

REAL TIME IDENTIFICATION OF CORNERING COEFFICIENTS AND IDEAL TWIN DRIVING ASSISTANCE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

By
Ahmet Faruk KELEŞ

Agust 2024

Real Time Identification Of Cornering Coefficients and Ideal Twin
Driving Assistance
By Ahmet Faruk KELEŞ
Agust 2024

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

İlker Temizer(Advisor)

Melih Çakmakçı(Co-Advisor)

Gökberk Kabacaoğlu

İsmail Uyanık

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

REAL TIME IDENTIFICATION OF CORNERING COEFFICIENTS AND IDEAL TWIN DRIVING ASSISTANCE

Ahmet Faruk KELEŞ
M.S. in Mechanical Engineering
Advisor: İlker Temizer
August 2024

Cornering coefficients play a crucial role in vehicles' lateral and longitudinal dynamics. They depend on many factors, including environmental factors such as road conditions. In this study, a method that identifies cornering coefficients in real-time by utilizing deep neural networks is developed. Three different neural network architectures with two different datasets are compared for this identification. Results show that a fully connected network trained with time-varying cornering coefficients performs best. Compared to constant cornering coefficients, this method improves the lateral force estimation between 42.62-75.47 % in experiments conducted on a 1/8 scale four-wheel drive four-wheel steering vehicle. A control method that utilizes the identified cornering coefficients to cancel out the changes in cornering coefficient by utilizing 4 wheel drive 4 wheel steering system is developed. The control method utilizes a nonlinear model predictive controller. The control system uses the driver's control inputs in an ideal front wheel drive front wheel steering twin of the vehicle with constant cornering coefficients set by the driver, to obtain reference velocities. The nonlinear model predictive controller can calculate the optimal control inputs for a 4-wheel drive, 4-wheel steering vehicle from these references. The simulation results show that this control method improves reference tracking by 85 % compared to a conventional configuration, i.e., a front wheel drive front wheel steering vehicle. The controller is tested on an experimental setting and found to be improving the results by 12.12 %. The reduction in improvement can be attributed to noise in measurements and delays in the control system.

Keywords: Vehicle Parameter Identification, Lateral Stability Control.

ÖZET

GERÇEK ZAMANL VIRAJ KATSAYS TEHŞISI VE DEAL KIZ ÖFÖR DESTEK SISTEMI

Ahmet Faruk KELEŞ

Makine Mühendisliği, Yüksek Lisans

Tez Danışmanı: İlker Temizer

Agustos 2024

Viraj katsayıları, araçların yanal ve boylamsal dinamiklerinde kritik bir rol oynar. Yol koşulları gibi çevresel faktörler dahil birçok faktöre bağlıdır. Bu çalışmada, derin sinir ağlarını kullanarak köşe alma katsayılarını gerçek zamanlı olarak tanımlayan bir yöntem geliştirilmiştir. Bu tanımlama için iki farklı veri setiyle üç farklı sinir ağı mimarisi karşılaştırılmış ve sonuçlar, zamanla değişen köşe alma katsayılarıyla eğitilmiş tam bağlantılı bir ağın en iyi performansa sahip olduğunu göstermiştir. Sabit köşe alma katsayıları kullanılmasına kıyasla, bu yöntem, 1/8 ölçekli dört tekerden çekişli dört tekerden yönlendirilebilir bir araç üzerinde yapılan deneylerde yanal kuvvet tahminini % 42.62-75.47 arasında iyileştirmiştir. Bu gerçek zamanlı tanımlamadan yararlanmak için, dört tekerden çekiş ve dört tekerden yönlendirme sisteminden faydalanan bir kontrol yöntemi geliştirilmiştir. Kontrol yöntemi, doğrusal olmayan bir model öngörülü kontrolcü kullanmaktadır. Kontrol sistemi, sürücünün kontrol girişlerini ideal önden çekişli önden yönlendirilebilir bir araç ikizinde kullanarak, sürücü tarafından belirlenen sabit köşe alma katsayılarıyla referans hızlar elde etmek için çalışır. Bu referanslardan yola çıkarak, doğrusal olmayan model öngörülü kontrolcü, dört tekerden çekişli dört tekerden yönlendirilebilir araç için optimal kontrol girişlerini hesaplayabilir. Simülasyon sonuçları, bu kontrol yönteminin referans takibini normal olarak sürülen önden çekişli önden yönlendirilebilir araca göre % 85 oranında iyileştirdiğini göstermektedir. Kontrolcü deneysel bir ortamda test edilmiş ve sonuçları % 12.12 oranında iyileştirdiği bulunmuştur. Gelişimdeki azalma, ölçümlerdeki gürültüye ve kontrol sistemindeki gecikmelere bağlanabilir.

Anahtar sözcükler: Araç Parametreleri Tehşisi, Yanal Denge Kontrolü .

Acknowledgement

I would like to thank my academic advisors, Dr. Melih Çakmakçı for his support, guidance, and invaluable expertise, and Prof. İlker Temizer, for whom, without his guidance, I would not have even considered a master's degree. I would like to thank my thesis committee Dr. Uyanık and Dr. Kabacaoğlu for their time and suggestions.

I want to thank my friends from the VEHCON group, Doğa Dağ and Kemal Köysüren for supporting me and my thesis. Without their efforts, this work would not have been possible

I express my gratitude for my girlfriend Ayşenur Katiöz and my friend Murathan Aydoğan, for being there for me when I needed it. I want to thank my brother Mustafa and sister-in-law Sena for keeping me fed and taken care of when I didn't have the time to cook.

I want to thank my parents Nizamettin and Aynur, for all their efforts to make me who I am, and for all their support. Feeling their presence behind made everything possible. I want to thank my brothers Enes and Mustafa for being role models to me throughout my life.

I want to thank MathWorks for all its documentation. I have learned a lot from reading the MATLAB documentation. Lastly, I want to thank Bilkent Shell gas station, for being open in the middle of the night and providing me with red bulls and tuna sandwiches, which gave me energy to conduct every experiment in this thesis.

Contents

1	Introduction	1
1.1	Cornering Coefficient Identification	1
1.2	Driving Assistance Systems	2
1.3	4 Wheel Drive 4 Wheel Steering Systems	3
1.4	Motivation and Contribution	3
1.5	Organization of the Thesis	4
2	Vehicle Modeling and Test Vehicle	5
2.1	Test Vehicle	5
2.2	Vehicle Model	7
2.3	Safety Protocols	10
2.4	Measurements and Data Collection	11
2.4.1	Data Filtering	11
2.4.2	Delay Analysis	13

3	Cornering Coefficient Identification With Deep Learning	14
3.1	Data Generation	14
3.1.1	LQR controller	15
3.1.2	Data Properties	17
3.2	Selection of Network Architecture and Data	19
3.3	Training	20
3.3.1	Training Results	20
3.4	Simulation Results	21
3.4.1	Simulation 1	21
3.4.2	Simulation 2	25
3.5	Experimental Results	30
3.5.1	Force Estimation Algorithm	30
3.5.2	Experiment 1	31
3.5.2.1	Results	34
3.5.3	Experiment 2	37
3.5.3.1	Results	40
3.5.4	Experiment 3	43
3.5.4.1	Results	46
3.6	Conclusion	49

4	Ideal Twin Driver Assistance	50
4.1	Controller Structure	50
4.2	Simulation Results	52
4.2.1	Simulation 1- Sine Inputs	52
4.3	Experiment Results	54
4.3.1	Experiment 1 - Sine Input and Comparison	54
4.3.2	Experiment 2 - Human Driver	57
5	Conclusion and Future Remarks	59

List of Figures

2.1	Modelled Vehicle	6
2.2	Bicycle model with lateral and longitudinal dynamics	8
2.3	Pajesca Curve For a Set of Parameters	9
2.4	Magnitude response curve of filter	12
2.5	An example IMU data and filtered version	12
2.6	Delay analysis	13
3.1	Distribution Of Dataset 1	18
3.2	Distribution Of Dataset 2	18
3.3	Simultion 1 Velocities	21
3.4	Simulation 1 input data	22
3.5	Simulation 1 LSTM-1 network cornering coefficient estimation result	22
3.6	Simulation 1 Conv-1 network cornering coefficient estimation result	23
3.7	Simulation 1 FC-1 network cornering coefficient estimation result	23

3.8	Simulation 1 LSTM-2 network cornering coefficient estimation result	24
3.9	Simulation 1 Conv-2 network cornering coefficient estimation result	24
3.10	Simulation 1 FC-2 network cornering coefficient estimation result	25
3.11	Simulation 12 Velocities	26
3.12	Simulation 2 input data	26
3.13	Simulation 2 LSTM-1 network cornering coefficient estimation result	27
3.14	Simulation 2 Conv-1 network cornering coefficient estimation result	27
3.15	Simulation 2 FC-1 network cornering coefficient estimation result	28
3.16	Simulation 2 LSTM-2 network cornering coefficient estimation result	28
3.17	Simulation 2 Conv-2 network cornering coefficient estimation result	29
3.18	Simulation 2 FC-2 network cornering coefficient estimation result	29
3.19	Experiment 1 input data	32
3.20	Experiment 1 velocity data	32
3.21	Results of estimation	33
3.22	Slip versus Lateral Force graph	33
3.23	Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α	34
3.24	Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α	35

3.25 Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α	35
3.26 Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α	36
3.27 Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α	36
3.28 Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α	37
3.29 Experiment 2 input data	38
3.30 Experiment 2 velocity data	38
3.31 Results of estimation	39
3.32 Slip versus Lateral Force graph	39
3.33 Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α	40
3.34 Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α	41
3.35 Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α	41
3.36 Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α	42
3.37 Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α	42

3.38 Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α	43
3.39 Experiment 3 input data	44
3.40 Experiment 3 velocity data	44
3.41 Results of estimation	45
3.42 Slip versus Lateral Force graph	45
3.43 Lateral force estimated by LSTM-1 Results vs Force Estimation algorithm vs Constant C_α	46
3.44 Lateral force estimated by Conv-1 Results vs Force Estimation algorithm vs Constant C_α	46
3.45 Lateral force estimated by FC-1 Results vs Force Estimation algorithm vs Constant C_α	47
3.46 Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α	47
3.47 Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α	48
3.48 Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α	48
4.1 Nonlinear Model Predictive Controller Structure	52
4.2 Control Inputs to ideal system and controller outputs	53
4.3 Velocities, Ideal vs Simulation vs Controller vs Direct Control	53
4.4 Control Inputs to ideal system and controller outputs	55

4.5	Velocities, Ideal vs Simulation vs Controller vs Direct Control . .	56
4.6	Control Inputs to ideal system and controller outputs	58
4.7	Velocities, Ideal vs Controller	58



List of Tables

2.1	Vehicle Parameters	7
3.1	Maximum values for data generation simulation input properties .	16
3.2	Neural Network Architectures	19
3.3	Training Settings	20
3.4	Training Results	20
3.5	Error in estimated cornering coefficients simulation 1	25
3.6	Error in estimated cornering coefficients simulation 2	30
3.7	Error in estimated forces experiment 1	37
3.8	Error in estimated forces experiment 2	43
3.9	Error in estimated forces experiment 3	49
4.1	Root mean square error in velocities	54
4.2	Error in velocities	56
4.3	Error in velocities	57

Chapter 1

Introduction

1.1 Cornering Coefficient Identification

Cornering Coefficients are important parameters in handling characteristics of vehicles [1]. They depend on both external conditions like road surface, temperature, pressure, and road wetness as well as the internal structure of tires ([2],[3],[4],[5]). As conditions change, cornering coefficients change. A few algorithms can identify the changing value of the cornering coefficient in real-time. Most of these algorithms utilize longitudinal slip between wheel velocity and ground speed to identify cornering coefficients. Still, these have the disadvantage of the requirement of additional hardware to measure longitudinal slips as well as limitations in measurement of longitudinal slip since the value is minimal under usual driving conditions [6]-[7]. [8] develops a novel algorithm that utilizes a GPS-based system that can accurately estimate cornering coefficients but has a convergence rate of 1 second. [9] utilizes the Levenberg-Marquart method to estimate cornering coefficients for 3-axle vehicles in real-time by iteratively solving the inverse problem from vehicle data. [10] and [11] use physics-informed neural networks to identify cornering coefficients instead of iterative solutions. [12] utilizes neural networks trained with experiment data to identify tire-road maximum friction coefficient μ_{max} .

In this thesis, a deep learning-based cornering coefficient estimation algorithm is developed to estimate cornering coefficients accurately and in real time with a very short convergence time. To achieve this estimation, a dataset needs to be generated, a network architecture needs to be selected, and a selected neural network needs to be trained.

1.2 Driving Assistance Systems

Society of Automobile Engineers (SAE) separates driving assistance and autonomous driving systems into six levels: no driving automation, driving assistance, partial driving automation, conditional driving automation, high driving automation, and full driving automation [13]. Almost all the current driving assistance and automation systems fall into the first three levels. Standard vehicle features like traction control systems fall into level 2, while more advanced features like lane centering and adaptive cruise control fall into level 3 [13]. Level 4 and higher-level autonomous systems are starting to emerge in commercial vehicles. These advanced systems reduce vehicle accident rates [14]. However, these systems can fail under non-ideal environmental conditions and cause more accidents [14]. [15] reports that advanced driver assistance systems (ADAS) can fail and affect safe driving under bad weather conditions.

Many studies consider the limits of traction in driver assistance systems. [16] develops a control framework for vehicle lateral stabilization and path tracking for autonomous systems using model predictive control. [17] utilizes a sliding mode controller for the traction control system in 4-wheel drive hybrid vehicles. [18] utilizes a PID control scheme that varies the slip ratio as a function of slip angle. Most of these works act on the information obtained only by slip angle. In this thesis, an alternative method to evaluate traction is presented, as well as a driver assistance system that engages continuously, not only when slips are high, like most of the systems present. For the implementation of this method, model predictive control is chosen because it is found to be one of the most effective control methods in driver assistance systems [19].

In this thesis, a novel driving assistance system that responds to changing traction conditions is developed and tested. The developed driving assistance system utilizes the identified traction conditions through cornering coefficients. The system works by creating an ideal twin of the vehicle with the cornering coefficients set by the driver and letting the driver drive the ideal twin while the controller follows the reference velocities generated by the ideal twin by utilizing a nonlinear model predictive controller.

1.3 4 Wheel Drive 4 Wheel Steering Systems

Four-wheel drive four-wheel steering systems have a lot of advantages over traditional front wheel drive front wheel steering systems. [20] reports that 4-wheel steering systems with counter steering ability have maneuvering advantages over conventional steering systems. [21] utilizes 4-wheel steering systems with an LQR controller and reports that 4-wheel steering systems can improve state accuracy between 13-45 % compared to traditional front steering systems with lower control effort. Because of these advantages, [22] concludes that these systems can be adapted for path tracking and autonomous control. [23] shows that 4-wheel steering systems are more stable under low traction conditions and can have better reference tracking.

In this thesis, a four-wheel drive four-wheel steering system is utilized with a model predictive control method to control the vehicle under changing traction conditions. The system is selected because it is better maneuverable under low traction conditions.

1.4 Motivation and Contribution

This thesis makes two main contributions to vehicle parameter identification and control research areas. The first contribution of this thesis is the development of a new cornering coefficient identification method that utilizes neural networks

trained with simulation data to identify cornering coefficients in an experimental setting accurately. The main advantage of this methodology is that, unlike most current methods that rely on iterative solutions to inverse problems, this method gives direct solutions and can be parallelized. With the current rise in parallel computing power with new GPUs, TPUs, and NPUs, this method can be run as fast as data can be collected, which makes it much more suitable to run in real-time. The usage of real time in this thesis means that it can be utilized online, but with some delays still in effect.

The second main contribution of this thesis is the development of an ideal twin-driving assistance system. This system utilizes the cornering coefficient identified by the neural network to assist human drivers under changing traction conditions. This is achieved by letting the driver control the ideal twin of the vehicle, which operates digitally under ideal traction conditions set by the driver. The controller takes the real traction conditions and calculates the optimal steering and throttle inputs, creating the same velocities as the ideal twin.

1.5 Organization of the Thesis

The outline of the rest of this thesis can be given as follows: In Chapter 2, the vehicle used for the experiments in this thesis is introduced and modeled. In Chapter 3, the training procedure of the neural networks used for estimating cornering coefficients is explained, and the trained neural networks are tested with experiment data. In Chapter 4, the control method for the driving assistance system is explained, and results are given. In Chapter 5, concluding remarks are given and discussed.

Chapter 2

Vehicle Modeling and Test Vehicle

This chapter details the vehicle used in this thesis and derives a dynamical model for the test vehicle. Section 1 introduces the test vehicle. Section 2 derives the dynamical model. Section 3 discusses the safety protocols implemented. Section 4 discusses measurement methods and data collection.

2.1 Test Vehicle

The vehicle used in this thesis is a custom-built 4-wheel drive, 4-wheel steering 1/8 scale vehicle shown in Figure 2.1. It has two electric motors controlling the front and the rear wheels independently, four steering servos that control the steering angle, four brake servos that control the brakes, and four suspension servos that control the height and stiffness of the suspension of each wheel. The vehicle runs on Jetson Nano and can communicate through WiFi using the Robot Operating System (ROS) with an external computer. A node on the vehicle publishes sensor data and drives the servo and DC motors. The driver assistance system works on the external computer, subscribing to the sensor data and publishing required

motor commands. The necessity of the external computer comes from the lack of cpu compute power on the jetson nano board on vehicle. With a better board, the system can be fully run on the vehicle itself. All the systems work at a 10 Hz, including sensor data sampling and publication, assistance system on an external computer, and motor actuation. Vehicle parameters are given in Table 2.1.

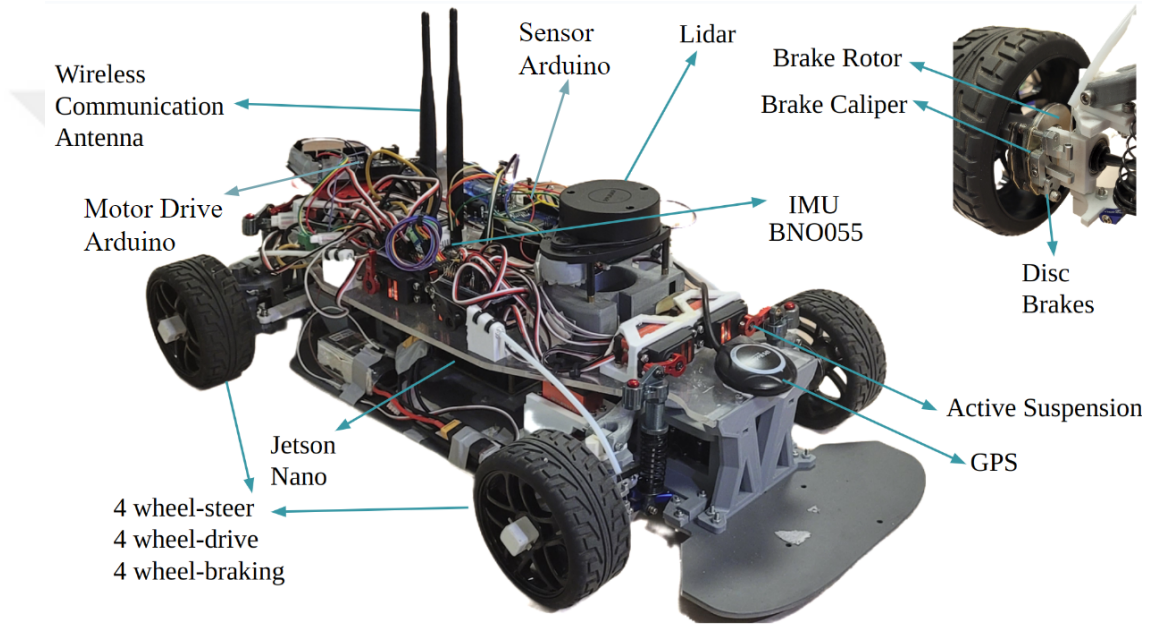


Figure 2.1: Modelled Vehicle

Table 2.1: Vehicle Parameters

Parameter	Value
Mass (m)	6.25 kg
Wheelbase (L)	0.4 m
Inertia (I)	0.8 $kg.m^2$
Wheel Radius (R_w)	5 cm
Length between the front axle and center of gravity (a)	0.19 m
Length between the rear axle and center of gravity (b)	0.21 m
Length between the center of gravity and GPS sensor (a_2)	0.25 m
Length between the center of gravity and IMU sensor (a_3)	0.07 m
Torque constant of the electric motor (K_t)	0.0024
Resistance of electric motor (Ω)	0.0081
Operating voltage of electric motor (V_o)	11.1
Power Limit To Electronic Speed Controller (P_f)	0.35

2.2 Vehicle Model

The dynamic model of the four-wheel drive steering vehicle can be represented in various ways. From a full car model with 14 degrees of freedom to a single block with 2 degrees of freedom [24],[25]. As the degree of freedom increases, the complexity of the model also increases. This research selects a bicycle model that captures both longitudinal and lateral dynamics because it accurately represents the vehicle's behavior while maintaining some simplicity. The model is shown in Figure 2.2. In Figure 2.2, u represents longitudinal velocity, v represents lateral

velocity, r represents yaw rate, δ represents steering angles, η represents the control input of electric motors and α represents the lateral slip angles. The accelerations can be calculated by equations (7), (8) and (9).

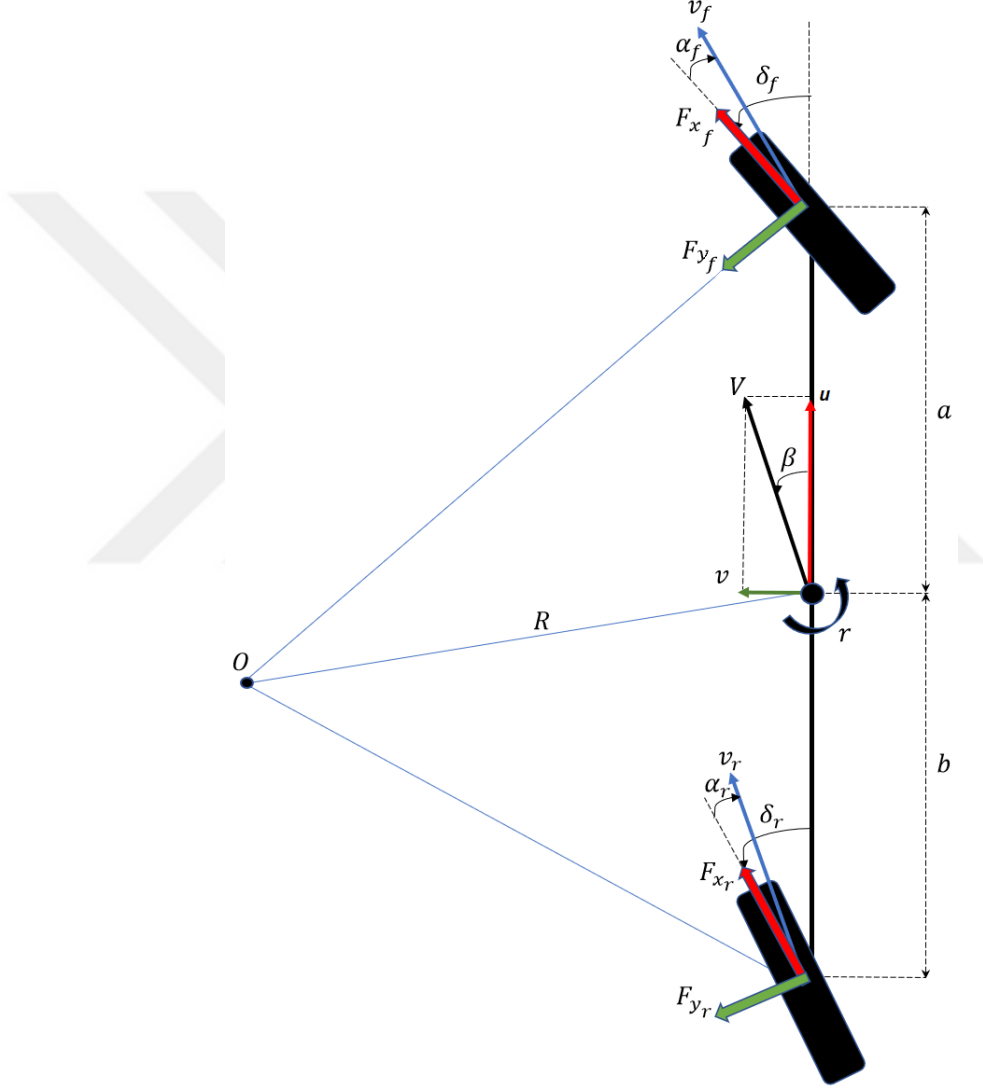


Figure 2.2: Bicycle model with lateral and longitudinal dynamics

$$\dot{u} = \frac{(F_{xf}\cos(\delta_f) - F_{yf}\sin(\delta_f) + F_{xr}\cos(\delta_r) - F_{yr}\sin(\delta_r))}{m} + vr \quad (2.1)$$

$$\dot{v} = \frac{(F_{xf}\sin(\delta_f) + F_{yf}\cos(\delta_f) + F_{xr}\sin(\delta_r) + F_{yr}\cos(\delta_r))}{m} - ur \quad (2.2)$$

$$\dot{r} = \frac{(F_{xf}\sin(\delta_f)a + F_{yf}\cos(\delta_f)a - F_{xr}\sin(\delta_r)b - F_{yr}\cos(\delta_r)b)}{I} \quad (2.3)$$

Here, F_x represents the traction force, and F_y represents the lateral force on the tires. Lateral forces can be modeled as a function of slips α . Pajesca's magic formula [26] represents the nonlinear relation between slip and force as:

$$F_y = D * \sin(C * \operatorname{atan}(B * \alpha - E * (B * \alpha - \operatorname{atan}(B * \alpha)))) \quad (2.4)$$

The formula comes from experiment results and is empirical. The parameters D , C , B , and E are usually obtained experimentally and depend on experiment conditions. The parameters change as conditions change, and they need to be recalculated. The formula can be visualized as shown in Figure 2.3.

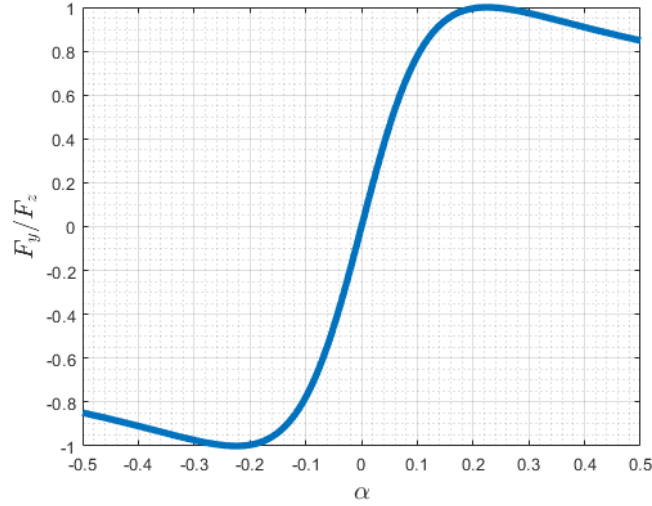


Figure 2.3: Pajesca Curve For a Set of Parameters

As it can be seen from the figure 2.3, for small values of slip, the relationship can be approximated as a linear relationship around 0 with cornering coefficient C :

$$F_{yf} = C_{\alpha f} \alpha_f \quad (2.5)$$

$$F_{yr} = C_{\alpha r} \alpha_r \quad (2.6)$$

The linearized coefficients are usually calculated from experiment results and depend on experiment conditions, the same as Pajesca's parameters. The lateral

slip can be formulated as a function of velocities for front and rear wheels as:

$$\alpha_f = \delta_f - \text{atan}\left(\frac{v + ar}{u}\right) \quad (2.7)$$

$$\alpha_r = \delta_r - \text{atan}\left(\frac{v - br}{u}\right) \quad (2.8)$$

Tractive forces come from the torque generated in the electric motors. The relation between tractive force and the control parameter η can be approximated as:

$$F_{xf} = \eta F_{max} \quad (2.9)$$

Where η has a range of $[-1,1]$. The parameter F_{max} comes from the characteristics of the electric motor as:

$$F_{max} = \frac{P_f V_0 K_t}{\Omega R_w} = 23.02 \quad (2.10)$$

The power limit P_f is applied to limit the vehicle's maximum acceleration because high accelerations can damage the vehicle's mechanical components; mostly, full power separates the tires from the wheels at high rpm. The calculated value is also experimentally verified with straight-line acceleration tests.

2.3 Safety Protocols

In order to ensure that the vehicle is always under control, there are several safety procedures. The first layer of safety is that every motor in the system has a master switch that controls the power to the motor. The switches are only turned on when every system is ready to start driving, and they are switched off whenever the test ends. This allows safe maintenance and examination of the vehicle. The second layer of safety comes from the Arduino that controls the motors. The Arduino always checks whether the system is properly connected to a Simulink model and has a decent wifi connection. In the event of a break in connection, the Arduino applies the brakes, cuts off the motors, and zeros the steering so that the vehicle stops safely.

2.4 Measurements and Data Collection

The test vehicle has many different sensors and measurement units. An IMU-Gyro measurement unit, BNO055, can measure accelerations, rotational velocities, and Euler angles. A GPS module can also measure position, velocity, and heading angle. From these sensors, lateral and longitudinal velocities and accelerations can be calculated as:

$$u = V_{gps} * \cos(\phi_{heading} - \phi_{vehicle}) \quad (2.11)$$

$$v = V_{gps} * \sin(\phi_{heading} - \phi_{vehicle}) - a_2 * r \quad (2.12)$$

$$\dot{u} = IMU_x + v * r - a_3 * r^2 \quad (2.13)$$

$$\dot{v} = IMU_y - u * r - a_3 * \dot{r} \quad (2.14)$$

$$r = IMU_\omega \quad (2.15)$$

$$\dot{r} = \frac{\delta IMU_\omega}{\delta t} \quad (2.16)$$

Where V_{gps} is GPS velocity, $\phi_{heading}$ is the heading angle from GPS, $\phi_{vehicle}$ is the Euler angle from the gyroscope, IMU_x, IMU_y, IMU_ω are acceleration and rotational velocity measurements from the IMU module. a_2 is the distance between the GPS sensor and the center of gravity, and a_3 is the distance between IMU and the center of gravity.

2.4.1 Data Filtering

A lowpass filter is utilized to reduce the noise in sensor measurement data. The utilized filter is selected as a 2nd order FIR filter with a Chebyshev window [27] with 40 sidelobe attenuation, 1.5 Hz cutoff frequency, and 10 hz sampling rate. The magnitude response curve is given in Figure 2.4. An example of unfiltered and filtered IMU data is shown in Figure 2.5. As can be observed from Figure 2.5, this filter creates a 0.1-second worth of delay.

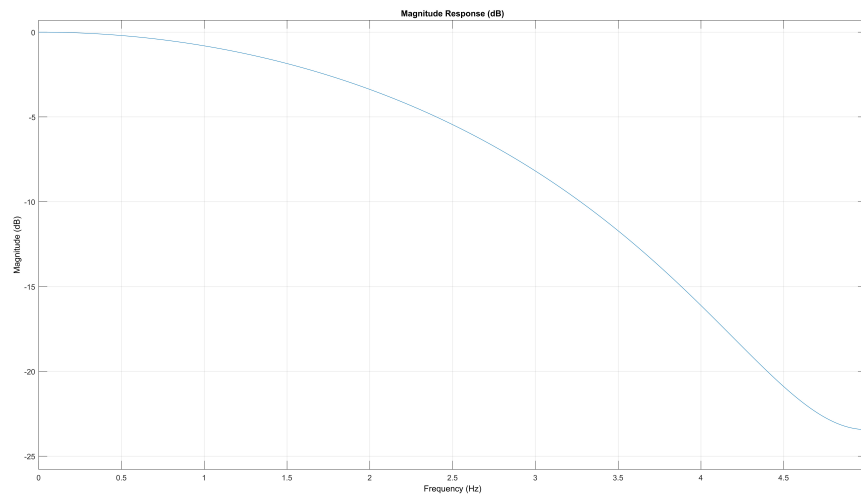


Figure 2.4: Magnitude response curve of filter

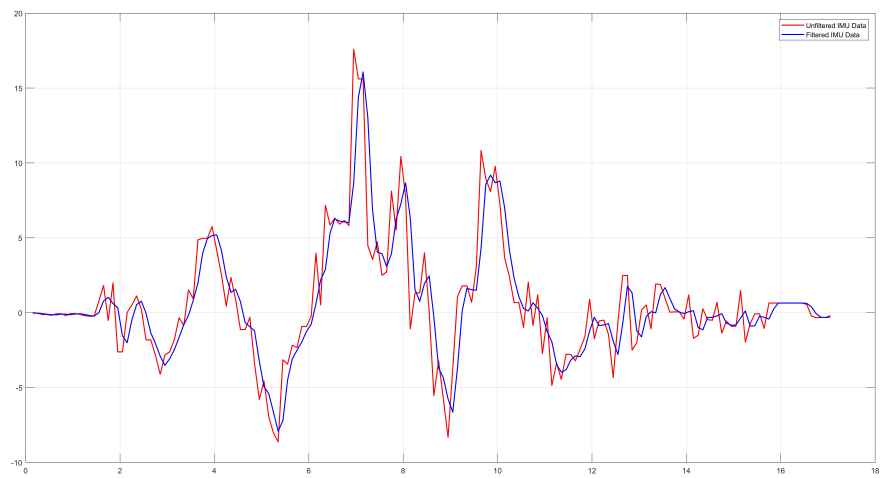


Figure 2.5: An example IMU data and filtered version

2.4.2 Delay Analysis

The Jetson Nano board collects the data on the vehicle. However, because the board does not have enough computing power to run models, all data is transferred to an external computer with a discrete GPU and modern CPU through a Robot Operating System network. This transfer creates a communication delay in the collected data as well as a delay in the actuation of control signals. The communication delay is identified as 0.2 seconds in one way. Another cause of delay in the system is the filtering of the sensor signals. As discussed in Section 2.4.1, the filter utilized creates a 0.1-second delay. Another source of delay is the GPS sensor, which has a delay of 0.1 seconds. With all these delays identified, a time chart is given in Figure 2.6. From the GPS signals to actuation, there is a total delay of 0.6 seconds.

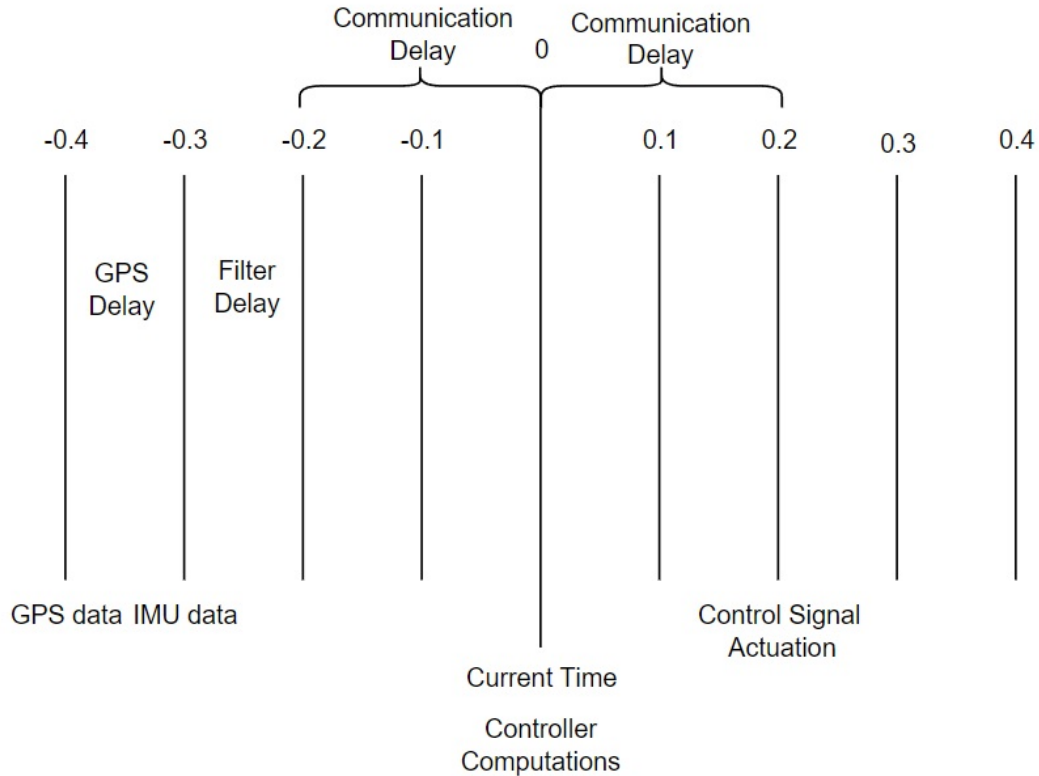


Figure 2.6: Delay analysis

Chapter 3

Cornering Coefficient Identification With Deep Learning

This chapter examines three different neural network architectures trained with two different datasets for their performance in predicting cornering coefficients on three different experiment setups. In Section 1, the data generation methodology for training is discussed. In Section 2, the training setup and results are shown. In Section 3, experiment setups are given, an algorithm for generating benchmark results is developed, and the performance of the neural networks is evaluated. In Section 4, performance results are discussed.

3.1 Data Generation

To generate a suitable dataset for training a neural network to predict changing cornering coefficients, two different simulation studies are conducted, and data is collected. In the first simulation study, the vehicle model given in Section [2.2](#)

is simulated with varying inputs and parameters. To ensure that the data generated is feasible, the simulation study is created with a reference following the LQR controller because LQR controllers offer robustness and stability [28]. At each simulation, a reference for u, r, α_r^* is generated as a sine wave with random amplitudes, bias, and frequencies, while lateral velocity reference is calculated from the other references. The reason for generating lateral velocity reference from slip reference is to ensure that the reference generated is of a stable system. Relation between other references and lateral velocity reference is given in Equation 3.1. From these reference values, the LQR controller calculates η, δ inputs that maintain stability in the system with proper reference tracking. For the first simulation dataset, cornering coefficients $C_{\alpha_f}, C_{\alpha_r}$ are generated as a sine wave with random amplitudes biases and frequencies. The amplitude, bias, and frequency range are given in Table 3.1.

$$v = \tan(\alpha_r^*) * u + b * r \quad (3.1)$$

In the second simulation dataset, cornering coefficients are constant in time but change from one simulation to another in the range of 1-400. The reference velocities are generated in the same way as the first simulation study. Simulations are conducted at 100 Hz solve rate with ODE5 [29] solver and sampled down to 10 Hz for training.

3.1.1 LQR controller

To find a suitable LQR controller, the nonlinear model that is given by equations 2.1-2.10 needs to be linearized. For linearization, the states are u, v, r , and the inputs are $\delta_f, \delta_r, \eta_f, \eta_r$. The linearized A matrix is given in Equation 3.2, and B matrix is given in Equation 3.3. For the first simulation dataset, the linearization point is taken as $C_\alpha = C_{\alpha}^{bias}, u = u_{bias}^*, v = 0, r = 0$. For the second simulation dataset, the linearization point is taken as $C_\alpha = C_{\alpha}, u = u_{bias}^*, v = 0, r = 0$. For both studies, Q and R matrices are tuned and given in Equations 3.4, 3.5. Simulation studies are conducted with these controllers.

$$\begin{pmatrix} -\frac{\text{Caf sin(DeltaF)}(v+a\,r) + \text{Car sin(DeltaR)}(v-b\,r)}{u^2\sigma_3} - \frac{\text{Caf sin(DeltaF)}(v-b\,r)}{u^2\sigma_2} & \frac{\text{Caf sin(DeltaF)} + \text{Car sin(DeltaR)}}{u\sigma_3} + \frac{\text{Caf sin(DeltaF)} + \text{Car sin(DeltaR)}}{u\sigma_2} & \frac{\text{Caf } a \text{ sin(DeltaF)} - \text{Car } b \text{ sin(DeltaR)}}{u\sigma_3} - \frac{\text{Caf } a \text{ sin(DeltaF)} - \text{Car } b \text{ sin(DeltaR)}}{u\sigma_2} \\ \frac{\text{Caf cos(DeltaF)}(v+a\,r) + \text{Car cos(DeltaR)}(v-b\,r)}{u^2\sigma_3} - \frac{\text{Caf cos(DeltaF)}(v-b\,r)}{u^2\sigma_2} - \gamma & \frac{\text{Caf cos(DeltaF)} + \text{Car cos(DeltaR)}}{u\sigma_3} + \frac{\text{Caf cos(DeltaF)} + \text{Car cos(DeltaR)}}{u\sigma_2} & -u - \frac{\sigma_1}{m} \\ \frac{\text{Caf } a \text{ cos(DeltaF)}(v+a\,r) - \text{Car } b \text{ cos(DeltaR)}(v-b\,r)}{u^2\sigma_3} - \frac{\text{Caf } a \text{ cos(DeltaF)}(v-b\,r)}{u^2\sigma_2} & -\frac{\sigma_1}{1} & -\frac{\text{Caf } a^2 \text{ cos(DeltaF)} + \text{Car } b^2 \text{ cos(DeltaR)}}{u\sigma_3} + \frac{\text{Caf } a^2 \text{ cos(DeltaF)} + \text{Car } b^2 \text{ cos(DeltaR)}}{u\sigma_2} \end{pmatrix}$$

where

$$\sigma_1 = \frac{\text{Caf } a \text{ cos(DeltaF)}}{u\sigma_3} - \frac{\text{Car } b \text{ cos(DeltaR)}}{u\sigma_2}$$

$$\sigma_2 = \frac{(v-b\,r)^2}{u^2} + 1$$

$$\sigma_3 = \frac{(v+a\,r)^2}{u^2} + 1$$

$$\begin{pmatrix} \frac{49 \cos(\text{DeltaF})}{2m} & \frac{49 \cos(\text{DeltaR})}{2m} & -\frac{\text{Caf sin(DeltaF)} + \frac{49 \text{EtaF sin(DeltaF)}}{2} + \text{Caf cos(DeltaF)} \sigma_2}{m} & -\frac{\text{Car sin(DeltaR)} + \frac{49 \text{EtaR sin(DeltaR)}}{2} + \text{Car cos(DeltaR)} \sigma_1}{m} \\ \frac{49 \sin(\text{DeltaF})}{2m} & \frac{49 \sin(\text{DeltaR})}{2m} & \frac{\text{Caf cos(DeltaF)} + \frac{49 \text{EtaF cos(DeltaF)}}{2} - \text{Caf sin(DeltaF)} \sigma_2}{m} & \frac{\text{Car cos(DeltaR)} + \frac{49 \text{EtaR cos(DeltaR)}}{2} - \text{Car sin(DeltaR)} \sigma_1}{m} \\ \frac{49 a \sin(\text{DeltaF})}{21} & -\frac{49 b \sin(\text{DeltaR})}{21} & \frac{\text{Caf } a \text{ cos(DeltaF)} + \frac{49 \text{EtaF } a \text{ cos(DeltaF)}}{2} - \text{Caf } a \text{ sin(DeltaF)} \sigma_2}{1} & -\frac{\text{Car } b \text{ cos(DeltaR)} + \frac{49 \text{EtaR } b \text{ cos(DeltaR)}}{2} - \text{Car } b \text{ sin(DeltaR)} \sigma_1}{1} \end{pmatrix}$$

where

$$\sigma_1 = \text{DeltaR} - \text{atan}\left(\frac{v-b\,r}{u}\right)$$

$$\sigma_2 = \text{DeltaF} - \text{atan}\left(\frac{v+a\,r}{u}\right)$$

Parameter	Amplitude	Bias	Frequency
u^*	5	5	4
α_r^*	0.5	0	4
r^*	4	0	4
C_α	180	90	8
C_α^2	250	0	0

Table 3.1: Maximum values for data generation simulation input properties

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} \quad (3.4)$$

$$R = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} \quad (3.5)$$

The generated data is then further examined to ensure that the data is of a stable system. With the simulation study ready, 8192 different scenarios are simulated for 15 seconds for each dataset, and the data is batched as 1-second packages. From the simulation data $u, v, r, \dot{u}, \dot{v}, \dot{r}, \eta, \delta$ are collected and used for training and validation.

3.1.2 Data Properties

In order to ensure that the data generated can represent the conditions that the real vehicle could face, the distribution of the datasets is inspected. For both datasets, the distribution of η, δ, u, v, r are similar. From the figures 3.1, 3.2, it can be seen that the operating range for the neural network is limited. For operations outside of this range can result in bad estimations. For both datasets, the operation range on u is between 0 and 6 centered around 3 m/s. v and r , are centered around 0 with ranges 1 and 4. η and δ are defined for every possible value for the vehicle. Cornering coefficient also has the same range for both datasets. However, the distributions are different for each dataset. Dataset 1 has a distribution closer to normal distributions and Dataset 2 has uniform distribution. This difference can effect the performance of estimation compared with each other.

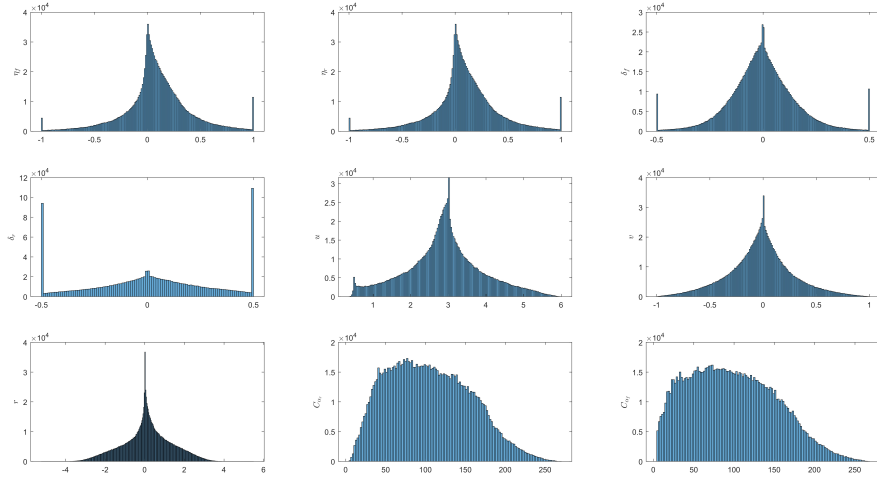


Figure 3.1: Distribution Of Dataset 1

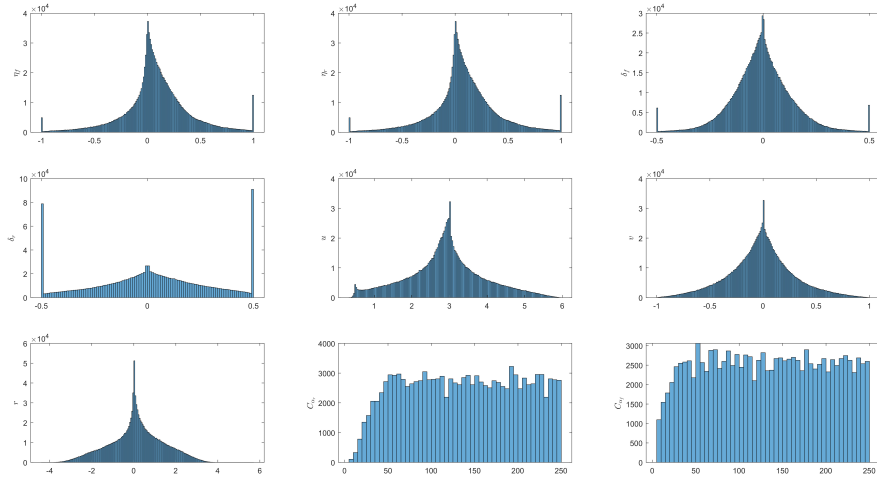


Figure 3.2: Distribution Of Dataset 2

3.2 Selection of Network Architecture and Data

Three different network architectures are examined to select a suitable network architecture: LSTM, Convolutional, and Fully Connected. The exact architectures are given in Table 3.2. Layers that are colored in red are only present for dataset 2. The network sizes are selected in order to equalize the inference times of the networks. The average inference times are 0.0082,0.0084,0.0072.

LSTM	Conv	FC
Input	Input	Input
BiLSTM(32)	Conv(32,3)	FC(32)
FC(32)	reLu	reLu
BiLSTM(32)	Conv(32,3)	FC(32)
FC(32)	reLu	reLu
FC(2)	Conv(32,3)	FC(32)
Scaling	reLu	reLu
	Conv(32,3)	FC(32)
	reLu	reLu
	Conv(32,3)	FC(32)
	reLu	reLu
	FC(2)	FC(32)
	Scaling	reLu
	GlobalPool	FC(2)
		Scaling
		GlobalPool

Table 3.2: Neural Network Architectures

3.3 Training

The generated neural networks are trained with the data generated in Section 3.1. All networks are trained with the ADAM [30] algorithm with a decaying learning rate. The initial learning rate is 0.001 for all training, and decay is 10^{-5} . L2 regularization is utilized with the weight of 10^{-4} . The batch size is set as 256 for all experiments as well. Training settings are given in Table 3.3.

Architecture	FC	Conv	LSTM
Loss Function	MAE	MAE	MAE
L2 Reg	0.0001	0.0001	0.0001
Batch Size	256	256	256
Learning Rate	0.0001	0.0001	0.0001
Training Length (#Epochs)	20	20	20

Table 3.3: Training Settings

3.3.1 Training Results

Results of the training are given in Table 3.3.1. These results show the performance of the training. Error values between dataset 1 and dataset 2 should not be compared since they represent different datasets, comparison should be made between network architectures. For dataset 1, the convolutional network has the lowest error with the lowest variance. For dataset 2, the LSTM network has the lowest error with the lowest variance.

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2
Training MAE	189.51	101.33	115.17	6.87	11.32	12.61
Validation MAE	219.14	127.22	129.12	6.73	13.18	5.82
Training Percent Error	10.10	5.98	7.54	2.95	4.91	12.83
Validation Percent Error	11.89	6.98	7.84	2.82	5.17	5.69
Validation Percent Error Standart Deviation	12.97	8.02	10.48	4.80	8.02	8.24

Table 3.4: Training Results

3.4 Simulation Results

In all the simulations, input data is generated the same way as data generation for training.

3.4.1 Simulation 1

In this simulation, cornering coefficients are varied as sine waves. This simulation allows the evaluation of the performance of the neural network in constantly changing conditions. The reference and vehicle velocities are given in Figure 3.3. The control inputs are given in Figure 3.4. Mean percent errors are given in table 3.5. Estimation results are given in Figures 3.5-3.10.

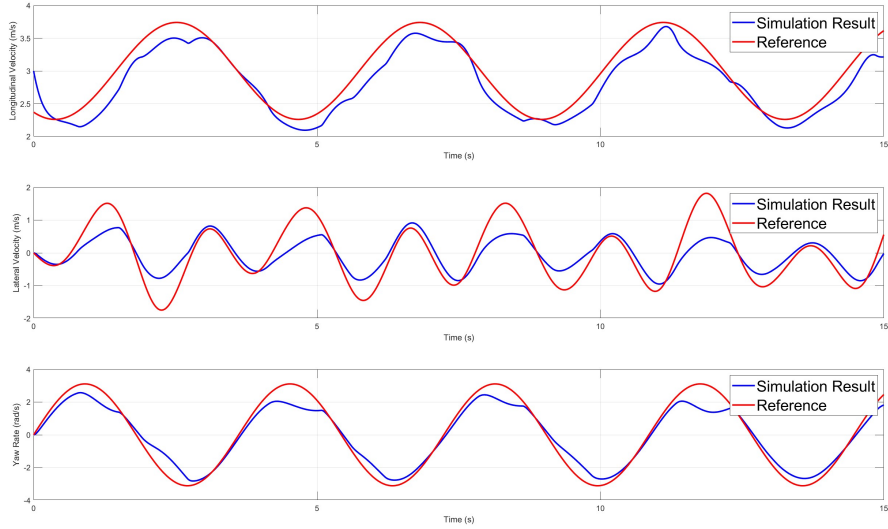


Figure 3.3: Simulation 1 Velocities

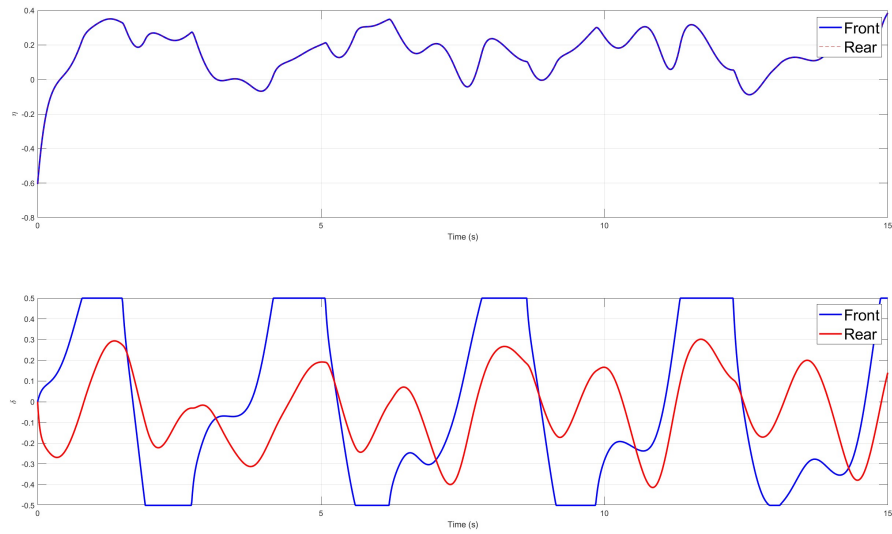


Figure 3.4: Simulation 1 input data

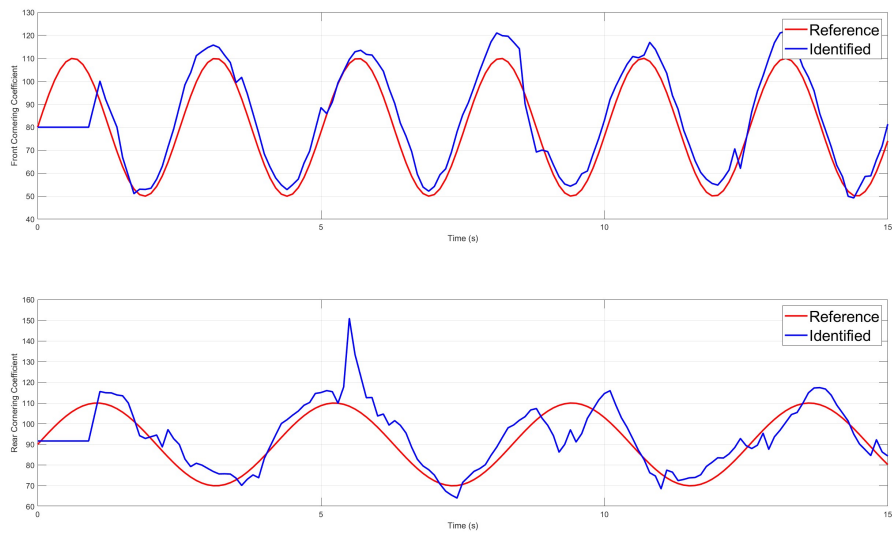


Figure 3.5: Simulation 1 LSTM-1 network cornering coefficient estimation result

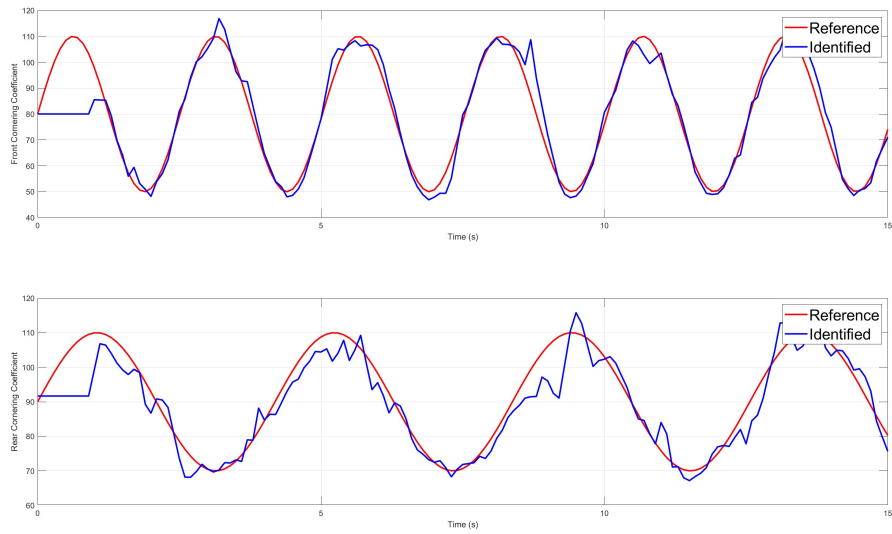


Figure 3.6: Simulation 1 Conv-1 network cornering coefficient estimation result

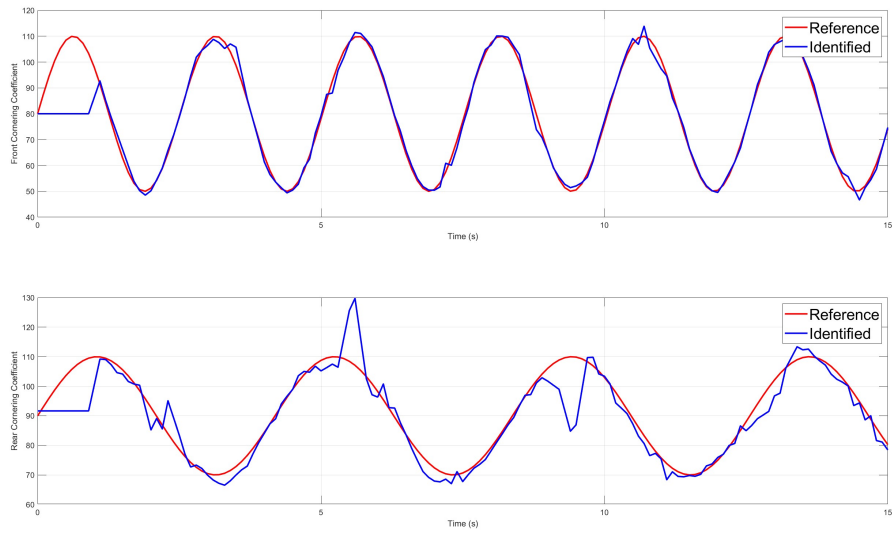


Figure 3.7: Simulation 1 FC-1 network cornering coefficient estimation result

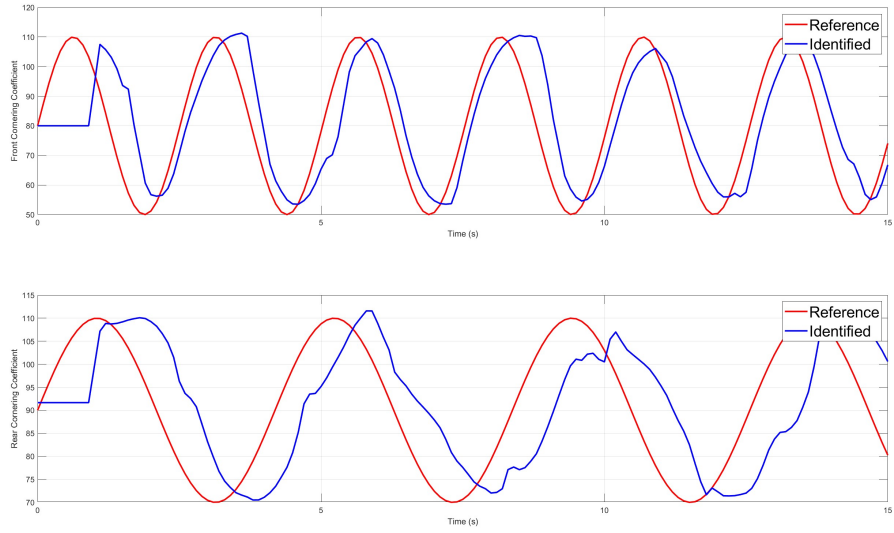


Figure 3.8: Simulation 1 LSTM-2 network cornering coefficient estimation result

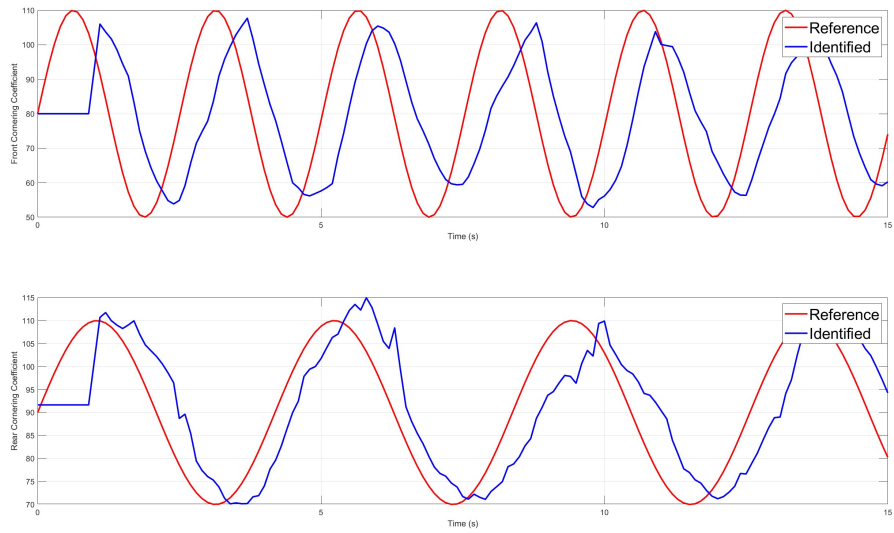


Figure 3.9: Simulation 1 Conv-2 network cornering coefficient estimation result

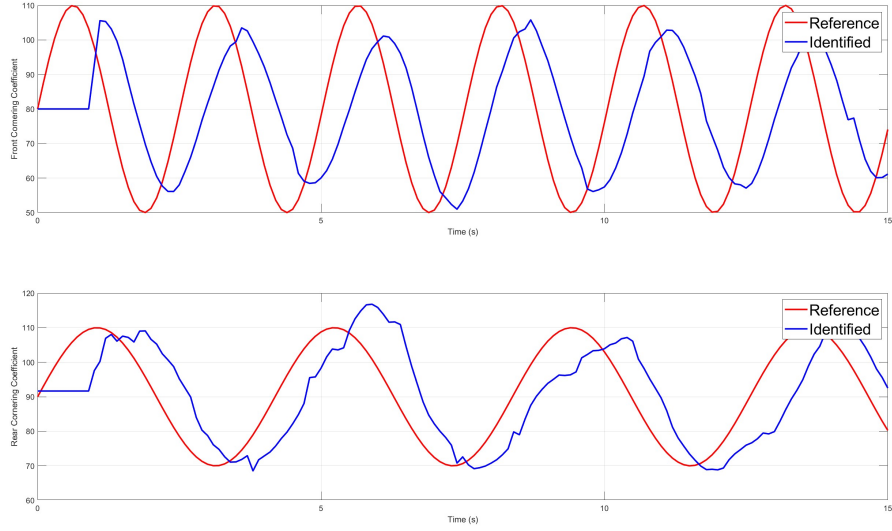


Figure 3.10: Simulation 1 FC-2 network cornering coefficient estimation result

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2
Error	1.83	1.04	1.27	3.07	3.66	3.69

Table 3.5: Error in estimated cornering coefficients simulation 1

In this simulation, Conv-1 network has the lowest mean percent error. Networks trained with dataset 1 have lower errors than networks trained with dataset 2. From Figures 3.8-3.10, it can be seen that these networks estimate with some delay. This delay behavior can be due to global pooling layers making the network slower to react to changes by averaging in time.

3.4.2 Simulation 2

In this simulation, cornering coefficients are varied as step functions. This simulation allows evaluation of the performance of the neural networks on sudden changes in traction. The reference and vehicle velocities are given in Figure 3.11.

The control inputs are given in Figure 3.12. Mean percent errors are given in table 3.6. Estimation results are given in Figures 3.13-3.18.

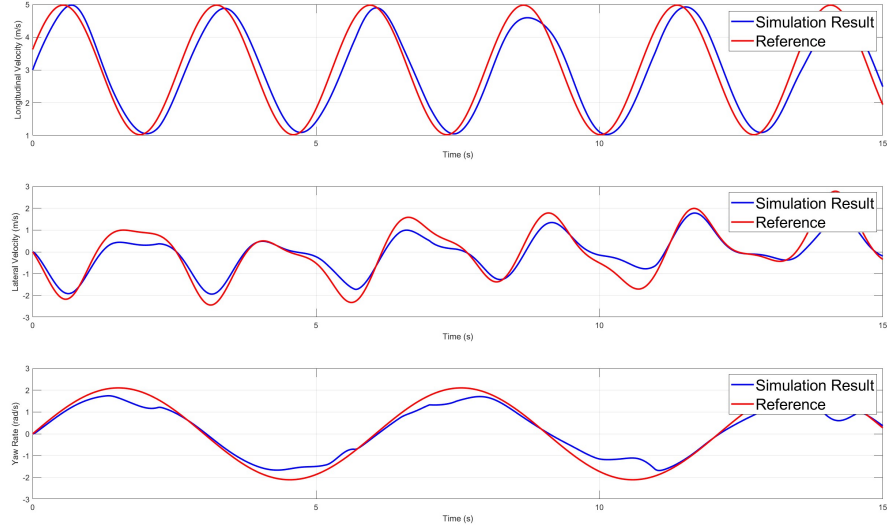


Figure 3.11: Simulation 12Velocities

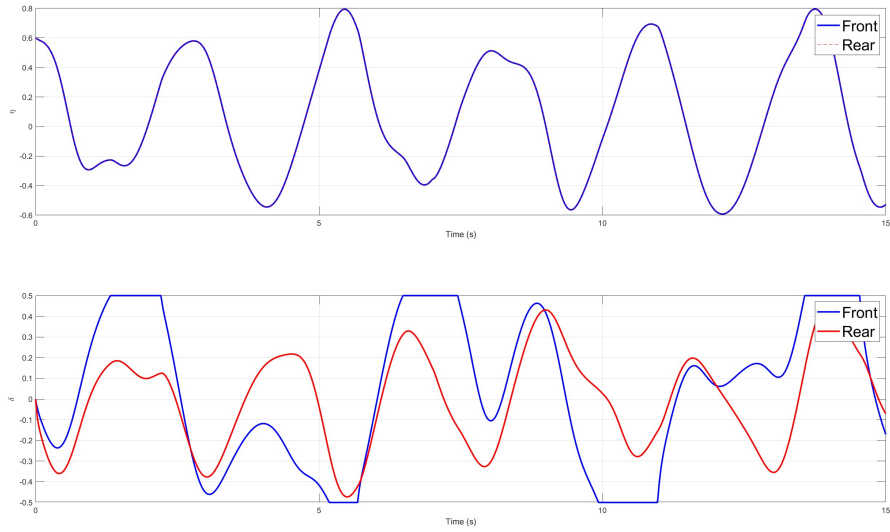


Figure 3.12: Simulation 2 input data

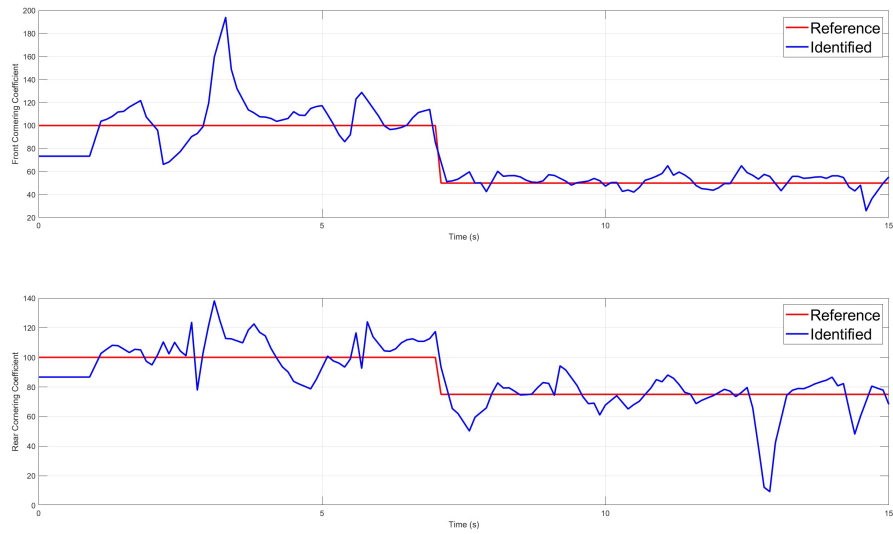


Figure 3.13: Simulation 2 LSTM-1 network cornering coefficient estimation result

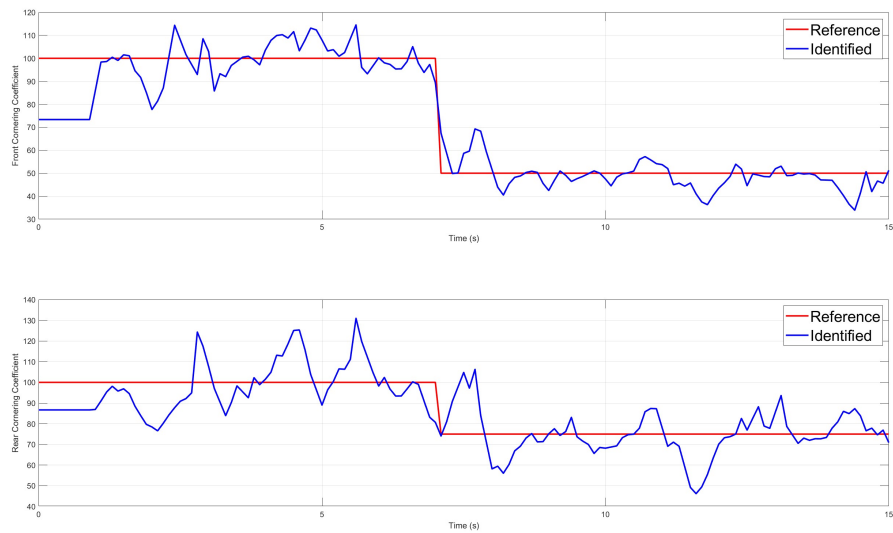


Figure 3.14: Simulation 2 Conv-1 network cornering coefficient estimation result

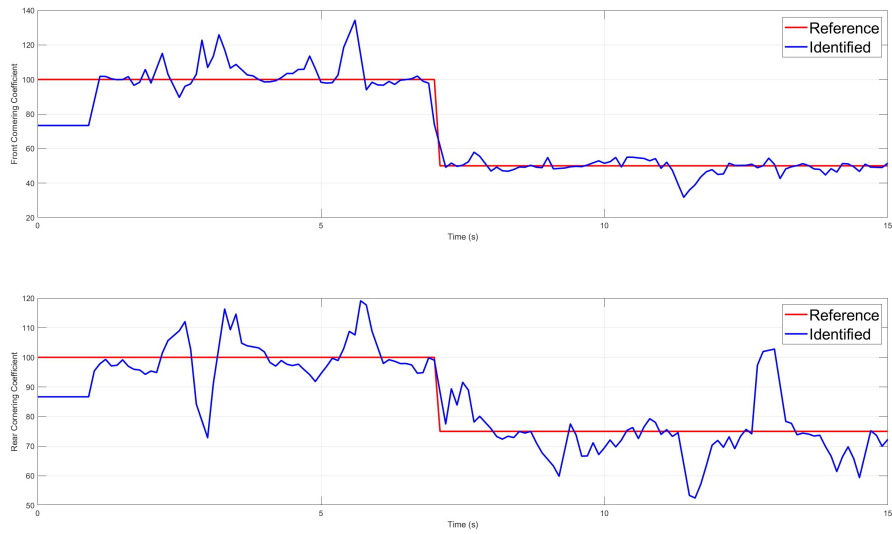


Figure 3.15: Simulation 2 FC-1 network cornering coefficient estimation result

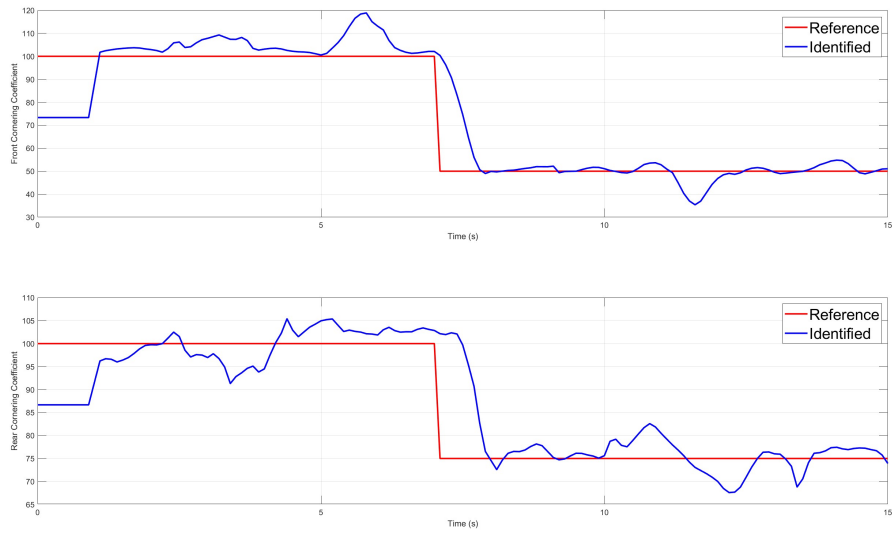


Figure 3.16: Simulation 2 LSTM-2 network cornering coefficient estimation result

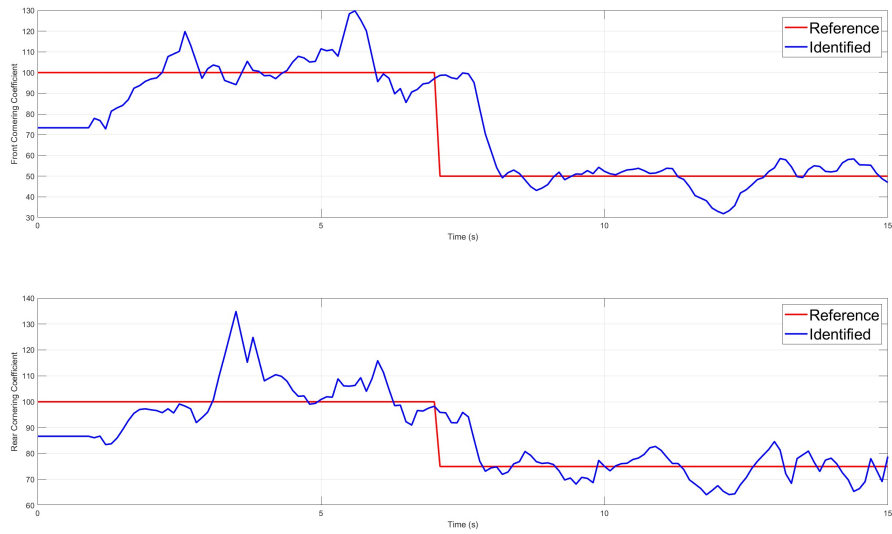


Figure 3.17: Simulation 2 Conv-2 network cornering coefficient estimation result

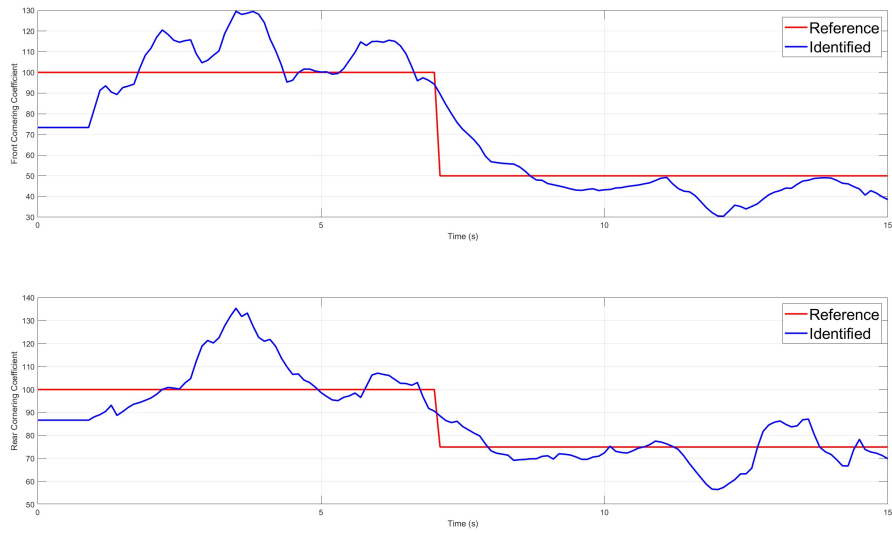


Figure 3.18: Simulation 2 FC-2 network cornering coefficient estimation result

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2
Error	3.60	1.33	1.54	1.79	1.96	2.14

Table 3.6: Error in estimated cornering coefficients simulation 2

In this simulation, Conv-1 network has the lowest mean percent error. From Figures (3.13-3.18), it can be seen that networks trained with dataset-2 has smoother estimations but the networks estimate with some delay, while networks trained with dataset 1 behave with more noise but the estimation settles quicker than networks trained with dataset 2. This observation is similar to the observation made in Section 3.4.1

3.5 Experimental Results

Three experiments were conducted to evaluate the neural network’s performance, and the data was collected. Collected data is processed to ensure that the experiment setup and vehicle model match, and a baseline truth value for lateral forces, on which the performance of the neural networks can be evaluated, is calculated. The calculation of a baseline truth value is explained in Section 3.5.1.

3.5.1 Force Estimation Algorithm

The vehicle model given in Section 2.2 is a bicycle model, which is a very simplified approach to vehicle dynamics. This approach may have some unmodelled effects like longitudinal friction and load transfer. To observe the impact of these unmodelled behaviors and create a baseline truth value to evaluate the performance of the neural networks, forces acting on the vehicle during the experiments are estimated. This estimation is done by utilizing the Equations (2.1-2.2-2.3) and experimental velocities and accelerations. A nonlinear least squares estimation algorithm matches the measured accelerations with estimated ones to find

the forces that would result in measured accelerations. This is done using Matlab's `lsqnonlin()` function with the Trust-Region-Reflective solver [31]. The code for the optimization is given in the appendix.

With the forces identified, the performance of the neural network can be evaluated by comparing the estimated force with the neural network predicted cornering stiffness multiplied by slip values as:

$$Error = \int_{\tau} ||F_y^{est} - C_{\alpha}^{NN} * \alpha|| * dt \quad (3.6)$$

3.5.2 Experiment 1

In experiment 1, the vehicle is controlled by a human driver. In this experiment, all four wheels are used for steering. The rear wheels are utilized in a counter-steering arrangement and are given the opposite steering to the front wheels. This allows much agile motion but with increased wheel slippage. The resulting motion is given in Figure 3.20. The force estimation algorithm is executed from the experiment data, and forces are estimated. To evaluate the performance of the estimation algorithm, the accelerations can be calculated by using (2.1,2.2,2.3) and compared with measured accelerations, and it is given in Figure 3.21. From the estimated forces, a slip vs. lateral force graph is generated. The generated graph is given in Figure 3.22

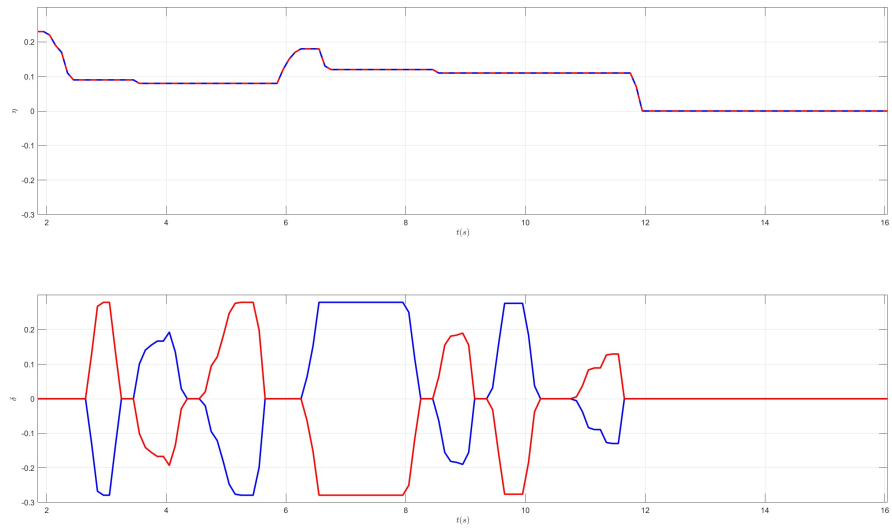


Figure 3.19: Experiment 1 input data

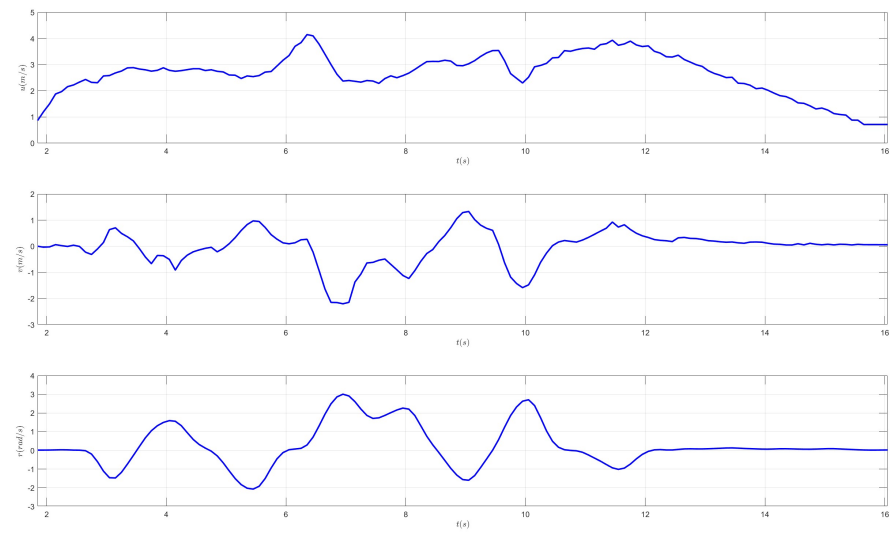


Figure 3.20: Experiment 1 velocity data

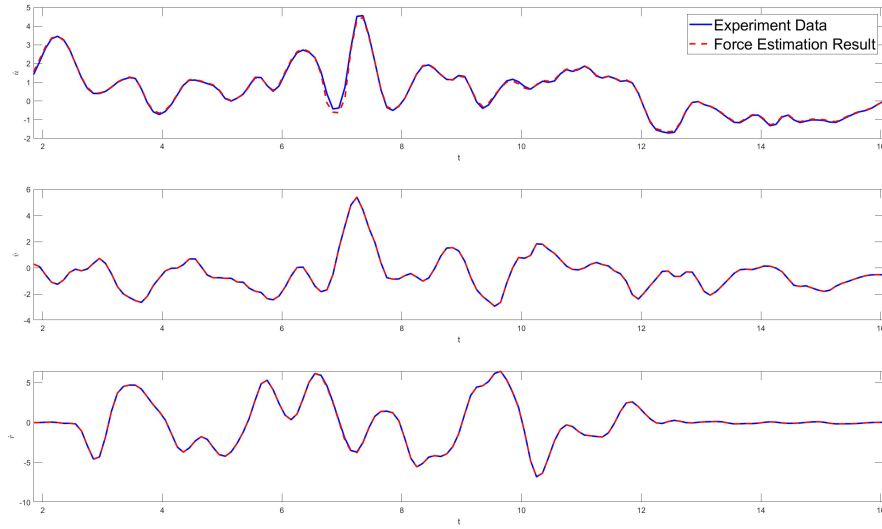


Figure 3.21: Results of estimation

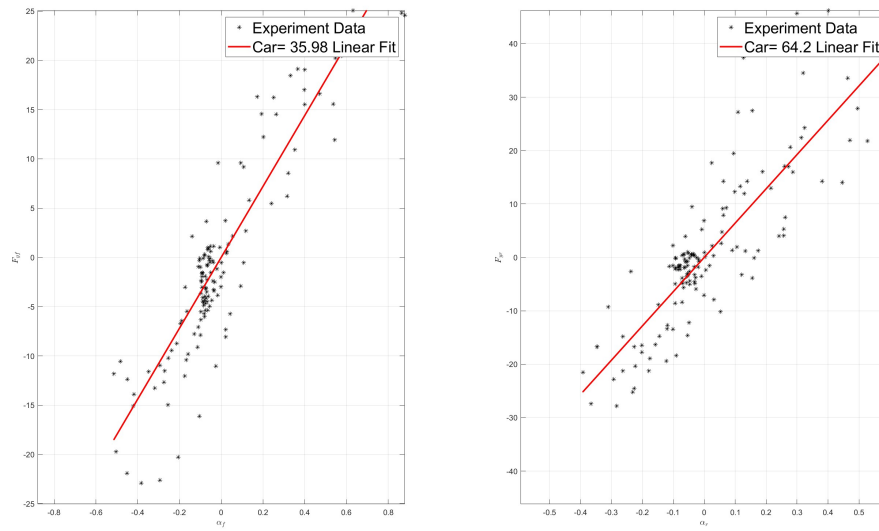


Figure 3.22: Slip versus Lateral Force graph

From Figure 3.19 and Figure 3.22, it can be observed that experiment data is noisy and does not always behave linearly. In Figure 3.22, the red line represents

the best-fit line computed with a linear least squares algorithm, and it will be used as a benchmark to compare the neural network performance.

3.5.2.1 Results

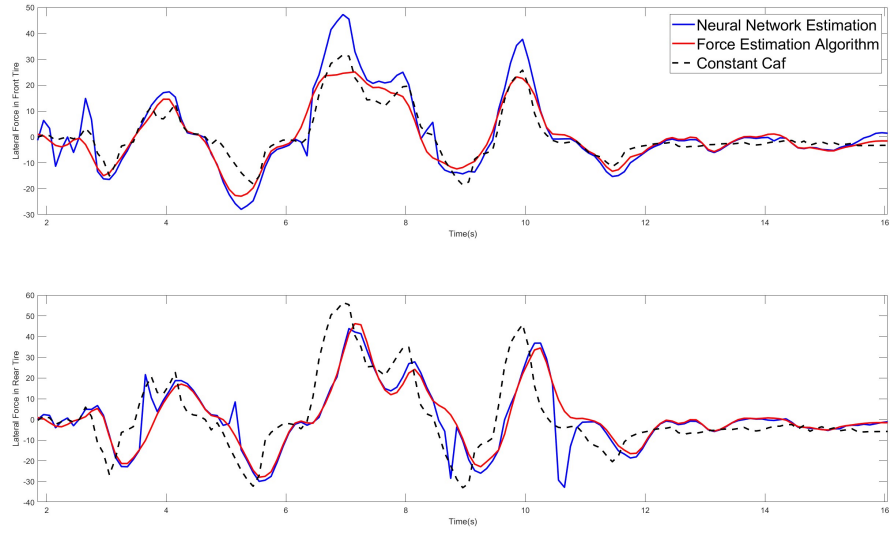


Figure 3.23: Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_{α}

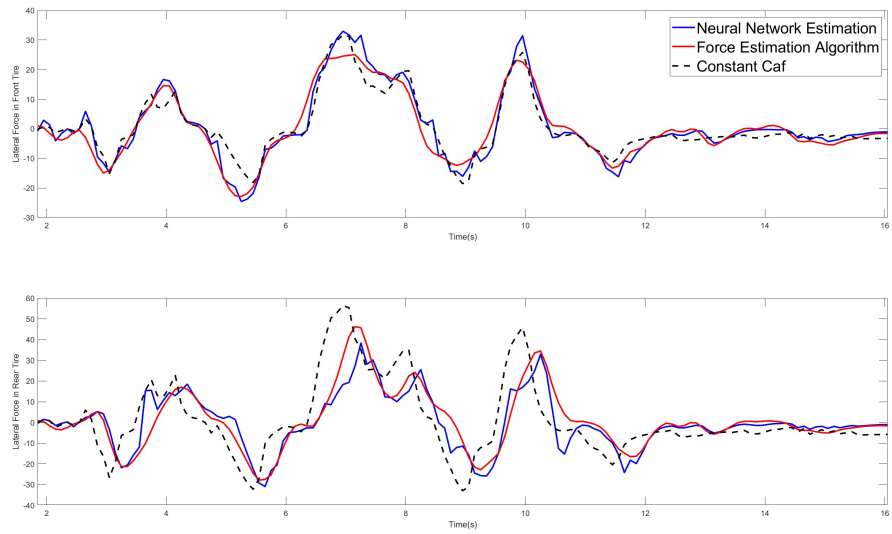


Figure 3.24: Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_{α}

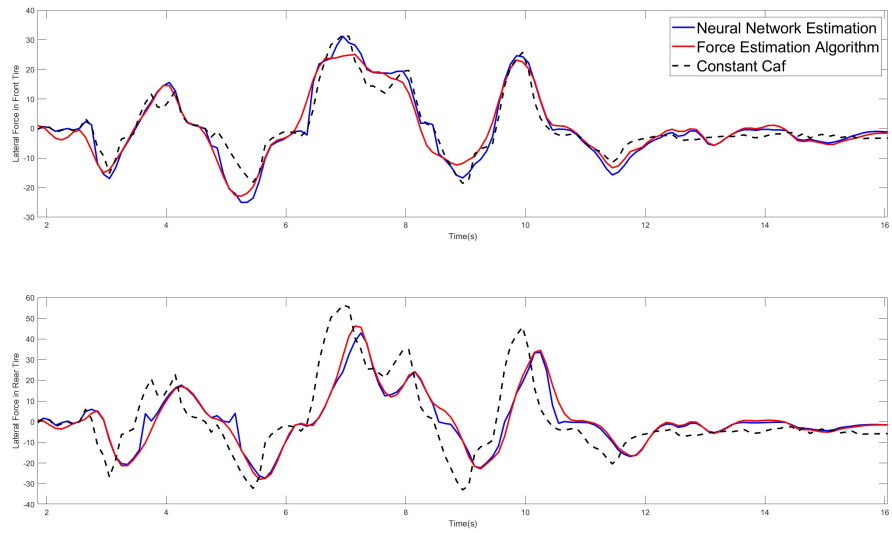


Figure 3.25: Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_{α}

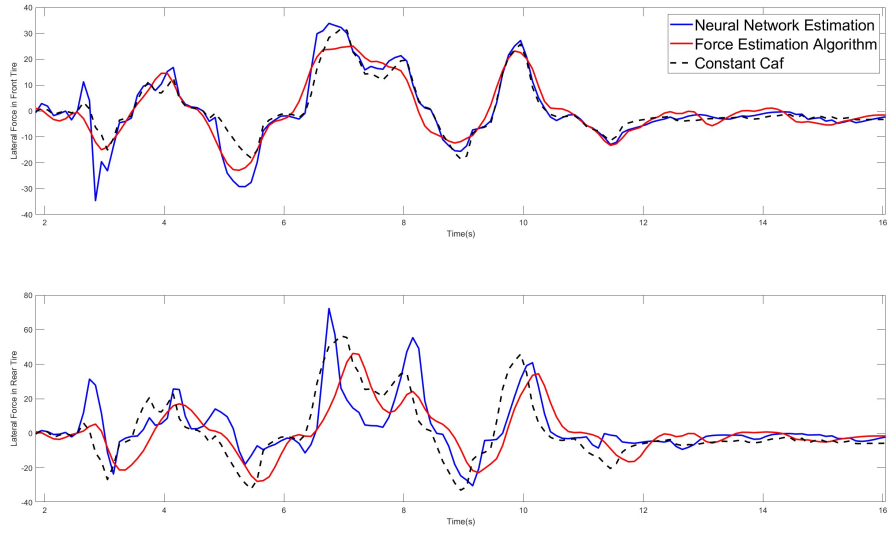


Figure 3.26: Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α

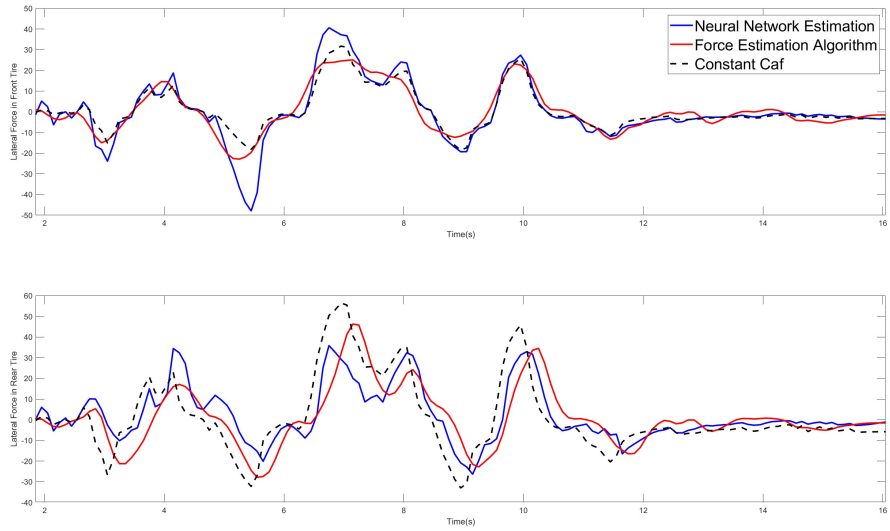


Figure 3.27: Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α

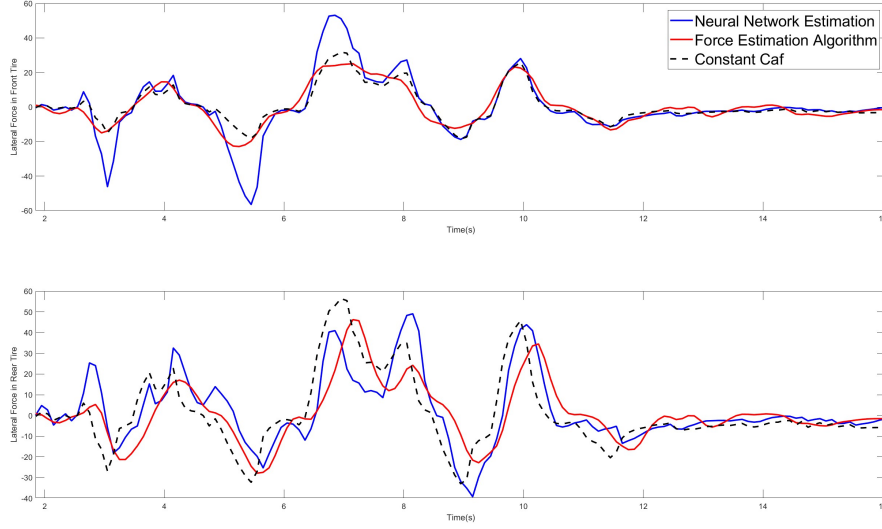


Figure 3.28: Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_{α}

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2	Constant
Error	5.85	5.85	3.08	11.82	10.53	12.76	13.09

Table 3.7: Error in estimated forces experiment 1

3.5.3 Experiment 2

In experiment 2, the vehicle is controlled by a human driver. In this experiment, only the front wheels are used for steering. The resulting motion is given in Figure 3.30. The force estimation algorithm is executed from the experiment data, and forces are estimated. To evaluate the performance of the estimation algorithm, the accelerations can be calculated by using Equations (2.1,2.2,2.3) and compared with measured accelerations, and it is given in Figure 3.31. From the estimated forces, a slip vs. lateral force graph is generated. The generated graph is shown in Figure 3.32

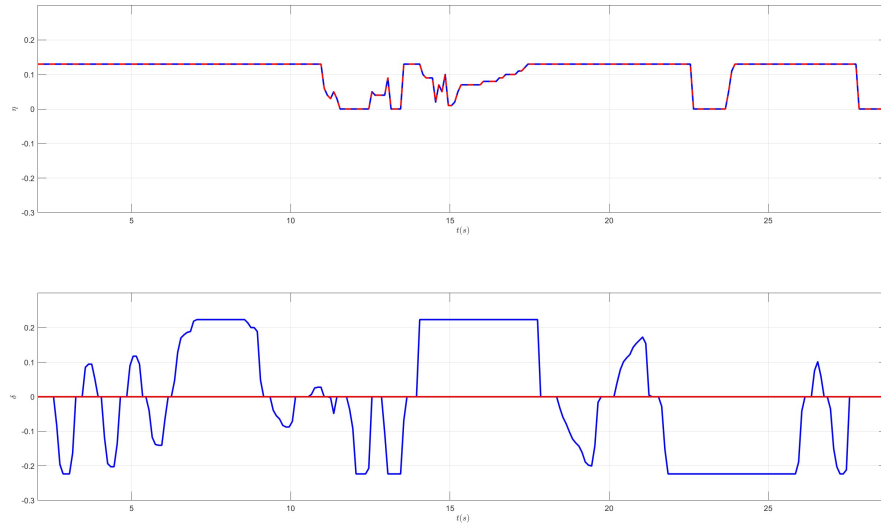


Figure 3.29: Experiment 2 input data

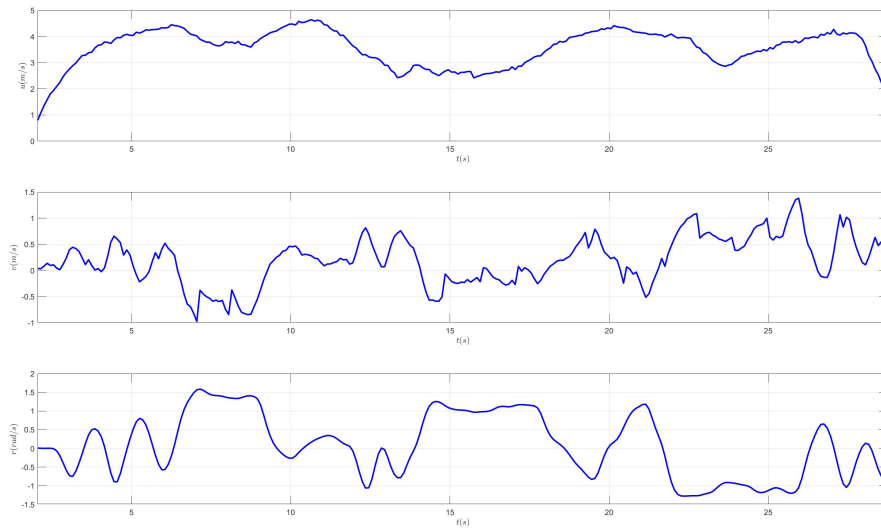


Figure 3.30: Experiment 2 velocity data

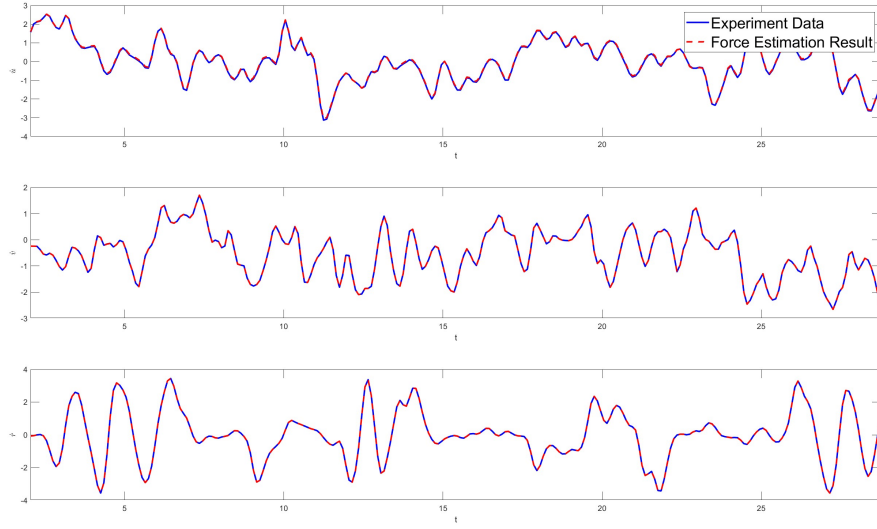


Figure 3.31: Results of estimation

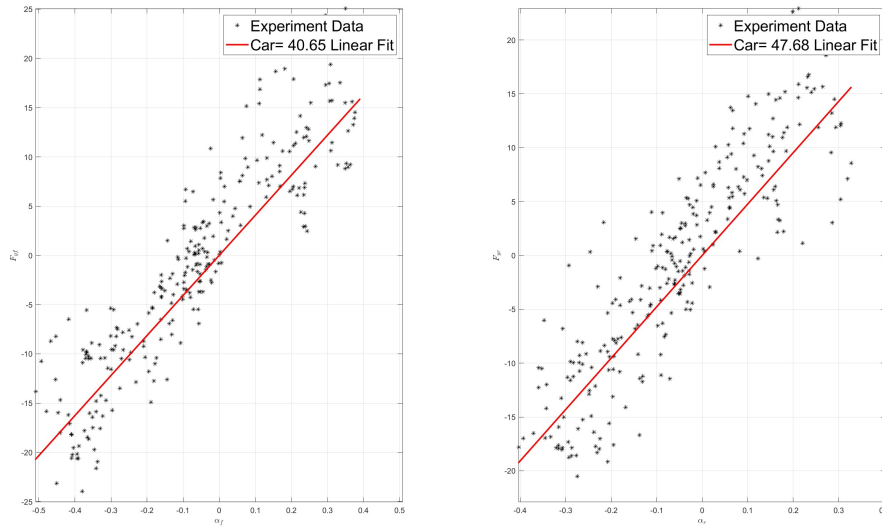


Figure 3.32: Slip versus Lateral Force graph

From Figure 3.30 and Figure 3.32, it can be observed that experiment data is noisy and does not always behave linearly. In Figure 3.22, the red line represents

the best-fit line computed with a linear least squares algorithm, and it will be used as a benchmark to compare the neural network performance.

3.5.3.1 Results

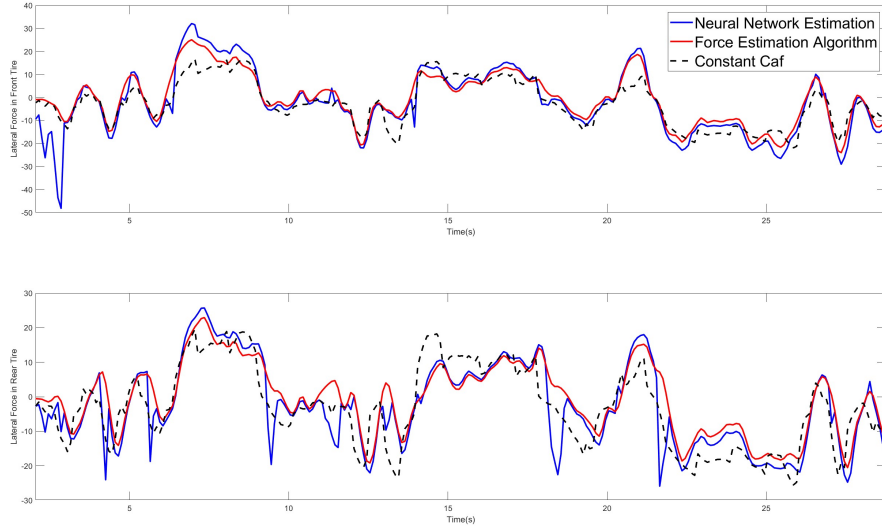


Figure 3.33: Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_{α}

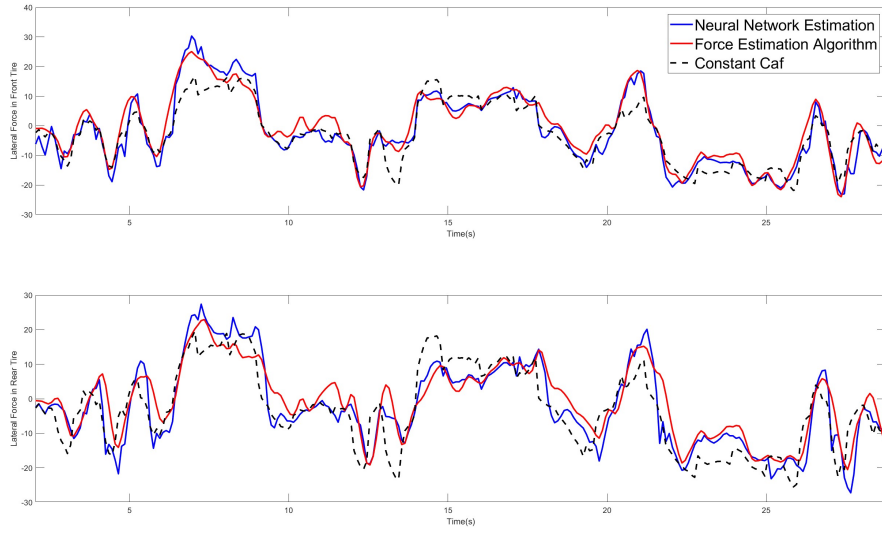


Figure 3.34: Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α

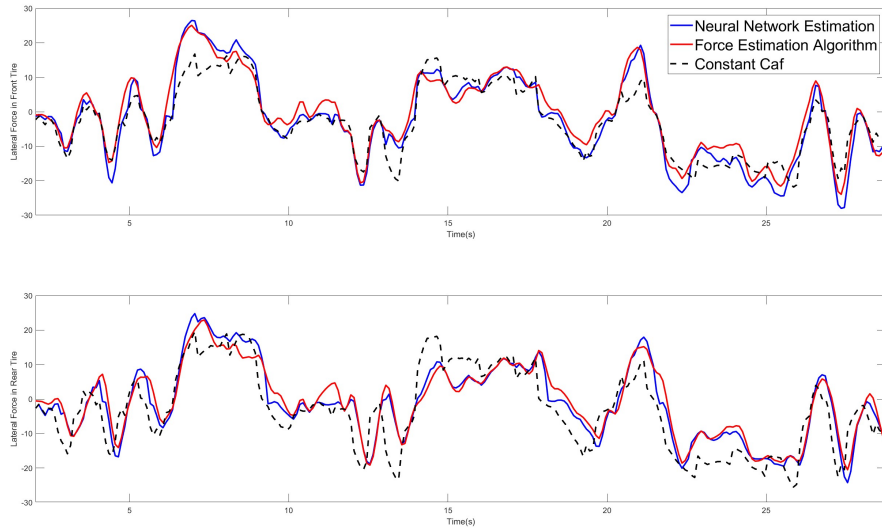


Figure 3.35: Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_α

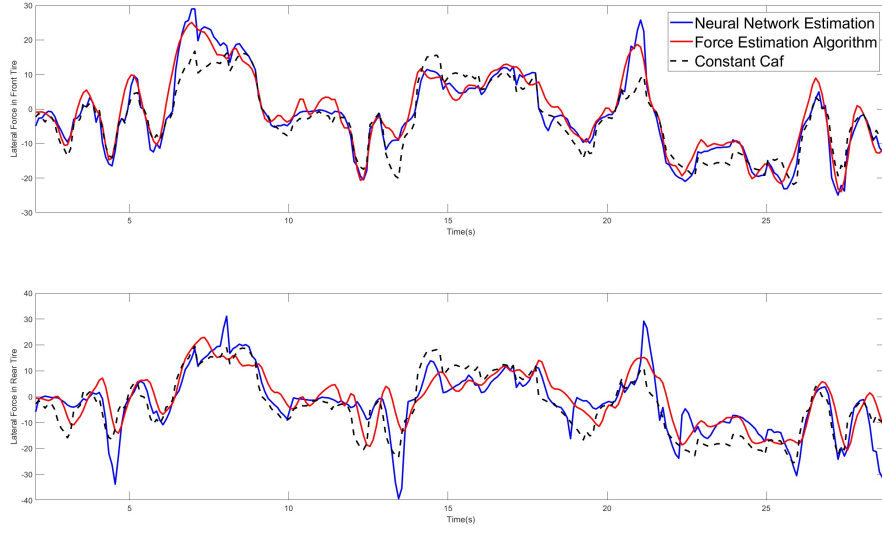


Figure 3.36: Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_α

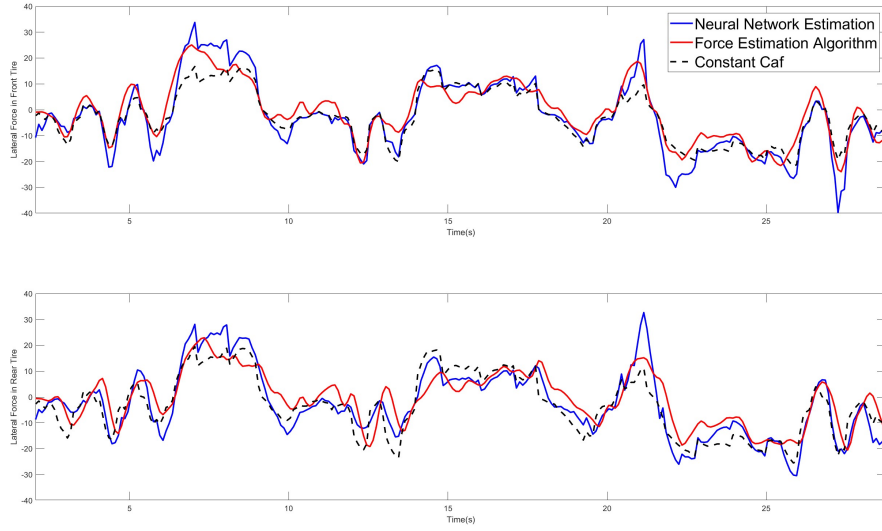


Figure 3.37: Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_α

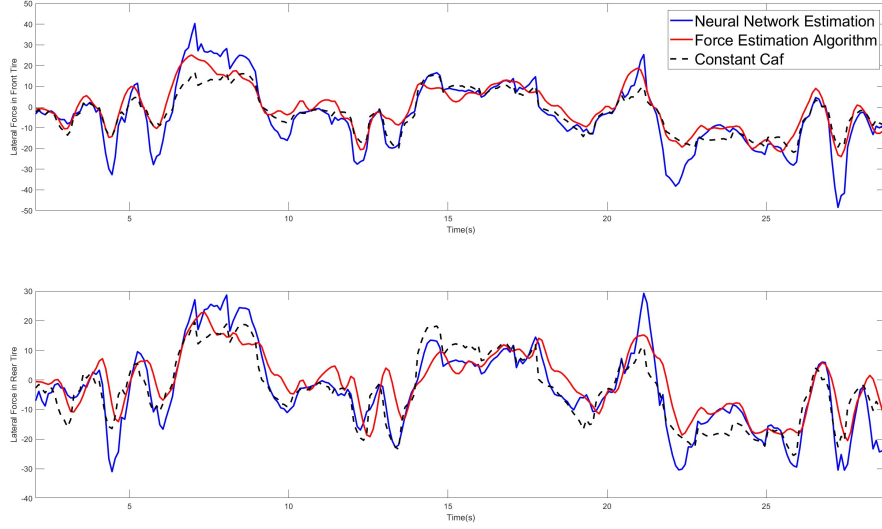


Figure 3.38: Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_{α}

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2	Constant
Error	6.18	6.10	4.06	7.58	9.32	10.88	11.86

Table 3.8: Error in estimated forces experiment 2

3.5.4 Experiment 3

In experiment 3, the vehicle is controlled by the control algorithm described in 4. This dataset evaluates the neural network's performance in a 4-wheel drive, 4-wheel steering setup. The resulting motion is given in Figure 3.40. Lateral and longitudinal forces are estimated by utilizing the algorithm described in Section 3.5.1. From these estimates, a slip vs lateral force graph is generated and is given in Figure 3.42.

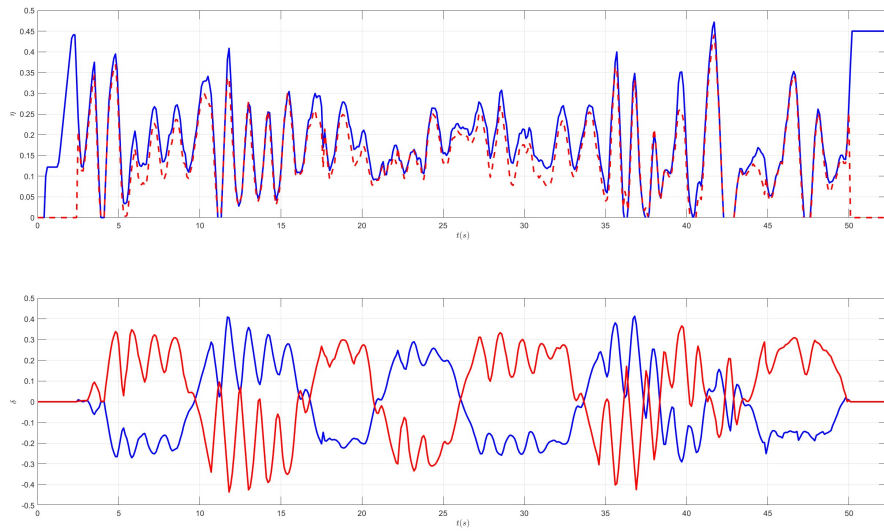


Figure 3.39: Experiment 3 input data

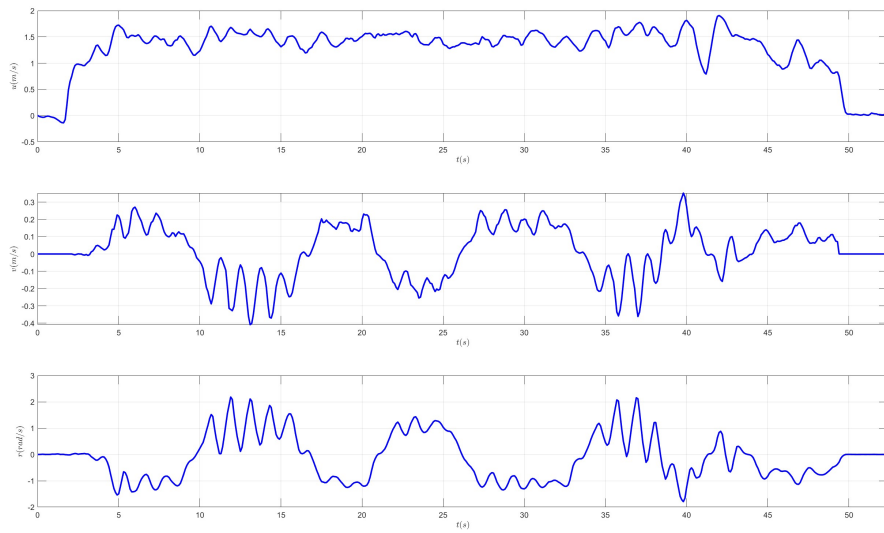


Figure 3.40: Experiment 3 velocity data

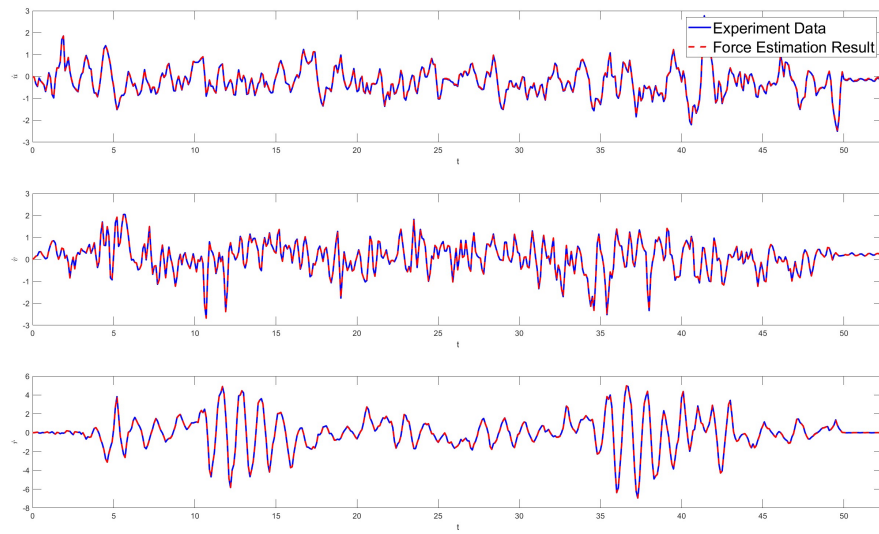


Figure 3.41: Results of estimation

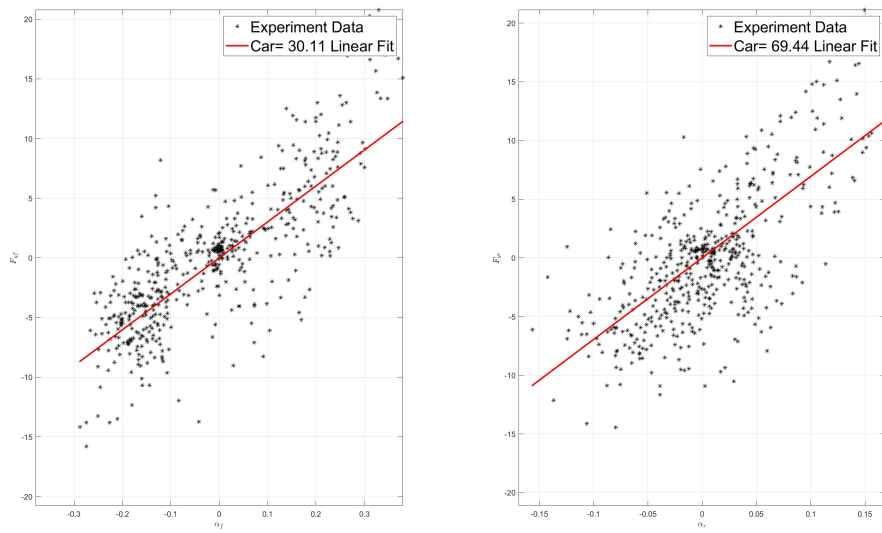


Figure 3.42: Slip versus Lateral Force graph

3.5.4.1 Results

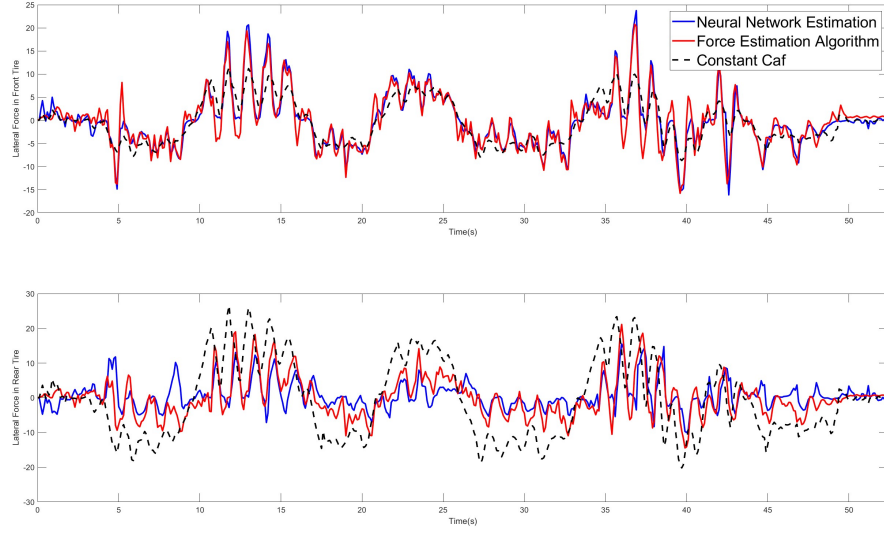


Figure 3.43: Lateral force estimated by LSTM-1 Results vs Force Estimation algorithm vs Constant C_α

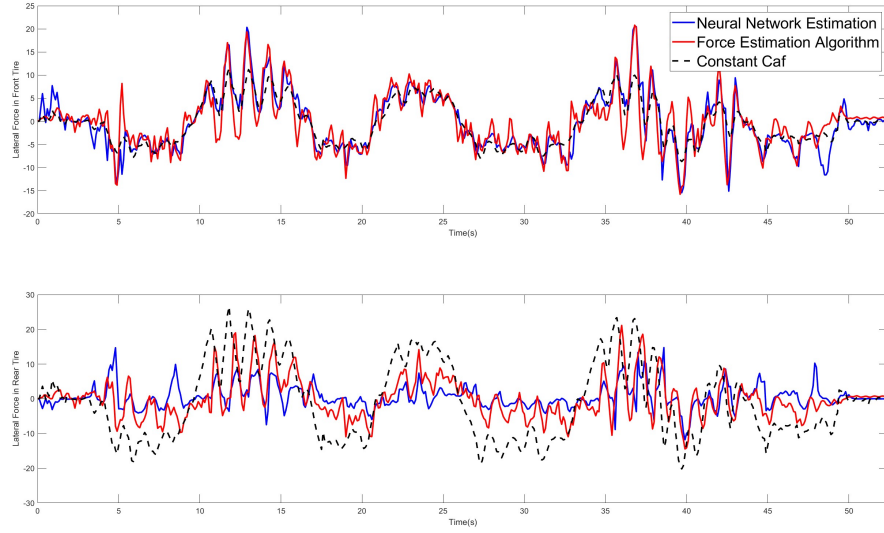


Figure 3.44: Lateral force estimated by Conv-1 Results vs Force Estimation algorithm vs Constant C_α

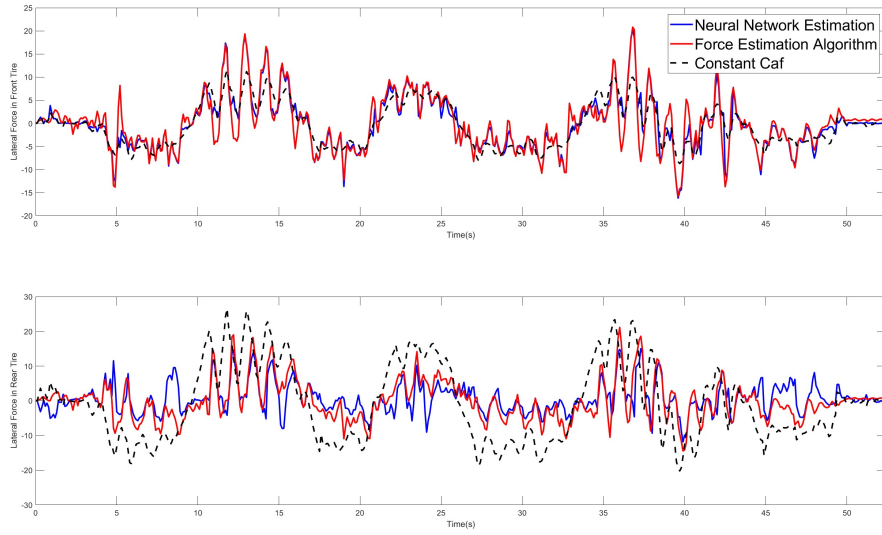


Figure 3.45: Lateral force estimated by FC-1 Results vs Force Estimation algorithm vs Constant C_{α}

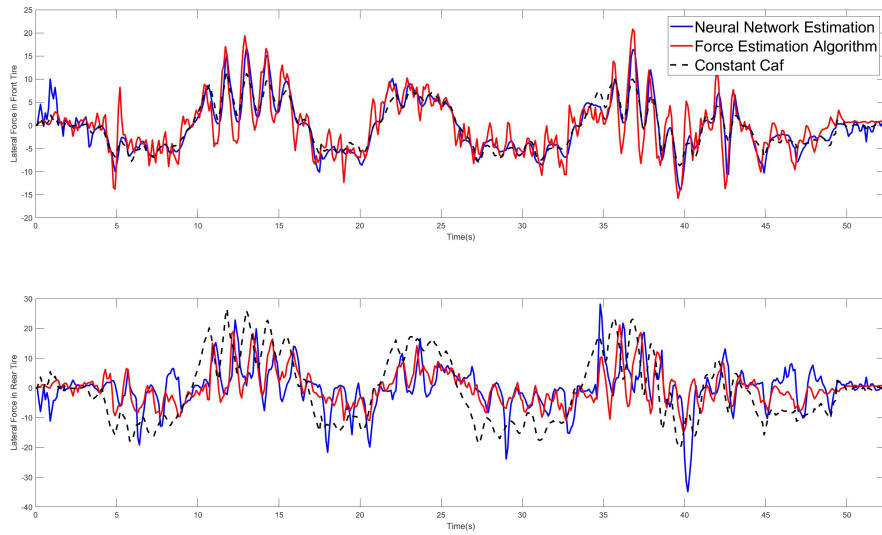


Figure 3.46: Lateral force estimated by LSTM-2 Results vs Force Estimation algorithm vs Constant C_{α}

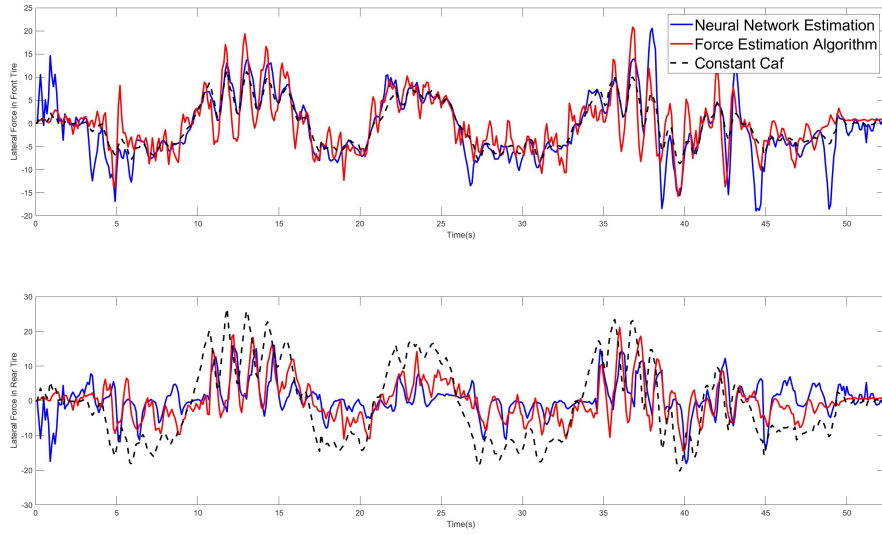


Figure 3.47: Lateral force estimated by Conv-2 Results vs Force Estimation algorithm vs Constant C_{α}

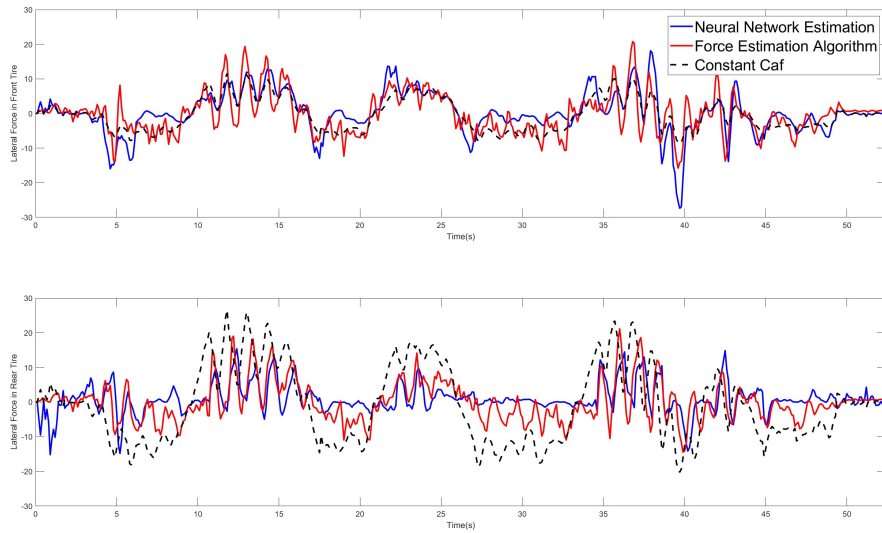


Figure 3.48: Lateral force estimated by FC-2 Results vs Force Estimation algorithm vs Constant C_{α}

Architecture	LSTM-1	Conv-1	FC-1	LSTM-2	Conv-2	FC-2	Constant
Error	4.37	5.56	3.93	6.51	7.07	7.31	6.85

Table 3.9: Error in estimated forces experiment 3

3.6 Conclusion

In all experiments, the fully connected network architecture trained with dataset 1 performed the best, with 75.47%, 65.76%, and 42.62% improvement over constant cornering stiffness on estimating lateral forces generated by tires. Because of its best performance, the FC-1 network is selected for the controller in the following chapter.

Chapter 4

Ideal Twin Driver Assistance

In this section, a novel assistance method is proposed that reduces slip in tires while generating the velocities that would be induced under ideal traction conditions from the inputs of the driver to a front wheel drive front wheel steering digital twin vehicle, under varying traction conditions by calculating the optimal control inputs under the identified conditions for a four-wheel drive four-wheel steering vehicle. This method allows the driver to drive without considering the road conditions, as if the conditions are always optimal while maintaining control. This is achieved by utilizing a nonlinear model predictive controller. The nonlinear model predictive controller receives the current states of the vehicle, driver inputs that are for an ideal digital twin of the vehicle that is front wheel drive front wheel steering, estimated cornering coefficients and computes the optimal inputs for the four-wheel drive four-wheel steering vehicle that matches the ideal vehicle with much less slip.

4.1 Controller Structure

This thesis uses a nonlinear model predictive controller [32] is utilized to control the vehicle. In Figure 4.1, the ideal vehicle model is a front wheel drive front wheel

steering bicycle model under ideal cornering coefficients. The estimated vehicle model is the 4-wheel drive 4-wheel steering vehicle model given in Section 2.2, with the cornering coefficients estimated using the method provided in Section 3. In the prediction loop, a prediction for control variables is made for the entire prediction horizon; from these predictions, velocity predictions are made by solving Equations 2.1-2.10. These predicted velocities are then compared with the velocities that are calculated in the cost function of the ideal vehicle model. The cost function's gradient in the optimization loop is computed based on the predicted input variables. Then, the predictions are updated through the SPQ minimization algorithm [33], and the prediction loop is computed again until the stop criteria are satisfied. In this optimization, the selection of the cost function is important because the cost function evaluates the optimality of the solution. In this work, the selected cost function consists of 4 parts and is given in Equation 4.1

$$J = \sum_t \gamma_1 * (\dot{x}_{ideal} - \dot{x})^2 + \gamma_2 * (\eta^2 + \delta^2) + \gamma_3 * (\dot{\eta}^2 + \dot{\delta}^2) + \gamma_4 * \alpha^2 \quad (4.1)$$

where γ are scaling constants. The first term of the cost function enforces obtaining the required velocities, the second term ensures that the magnitude of the control inputs is not unnecessarily high, the third term ensures that the control variables are smooth in time and the fourth term enforces that the slips are as low as possible. This control structure works within a multi-rate system. The ideal vehicle is simulated at 100 Hz while the controller only works on 10 Hz to save computation time. With this whole setup, the experiment can be conducted.

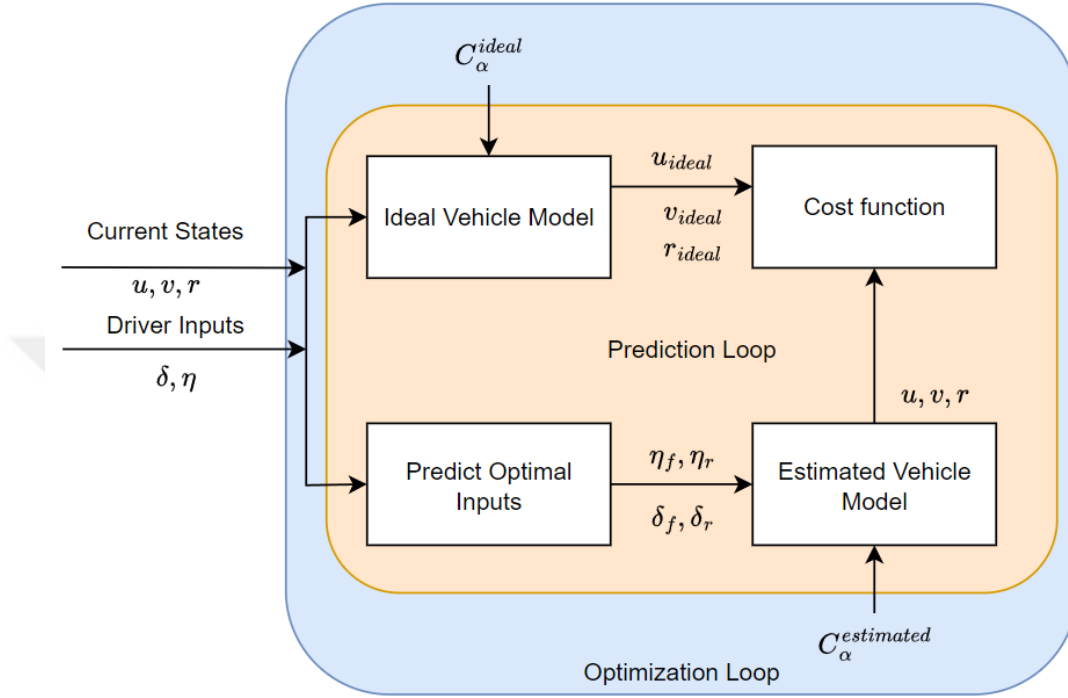


Figure 4.1: Nonlinear Model Predictive Controller Structure

4.2 Simulation Results

4.2.1 Simulation 1- Sine Inputs

In this simulation, the ideal twin vehicle is driven by sinusoidal steering and throttle inputs given in Equations 4.2,4.3. The ideal twin inputs and controller outputs are presented in Figure 4.2. The ideal system velocities, controlled system velocities, and directly driven system velocities are given in Figure 4.3. The ideal twin cornering coefficients are set to 100 N/rad in the front and rear tires. For the simulation of the actual vehicle, the cornering stiffness are set to 30.11 N/rad and 69.44 N/rad for the front and rear tires respectively.

$$\eta_f = 0.05 + 0.2 \sin\left(\frac{\pi t}{5}\right) \quad (4.2)$$

$$\delta_f = 0.2 \sin\left(\frac{\pi t}{4}\right) \quad (4.3)$$

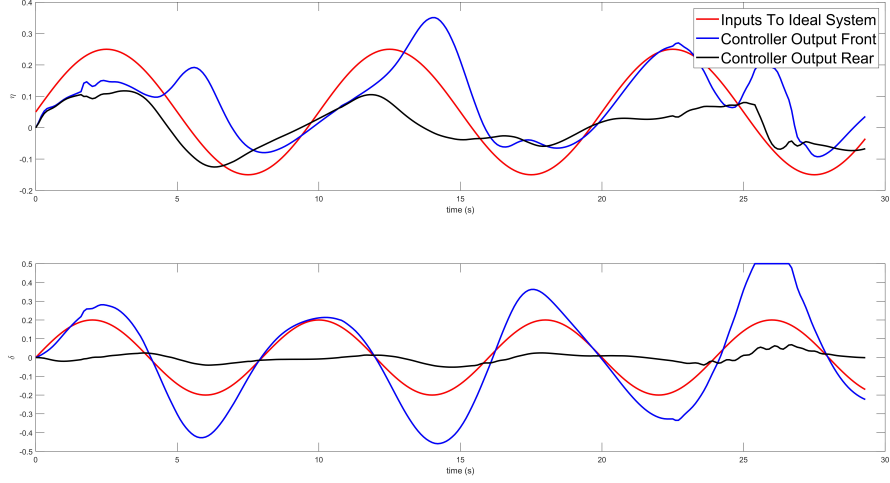


Figure 4.2: Control Inputs to ideal system and controller outputs

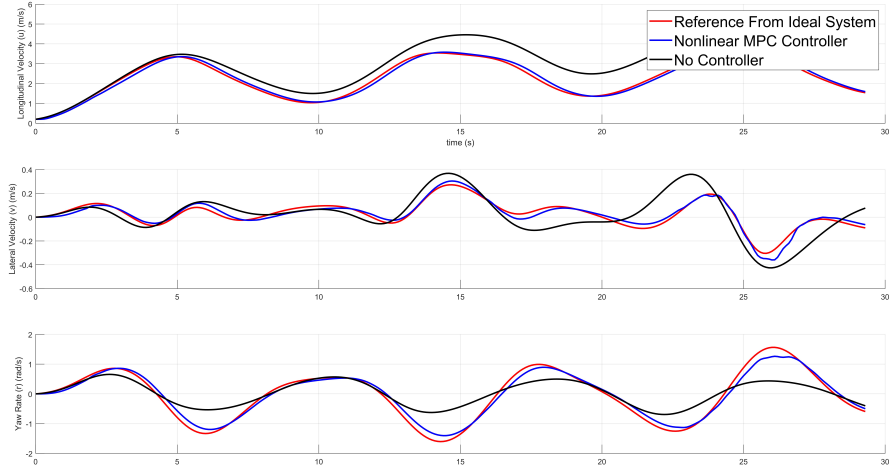


Figure 4.3: Velocities, Ideal vs Simulation vs Controller vs Direct Control

From these velocities, the reference following root means squared errors are calculated and are given in Table 4.1. The table shows that the controller successfully adapts the inputs to low cornering coefficient conditions and follows the

reference velocities for ideal conditions with 85.56 % less error than the directly controlled case.

RMSE	Controlled Vehicle Simulation	Directly Driven Vehicle Simulation
u	0.1276	1.012
v	0.025	0.107
r	0.1395	0.468
Total	0.292	1.589

Table 4.1: Root mean square error in velocities

4.3 Experiment Results

In the following experiments, the vehicle is controlled through a control PC on which the computations are run for the nonlinear model predictive controller and neural network. The computed control signals are then sent to the Jetson Nano board on the vehicle, which creates some delay discussed in Section 2.4.2. All measurements given are shifted in time to correct for these delays in the figures below. The ideal twin cornering coefficients for all experiments are set to 100 N/rad for front and rear tires.

4.3.1 Experiment 1 - Sine Input and Comparison

In this experiment, the vehicle is driven by a fixed set of inputs to compare the controller's performance with a case without a controller. The steering is selected as a sine wave when longitudinal velocity is higher than a selected set point, and throttle is given as constant when longitudinal velocity is lower than a given set

point, formulated as follows:

$$\eta_f = \begin{cases} 0.2 & \text{if } u < 2 \\ 0 & \text{else} \end{cases} \quad (4.4)$$

$$\delta_f = \begin{cases} \frac{\pi}{12} \sin(\frac{\pi}{2}t) & \text{if } u > 0.75 \\ 0 & \text{else} \end{cases} \quad (4.5)$$

With these inputs, a simulation with ideal cornering coefficients, simulations with identified cornering coefficients, an experiment with the discussed controller, and an experiment where these inputs are given to the vehicle are conducted. The control inputs and controller outputs are given in Figure 4.4, and the resulting velocities are given in Figure 4.5.

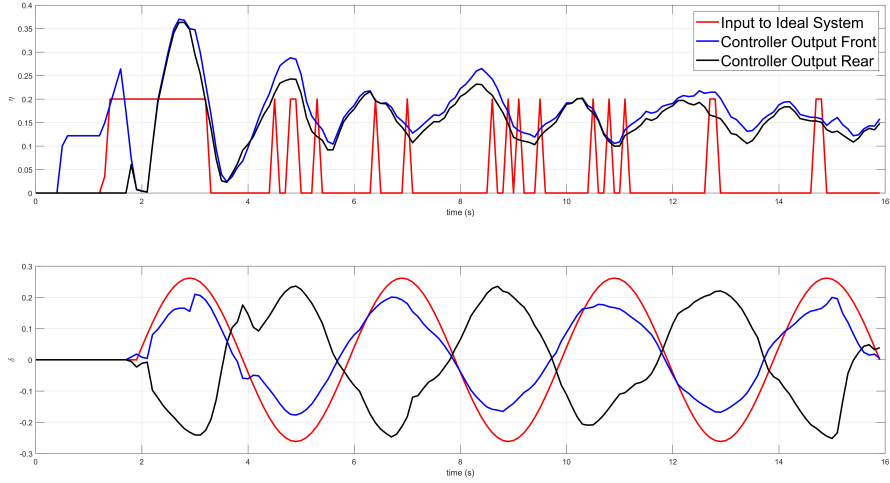


Figure 4.4: Control Inputs to ideal system and controller outputs

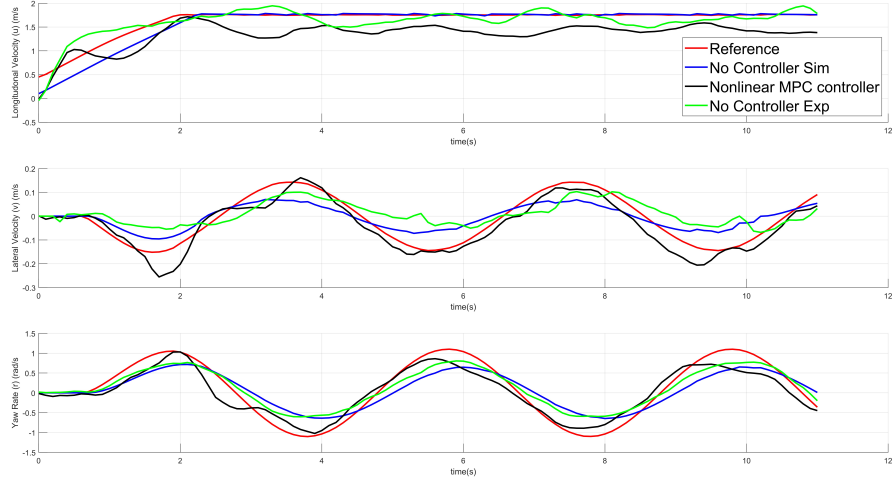


Figure 4.5: Velocities, Ideal vs Simulation vs Controller vs Direct Control

Root mean square errors between the ideal system velocities and measured velocities are calculated and given in Table 4.2. Total errors for each case are 0.62, 0.49, 0.62, respectively. However, because η is defined as a function of longitudinal velocity, comparing longitudinal velocity error is not meaningful. Excluding the longitudinal velocity, errors become 0.49, 0.29, and 0.33, which shows that the controlled system follows the reference better than the directly controlled system.

RMSE	Vehicle Simulation	Controller Experiment	Vehicle Experiment
u	0.13	0.32	0.16
v	0.14	0.04	0.06
r	0.35	0.25	0.27

Table 4.2: Error in velocities

Another way to compare the data is to examine the linear correlation coefficients between velocities. This comparison method better examines the relation between two data since it doesn't only compare numerical data; it also compares distributions. The correlation coefficients are given in Table 4.3. The mean of the

correlation coefficients is 0.96, 0.93, 0.88, respectively. Excluding the longitudinal velocity for the same reasons as before, the mean of the correlation coefficients becomes 0.94, 0.95, 0.90. This shows that the controller works better than the directly controlled case.

Correlation Coefficient	Vehicle Simulation	Controller Experiment	Vehicle Experiment
u	0.98	0.89	0.85
v	0.95	0.95	0.82
r	0.93	0.96	0.98

Table 4.3: Error in velocities

4.3.2 Experiment 2 - Human Driver

In this experiment, the ideal vehicle is driven by a human driver through a game controller connected to the main PC. The controller computes the optimal control signals for the vehicle to follow the reference generated by the human driver. The longitudinal velocity of the ideal vehicle is limited by limiting the throttle input in the same way as Section 4.4.1 and formulated as follows:

$$\eta_f = \begin{cases} \eta & \text{if } u < 2 \\ 0 & \text{else} \end{cases} \quad (4.6)$$

$$\delta_f = \begin{cases} \delta & \text{if } u > 0.75 \\ 0 & \text{else} \end{cases} \quad (4.7)$$

The resulting inputs from the driver and computed control signals are given in figure 4.6. The resulting velocities are shown in Figure 4.7. The oscillatory behavior observed in velocities could be attributed to the communication delay between the vehicle and control PC, which is discussed in Section 2.4.2.

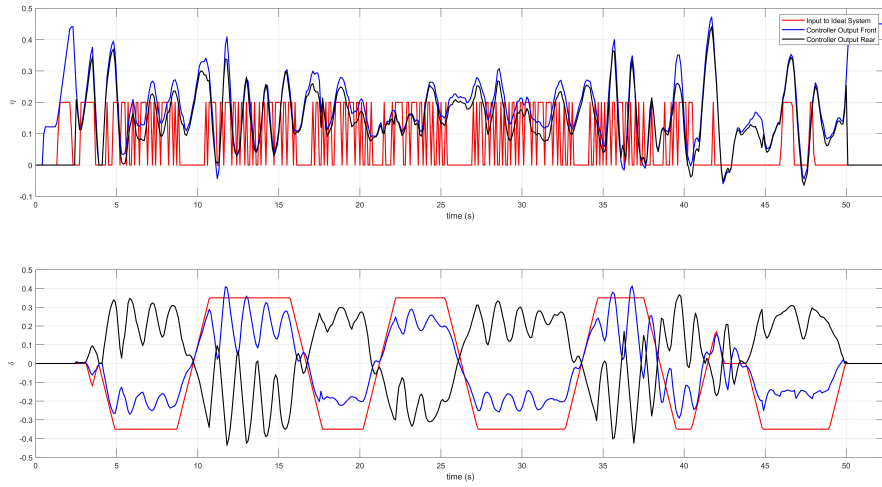


Figure 4.6: Control Inputs to ideal system and controller outputs

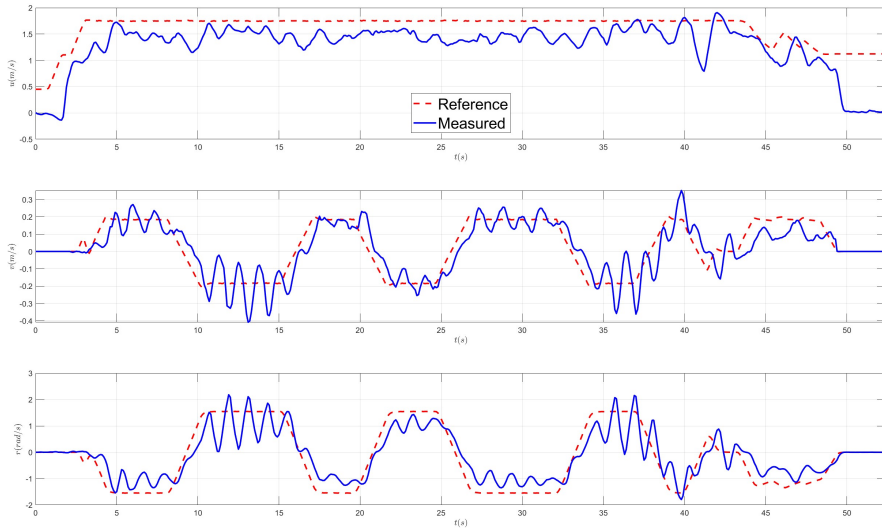


Figure 4.7: Velocities, Ideal vs Controller

Chapter 5

Conclusion and Future Remarks

In this thesis, a real-time cornering coefficient estimation method is developed. The identified cornering coefficients are used in an ideal twin driving assistance system that allows drivers to drive the vehicle without considering the current traction levels. The identification is done by training a neural network on the simulation data of the vehicle. Three different network architectures with two different data sets are evaluated for neural network training. The best possible candidate is found as a fully connected network that is trained with a dataset that has varying cornering coefficients. This selection is made by comparing the estimation performance of the neural networks with that of the experiment data. The performance of the neural networks is evaluated by comparing their estimations of lateral forces with each other and a constant cornering coefficient case. The results show that the selected fully connected network estimates the lateral force 75.47%, 65.76%, and 42.62% better on different experiment data compared to constant cornering stiffness.

The developed ideal twin driving assistance method utilizes the identified cornering stiffness by creating a perfect vehicle twin. The ideal vehicle has constant cornering coefficients determined by the driver. The velocities generated by the driver on the ideal vehicle are used as reference velocities in a nonlinear model predictive controller that utilizes the 4-wheel drive 4-wheel steering arrangement. The nonlinear model predictive controller computes the optimal control inputs to

achieve the reference velocities under the identified conditions. The simulation results show that compared to normally driving the vehicle, the driver assistance system improves the reference following by 85.56 %. The experiment results show that improvement decreases to 12.12 %. This decrease in improvement between simulation and experiment can be attributed to noise in measurements and delays in the control system.

The system's performance can be significantly improved by fine-tuning the neural networks' training and creating a more representative dataset. Another way to improve these results would be to use better hardware on the test vehicle and run the controller and the neural network on the vehicle. This would significantly reduce errors by removing the communication delay from the system.

Other estimation methods could also be utilized for better performance. Reinforcement learning could be a good alternative to the method utilized in this work.

Another way to improve this system would be to use a dynamic model with higher fidelity. This work utilizes a bicycle model with longitudinal and lateral dynamics. This modeling ignores the longitudinal tire model, friction, and load transfer from vertical plane motions. To improve the performance of the system, these principles can be considered in dynamic modeling.

Bibliography

- [1] R. W. Allen, T. T. Myers, T. J. Rosenthal, and D. H. Klyde, “The effect of tire characteristics on vehicle handling and stability,” *SAE Transactions*, vol. 109, pp. 1039–1051, 2000.
- [2] M. Reiter and J. Wagner, “Automated automotive tire inflation system – effect of tire pressure on vehicle handling,” *IFAC Proceedings Volumes*, vol. 43, no. 7, pp. 638–643, 2010. 6th IFAC Symposium on Advances in Automotive Control.
- [3] C. Lugaro, M. Alirezaei, I. Konstantinou, and A. Behera, “A study on the effect of tire temperature and rolling speed on the vehicle handling response,” *SAE Technical Paper*, 04 2020.
- [4] K. Waluś and Z. OLSZEWSKI, “Analysis of tire-road contact under winter conditions,” *Proceedings of the World Congress on Engineering 2011, WCE 2011*, vol. 3, pp. 2381–2384, 07 2011.
- [5] V. Swamy, H. Shivayogi, and N. Mathivanan, “Selection of optimal tire and design optimisation of steering system for a formula student race car through tire data treatment,” *Journal of Physics: Conference Series*, vol. 1478, p. 012032, 04 2020.
- [6] F. Gustafsson, “Monitoring tire–road friction using the wheel slip,” *IEEE Control Systems Magazine*, vol. 18, 1998.
- [7] W. R. Pasterkamp and H. B. Pacejka, “The tyre as a sensor to estimate friction,” *Vehicle System Dynamics*, vol. 27, no. 5-6, pp. 409–422, 1997.

- [8] R. R. J. O. Hahn and L. Alexander, “Gps-based real-time identification of tire-road friction coefficient,” *Dynamic Systems and Control*, 2002.
- [9] C. Leao Pereira, R. da Costa Neto, and B. Loiola, “Cornering stiffness estimation using levenberg–marquardt approach,” *Inverse Problems in Science and Engineering*, vol. 29, 05 2021.
- [10] K. Koysuren, A. F. Keles, and M. Cakmakci, “Online parameter estimation using physics-informed deep learning for vehicle stability algorithms,” in *2023 American Control Conference (ACC)*, pp. 466–471, 2023.
- [11] T. Hegedűs, B. Németh, and P. Gáspár, “Identification of tire characteristics using physics-informed neural network for road vehicles,” in *2024 32nd Mediterranean Conference on Control and Automation (MED)*, pp. 706–711, 2024.
- [12] N. Lampe, K.-P. Kortmann, and C. Westerkamp, “Neural network based tire-road friction estimation using experimental data,” *IFAC-PapersOnLine*, vol. 56, no. 3, pp. 397–402, 2023. 3rd Modeling, Estimation and Control Conference MECC 2023.
- [13] S. of Automobile Engineers, “Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles,” tech. rep., Society of Automobile Engineers, 2021.
- [14] D. S. Abdel-Aty, Mohamed, “A matched case-control analysis of autonomous vs human-driven vehicle accidents,” *Nature Communications*, vol. 15, 2024.
- [15] J. K. C.-G. Roh and I.-J. Im, “Analysis of impact of rain conditions on adas,” *Sensors*, vol. 20, no. 23, 2020.
- [16] E. Hashemi, Y. Qin, and A. Khajepour, “Slip-aware driver assistance path tracking and stability control,” *Control Engineering Practice*, vol. 118, p. 104958, 2022.

- [17] K. Han, M. Choi, B. Lee, and S. B. Choi, “Development of a traction control system using a special type of sliding mode controller for hybrid 4wd vehicles,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 264–274, 2018.
- [18] J. H. Park and C. Y. Kim, “Wheel slip control in traction control system for vehicle stability,” *Vehicle System Dynamics*, vol. 31, no. 4, pp. 263–278, 1999.
- [19] A. Musa, M. Pipicelli, M. Spano, F. Tufano, F. De Nola, G. Di Blasio, A. Gimelli, D. A. Misul, and G. Toscano, “A review of model predictive controls applied to advanced driver-assistance systems,” *Energies*, vol. 14, no. 23, 2021.
- [20] K. Spentzas, I. Alkhazali, and M. Demic, “Kinematics of four-wheel-steering vehicles,” *Forschung im Ingenieurwesen*, vol. 66, no. 5, pp. 211–216, 2001.
- [21] H. Tourajizadeh, M. Sarvari, and S. Ordoo, “Modeling and optimal control of 4 wheel steering vehicle using lqr and its comparison with 2 wheel steering vehicle,” in *2018 6th RSI International Conference on Robotics and Mechatronics (IcRoM)*, pp. 106–113, 2018.
- [22] S. Yim, “Comparison among active front, front independent, 4-wheel and 4-wheel independent steering systems for vehicle stability control,” *Electronics*, vol. 9, no. 5, 2020.
- [23] M. Deremetz, R. Lenain, A. Couvent, C. Cariou, and B. Thuilot, “Path tracking of a four-wheel steering mobile robot: A robust off-road parallel steering strategy,” in *2017 European Conference on Mobile Robots (ECMR)*, pp. 1–7, 2017.
- [24] *A Fault Tolerant Vehicle Stability Control Using Adaptive Control Allocation*, vol. 1 of *Dynamic Systems and Control Conference*, 09 2018.
- [25] A. G. Ulsoy, H. Peng, and M. Çakmakci, *Automotive control systems*. Cambridge University Press, 2012.

- [26] N. L. Bakker, E. and H. Pacejka, “Tyre modelling for use in vehicle dynamics studies,” *SAE Technical Paper*, 1987.
- [27] T. Parks and J. McClellan, “Chebyshev approximation for nonrecursive digital filters with linear phase,” *IEEE Transactions on Circuit Theory*, vol. 19, no. 2, pp. 189–194, 1972.
- [28] N. Lehtomaki, N. Sandell, and M. Athans, “Robustness results in linear-quadratic gaussian based multivariable control designs,” *IEEE Transactions on Automatic Control*, vol. 26, no. 1, pp. 75–93, 1981.
- [29] J. Dormand and P. Prince, “A family of embedded runge-kutta formulae,” *Journal of Computational and Applied Mathematics*, vol. 6, no. 1, pp. 19–26, 1980.
- [30] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014.
- [31] T. F. Coleman and Y. Li, “An interior trust region approach for nonlinear minimization subject to bounds,” *SIAM Journal on Optimization*, vol. 6, no. 2, pp. 418–445, 1996.
- [32] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control*, pp. 43–66. London: Springer London, 2011.
- [33] S. J. W. Nocedal, J., *Numerical Optimization*. Springer Verlag, 2006.