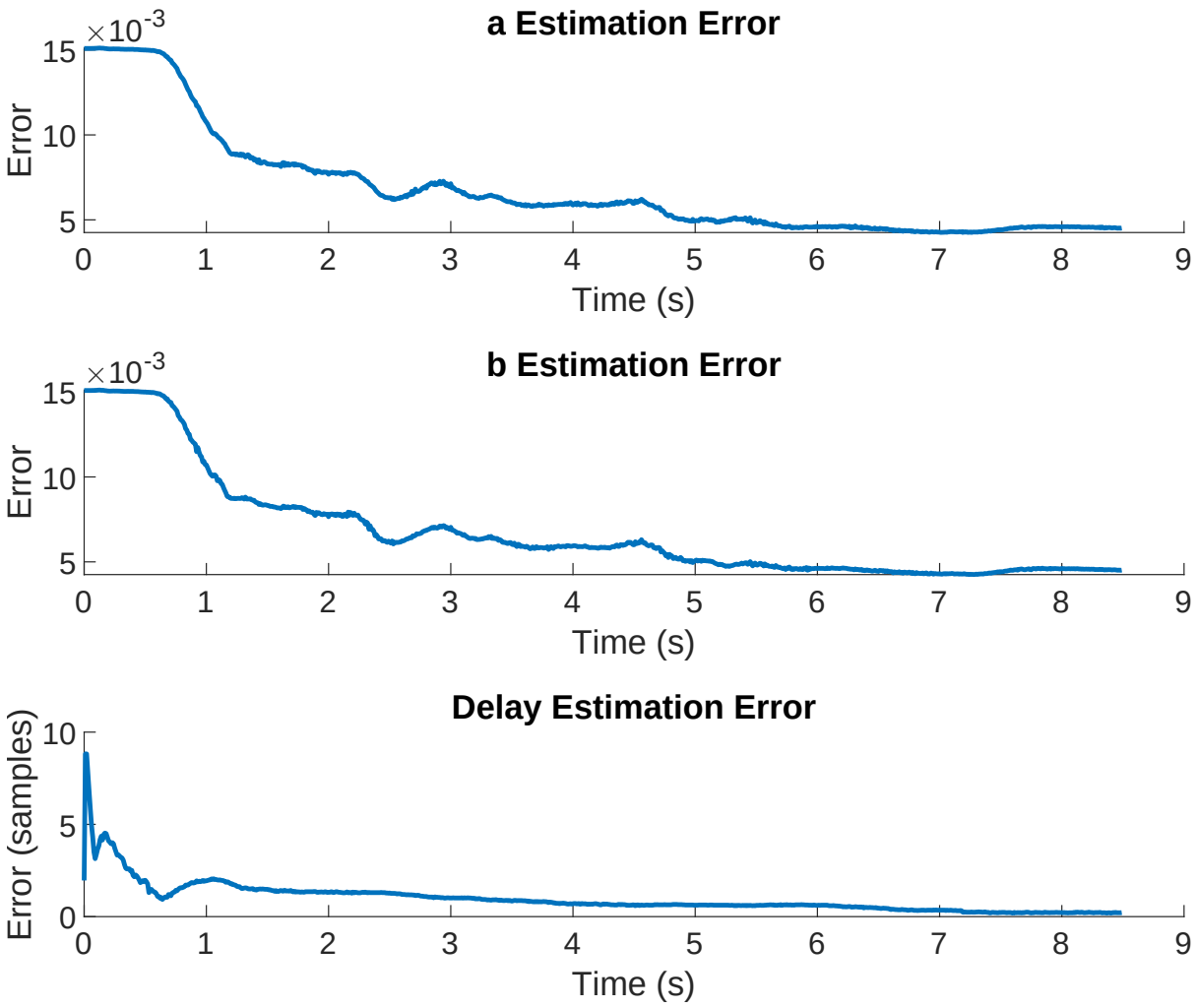


Figure 4.14: State Feedback Controller Parameter Estimation Errors, V_x = 10 m/s

slower systems, but advances in computing power have made it feasible to apply to a wide variety of control systems. The MPC control law for this analysis was derived following [64, 65].

4.6.1 Control Formulation

In order to minimize lateral path error and heading error, the bicycle model was derived in error states. Desired heading is defined as the heading to the current target waypoint. The two error states chosen are shown in Equations (4.23) and (4.24)

$$e_\psi = \psi - \psi_{des} \tag{4.23}$$

$$e_{lat} = Lsin(e_\psi) \tag{4.24}$$

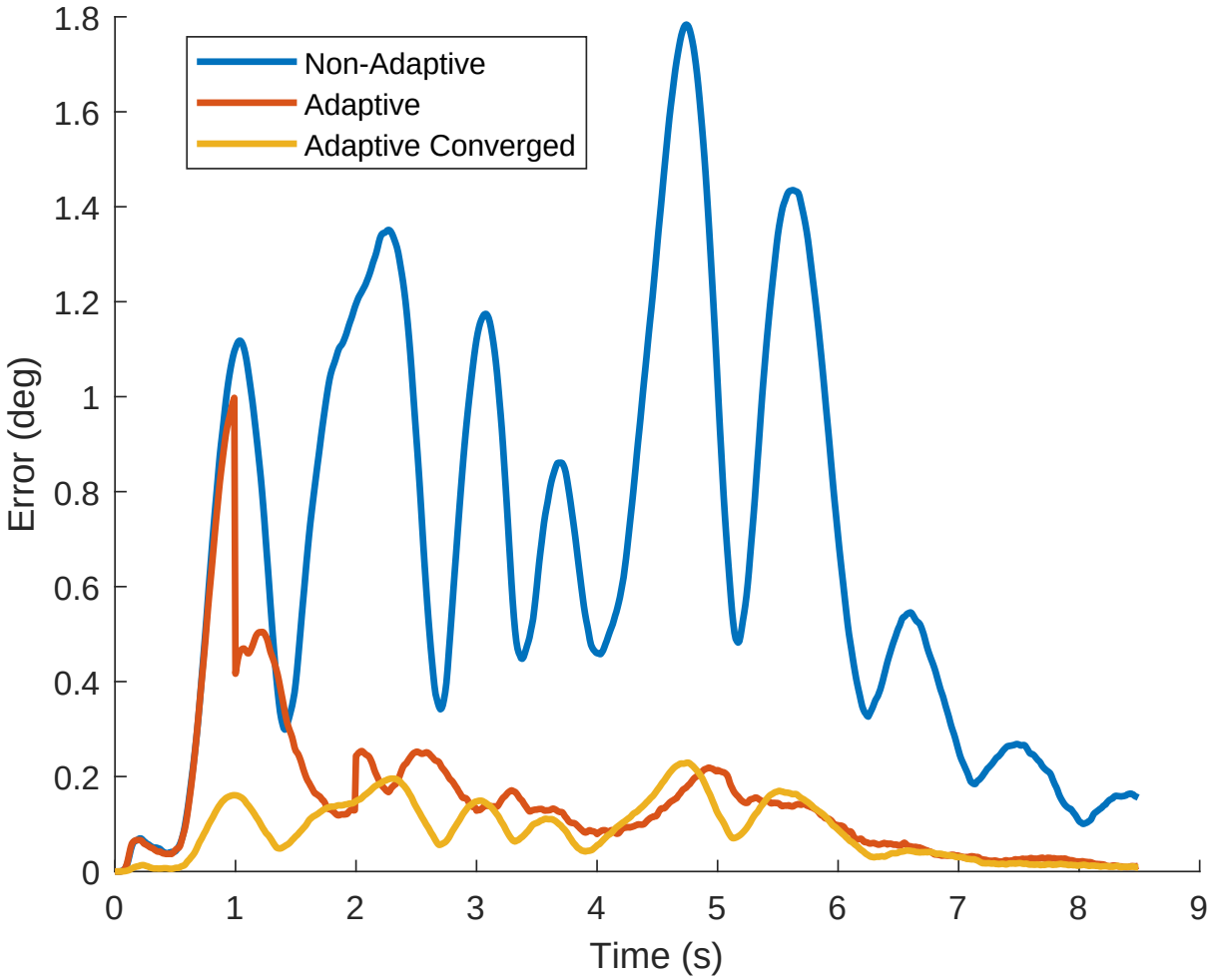


Figure 4.15: State Feedback Controller Steer Angle Prediction Errors, V_x = 10 m/s

where L represents the longitudinal distance to the target waypoint in the vehicle frame. The derivatives of the error states are shown in Equations (4.25) - (4.28),

$$\dot{e_\psi} = \dot{\psi} - \dot{\psi_{des}} \tag{4.25}$$

$$\ddot{e_\psi} = \ddot{\psi} - \ddot{\psi_{des}} \tag{4.26}$$

$$\dot{e_{lat}} = V_y + V_x sin(e_\psi) = V_y + V_x sin(\psi - \psi_{des}) \tag{4.27}$$

$$\ddot{e_{lat}} = \dot{V_y} + V_x \dot{e_\psi} \tag{4.28}$$

assuming a small angle approximation for heading error (e_ψ) and a constant longitudinal velocity (V_x). The error state equations can be combined with the dynamic bicycle model to form the state space model shown in Equation (4.29)

$$\begin{bmatrix} \dot{e}_{lat} \\ \ddot{e}_{lat} \\ \dot{e}_{\psi} \\ \ddot{e}_{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-C_0}{mV_x} & \frac{C_0}{m} & \frac{-C_1}{mV_x} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-C_1}{I_{zz}V_x} & \frac{-C_1}{I_{zz}} & \frac{-C_2}{I_{zz}V_x} \end{bmatrix} \begin{bmatrix} e_{lat} \\ \dot{e}_{lat} \\ e_{\psi} \\ \dot{e}_{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{c_{\alpha,f}}{m} \\ 0 \\ \frac{ac_{\alpha,f}}{I_{zz}} \end{bmatrix} \delta \tag{4.29}$$

where

$$C_0 = C_{\alpha f} + C_{\alpha r} \tag{4.30}$$

$$C_1 = aC_{\alpha f} + bC_{\alpha r} \tag{4.31}$$

$$C_2 = a^2 C_{\alpha f} + b^2 C_{\alpha r} \tag{4.32}$$

This model is used for the non-delayed runs, but was found to be insufficient once the delay was introduced into the system. Because the dynamics introduced by the delayed actuator are not negligable, the MPC controller could not maintain stability when the steering dynamics were not included in the model. For this reason, an augmented model was used when the delay was present in the system as shown in Equation (4.33).

$$\begin{bmatrix} \dot{e}_{lat} \\ \ddot{e}_{lat} \\ \dot{e}_{\psi} \\ \ddot{e}_{\psi} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{-C_0}{mV_x} & \frac{C_0}{m} & \frac{-C_1}{mV_x} & \frac{c_{\alpha,f}}{m} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \frac{-C_1}{I_{zz}V_x} & \frac{-C_1}{I_{zz}} & \frac{-C_2}{I_{zz}V_x} & \frac{ac_{\alpha,f}}{I_{zz}} \\ 0 & 0 & 0 & 0 & -\frac{1}{\tau_\delta} \end{bmatrix} \begin{bmatrix} e_{lat} \\ \dot{e}_{lat} \\ e_{\psi} \\ \dot{e}_{\psi} \\ \delta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \frac{1}{\tau_\delta} \end{bmatrix} d \tag{4.33}$$

For the uncompensated run, the steering time constant (τ_δ) was set to the nominal value given in Section 4.1. For the compensated and adaptive runs, the steering time constant was set to the value designed for in Chapter 3 (≈ 0.075 seconds). The pure delay was not considered in the MPC model.

Once the error state model is derived, it is used to calculate the optimal control input in order to follow the reference path. The state space model is first discretized at the desired sample rate, in this case 100 Hz, to obtain the discrete state transition and input matrices A_d

and B_d, respectively. Next the model is reformulated to represent the change in states (Δx) based on the change in input (Δu) as shown in Equation (4.34).

$$\Delta x_{k+1} = x_{k+1} - x_k = A_d \Delta x_k + B_d \Delta u_k \tag{4.34}$$

The model is then augmented with the measurement of the "output" state, in this case lateral path offset. This is shown in Equations (4.35) - (4.36).

$$\begin{bmatrix} \Delta x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} A_d & 0 \\ C_d A_d & 1 \end{bmatrix} \begin{bmatrix} \Delta x_k \\ y_k \end{bmatrix} + \begin{bmatrix} B_d \\ C_d B_d \end{bmatrix} \Delta u_k \tag{4.35}$$

$$y_k = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x_k \\ y_k \end{bmatrix} \tag{4.36}$$

This augmentation effectively embeds an integrator into the system model, driving the steady state error to zero.

The cost function J used to calculate the optimal control input is shown in Equation (4.37).

$$J = (R_s - Y)^T (R_s - Y) + \Delta U^T \bar{R} \Delta U \tag{4.37}$$

Y and ΔU represent the predicted output over the prediction horizon N_p and the control input over the control horizon N_c, respectively. They are related in Equation (4.38).

$$Y = F x_k + \Phi \Delta U \tag{4.38}$$

where,

$$F = \begin{bmatrix} CA & CA^2 & CA^3 & ... & CA^{NP} \end{bmatrix}^T \tag{4.39}$$

$$\Phi = \begin{bmatrix} CB & 0 & 0 & ... & 0 \\ CAB & CB & 0 & ... & 0 \\ CA^2B & CAB & CB & ... & 0 \\ ... & ... & ... & ... & ... \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & ... & CA^{N_p-N_c}B \end{bmatrix} \tag{4.40}$$

The matrices R_s and $\bar{R}$ are defined in Equation (4.41)

$$R_s = [1 \;\; 1 \;\; ...1]^T_{1xN_p} \, r_k, \quad \bar{R} = r_w I_{N_c x N_c}, \quad r_w = \text{input weighting} \tag{4.41}$$

. R_s defines the setpoint for each sample within the prediction horizon. In this case, since the state being controlled is lateral error, each value in R_s will be 0. $\bar{R}$ is the input weighting matrix that determines the aggressiveness of the controller. Due to the fact that in this thesis the MPC is unconstrained, there exists an analytical solution for the optimal control input. This solution is shown in Equation (4.42).

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (R_s - F x_{ki}) \tag{4.42}$$

The full control sequence ΔU is computed at every timestep, but only the first input of the sequence is used. The sequence is then re-computed at the next timestep.

4.6.2 MPC Results

Because the MPC controller used a model of the steering response to predict future outputs and calculate the optimal control input, it was more sensitive to changes in the steering dynamics than the other controllers. The pure delay had an especially negative affect on the controller performance, which was expected as it was not incorporated into the MPC model. The error statistics for the Monte Carlo run at 10 m/s are shown in Figure 4.16. The compensation

provides a significant reduction in both maximum and mean heading and lateral path error compared to the uncompensated run. The adaptive run also shows improvement over the non-adapitve compensation in maximum heading error and maximum lateral error. The MPC controller, along with the state feedback controller, are the only two outer-loop controllers where a noticeable improvement was provided by the adaptive algorithm compared to the non-adaptive run. The state feedback and MPC controllers were expected to show similar results, although MPC was expected to perform better because of the incorporation of steering dynamics into the model. For Sims 1, 3 and 4 the two controllers performed as expected, with the MPC controller providing slightly lower mean and maximum heading and lateral errors for each of these runs. For Sim 2, however, the state feedback controller had lower maximum errors. This is likely due to the sensitivity of the MPC to inaccuracies in the steering model. When the steering dynamics differed significantly from their nominal values this led to the MPC controller performing poorly and likely led to high maximum errors on some runs. The mean errors of the MPC controller were very similar to those of the state feedback controller and the other controllers tested. Run 5 again showed the best performance of the simulations with delay across all three error metrics, as expected. The maximum and mean steer angle errors are reduced again in

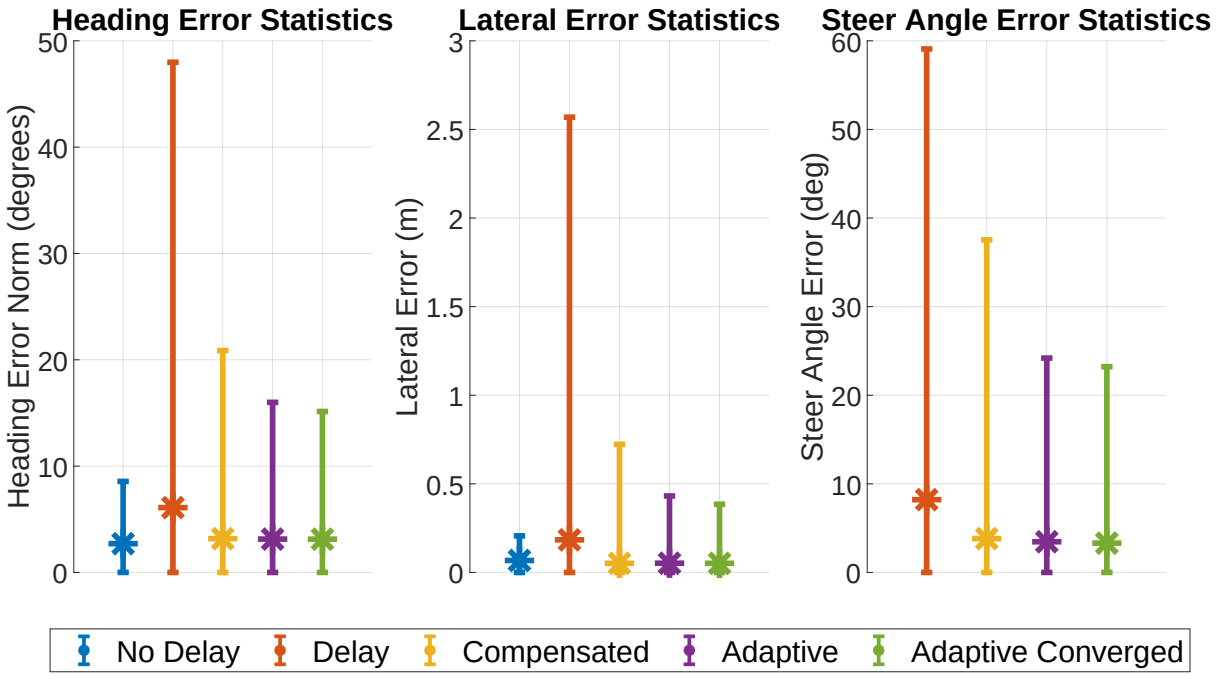


Figure 4.16: MPC Double Lane Change Error Statistics V_x = 10 m/s

the compensated and adaptive runs. The MPC controller exhibits higher mean and maximum steer angle errors than the other controllers due to the more aggressive steer angle commands calculated by the controller. The actuator struggles to track these commands, especially in the uncompensated runs. The compensation is able to provide a significant improvement in both mean and maximum steer angle error, and the adaptive run shows noticably lower maximum error and slightly lower mean error. This shows that with the SP compensation included, the system is better able to track the desired steer angles output by the MPC controller, which leads to the improved path tracking performance in the compensated and adaptive runs. The "Adaptive Converged" run shows the best tracking of desired steer angle, as expected. The

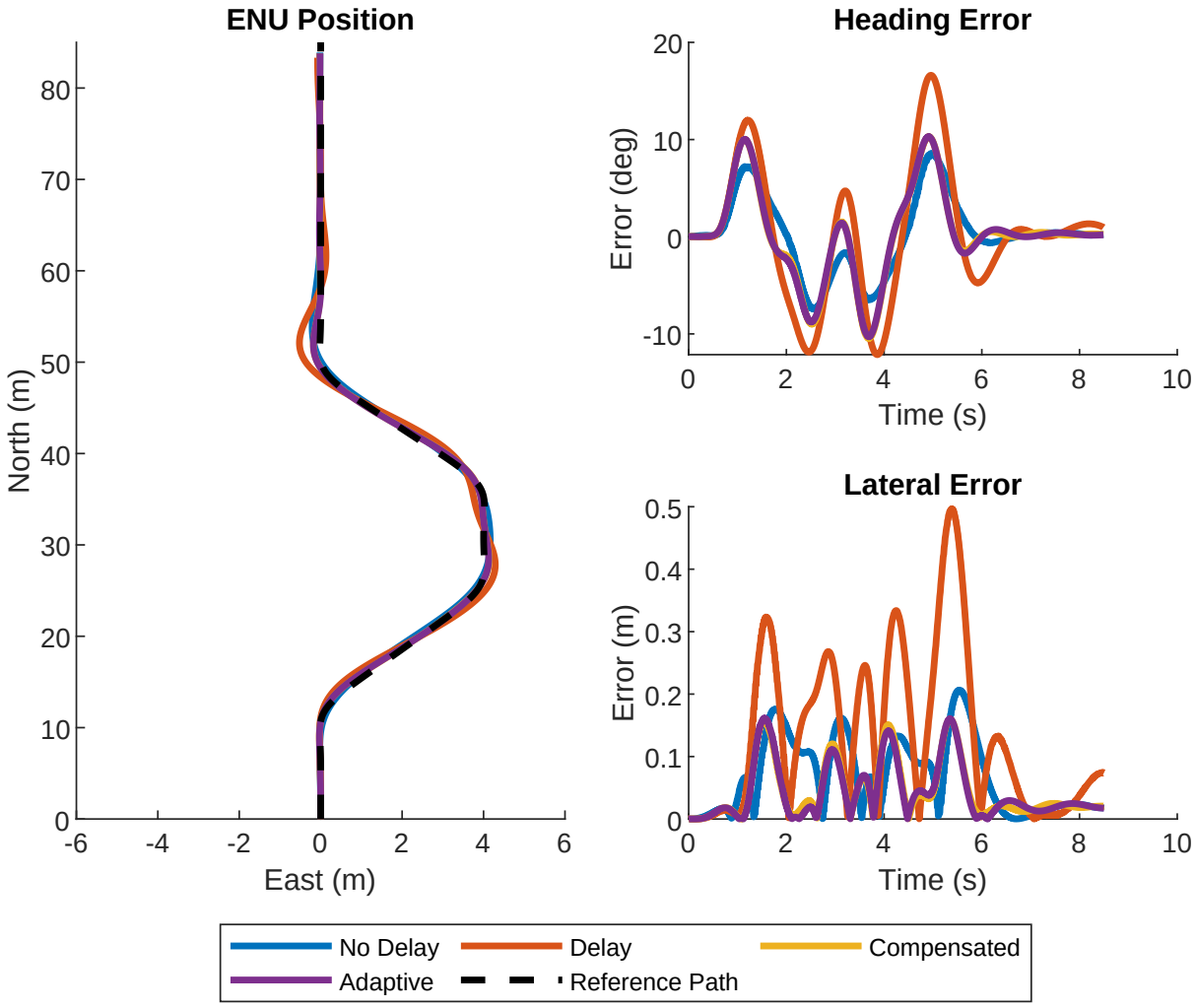


Figure 4.17: MPC Double Lane Change Performance, V_x = 10 m/s

system performance for each of the 10 m/s runs are compared in Figure 4.17. The controller is able to track the path relatively well, even for the uncompensated run. The SP compensation is

able to reduce the heading and lateral errors almost back to the values of the control run. The non-delayed run has one of the lowest average lateral errors of the four outer-loop controllers, staying within 0.2 meters of the path, on average, throughout the run. This was expected as MPC computes the optimal control input and should display very good path tracking performance when an accurate model is used. The Compensated and Adaptive runs were also able to stay within 0.2 meters of the path for the majority of the runs.

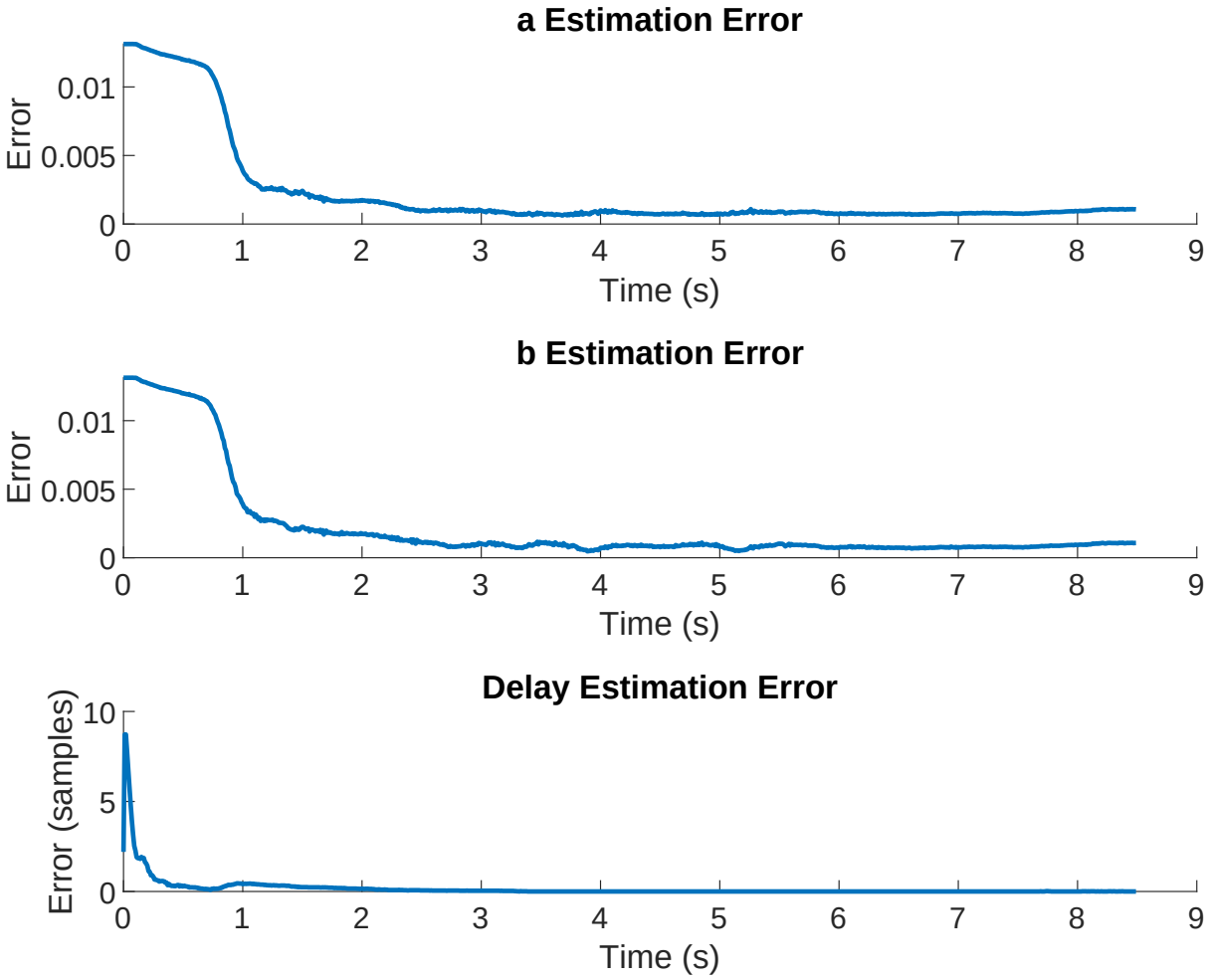


Figure 4.18: MPC Parameter Estimation Errors, V_x = 10 m/s

The parameter estimation algorithm showed similar results to the previous controllers for the MPC test as seen in Figure 4.18. The dynamic parameter estimate errors converged to under 0.005 on average, and the pure delay estimation errors converged nearly to zero by the end of the run. The parameter estimates converged more quickly than with the previous controllers, likely due to the more aggressive control inputs from the MPC, creating more excitation for the adaptation. This also explains the adaptive algorithm providing a noticeable improvement,

as the parameters converged quicker and the SP model was updated with accurate parameters earlier in the run. Once the SP model converged after 1 second, the prediction error quickly dropped below 0.2 degrees, on average, and stayed there for the rest of the run as shown in Figure 4.19. This improvement in prediction error is responsible for the reduction in maximum error in the adaptive run compared to the non-adaptive compensation. The more noticeable improvement from the adaptive run is also due to the higher prediction error during the non-adaptive run compared to the other outer-loop controllers. Run 5 shows average prediction error under 0.2 degrees for the entirety of the run, leading to better overall control performance.

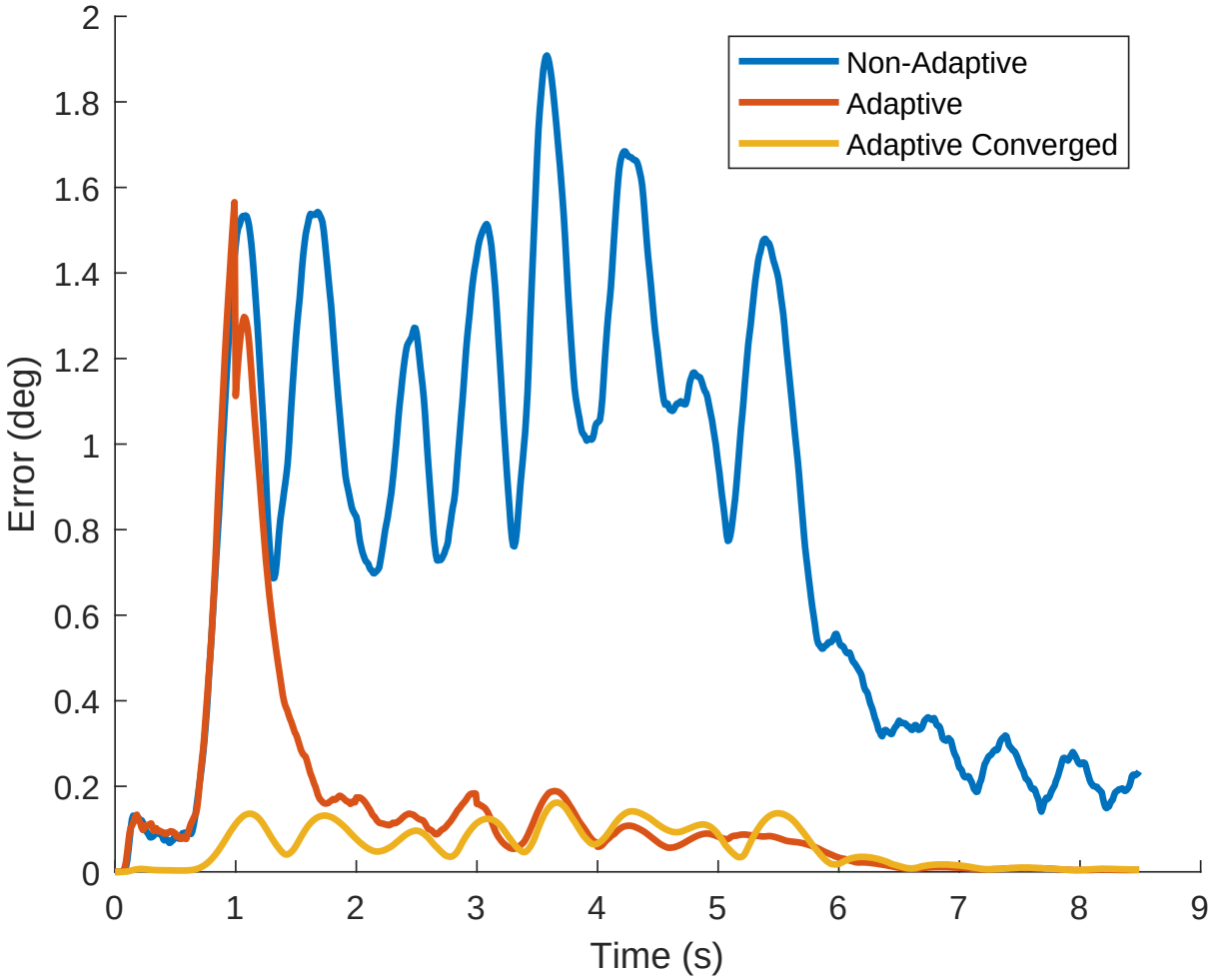


Figure 4.19: MPC Steer Angle Prediction Errors, V_x = 10 m/s

4.7 Comparison of Compensation Algorithm on Various Outer-Loop Controllers

The compensation algorithm was tested with four different outer-loop controllers in a path following simulation in order to show the improvement provided by both adaptive and non-adaptive compensation. Both inner-loop algorithms were able to provide improvement in heading error, lateral error, and overall path following performance for each of the controllers. The improvements were due only to the introduction of the inner-loop, as the outer-loop controllers were not re-tuned when delay was introduced, with the exception of the MPC controller where a steering model had to be included to maintain stability.

The discrete heading controller was one of the simplest controllers tested. Because the controller only considered desired heading, it had a tendency to undershoot or overshoot the path and required tuning of the look-ahead distance to achieve desirable path tracking performance. When delay was introduced, the discrete heading controller displayed increased overshoot and oscillation, but the compensation algorithm improved the performance to acceptable levels by speeding up the steering response.

The pure pursuit kinematic controller was also very simple to implement, and did not use any feedback in the calculation of the desired steer angle. It showed very similar results to the heading controller, with slightly higher maximum errors when delay was introduced. The compensation provided significant improvement for the pure pursuit controller, as expected. The adaptive algorithm again did not provide noticeable improvement over the non-adaptive algorithm for this controller, but again was able to converge to an accurate model.

The state feedback controller was the first controller tested where the calculated control input was based on a dynamic model of the vehicle. Due to the fact that the steering dynamics were not included in the model, the state feedback controller had higher errors than either of the previous two controllers when delay was introduced into the system. The compensation was again able to provide improvement in error and overall path tracking, but the improvement was less drastic for the state feedback controller, especially in regard to the maximum heading and lateral errors. The adaptive algorithm for this outer-loop controller did provide noticably lower maximum heading and lateral errors, in contrast to the discrete heading and pure pursuit

controllers. This is likely due to the more aggressive control inputs provided by the state feedback controller. This led to higher prediction error for the non-adaptive algorithm and a more noticeable improvement from the adaptive algorithm.

The MPC controller was the most complex controller tested and also the most sensitive to the introduction of delay. When the delay was not included in the MPC model but was present in the system, the controller struggled to maintain stability. Even when the delay was accounted for in the model, if the modeled delay did not match the true delay it led to large errors. This can be seen in the maximum heading and lateral position errors for the uncompensated run, which are significantly higher than any of the previous outer-loop controllers, despite the mean errors being very similar to the other controllers. This shows the sensitivity of MPC to inaccuracies in the actuator model. MPC was also found to be sensitive to pure delay, as there is no way to account for it in the model. The SP compensation was able to improve the average MPC performance, although the maximum errors for both the compensated and adaptive runs were higher than those of the heading and pure pursuit controllers. On average, he parameter estimate errors converged faster than in any of the previous tests, likely due to the greater excitation from more aggresive control inputs. This quick convergence of parameter estimates and the higher prediction error in the non-adaptive run led to the most noticeable improvement provided by the adaptive algorithm for any of the outer-loop controllers.

Perhaps the best indication of the improvement provided by both the non-adaptive and the adaptive inner-loop SP compensation is the reduction in steer angle error. The compensated and adaptive runs for all four controllers showed reduced steer angle error, meaning they were better able to track the desired steer angle commanded by the outer-loop controller. For any well tuned path following controller, this should lead to better overall performance.

Table 4.2 shows the improvement in mean error provided by the Compensated (non-adaptive) and "Adaptive Converged" Sims (3 and 5) compared to the uncompensated performance. The last column shows the improvement in prediction error from run 3 to run 5 for each controller. Both the adaptive and non-adaptive compensation is able to improve all of the mean errors for each of the controllers. Sim 5 shows a greater improvement for each of the controllers except for the kinematic controller, where the improvement in heading and lateral

error is slightly less than that of the non-adaptive controller. The reason for this is unknown, as the better tracking of desired steer angle and better predictions should have led to better overall control performance. The mean prediction error for each of the controllers was improved in Sim 5. While Sim 5 did not provide greater improvement than Sim 3 in every error metric, it did always improve the mean steer angle error, showing that the more accurate model used for predictions led to better tracking of the desired steer angles.

Table 4.2: Monte Carlo Error Improvements

Controller	**Mean Heading Error *Improvement* (Degrees)** Non-Adaptive / Adaptive	**Mean Steer Angle Error *Improvement* (Degrees)** Non-Adaptive / Adaptive	**Mean Lateral Error *Improvement* (Meters)** Non-Adaptive / Adaptive	**Mean Prediction Error *Improvement* (Degrees)** Adaptive
Heading Controller	1.15 / 1.15	1.54 / 1.60	0.08 / 0.08	0.36
Kinematic Controller	1.24 / 1.22	1.73 / 1.74	0.12 / 0.11	0.50
State Feedback	2.95 / 3.07	3.73 / 4.00	0.20 / 0.20	0.58
MPC	2.92 / 2.96	4.39 / 4.91	0.13 / 0.13	0.73

Chapter 5

Gazebo Simulation

Gazebo was used to perform software-in-the-loop (SIL) testing in order to ensure that the algorithm would perform well before being implemented on the test vehicle in real time. Additionally, SIL testing allows for testing at a wider range of speeds and under more realistic driving conditions than real time testing. This chapter describes the mode, environment, setup, and results of the Gazebo experiment.

5.1 Simulation Setup

Dataspeed provides a Gazebo model of a Lincoln MKZ that interfaces with ROS and nearly matches the software structure of the actual test vehicle. This allows for the control algorithm to be tested in simulation and then transitioned to real time implementation with minimal modifications. A lanekeeping test was conducted at various speeds using a test track environment provided in the Dataspeed Gazebo simulation package. The test track contains turns of varying radii as well as straightaways and lane-changes. This allows for the algorithm to be tested on a variety of common driving maneuvers.

The simulated MKZ includes a front facing camera that is used to track the lane lines and determine a desired path. The Dataspeed lane following node subscribes to the simulated camera output and determines the center-point of the lane. It then publishes the desired path in the vehicle frame. The simulation includes a graphic user interface (GUI) that shows the simulated camera's point of view as well as the extracted lane-lines and the desired vehicle path. This GUI is shown in Figure 5.1.

During the tests, the vehicle's speed was held constant by a built-in longitudinal controller provided in the simulation package. Runs were conducted at speeds of 10, 15, and 20 meters per second. The outer-loop path following controller was the discrete heading controller described in Section 4.3, and the inner-loop compensation was inserted between the heading controller and the simulated steering system.

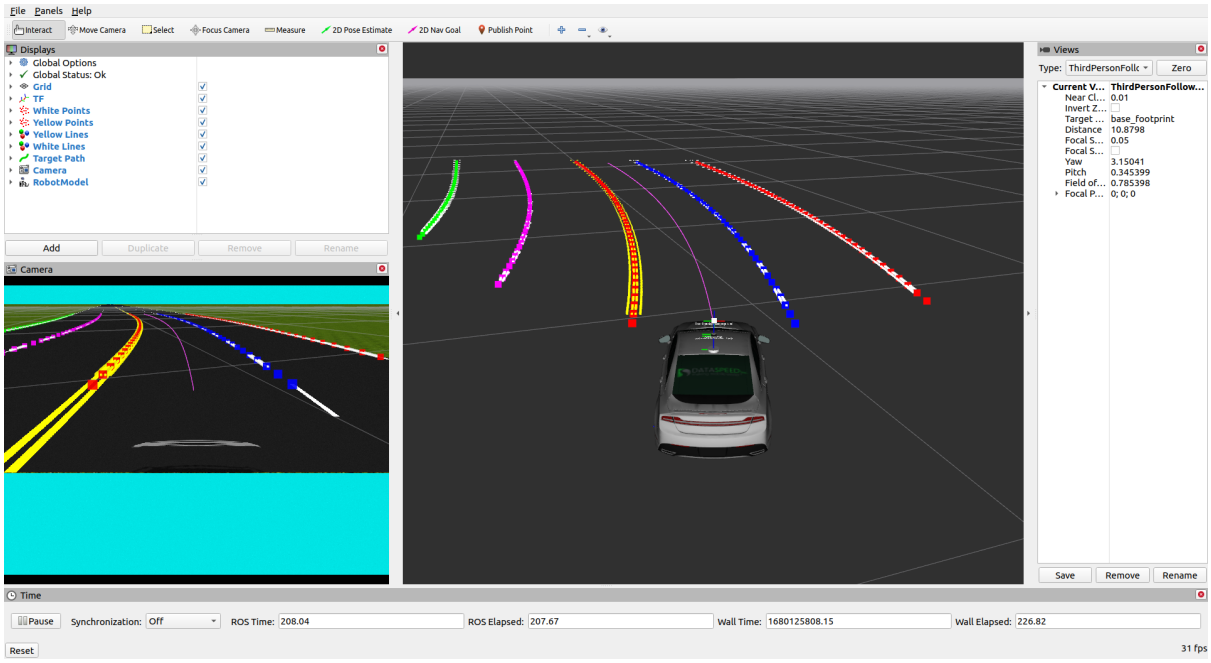


Figure 5.1: Gazebo Simulation GUI

The software structure of the Gazebo simulation is shown in Figure 5.2. The waypoint manager, `/path_following_node` takes the desired path, labeled `/target_path`, from the lane-tracking node and calculates a heading error. This heading error is fed into the `outer_loop` node, where the heading controller calculates a desired steer angle. This steer angle is either fed directly into the simulated steering system or into the `inner_loop` node, depending on the run. If active, the inner-loop calculates a new steer angle and publishes it to the `/vehicle/steering_cmd` topic, which is sent through the simulated CAN bus and to the steering system. The parameter estimation algorithm is updated within the `inner_loop` node, and utilizes commanded steer angle and measured steer angle from the `/vehicle/steering_report` topic. The longitudinal velocity is controlled by the `/vehicle/ulc_node`.

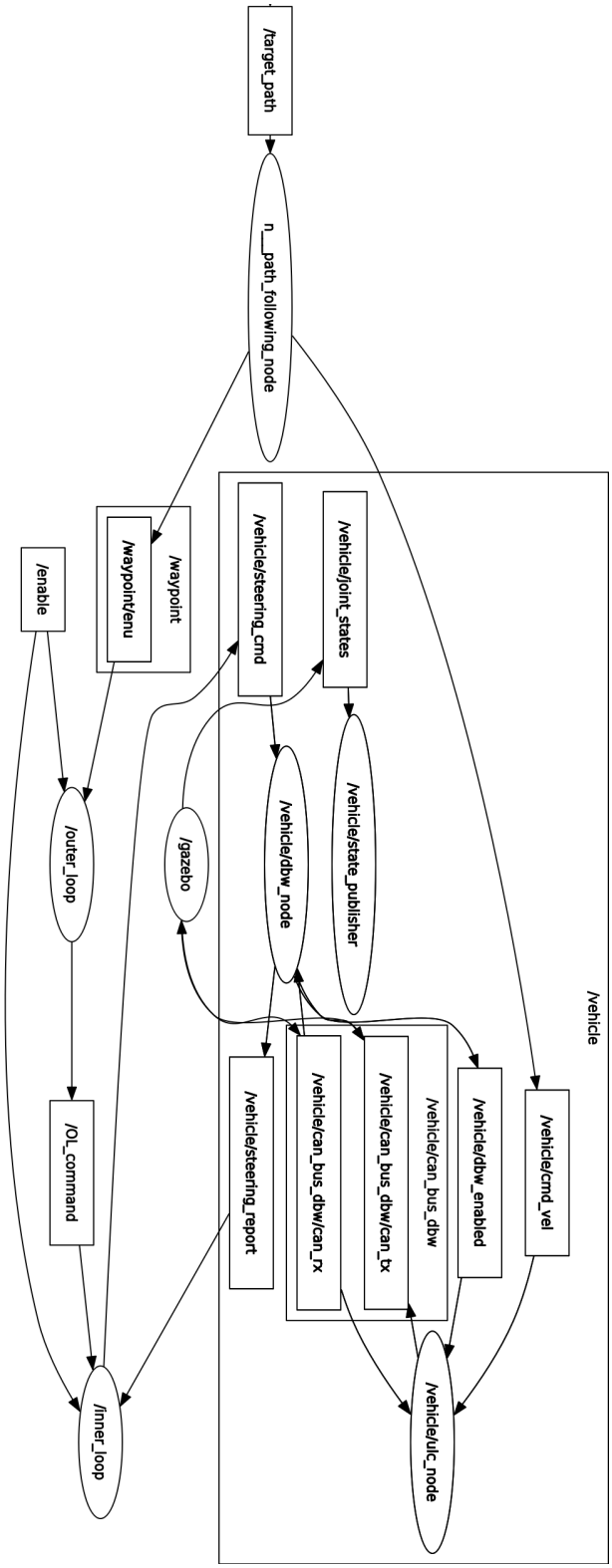


Figure 5.2: Gazebo Simulation Software Architecture

For each speed tested, one lap was run for each of the following configurations.

- Control - Inner-loop not active. Steering commands passed directly from outer-loop to steering system
- Compensated - Inner-loop active with static steering model. No adaptation of SP.
- Adaptive - Full adaptive SP inner-loop algorithm active

The Gazebo MKZ model includes steering dynamics built into the simulation. The response of the simulated steering wheel for a portion of one of the control runs is shown in Figure 5.3, compared to the modeled response used to initialize the Smith predictor. The modeled response is shown to closely follow the measured response, however the outer-loop steering command is shown to be very oscillatory. Due to this, a low-pass filter was inserted between the outer and inner loops to obtain a smoother desired steer angle.

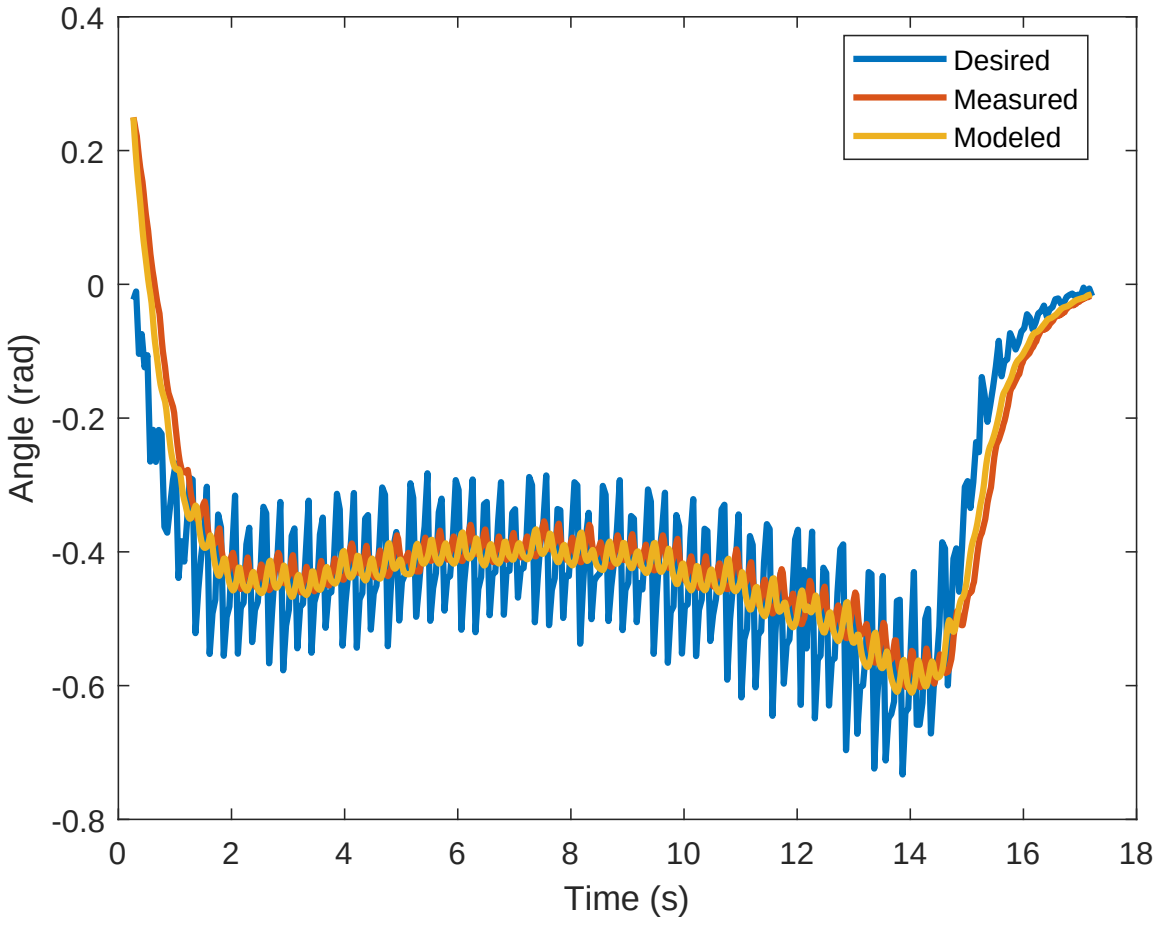


Figure 5.3: MKZ Gazebo Simulation Steering Response

The path of the simulated Gazebo track is shown in Figure 5.4, plotted in the local ENU frame. The startpoint of the path is denoted with a star.

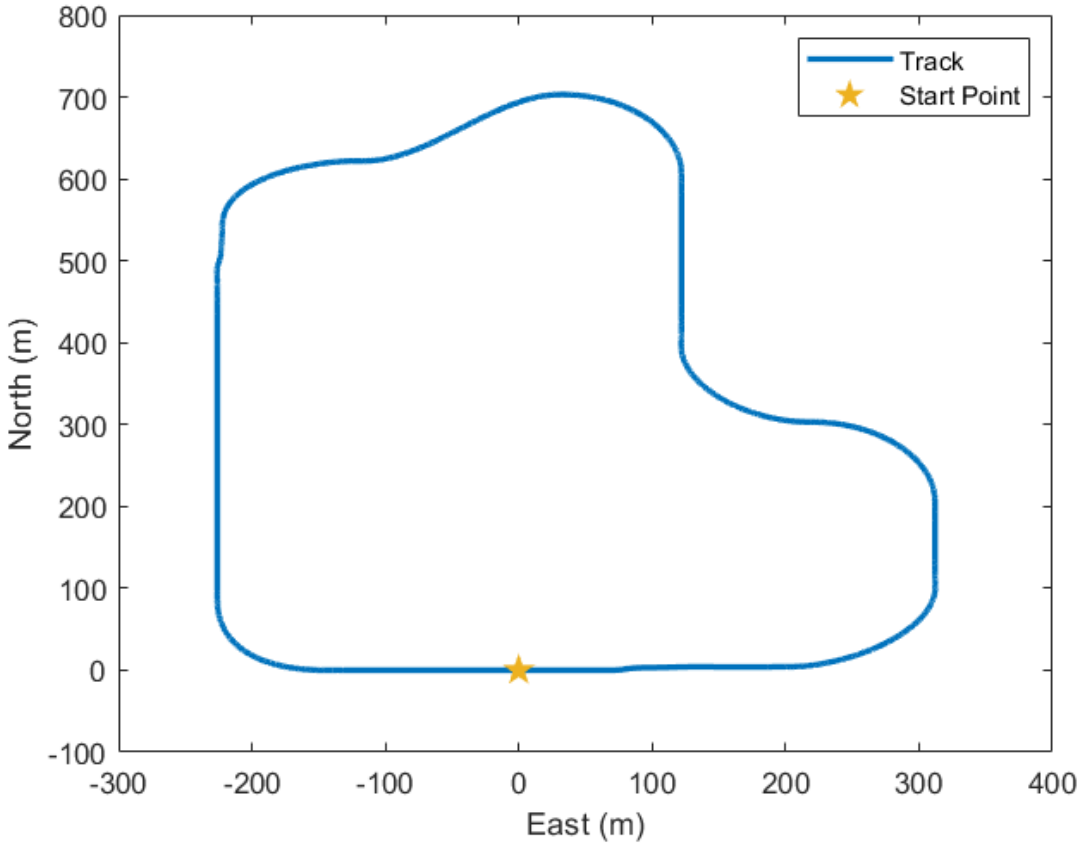


Figure 5.4: Simulated Test Track Path

5.2 Results

The heading errors for the control, compensated, and adaptive runs at each of the three speeds are plotted versus time in Figure 5.5. The error is shown to be higher and more oscillatory for the control run compared to the compensated and adaptive runs, especially during the faster simulation runs. The adaptive run also performs noticeably better than the compensated run. Again, this improvement is more apparent in the faster simulation runs.

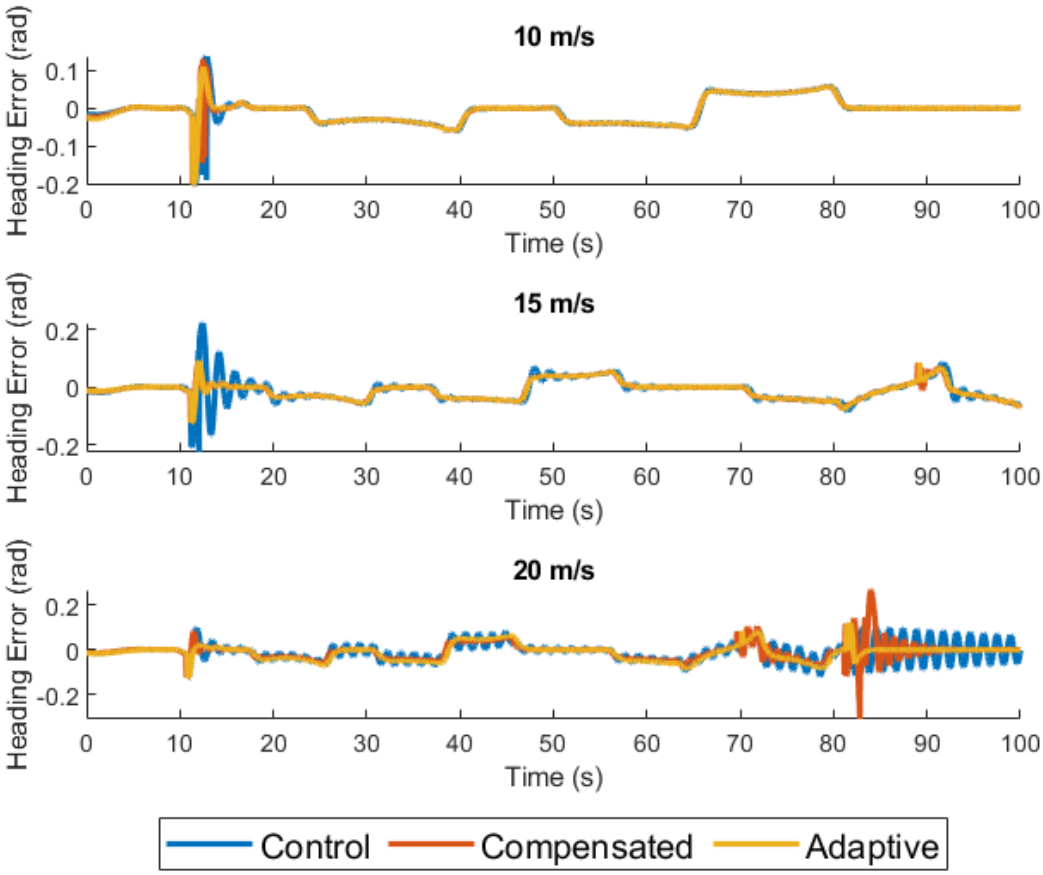


Figure 5.5: Heading Error Versus Time for Gazebo Simulation at Various Speeds

The steering angle is plotted versus time for each of the runs in Figure 5.6. The uncompensated (control) run is again shown to display more oscillatory steering response than the compensated and adaptive runs, especially at higher speeds. This was expected as the uncompensated system has more difficulty tracking the desired maneuvers as longitudinal speed increases.

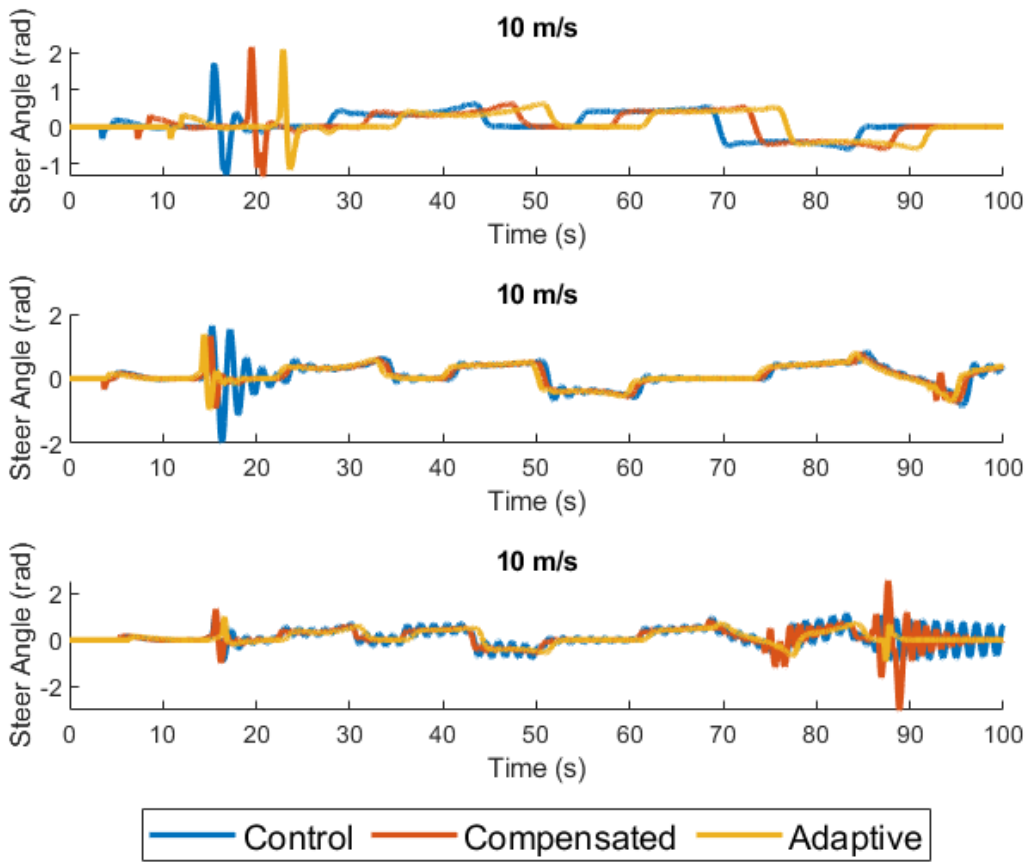


Figure 5.6: Steering Angle Versus Time for Gazebo Simulation at Various Speeds

The heading error statistics for each run are shown in Figure 5.7. Note that the burs represent the maximum and minimum errors while the asterisks represent the mean errors. The compensated and adaptive runs are shown to reduce both mean and maximum error for the 15 m/s run. There is little improvement shown for the 10 m/s run, likely because the dynamics are slow enough at that speed that the uncompensated controller is able to track them without excess error. In the 20 m/s lap, the mean errors are reduced but the compensated run has a higher maximum error than both the adaptive and control runs.

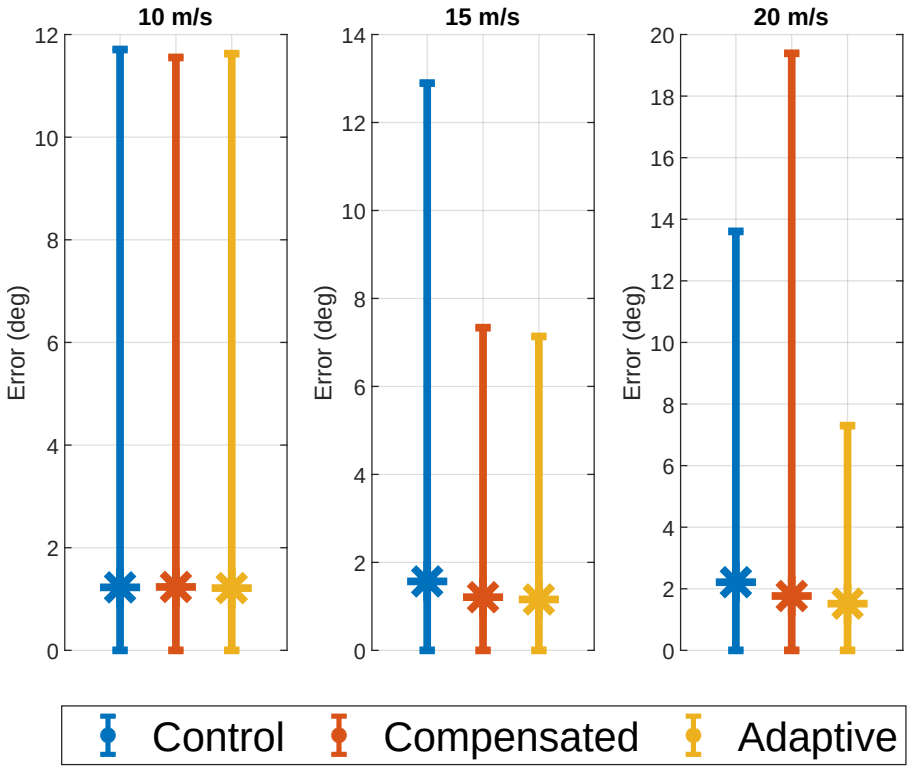


Figure 5.7: Heading Error Statistics for Gazebo Simulation at Various Speeds

The steer angle error, calculated as the difference in the filtered outer-loop command and the current measured steer angle, is shown in Figure 5.8. For the 10 m/s lap, the mean and maximum errors for each of the three runs are similar. There is a small improvement in mean error, shown again by the asterisk, from both the compensated and adaptive runs, while the maximum error for the compensated run is higher than the control run. This is likely due to prediction error spikes, which can lead to control errors. The 15 m/s run shows an improvement in both mean and maximum errors for the compensated and adaptive runs. The 20 m/s test shows interesting results. The compensated lap has a lower mean error than the control but a very similar maximum error. The adaptive run shows a large improvement in maximum error but performs slightly worse in terms of mean error. For all three speeds, the compensated laps show higher maximum steering error than the adaptive runs. This is likely due to larger prediction errors when the steering model is not adapted with the new parameter estimates.

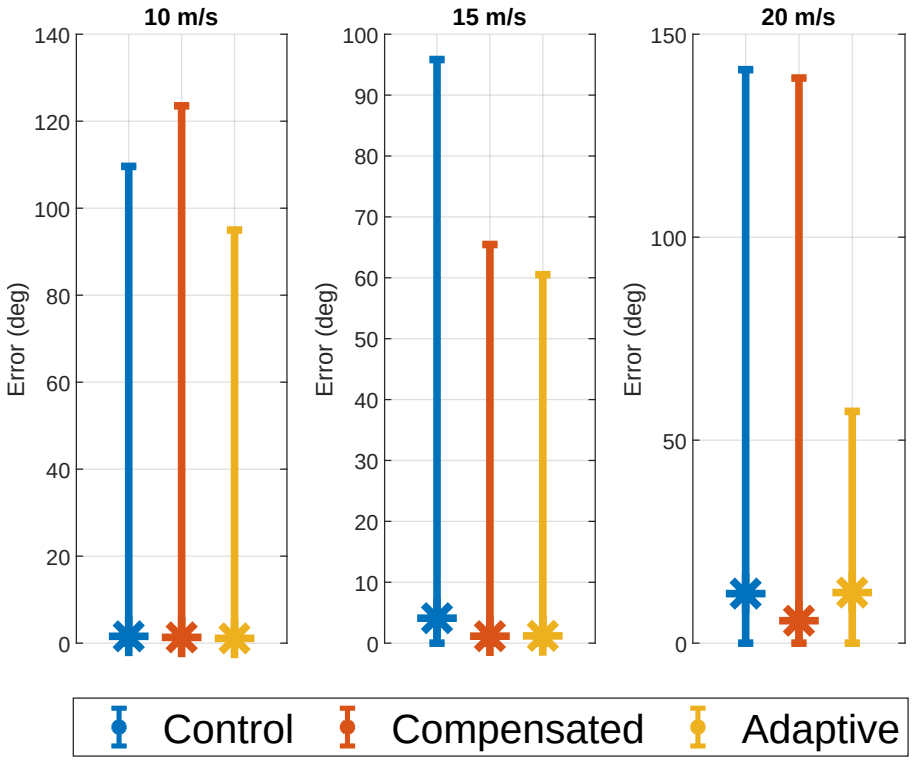


Figure 5.8: Steer Angle Error Statistics for Gazebo Simulation at Various Speeds

The prediction errors for the compensated and adaptive runs at each speed are compared in Figure 5.9. The adaptive algorithm provides significant improvement in maximum prediction error for each speed, with the most noticeable difference occuring during the 20 m/s lap. The mean error is also reduced slightly for each of the runs. The adaptive algorithm proves to be valuable in minimizing prediction error and is likely responsible for the lower maximum heading and steer angle errors during most of the adaptive tests.

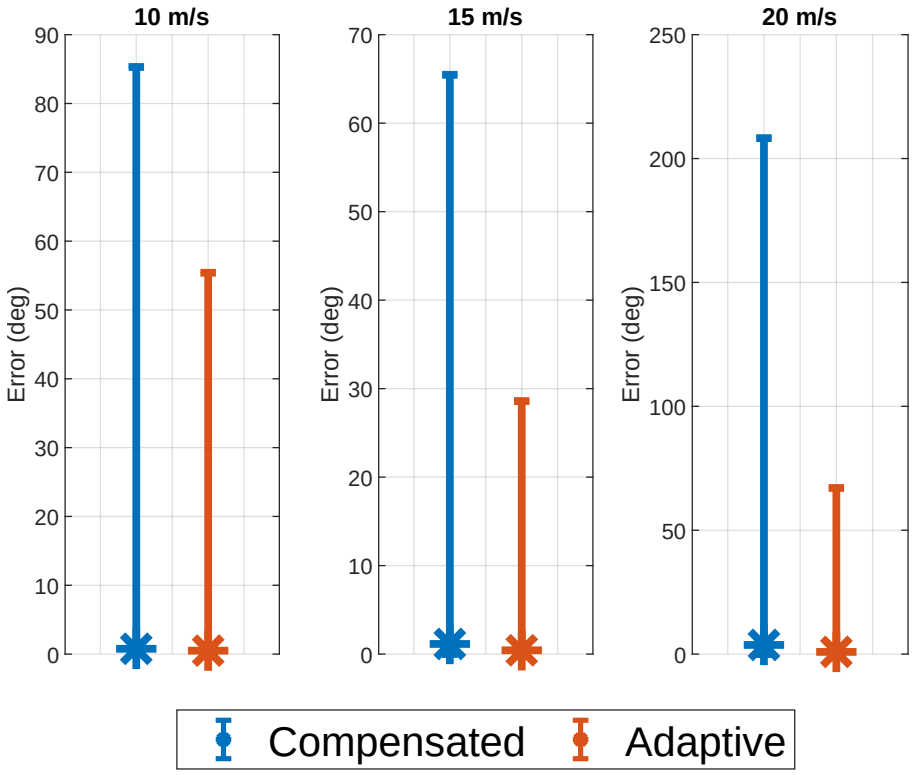


Figure 5.9: Prediction Error Statistics for Gazebo Simulation at Various Speeds

The Gazebo simulation shows that the algorithm is able to provide improvement in a lane-keeping control system, again lowering heading and steer angle error for most runs. The parameter estimation algorithm was again able to converge to an accurate model and to provide improved prediction error over the non-adaptive runs. The algorithm provided the most noticeable improvements on the 20 m/s runs. This was expected, as an increased velocity means higher dynamic maneuvers must be performed to maintain the desired path.

Chapter 6

Real-World Experiment and Results

In order to examine the performance of the compensation algorithm on real steering actuators, multiple tests were performed using three different vehicles. First, the compensation was tested in a path following experiment using a Lincoln MKZ outfitted with DBW hardware. This will show the performance improvements provided by the algorithm in a dynamic, real-time test. Next, the parameter estimation algorithm was validated on data from the steering actuators of two additional vehicles: an autonomous IndyCar and a Peterbilt 579 tractor trailer. The results of each of these tests will be presented and discussed in this chapter.

6.1 MKZ Description and Software Architecture

The first test vehicle used for experimental verification was a Lincoln MKZ outfitted with a Dataspeed drive-by-wire (DBW) kit (shown in Figure 6.1). The testing was done on a skid pad at Auburn University's National Center for Asphalt Testing (NCAT). The DBW hardware interfaces with the vehicle's CAN bus and the existing steering motor to provide steering actuation. Due to the multiple software levels the steering command must travel through to reach the steering wheel, there exists a non-negligable communication delay between the control computer and the steering system. Additionally, the steering module which controls the steering actuator has relatively slow dynamics, as shown in Figure 4.1. A diagram of the software architecture from the control computer to the steering actuator is shown in Figure 6.2 (adapted from [52]).

The steering command must pass from the control computer, through two Robot Operating System (ROS) drivers, into the USB-CAN tool, and to the Dataspeed steering module, which sends it over the vehicle's CAN network to the built-in steering system within the MKZ. The

Figure 6.1: MKZ Test Vehicle

sensosrs used on the MKZ include a fused GPS-INS system, an encoder, measuring steering wheel angle, an IMU, providing acceleration and angular velocity measurements, and wheel speed encoders, providing vehicle longitudinal speed.

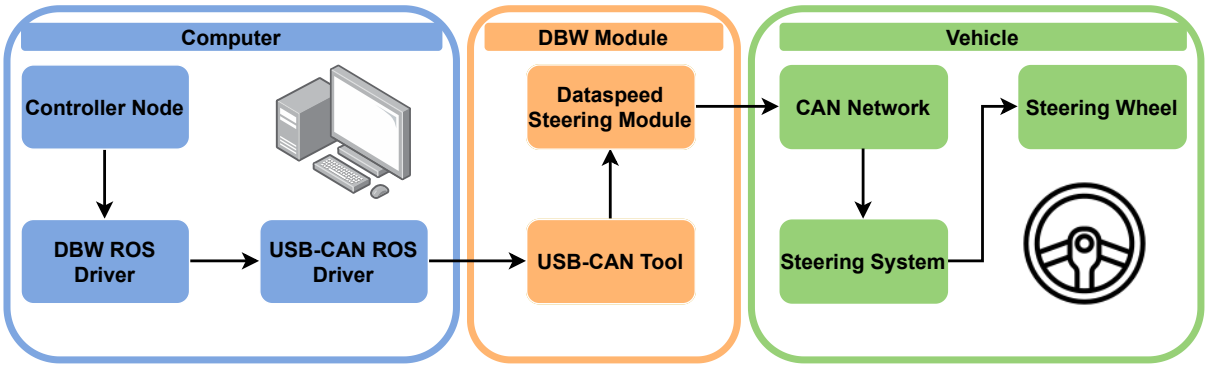


Figure 6.2: MKZ DBW Software/Hardware Architecture

6.2 MKZ Static Actuator Test

To ensure that the compensation algorithm was performing as expected before conducting dynamic testing, static data was taken, comparing the uncompensated and compensated performance of the actuator. A sine wave with a frequency of 1.5 rad/s and an amplitude of 3 radians ($\approx$ 172 degrees) was used as the outer-loop desired steer angle for the first test. The compensated and adaptive algorithms were tested and their responses were compared to that of the uncompensated actuator. The plots of desired and measured steer angle versus time are shown

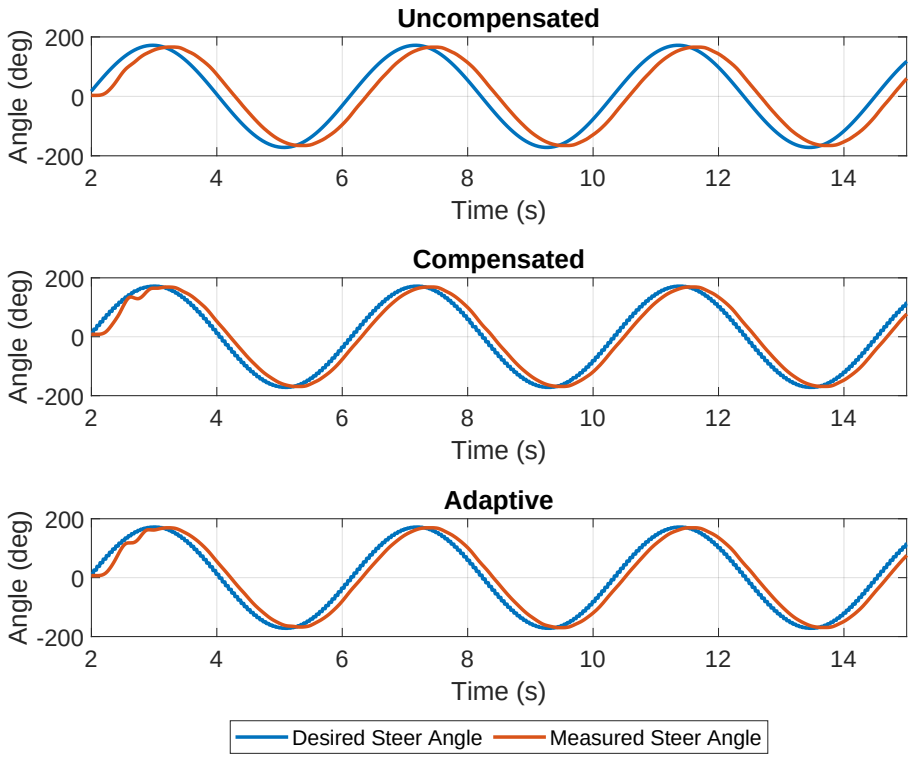


Figure 6.3: Static Actuator Response to Sinusoidal Input

in Figure 6.4. Both the compensated and adaptive responses are shown to more closely match the desired response, exhibiting less phase lag.

The inner-loop commands for the compensated and adaptive tests are compared to the outer-loop commands in Figure 6.4. The inner-loop commands are shown to lead the outer-loop for both runs. This was expected, as the Smith predictor (SP) provides a phase lead in order to reduce the effects of pure delay. The steer angle prediction error for the compensated and adaptive runs are compared in Figure 6.5. Once the parameter estimates converge around the 5 second mark, the error in the adaptive test stays below that of the non-adaptive test.

The static actuator test demostrates that the inner-loop compensation is working as expected. It provides closer tracking of a sinusoidal reference for both the non-adaptive and adaptive tests. Additionally the adaptive algorithm exhibits lower prediction errors than the non-adaptive test, showing that the parameter estimates have converged to a more accurate model. The control performance remains almost identical for the non-adaptive and adaptive runs, which was expected as the non-adaptive run was initialized with a model known to be

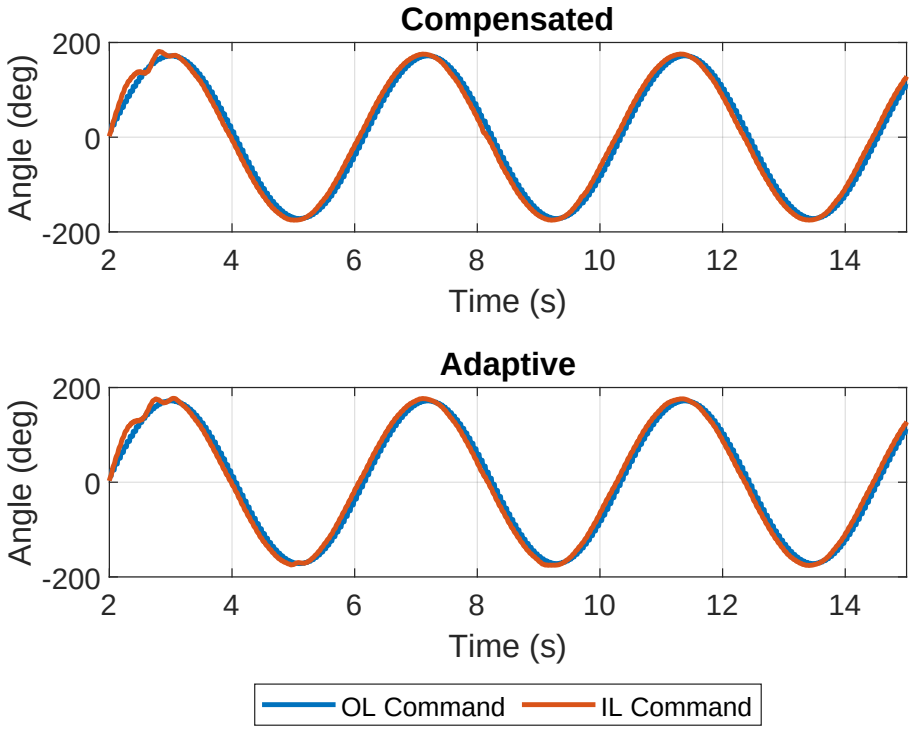


Figure 6.4: Outer and Inner Loop Steering Commands for Static Test

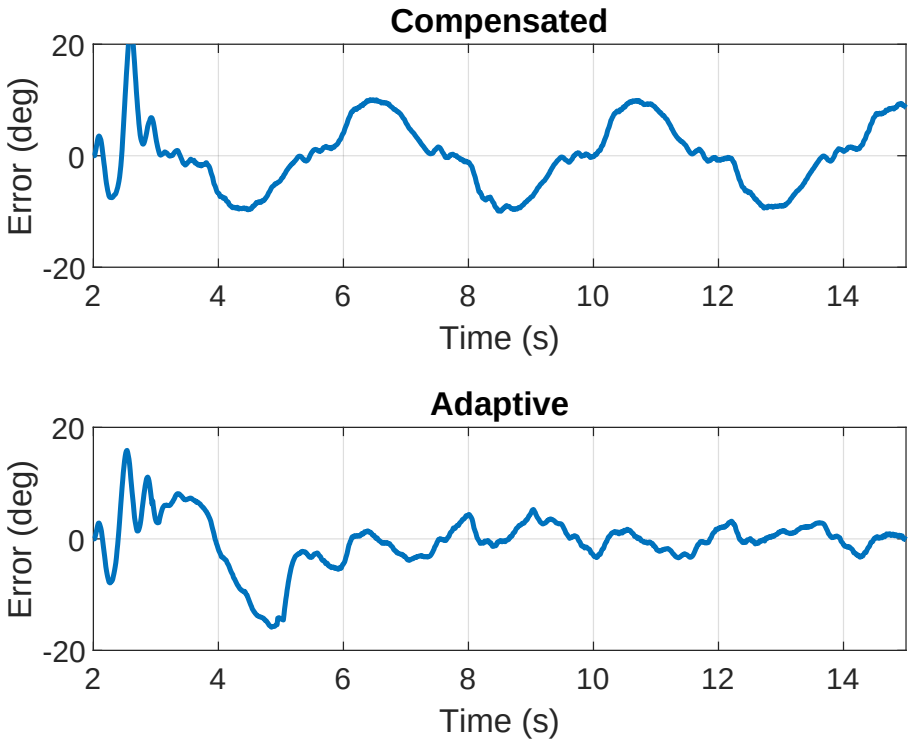


Figure 6.5: Prediction Error for Static Test

relatively accurate. Since the algorithm performed well in static tests, it was then applied to a real-time path following scenario, described in the following section.

6.3 MKZ Waypoint Following Experiment

6.3.1 Experiment Description

To determine the efficacy of the inner-loop compensation algorithm in a real-time dynamic implementation, an experiment was conducted in a waypoint following architecture. First, the MKZ was driven manually to record a desired path on the skid pad at NCAT. Waypoints were recorded from an on-board GPS-INS system with high run-to-run accuracy. The points were then rotated from the ECEF frame into the local NED frame. Once the path was recorded, the vehicle was returned to the starting point and the DBW system was enabled. The outer-loop controller used for the tests was a discrete heading controller, as described in Section 4.3. A block diagram of the control architecture used for the experiment is shown in Figure 6.6. The heading controller was chosen in the outer-loop because it is simple and relatively easy to tune, but any path tracking controller could have been used in this architecture.

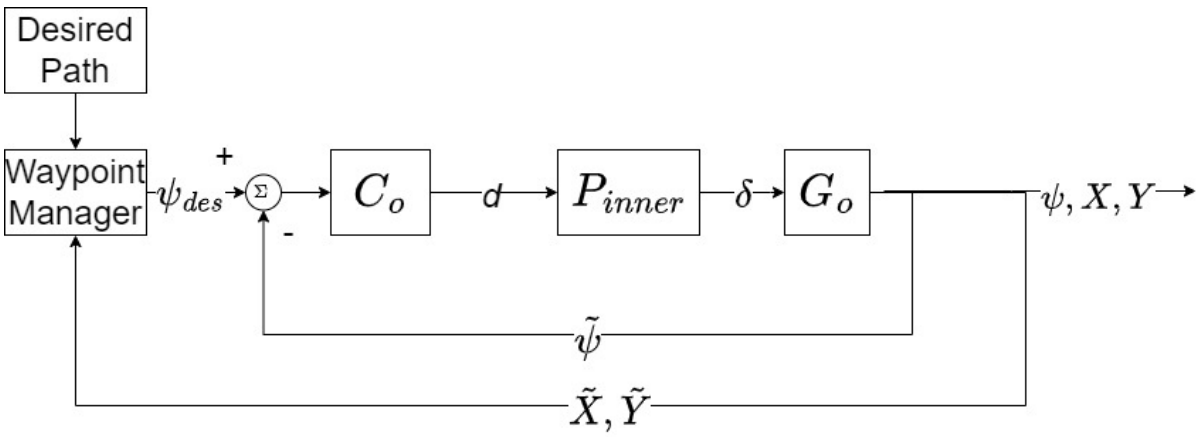


Figure 6.6: Control Architecture For MKZ Test

The desired path was fed into a waypoint manager, which selected a desired waypoint based on a the look-ahead distance. The look-ahead distance used for each test was 12 meters. The heading error from the vehicle to the desired waypoint was then calculated using the vehicle's current heading and position and the waypoint's position in the NED frame. This calculated heading error was fed into the outer-loop controller, which calculated a desired steer

angle. This steering command was either fed directly to the actuator or into the inner-loop, depending on the type of test. The outer-loop was run at 20 Hz and the inner-loop was run at 100 Hz.

Due to noise in the desired waypoint and current position measurements, the heading error calcualted by the waypoint manager was relatively noisy. This was amplified by the outer-loop heading controller. Because of this, a second order low-pass filter with a bandwidth of 15 rad/s was implemented between the inner and outer loops. In other words, the steering commands passed from the heading controller were filtered before passing into the inner-loop controller. This helped to reduce oscillation in commanded steer angle and to provide a smoother desired steer angle for the inner-loop to track. The filter reduces the path-following bandwidth, but the inner-loop compensation is still able to improve performance. This filtering would not be necessary if the outer-loop controller provided smoother steering commands.

The throttle was controlled manually and the vehicle speed was maintained around 5 to 10 meters per second, depending on the maneuver. Higher speeds were not tested due to the space limitations of the skid pad. Multiple runs were recorded and compared to show the effects of the inner-loop compensation. The individual tests are described below:

Test 1. Control Run: The outer-loop steering command is passed directly to the steering system. No inner-loop compensation.

Test 2. Compensated Run: The inner-loop compensation is activated, using a model obtained from prior steering response data. The model is not adapted during this run.

Test 3. Adaptive Compensation Run: The inner-loop is activated and allowed to adapt. It is initialized with the same model as the compensated run such that adaptation is expected to be minimal.

Test 1 will be used as the control run to compare to the performance in test 2 and test 3. Tests 2 and 3 will compare the compensated and adaptive performance to the performance shown in Test 1. This will attempt to show improved closed loop performance and reduced steer angle oscillations in Tests 2 and 3. The estimation algorithm was run during each of the tests, but was only used to adapt the SP model during Test 3.

Two maneuvers were performed during the testing. The first was a double lane change type maneuver, performed at speeds of approximately 15 and 20 miles per hour (7 and 9 m/s). The desired path for the double lane change maneuver is shown in Figure 6.7.

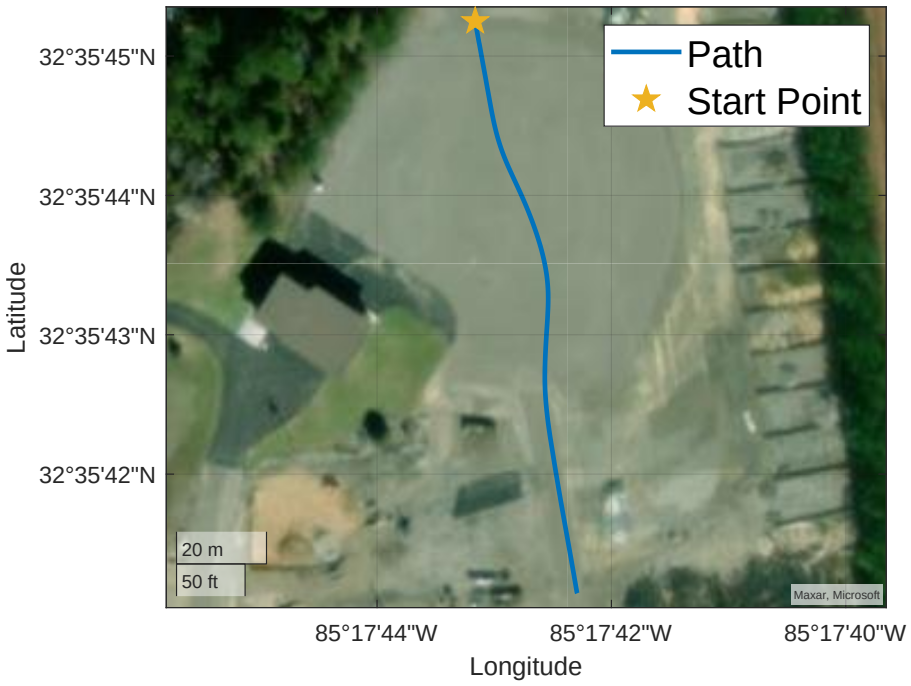


Figure 6.7: Double Lane Change Desired Path

The second maneuver performed was a slalom maneuver, shown in Figure 6.8. This was performed at speeds of approximately 10 and 15 miles per hour (5 and 7 m/s). This maneuver was chosen to test the systems performance with a sinusoidal type reference. For each maneuver, 6 runs were conducted: 2 control runs, one at a higher speed and one at a lower speed, and 1 compensated and 1 adaptive run for each speed. The description of the individual runs is shown in Table 6.1. The control, compensated, and adaptive runs for each speed are compared to show the performance improvements in the following sections.

Table 6.1: MKZ Testing Run Descriptions

Run Number	Double Lane Change	Slalom
1	Test 1, 15 mph	Test 1, 10 mph
2	Test 2, 15 mph	Test 2, 10 mph
3	Test 3, 15 mph	Test 3, 10 mph
4	Test 1, 20 mph	Test 1, 15 mph
5	Test 2, 20 mph	Test 2, 15 mph
6	Test 3, 20 mph	Test 3, 15 mph

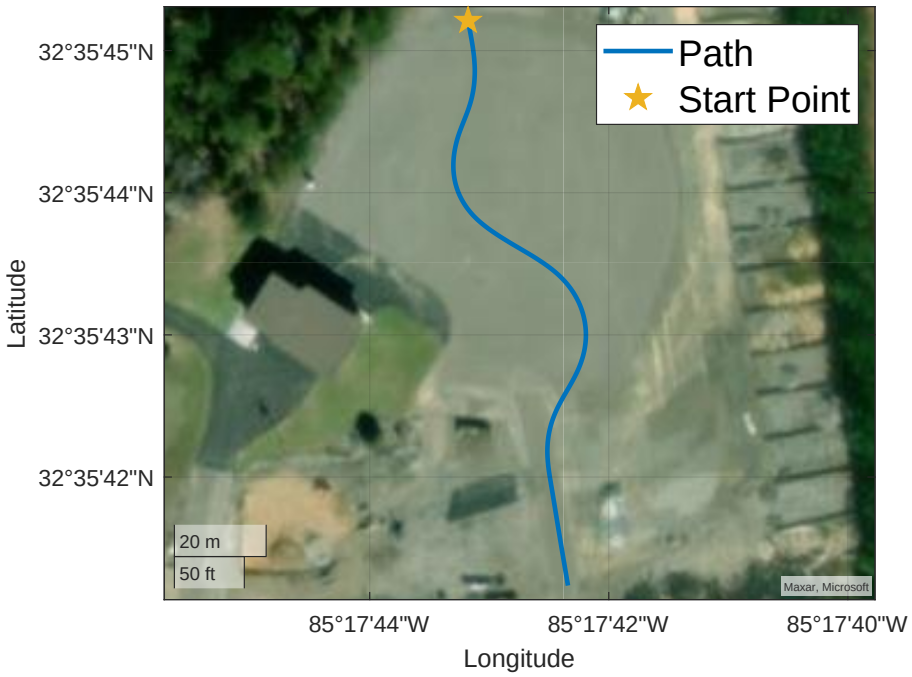
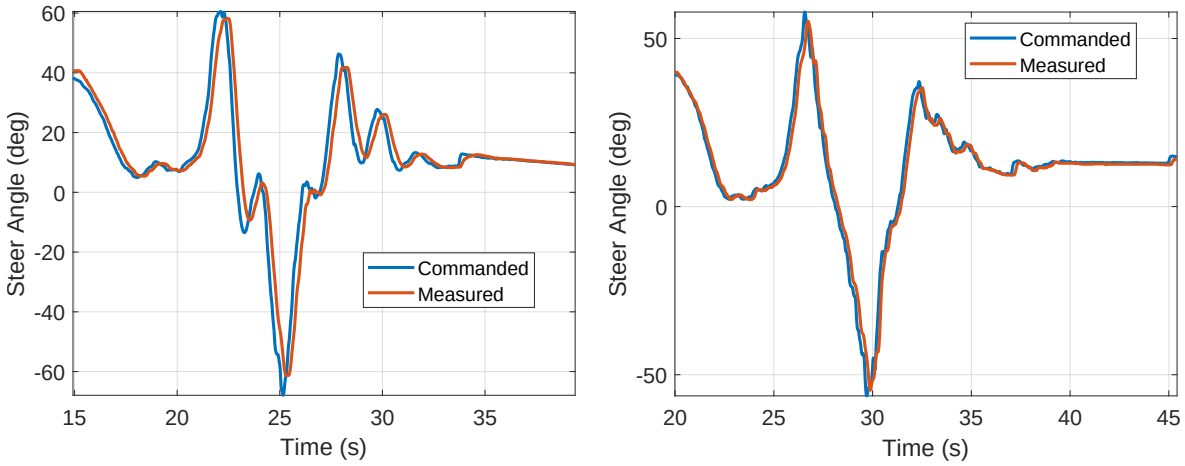


Figure 6.8: Slalom Maneuver Desired Path

6.3.2 Double Lane Change Results

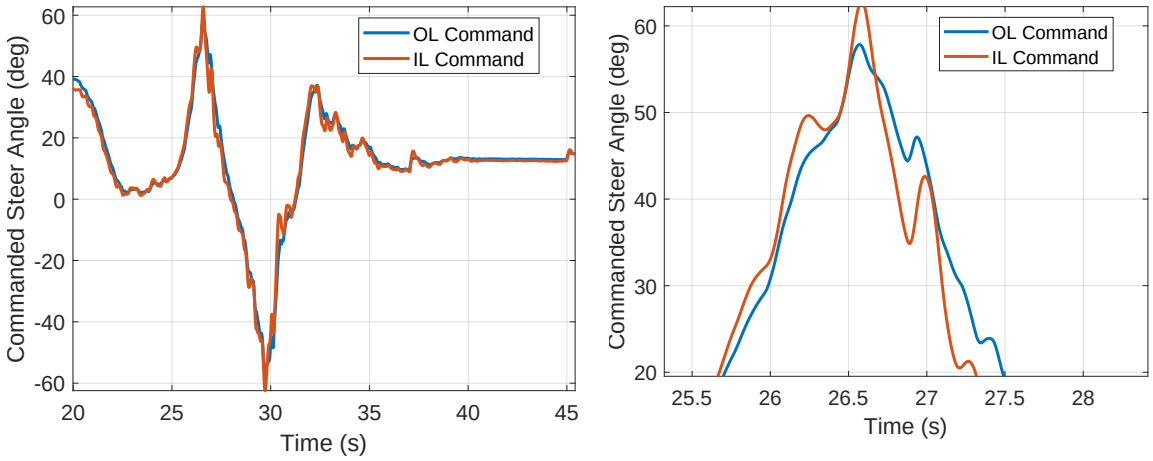
The results from the MKZ double lane change test are presented and discussed in this section. Generally, the compensated and adaptive runs were better able to track the desired steer angle commanded by the outer-loop controller. The improvement was most noticeable in the 20 mph run. This was expected as a higher velocity leads to a more dynamic steering response required to track the reference path. The outer-loop steering command and the measured steer angle are plotted versus time for the 20 mph control and adaptive runs in Figure B.12. The adaptive run tracks the desired steer angle with less phase lag than the control run. The non-adaptive run at this speed provided very similar performance to the adaptive run.

The SP mitigates the effect of pure delay by using a model of the system to predict future states, which are fed back to the controller. This effectively introduces a phase lead into the system, compensating for the lag introduced by the delay. This effect can be seen in Figure 6.10, where the inner-loop and outer-loop desired steer angles are plotted versus time. The inner-loop commanded steer angle is shown to lead the outer-loop command slightly which helps to reduce the negative effects of the pure delay and to increase the bandwidth of the system.



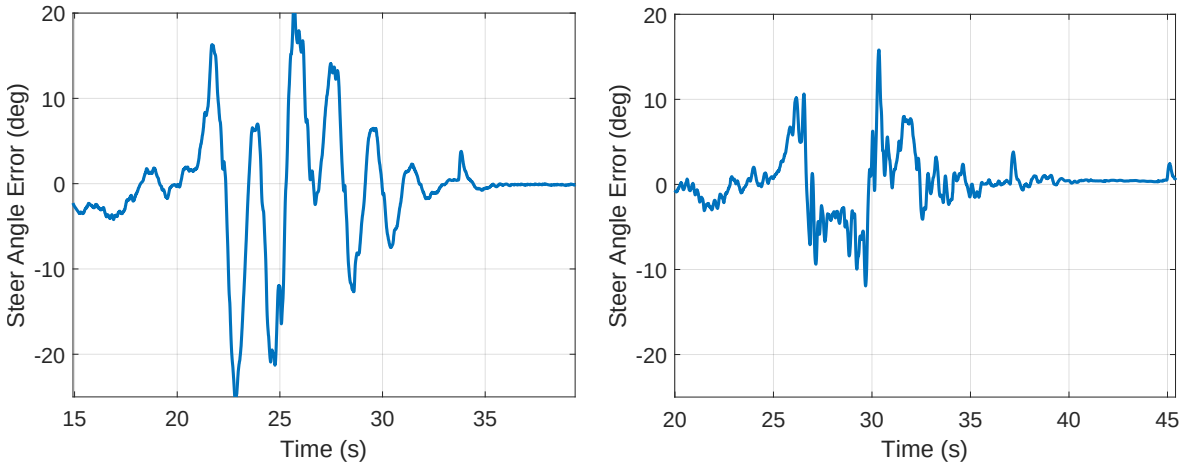
(a) Control Run (b) Adaptive Run

Figure 6.9: Commanded and Measured Steer Angle: 20 mph Double Lane Change



(a) Full Run (b) Enhanced Portion of Run

Figure 6.10: Outer-Loop and Inner-Loop Steer Angle Commands: 20 mph Double Lane Change



(a) Control Run (b) Adaptive Run

Figure 6.11: Steer Angle Error: 20 mph Double Lane Change

The steer angle error is defined as the difference between the measured and desired steer angle. It is plotted for the two runs in Figure 6.11. The adaptive run is able to maintain a lower error for the majority of run. The faster steering response provided by the compensation leads to slightly lower heading error for the adaptive run compared to the control. The heading error for each run is plotted versus the X-component of the target waypoint in the vehicle frame in Figure 6.12. It is apparent that there is very little difference in performance between the compensated and adaptive runs. This was expected as the non-adaptive run was initialized with a steering model known to be relatively accurate, so any improvements from the adaptation were expected to be small.

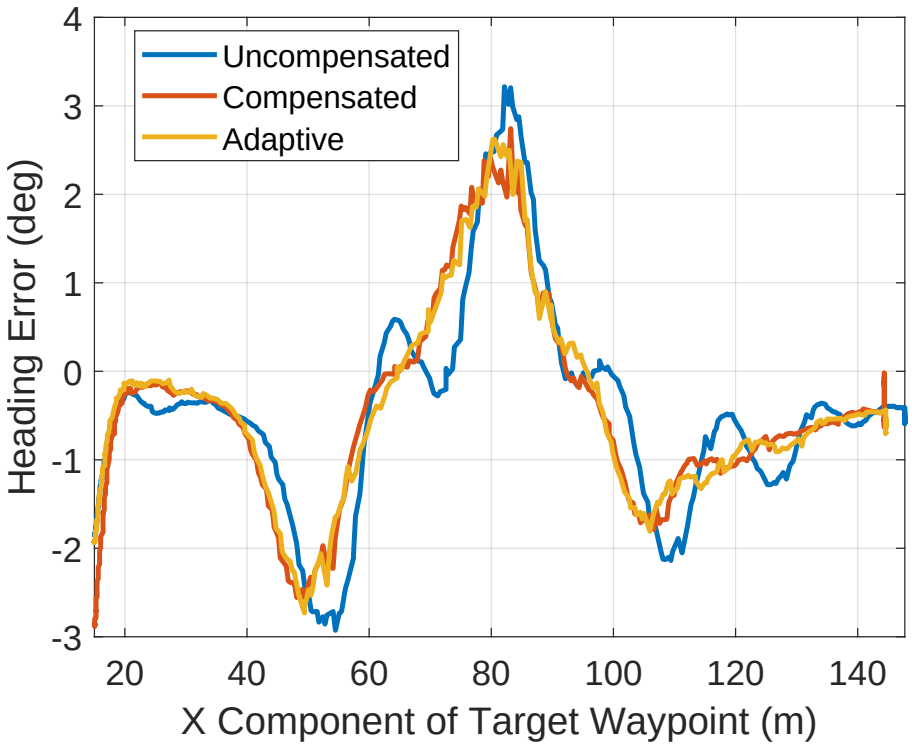


Figure 6.12: Double Lane Change Desired Path

The parameter estimation algorithm was active on all runs in which the inner-loop was running, even when the estimates were not being used to adapt the prediction model. The parameter estimates are plotted versus time for runs in which the inner loop was active (2, 3, 5, and 6) in Figure 6.13. The estimates initially converge quickly and remain constant for around 10 seconds. This is because the vehicle is static at the start of each run such that the heading error remains constant until the vehicle begins to move. This means that the only change in desired steer angle before the vehicle begins moving is the initial step input. Because of this, there is limited excitation for the parameter estimation algorithm. Once the vehicle begins moving and the steering commands become more dynamic, the parameter estimates begin to change again, and converge to new values by the end of the run. For each of the 4 runs shown, the dynamic parameters ($\hat{a}$ and $\hat{b}$) converge to similar values. In other words, the parameter estimates show a similar trend across all 4 runs. The estimate of pure delay converges to a value between 5 and 10 samples for each of the runs. Similar results were seen across all runs because the estimates were initialized with the same values for each run. Additionally,

the desired path for the runs was the same, leading to very similar desired and measured steer angles across all four runs.

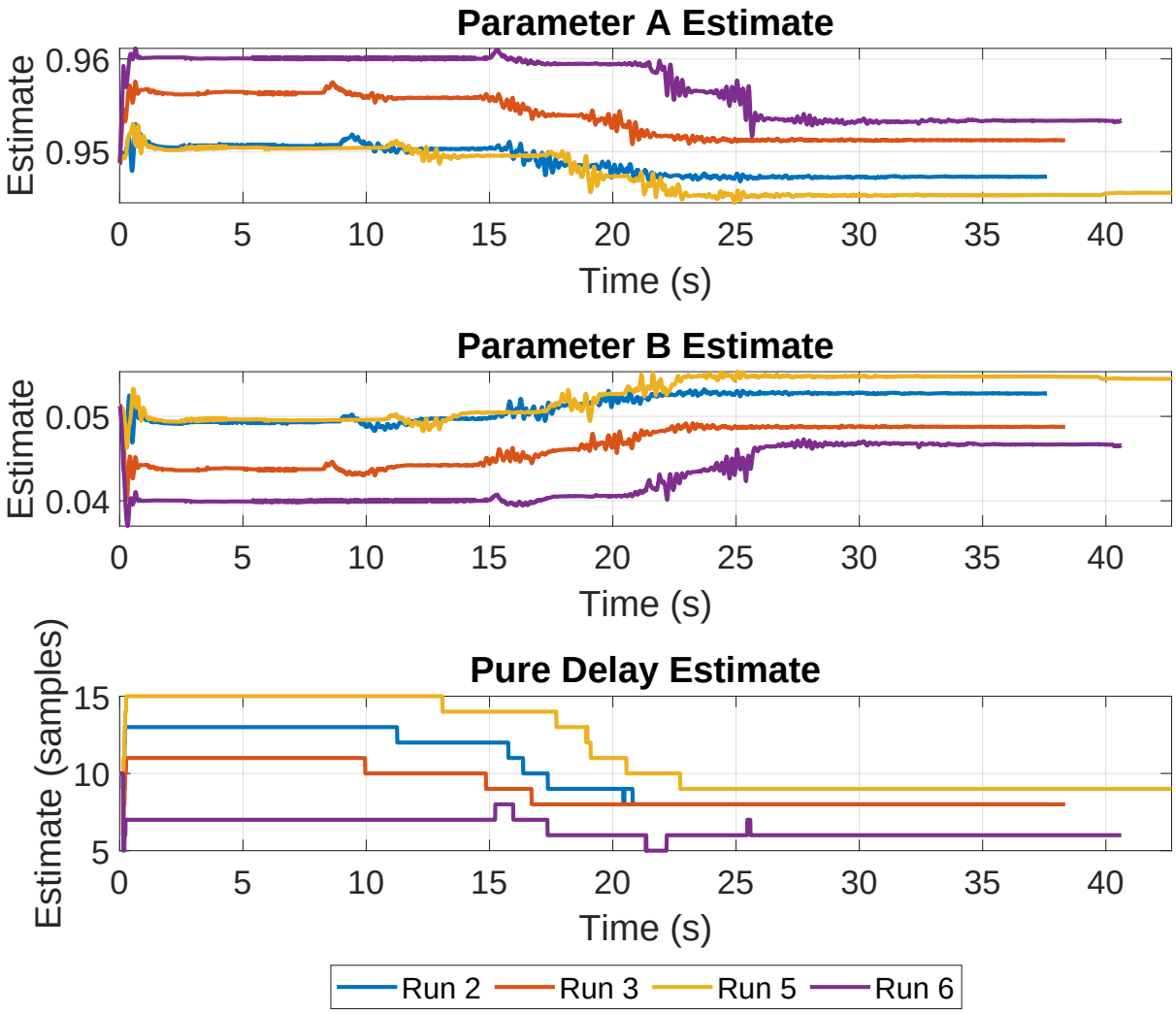


Figure 6.13: Double Lane Change Parameter Estimates, 20 mph

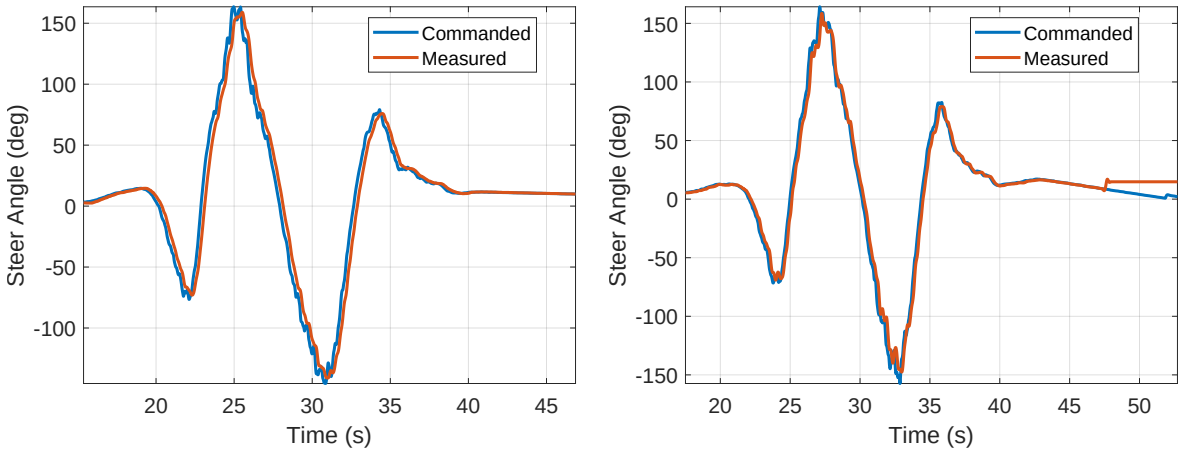
The 15 mph double lane change results followed a similar trend, but the improvement provided by the compensation was less noticeable because of the less dynamic nature of the maneuver due to the lower speed. The results for the slower run as well as additional results from the 20 mph run can be found in Appendix B.

6.3.3 Slalom Results

The results from the slalom test runs are presented and discussed in this section. The results for this maneuver follow a similar trend to the double lane change tests. The compensation is able to provide more accurate tracking of the desired steer angle. The improvement in steering

error is not as apparent as the 20 mph double lane change results. This is most likely because of the slower speeds of the slalom run. The uncompensated actuator is able to track the desired steer angle for lower speed maneuvers better than for high speed maneuvers. The outer-loop steering command is plotted versus time and compared to the measured steer angle for the 15 mph control and adaptive slalom runs in Figure 6.14. The compensated run is not shown here, but additional data from the slalom test can be found in Appendix B.

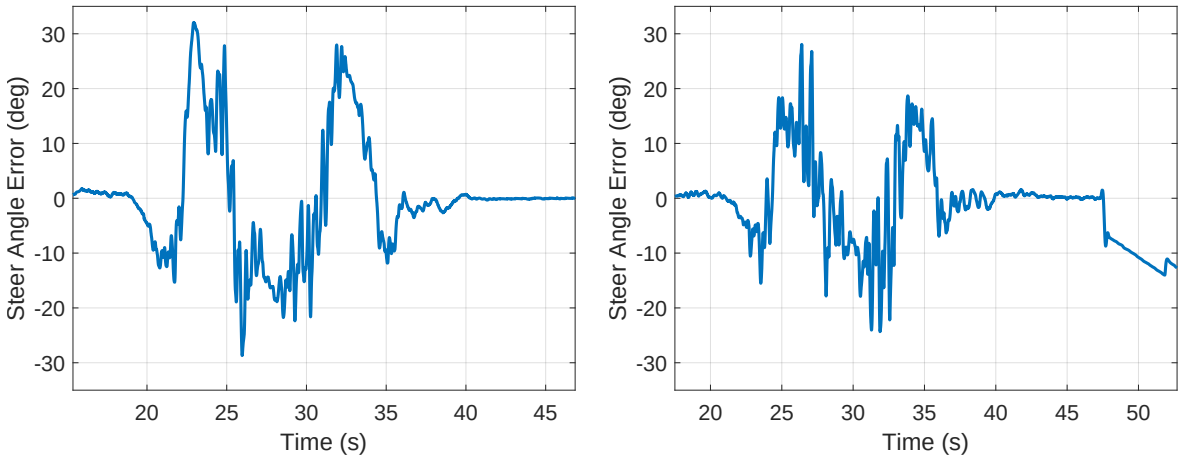
The measured steer angle in the adaptive run closesly tracks the commanded angle, whereas the control run displays greater phase lag. The steer angle errors for both runs are plotted in Figure 6.15. While the improvement provided by the adaptive algorithm is less noticeable than the improvement shown in the double lane change, the maximum error is reduced and the spikes in error are lower in the adaptive run. The non-adaptive compensation performs similarly to the adaptive run, and is shown in Appendix B. The compensation does not provide a noticeable improvement in heading error for this maneuver, but it also does not degrade performance.



(a) Control Run (b) Adaptive Run

Figure 6.14: Commanded and Measured Steer Angle: 15 mph Slalom

The parameter estimation algorithm was again active for every run in which the inner-loop was running (runs 2, 3, 5, and 6). The parameter estimates for these runs are plotted versus time in Figure 6.16. The parameters are again initiailized at the same values. The estimates again converge initially and re-converge as the vehicle performns the maneuver. The pure delay estimates converge to values between 6 and 8 samples. For each of the 4 runs, the estimates of a and b converge to very similar values. The same trend is seen here as in the double lane



(a) Control Run (b) Adaptive Run

Figure 6.15: Steer Angle Error: 15 mph Slalom Test

change results. The parameters are initialized with their nominal values and the estimate of $\hat{a}$ converges to a lower value by the end of the run while $\hat{b}$ increases. The resulting model is slightly faster than the nominal model, although it remains very close to nominal values.

Both the adaptive and compensated runs were found to improve the steering performance of the MKZ for both maneuvers tested. The inner-loop provided closer tracking of desired steer angles, and the inner-loop command was shown to lead the outer-loop command, compensating for the negative effects of the pure delay present in the steering system. The parameter estimates were able to converge for each run, and stability was maintained when the estimates were used to adapt the prediction model online. Improvements in overall path tracking performance were not always provided by the inner-loop, but this is most likely caused by outer-loop controller tuning. The inner-loop SP compensation simply attempts to track the desired steer angle as accurately as possible, so for certain outer-loop controllers and tunings this may not lead to a noticeable improvement in path tracking performance.

6.4 Further Analysis of Parameter Estimation Performance on MKZ

6.4.1 Estimation Performance with Inaccurate Initial Model

For each of the tests discussed previously in this section, the parameter estimation algorithm was initialized with the same values based on a model obtained from steering response data taken from the MKZ. This model is known to closely match the true steering response. The

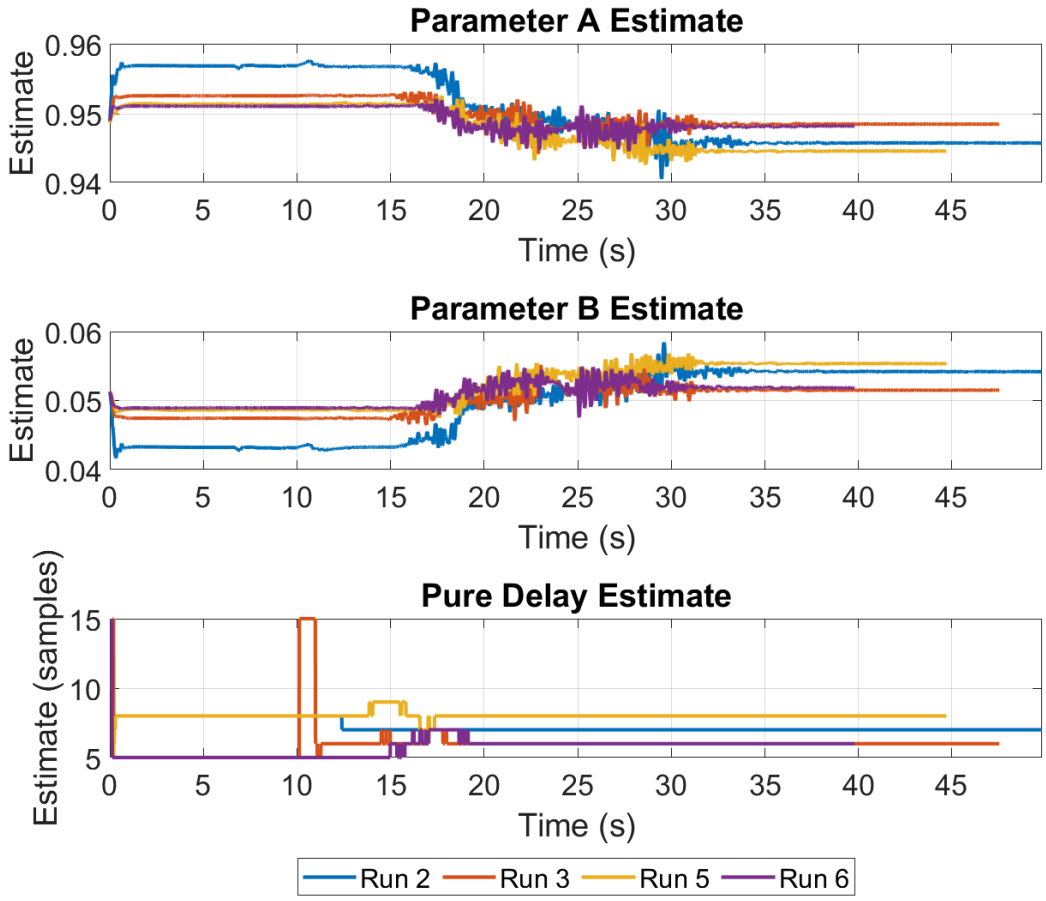


Figure 6.16: Slalom Parameter Estimates, 15 mph

goal of the parameter estimation is to adapt to changes in steering response and to correct for modeling inaccuracies in order to minimize prediction error. In order to examine the algorithm's performance when initialized with incorrect parameters, three additional test runs were conducted with varying initial parameter values. Table 6.2 shows the initial values for each run.

Table 6.2: Initial Parameter Values

Run	Initial Parameters
Run A	$\hat{a}_0 = 0.9487$, $\hat{b}_0 = 0.0513$, $\hat{\alpha}_0 = 10$
Run B	$\hat{a}_0 = 0.94$, $\hat{b}_0 = 0.06$, $\hat{\alpha}_0 = 8$
Run C	$\hat{a}_0 = 0.97$, $\hat{b}_0 = 0.03$, $\hat{\alpha}_0 = 12$

Run A was initialized with the nominal model obtained from experimental data and used for each of the runs previosly discussed. Run B started with a model with faster dynamics and smaller pure delay than the nominal model and Run C was initialized with slower dynamics and a larger delay value than run A. The reference path for each run was the slalom maneuver

shown previously in Figure 6.8, and each run was conducted at speeds of around 15 mph. The parameter estimates are plotted versus time for each of the three runs in Figure 6.17. Despite different starting values for both delay and dynamic parameters, the estimates converge to similar values. The dynamic parameters converge to very similar values across the three runs. The final delay estimates are a bit more varied, but converge to values between 5 and 9 samples. The prediction error for each of the three runs is plotted in Figure 6.18. The prediction error remains less than 10 degrees for the majority of each run, showing that the estimation algorithm is able to converge to an accurate model with small inaccuracies in the initial parameter values. Run A exhibits the lowest errors throughout the run. This was expected, as it was initialized with the nominal steering model.

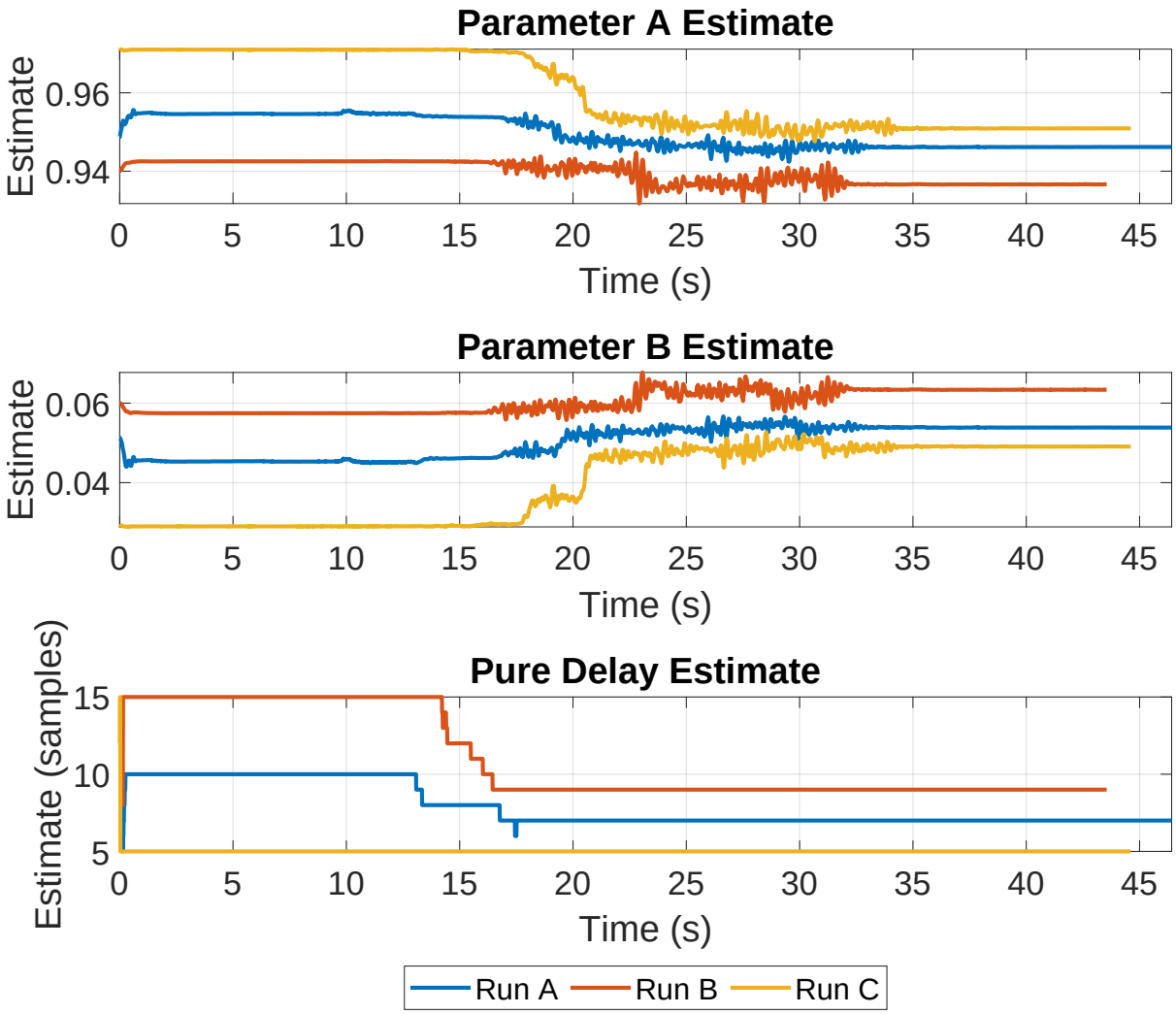


Figure 6.17: Parameter Estimates With Different Initial Values

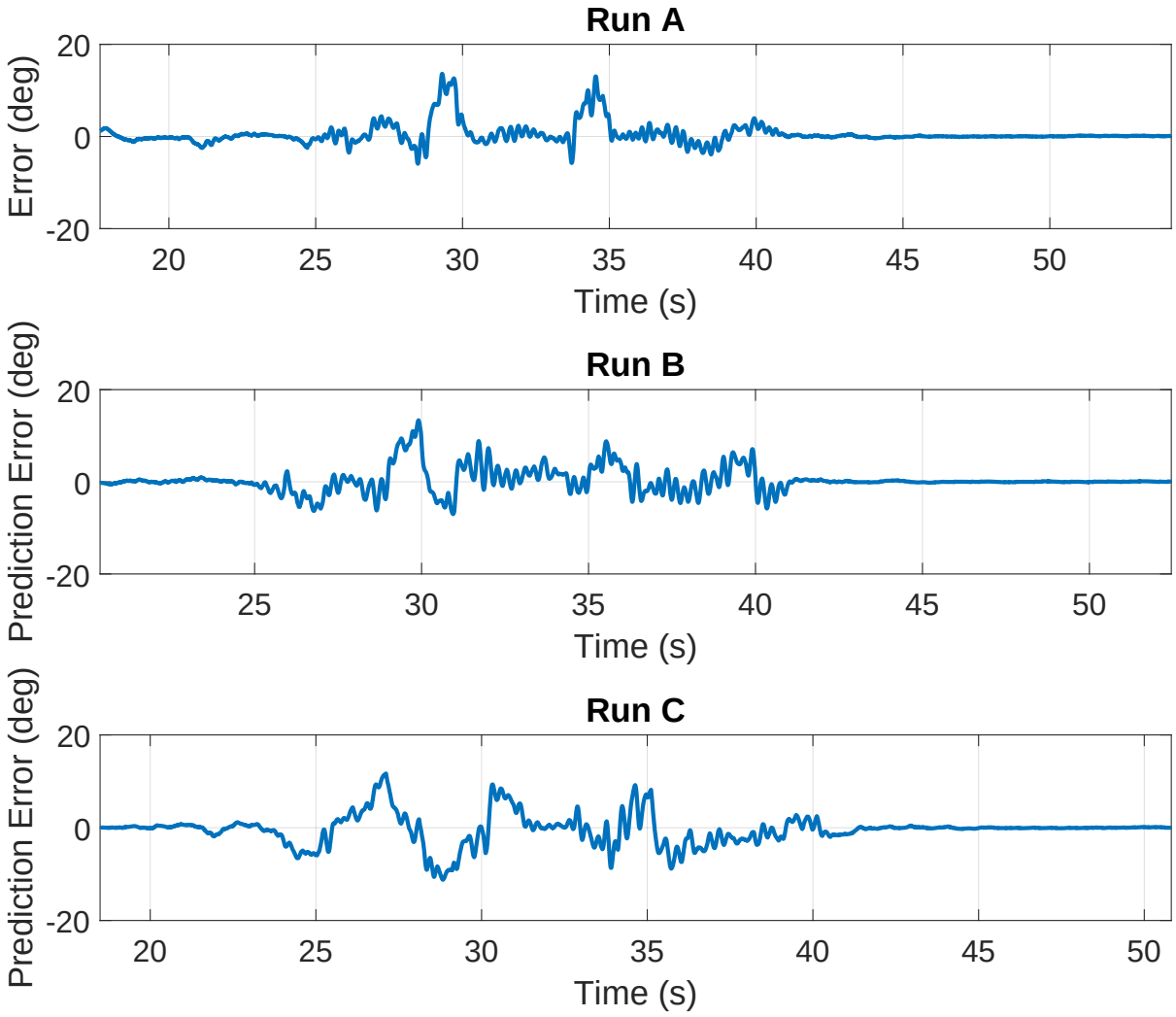


Figure 6.18: Steer Angle Prediction Error for Different Parameter Initializations

6.4.2 Adaptive Compensation Performance with Inaccurate Initial Model

To further validate the parameter estimation algorithm, two additional experiments were conducted comparing the adaptive and non-adaptive performance of the algorithm in the presence of an incorrect model. The Smith predictor performance is especially susceptible to inaccurate modeling of pure delay. To illustrate this and to show the improvement provided by adapting the prediction model, both the adaptive and non-adaptive run were initialized with a delay value of 15 samples (0.15 seconds). This value is known to be higher than the typical delay present in the MKZ steering system. The dynamic parameters were initialized with their nominal values, shown in Table 6.2 for Run A. The prediction errors for the non-adaptive and adaptive runs are shown in Figure 6.19. This test was conducted with the slalom maneuver as the reference path.

The prediction error of the non-adaptive run shows large oscillations and has a maximum error of over 20 degrees, while the adaptive run error is less oscillatory and stays below 10 degrees of error for most of the run.

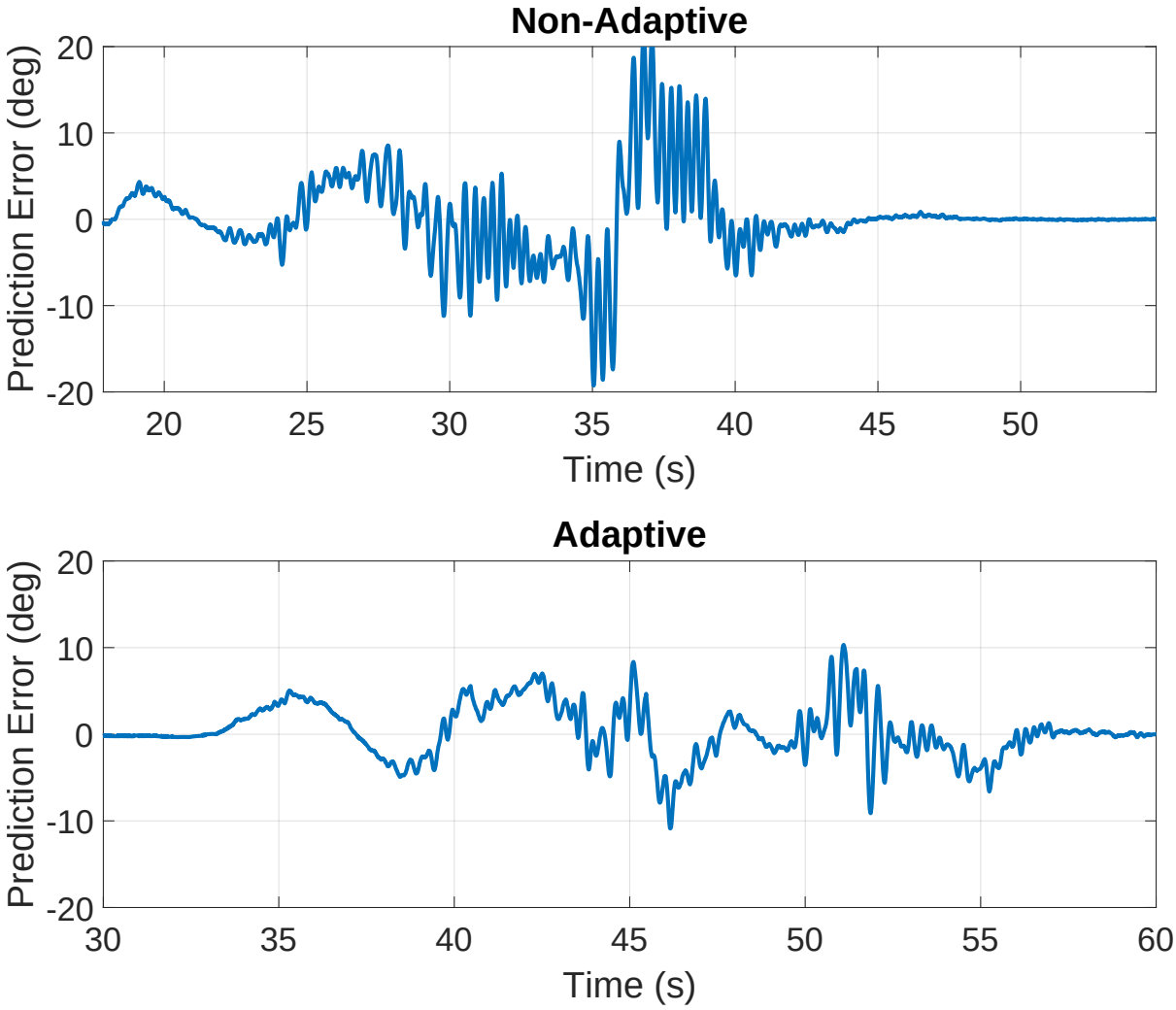
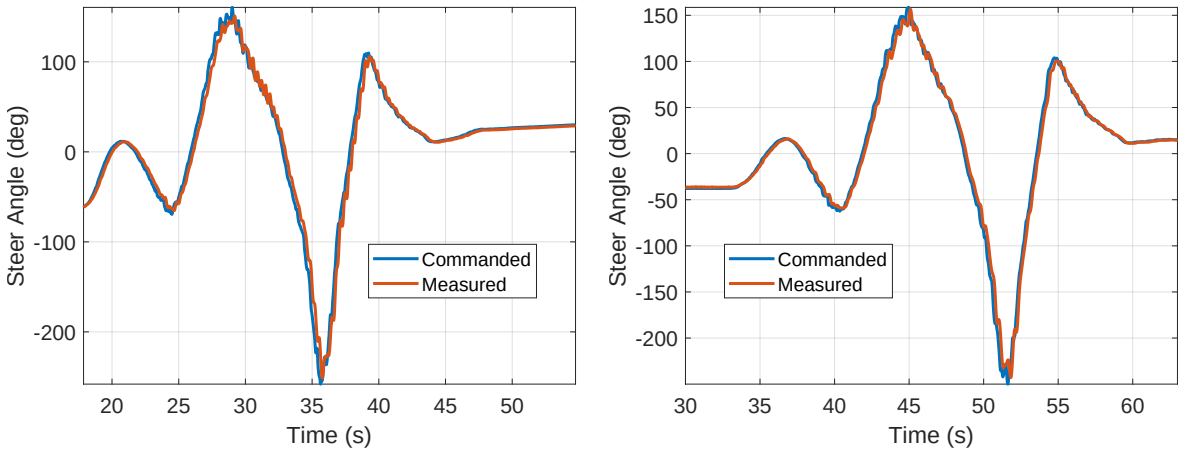


Figure 6.19: Prediction Error - Inaccurate Model

The prediction errors due to the inaccurate model also lead to poor inner-loop control performance for the non-adaptive run. The desired and measured steer angles for the two tests are compared in Figure 6.20. The adaptive run provides closer tracking of the desired steer angle with less oscillations. This illustrates the negative effects on control performance that arise from a model of pure delay that is off by only 0.05 seconds. It also shows the improvement that the adaptation can provide when the prediction model is initialized with an incorrect model or the steering response changes.



(a) Non-Adaptive Run

(b) Adaptive Run

Figure 6.20: Commanded and Measured Steer Angle: Inaccurate Prediction Model

6.4.3 Estimation Performance with No Initial Model

Another scenario in which estimating steering parameters would be valuable is if the control designer had no prior knowledge of the steering actuator response. In this case, the estimation algorithm could be used in a calibration run to obtain an estimate of the steering parameters in order to initialize the inner-loop compensation. In order to examine the algorithm's performance with no prior knowledge of the steering parameters, a test was run with the parameters initialized at the following values: $\hat{a}_0 = 0$, $\hat{b}_0 = 1$, and $\hat{\alpha}_0 = 1$. The inner-loop controller was not active for these tests; the outer-loop command was passed directly to the steering system. The run was conducted twice, again using the slalom reference path. The parameter estimates are plotted versus time for each run in Figure 6.21.

The parameter estimates converge to similar values in both tests. The final pure delay estimate is 15 samples (0.15 seconds) for each of the runs. At the start of the runs the delay estimate immediately jumps to 15 samples, which is the highest value in the search range used for the experimental tests. This is due to the initialization of the dynamic parameters. $\hat{a}$ is initialized at 0 and $\hat{b}$ is initialized at 1, meaning the initial model assumes the commanded steer angle is instantly achieved. The algorithm tries to minimize prediction error, and at the start of the run the lowest prediction error will come from the highest delay value within the search range. As the run continues the dynamic parameters converge to more realistic values. The delay estimate never deviates from 15, however with a longer run it would be expected to

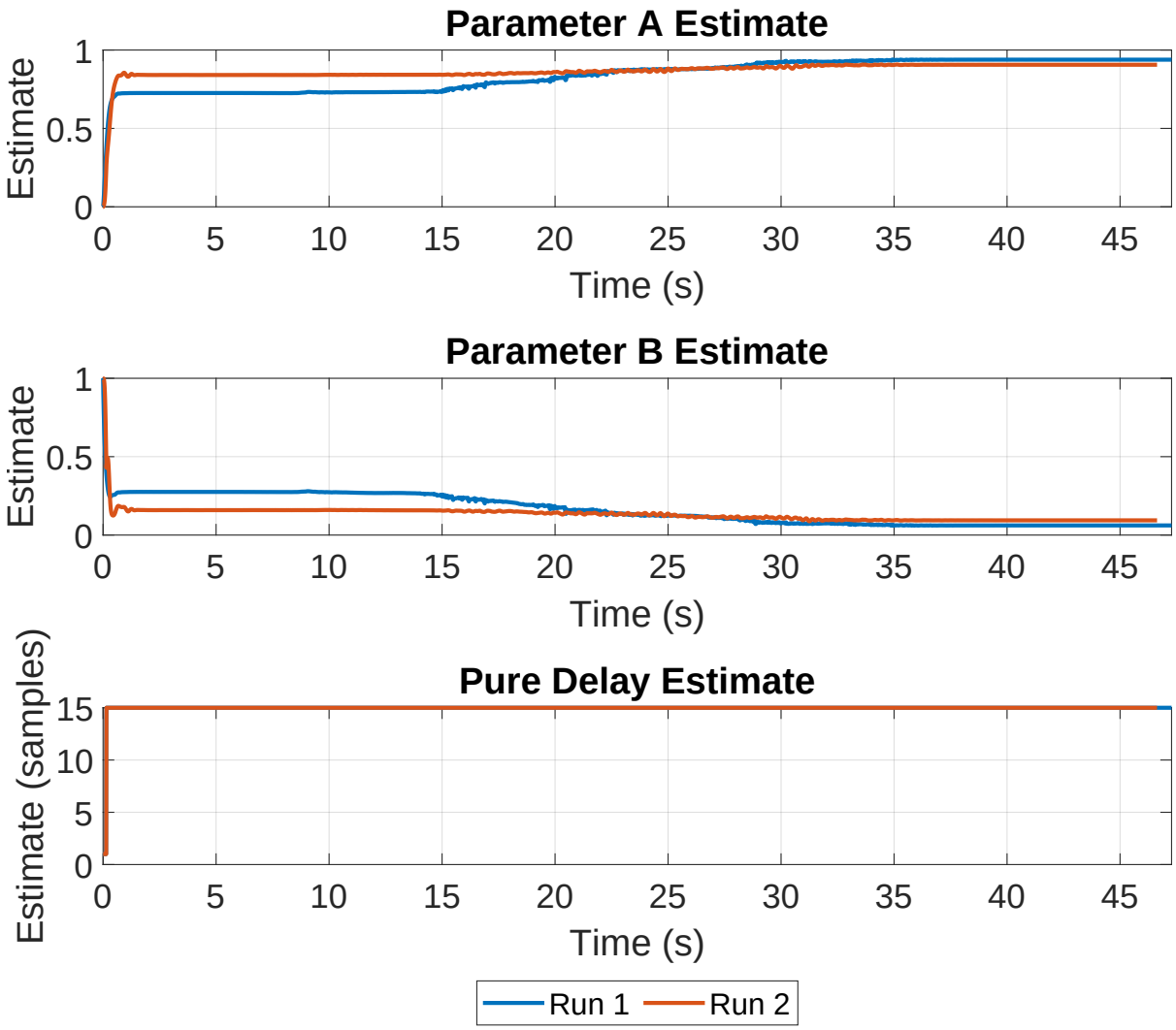


Figure 6.21: Parameter Estimates, No Initial Model

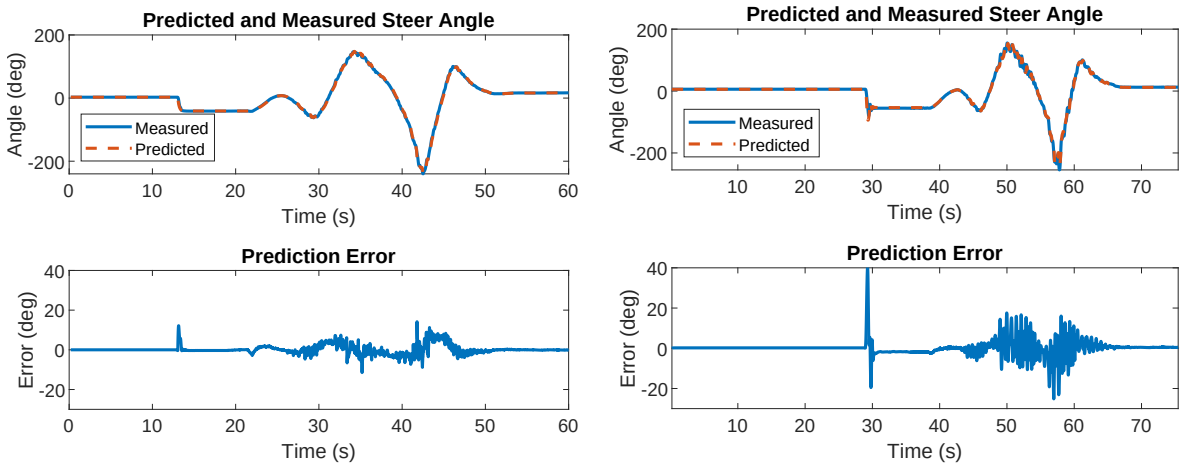
converge to a more realistic value. The final estimated steering model from each run is shown in Equations (6.1) - (6.2).

$$Run\ 1: \quad \hat{G}(z) = z^{-15}\frac{0.06143}{z - 0.9385} \tag{6.1}$$

$$Run\ 2: \quad \hat{G}(z) = z^{-15}\frac{0.09391}{z - 0.9062} \tag{6.2}$$

To determine the accuracy of the prediction models, they were used post-process to predict future steer angles based on the steering commands during the run. The predictions were compared to the measured steer angle from each run. The predicted and measured steer angles as well as prediction error for both runs are shown in Figure 6.22. Run 1 is shown to converge

to an accurate model, with errors under 10 degrees for much of the run. Run 2 converges to a usable but less accurate model, with errors under 20 degrees for the majority of the run.



(a) Run 1 (b) Run 2

Figure 6.22: Predicted and Measured Steer Angle and Prediction Error for Runs 1 and 2

In Figure 6.23, the step responses of the models obtained in Run 1 and Run 2 (Equations (6.1) and (6.2)) are compared to the measured step response of the actuator and the response of the "nominal" model shown in Table 6.2 (Run A). The model obtained in Run 1 is shown to closely follow both the nominal model and the measured response, while Run 2 is shown to be slightly less accurate.

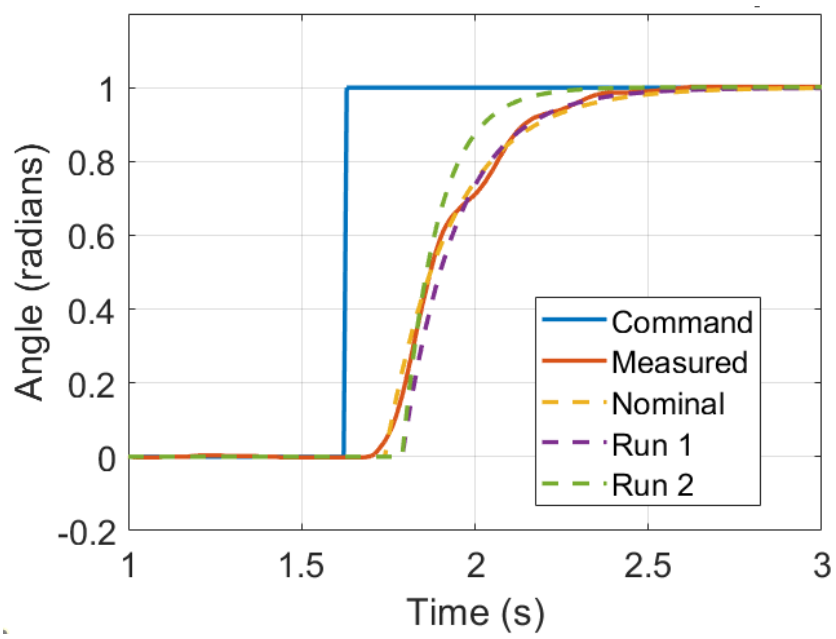


Figure 6.23: Step Response of Estimated and Nominal Models

Next, the performance of the estimation algorithm was examined when the delay estimate was held constant. In other words, only the dynamic portion of the steering response was estimated. The delay estimate was held constant at its nominal value of 10 samples. The algorithm converged to the model shown in Equation (6.3).

$$\hat{G}(z) = z^{-10}\frac{0.0528}{z - 0.9470} \tag{6.3}$$

The estimates converge nearly to the nominal values. This estimated model was then compared to the nominal and measured step responses. The comparison is shown in Figure 6.24. The estimated model closely tracks the nominal and the measured response to a step input. This shows that a very accurate model can be obtained when a good estimate of pure delay is known. Therefore, better results are likely possible if the estimation algorithm is allowed to converge to a model using only the initial estimate of delay for a short amount of time. This would allow the dynamic parameters to converge while the delay estimate is held constant. Once the dynamic parameters have been given ample time to converge, the delay estimation could be turned on in order to track possible changes in pure delay. This would avoid the issue of the delay estimate over-correcting for an inaccurate model by converging to a value that is too high or too low.

The parameter estimation algorithm was shown to converge to an accurate steering model from a variety of initializations. It was able to improve prediction error and control performance when initialized with a poor model. It was also able to converge to a relatively accurate steering model with no prior knowledge of the steering system. These tests validate the estimation algorithm and show its ability to maintain accurate predictions of steer angle despite changing or incorrectly modeled steering dynamics.

6.5 Parameter Estimation for Additional Vehicles

To further test the robustness of the parameter estimation algorithm, it was validated on data from two additional test vehicles. The first was a Peterbilt 579 tractor trailer outfitted with DBW

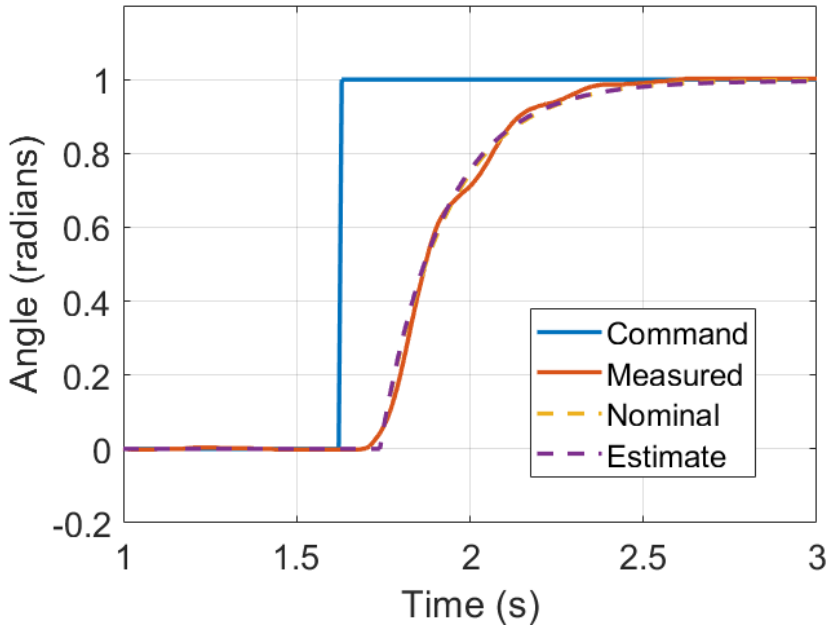


Figure 6.24: Step Response of Nominal and Adaptive (with Fixed Sample Delay Estimate) Models

hardware. The Peterbilt is shown in Figure 6.25. The second was an autonomous IndyCar, also outfitted with a DBW system, shown in Figure 6.26.

Figure 6.25: Peterbilt 579

Figure 6.26: Autonomous IndyCar

6.5.1 Peterbilt Steering Estimation

In order to test the estimation algorithm on the Peterbilt truck, dynamic data was collected on the NCAT test track. Measured and commanded steer angle data from a portion of the dataset was then used in the parameter estimation algorithm in post process. The portion of the data chosen included a double lane change maneuver and a banked turn. The GPS derived path of the chosen portion of data is shown in Figure 6.27. The data was taken at speeds of around 20 mph.

The parameter estimates are plotted versus time for the selected portion of the run in Figure 6.28. The parameters were initialized with no prior knowledge ($\hat{a} = 0$, $\hat{b} = 1$, and $\hat{\alpha} = 1$). The estimates converge in around 40 seconds to a relatively accurate model. The predicted and measured steer angles are plotted with the prediction error in Figure 6.29. The error is shown to stay below 5 degrees for much of the run, indicating that the estimation algorithm was able to obtain an accurate model.

6.5.2 Indy Car Steering Estimation

Next, the estimation algorithm was tested on data from the steering actuator of an autonomous IndyCar. The data used was from a dataset recorded on the Las Vegas Motor Speedway. A 55

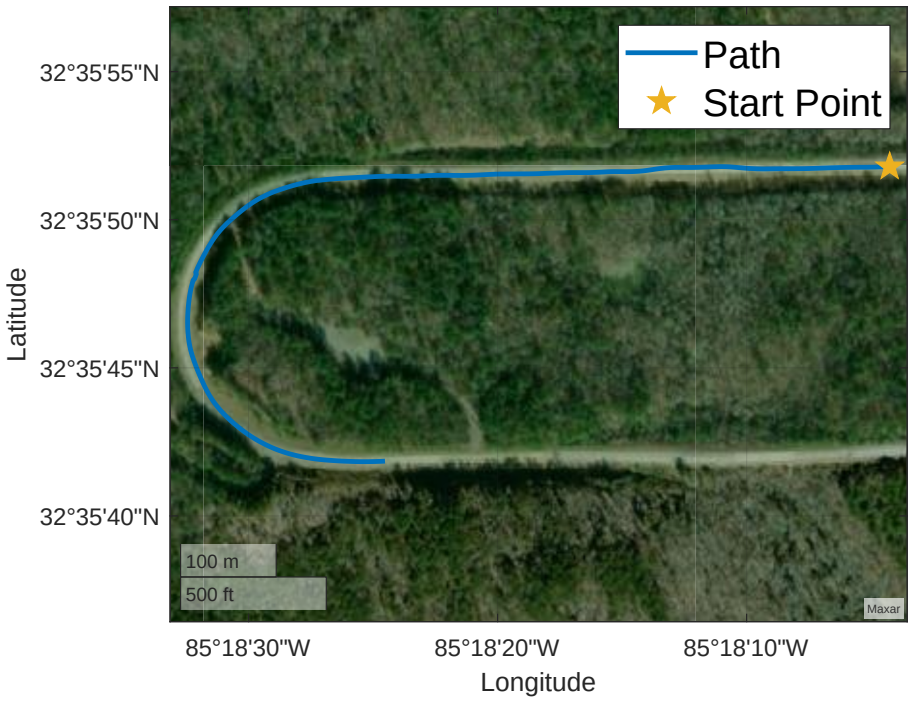


Figure 6.27: Peterbilt Path

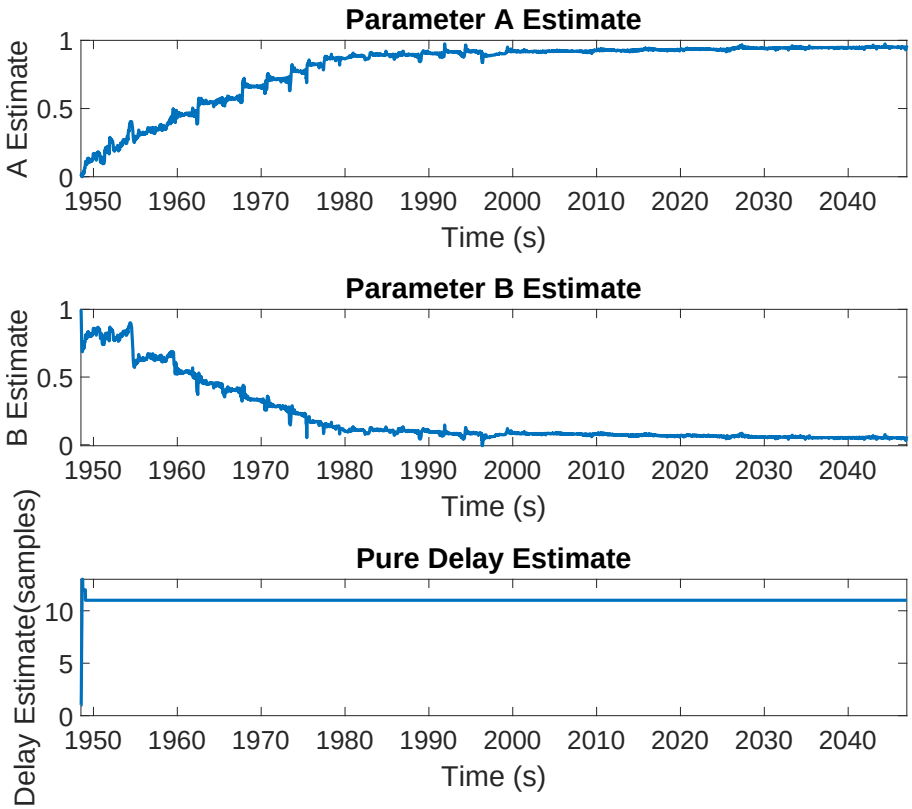


Figure 6.28: Peterbilt Parameter Estimates

second portion of this data was chosen and used in the parameter estimation algorithm in post process. The parameters were again initialized with no prior knowledge.

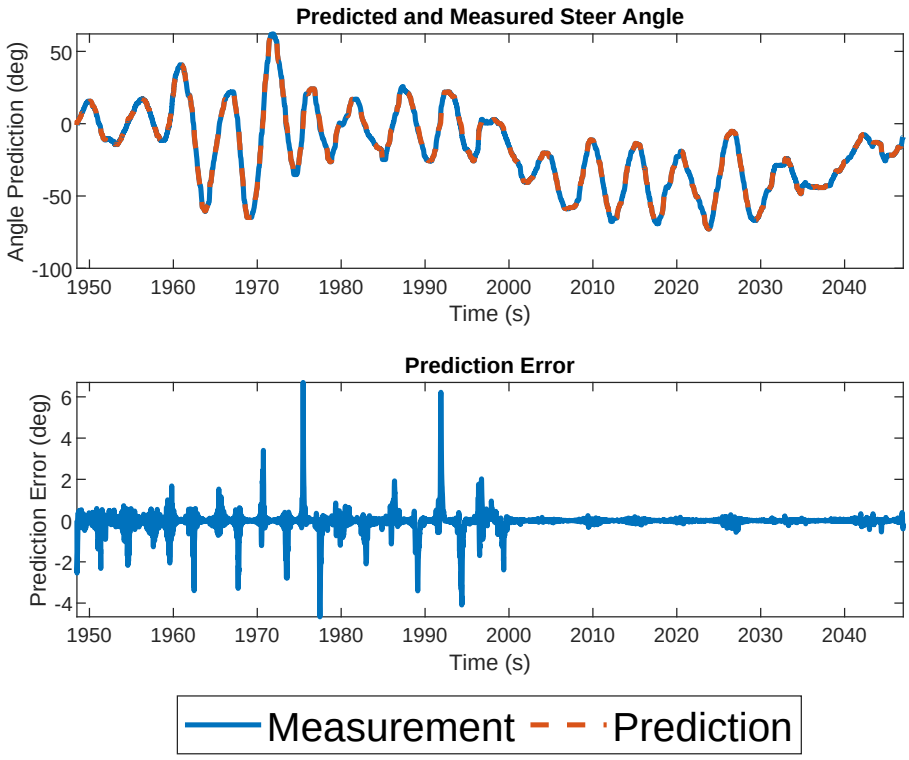


Figure 6.29: Peterbilt Predicted and Measured Steer Angle, Prediction Error

The parameter estimates are plotted versus time in Figure 6.30. The estimates converge within around 30 seconds. Interestingly, the delay estimate immediately converges to zero. This is likely because of the different software and hardware present on the IndyCar compared with the other test vehicles. The delay present between the control computer and steering system may be significantly smaller than that of the other two steering systems.

The measured and predicted steer angles as well as the prediction errors are plotted in Figure 6.31. The prediction errors for this dataset are very low. In fact, the maximum error for the run is around 0.3 degrees, which is lower than any of the other tests. The reason for the increased accuracy is unknown, but could be due to a lower amount of delay between the computer and the steering system.

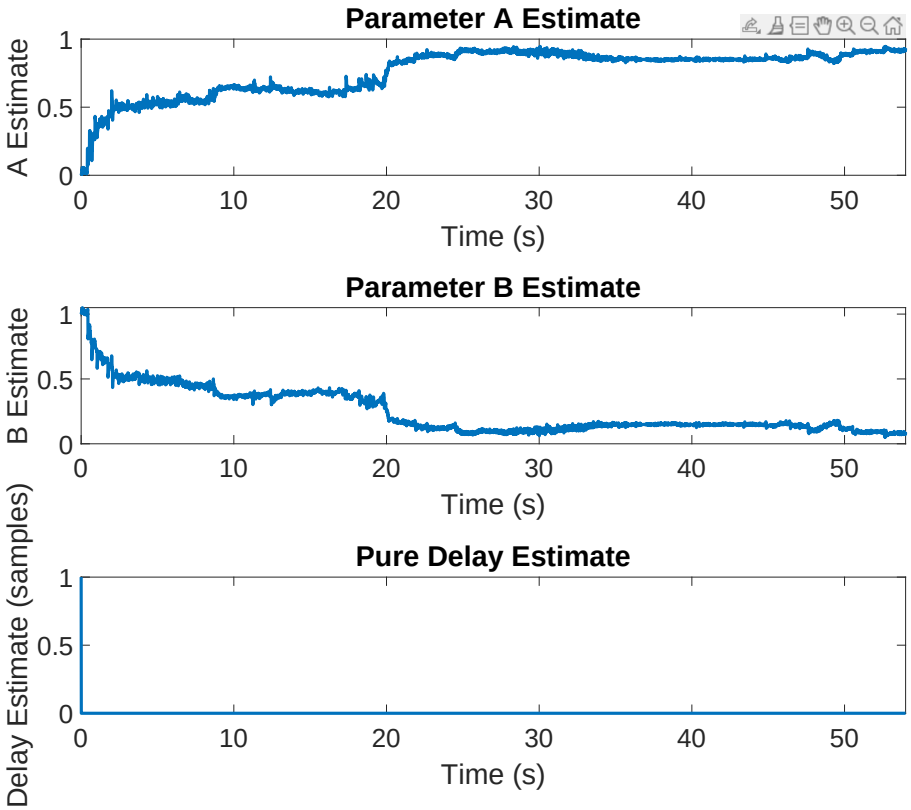


Figure 6.30: IndyCar Parameter Estimates

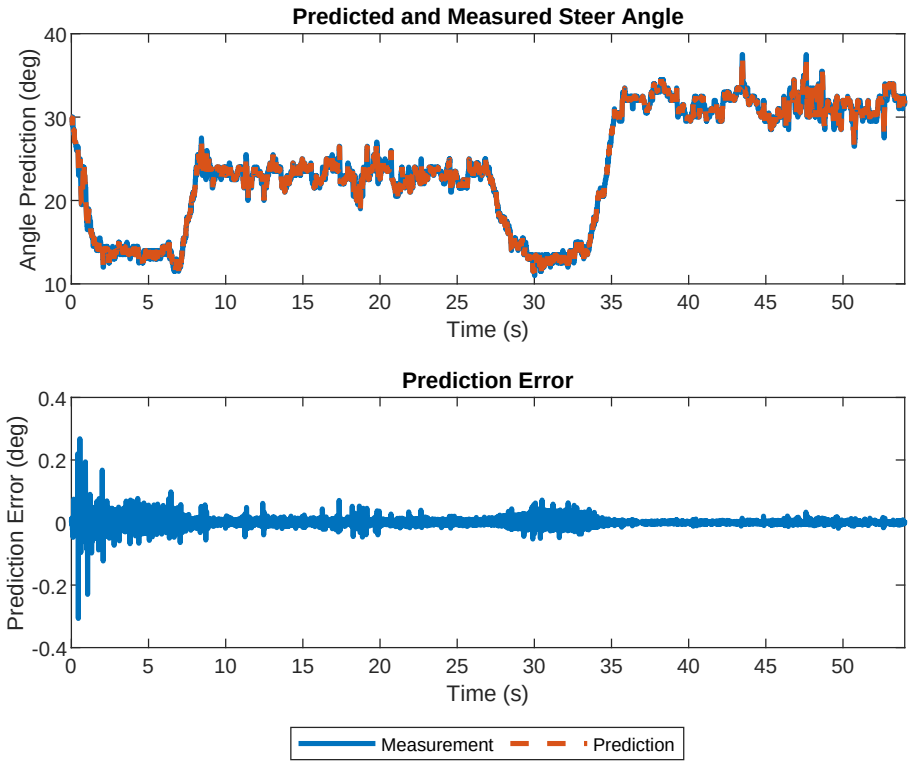


Figure 6.31: IndyCar Predicted and Measured Steer Angle, Prediction Error

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis presented an adaptive Smith predictor (SP)-based inner-loop compensation method for steering delay in a vehicle lateral control system. In Chapter 2, modeling and control design for time delay systems was discussed. A brief description of vehicle lateral dynamics and coordinate frames was also included. Chapter 3 presented the inner-loop control design, as well as the steering parameter estimation algorithm. Chapter 4 provided an analysis of the inner-loop algorithm's performance with a variety of path following controllers and examined the robustness to incorrectly modeled steering dynamics. Chapter 5 tested the algorithm in a higher fidelity simulation environment in a lane-keeping scenario. Chapter 6 tested the control and estimation in real-time, using a Lincoln MKZ with DBW hardware for experimental verification. The parameter estimation algorithm performance was also examined for two additional steering actuators: a Peterbilt and an autonomous Indy car.

The compensation algorithm was shown to be stable in all tests performed and provided improvement to both steering response and overall control performance with four different outer-loop controllers in the simulation described in Chapter 4. In each of the tests, the inner-loop controller was able to track the desired steer angle better than the uncompensater actuator, leading to lower heading and lateral errors. The adaptive algorithm did not always provide improvement in control performance over the non-adaptive runs, but it maintained stability and was able to improve the prediction model.

Chapter 5 showed the performance of the compensation in a lane-keeping SIL simulation performed in Gazebo, with a higher fidelity vehicle model of the Lincoln MKZ provided

by Dataspeed. Performance improvements were shown in more realistic and varied driving conditions than the Monte Carlo simulation and the real-time testing.

In the real-time experiment presented in Chapter 6, the algorithm was shown to provide improvement in steering response and in heading error for a waypoint following test using a Lincoln MKZ. The compensated actuator was able to track the desired steer angle from the outer loop controller better than the uncompensated actuator, leading to better overall control performance. The parameter estimation algorithm was stable when used to adapt the prediction model, and converged during each run. When the algorithm was initialized with an inaccurate model, the adaptation was shown to converge and provide a more accurate prediction model. The estimation algorithm was also shown to converge succesfully to models of two additional steering actuators.

In conclusion, the algorithm was shown in simulation and real-time tests to be succesful at mitigating the effects of delay in a lateral control system. The inner-loop method was validated in these tests and was shown to accomplish the following goals:

1. The algorithm was able to increase system bandwidth in the presence of actuator delay without the use of a vehicle model and without altering the outer-loop controller
2. The algorithm was shown to be robust to inaccuracies in the modeled steering dynamics
3. The parameter estimation was able to converge to an accurate model with little or no prior knowledge and maintained stability when used to adapt the prediction model in-run

7.2 Future Work

The proposed delay compensation method was shown to be effective, but there are many opportunities for improvement and future research. First, the inner-loop controller design and tuning could be improved. It is common for steering modules, especially those installed as part of a DBW kit such as the one in the MKZ, to impose torque and angular velocity limits on the steering wheel. The current design does not consider actuator saturation or constraints. These could be incorporated into the control design, ensuring that the commanded steer angles stay within the limits of the actuator.

Methods of adapting the inner-loop controller parameters based on the estimated steering response could be investigated. The current design does not alter controller gains or coefficients based on the estimated steering model. Because of this, some tuning would be required to apply the controller to different steering actuators. Adding adaptation to the control law could reduce or remove this need for tuning when applying the compensation to different vehicles.

Another valuable avenue to explore could be coupling the inner-loop compensation method with an outer-loop predictive method such as MPC. This is briefly investigated in Chapter 4, where model predictive control (MPC) is augmented with a model of the inner-loop response. This could potentially provide improved system performance, with the inner-loop improving the actuator response and the outer-loop controller using a model of this response to compute optimal control inputs. The parameter estimation portion of the algorithm could also be coupled with an outer-loop predictive method to adapt the prediction model and reduce error.

It could also be valuable to explore applying the algorithm to longitudinal control systems. Throttle and braking actuators can be susceptible to communication delays and dynamic delays similar to steering actuators. Delay compensation could lead to improved longitudinal control performance.

There is also room for improvement in the parameter estimation algorithm. Higher fidelity nonlinear models of vehicle steering systems could be investigated for use in the estimation. Additionally, the relationship between vehicle states (velocity, yaw rate, etc.) and change in the steering response could be explored. Many methods exist to estimate pure time delay. It would be worth while to examine and evaluate the performance of other delay estimation algorithms to compare to the one currently in use.

References

[1] United States. Department of Transportation. Bureau of Transportation Statistics. State Transportation Statistics (STS). 2019. Publisher: Not Available.

[2] Highway Statistics 2021 - Policy | Federal Highway Administration.

[3] Automated Vehicles for Safety | NHTSA.

[4] Santokj Singh. Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey. Technical Report DOT HS 812 506, Nationa Highway Traffic Safety Administration, Washington, DC, March 2018.

[5] SAE Levels of Driving Automation™ Refined for Clarity and International Audience.

[6] Qing Ye, Ruochen Wang, Yinfeng Cai, and Long Chen. Research on modeling and compensation control strategy of automatic steering system. *Science Progress*, 103(1):0036850419875027, January 2020. Publisher: SAGE Publications Ltd.

[7] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, March 2016. Conference Name: IEEE Transactions on Intelligent Vehicles.

[8] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van

Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692, September 2006.

[9] Philip Polack, Florent Altché, Brigitte d'Andréa Novel, and Arnaud de La Fortelle. The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 812–818, June 2017.

[10] Riccardo Marino, Stefano Scalzi, and Mariana Netto. Nested PID steering control for lane keeping in autonomous vehicles. *Control Engineering Practice*, 19(12):1459–1467, December 2011.

[11] Amirreza Mirbeygi Moghaddam, Ali Ghaffari, and Alireza Khodayari. Adaptive comfort-oriented vehicle lateral control with online controller adjustments according to driver behaviour and look-ahead dynamics. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, 234(2):272–287, June 2020. Publisher: SAGE Publications.

[12] H. Atoui, V. Milanés, O. Sename, and John J. Martinez. Real-Time Look-Ahead Distance Optimization for Smooth and Robust Steering Control of Autonomous Vehicles. In *2021 29th Mediterranean Conference on Control and Automation (MED)*, pages 924–929, June 2021. ISSN: 2473-3504.

[13] William Bryan. *Staying Inside the Lines: Vehicle Agnostic Path Following Using Cascaded Adaptive Control*. PhD thesis, Auburn University, 2020.

[14] Ahmad Reda, Ahmed Bouzid, and József Vásárhelyi. Model Predictive Control for Automated Vehicle Steering. *Acta Polytechnica Hungarica*, 17(7):163–182, 2020.

[15] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat . MPC-based approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*, 3(2-4):265–291, January 2005. Publisher: Inderscience Publishers.

[16] O. J. M. Smith. Closer Control of Loops with Dead Time. *Chemical Engineering Progress*, 53:217–219, 1957.

[17] Xuan-Kien Dang, Zhi-Hong Guan, Tao Li, and Ding-Xue Zhang. Joint Smith predictor and neural network estimation scheme for compensating randomly varying time-delay in networked control system. In *2012 24th Chinese Control and Decision Conference (CCDC)*, pages 512–517, May 2012. ISSN: 1948-9447.

[18] A. Bahill. A simple adaptive Smith-predictor for controlling time-delay systems: A tutorial. *IEEE Control Systems Magazine*, 3(2):16–22, May 1983. Conference Name: IEEE Control Systems Magazine.

[19] Jianbo Bai, Shengwei Wang, and Xiaosong Zhang. Development of an adaptive Smith predictor-based self-tuning PI controller for an HVAC system in a test room. *Energy and Buildings*, 40(12):2244–2252, January 2008.

[20] Christopher Thomas, Sun Yi, Shava Meadows, and Ryan Sherrill. Adaptive Smith Predictor for Teleoperation of UAVs with Time-varying Internet Delay. *International Journal of Control, Automation and Systems*, 18(6):1465–1473, June 2020.

[21] Chien-Liang Lai, Pau-Lo Hsu, and Bor-Chyun Wang. Design of the adaptive Smith Predictor for the time-varying network control system. In *2008 SICE Annual Conference*, pages 2933–2938, August 2008.

[22] H. Fujikawa and S. Yamada. A design method of self-tuning Smith predictor for unknown time delay system. In *Control and Instrumentation Proceedings IECON '91: 1991 International Conference on Industrial Electronics*, pages 1801–1806 vol.3, October 1991.

[23] M. Benouarets and D.P. Atherton. Autotuning design methods for a Smith predictor control scheme. In *1994 International Conference on Control - Control '94.*, volume 1, pages 795–800 vol.1, March 1994.

[24] DANIEL L. LAUGHLIN, DANIEL E. RIVERA, and MANFRED MORARI. Smith predictor design for robust performance. *International Journal of Control*, 46(2):477–504, August 1987. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207178708933912.

[25] T. H. Lee, Q. G. Wang, and K. K. Tan. Robust Smith-predictor controller for uncertain delay systems. *AIChE Journal*, 42(4):1033–1040, 1996. _eprint: https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690420415.

[26] M. R. Stojic, F. S. Matijevic, and L. S. Draganovic. A robust Smith predictor modified by internal models for integrating process with dead time. *IEEE Transactions on Automatic Control*, 46(8):1293–1298, August 2001. Conference Name: IEEE Transactions on Automatic Control.

[27] Dongkwon Lee, Moonyong Lee, Suwhan Sung, and Inbeum Lee. Robust PID tuning for Smith predictor in the presence of model uncertainty. *Journal of Process Control*, 9(1):79–85, February 1999.

[28] R. Sanz, P. García, and P. Albertos. A generalized smith predictor for unstable time-delay SISO systems. *ISA Transactions*, 72:197–204, January 2018.

[29] S. Uma and A. Seshagiri Rao. Enhanced modified Smith predictor for second-order non-minimum phase unstable processes. *International Journal of Systems Science*, 47(4):966–981, March 2016. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207721.2014.911385.

[30] Weidong Zhang, Danying Gu, Wei Wang, and Xiaoming Xu. Quantitative Performance Design of a Modified Smith Predictor for Unstable Processes with Time Delay. *Industrial & Engineering Chemistry Research*, 43(1):56–62, January 2004. Publisher: American Chemical Society.

[31] A. Seshagiri Rao and M. Chidambaram. Enhanced Smith Predictor for Unstable Processes with Time Delay. *Industrial & Engineering Chemistry Research*, 44(22):8291–8299, October 2005. Publisher: American Chemical Society.

[32] Asal Nahidi, Amir Khajepour, Alireza Kasaeizadeh, Shih-Ken Chen, and Bakhtiar Litkouhi. A study on actuator delay compensation using predictive control technique with experimental verification. *Mechatronics*, 57:140–149, February 2019.

[33] Dvij Kalaria, Qin Lin, and John M. Dolan. Delay-aware Robust Control for Safe Autonomous Driving. *arXiv:2109.07101 [cs]*, November 2021. arXiv: 2109.07101.

[34] M. Wang, S. P. Hoogendoorn, W. Daamen, B. van Arem, B. Shyrokau, and R. Happee. Delay-compensating strategy to enhance string stability of adaptive cruise controlled vehicles. *Transportmetrica B: Transport Dynamics*, 6(3):211–229, July 2018.

[35] Yousef Alipouri, Hasan Alipour, and Biao Huang. Multiple step ahead prediction based high order discrete-time sliding mode control design with actuator and communication delays. *Journal of the Franklin Institute*, 357(12):7845–7863, August 2020.

[36] Y. Tipsuwan and Mo-Yuen Chow. Gain scheduler middleware: a methodology to enable existing controllers for networked control and teleoperation - part I: networked control. *IEEE Transactions on Industrial Electronics*, 51(6):1218–1227, December 2004. Conference Name: IEEE Transactions on Industrial Electronics.

[37] N.B. Loden and J.Y. Hung. An adaptive PID controller for network based control systems. In *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, pages 6 pp.–, November 2005. ISSN: 1553-572X.

[38] S. Bjorklund and L. Ljung. A review of time-delay estimation techniques. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 3, pages 2502–2507 Vol.3, December 2003. ISSN: 0191-2216.

[39] R. Pupeikis. Recursive Estimation of the Parameters of Linear Systems with Time Delay. *IFAC Proceedings Volumes*, 18(5):787–792, July 1985.

[40] G. Ferretti, C. Maffezzoni, and R. Scattolini. Recursive estimation of time delay in sampled systems. *Automatica*, 27(4):653–661, July 1991.

[41] A. Elnaggar, G.A. Dumont, and A.-L. Elshafei. Recursive estimation for system of unknown delay. In *Proceedings of the 28th IEEE Conference on Decision and Control,*, pages 1809–1810 vol.2, December 1989.

[42] Gang Zheng, Andrey Polyakov, and Arie Levant. Delay estimation via sliding mode for nonlinear time-delay systems. *Automatica*, 89:266–273, March 2018.

[43] Haitao Xing, Jeroen Ploeg, and Henk Nijmeijer. Smith Predictor Compensating for Vehicle Actuator Delays in Cooperative ACC Systems. *IEEE Transactions on Vehicular Technology*, 68(2):1106–1115, February 2019. Conference Name: IEEE Transactions on Vehicular Technology.

[44] D. Yanakiev and I. Kanellakopoulos. Longitudinal control of automated CHVs with significant actuator delays. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 5, pages 4756–4763 vol.5, December 1997. ISSN: 0191-2216.

[45] Sungyoul Park, Beomjun Kim, Kyuwon Kim, Youngseop Son, and Kyongsu Yi. Time delay compensation for environmental sensors of high-level automated driving systems. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 555–560, June 2015. ISSN: 1931-0587.

[46] Jun Yao, Jin Qiu Zhang, Ming Mei Zhao, and Xin Li. Adaptive control of a nonlinear suspension with time-delay compensation. *Journal of Vibroengineering*, 21(3):684–695, 2019. Number: 3 Publisher: JVE International Ltd.

[47] Aziz Sezgin, Yuksel Hacioglu, and Nurkan Yagiz. Sliding Mode Control for Active Suspension System with Actuator Delay. *International Journal of Mechanical and Mechatronics Engineering*, 10(8):1456–1460, June 2016.

[48] Constantin Caruntu. Networked Predictive Control for Time-varying Delay Compensation with an Application to Automotive Mechatronic Systems.

[49] Sergio Trimboli, Stefano Di Cairano, Alberto Bemporad, and Ilya V. Kolmanovsky. Model predictive control for automotive time-delay processes: An application to air-to-fuel ratio control*. *IFAC Proceedings Volumes*, 42(14):90–95, 2009.

[50] Kenta Tominaga, Yu Takeuchi, Hiroaki Kitano, Uno Tomoki, Rien Quirynen, and Stefano Cairano. GNSS-Based Lane Keeping Assist System Using Model Predictive Control and Time Delay Compensation. SAE Technical Paper 2020-01-1023, SAE International, Warrendale, PA, April 2020. ISSN: 0148-7191, 2688-3627.

[51] Nazli E. Kahveci and Petros A. Ioannou. Automatic steering of vehicles subject to actuator saturation and delay. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 119–124, October 2011. ISSN: 2153-0017.

[52] Shaobing Xu, Huei Peng, and Yifan Tang. Preview Path Tracking Control With Delay Compensation for Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):2979–2989, May 2021.

[53] William Thomas Kennedy and David M. Bevly. Adaptive Actuator Delay Compensation for a Vehicle Lateral Control System. pages 2023–01–0677, Detroit, Michigan, United States, April 2023.

[54] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback control of dynamic systems*. Pearson, Ny, NY, eighth edition edition, 2019.

[55] William Palm. *System Dynamics*. McGraw-Hill, 3 edition, 2014.

[56] G.J. Silva, A. Datta, and S.P. Bhattacharyya. Controller design via Pade approximation can lead to instability. In *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, volume 5, pages 4733–4737 vol.5, December 2001.

[57] Coordinate Systems in Vehicle Dynamics Blockset - MATLAB & Simulink.

[58] Comparison of 3-D Coordinate Systems - MATLAB & Simulink.

[59] Zeng Qun and Huang Juhua. Modeling and Simulation of the Electric Power Steering System. In *2009 Pacific-Asia Conference on Circuits, Communications and Systems*, pages 236–239, May 2009.

[60] W. Michiels and S.-I. Niculescu. On the delay sensitivity of Smith Predictors. *International Journal of Systems Science*, 34(8-9):543–551, July 2003. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207720310001609057.

[61] C. Santacesaria and R. Scattolini. Easy tuning of smith predictor in presence of delay uncertainty. *Automatica*, 29(6):1595–1597, November 1993.

[62] Weining Feng. On practical stability of linear multivariable feedback systems with time-delays. *Automatica*, 27(2):389–394, March 1991.

[63] Richard Wallace, Anthony Stentz, Charles Thorpe, Hans Maravec, William Whittaker, and Takeo Kanade. *First Results in Robot Road-Following*. January 1985. Pages: 1095.

[64] Fitri Yakub and Yasuchika Mori. Comparative study of autonomous path-following vehicle control via model predictive control and linear quadratic control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 229(12):1695–1714, October 2015. Publisher: IMECHE.

[65] Liuping Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. Advances in Industrial Control. Springer, 2009.

Appendices

Appendix A

Additional Monte Carlo Data

A.1 Additional 10 m/s Data

The following sections provide the steering angle errors of the Monte Carlo simulations averaged accross all 100 runs. This data is from the 10 m/s simulation. The data is provided for each of the 4 controllers.

A.1.1 Discrete Heading Controller

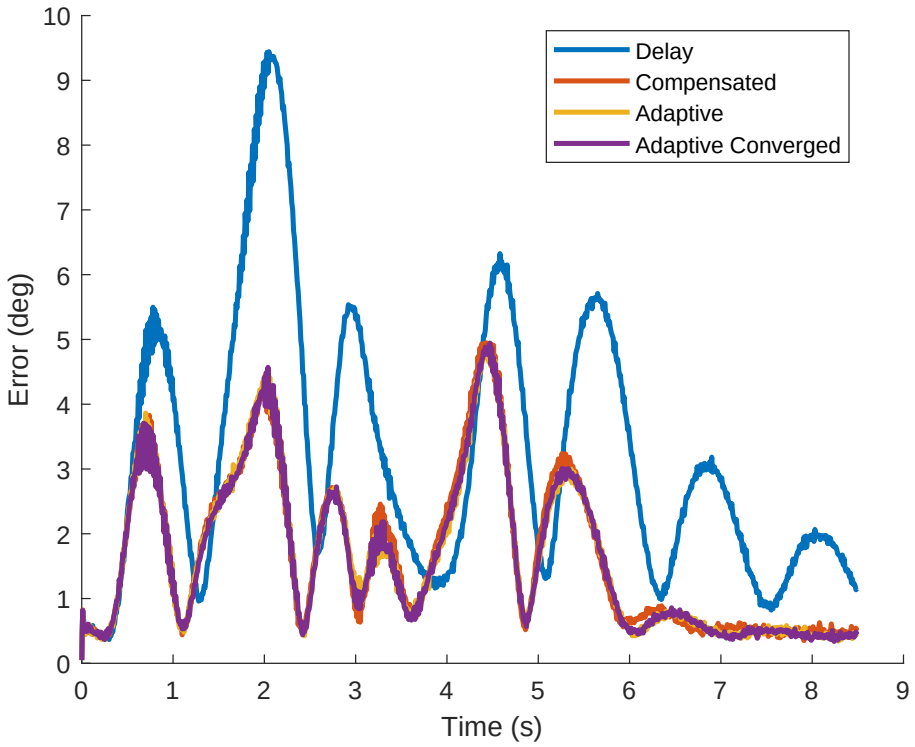


Figure A.1: Heading Controller Steer Angle Error, 10 m/s

A.1.2 Pure Pursuit Controller

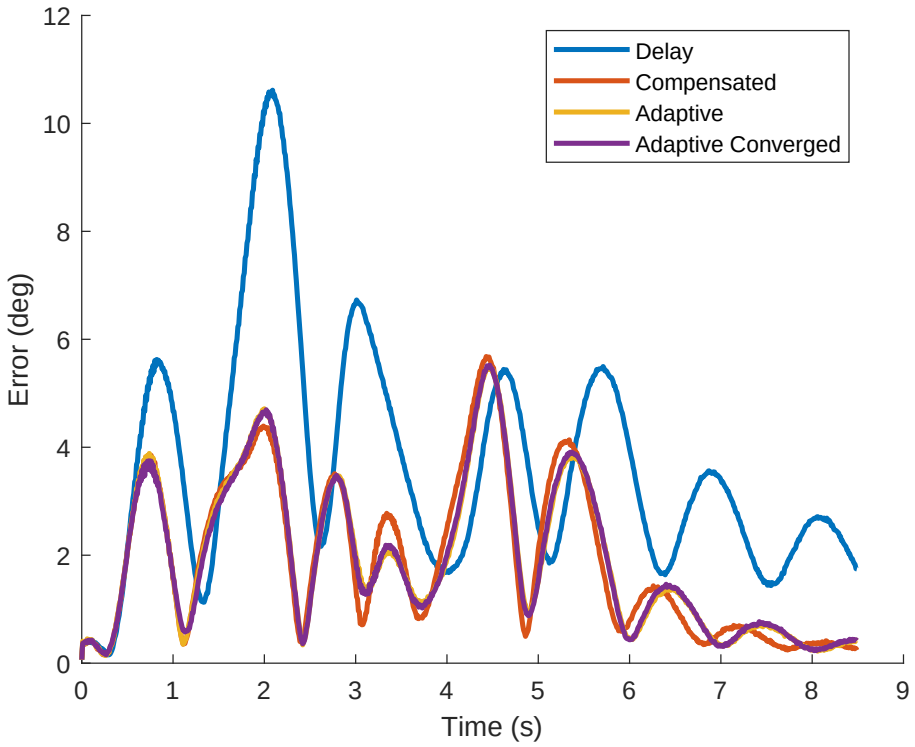


Figure A.2: Pure Pursuit Controller Steer Angle Error, 10 m/s

A.1.3 State Feedback Controller

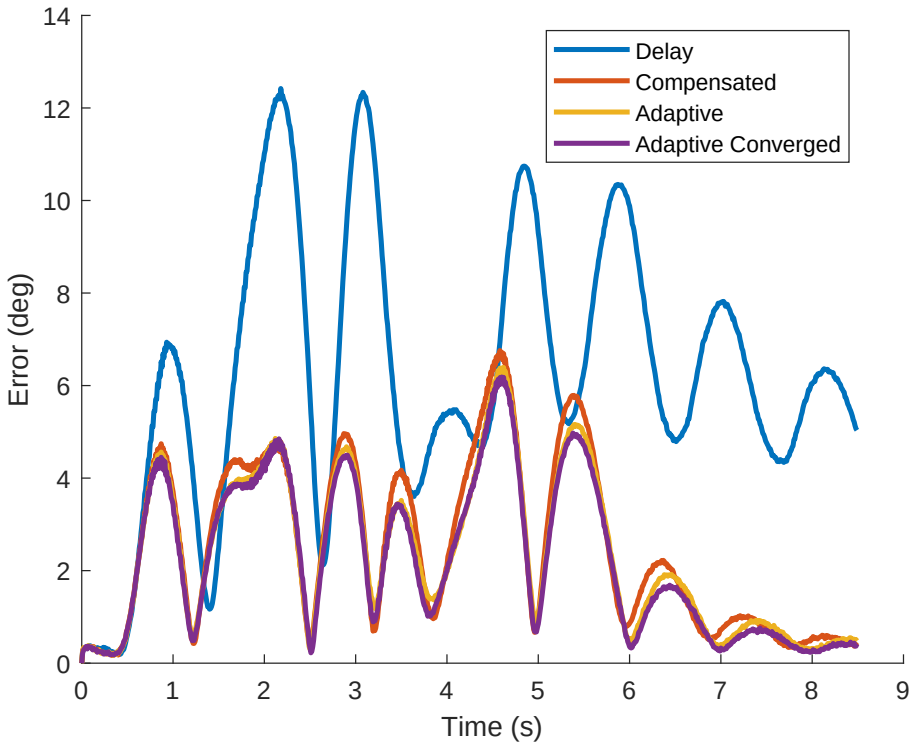


Figure A.3: State Feedback Controller Steer Angle Error, 10 m/s

A.1.4 MPC

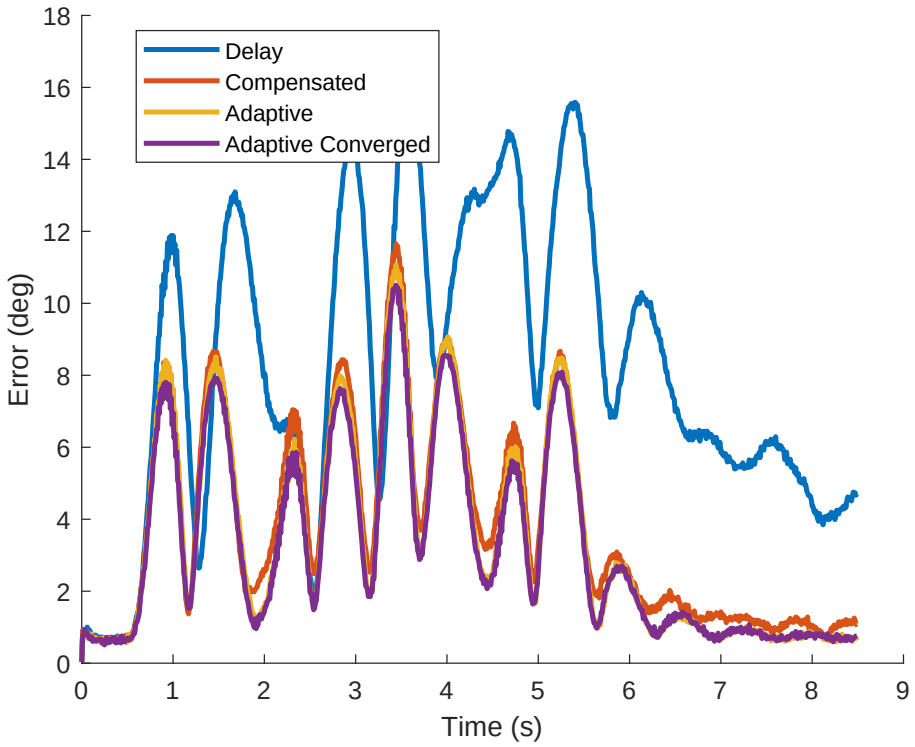


Figure A.4: MPC Controller Steer Angle Error, 10 m/s

A.2 15 m/s Data

A.2.1 Discrete Heading Controller

This section provides the data from the 15 m/s Monte Carlo simulation of the discrete heading contoller. The error statistics, path tracking performance, parameter estimation performance, prediction errors, and steer angle errors are shown in the figures below.

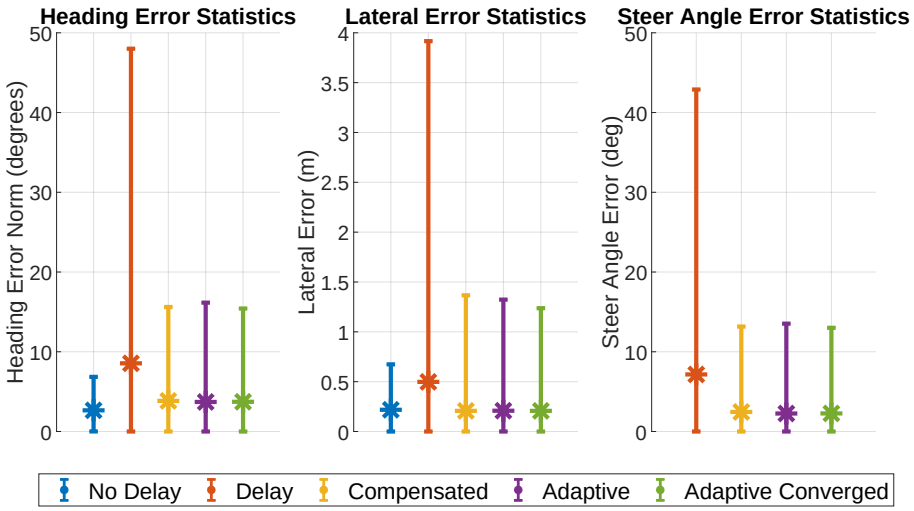


Figure A.5: Heading Controller Error Statistics, 15 m/s

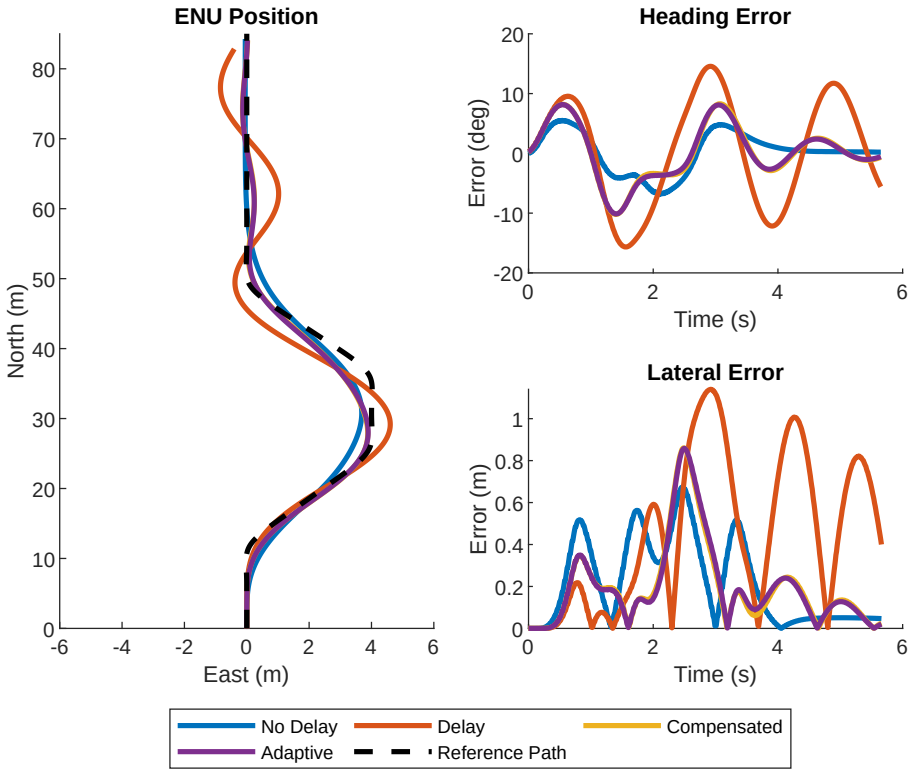


Figure A.6: Heading Controller Double Lane Change Performance, 15 m/s

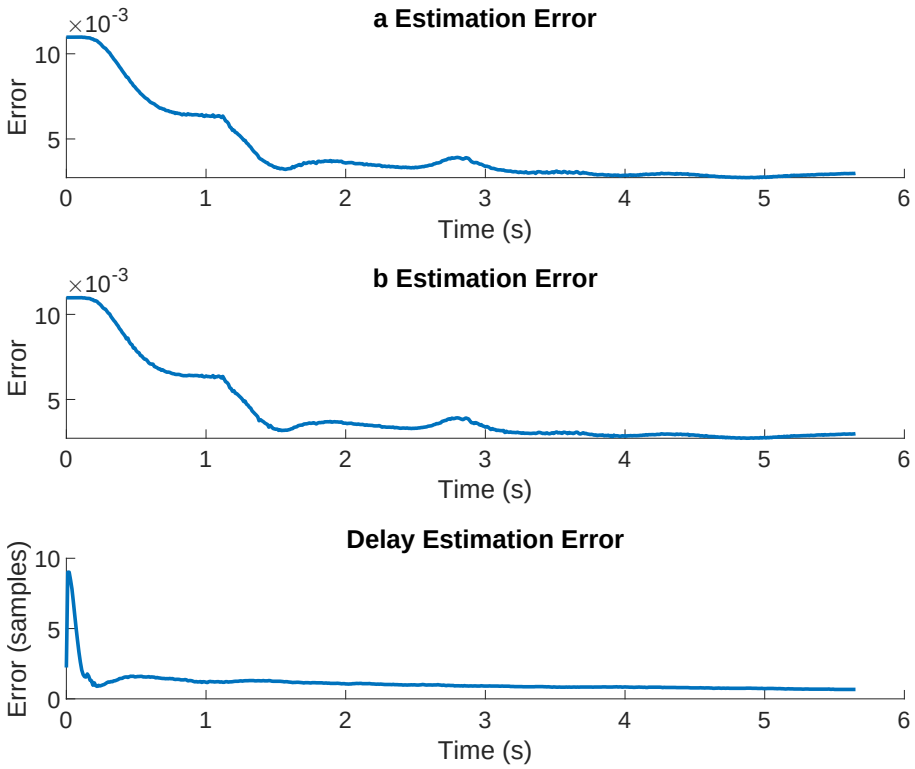


Figure A.7: Heading Controller Parameter Estimation Errors, 15 m/s

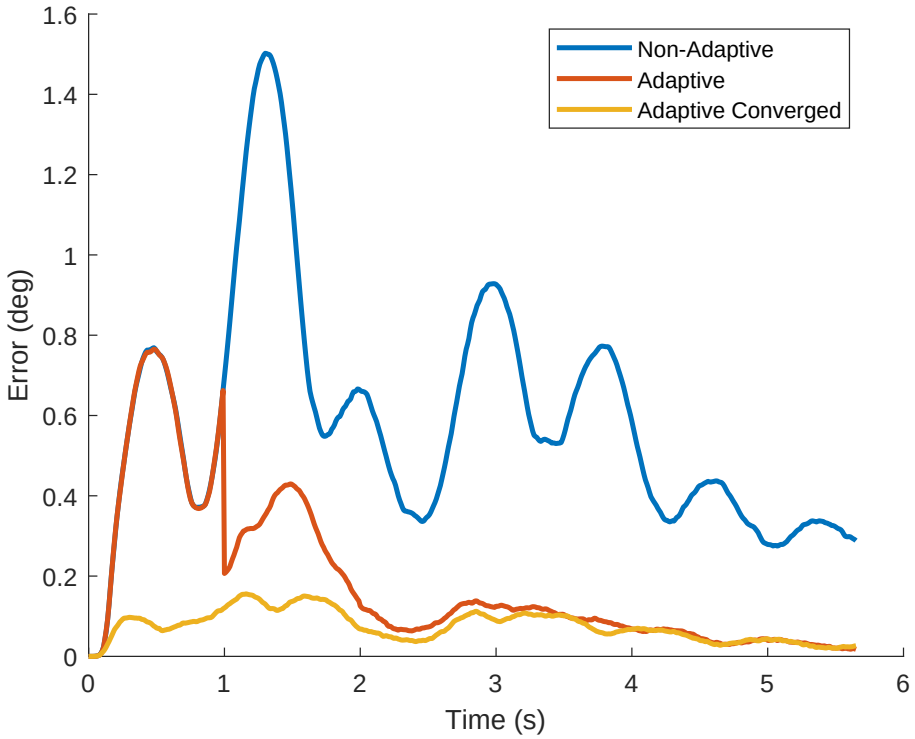


Figure A.8: Heading Controller Steer Angle Prediction Errors, 15 m/s

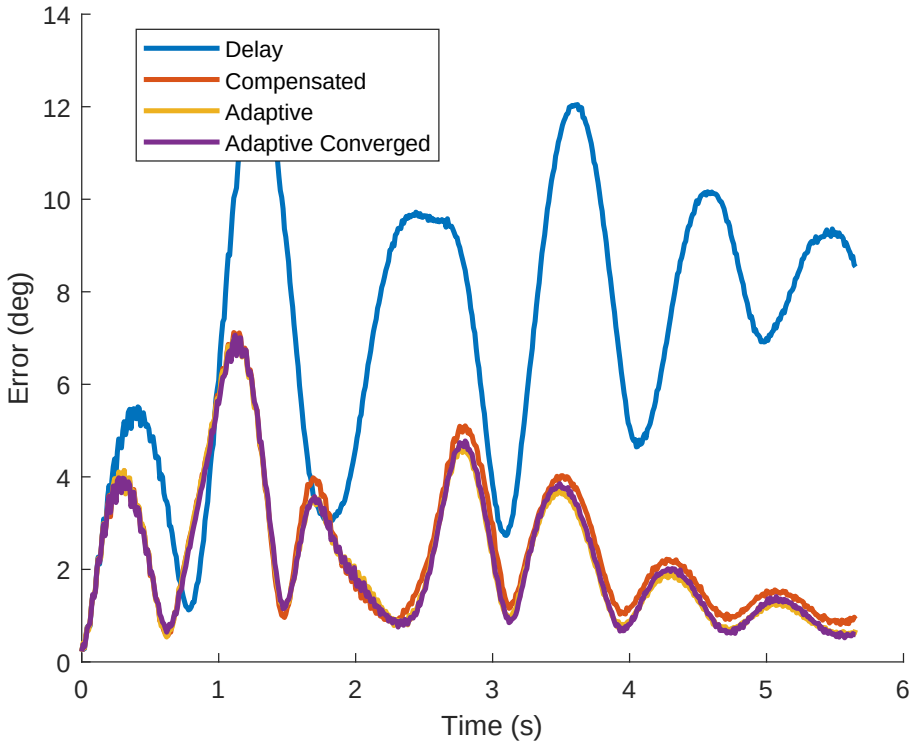


Figure A.9: Heading Controller Steer Angle Error, 15 m/s

A.2.2 Pure Pursuit Controller

This section provides the data from the 15 m/s Monte Carlo simulation of the pure pursuit contoller. The error statistics, path tracking performance, parameter estimation performance, prediction errors, and steer angle errors are shown in the figures below.

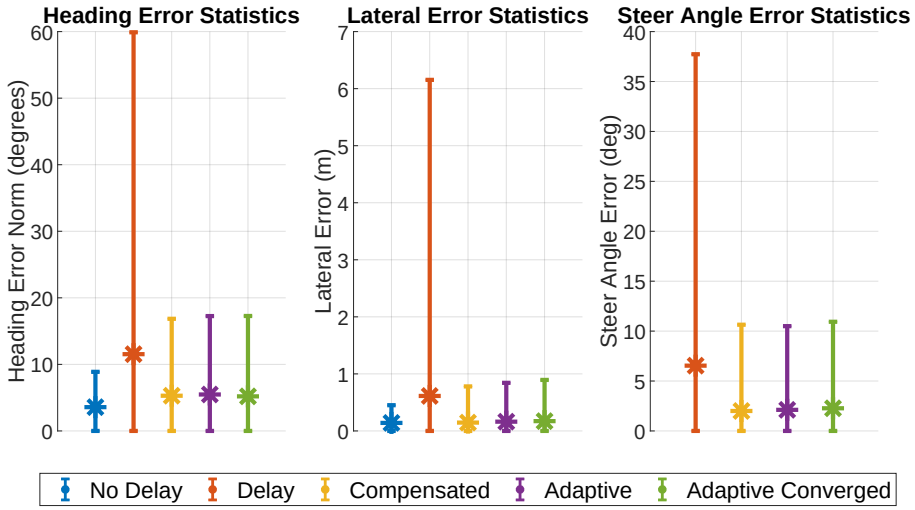


Figure A.10: Pure Pursuit Kinematic Controller Error Statistics, 15 m/s

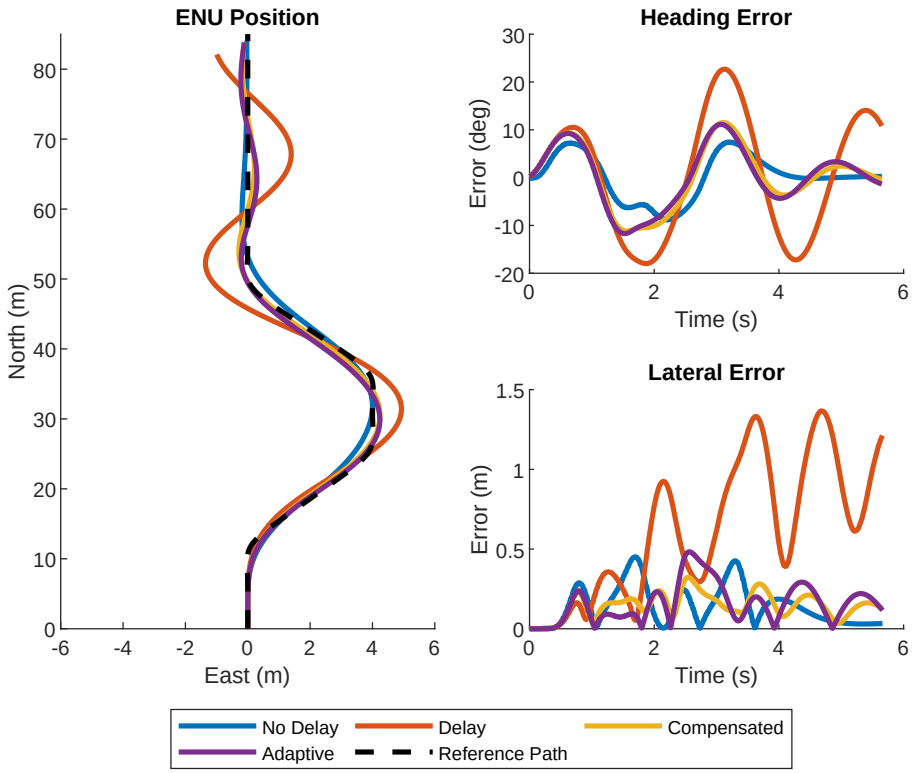


Figure A.11: Pure Pursuit Kinematic Controller Double Lane Change, 15 m/s

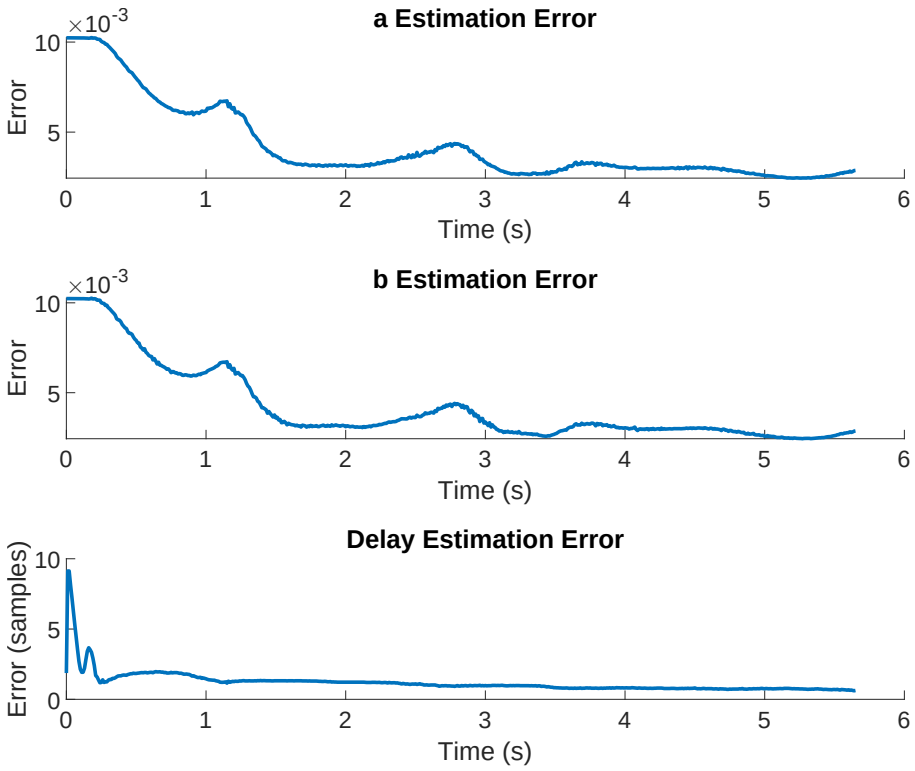


Figure A.12: Pure Pursuit Kinematic Controller Parameter Estimation Errors, 15 m/s

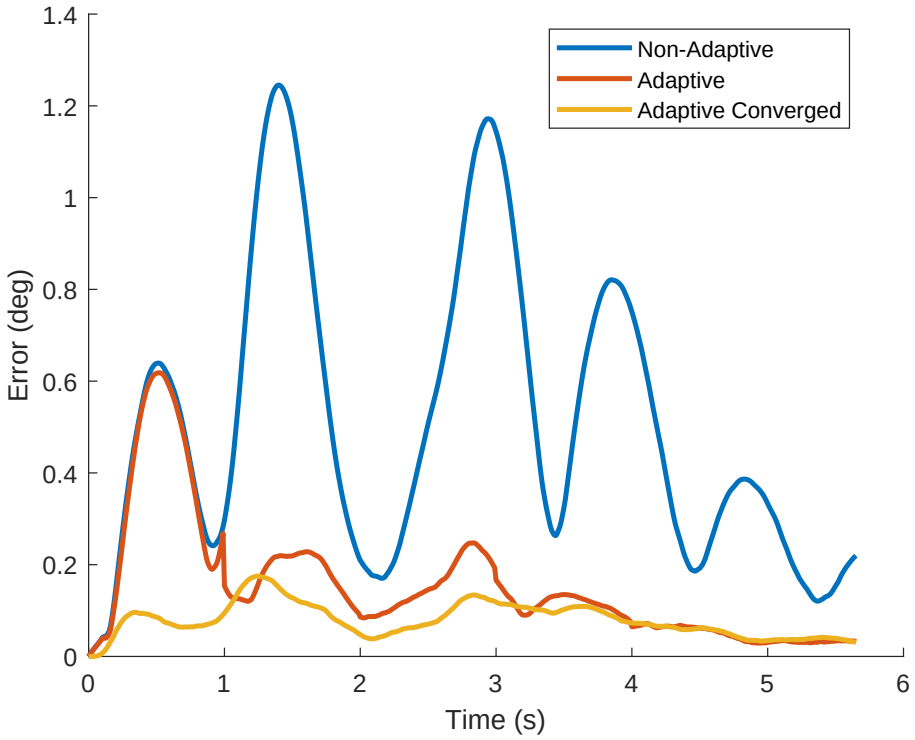


Figure A.13: Pure Pursuit Kinematic Controller Steer Angle Prediction Errors, 15 m/s

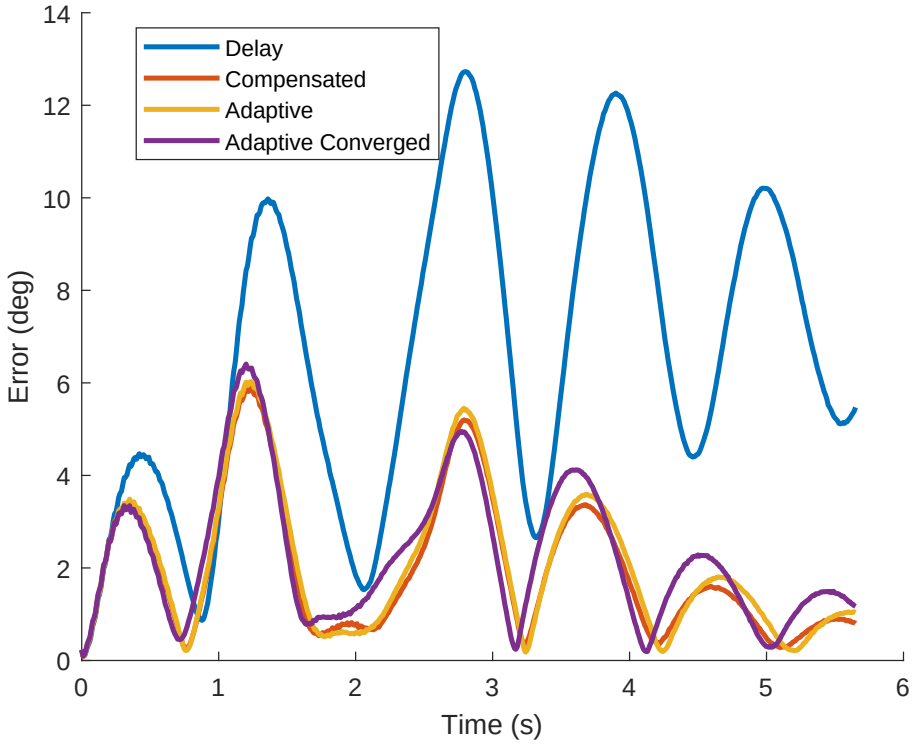


Figure A.14: Pure Pursuit Controller Steer Angle Error, 15 m/s

A.2.3 State Feedback Controller

This section provides the data from the 15 m/s Monte Carlo simulation of the state feedback contoller. The error statistics, path tracking performance, parameter estimation performance, prediction errors, and steer angle errors are shown in the figures below.

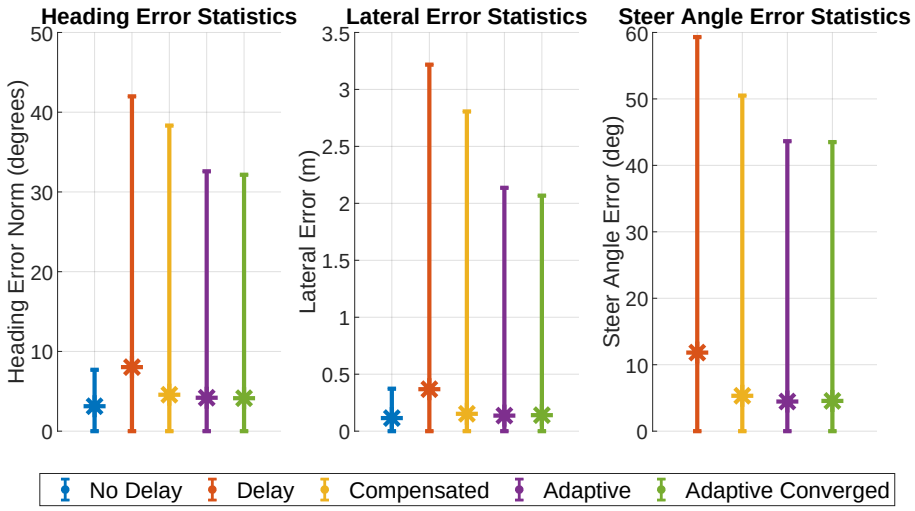


Figure A.15: State Feedback Controller Error Statistics, 15 m/s

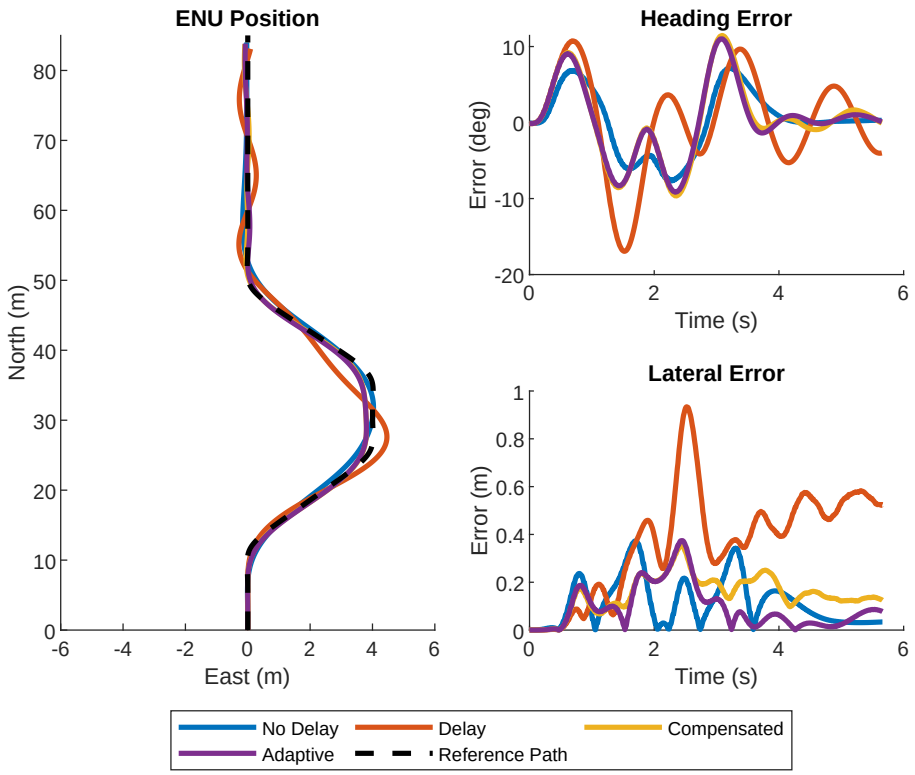


Figure A.16: State Feedback Controller Double Lane Change Performance, 15 m/s

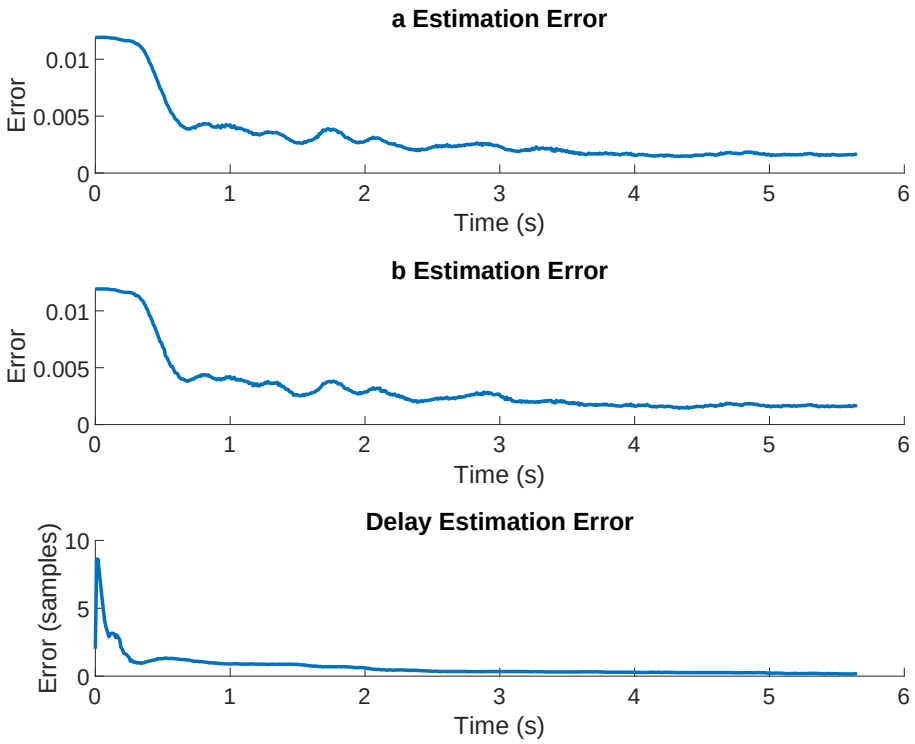


Figure A.17: State Feedback Controller Parameter Estimation Errors, 15 m/s

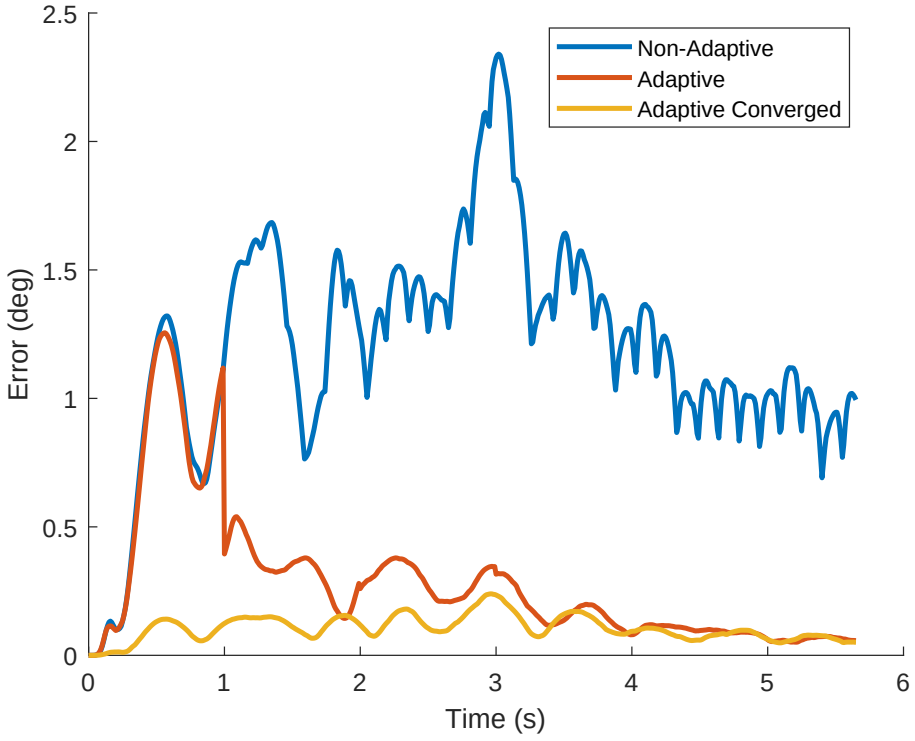


Figure A.18: State Feedback Controller Steer Angle Prediction Errors, 15 m/s

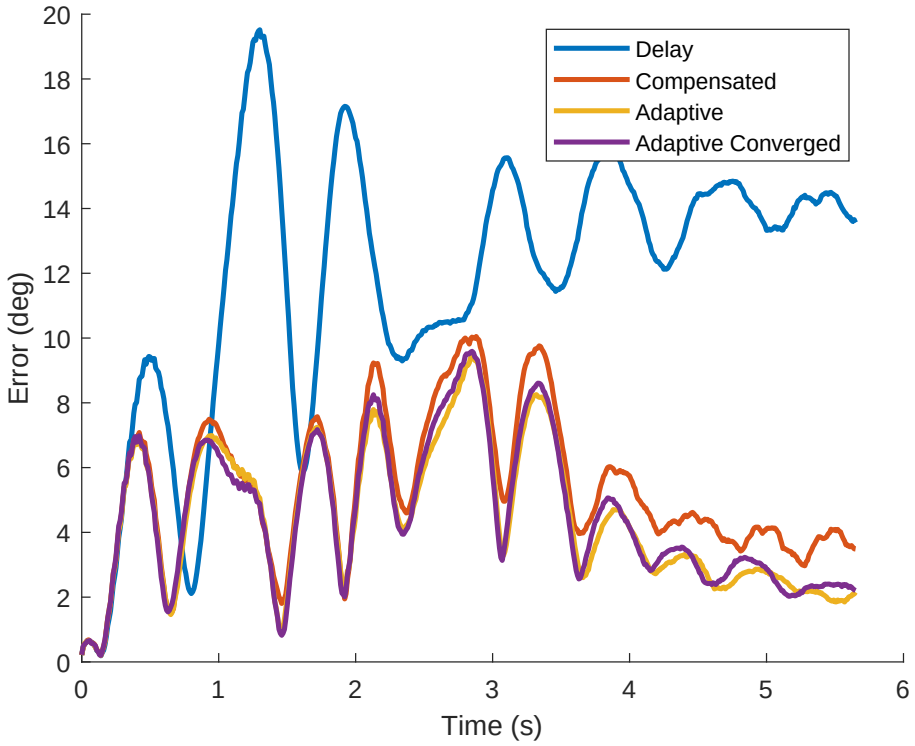


Figure A.19: State Feedback Controller Steer Angle Error, 15 m/s

A.2.4 MPC

This section provides the data from the 15 m/s Monte Carlo simulation of the MPC contoller. The error statistics, path tracking performance, parameter estimation performance, prediction errors, and steer angle errors are shown in the figures below.

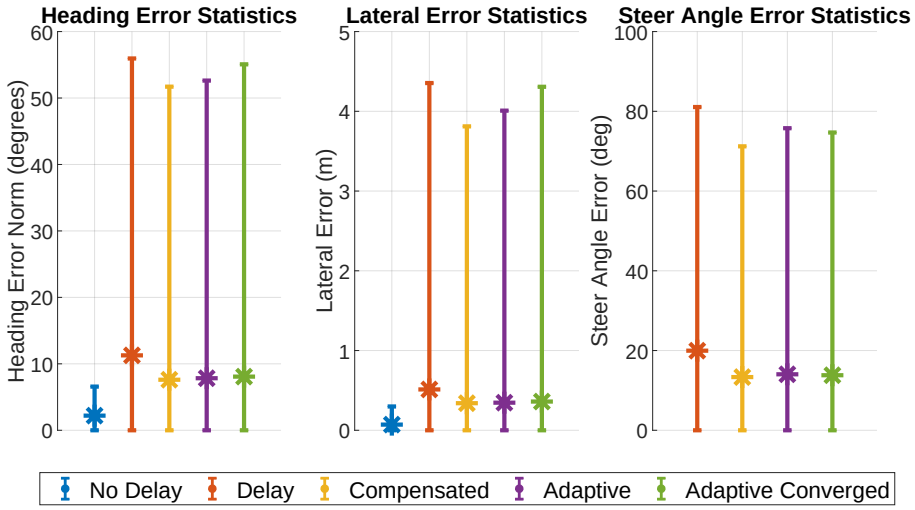


Figure A.20: MPC Double Lane Change Error Statistics, 15 m/s

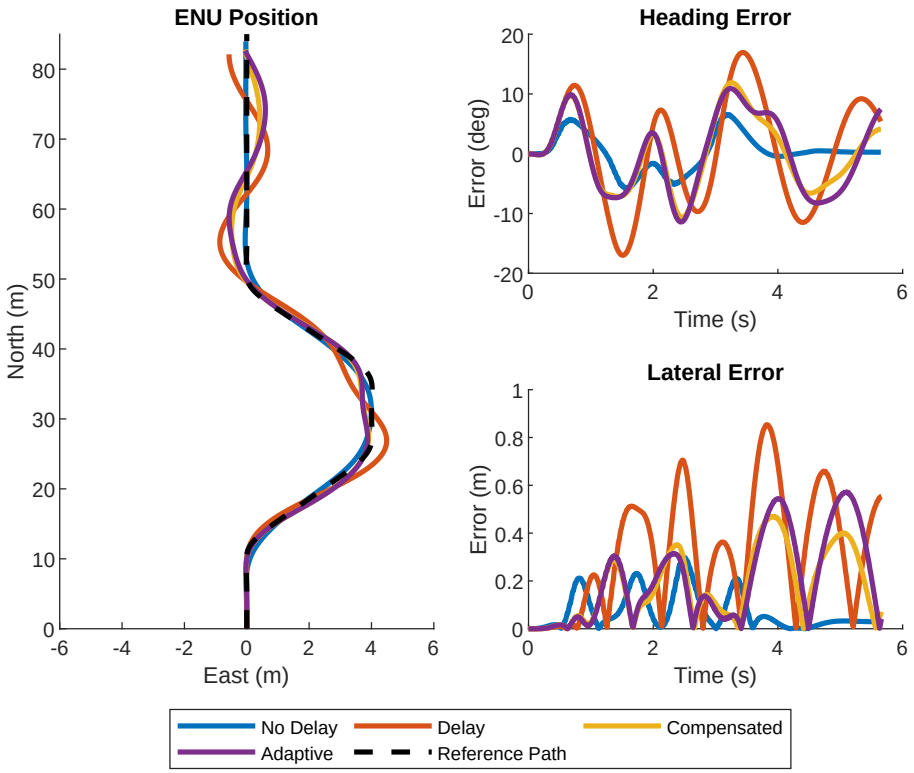


Figure A.21: MPC Double Lane Change Performance, 15 m/s

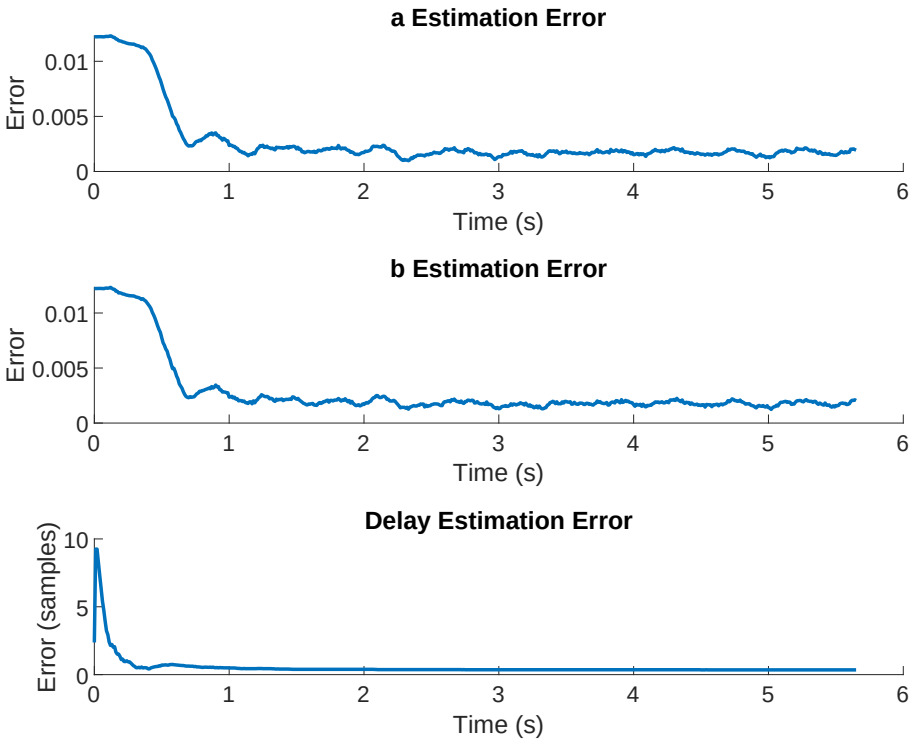


Figure A.22: MPC Parameter Estimation Errors, 15 m/s

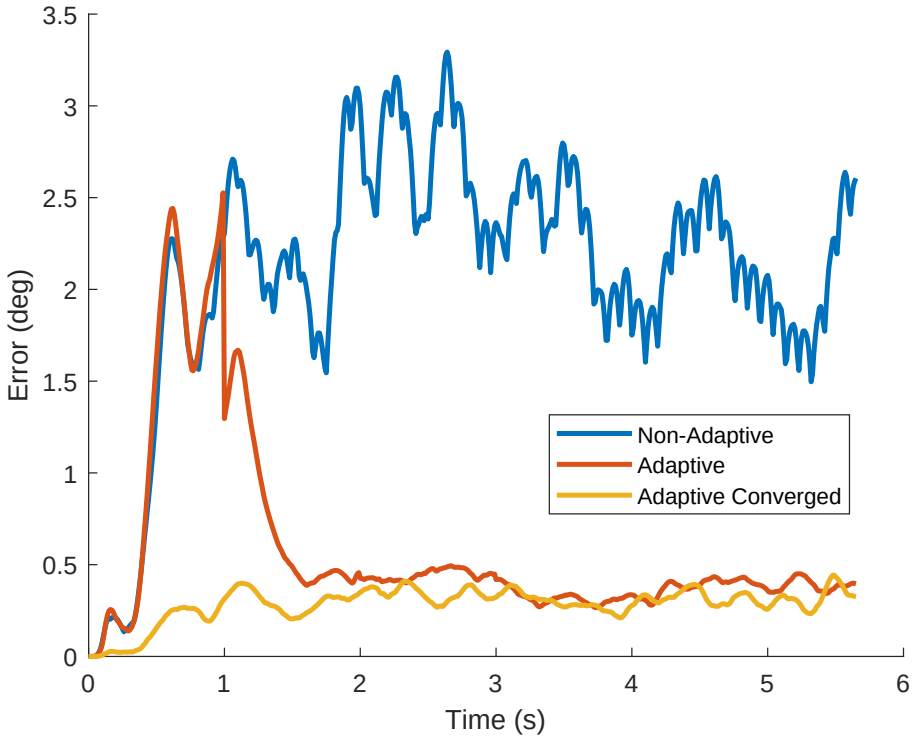


Figure A.23: MPC Steer Angle Prediction Errors, 15 m/s

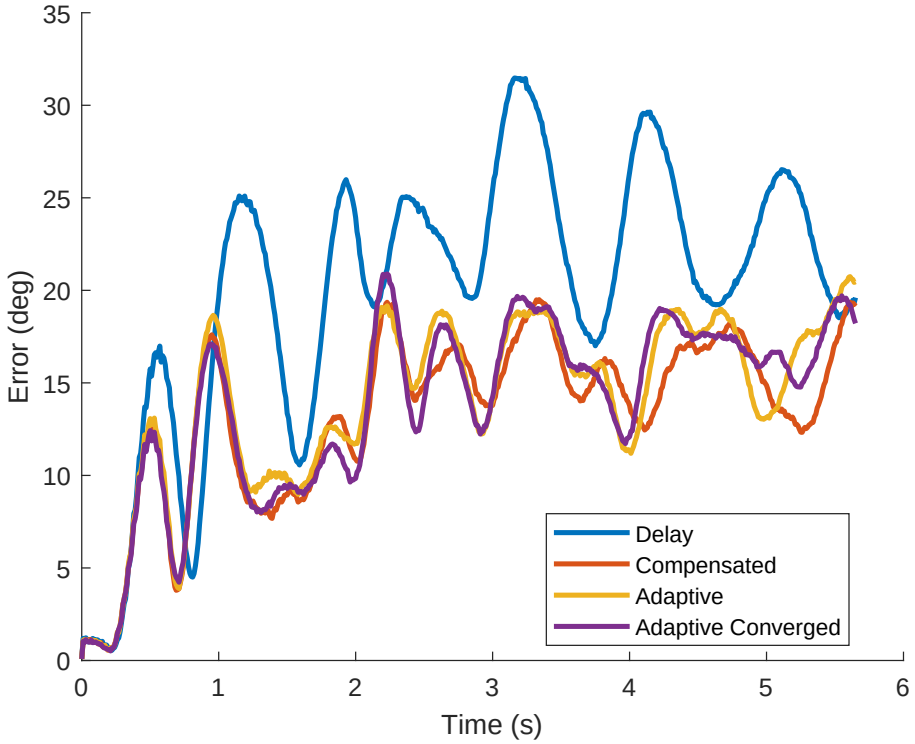


Figure A.24: MPC Controller Steer Angle Error, 15 m/s

Appendix B

Additional MKZ Data

This appendix contains additional data from the real time tests conducted using the MKZ test vehicle. Additional steering response data from both the double lane change and slalom runs are provided.

B.1 Double Lane Change Maneuver

The figures in this section show the MKZ steering response as well as the inner-loop and outer-loop commands for each of the double lane change datasets. The descriptions of the setup for each of the runs is shown in Table 6.1.

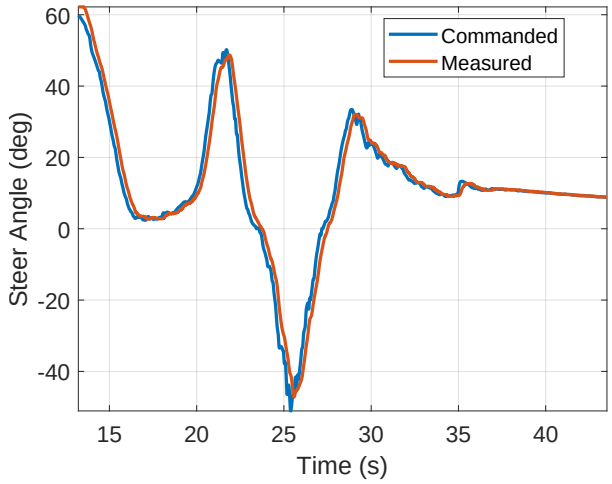
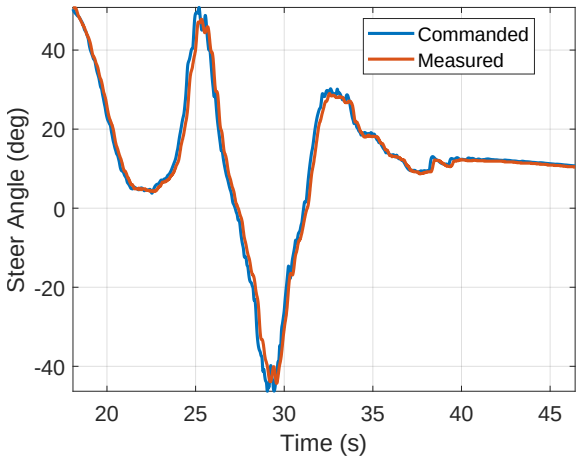
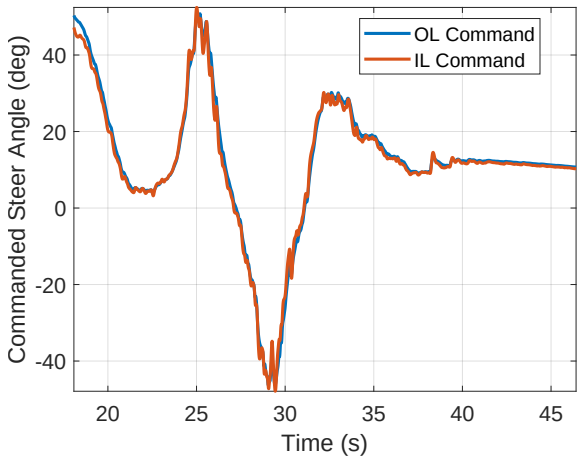


Figure B.1: Commanded and Measured Steer Angle: Double Lane Change Run 1

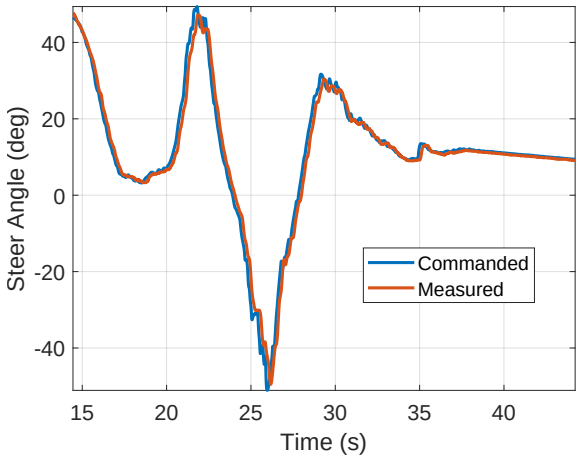


(a) Commanded and Measured Steer Angle

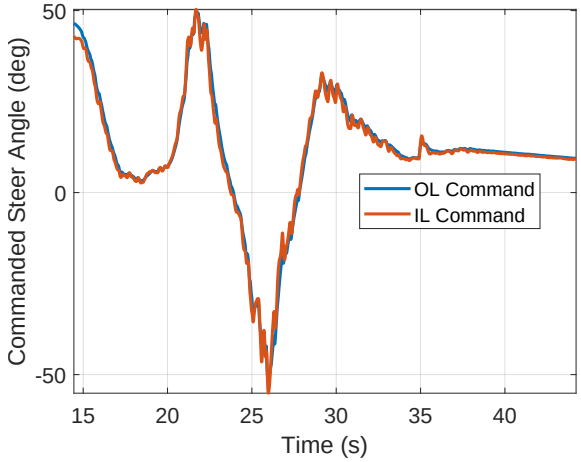


(b) Inner and Outer-Loop Commands

Figure B.2: Steering Response and Controller Inputs: Double Lane Change Run 2



(a) Commanded and Measured Steer Angle



(b) Inner and Outer-Loop Commands

Figure B.3: Steering Response and Controller Inputs: Double Lane Change Run 3

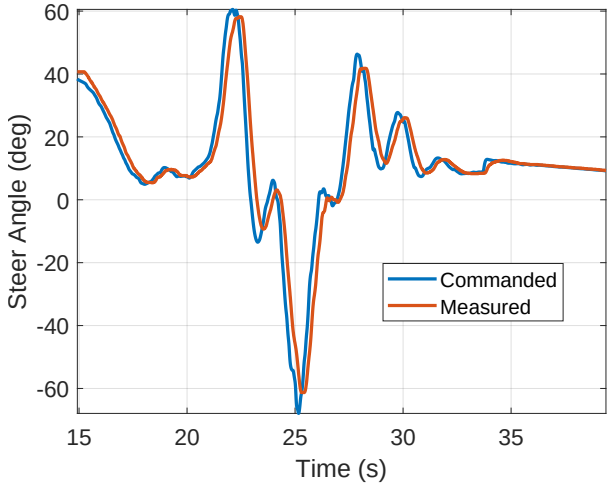
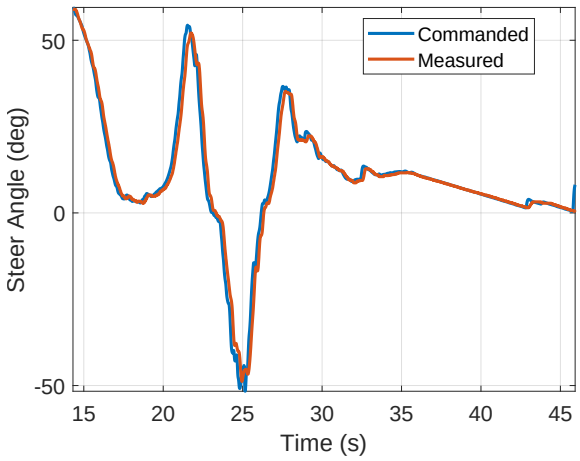
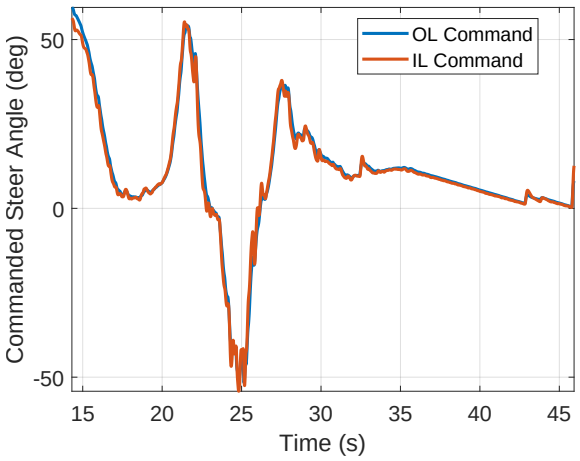


Figure B.4: Commanded and Measured Steer Angle: Double Lane Change Run 4

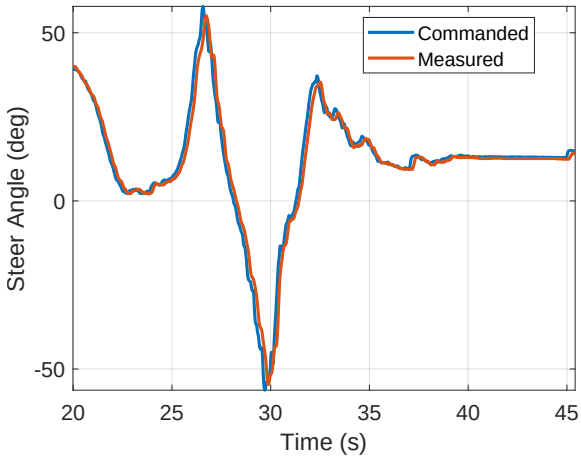


(a) Commanded and Measured Steer Angle

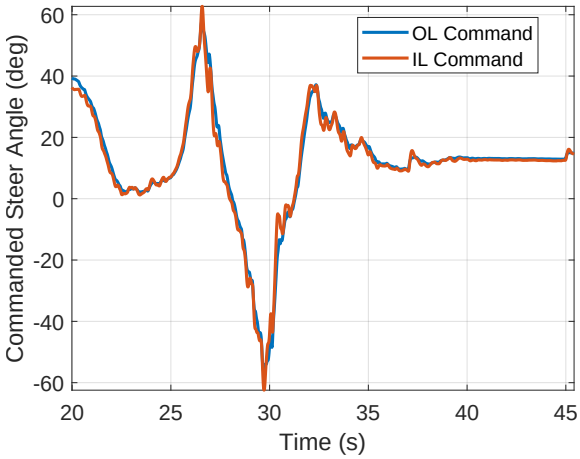


(b) Inner and Outer-Loop Commands

Figure B.5: Steering Response and Controller Inputs: Double Lane Change Run 5



(a) Commanded and Measured Steer Angle



(b) Inner and Outer-Loop Commands

Figure B.6: Steering Response and Controller Inputs: Double Lane Change Run 6

B.2 Slalom Maneuver

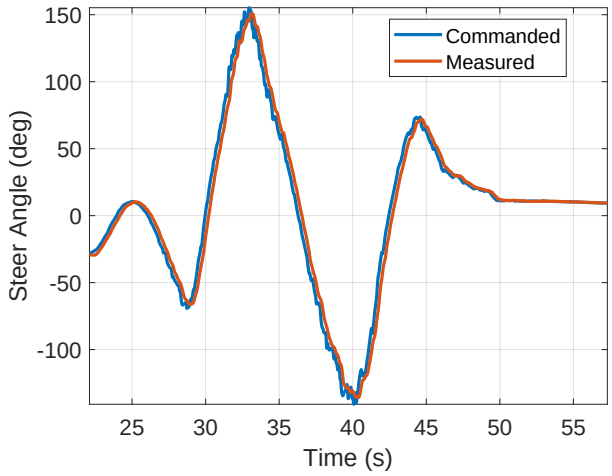
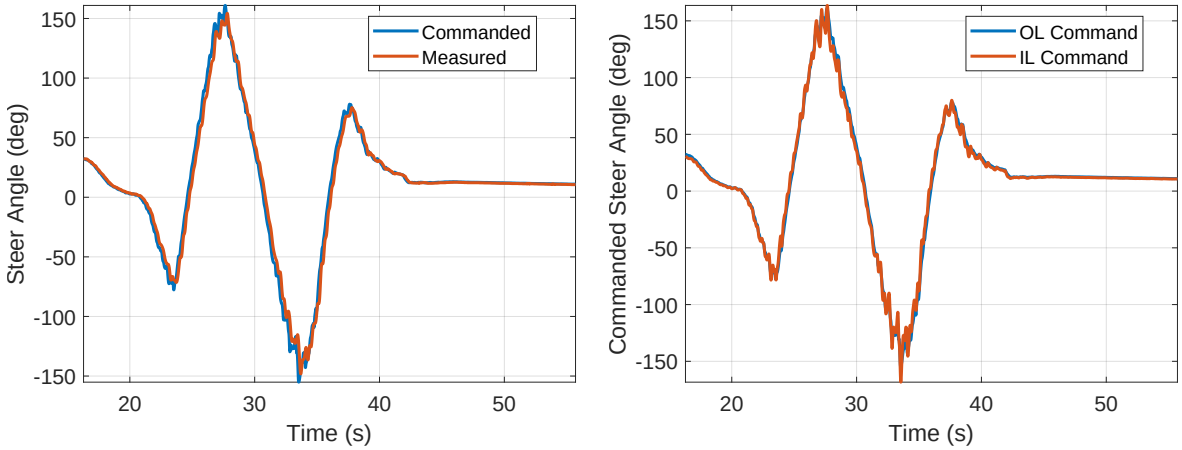


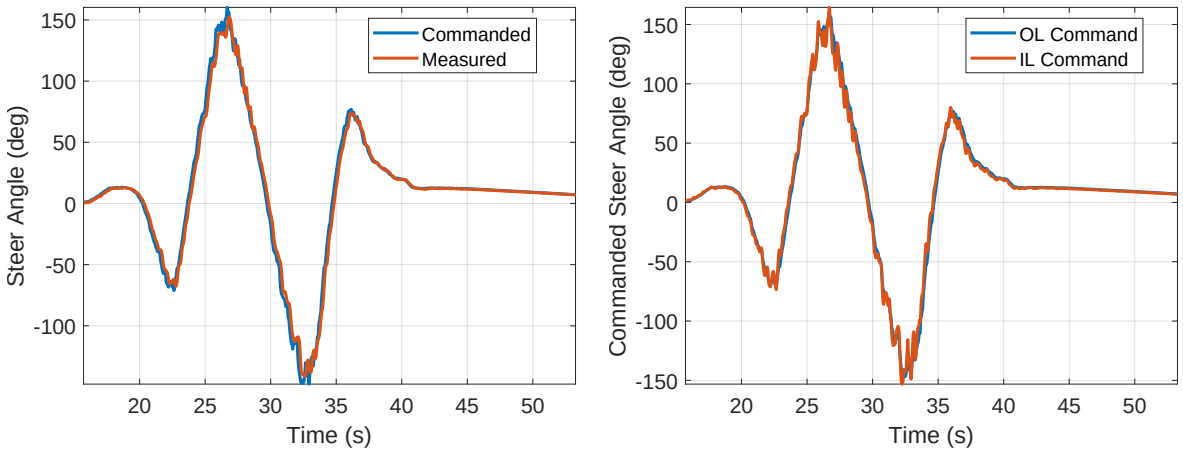
Figure B.7: Commanded and Measured Steer Angle: Slalom Run 1



(a) Commanded and Measured Steer Angle

(b) Inner and Outer-Loop Commands

Figure B.8: Steering Response and Controller Inputs: Slalom Run 2



(a) Commanded and Measured Steer Angle (b) Inner and Outer-Loop Commands

Figure B.9: Steering Response and Controller Inputs: Slalom Run 3

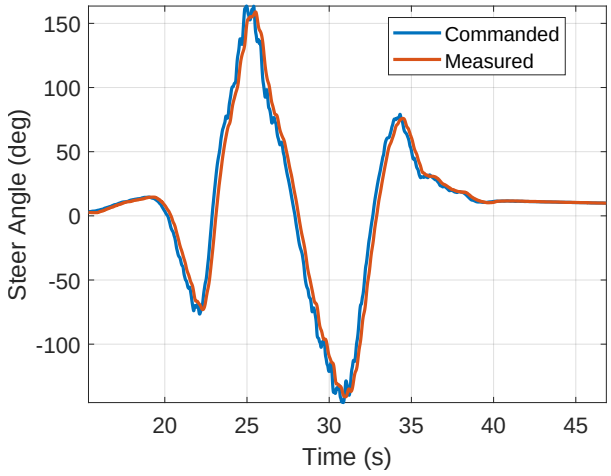
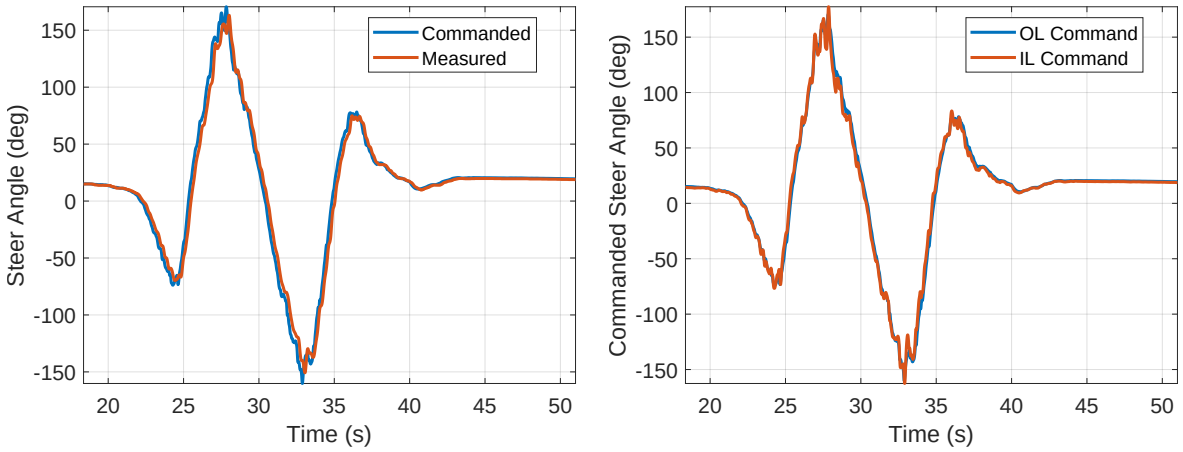
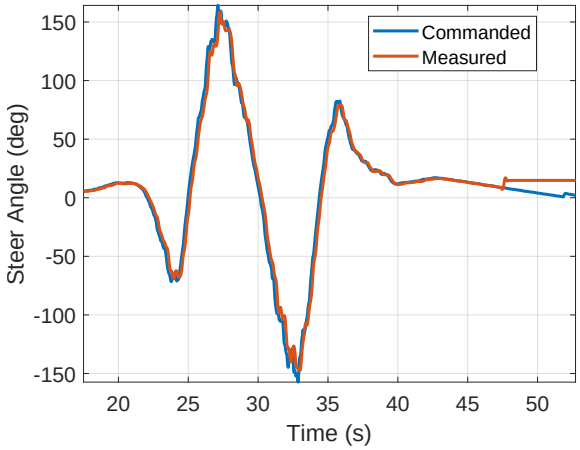


Figure B.10: Commanded and Measured Steer Angle: Slalom Run 4

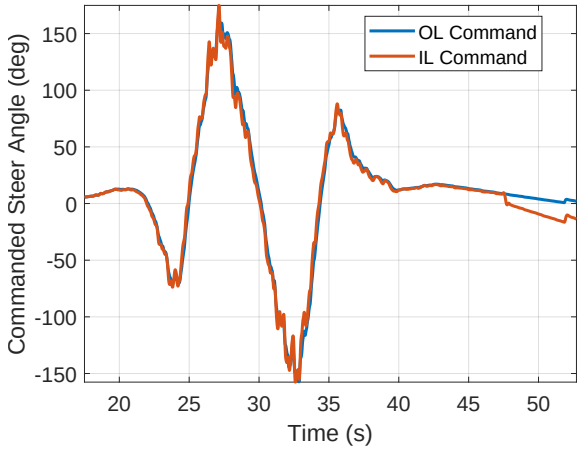


(a) Commanded and Measured Steer Angle (b) Inner and Outer-Loop Commands

Figure B.11: Steering Response and Controller Inputs: Slalom Run 5



(a) Commanded and Measured Steer Angle



(b) Inner and Outer-Loop Commands

Figure B.12: Steering Response and Controller Inputs: Slalom Run 6