Data-Driven Hardware-in-the-Loop Plant Modeling for Self-Driving Vehicles

Hannah Grady, Nicholas Nauman, and Md Suruz Miah
Department of Electrical and Computer Engineering
Bradley University
Peoria, Illinois, 61615, USA
E-mail: {hgrady,nnauman}@mail.bradley.edu, smiah@bradley.edu

***Abstract*—In this paper, we present data-driven hardware-in-the-loop (HIL) plant models of different subsystems of a self-driving vehicle. Despite numerous concerns, the automotive industry is still investing remarkable resources for producing self-driving vehicles to be available in the market. Among numerous challenges in the development process of such vehicles are the validation and testing process of various subsystems. Here we provide data-driven models of different subsystems so that automotive industries can validate and test autonomous vehicles without the need of a physical vehicle, which indeed reduces the considerable amount of cost for automotive industries. The vehicle subsystems considered in this work include steering, acceleration, brake, shift, speed, and speed control subsystems. Each of these subsystems is either a multi-input single output or single-input single output system. A Lexus RX450H self-driving vehicle is employed to collect raw data (inputs and outputs data for different subsystems) offline. We used the deep learning toolbox available in the commercial software package, MATLAB/SIMULINK, for modeling each of these systems. Therefore, the results of such modeling could be used for validating and testing without the need for a physical self-driving vehicle. As such, the proposed modeling results could be useful for reducing cost of vehicles' development process since a physical vehicle is not required for validation and testing. However, the performance of the steering, acceleration, and brake subsystems demonstrated for conciseness purpose. The contribution of this project is twofold. The first part is that we were able to take real time raw data collected from a Lexus RX450H vehicle and use it to develop machine learning models to represent the vehicle subsystems. The second portion of our contribution is that we took the models that we created and tested them using Hardware-in-the-Loop.**

***Index Terms*—Data-driven modeling, deep learning, hardware-in-the-loop, self-driving vehicle, vehicle subsystems**

I. Introduction

Due to escalating demand of self-driving vehicles, automotive industry is gradually shifting its attention in manufacturing a variety of autonomous vehicles. The advancement of new technologies led the automotive industry to develop such vehicles for commercial and personal use as they mainly aim to reduce crashes, energy consumption, pollution, and congestion. However, the development of process of autonomous vehicles still faces signifant challenges to overcome. As it is apparent, each self-driving vehicle is composed of varaious subsystems that researchers of autonomous vehicle community analyze, design, develop, and test. Intuitively, self-driving vehicles require thorough testing at the factory level before their demonstrations for the commercial purpose. Such a testing procedure for each physical vehicle is inefficient and costly as it involves testing of different subsystems, such as steering, brake, shift, to name a few, of a vehicle. In this work, we model different modules/subsystems of a self-driving vehicle using machine learning techniques, where a physical self-driving vehicle is not required for testing.

Note that modeling vehicle subsystems is a pre-requisite for the development of reliable controllers for autonomous vehicles to be managed in the era of modern transportation systems in general. In particular, this work models different subsystems for a self-driving vehicle of the vehicle fleet shown in Fig. 1. Such modeling process would allow companies to continue crucial deliveries or transports of their products even if there is a shortage of drivers.

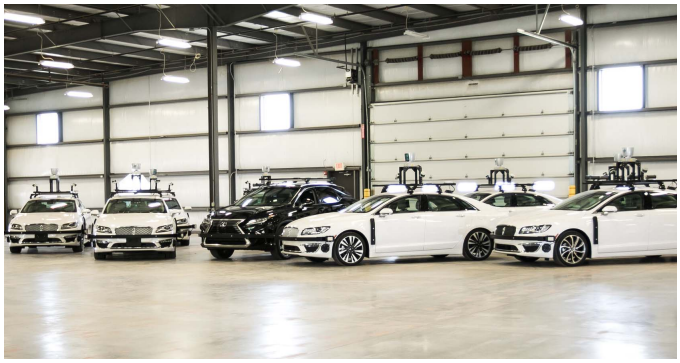


Fig. 1: Autonomous vehicle fleet (courtesy of AutonomouStuff Inc., https://autonomoustuff.com)

Autonomous vehicles have the ability to make roads safer for drivers and pedestrians alike. To develop a reliable and safe transportation system for modern world, a large body of research in the field of autonomous vehicles is being conducted in recent decades [1], [2]. In fact, the research on this topic has been conducted from a variety of perspectives, such as the application of connected and autonomous vehicle technology [3], navigation and path control of self-driving vehicles [4], [5]. Authors in [6] worked on active front steering for steer-by-wire vehicle. Steering system of high speed intelligent vehicles has been modeled using system identification [7]. Therefore,

This work was partially supported by AutonomouStuff (https://autonomoustuff.com/)

it is clear that numerous research pertainting to self-driving vehicles has been pursued, especially, in the past two decades. There are two operating modes considered for the autonomous vehicle employed in this work:

- Manual-drive and
- by-wire or autonomous [6].

Manual-drive operation occurs when the driver has complete control over the vehicle, including controlling the steering, speed, acceleration, and braking of the vehicle. Autonomous operation occurs where the vehicle is controlled by a computer under driver supervision. Usually, there are six main subsystems of a self-driving vehicle: 1) Steering, 2) acceleration, 3) brake, 4) shift, 5) speed, and 6) speed control (cruise). These subsystems are to be modeled to get an accurate representation of how the vehicle is to be controlled. The first subsystem we shall model is the steering model, and then we will move onto the other subsystems. There are already controllers in place for the Lexus vehicle platform, but their ability to match the desired input specified by the vehicle operator is lower than expected due to non-linear behaviors of the torque voltages required to control each subsystem. The scope of this project includes developing mathematical and/or data-driven models for each subsystem so that one can easily implement control systems that improve the reliability of the autonomous vehicle platform. These models will be considered reliable if they can track actual vehicle subsystem behaviors with a minimum best fit percentage between 85 to 95 percent and fall within the error bounds defined for each subsystem. Once these models are developed, they will be tested using Hardware-in-the-Loop (HIL) bench. Once there is confidence in each model, these models can be used on the HIL bench to develop controllers that will remove the non-linear behaviors of each vehicle subsystem.

II. Self-Driving Vehicle Subsystems

In this section, we illustrate subsystems of the self-driving vehicle highlighting the hardware-in-the-loop modeling process conducted in the work. As mentioned, we focus on six main subsystems which are the steering, acceleration, brake, speed, shift, and speed control systems. Fig. 2 outlines these six subsystems in a Lexus self-driving vehicle provided by the company, Hexagon | AutonomouStuff (www.autonomoustuff.com). However, actual self-driving vehicle (Lexus RX450H) that is used for collecting input-output data for all of these subsystems is shown in the picture of Fig. 3. A team of the project members visited the site to collect raw data for modeling steering, brake, and acceleration subsystems of the vehicle.

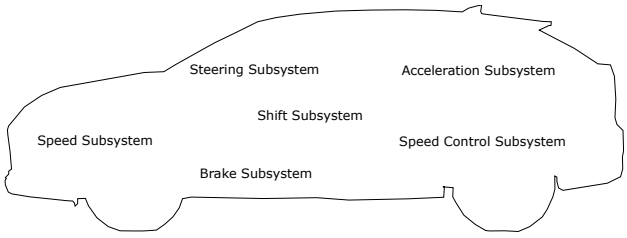


Fig. 2: Hexagon Lexus self-driving vehicle showing different subsystems

Fig. 3: AutonomouStuff Lexus RX450H vehicle.

Each of the six subsystems will be treated as a multiple-input single-output (MISO) system. For each subsystem, every input that is a torque voltage is actually two torque voltage signals, and thus can not be treated as a single input. Also, each subsystem will be treated as a single output system. The brake subsystem is shown to have two outputs but the behaviors of one of the outputs is already known so it will be modeled based on the other output's behavior. In the following we give a brief description of each of the subsystems of the self-driving vehicle modeled in this work. Each subsystem sends torque voltages to a motor that will control the subsystem.

A. Steering Subsystem

The steering subsystem of a generic self-driving vehicle consists of the steering arm, knuckle, rods, drag link, steering gear box, drop arm, cross shaft, steering column, steering shaft, and steering wheel [1]. Fig. 4 depicts steering subsystems in the by-wire (autonomous) mode. In Fig. 4, the motor would

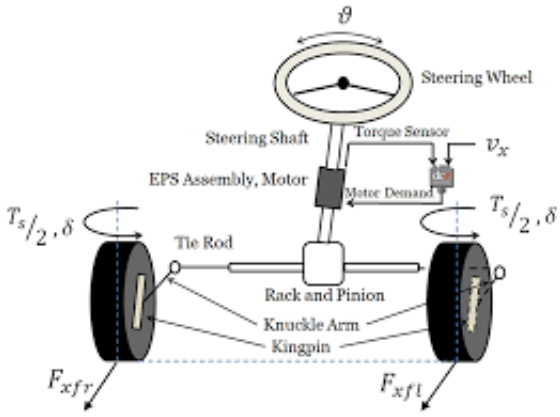


Fig. 4: Steering subsystem in autonomous mode (courtesy of Autonomoustuff.com).

turn the pinion arms which would change the steering angle of the vehicle. The idea of by-wire mode for other vehicle

[1] https://www.theengineerspost.com/car-steering-system-in-automobile/

subsystems is controlled in a similar manner with torque voltages being applied to a motor that controls the vehicle subsystem.

The ultimate goal of this subsystem is to control the steering angle for the vehicle to navigate in the desired heading. Therefore, the control system is designed to produce appropriate voltages to be applied to the power steering motors for the steering orient in the target direction. The block diagram of the control system designed for this subsystem is shown in Fig. 5, where the target and the actual orientations of the vehicle are denoted by θ^{ref} and $\theta(t)$ at time $t \geq 0$, respectively. The torque

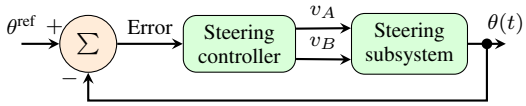


Fig. 5: Steering subsystem block diagram.

voltages applied to the power steering motor are denoted by v_A and v_B, respectively. The error signal, which is the difference between the target and the actual directions of the vehicle, is passed on to the steering controller. The outputs of the steering controller are the torque voltages that are applied to the power steering motors of the steering subsystems of the vehicle.

B. Brake Subsystem

The brake subsystem is made up of the master cylinder, brake rotor, brake drum, brake pad, brake caliper, brake shoe, brake booster, brake pedal, wheel speed sensors, ABS module, and brake lines [2]. A high-level block diagrams showing the inputs and outputs of a conventional brake subsystem of a self-driving vehicle is shown in Fig. 6. The subsystem takes

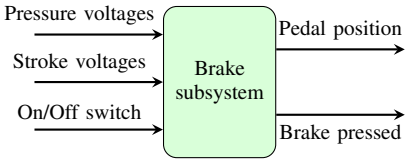


Fig. 6: Brake subsystem block diagram

the brake pedal pressure voltages, brake pedal stroke voltages, and the brake pedal on/off switch values as inputs. Using these values, it generates a new brake pedal position (in percentage) and a boolean value called brake pressed. This boolean value indicates to the user whether or not the brake pedal is being pressed.

C. Acceleration Subsystem

The acceleration subsystem consists of the throttle pedal, engine, and crankshaft [3]. It is a multiple-input multiple-output system, whose block diagram is depicted in Fig. 7. Taking the acceleration pedal voltages as the input, the subsystem then produces the acceleration pedal position as the output. The output represents how much the pedal is pressed as a percentage.

[2]https://oards.com/car-brake-system-components/

[3]https://harrisautomotiverepair.com/car-accelerator/

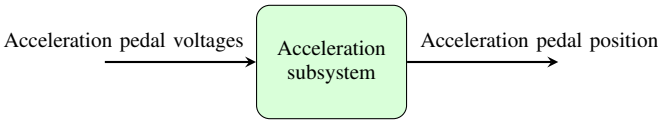


Fig. 7: Acceleration subsystem block diagram

D. Shift Subsystem

The components that make up the shift subsystem are the friction clutch, bands, spring loaded valve, load sensor, shift valve, torque converter, gear stick, seals, and gasket [4]. This subsystem can be classified as a single-input single-output system. The subsystem takes the desired shifter gear value from the user as the input. Within the subsystem, the actual shifter gear is changed to better reflect the desired gear. This actual shifter gear value is then the output of the shift subsystem.

E. Speed Subsystem

This speed subsystem would fall under the multiple-input single-output system. The subsystem has four inputs, the acceleration pedal position, brake pedal position, and shifter gear. Using these inputs, the speed subsystem finds the vehicle speed. This vehicle speed is then the output of the system.

F. Speed Control Subsystem

The speed control subsystem components include the cruise control switch, vehicle speed sensor, throttle sensor, and brake switch [5]. The Desired Vehicle Speed input is set by the user and sent to the speed control subsystem. Taking this input, the subsystem calculates the Actual Vehicle Speed. This value is then sent out to the rest of the vehicle system.

III. Modeling Vehicle Subsystems

To start the modeling process, we first collected raw data of the subsystems illustrated in the previous section. The input-output data are then used to model different subsystems of the self-driving vehicle using deep learning toolbox available in a commercial modeling software SIMULINK[6]. To demonstrate the modeling accuracy, to evaluate the effectiveness of our models, and to develop safe and reliable controllers for each subsystem, the following system requirements are to be met:

- Each vehicle subsystem will be able to track nonlinearities associated with small changes in the output of each subsystem.
- Each vehicle subsystem plant model will track any desired inputs within the specified error bounds:
  - Steering angle within 5 degrees
  - Pedal positions within 5%
- Each vehicle subsystem will be modeled independently from other vehicle subsystems

[4]https://oards.com/car-automatic-transmission-system-components/

[5]https://www.freeasestudyguides.com/electrical-cruise-control-systems.html

[6]https://www.mathworks.com/products/simulink.html

The vehicle plant models will satisfy the requirements listed below:

- The resulting plant model will consist of accurate subsystem models, as defined above
- The subsystems can be used to create a hardware-in-the-loop (HIL) testbench
- The subsystems can handle very small changes in their output accurately

All of the subsystems have nonlinear behaviors when there are small changes in the output of the subsystem. The steering subsystem, for example, should behave in a smooth, continuous manner. However, a team of researchers from AutonomouStuff observed that when trying to implement features such as lane tracking for the Lexus vehicle platform, that when small changes in the steering angle (less than five degrees) occurred, the torque voltages would momentarily stall and then suddenly change causing a more drastic change. This behavior made it very challenging to complete features like lane tracking. The plant models developed in this work will aid in the development of controllers that will remove the nonlinearities allowing features like lane tracking to be implemented in a safer, smoother manner.

A. Steering System Modeling

To develop a model of the steering system, we employed multiple methods. The first method that we tried was generating a NARX model and a transfer function model, making use of the System Identification Toolbox in MATLAB. The NARX model we developed was not accurate and did not track the output of the steering system data. The transfer function model we developed was more reliable tracking the output of the steering system data, but was unable to meet the error requirements, as there were sudden changes that forced high error peaks. Then we employed a neural network model, making use of the Neural Network Time Series Toolbox in MATLAB. With the neural network model, the model was able to reliably and accurately track the output of the steering system data and stayed within the error requirements.

B. Acceleration System Modeling

To develop a model of the acceleration system, we employed multiple methods. The first method that we tried was generating a NARX model and a transfer function model, making use of the System Identification Toolbox in MATLAB. The NARX model we developed was accurate for the by-wire model and tracked the output of the acceleration system data, but did not meet the error requirements. The transfer function model we developed was more reliable tracking the output of the acceleration system data for the manual mode operation, but was unable to meet the error requirements, as there were sudden changes that forced high error peaks. Then we employed a neural network model, making use of the Neural Network Time Series Toolbox in MATLAB. With the neural network model, the model was able to reliably and accurately track the output of the acceleration system data and stayed within the error requirements.

C. Brake System Modeling

To create a brake system model, we first tried generating transfer function models using MATLAB's System Identification Toolbox. After generating multiple transfer function models and trying many different data log combinations, we still couldn't find a model that was completely accurate across all combinations and that was below the error bound of 5% for all combinations. Once this became clear Neural Network models were created using the Neural Network Time Series Toolbox provided by MATLAB to see if they could create accurate brake models. Using this we were able to create a model that accurately tracked the data we collected from the brake system and that stayed within the 5% error bound.

IV. Validation and Testing

A. Experimental Setup

We visited the workplace of AutonomouStuff in order to collect data from the steering, acceleration, and braking subsystems in an autonomous vehicle. The data we collected will be used to generate and then verify our models. We collected data on the Lexus RX450H vehicle platform shown in Fig. 3. The following the hardware components required to collect the data:

- Laptop
- PACMod ECU
- CANCase
- CAN bus

The interfacing software, Vector CANAlyzer, installed in the laptop is used to pass commands or log data, such as steering angle or acceleration or brake pedal position. This data is sent or received using the CANCase and CAN bus. These are connected to the AutonomouStuff designed PACMod ECU, which sends torque voltages to the desired vehicle subsystem allowing the laptop to either control the desired vehicle system or log data. The Vector CANAlyzer software is installed on the laptop and is used to parse the collected data that is sent from the CANCase. This is how we were able to collect logs of data that we would use to develop plant models of the autonomous vehicle subsystems.

Each subsystem that we are modeling is set up in a similar manner. In manual mode, the torque voltages that control each subsystem are sent by the vehicle's electronic control unit (ECU). In order to control the vehicle autonomously, the vehicle subsystem switches to by-wire mode. In by-wire mode, the torque voltages from the vehicle's ECU are discarded by open-circuiting the motors that control each subsystem. Instead, the PACMod ECU built by AutonomouStuff sends the torque voltages to the motor using relays. In Fig. 8, the experimental setup for data collection is shown. The laptop is used to collect the data from the desired vehicle subsystem by the use of Vector's CANAlyzer software. The CAN Case collects data from the PAC Mod and ECU and sends the data using a CAN bus to the laptop which is then parsed and displayed through the use of CANAlyzer. The ECU and PAC

Fig. 8: Setup for collecting data in the Lexus RX450H self-driving vehicle.

Mod are not shown in Fig. 8 as they are fixed behind panels of the vehicle.

Using the data we collected for the steering, acceleration, and braking subsystems, we initially separated the data so we could analyze the by-wire and manual modes individually. This effort was made to identify if the models we developed could be used interchangeably regardless of what mode the autonomous vehicle subsystems were operating in. After analyzing the models we developed, we determined that they are not interchangeable as there were differences in the models. When the vehicle subsystem is in by-wire mode, the controller generates the torque voltages that are applied to each subsystem motor. As a result, we decided the most accurate model of each subsystem would be developed using the data when the subsystems are in manual mode.

To develop the model of the steering, acceleration, and brake subsystems using a neural network architecture, we used one data subset to train the model and then used an additional subset of data to verify the model worked. The neural network model used a feedforward network that had one hidden layer containing three neurons. The model was also trained with twelve delay units to achieve the required accuracy, meaning that the first twelve output samples of the model should be ignored.

B. Neural Network Modeling

1) Steering System: In Fig. 9, the top plot shows the simulated output of the neural network when the input torque voltages and desired steering angle from a subset of data (subset #1) collected offline is fed into the model. The bottom plot shows the error between the desired steering angle and the output of the neural network. For the steering model, the error should be less than 5 degrees or approximately 0.1 radians.

2) Acceleration System: In Fig. 10(a), the top plot shows the simulated output of the neural network when the input torque voltages and desired acceleration pedal position from

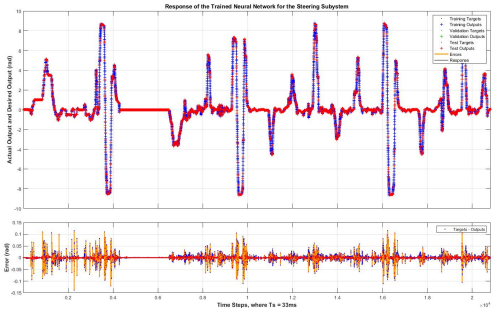
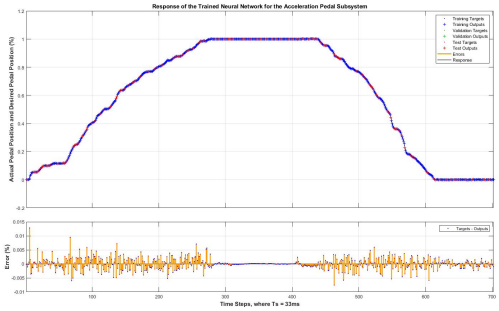
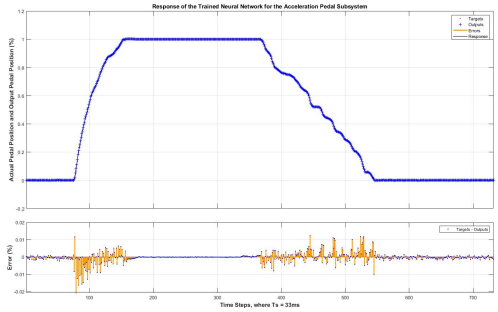


Fig. 9: Steering model performance in tracking desired steering angle and the error based on the data (subset 1).

data subset 1 are fed into the model. The bottom plot shows the error between the acceleration pedal position and the output of the neural network. For the acceleration model, the error should be less than 5% or 0.05. Fig. 10(b) shows the same plots except data subset 2 was used instead.



(a) Model output tracking and Error Plots for Data Subset 1

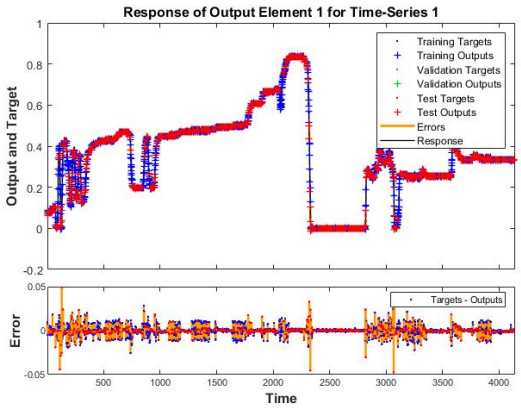


(b) Model Output Tracking and Error Plots for Data Subset 2

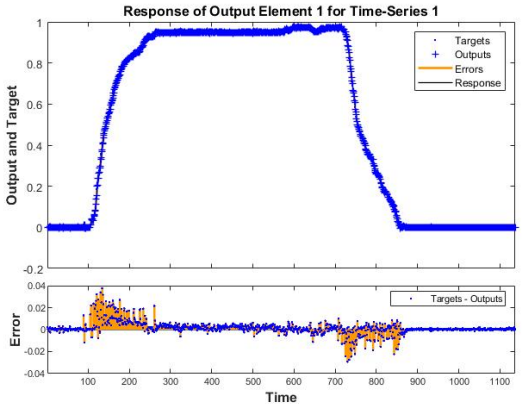
Fig. 10: Acceleration System Training Plots

3) Brake System: The brake system was modeled using only the manual log data. It was trained using the data from the first brake log, and the result and error plots can be seen in Fig. 11(a). The plots show that the model is able to predict the outputs relatively accurately, along with keeping the error below 5%. In order to further verify the accuracy of the model, it was tested using data from the second brake log. The results

from that test, shown in Fig. 11(b), also show that the model can accurately predict the output values and do this within the accepted error bound of 5%. The graphs show how the model tracks the data log and the error between the model output and the data log output. The y-axis for both graphs, "Output and Target" represents the Actual and Desired Brake Pedal Position as a percentage. The x-axis for both graphs represents the Time Steps, with T_s equal to 33ms.



(a) Model Output Tracking and Error Plots

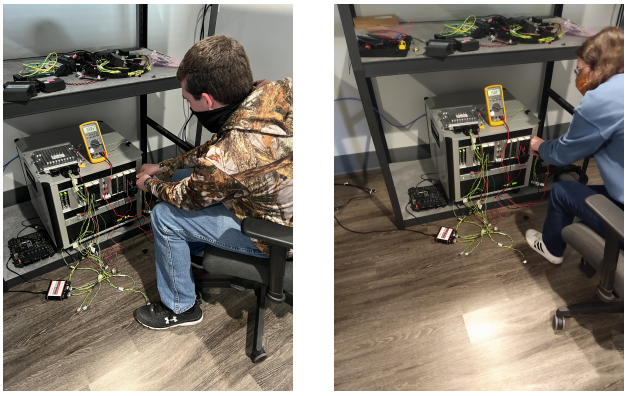


(b) Model Output Tracking and Error Plots for Log Two Data

Fig. 11: Performance of the brake system modeling.

C. Discussion

For each vehicle subsystem, the results show that the neural network models were the most accurate models. Other models that we developed, such as the transfer function models, were relatively accurate when tracking the data, but those models were consistently falling outside of the error bounds that were defined at the start of the project. The neural network models, however, track the data more accurately and reduced the error significantly so that all error requirements were met. We first proved that the neural network models that we developed were accurate using subsets of data as inputs to a Simulink model. Then we visited AutonomouStuff to generate the models on their HIL bench, as shown in Fig. 13, using Control Desk to



(a) (b)

Fig. 12: Verifying voltage measurements on the HIL bench.

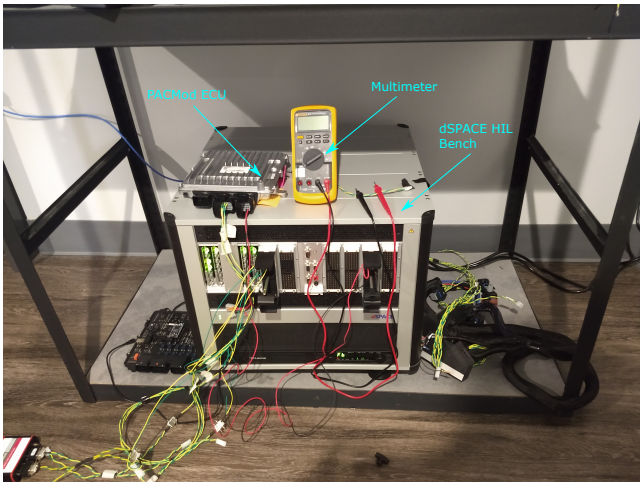


Fig. 13: AutonomouStuff HIL bench configuration

interface. Due to time constraints, AutonomouStuff was not able to get the proper hardware for this project connected to the HIL bench but we were able to test our models in an open-loop configuration and measure outputs to show that the correct voltages would be fed to any hardware that would be connected, shown in Fig. 12. One downside to using neural networks as the plant model is it causes some complexities when trying to include these models into the HIL bench software for use in developing and testing controllers.

V. Conclusion and Future Work

In this paper, we presented the effectiveness of the data-driven hardware-in-the-loop plant modeling process using deep learning. The neural network time series modeling method allowed us to train models using the data we collected. These models that were developed met the predefined accuracy requirements, and so were tested using an open loop setup. For the conciseness, we presented HIL plant modeling for three subsystems, which are steering, brake, and acceleration subsystems of the Lexus vehicle available at AutonomouStuff (the company who sponsored the project). Further testing should

be completed for these subsystem models. Since time and availability constraints prevented us from testing the models using Hardware-in-the-Loop. Further testing is important to make sure the developed models are accurate and compatible with testing using hardware. Future work could be to start developing controllers based off of these models.

REFERENCES

[1] M. Liu, G. Li, and B. Liu, "Real-time knowledge discovery from public data of internet to improve decision-making of autonomous vehicles," in *2017 Second International Conference on Reliability Systems Engineering (ICRSE)*, 2017, pp. 1–3.

[2] B. Prasad, Q. Y. Huang, and J.-J. Tang, "Development of a prototype ev autonomous vehicle for systematic research," in *2020 International Computer Symposium (ICS)*, 2020, pp. 459–461.

[3] U. D. of Transportation, "Vehicle-to-vehicle communication technology," https://www.nhtsa.gov/technology-innovation/vehicle-vehicle-communication.

[4] M.-T. Duong, T.-D. Do, and M.-H. Le, "Navigating self-driving vehicles using convolutional neural network," in *2018 4th International Conference on Green Technology and Sustainable Development (GTSD)*, 2018, pp. 607–610.

[5] E. Chajan, J. Schulte-Tigges, M. Reke, A. Ferrein, D. Matheis, and T. Walter, "Gpu based model-predictive path control for self-driving vehicles," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 1243–1248.

[6] S. A. Saruchi, H. Zamzuri, S. A. Mazlan, M. H. M. Ariff, and M. A. M. Nordin, "Active front steering for steer-by-wire vehicle via composite nonlinear feedback control," in *2015 10th Asian Control Conference (ASCC)*, 2015, pp. 1–6.

[7] B. Li, R. Wang, Y. Zhang, and Z. Wang, "Modeling of steering system of high speed intelligent vehicle by system identification," in *Proceedings of the IEEE International Vehicle Electronics Conference (IVEC'99) (Cat. No.99EX257)*, 1999, pp. 243–246 vol.1.