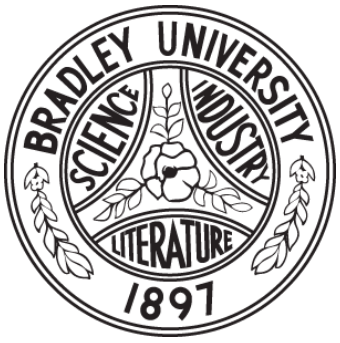
Design of a Simulink-Based Control Workstation for Mobile Wheeled Vehicles with Variable-Velocity Differential Motor Drives

Kevin Block, Timothy De Pasion, Benjamin Roos, Alexander Schmidt

Dr. Gary Dempsey

Electrical Engineering Department

May 3, 2016



ABSTRACT

A differential drive system consisting of two direct current motors can be utilized in mobile wheeled vehicles for forward, reverse, and steering operating modes, but are plagued with nonlinear characteristics in low velocity applications. Static and Coulomb friction are the most well documented of these nonlinear effects; however, cogging torque and encoder resolution can also have a significant role at low velocity. A common method of controlling differential drive systems is proportional, integral, and derivative (PID) control. When PID control is used in conjunction with a model-based design approach, the effects of the aforementioned nonlinear characteristics can be reduced. The purpose of this project was to design and implement a Simulink-based workstation that will be utilized in the development, simulation, and implementation of model-based controllers for differential drive systems. The workstation consists of two primary subsystems: the Simulink subsystem and the experimental platform. The Simulink subsystem consists of dynamic and kinematic models of a theoretical vehicle as well as a cogging torque model. The experimental platform consists of two motors and two generators to simulate the theoretical vehicle models. The experimental platform is controlled by a microcontroller and integrates with Simulink via serial communication through MATLAB®.

I. INTRODUCTION

A. Problem Background

A differential drive consisting of two direct current (DC) motors is often utilized in mobile wheeled vehicles for forward, reverse, and steering operating modes [1]. DC gear-head motors with integrated rotary encoders are normally used in these applications because of their low cost and the ability for proper matching to load conditions and desired vehicle velocities [2]. The controller and power electronics for this type of system are low cost and usually consist of a microcontroller and a dual H-bridge interface. Closed-loop velocity control can be implemented in software and high power efficiency is possible by utilizing pulse-width modulation to drive the motors.

Proportional plus integral (PI) control strategies are commonly used in differential drive systems because controllers minimize the effects of external disturbances due to motor mismatch, battery supply variation, vehicle payload, and terrain changes [3]. Typically, the PI controller is tuned to minimize steady-state error as well as settling time; however, they are not specifically tuned for optimal rejection of disturbances in most applications. Better control systems can be designed if motor nonlinearities are identified and used to develop model-based control algorithms. The best results are yielded when these algorithms are used in conjunction with PI tuning methods. Adaptive controllers can also improve performance in systems with strong nonlinearities [7, 11-13]. Identification of the nonlinear parameters can be used to explain differences between predicted and actual system performance, which is critical for high-volume applications [4-10].

Although nonlinear characteristics exist in motor drive systems, the three friction (static, Coulomb, and linear viscous) parameter model is considered adequate for the majority of velocity control applications [4]. Nonlinearities such as Stribeck friction [3], cogging torque [14], and rotary encoder resolution [15] become more important in position control and low-velocity applications. Cogging torque results from the interaction between the motor's permanent magnets and the commutator segments. The effect of this nonlinearity is observed as torque or velocity ripple. In velocity control applications, this ripple results in loss of accuracy at low velocities. At medium and high velocities, cogging torque is filtered by the motor's inertia. The part of a closed-loop control system that is the most sensitive to these non-linear effects is the feedback sensor [4]. The quantization error of the rotary encoder will limit the overall accuracy of the velocity or position measurement. It is noted that many motor and sensor nonlinearities can be minimized by purchasing higher cost motors equipped with high resolution rotary encoders.

As previously mentioned, model-based control algorithms can be implemented to reduce motor and vehicle nonlinearities without purchasing higher cost motor platforms; however, in order to reduce vehicle nonlinearities a vehicle model must be developed. Vehicle modeling involves the design of a software or mathematical representation of a physical vehicle. Vehicle models are used in advanced control methods to improve path following [1, 3]. In addition to reducing nonlinearities and improving path following, the vehicle model can be used as a test bed for the differential drive control algorithms. An ideal vehicle test bed would consist of a dynamic and kinematic model [3].

Simulink is one of the most popular platforms for physical system modeling [16]. Publications exist for modeling mobile wheeled vehicles in the Simulink environment; however, the majority of these publications do not include all of the nonlinear characteristics of the drive system at low velocities [3]. Typically, the vehicle models include a kinematic subsystem, but neglect the effect of dynamics (mass, inertia, friction). Simulink is being used by companies such as Caterpillar Incorporated, Northrop-Grumman, and Boeing to model complete products. Many of these large-scale products require years to develop actual prototypes for engineering testing. Development of a physical system model in the Simulink environment allows multiple engineers to work on their individual subsystems. In many cases, the vehicle or aircraft must be designed in simulation first to minimize engineering costs.

B. Problem Statement

The project objective was to design a control workstation that consists of a Simulink model, an experimental platform, and a graphical user interface (GUI). The Simulink model accurately depicts a physical differential steering control system used in wheeled mobile vehicle applications. The model includes nonlinear motor and sensor characteristics that are present at low velocity as well as a dynamic and kinematic vehicle subsystem. This subsystem can be used to test the tracking performance of the steering control system under various modeled internal and external disturbances such as motor mismatch, vehicle mass and inertia, battery supply changes, and terrain variations. Model-based control algorithms have been developed to reduce the effect of these nonlinearities and disturbances. The best algorithm is included in the final Simulink product. The experimental platform is a physical system capable of simulating the use of the theoretical vehicle being modeled in Simulink. The primary components of the experimental platform are four DC brushed motors and an 8-bit microcontroller. The experimental platform is low-cost and designed to accommodate variable speeds, variable loads, operate reliably over a wide temperature range, and has the ability to minimize the effects of internal and external disturbances. The GUI is used to control the Simulink model and the experimental platform and also is used to monitor either system as they operate.

C. Constraints of the Solution

The constraints in Table I were defined by discussions with Dr. Gary Dempsey.

TABLE I. CONSTRAINTS OF THE PROJECT

Constraints
24 volt Pittman motor
8-bit microcontroller
Operate in an ambient temperature range of 0° to 45° Celsius
Blocks and functions from standard Simulink library
Stable
Safe
Commands limited to prevent saturation
Controlled by GUI

The first two constraints deal with the limitations of the physical components of the system. The next five constraints deal with how the system operates and how it responds to different situations. Those five constraints are needed to ensure that the control workstation is useable for other engineers that were not involved in its development. The last five constraints also ensure that the workstation can be used repeatedly and provide consistent results over many different tests.

II. STATEMENT OF WORK

A. *Nonfunctional Requirements*

The nonfunctional requirements of the project can be seen in Table II. There are three nonfunctional requirements for the project; they are listed in order of importance from most to least important. Each of the three nonfunctional requirements has a metric which can be seen in Appendix C.

TABLE II. NONFUNCTIONAL REQUIREMENTS OF THE SYSTEM

Nonfunctional Requirements
The experimental platform and Simulink model should be reliable
The velocity command should be easy to issue to the Simulink model and experimental platform
The load of the Simulink model and experimental platform should be easy to manipulate

The first nonfunctional requirement is that the experimental platform and Simulink model shall be reliable. Reliability means that the systems shall perform its designed function throughout the entire range of possible user inputs. The second nonfunctional requirement is that the velocity command shall be easy to issue to the Simulink model and the experimental platform. This means that the user should experience minimal difficulty when issuing a velocity command. The third nonfunctional requirement is that the load of the Simulink model and experimental platform shall be easy to manipulate.

B. *Functional Requirements*

The project has three different subsystems. The three subsystems are: the Simulink model, the experimental platform, and the GUI. These three subsystems, when combined, make up the workstation. The functions that correspond to each subsystem can be seen in Table III. The first subsystem in the project is the Simulink model. This subsystem has five functions, each of the five functions deal with how the Simulink system will model the theoretical vehicle. The five functions all have a specification that starts with the phrase "Model to within…" This phrase means that the Simulink model will have the same result as the experimental platform within a certain percentage. The second subsystem is the experimental platform. This subsystem has only one function and it relates to the DC generator loads. The specification for this function also starts with the phrase "Model to within…" and it means that the DC generator loads will be the same to within ±50% as the theoretical vehicle torque disturbances. The third group of functions relate to the overall system. There are four functions in this section, two of the specifications for

the functions start with the phrase “Difference between input and output of….” These specifications mean that if there is an error of some sort due to an input command, then the output will match the input command to within the specified value. The other type of specification for the controller subsystem is a specification that starts with the phrase “Shaft rotations per minute (RPM) change of….” This phrase means that if there is a change that occurs to the system due to external torque or some other source that the controller will make sure that the RPM does not change outside of the specified range.

TABLE III. TABLE OF SUBSYSTEMS, THEIR FUNCTIONS, AND SPECIFICATIONS

System	Function	Specification
Simulink Subsystem	The Simulink Motor model shall accurately model the physical motors	Performance Specification: Model to within ±20%
Simulink Subsystem	The rotary encoder shall be accurately modeled	Performance Specification: Model to within ±20%
Simulink Subsystem	Cogging torque shall be accurately modeled	Performance Specification: Model to within ±50%
Simulink Subsystem	Pulse-width modulation shall be accurately modeled	Performance Specification: Model to within ±20%
Simulink Subsystem	Simulink H-Bridge model shall accurately model the physical H-Bridge	Performance Specification: Model to within ±20%
Experimental Platform Subsystem	The DC generator loads shall be designed to mimic the prototype vehicle	Performance Specification: Model to within ±50%
Overall	The drive control system shall minimize the effect of external torque disturbances	Performance Specification: Shaft RPM change of less than or equal to 40%
Overall	The drive control system shall reduce vehicle tracking errors for step and ramp commands	Performance Specification: Difference between input and output of less than or equal to 20%
Overall	The drive control system shall reduce vehicle tracking errors for parabolic commands	Performance Specification: Difference between input and output of less than or equal to 40%
Overall	The drive control system shall reduce the effects of motor mismatch	Performance Specification: Shaft RPM change of less than or equal to 15%
Graphical User Interface Subsystem	The graphical user interface shall send and receive commands	Interface Performance Specification: Communicate successfully with the Simulink model and experimental platform

The proposed design for the project consists of the three subsystems seen in Fig. 1: the Simulink subsystem, the experimental platform, and the GUI. The design that was chosen uses a PID model-based controller with feed-forward compensation, RS-232 serial communication, a current source, LMD18200 H-Bridge, ATmega128 microcontroller, a combination of standard library blocks and equation-based models for Simulink, and Simulink for the design and implementation of the vehicle kinematic and dynamic models. This design allowed the team to use their experience and expertise to efficiently complete the project. Along with that, the combinations of different parts were analyzed to ensure that the specifications of the different parts met the constraints and requirements of the project.

C. *Design Overview*

1) System Block Diagram (Black Box)

A system block diagram or black box can be seen in Fig. A-1. The inputs to the black box are a user input of an unspecified quantity as well as power. The user input indicates that there is user control of the system; however, the exact amount of control the user has can vary. The second input, power, is the electricity from an electrical outlet or battery that provides energy for the system to operate. There are two different output lines coming from the black box. One output line is a set of vehicle attributes. This output consists of position *X*, position *Y*, velocity, acceleration, and orientation of the vehicle. These five outputs are the attributes of the theoretical vehicle that are measured in testing. The other output is the physical rotational motion of the motors.

2) Subsystem Block Diagram (Glass Box)

The glass box of the system can be seen in Fig. 1 and contains three subsystems: the Simulink model, the experimental platform, and the GUI. The Simulink model contains the theoretical vehicle and the differential drive system. The experimental platform is a physical mock-up that will be used to implement the models and control algorithm developed in the Simulink model. The platform consists of a variety of components such as: two 24 volt Pittman Motor GM9236S015-R1 DC motors, two DC generators that are the same model number as the DC motors, a dual H-bridge/pulse-width modulation (PWM) interface, a microcontroller that contains the control algorithm and communication software, and two rotary encoders. The GUI subsystem is the connection between the two other subsystems; it takes user input from the user of the workstation and communicates it to both the Simulink subsystem and the experimental platform. The GUI sends velocity and disturbance commands to both the Simulink models and the experimental platform. In return, the GUI receives motor velocity data from the experimental platform and can use that to calculate various parameters of the theoretical vehicle. The Simulink model returns those same parameters; however, it is able to calculate them prior to sending them. Finally, the GUI takes the information that it receives from the two other subsystems and displays it to the user.

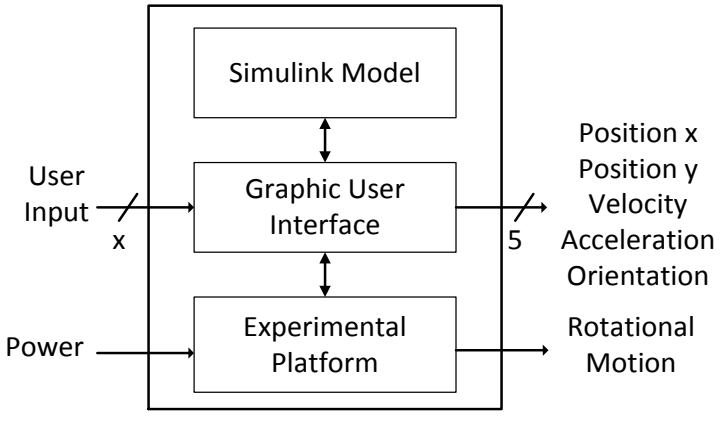


Fig. 1. Glass box of the entire system

The initial glass box in Fig. 1 can be further subdivided into two other glass boxes based on the Simulink model and the experimental platform. The glass box in Fig. B-1 is of the Simulink model. The different

blocks in Fig. B-1 were all modeled in Simulink and the only input into the system is the user input. This user input goes into the differential motor model and the generator set/dynamic model, which together is called the motor coupling. The user input causes left and right rotational velocity to be produced from the motor coupling. The rotational velocity is sensed by the rotary encoder model, which converts the rotational velocity into pulses per rotation. The measurement of pulses per rotation is an input into the controller block, which will actually hold the control algorithm that will regulate the differential drive system model. The output of the controller is a voltage or PWM command, which will feed into the H-Bridge model, which will in turn control the differential motor model through a voltage input. The rotational velocity is also fed into the vehicle kinematic model which converts the rotational velocity into translational velocity, position *X*, position *Y*, acceleration, and orientation of the vehicle.

Figure B-2 shows the glass box of the experimental platform. The inputs into the system are user input and power. The microcontroller receives power from the power input and user input from an RS-232 serial communication port. The microcontroller communicates with the H-Bridge/PWM interface and the generator set/current source. The microcontroller outputs the motor PWM which goes to the H-Bridge interface. The microcontroller also outputs an inter-integrated circuit (I^2C) signal, which goes into an external digital to analog converter (DAC), which controls the generator set. The H-Bridge steps up the voltage from the microcontroller to a voltage range that can control the motor over a wide velocity range. The motor platform is coupled to the generator set/current source. The generator set/current source introduces a torque load to the motors to simulate different driving conditions for the motors. The motor platform and the generator set/current source together are called the motor/generator coupling. The generator set causes a change in the motor rotational velocity, and that changed rotational velocity is sensed by the rotary encoder, which translates the rotational velocity into pulses. The output of the rotary encoder feeds back into the microcontroller, which is used in the control algorithm to adjust the velocity of the motor.

3) Microcontroller Software State Diagram

A high-level state diagram of the microcontroller software can be seen in Fig. 2. The software consists of four states: *initialization*, the *hold state*, the *interrupt state*, and the *communication state*. Initialization prepares the microcontroller for serial communication with the GUI, sets up the motor PWM, and sets up the 1 millisecond interrupt. After moving through the *initialization state* and into the *hold state* the software will wait for a 1 millisecond interrupt flag. Upon the flag being set the software will move to the interrupt state and stay there until the interrupt software is complete. The interrupt software consists of the model-based PID controller, feed-forward controller, anti-windup software, dynamic model, and I^2C communication software. Every one hundred and fifty interrupts the communication flag will be set and the software will move from the hold state to the *communication state*. The *communication state* is where the serial communication with the GUI takes place. The *communication state* can be interrupted by the interrupt state; furthermore, once the interrupt software is complete the microcontroller will return to the *communication state*.

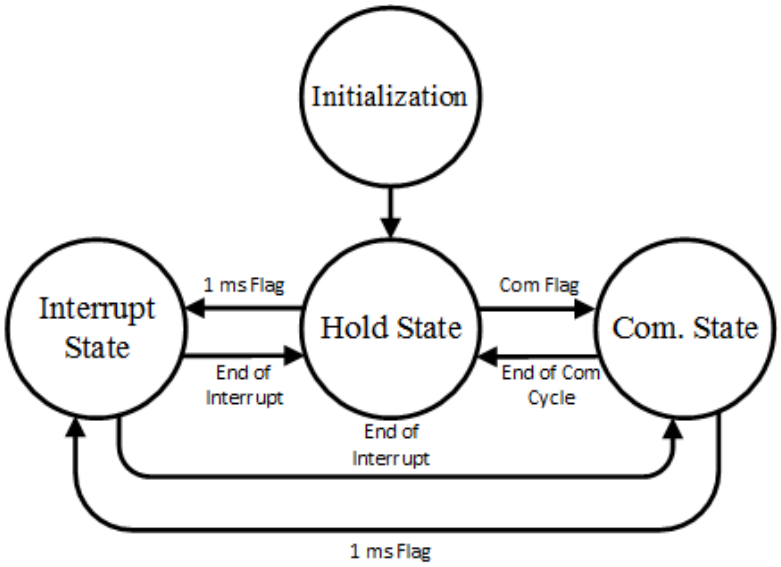


Fig. 2. State diagram of the microcontroller

4) Division of Labor

Assigning tasks to certain members of a team was essential to meeting the final deadline and to produce a good final product. For this project there are four main areas of work. Alexander Schmidt worked with cogging torque and modeling the physical motors. Benjamin Roos was responsible for the generator load that will be applied to the experimental system. Kevin Block handled the communication between the Graphical User Interface (GUI) and the experimental platform. Timothy De Pasion was responsible for the vehicle design and modeling along with rotary encoder resolution methods and models. All team members were involved with creating Simulink models of the different parts of the experimental system. The controller development for the project was completed by all four members of the team, while Kevin Block programmed the control algorithm in the microcontroller. A high level table of the division of labor can be seen in Table G-I. A detailed task list of the project engineer assignment can be found in Table G-II and Table G-III.

5) Experimental Platform

The experimental platform consists of two motors each coupled to an additional motor acting as a generator. These motors simulate the vehicle's differential drive, and the generators produce the opposition torque required to mimic the combination of the vehicle's dynamic model and terrain disturbances. Induced voltage from rotation of the generators is used to power an impedance load, generating current and therefore opposition torque. This opposition torque is a function of the vehicle dynamic model, the terrain material, and inclination angle, so the opposition current is a function of both motor speed and an added active load. This active load current source comprises an op-amp voltage follower controlling a bipolar junction transistor, offering precise current control. A schematic of the system can be seen in Fig. 3. Resistance *Rs* models the 1 Ω sensing resistor to transform input voltage commands into generator current load. Voltage source *Vg* and impedances *sLa* and *Ra* model the induced voltage and winding impedance of the generator. A compensator circuit was added to the design to control the inherent instability of the circuit caused by the winding inductance. This design process can be seen in detail in Appendix L.

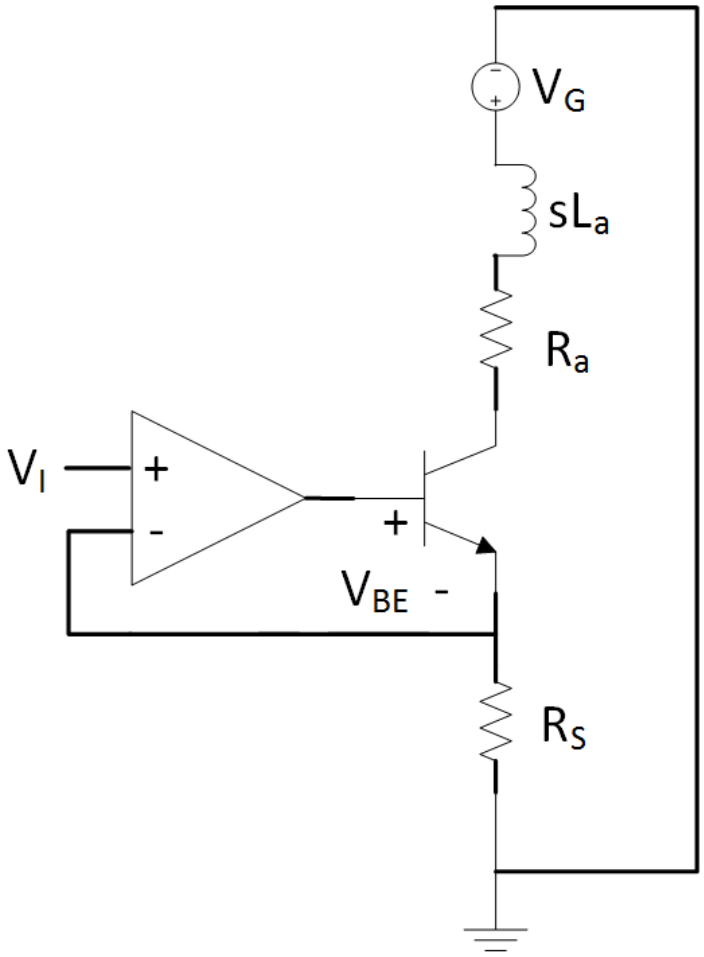


Fig. 3. Op-Amp voltage follower controlled transistor for precise generator torque control

The experimental platform integrates the motor and generator sets to the GUI through an ATmega128 microcontroller. The microcontroller executes the vehicle velocity control algorithm and handles all communication with the GUI via RS-232 communication. Because of inertia differences between the generator and theoretical vehicle, the experimental platform torque loads require correction to match the acceleration plant response of the experimental platform to the Simulink system. The torque correction software runs in tandem with the controller software on the microcontroller. Detailed design of the torque correction and acceleration matching system can be seen in Appendix M.

6) Simulink System

The Simulink system models all of the experimental platform hardware, the vehicle's motors, and the vehicle kinematic and dynamic subsystems. Vehicle modeling research was required to accurately determine the number of forces to be represented in the dynamic model, including gravity and friction. Nonlinear motor characteristics such as friction forces and cogging torque were also modeled. Cogging torque was modeled based on physical motor measurements, and research on best practices for modeling and measurement was completed in the fall of 2015. A project constraint was to only use standard Simulink library blocks to make the final product accessible to a wider range of interested researchers or academic institutions. A detailed design of the Simulink system can be seen in Appendix H.

An important detail of the Simulink system is that it is very complex, it contains over eighty different subsystems and is at some points is eight subsystems deep. The complexity of the system caused one major problem. The final integration of all of the individual Simulink models revealed that the simulation time for the complete Simulink model was well over 600 seconds to simulate a one second test. The fix for this problem was to set sample times for the various blocks inside of the Simulink model. This decreased the simulation time considerably to around sixty seconds to simulate a one second test. This was better than the previous result, but still too slow. The next step was to convert the continuous Simulink models to their discrete counterparts. This was the final step and decreased the simulation time to one second to simulate one second in the Simulink model.

7) Graphical User Interface

A GUI was used to integrate the Simulink models and the experimental platform. The GUI was developed in MATLAB to allow it to communicate natively with the Simulink models. In addition, MATLAB has integrated tools for communicating via serial; therefore, the experimental platform is able to communicate directly to the GUI. Creating the GUI via Java or Excel was briefly considered an option; however, the team had no experience making GUIs in Java and neither Java nor Excel has integrated tools for communicating with either Simulink or via RS-232. A detailed design of the GUI can be seen in Appendix I.

8) Control System

The control system was developed as a proof of concept of the operation of the workstation as a whole. Both the Simulink model and experimental platform use the same motor velocity controller which uses the rotary encoders embedded in the motors as feedback sensors. The control system consists mainly of a

vehicle plant model-based proportional-integral (PI) feedback controller. It also contains a model-based feed forward controller, an anti-windup system to avoid unnecessary overshoot, and an open loop rate-limiting system. The maximum acceleration in the system is limited to 350 RPM/s. This value was chosen as it corresponds to the maximum allowable H-bridge current according to Simulink simulations. The controller design process can be seen in detail in Appendix Q.

D. *Economic Analysis*

The cost of the project can be seen in Table E-I. The total cost of the project is $5,987.00. The most expensive items needed for the project were the software licenses required for MATLAB/Simulink, which cost $1,000 each, and the motors, which cost $283 each. Four licenses for MATLAB/Simulink and five motors were required. Those two items account for the majority of the cost that was associated with the project. One assumption that was made during the project formulation was that Bradley University will have most of the items needed for the project in stock ($5,887.00 worth of items) so the total cost of the project was only $100.00 ($5,987.00-$5,887.00) for the cost of miscellaneous items, including current source components and heat sinks. The project was funded by the Electrical and Computer Engineering Department at Bradley University.

III. DESIGN TESTING AND VALIDATION

The Simulink and experimental platform were tested in a variety of different ways. There were tests that compared the error between parts of the experimental platform and their corresponding Simulink models. There were also tests that existed in only Simulink and in only the experimental platform. There were some constants that existed in all of the different tests. The most important constant was that the range that the entire system operated over was 20 RPM to 400 RPM. A detailed testing plan for each of the functions can be seen in Appendix D.

A. *Nonfunctional Requirement Testing*

There were three nonfunctional requirements for the system. They can be seen in Table III and are measured by metrics, which can be seen in Appendix C. These nonfunctional requirements were verified by the team as well as two additional students in the electrical engineering department at Bradley University.

1) The experimental platform and Simulink model should be reliable

This nonfunctional requirement meant that the workstation should perform its designed function for the entire range of possible user input. The requirement was tested using a variety of premade tests that covered a large range of possible inputs into the system. Further testing was performed by modifying the vehicle dynamic model friction coefficient, inclination angle, and vehicle payload and then running step, ramp, or parabolic commands of varying magnitude. Based on the testing performed the experimental

platform and Simulink model were rated as *Reliable*. A rating of *Very Reliable* was not given because it was not possible to test every possible user input combination for the workstation.

2) The velocity command should be easy to issue to the Simulink model and experimental platform

This nonfunctional requirement meant that the user of the workstation would be able to easily issue a velocity command to both the Simulink model and the experimental platform. The requirement was tested by having a pair of students in the electrical engineering department separately test the workstation by issuing velocity commands to both subsystems. Based on the students' experience the GUI was modified to make the commands easier to issue. The feedback of the students' led to a rating of *Very Easy to Issue* to be given to this nonfunctional requirement.

3) The load of the Simulink model and experimental platform should be easy to manipulate

This nonfunctional requirement meant that the user of the workstation would be able to easily manipulate the load on both the Simulink model and the experimental platform. The requirement was tested by having a pair of students in the electrical engineering department separately test the workstation by manipulating the load on both subsystems. Based on the students' experience the GUI was modified to make the loads easier to manipulate. The feedback of the students' led to a rating of *Very Easy to Manipulate* to be given to this nonfunctional requirement.

B. *Functional Requirement Testing*

There were eleven functional requirements for the system, which can be seen in Table III. The functional requirements are measured with specifications that can be seen in Table III, while the detailed testing plans can be seen in Appendix D.

1) The Simulink motor model shall accurately model the physical motors

The experimental motors were modeled in Simulink through experimental measurements of a motor. The experimental motors and Simulink model motors were compared to their respective transient responses. A 20% range was allocated for error between the two systems. Unfortunately, the specification was not met due to several factors. One of the main causes of error was due to the fact the measurement method used a derivative calculation for output velocity on the experimental motors. Another source of error was experimental output measurements of the motor had to be filtered due to the noise in the measurements. Filtering those outputs reduced the transient response effects that were being measured. Refer to Fig. R-1 to view the filtered transient responses between the two systems at 12 volts. Refer to Table R-I to view the percent errors between the two systems.

2) The rotary encoder shall be accurately modeled

The rotary encoder model shown in Fig. R-2 was modeled in Simulink and the output frequency was compared between the Simulink model and the physical rotary encoder attached to the motor. The result of the test can be seen in Fig. R-3. The large error at the beginning of the figure is at the low end of our velocity range. The Simulink rotary encoder had an average error of 3.47%, which was within the 20% acceptable error range, so this specification was met and the functional requirement was satisfied.

3) Cogging Torque shall be accurately modeled

Cogging torque was modeled within the Simulink motor model. The expectations for cogging torque were to have no more than a 50% peak to peak variation between the experimental motor and the Simulink motor mode. The 50% specification was met with an average error of 14%. Refer to Table R-II to view peak to peak values for cogging torque.

4) Pulse-Width Modulation shall be accurately modeled

The pulse-width modulation (PWM) was modeled in Simulink and can be seen in Fig. R-4. The duty cycle output of the PWM was compared against the duty cycle output of the microcontroller from 0% to 100% duty cycle in 4% steps. The results can be seen in Fig. R-5. The error in percent for each duty cycle tested is 0.24%, which is below 20% so the PWM meets the specification and the functional requirement is met.

5) Simulink H-Bridge model shall accurately model the physical H-Bridge

The Simulink H-Bridge model can be seen in Fig. R-6. The voltage output of the Simulink H-Bridge was compared against the voltage output of the physical H-Bridge over a 0 to 24 volt range in 0.5 volt steps. The results can be seen in Fig. R-7. The percent error over the 0 to 24 volt range is 10.04% and below the 20% maximum, which means that the H-Bridge model met the specification and the functional requirement is satisfied.

6) The DC generator loads shall be designed to mimic the prototype vehicle

The torque correction and acceleration matching system to match the experimental platform plant to the Simulink vehicle plant was evaluated by testing open loop response of four parameters: settling time, overshoot, steady-state error, and absolute error over the Simulink response settling time. These parameters in both systems were compared against each other for a variety of open loop voltage and torque disturbance inputs as seen in the generator load testing plan of Appendix D. The specification states the experimental platform shall mimic the Simulink model within 50% of these parameters. The error plots for the four different parameters can be seen in Fig. R-8 to R-11.

The overall average parameter test measurements are as follows:

- Average Settling Time Error = 70.4%
- Average Overshoot Error = Undefined

- Average Steady-State Error = 24.6%
- Average Absolute Error = 34.3%

The average overshoot error is undefined because even with the inclusion of dynamic forces, the Simulink vehicle plant largely appears to be first-order. Thus, many Simulink open loop responses have zero overshoot. The experimental platform failed to meet the 50% error specification for average settling time and overshoot because of the dominating first-order characteristic of the vehicle plant, while the experimental platform velocity outputs were noisy due to the velocity derivative calculation. It did meet specification, however, for average steady-state error and absolute error.

A general trend can be observed for each test parameter. As the steady-state current load increases, the percent error for a given parameter increases. Increases in steady-state current loads decrease the amount of net torque available to be reduced during acceleration in the motors. Thus, the torque correction and acceleration matching system better matches the theoretical vehicle plant at lower current load conditions.

7) The drive control system shall minimize the effect of external torque disturbances

After reaching a given steady-state speed, the motors were given a torque disturbance of various magnitudes as can be seen in the torque disturbance testing plan of Appendix D. The disturbance specification for the control system states the maximum instantaneous deviation from the steady-state command must be limited to a 40% error. Refer to Fig. R-12 to see the disturbance overshoot response and Fig. R-13 to see the disturbance error response. The maximum disturbance overshoot occurred at 20 RPM as 36% for the experimental platform and 38% for the Simulink model. This specification has been met in both systems.

8) The drive control system shall reduce vehicle tracking errors for step and ramp commands

Motors in both systems were given step commands of various magnitudes to measure the controller's tracking ability according to the step tracking error testing plan in Appendix D. The step tracking specification states the response error shall be less than 20% over 4 seconds. The step error plots can be seen in Fig. R-14 and Fig. R-15. A maximum error of 13% occurs with a 20 RPM command in the Simulink model, and a maximum error of 22% occurs with a 20 RPM command in the experimental platform. The Simulink model meets this specification, but the experimental platform barely misses it.

Both systems must also match ramp commands with a specification of 20% error over 4 seconds. Both systems were tested with various magnitudes of ramp inputs, or various accelerations, as described in the ramp error testing plan of Appendix D. The ramp error plots can be seen in Fig. R-16 and Fig. R-17. The experimental platform meets the ramp tracking specification with a maximum error of 19% at 20 RPM/s. The Simulink model, however, has a maximum error of about 34% at 400 RPM/s. This result shows an inherent limitation in the system with no braking or reversal control. In this case, the Simulink model more accurately models the theoretical vehicle, as the limited deceleration is caused by the large vehicle inertia. This deceleration matching is not available in the experimental platform because the generators are designed to be unidirectional.

9) The drive control system shall reduce vehicle tracking errors for parabolic commands

Parabolic inputs of varying magnitudes were input as commands into both the Simulink model and experimental platform to test parabolic tracking as seen in the parabolic error testing plan of Appendix D. The controller must match the command with less than or equal to 40% error over 4 seconds. The error plots for the parabolic commands can be seen in Fig. R-18 and Fig. R-19. Both systems have less than 40% parabolic tracking error over the entire parabolic input magnitude test range. The controller meets the parabolic tracking specification for both systems.

10) The drive control system shall reduce the effects of motor mismatch

To test the controller's rejection of motor mismatch, motors were varied to the worst case mismatch of motor parameters in the Simulink model. Percent difference in motor response for a given range of step inputs was measured as described in the motor mismatch testing plan of Appendix D. The controller shall regulate mismatched motors within 15% of each other over 4 seconds. The error plot can be seen in Fig. R-20. The maximum speed variance between the motors is about 4% with a 20 RPM command. The controller meets the motor mismatch specification.

11) The graphical user interface shall send and receive commands

The GUI functional requirement specified that the GUI must be able to send and receive commands. This functional requirement was verified by checking whether test signals that were sent by the GUI were received by the Simulink system and the experimental platform. The GUI was also tested by checking whether commands sent by the Simulink system and the experimental platform were received by the GUI. Since both of these tests returned results that showed the GUI was both sending and receiving commands, this functional requirement was also met.

IV. SUMMARY/CONCLUSIONS

In summation, a control workstation was developed to provide the faculty and staff at Bradley University with a tool for researching and implementing control algorithms. The workstation is composed of three subsystems: a Simulink model, an experimental platform, and a GUI for integration. The workstation will serve as a platform for future research as well as an educational aid for future Electrical Engineering students at Bradley University. The inclusion of a vehicle dynamic model as well as a model of cogging torque is the significant advantage to using this platform over other similar products. In addition, the GUI was designed with a user focus to further facilitate the workstation's educational function. Some of the key topics covered by this project were: modeling, controller development, system integration, team coordination, and circuit design. The team members benefited significantly from the experience gained in these topics.

V. REFERENCES

[1] M. Niţulescu, “Theoretical Aspects in Wheeled Mobile Robot Control”, IEEE International Conference on Automation, Quality and Testing, Robotics, May 2008.

[2] Edouard Ivanjko, Toni Petrinic, Ivan Petrovic, “Modeling of Mobile Robot Dynamics”, University of Zagreb, Faculty of Electrical Engineering and Computing 10000 Zagreb, Unska 3, Croatia, http://www.researchgate.net/profile/Edouard_Ivanjko/publication/228561343_ Modelling_ of_Mobile_Robot_Dynamics/links/004635256b78692e72000000.pdf

[3] J. C˘erkala, A. Jadlovská, “Mobile Robot Dynamics with Friction in Simulink”, Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovak Republic, http://www2.humusoft.cz/www/papers/ tcb2014/016_cerkala.pdf

[4] Gary Dempsey, “ECE 441 Control Theory I Workbook”, Bradley University, Electrical & Computer Engineering Department, August 2014.

[5] Gary Dempsey, “ECE 442 Control Theory II Workbook”, Bradley University, Electrical & Computer Engineering Department, January 2015.

[6] Michael Barngrover, “Investigation of Precision Modeling and Control for Plants with High Degrees of Friction and Load Variation”, MSEE Thesis, Electrical and Computer Engineering Department, Bradley University, August 2007.

[7] Rukmani Ayyempet Mohanganesh, "Control Methods for Precise Positioning of a 2-DOF Robot Arm System," MSEE Design Project, Electrical and Computer Engineering Department, Bradley University, December 2011.

[8] Simon Benik and Adam Olson, "DC Motor-Clutch-Generator Control Workstation," Bradley University, Electrical & Computer Engineering Senior Project, 2006-2007.

[9] Andrew Fouts and Kurtis Liggett, “An Observer-based Engine/Cooling Control System”, Senior Capstone Project, Electrical and Computer Engineering Department, Bradley University, May 2011.

[10] C. Edwards and E. Smith, “A Design of a Simulink-Based 2-DOF Robot Arm Control Workstation”, Bradley University, Electrical & Computer Engineering Senior Project, May 2007, http://cegt201.bradley.edu/projects/proj2007/twodofra/

[11] Manfred Meissner and Christopher Spevacek, "Implementation of Conventional and Neural Controllers Using Position and Velocity Feedback," Bradley University, Electrical & Computer Engineering Senior Project, 1999-2000.

[12] Gary Dempsey, "Using Conventional Controllers with the CMAC Neural Network," Proceedings of the Artificial Neural Networks in Engineering (ANNIE 99 Conference), St. Louis, Mo., November 1999.

[13] Gary Dempsey, Manfred Meissner, and Christopher Spevacek, “Using a CMAC Neural Network in Noisy Environments”, Proceedings of the Artificial Neural Networks in Engineering (ANNIE) 2003 Conference, St. Louis, Mo., November 2003.

[14] WeiWu, “DC Motor Parameter Identification Using Speed Step Responses”, Hindawi Publishing Corporation, Modelling and Simulation in Engineering, Volume 2012, Article ID 189757, September 2012.

[15] Roberto Petrella, Marco Tursini, Luca Peretti, Mauro Zigliotto, “Speed Measurement Algorithms for Low-Resolution Incremental Encoder Equipped Drives: a Comparative Analysis”, International Aegean Conference on Electrical Machines and Power Electronics, September, 2007.

[16] SIMULINK, Simulation and Model-Based Design, http://www.mathworks.com/ products/simulink/

[17] C. Ta and Y. Hori, “Convergence Improvement of Efficiency-Optimization Control of Induction Motor Drives,” *IEEE Transactions on Industry Applications*, vol. 37, no. 6, pp 1746 – 1753, Dec, 2001.

[18] Z. Zhu, “A Simple Method for Measuring Cogging Torque in Permanent Magnet Machines”. 2009.

VI. APPENDIX

A. *System Black Box*

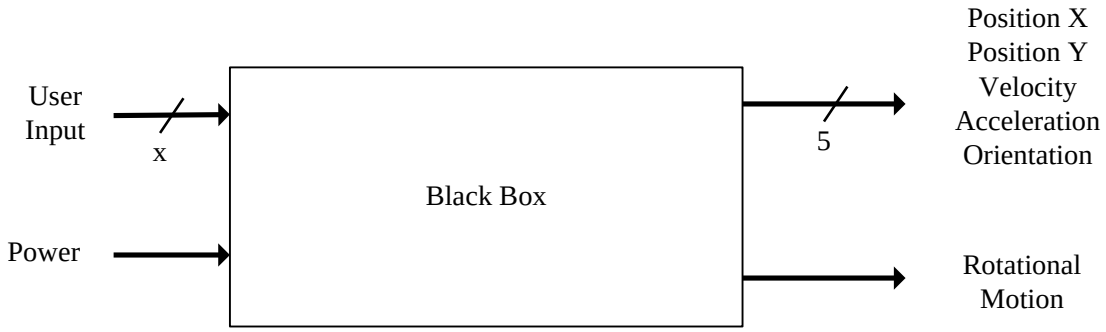


Fig. A-1. Black box of the entire project

B. *Detailed Subsystem Glass Boxes*

This section contains the glass boxes of the Simulink system and the experimental platform.

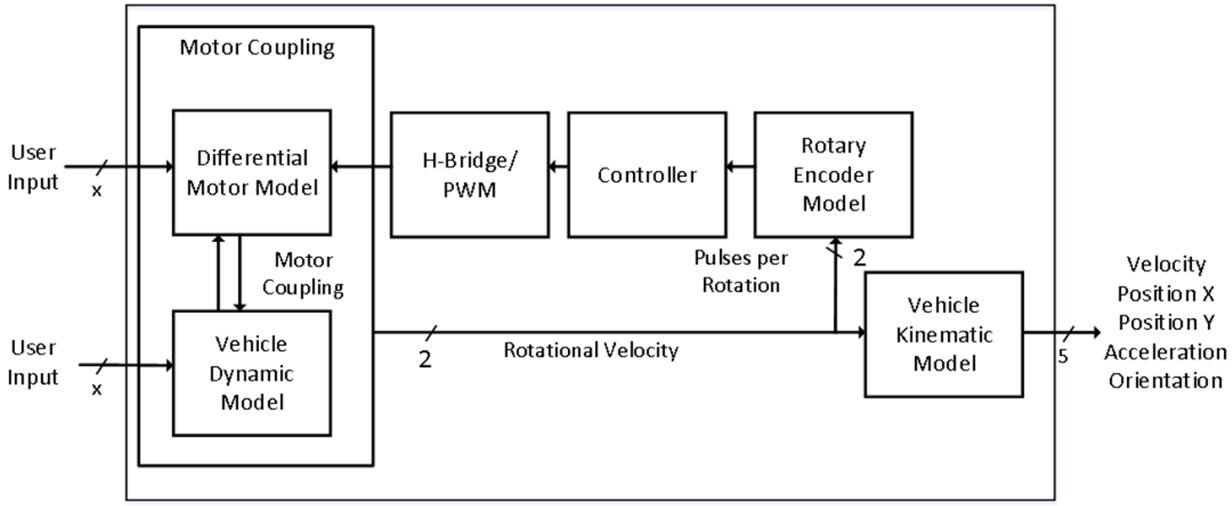


Fig. B-1. Glass box of the Simulink Subsystem

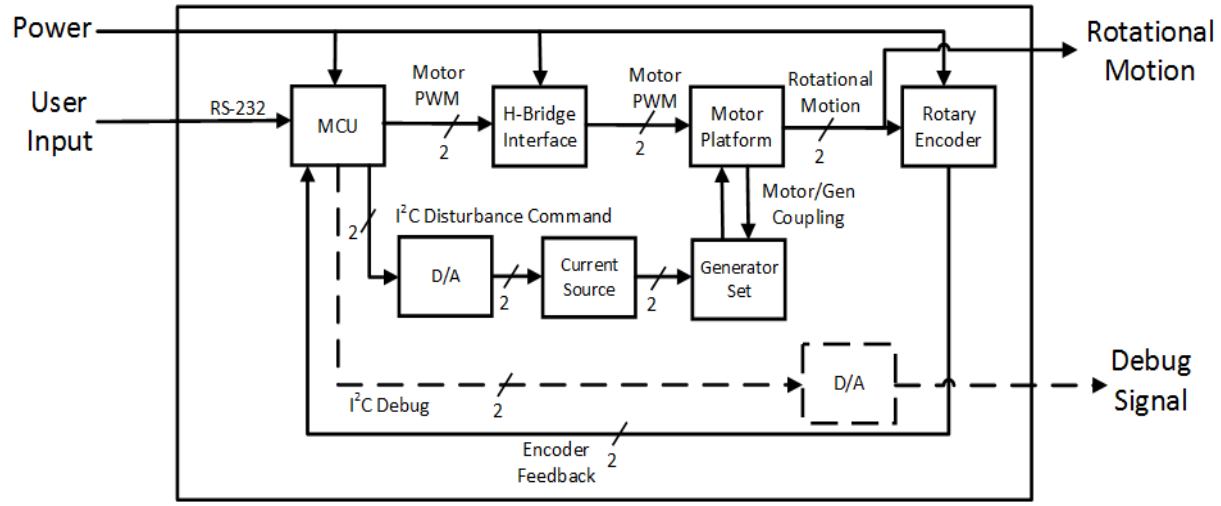


Fig. B-2. Glass box of the experimental platform

C. *Metrics*

Objective: The experimental platform and Simulink model should be reliable.
Metric:

- Very reliable 5 points
- Reliable 4 points
- Average Reliability 3 points
- Not reliable 2 points
- Very unreliable 1 point

Objective: The velocity command shall be easy to issue to the Simulink Model and the experimental platform.
Metric:

- Very easy to issue 5 points
- Easy to issue 4 points
- Average difficulty to issue 3 points
- Difficult to issue 2 points
- Very difficult to issue 1 point

Objective: The load of the Simulink model and the experimental platform should be easy to manipulate.
Metric:

- Very easy to manipulate 5 points
- Easy to manipulate 4 points
- Average difficulty to manipulate 3 points
- Difficult to manipulate 2 points
- Very difficult to manipulate 1 point

D. *Detailed Testing Plans for Functional Requirement*

The Simulink motor model shall accurately model the physical motors:
The detailed testing for the Simulink motor model will test the velocity output of the Simulink model against the open-loop physical motors when a range of step inputs is applied between 20 RPM and 400 RPM with a 20 RPM step size. Settling time, percent overshoot, and steady-state error will be recorded from the resulting waveforms of each velocity input for both systems. The average percent difference of these three measurements of the Simulink motor models across the given velocity range must be within 20% of the physical motor's measurements.

The rotary encoder shall be accurately modeled:
The detailed testing for the rotary encoder model will test the output frequency of the Simulink model against the output frequency of the actual rotary encoder that is on the motor. The test will occur over the range of 0 volts to 24 volts and have 0.5 volt steps. The average error will be computed. The error must be within ±20% for the system to meet the specification.

Cogging torque shall be accurately modeled:
The detailed testing for the cogging torque model will test the motor's cogging torque output against the Simulink model's output. The Simulink motor and the physical motor's current waveforms shall be changed into torque by using the K_t constant. Those two waveforms shall be correlated between each other. The correlation between the two waveforms cannot deviate more than ±50%. The base point for this percentage is based off of the max peak-to-peak values of torque ripple.

Pulse-width modulation shall be accurately modeled:
The detailed testing of the PWM model will consist of a comparison of the microcontroller PWM output to the PWM output of the model. The test will measure the average error in the percent duty cycle over the range of a 0% duty cycle to a 100% duty cycle in 4% steps. The error must be within ±20% for the system to meet the specification.

The Simulink H-bridge model shall accurately model the physical H-bridge:
The detailed testing of the Simulink H-Bridge model will test the output voltage of the H-Bridge model against the output voltage of the actual H-Bridge. The test will occur over the range of 0 volts to 24 volts and will occur in 0.5 volt steps. The average error of each point will be computed and it must be within ±20% for the system to meet the specification.

The DC generator loads shall be designed to mimic the prototype vehicle:
Using an open-loop motor, a steady-state rotational speed will be set with voltage increments from 4 volts to 16 volts with 4 volt steps in both systems. For a given voltage input, a torque disturbance will be input as a step command in the form of currents ranging between 0.3 amps to 1.5 amps in 0.3 amp increments. Settling time, percent overshoot, steady-state error, and absolute tracking error over the settling time of the Simulink model will be measured from the resulting motor speed waveform in both systems. The average absolute error of the experimental platform generator over the given voltage and current ranges shall be less than 50% as compared to the Simulink vehicle model for these four measurements.

The drive control system shall minimize the effect of external disturbances:
Both the closed-loop Simulink model and the experimental platform will be set at a steady-state rotational velocity ranging from 20 RPM to 400 RPM in 20 RPM steps. For a given speed, a torque disturbance will be input ranging between 0.3 amps to 1.5 amps in 0.3 amp steps. The average minimum instantaneous resulting rotational speed shall be within 40% of the set steady-state speed over the given speed and current ranges.

The drive control system shall reduce vehicle tracking errors for step and ramp commands:
Both the closed-loop Simulink model and experimental platform will be given step commands ranging from 20 RPM to 400 RPM in 20 RPM steps lasting 4 seconds, and ramp commands ranging from 20 RPM/s to 400 RPM/s in 20 RPM/s steps lasting 4 seconds. The ramp commands will be limited to a maximum of 400 RPM while accelerating and then immediately decelerate to a minimum of 20 RPM. This cycle will repeat until the test time reaches 4 seconds. The average maximum instantaneous error between the command and output should be less than or equal to 20% over the given speed range.

The drive control system shall reduce vehicle tracking errors for parabolic commands:
Both the closed-loop Simulink model and experimental platform will be given parabolic commands ranging from 20 RPM/s^2 to 400 RPM/s^2 in 20 RPM/s^2 steps lasting 4 seconds. The parabolic commands will be limited to a maximum of 400 RPM while accelerating and then immediately decelerate to a minimum of 20 RPM. This cycle will repeat until the test time reaches 4 seconds. The average error between the command and output should be less than or equal to 40% over the given speed range and time period.

The drive control system shall reduce the effects of motor mismatch:
The detailed testing for motor mismatch will test the velocity output of both Simulink model motors at worst matched condition, meaning motor friction parameters are varied to their maximums and minimums in each motor respectively according to datasheet values. No load will be applied to the motors and motor seed waveforms will be measure for open loop inputs of 2 to 24 volts with 2 volt increments. The RPM average deviation for all inputs cannot be more than ±15% compared to the same data taken with perfectly matched Simulink motor models.

The graphical user interface shall send and receive commands:
The detailed testing of the graphical user interface will consist of a test to see if the GUI can send and receive signals. Test signals will be sent by the GUI and then both the Simulink model and the experimental platform will be checked to see if the signals have been received. The Simulink model and experimental platform will then send a signal back to the GUI, which will then be checked to see if the GUI has received the signal.

E. *Budget*

This section of the Appendix contains the budget and cost of the individual parts that are needed for the project.

TABLE E-I. COST OF INDIVIDUAL COMPONENTS

Part	Price
H-Bridge (LMD 18200)	$ 88.00
Atmel Development Board	$ 340.00
Atmega128 Microcontroller	$ 44.00
24V Pittman Motors	$ 1,415.00
Atmel Studio 6	$ -
MATLAB/Simulink	$ 4,000.00
Miscellaneous	$ 100.00

F. *Detailed Gantt Chart*

This section of the appendix contains a detailed Gantt chart that shows the project schedule.

ACTIVITY	START	DURATION
Individual Component Research and Modeling	1	11
Project Proposal Presentation and Paper	1	5
Project Webpage	6	30
Progress Presentation 1	9	3
Experimental Platform Integration	12	4
Simulink Integration	13	4
Controller Development	16	4
Testing of the Subsystems	20	4
Progress Presentation 2	20	3
Integration of Simulink and Experimental Platform	24	2
Testing of the Integrated System	25	2
Project Demonstration	26	1
Final Report	27	6
Final Presentation	28	2
Poster Presentation	31	2

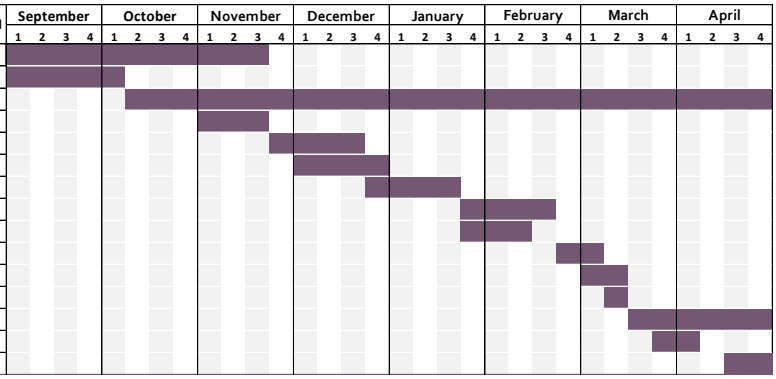


Fig. F-1. High level Gantt chart of the project

Task Name	Start Period	Duration
Team Coordination	1	2
Initial Simulink Modeling	1	3
Project Proposal Document	3	10
MATLAB Code Model based on initial Simulink Model	4	2
Cogging Torque Experimental and Research	4	7
Efficiency, Motor, Motor-Gen Load Curve	4	2
Team Coordination	5	1
Generator Simulink Model	5	3
Serial communication research and implementation	5	5
Physical Vehicle Prototype Design	5	3
Initial Controller Development	7	3
Rotary Encoder Resolution Algorithm	7	4
Rotary Encoder Model	7	4
Generator Experimental	8	2
Proposal Presentation	8	4
Team Coordination	9	1
Current Source Research and experimental	9	7
AtMega PWM Setup	10	2
Cogging Torque Model	11	7
Command Conditioning Simulink	11	1
H-Bridge Model	12	2
Team Coordination	13	1
Input Command	13	5
PWM Model	14	7
Experimental Thermal Measurements	15	2
Web Page Development/Updates	15	5
Team Coordination	17	1
PWM Frequency Optimized	17	3
Digital Filter	17	4
Disturbance Command	18	4
Kinematic Model	18	4
Progress Report Presentation 1	20	4
Thermal Simulink Model	20	4
Team Coordination	21	1
I^2C and new D/As	22	4

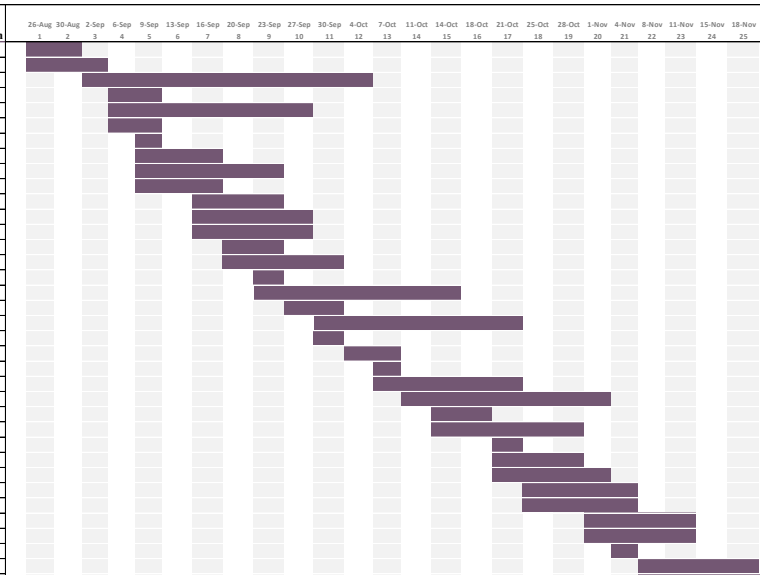


Fig. F-2. Part 1 of the detailed Gantt chart

Task Name	Start Period	Duration
Dynamic Model	22	6
Performance Review Document	23	1
Experimental Platform Integration	24	9
Team Coordination	25	1
Efficiency Model	27	4
Simulink Integration	28	8
Team Coordination	29	1
Controller Development	35	8
Team Coordination	37	1
Controller Model	39	8
Team Coordination	41	1
Controller Code	41	5
Terrain Testing	42	1
Simulink GUI Control	42	9
Team Coordination	45	1
Progress Report Presentation 2	46	4
Experimental Platform Testing	47	8
Simulink System Testing	47	8
Team Coordination	49	1

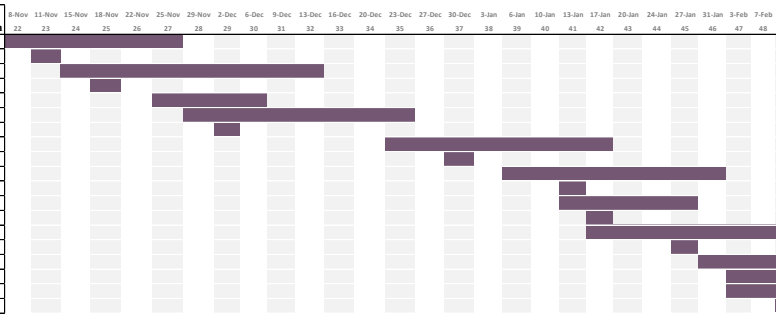


Fig. F-3. Part 2 of the detailed Gantt chart

Task Name	Start Period	Duration
Simulink GUI Control	42	9
Team Coordination	45	1
Progress Report Presentation 2	46	4
Experimental Platform Testing	47	8
Simulink System Testing	47	8
Team Coordination	49	1
Experimental Platform and Simulink Integration	52	5
Team Coordination	53	1
Experimental Platform and Simulink Testing	56	8
Team Coordination	57	1
Demo Prep	57	2
Final Report	60	5
Team Coordination	61	1
Final Presentation	62	4
Poster Presentation	64	3
Team Coordination	65	1

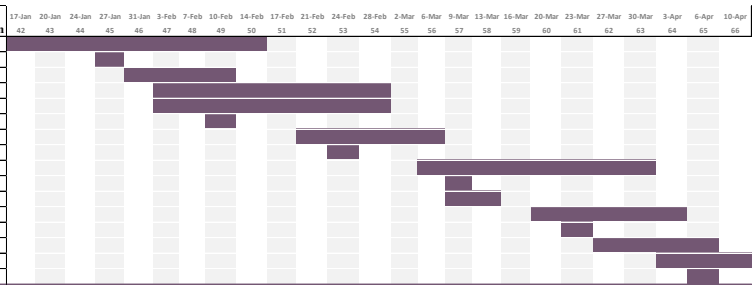


Fig. F-4. Part 3 of the detailed Gantt chart

G. *Detailed Division of Labor*

This section of the appendix contains a detailed division of labor that shows who will do every task that is listed in the Gantt chart. The team coordination events that are present in the Gantt chart are not present in the division of labor because they are meetings that take place and are necessary to account for in the Gantt chart, but are not necessary to put in the division of labor.

TABLE G-I. HIGH LEVEL DIVISION OF LABOR

Task Name	**Team Member Name**
Cogging Torque	Alexander Schmidt
Motor Models	Alexander Schmidt
Generator Load and Model	Benjamin Roos
Communication	Kevin Block
C-Code	Kevin Block
Vehicle Design and Modeling	Timothy De Pasion
Rotary encoder resolution	Timothy De Pasion
Controller development and model	All team members
Integration	All team members

TABLE G-II. PART 1 OF THE DETAILED DIVISION OF LABOR

Task name	Resource name
Initial Simulink Modeling	Alex, Ben, Kevin, Tim
Project Proposal Document	Alex, Ben, Kevin, Tim
Initial Controller Development	Alex, Ben, Kevin, Tim
Proposal Presentation	Alex, Ben, Kevin, Tim
Web Page Development/Updates	Alex, Ben, Kevin, Tim
Progress Report Presentation 1	Alex, Ben, Kevin, Tim
Performance Review Document	Alex, Ben, Kevin, Tim
Controller Development	Alex, Ben, Kevin, Tim
Controller Model	Alex, Ben, Kevin, Tim
Experimental Platform and Simulink Integration	Alex, Ben, Kevin, Tim
Experimental Platform and Simulink Testing	Alex, Ben, Kevin, Tim
Progress Report Presentation 2	Alex, Ben, Kevin, Tim
Demo Prep	Alex, Ben, Kevin, Tim
Final Report	Alex, Ben, Kevin, Tim
Final Presentation	Alex, Ben, Kevin, Tim
Poster Presentation	Alex, Ben, Kevin, Tim
Dynamic Model	Ben and Tim
Efficiency, Motor, Motor-Gen Load Curve	Alex and Ben
Experimental Thermal Measurements	Alex and Ben
Thermal Simulink Model	Alex and Ben
Experimental Platform Integration	Alex and Ben
Experimental Platform Testing	Alex and Ben
Simulink Integration	Kevin and Tim
Simulink System Testing	Kevin and Tim

TABLE G-III. PART 2 OF THE DETAILED DIVISION OF LABOR

Task name	Resource name
Cogging Torque Experimental and Research	Alex
Cogging Torque Model	Alex
PWM Frequency Optimized	Alex
Efficiency Model	Alex
Generator Simulink Model	Ben
Generator Experimental	Ben
Current Source Research and experimental	Ben
Digital Filter	Ben
Serial communication research and implementation	Kevin
AtMega PWM Setup	Kevin
Input Command	Kevin
Disturbance Command	Kevin
I^2C and new D/As	Kevin
Controller Code	Kevin
Simulink GUI Control	Kevin
MATLAB Code Model based on initial Simulink Model	Tim
Physical Vehicle Prototype Design	Tim
Rotary Encoder Resolution Algorithm	Tim
Rotary Encoder Model	Tim
Command Conditioning Simulink	Tim
H-Bridge Model	Tim
PWM Model	Tim
Kinematic Model	Tim
Terrain Testing	Tim

H. *Detailed Simulink Model*

The highest level part of the Simulink System can be seen in Fig. H-1

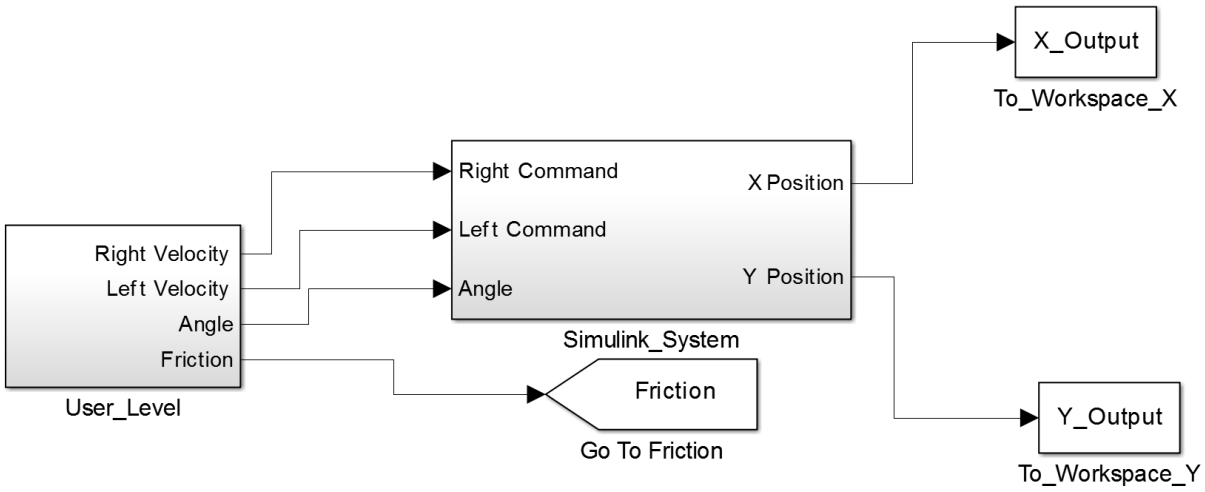


Fig. H-1. High Level Simulink System

It is separated into two subsystems; the User Level seen in Fig.H-2 and the Simulink system seen in Fig.H-3.

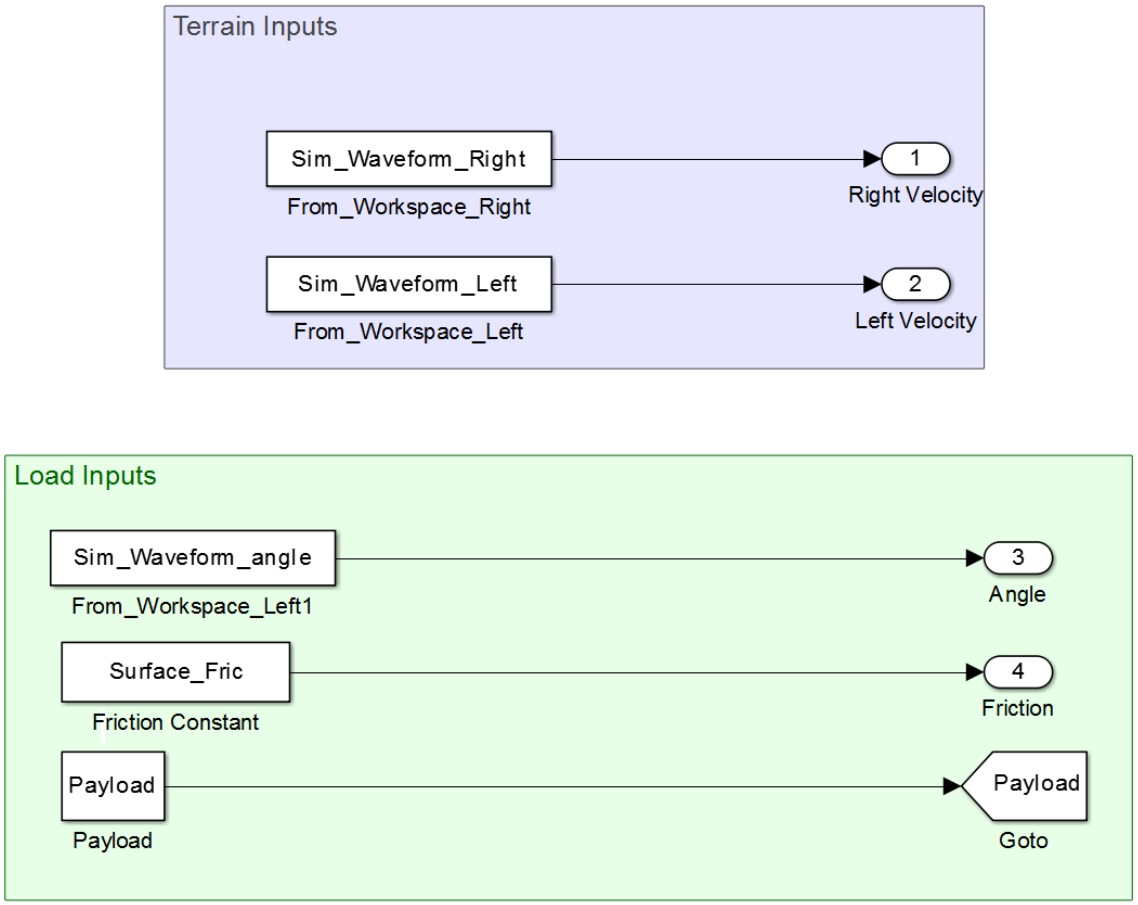


Fig. H-2. User Level of the Simulink System

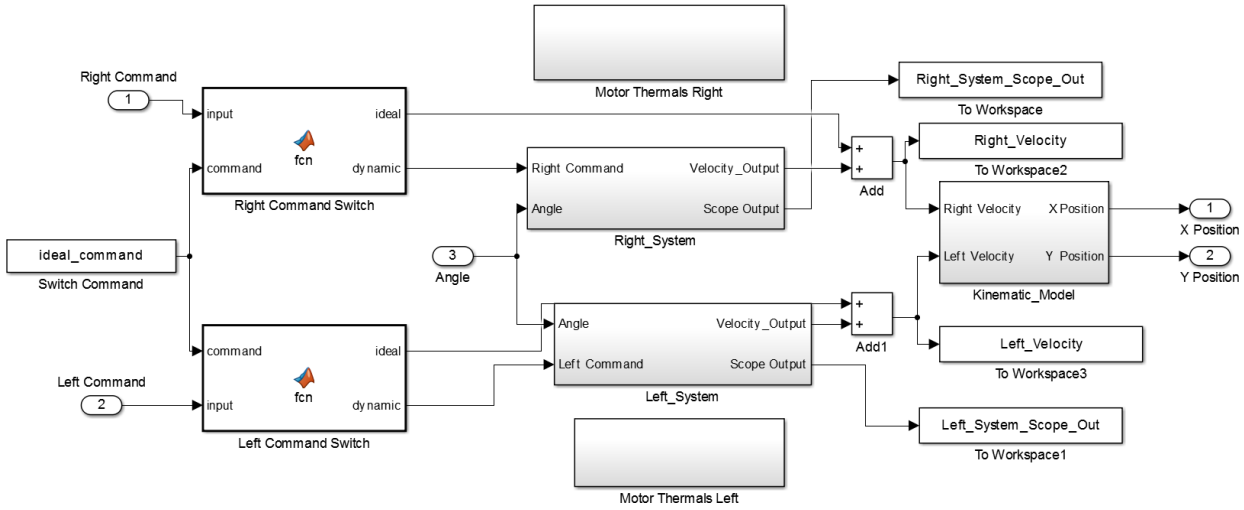


Fig. H-3. Simulink System

The Simulink system is separated into five subsystems: Motor Thermals Right, Motor Thermals Left, Right System, Left System, and the Kinematic Model. There is also a MATLAB function in this model.

The MATLAB function code can be seen in Fig. H-4.

```
function [ideal, dynamic] = fcn(input,command)
%#codegen
if(command == 1)
    ideal = input;
    dynamic = 0;
else
    dynamic = input;
    ideal = 0;
end
```

Fig. H-4. Ideal Switch MATLAB Function

The left and right thermal models are the same and the thermal subsystem can be seen in Fig. H-5.

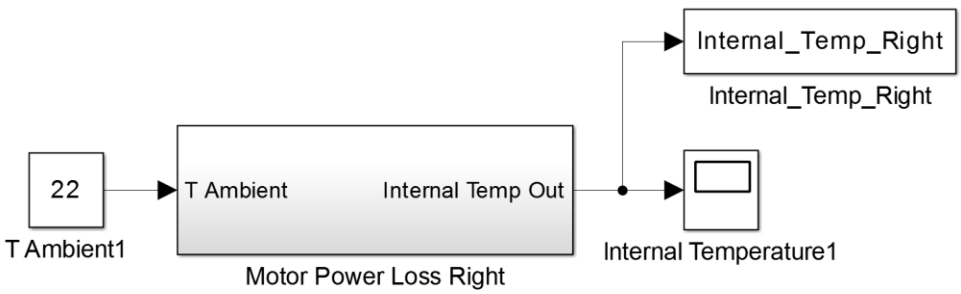


Fig. H-5. Thermal Subsystem

The thermal subsystem has one subsystem: the motor power loss subsystem, which can be seen in Fig. H-6.

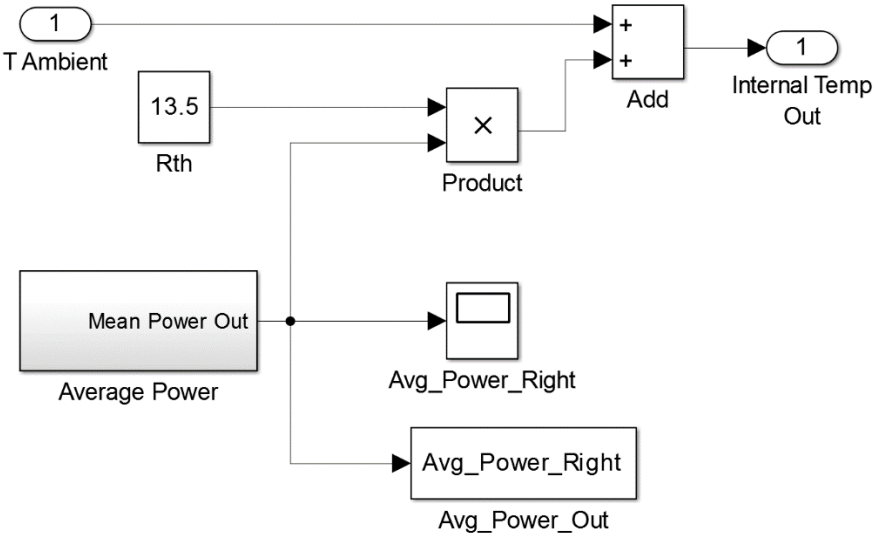


Fig. H-6. Motor Power Loss

Fig. H-6 has one subsystem, which is the average power subsystem, seen in Fig. H-7.

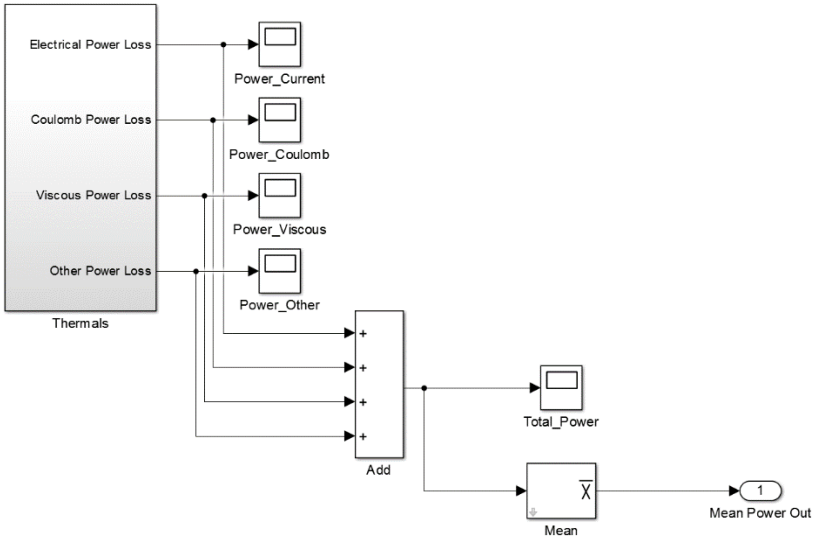


Fig. H-7. Average Power Subsystem

This subsystem has one subsystem, the thermal subsystem, which can be seen in Fig. H-8.

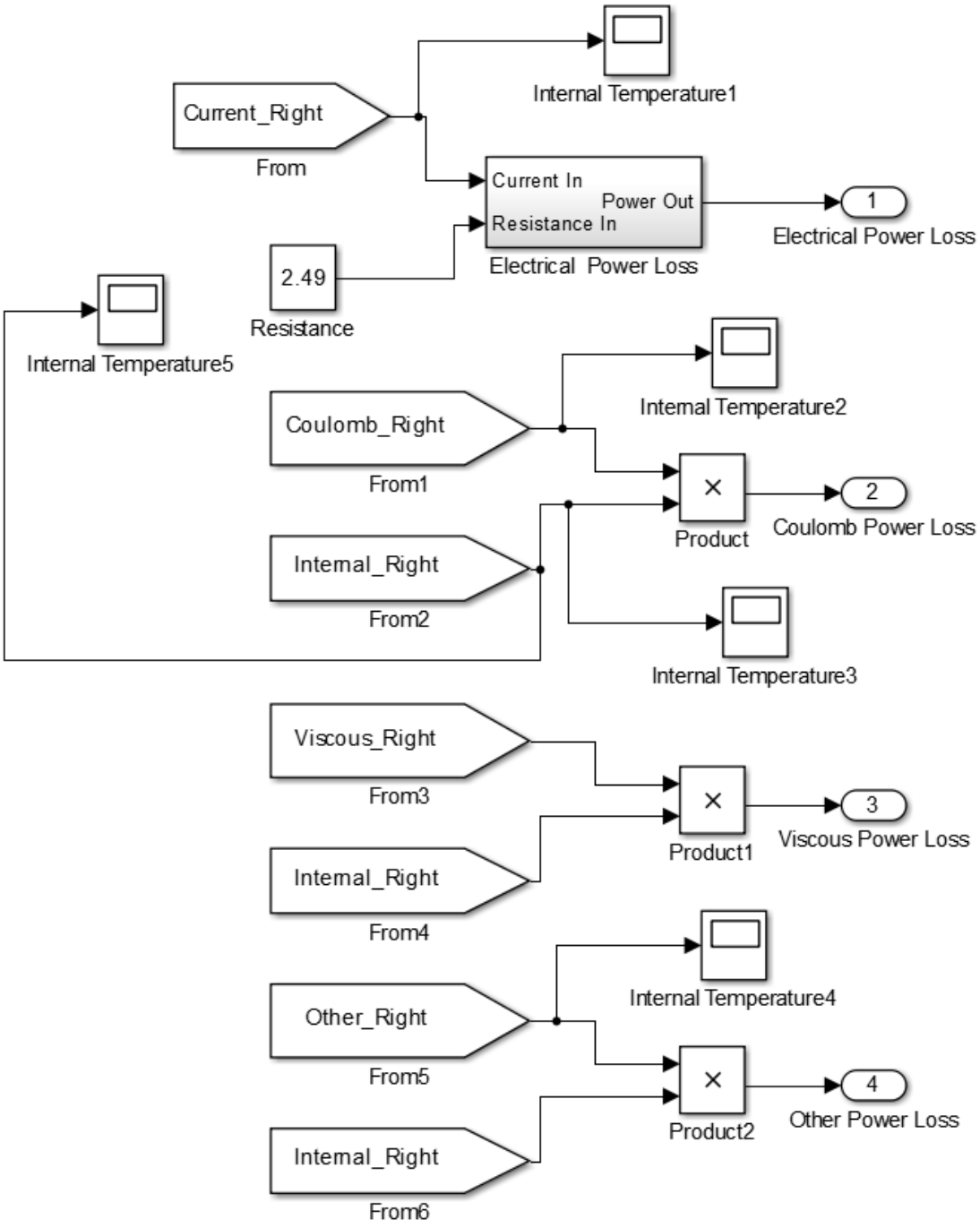


Fig. H-8. Thermal Subsystem

This subsystem also has one subsystem, the electrical power loss subsystem, seen in Fig. H-9.

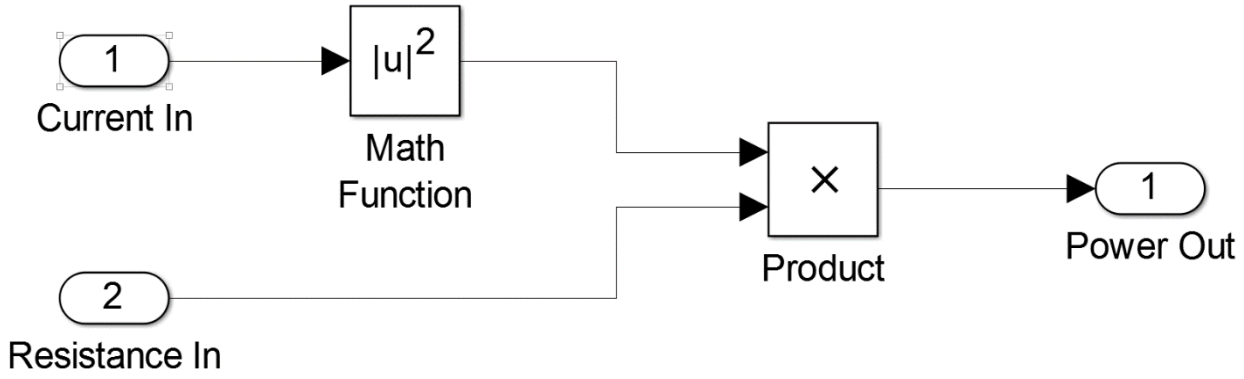


Fig. H-9. Electrical Power Loss Subsystem

The left and right systems are also the same. The model can be seen in Fig. H-10. There are five subsystems in the left and right system models: command conditioning, controller, designed models, average models, and the motor-dynamic coupling.

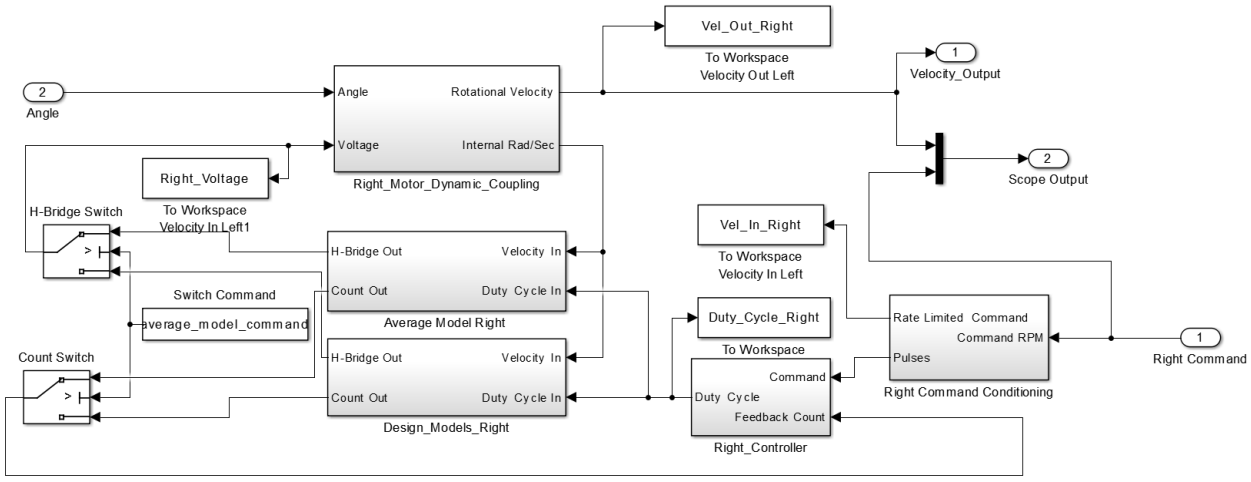


Fig. H-10. Motor-Dynamic Coupling Subsystem

The command conditioning model can be seen in Fig. H-11.

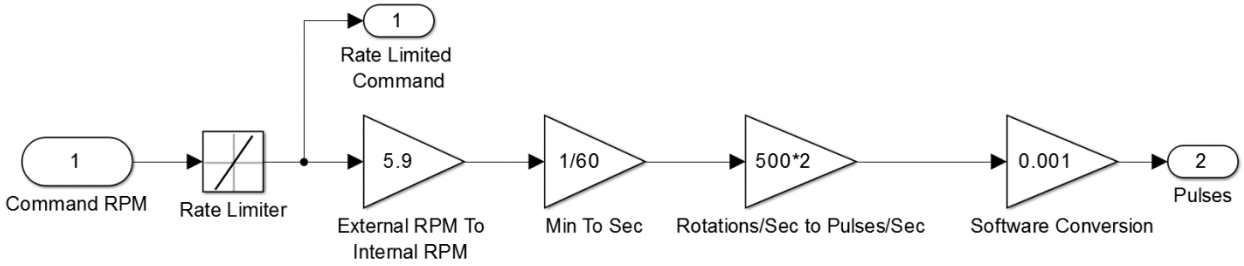


Fig. H-11. Command Conditioning Subsystem

The controller model can be seen in Fig. H-12.

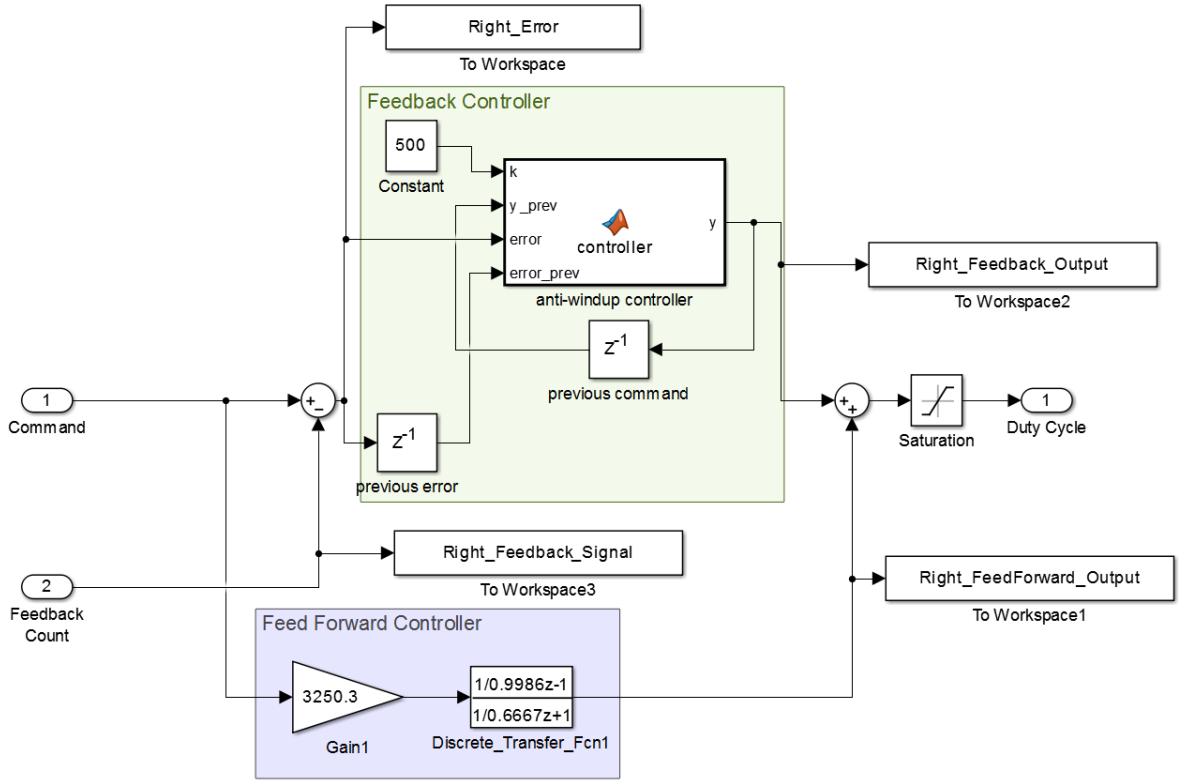


Fig. H-12. Controller Subsystem

The controller MATLAB code can be seen in Fig. H-13.

```
function y = controller(k,y_prev,error,error_prev)
%#codegen

if (y_prev >= 1023)
    if (error > 0)
        y = k*1.01*error;
    else
        y = k*(1.01*error - 0.9902*error_prev) + y_prev;
    end
elseif (y_prev <= 0)
    if (error > 0)
        y = k*(1.01*error - 0.9902*error_prev) + y_prev;
    else
        y = k*1.01*error;
    end
else
    y = k*(1.01*error - 0.9902*error_prev) + y_prev;
end
```

Fig. H-13. Controller Subsystem

The designed models block can be seen in Fig. H-14.

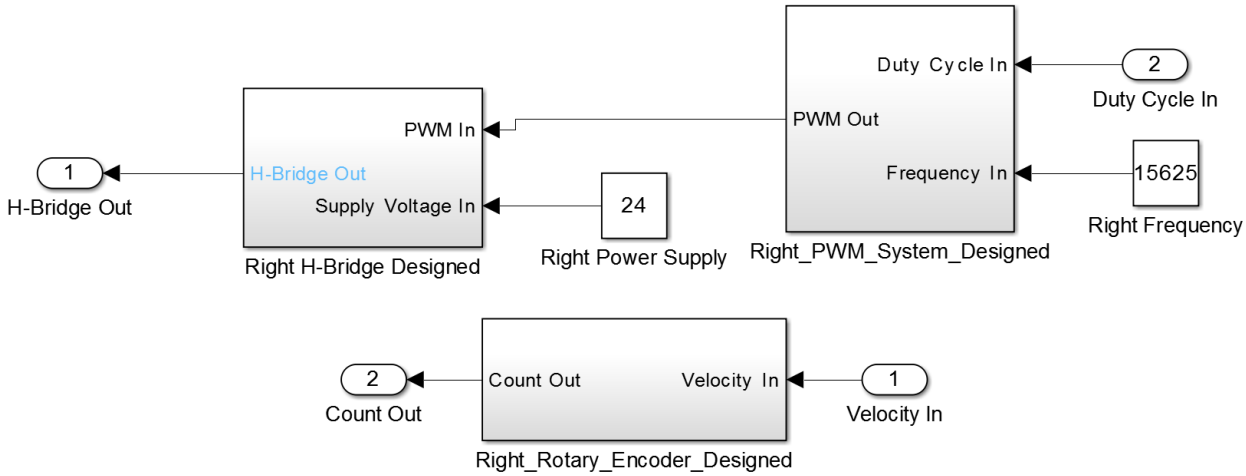


Fig. H-14. Designed Model Subsystem

There are three subsystems in the designed model: the H-Bridge, the rotary encoder, and the PWM system. The H-Bridge model can be seen in Fig. H-15.

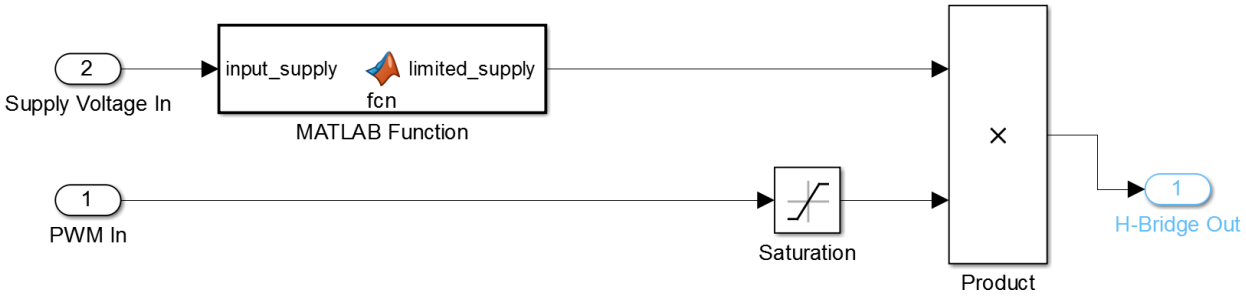


Fig. H-15. H-Bridge Model

The H-Bridge designed model has a MATLAB function in it that can be seen in Fig. H-16.

```
function limited_supply = fcn(input_supply)
%#codegen
if(input_supply<=10.4)
    limited_supply = 0;
else
    limited_supply=input_supply;
end
```

Fig. H-16. H-Bridge MATLAB Function

The rotary encoder model can be seen in Fig. H-17.

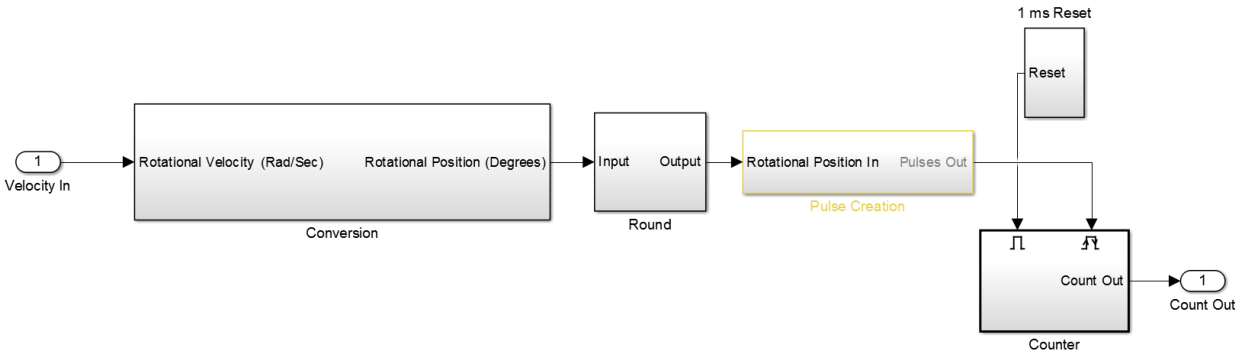


Fig. H-17. Rotary Encoder Model

The rotary encoder model has five subsystems: the conversion subsystem, the rounding subsystem, the pulse creation subsystem, the counter subsystem, and the reset subsystem. The conversion subsystem can be seen in Fig. H-18. It contains an integrator to convert the rotational velocity into rotational position.

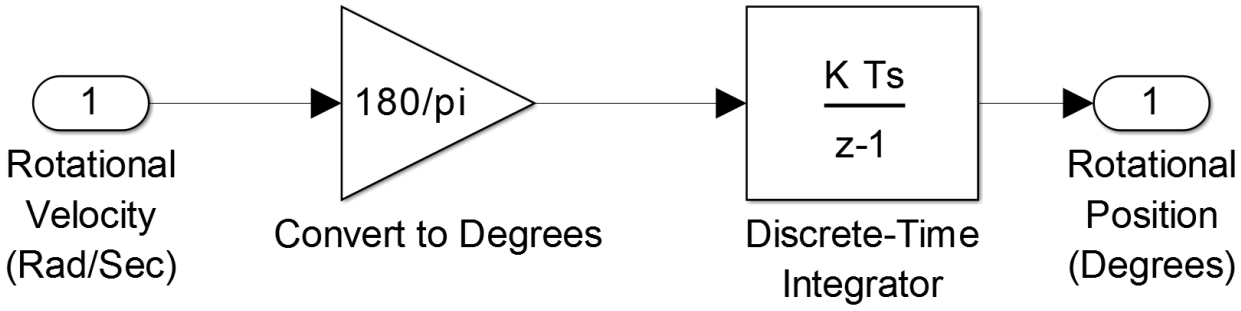


Fig. H-18. Rotary Encoder Conversion Subsystem

The rounding subsystem can be seen in Fig. H-19.

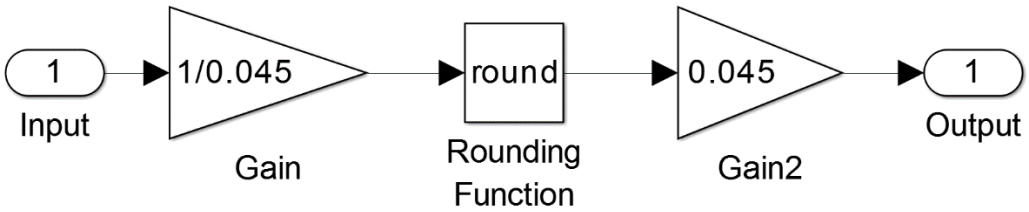


Fig. H-19. Rotary Encoder Rounding Subsystem

The pulse creation subsystem can be seen in Fig. H-20. It generates a pulse out for every 0.72 degrees that the motor spins.

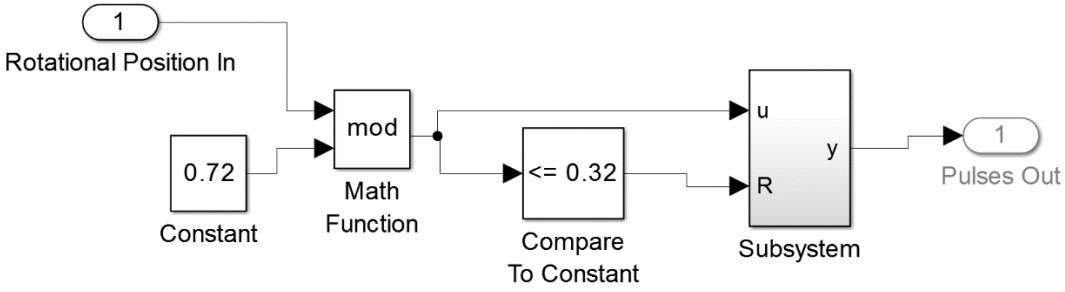


Fig. H-20. Rotary Encoder Pulse Creation Subsystem

The counter subsystem can be seen in Fig. H-21.

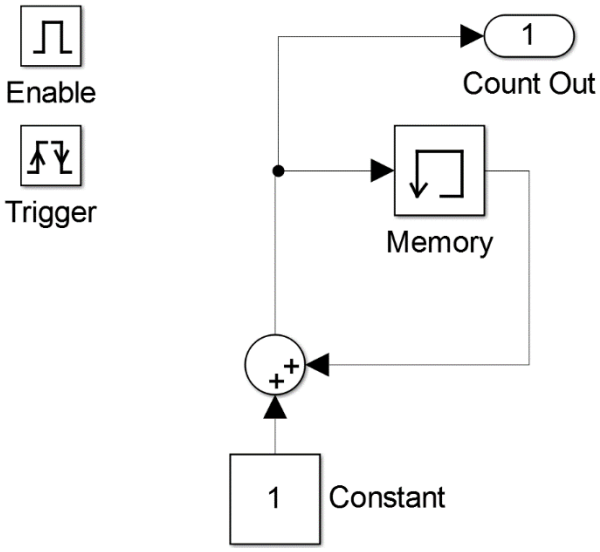


Fig. H-21. Rotary Encoder Counter Subsystem

The reset subsystem can be seen in Fig. H-22.

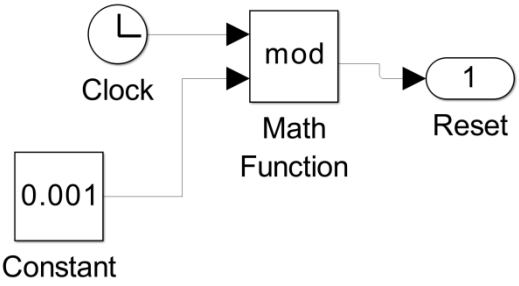


Fig. H-22. Rotary Encoder 1ms Reset Subsystem

The PWM system model can be seen in Fig. H-23.

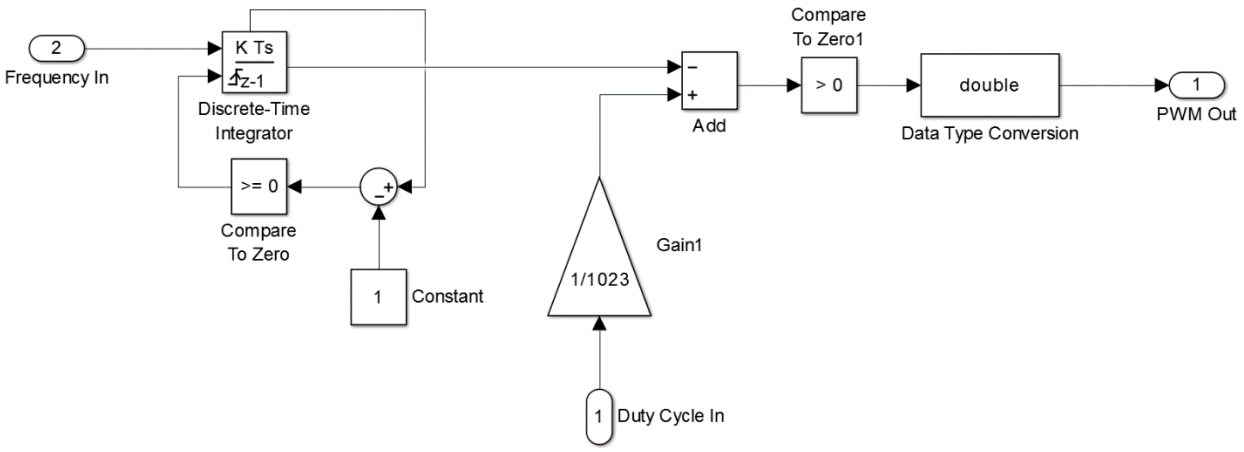


Fig. H-23. PWM Model

The average model can be seen in Fig. H-24.

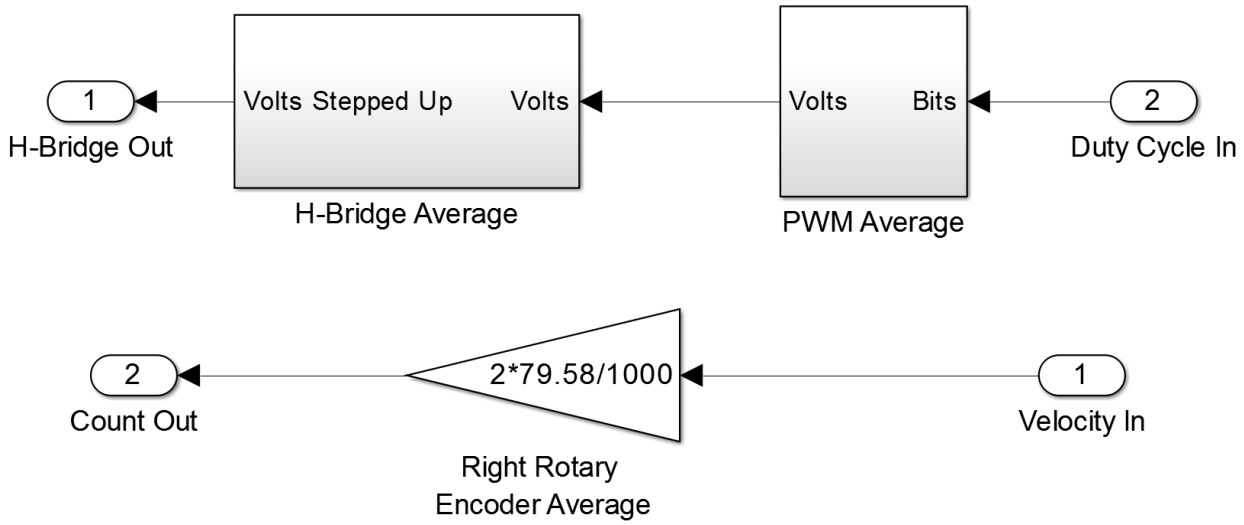


Fig. H-24. Average Models

There are two subsystems in the average model system: the average H-Bridge and the average PWM. The average H-Bridge can be seen in Fig. H-25.

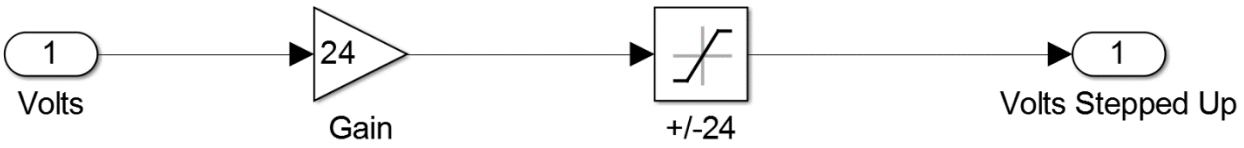


Fig. H-25. Average H-Bridge Model

The average PWM can be seen in Fig. H-26.

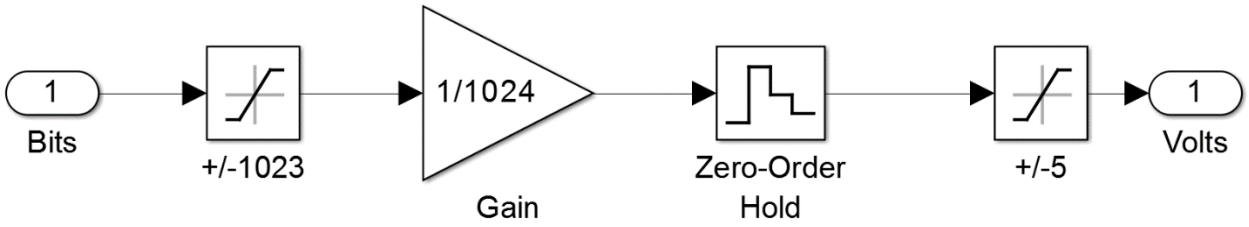


Fig. H-26. Average PWM Model

The motor-dynamic coupling can be seen in Fig. H-27.

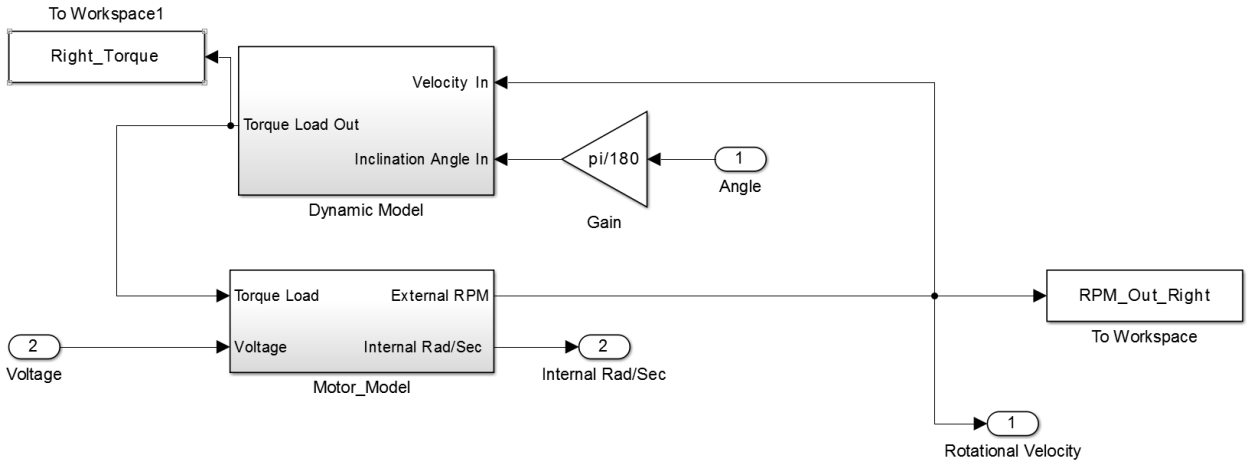


Fig. H-27. Motor-Dynamic Coupling Subsystem

There are two subsystems in the motor-dynamic coupling: the dynamic model and the motor model. The dynamic model can be seen in Fig. H-28.

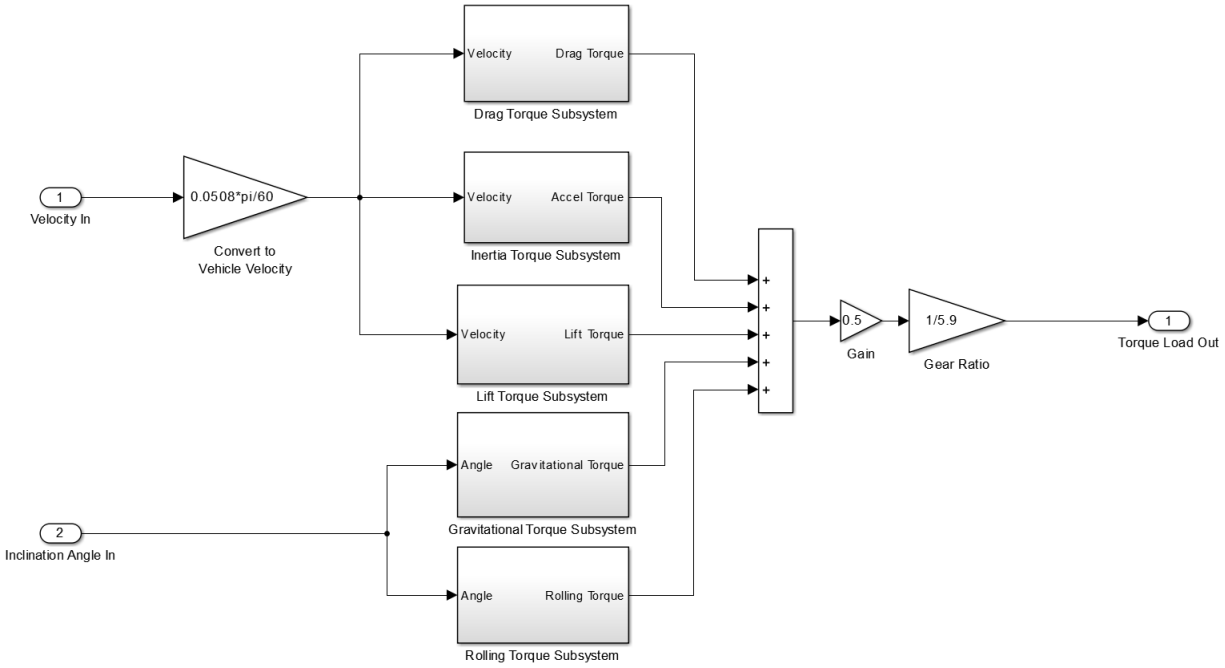


Fig. H-28. Dynamic Model Subsystem

There are five subsystems in the dynamic model: drag torque, inertia torque, lift torque, gravitational torque, and rolling torque. The drag torque model can be seen in Fig. H-29.

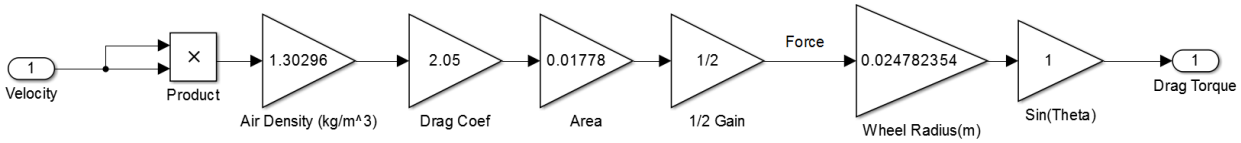


Fig. H-29. Aerodynamic Drag Torque Subsystem

The inertia torque model can be seen in Fig. H-30.

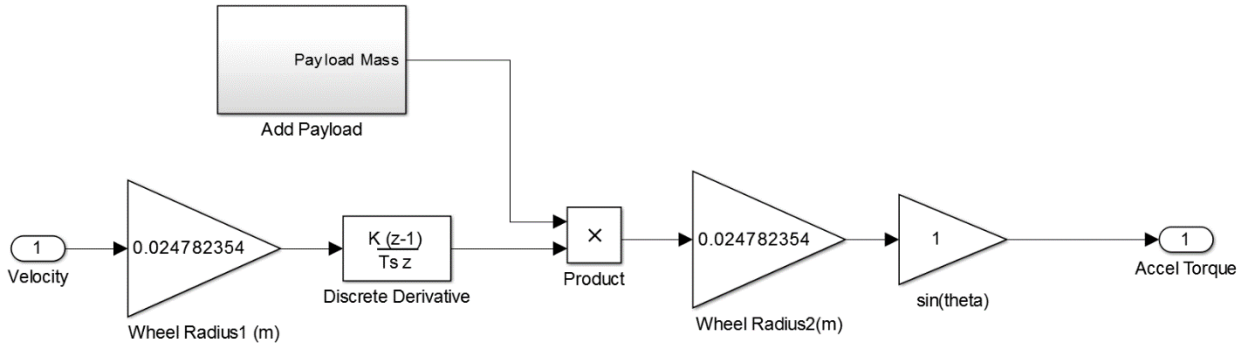


Fig. H-30. Inertia Torque Subsystem

The gravitational torque model can be seen in Fig. H-31.

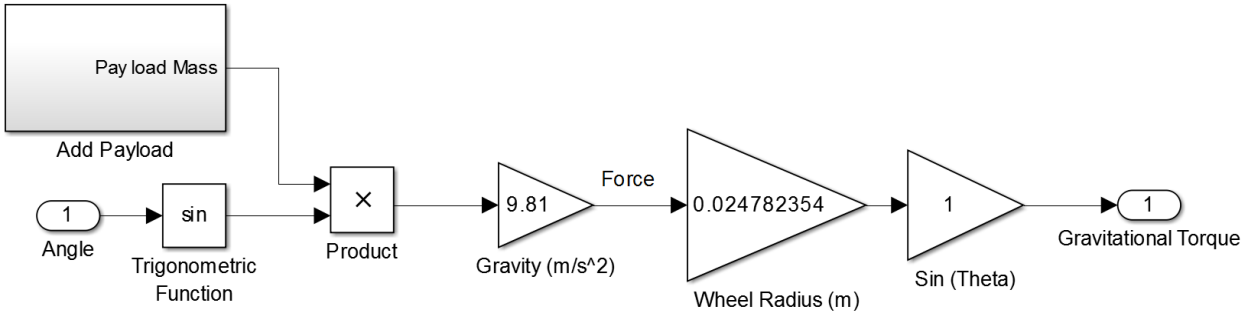


Fig. H-31. Gravitational Torque Subsystem

The rolling torque model can be seen in Fig. H-32.

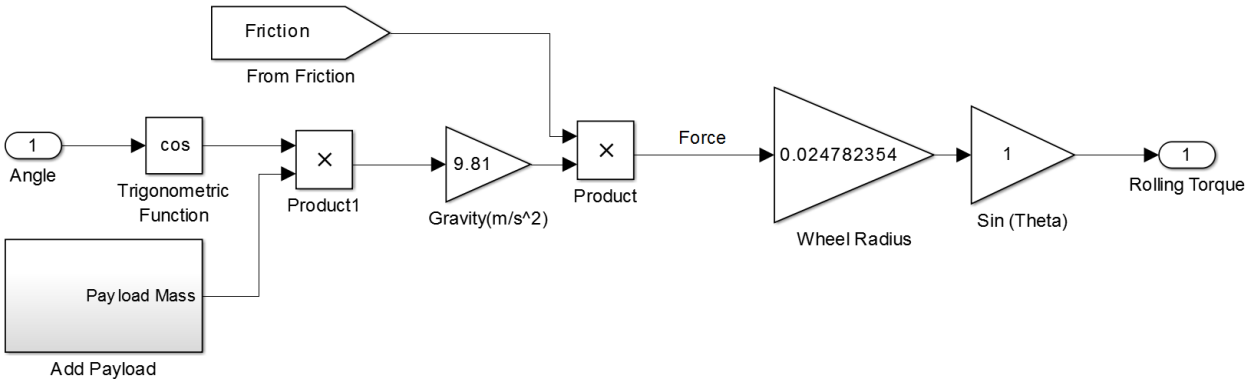


Fig. H-32. Rolling Torque Subsystem

The motor model can be seen in Fig. H-33.

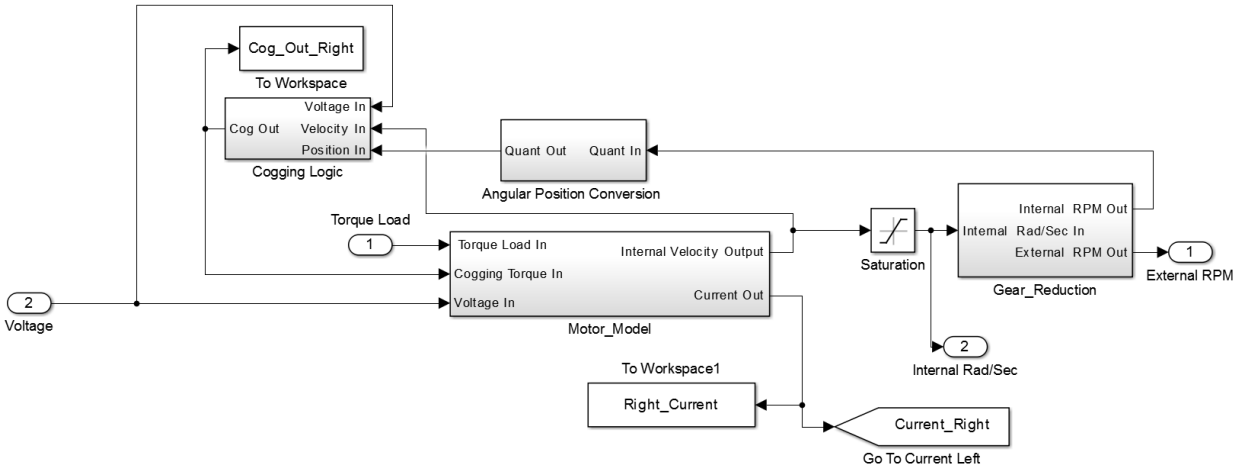


Fig. H-33. Motor Model

There are four subsystems in the motor model: cogging logic, angular position conversion, gear reduction, and motor model. The cogging logic subsystem can be seen in Fig. H-34.

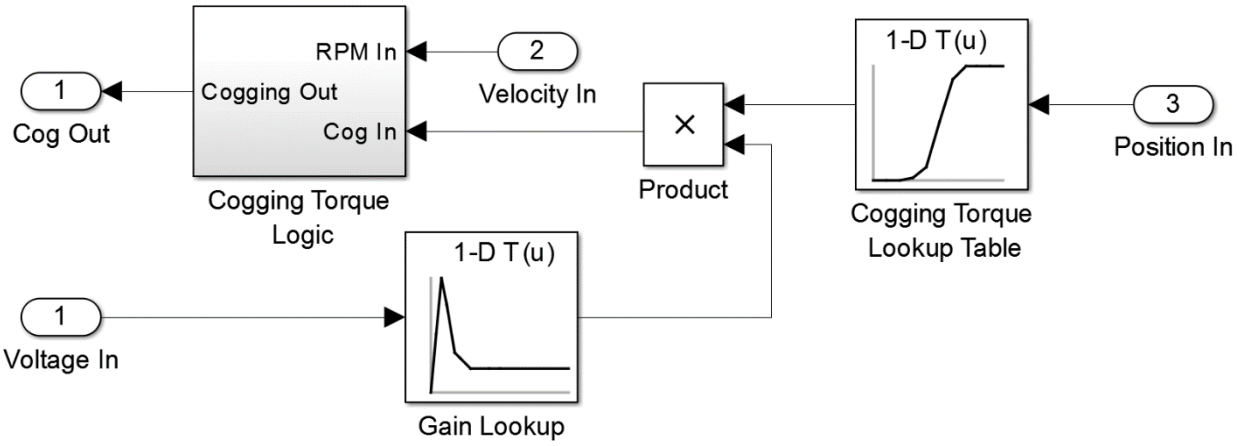


Fig. H-34. Cogging Logic Subsystem

The cogging logic block has one subsystem, cogging torque logic, which can be seen in Fig. H-35.

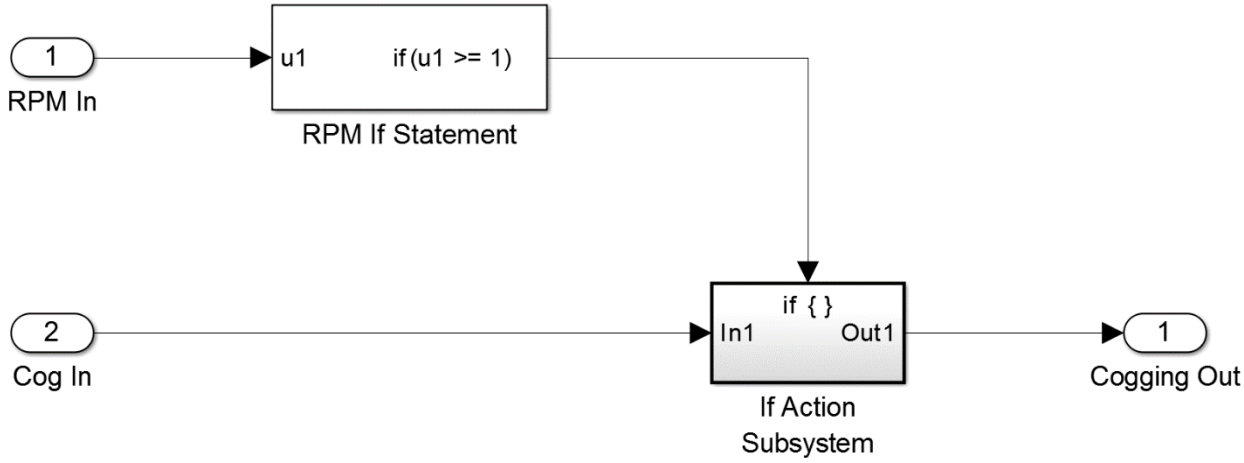


Fig. H-35. Cogging Torque Logic Subsystem

The angular position conversion block can be seen in Fig. H-36.

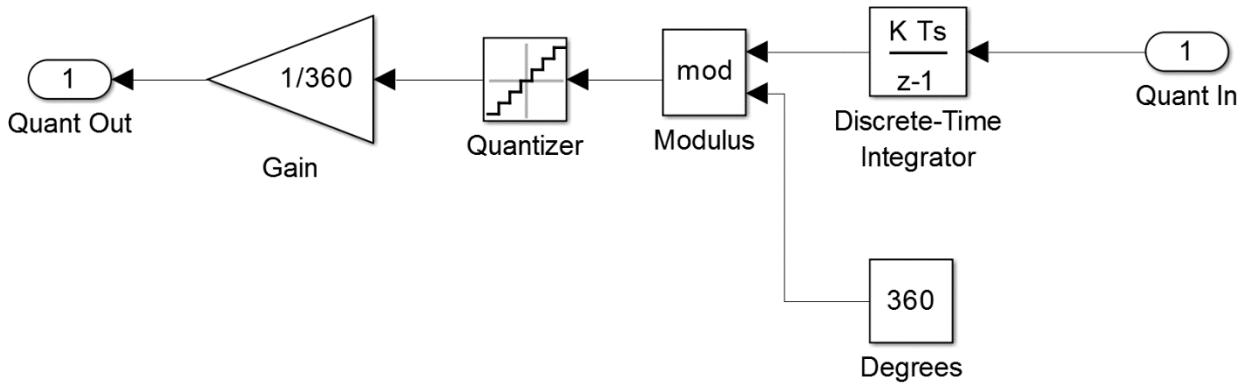


Fig. H-36. Angular Position Conversion Subsystem

The gear reduction block can be seen in Fig. H-37.

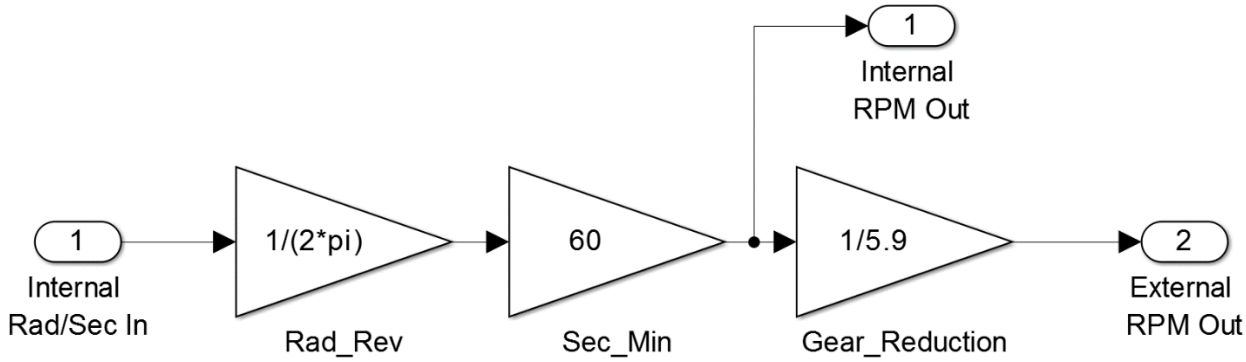


Fig. H-37. Gear Reduction Subsystem

The motor model subsystem can be seen in Fig. H-38.

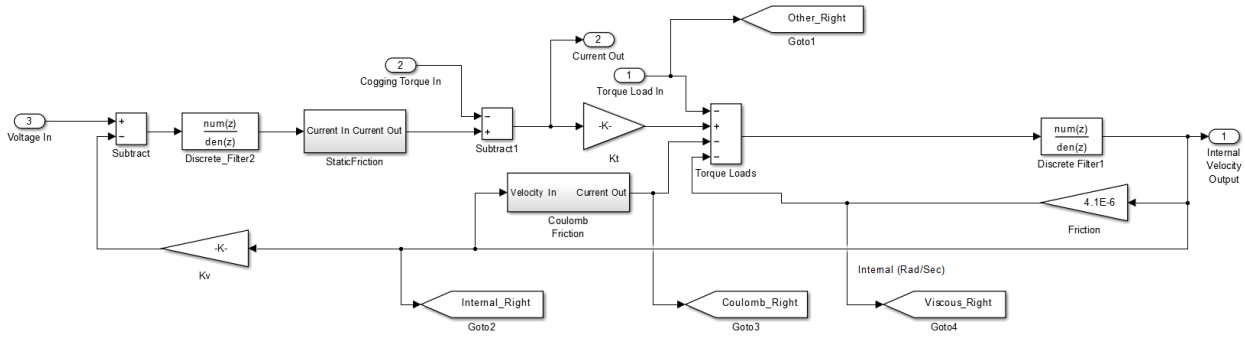


Fig. H-38. Motor Model Subsystem

There are two subsystems in the motor model system: a static friction block and a Coulomb friction block. The static friction block can be seen in Fig. H-39 and the code for the static friction can be seen in Fig. H-40.

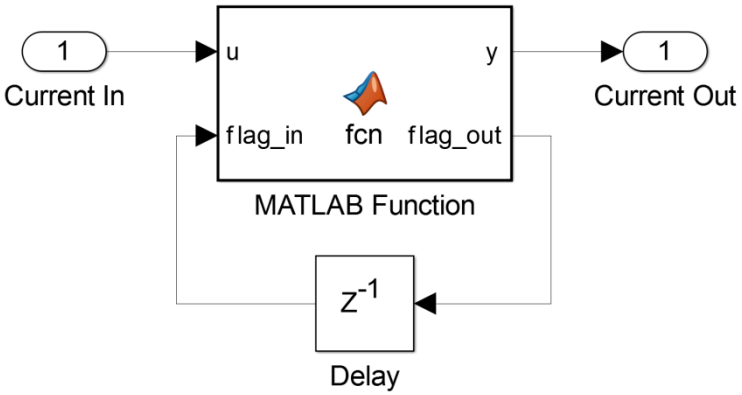


Fig. H-39. Static Friction Subsystem

```
function [y,flag_out] = fcn(u,flag_in)
if u >= 0.1738
    flag_out = 1;
elseif u == 0
    flag_out = 0;
else
    flag_out = flag_in;
end


y = u*flag_out;
```

Fig. H-40. Static Friction MATLAB Function

The Coulomb friction block can be seen in Fig. H-41.

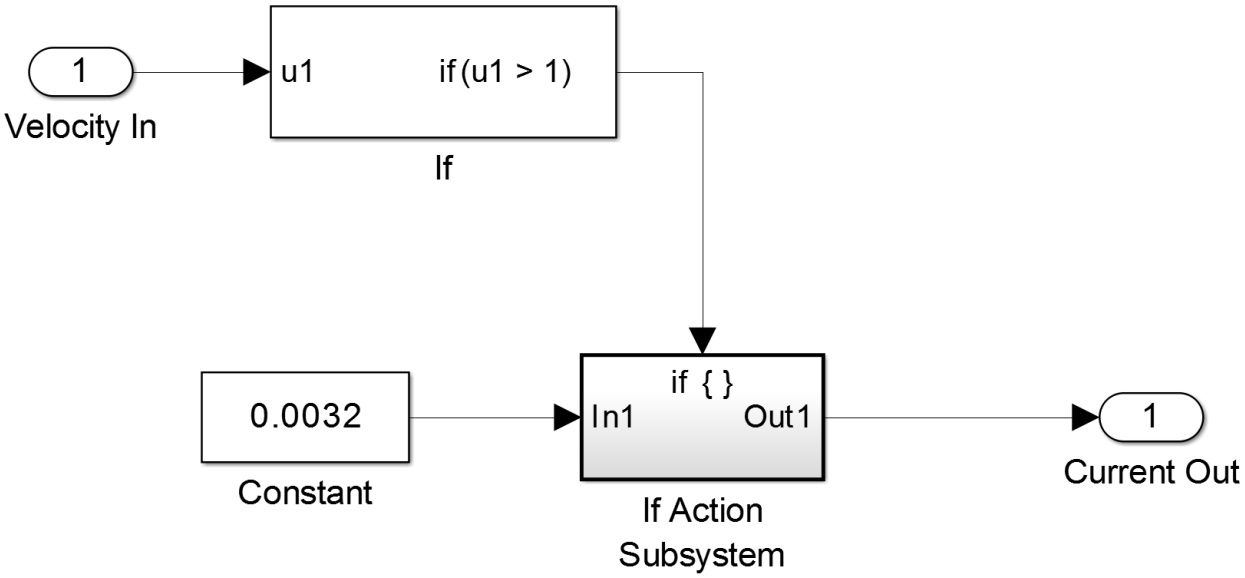


Fig. H-41. Coulomb Friction Subsystem

The Kinematic Model can be seen in Fig. H-42.

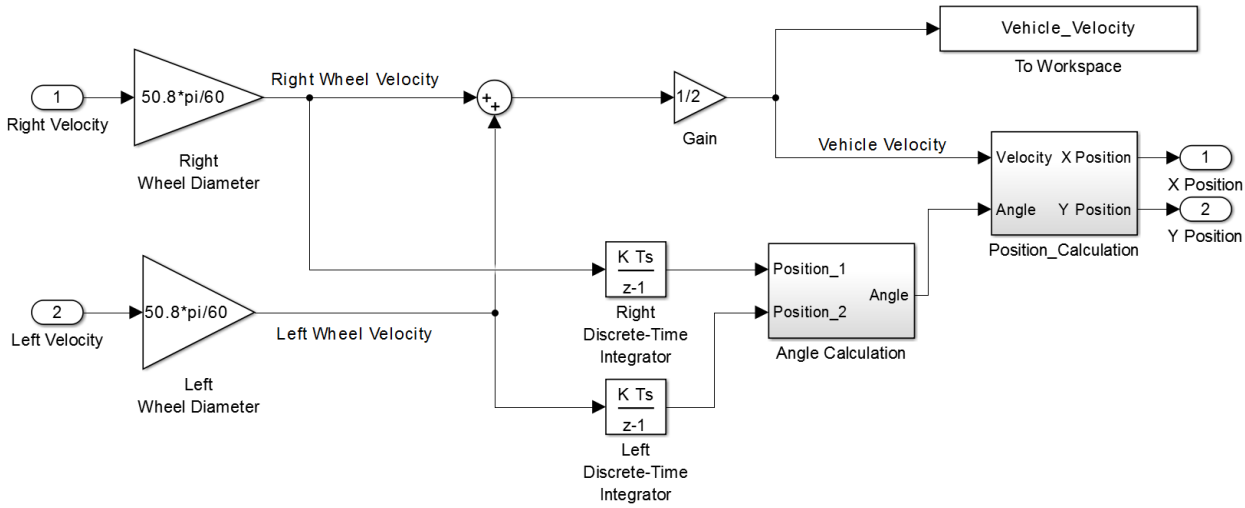


Fig. H-42. Kinematic Model

There are two subsystems in the kinematic model; the angle calculation subsystem seen in Fig. H-43 and the position calculation subsystem seen in Fig. H-44.

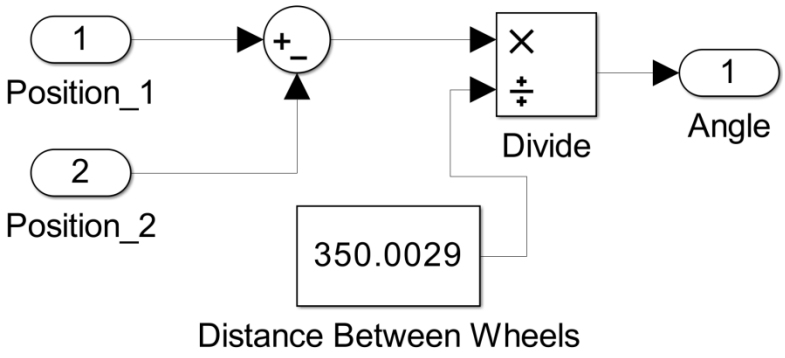


Fig. H-43. Angle Calculation Subsystem

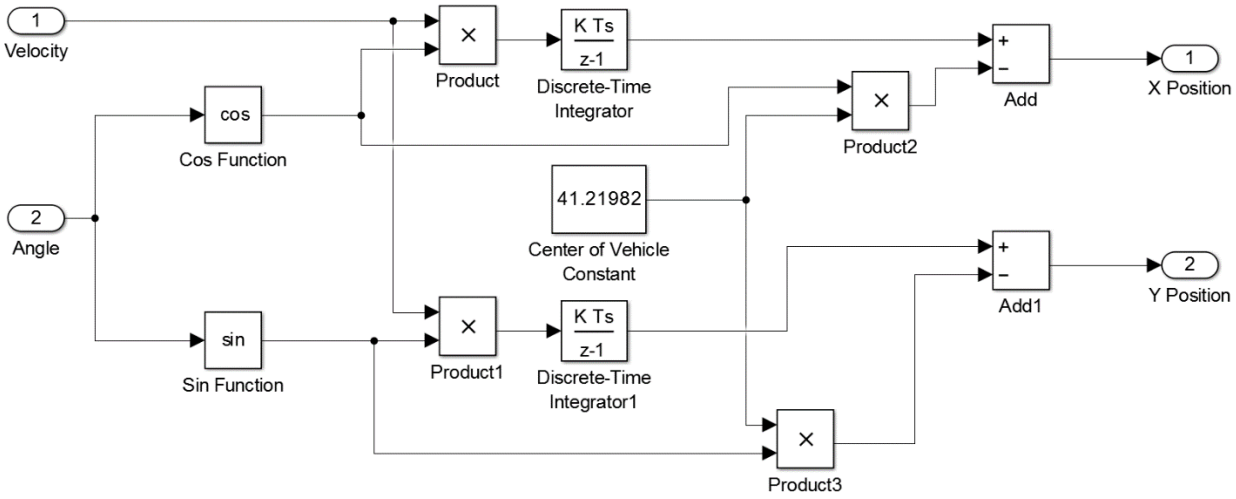


Fig. H-44. Position Calculation Subsystem

I. *Detailed GUI Design*

The design of the MATLAB graphical user interface (GUI) for use in the project started with a list of tasks that the GUI needed to be able to accomplish. That list can be seen in Table I-I.

TABLE I-I GRAPHICAL USER INTERFACE REQUIREMENTS

Needs for GUI
Change the surface friction coefficient
Change the inclination angle
Set vehicle velocity
Display motor velocity
Display vehicle path
Display vehicle acceleration
Display error signal
Stop Button
Start Button
Select from a list of tests or create own test
Toggle plots
Add a payload
Toggle between the experimental platform and Simulink System
Change motor parameters
Change gravity
Select an ideal vehicle

The functions of the GUI were taken from this table and expanded. The first step was to design the ability to create a test in the GUI or allow the user to select a premade test to run. Those portions of the GUI can be seen in Fig. I-1 and Fig. I-2.

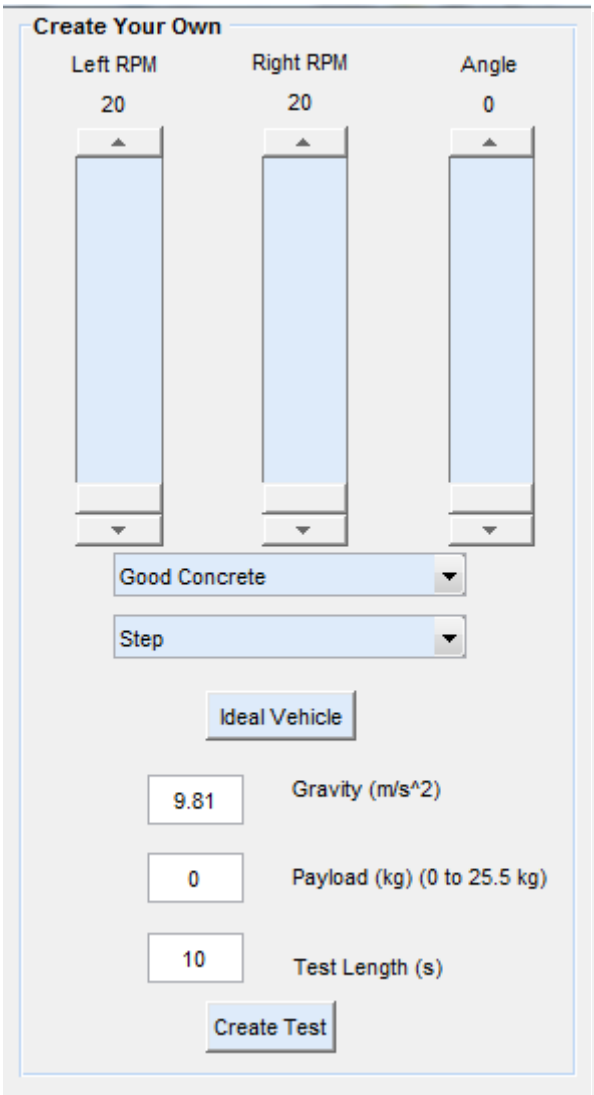


Fig. I-1. GUI Create Test

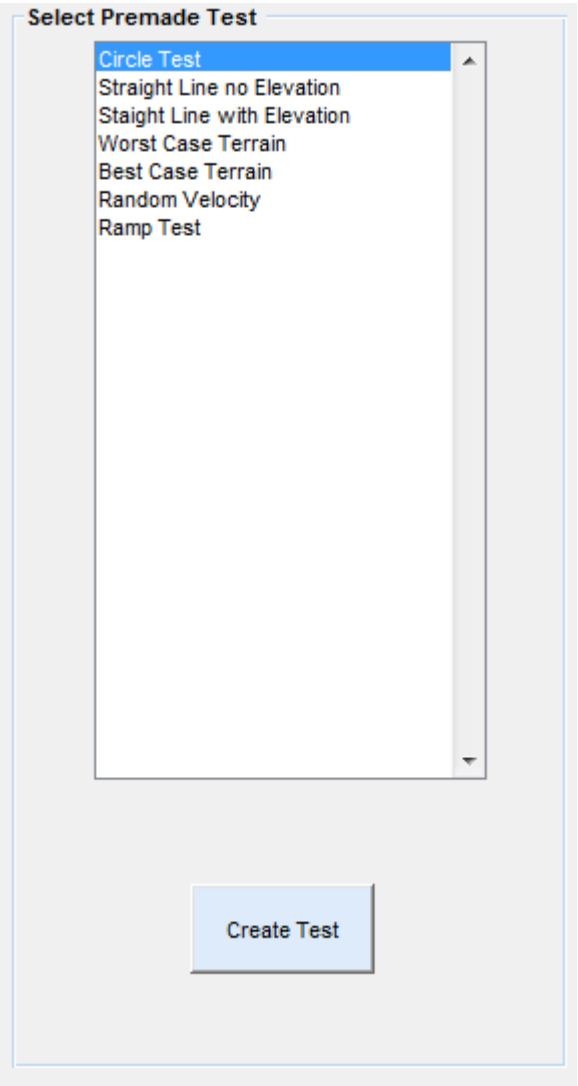


Fig. I-2. GUI Select a Premade Test

The next step was to give the ability to the user to change the motor parameters of the two motors in the system. This uses editable text boxes. The part of the GUI that allows this can be seen in Fig. I-3.

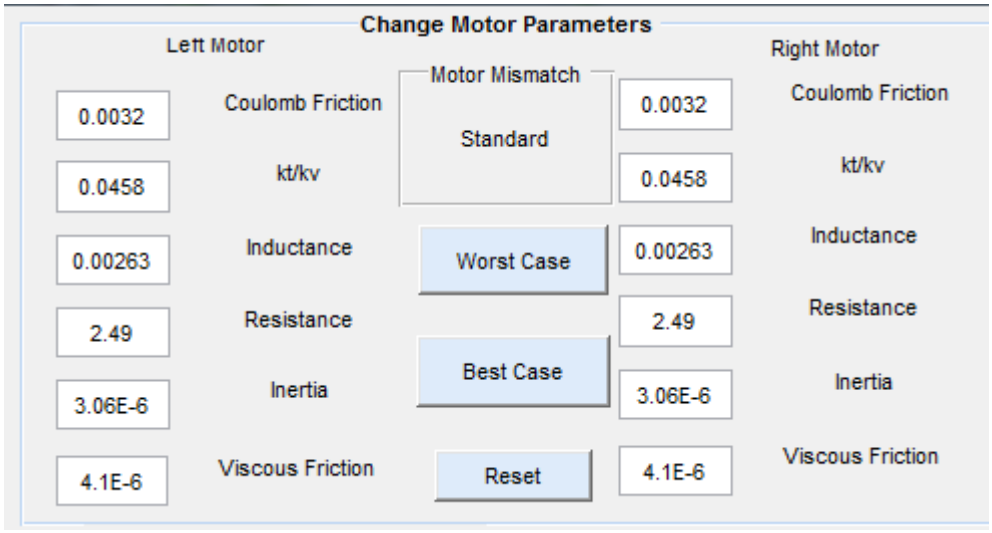


Fig. I-3. Motor Parameter Editor

After this was completed, the next part was adding the ability to toggle between the different systems, and select different types of plotting and testing for each system. The buttons to toggle between different aspects can be seen in Fig. I-4.

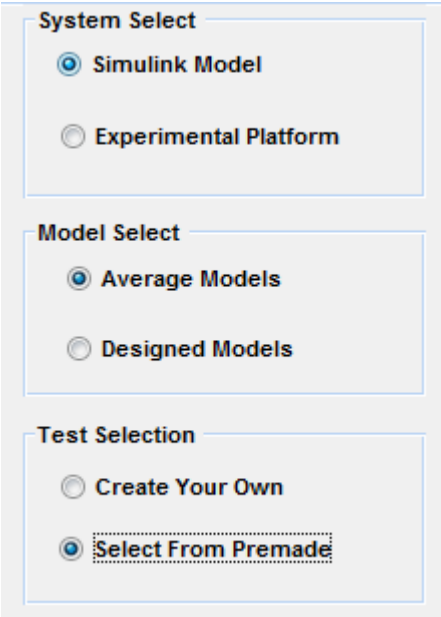


Fig. I-4. GUI Toggle Buttons

The GUI with the Simulink system selected can be seen in Fig. I-5.

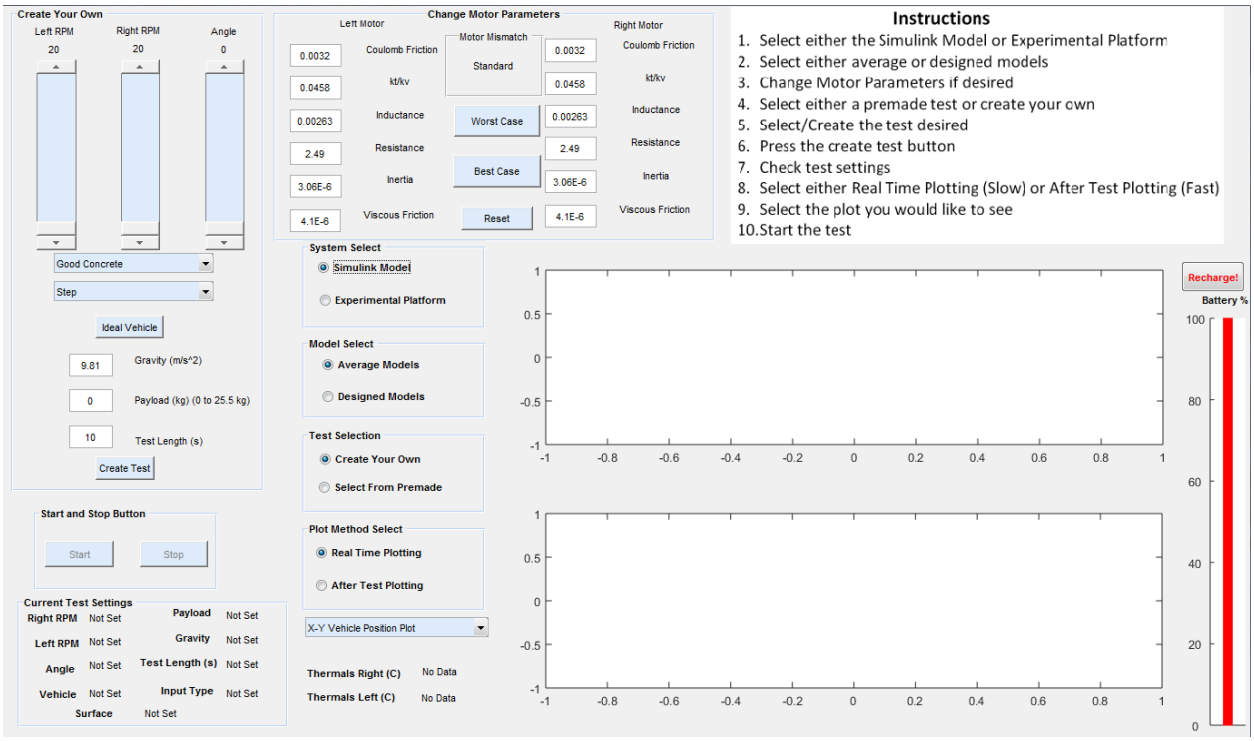


Fig. I-5. GUI Simulink System Mode

The GUI with the experimental platform selected can be seen in Fig. I-6.

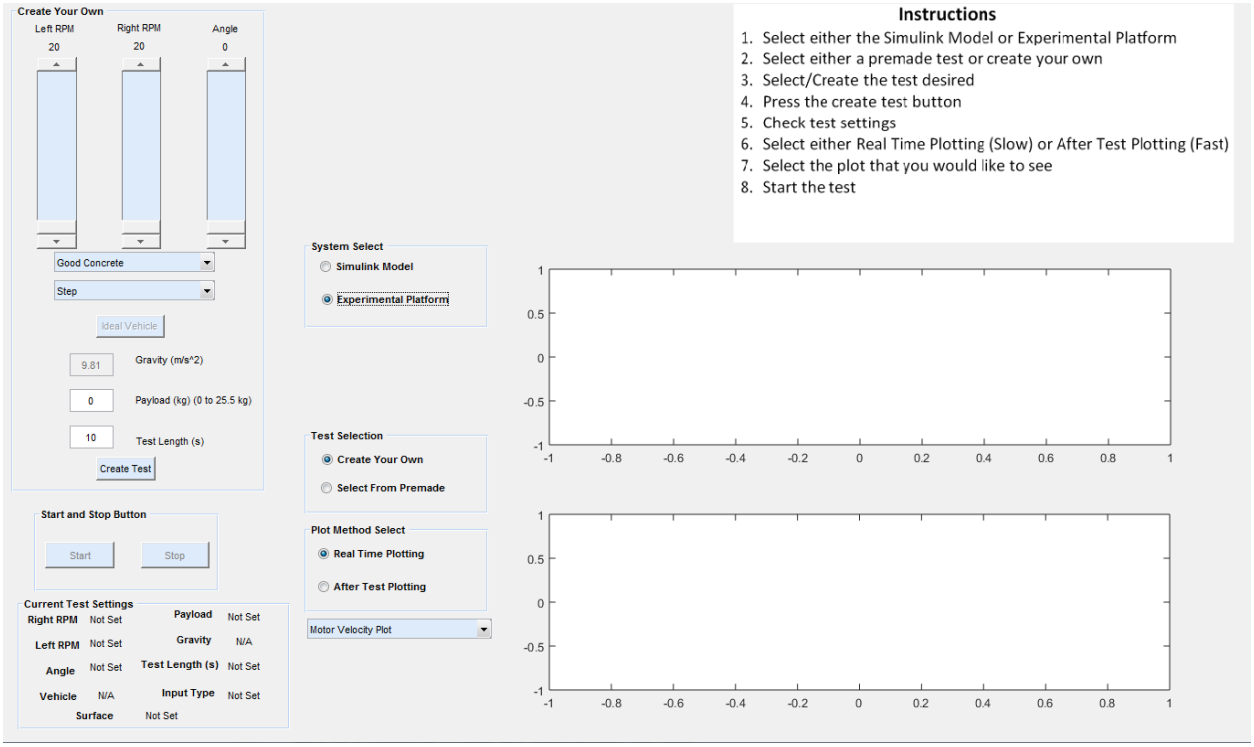


Fig. I-6. GUI Experimental Platform Mode

The reason for the change in the GUI is because some of the options that are available in the Simulink System aren't available with the experimental platform. The options were removed to prevent confusing the user. The next step that had to be taken was adding the part of the GUI that allows the user to choose between the different plots that they want to see. To do this the user first chooses between real time plotting and after test plotting. This was necessary because the real time plotting in the Simulink system is slow because the computer has to use a listener to record every value that goes through a block in Simulink. This is a very computational intensive process and slows the simulation of the system by a factor of ten.

After the user chooses between real time and after test plotting, the next step is to choose which plot to see. This choice can be seen in Fig. I-7.

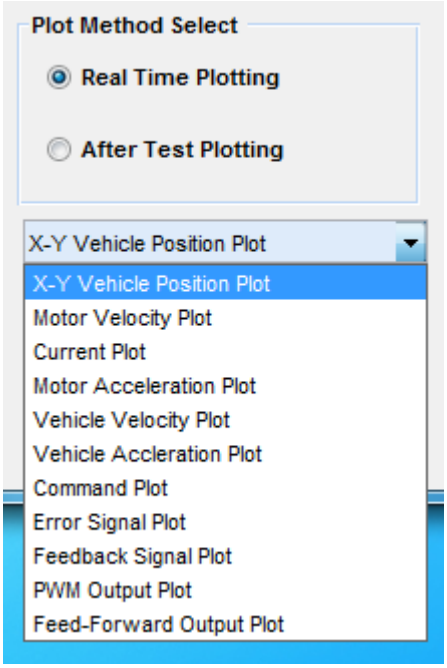


Fig. I-7. GUI Plot Selection

The last step was actually plotting the results of a test. This part of the GUI can be seen in Fig. I-8.

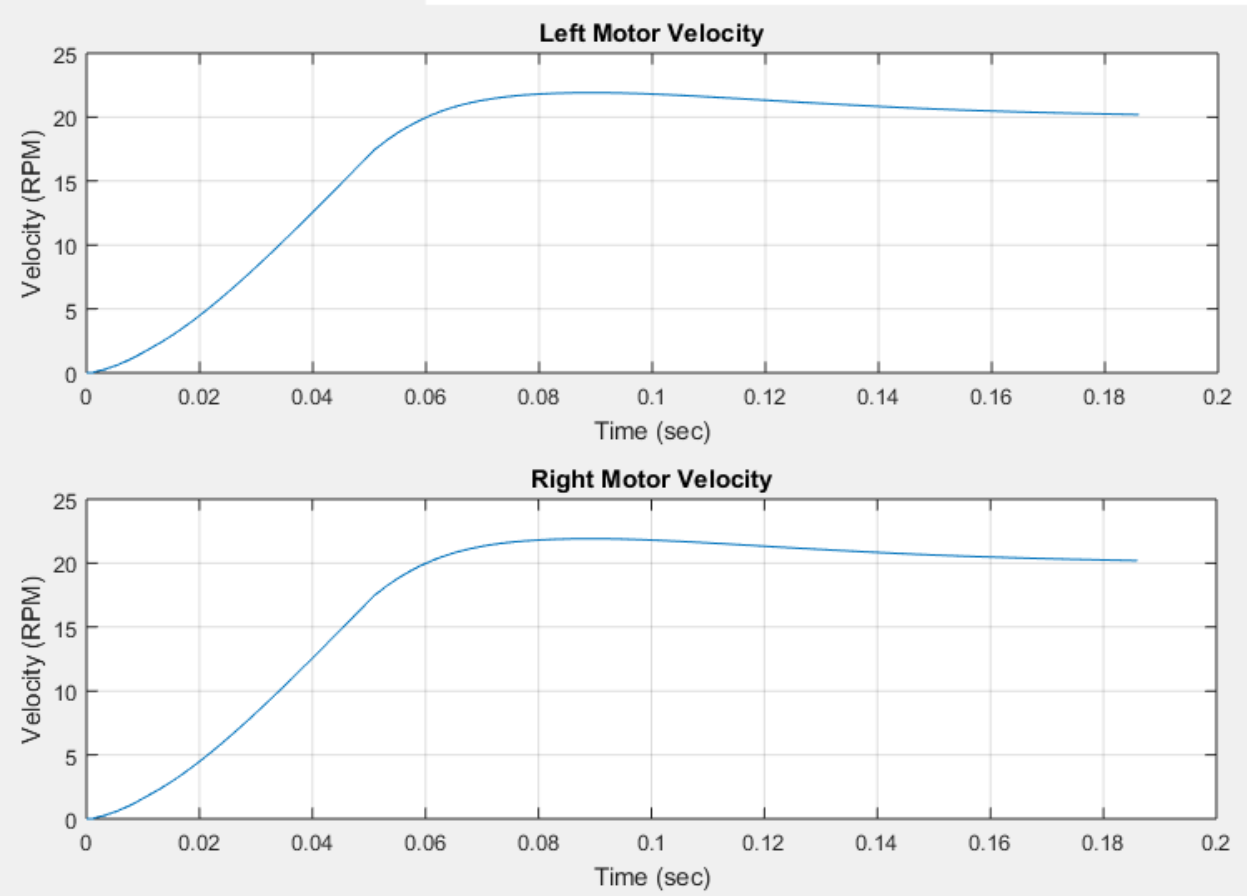


Fig. I-8. GUI Plots

After the whole GUI was designed, it was given to two senior electrical engineering students to test. The results of this test were adding instructions, a battery indicator, and the ability to see what type of test is selected before running the final test. These changes can be seen in Fig. I-9, Fig. I-10, and Fig. I-11.

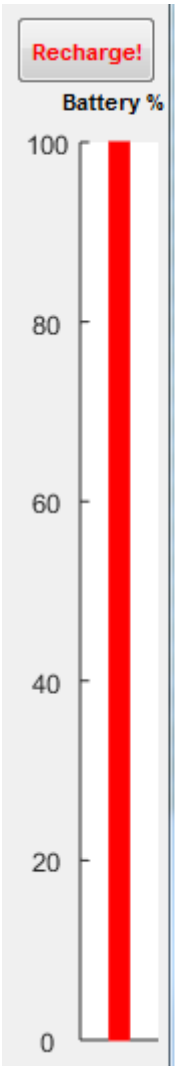


Fig. I-9. GUI Battery Monitor

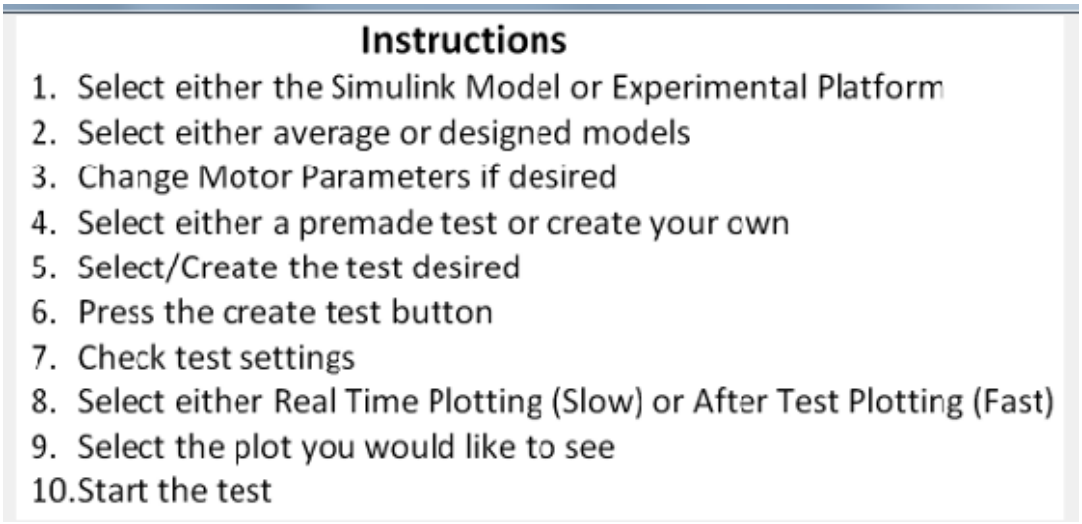


Fig. I-10. GUI Instructions

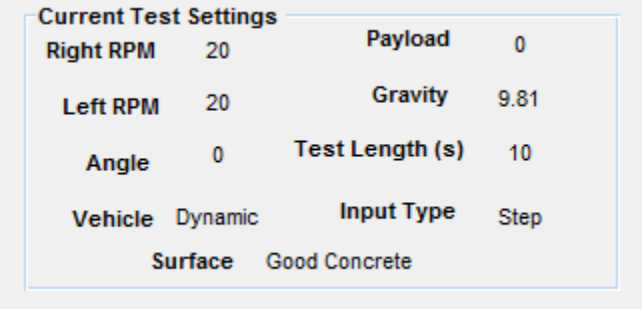


Fig. I-11. Current Test Details

After these changes were implemented the GUI was finished. The final GUI can be seen in Fig. I-12.

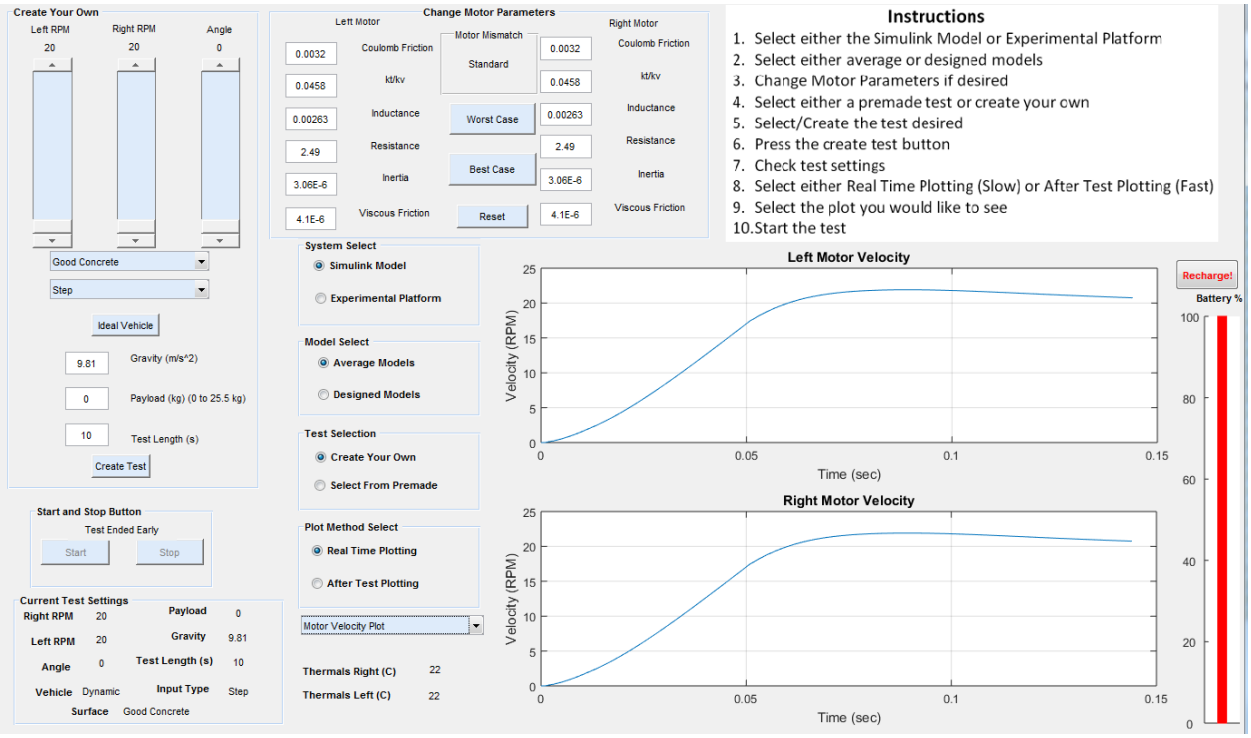


Fig. I-12. Final GUI

J. *Detailed Cogging Torque*

For cogging torque, there was research on how it would be tested and implemented. Originally the "Scale Method" was devised and a scale was going to be used at small increments to view the load at different shaft angles. After a sufficient amount of research, it was determined this method would not be accurate enough. A figure of the setup can be seen in Fig. J-1. The "Frequency Method" was then devised, which can be seen in Fig. J-2. The frequency method measured the rotary encoder jitter and mapped it for

analysis. By mapping the jitter, a display of cogging torque can be shown. Unfortunately, this method failed due to the lack of resolution of the oscilloscopes. A wide enough spread could not be obtained to be able to show one full revolution of cogging torque. The final method was then created, which was called the "Current Method". The Current Method used a current probe attached up to one of the older oscilloscopes owned by Bradley University, and this allowed a current versus time plot to be generated. The current can be translated into torque with the motor torque constant *Kt*. This method was the implemented solution to modeling cogging torque, and this can be seen in Fig. J-3. It can be verified that Fig. J-3 is one full revolution of the motor not only by the number of samples given from the rotary encoder, but also the number of peaks in the graph. There are seven major and seven minor peaks in Fig. J-3, and these peaks correlate to the two permanent magnets and seven iron core segments of the Pittman motor. This further validated the readings that were achieved from the oscilloscope.

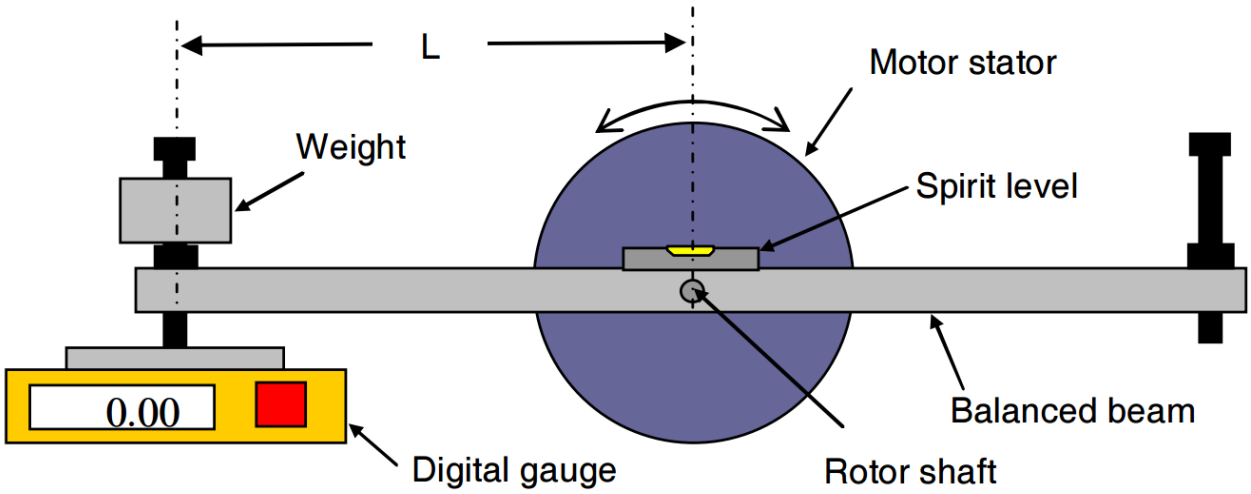


Fig. J-1. Scale Method Testing [18]

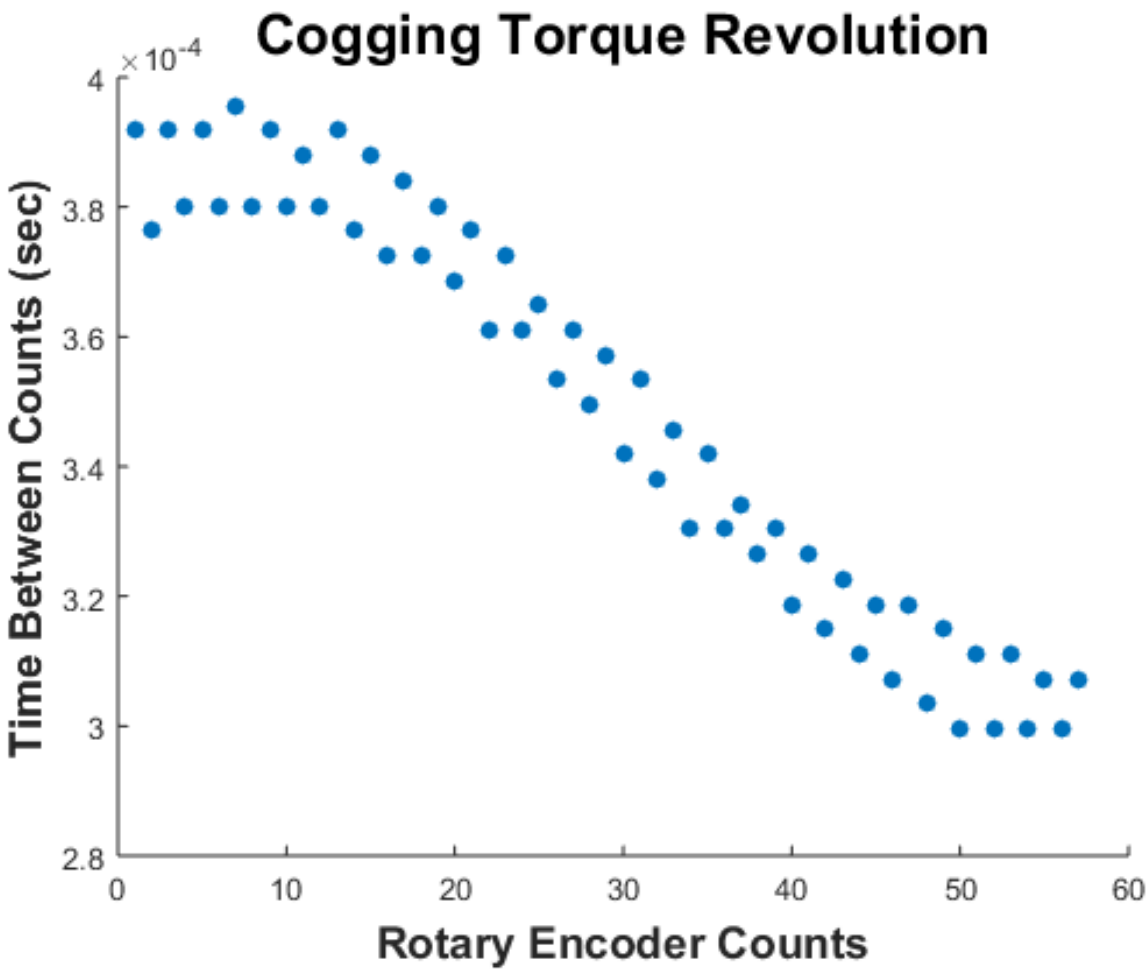


Fig. J-2. Frequency Method Testing

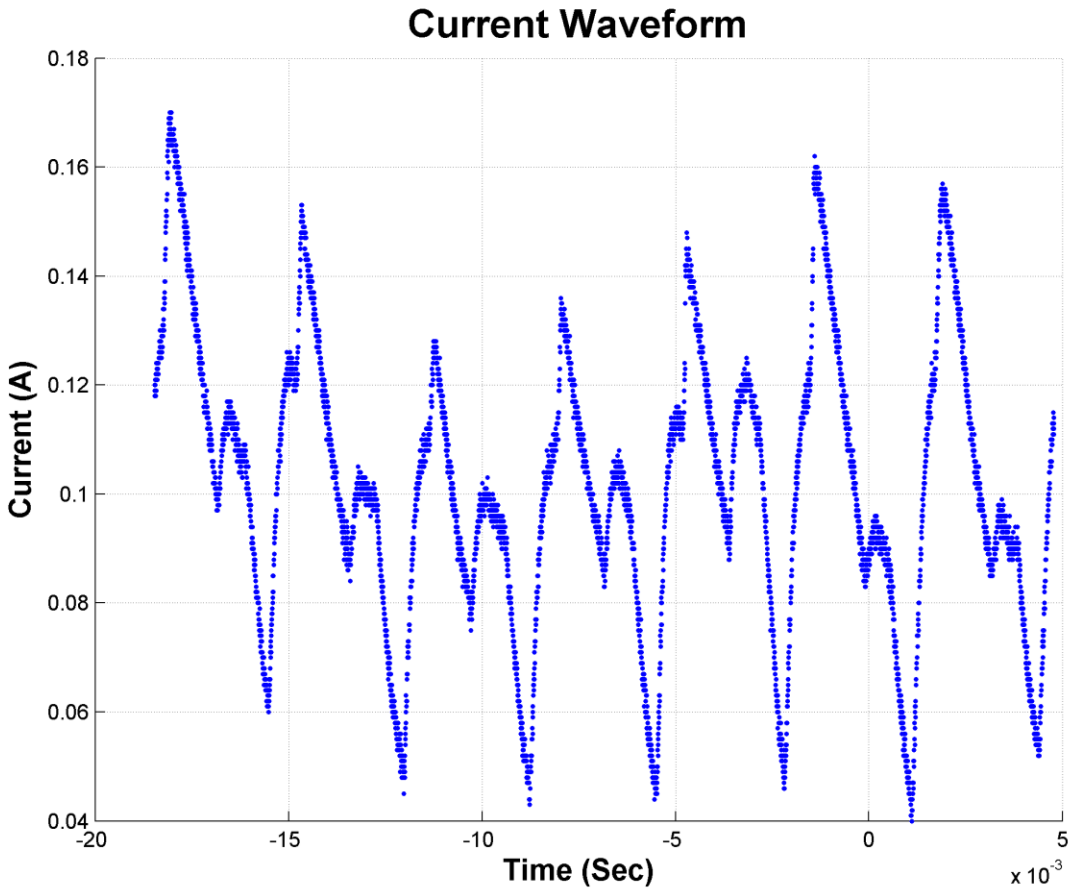


Fig. J-3. Current Method Testing

K. *Detailed Motor Modeling*

The general motor model can be seen in Fig. K-1. The electrical and mechanical transfer functions were calculated via experimental procedures. The values found experimentally for the motor model can be seen in Table K-I.

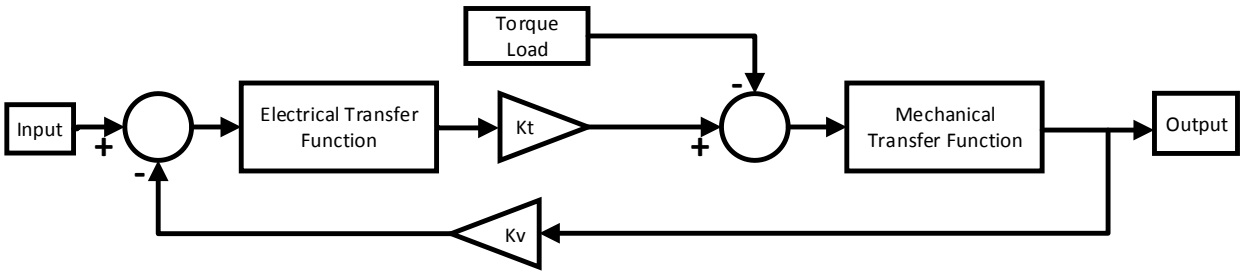


Fig. K-1. General Motor Model

TABLE K-I. Motor Parameter Values

Constant	Experimental	Data Sheet	Units
Viscous Friction	4.11E-06	3.54E-06	Nm/Rad/Sec
Coulomb Friction	0.0032	0.0056	Nm
Kv	0.0431	0.0458	V/Rad/Sec
Kt	0.0431	0.0458	Nm/A

L. *Detailed Current Source Design*

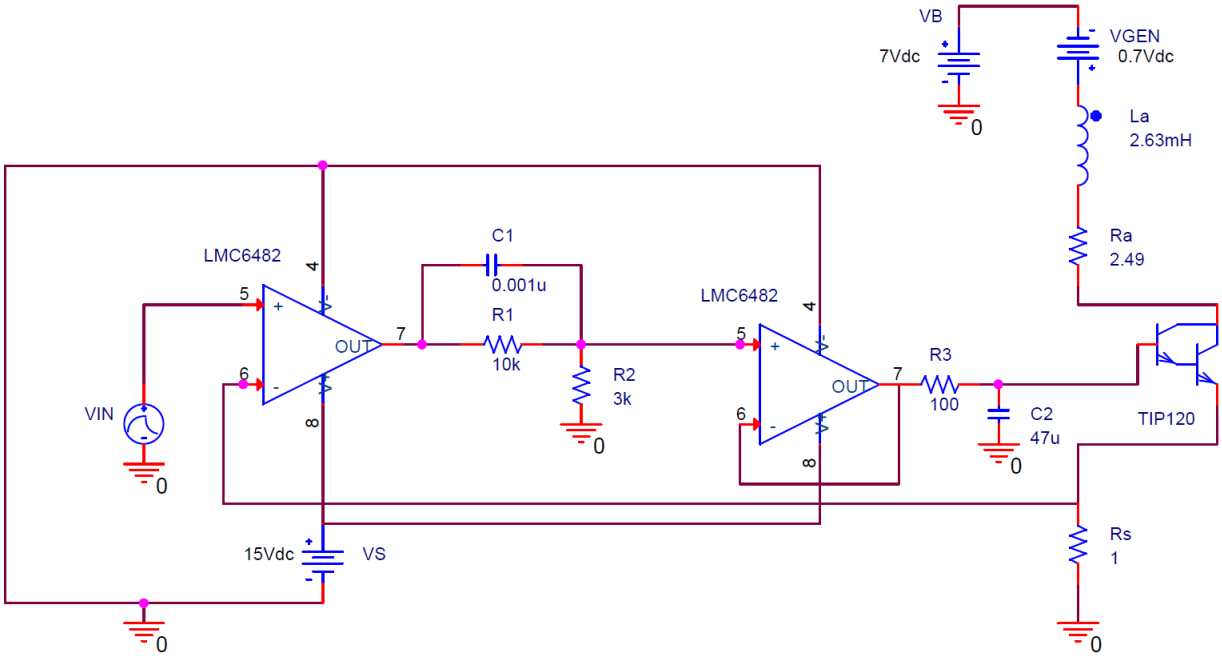


Figure L-1: Final Current Source Schematic.
V_{IN} is the voltage input from the microcontroller. V_{GEN} is the generator induced voltage. *La* and *Ra* are the generator winding impedances.

The final current source schematic includes added control circuitry to ensure current output stability.

$$G_{plant} = G_{op-amp} * G_{BJT-genset} * G_{gain-mod} \quad [L-1]$$

$$G_{op-amp}(s) = (119 * 10^3)\frac{1}{\left(\frac{1}{2\pi * 10}s + 1\right)\left(\frac{1}{2\pi * 1.22 * 10^6}s + 1\right)} \quad [L-2]$$

$$G_{BJT-genset}(s) = \frac{1}{(\frac{1}{2\pi * 10^4}s + 1)^2(\frac{1}{2\pi * 3 * 10^5}s + 1)^2} \quad [L-3]$$

The unstable circuit plant used to design the added control circuitry consists of three parts: the op-amp plant in [L-2], the transistor and generator combination plant in [L-3], and a nonlinear gain modification used to roughly model nonlinear behaviors of the combined system such as the op-amp slew rate shown in Fig. L-2. This gain modification approximates the behavior of a slew rate by attenuating low frequency signals but passing through higher frequency signal components. Specific gain values were chosen to

match the mathematical circuit plant model to PSPICE circuit simulation results at the point where phase margin was approximately zero. In order to match the model with PSPICE results more easily, only gain values were tuned, and the phase of the gain adjustment was kept at zero. The linear components are shown in the Laplace domain.

The previous plant models were derived from PSPICE frequency responses of the separate components in the system.

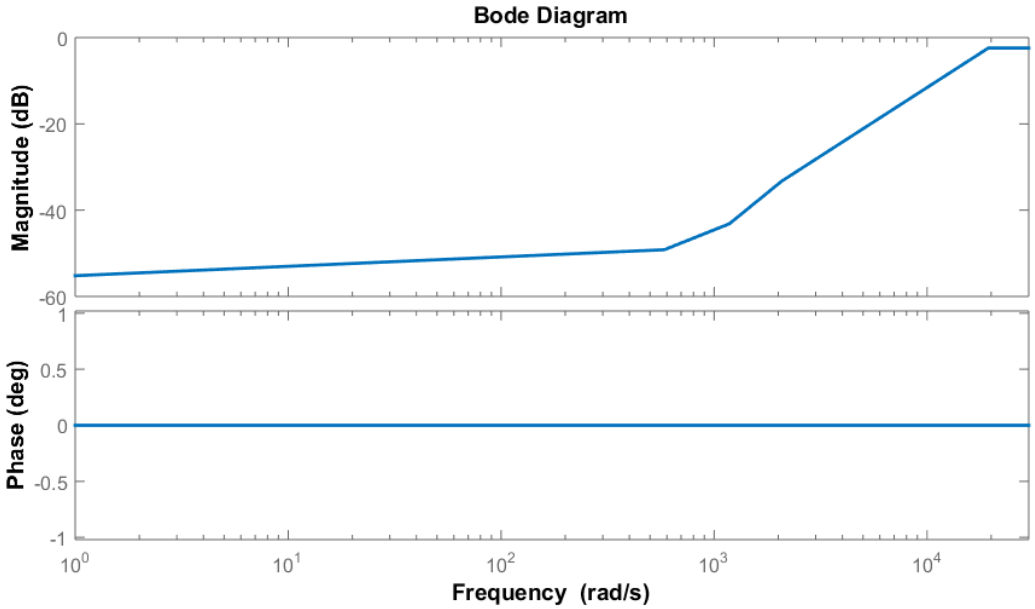


Figure L-2: Nonlinear Current Source Plant Gain Adjustment for Better PSPICE Matching.

$$G_{compensator}(s) = 0.233\frac{\left(\frac{1}{2\pi * 3.18 * 10^4}s + 1\right)}{\left(\frac{1}{2\pi * 15.9}s + 1\right)\left(\frac{1}{2\pi * 1.38 * 10^5}s + 1\right)\left(\frac{1}{2\pi * 10^6}s + 1\right)} \qquad [L-4]$$

Equation [L-4] shows the Laplace domain model of the added controller circuitry.

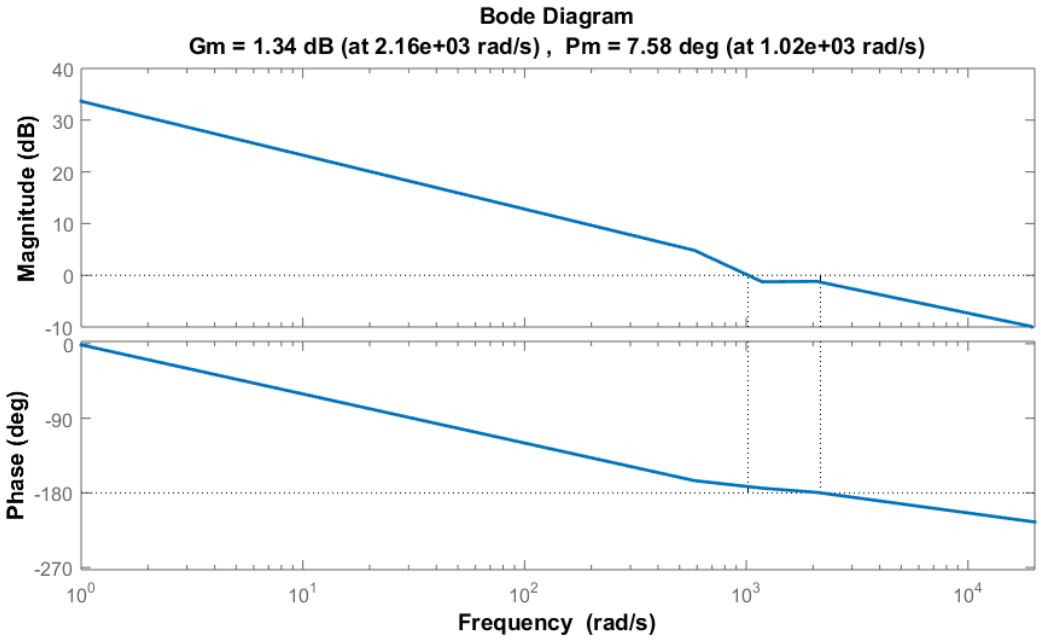


Figure L-3: Compensated Nonlinear Frequency Response of the Open Loop Current Source Circuit

The additional controller circuitry adds a zero close to the crossover frequency to add phase margin to the system as well as a pole close to DC at the transistor base to block effects of the winding inductance from the op-amps. Fig. L-3 shows a poor phase margin for the resulting system. This poor phase margin is not necessarily indicative of the true phase margin in the physical circuit because the nonlinear gain serves only as a design estimation. The value of capacitor *C2* in Fig. L-1 was tuned to 100 µF to yield a reliable phase margin. Evidence of a high phase margin can be seen in Fig. L-4, which shows a relatively low overshoot for the step response.

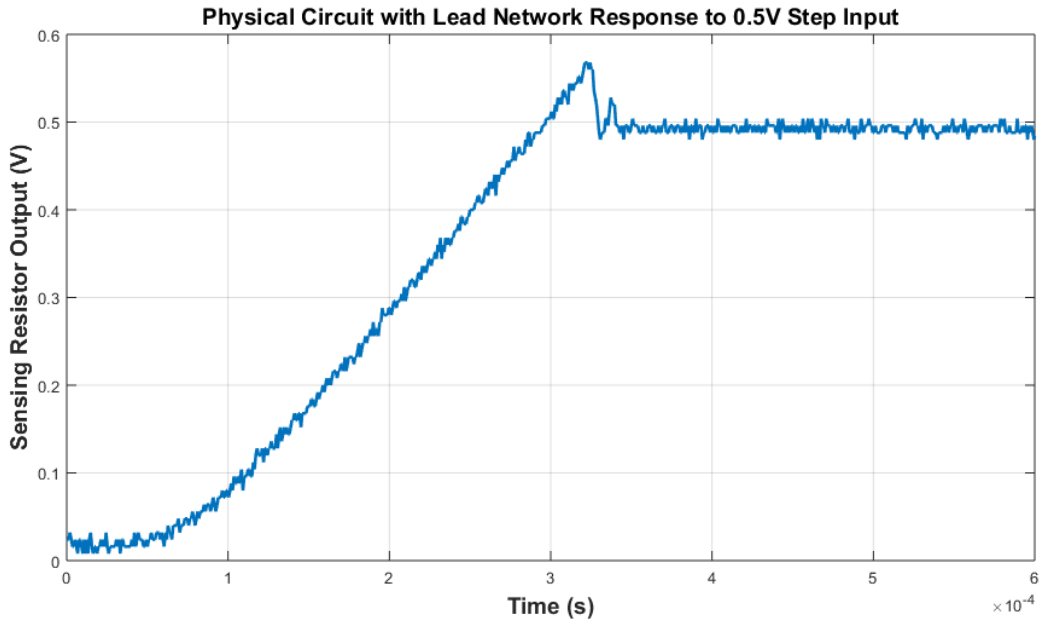


Figure L-4: Compensated Physical Circuit Step Response

M. *Detailed Torque/Acceleration Matching*

The experimental platform open loop response should accurately match that of the Simulink vehicle model in order for the system to accurately model the theoretical vehicle. Motor acceleration should be matched for accurate simulation of a velocity control system.

$$\frac{T_{SIM}}{J_{SIM}} = \frac{T_{EXP}}{J_{EXP}} = rotational\ acceleration \qquad [M-1]$$

As seen in [M-1], the ratio of the net motor torque to the system inertia is equal to the rotational acceleration of the motor. The Simulink vehicle model contains both the inertia of the motors as well as the vehicle itself. The experimental platform only contains the inertia of the motor and coupled generator. Thus, in order to match the acceleration of the Simulink model, the net motor torque in the experimental platform must be reduced.

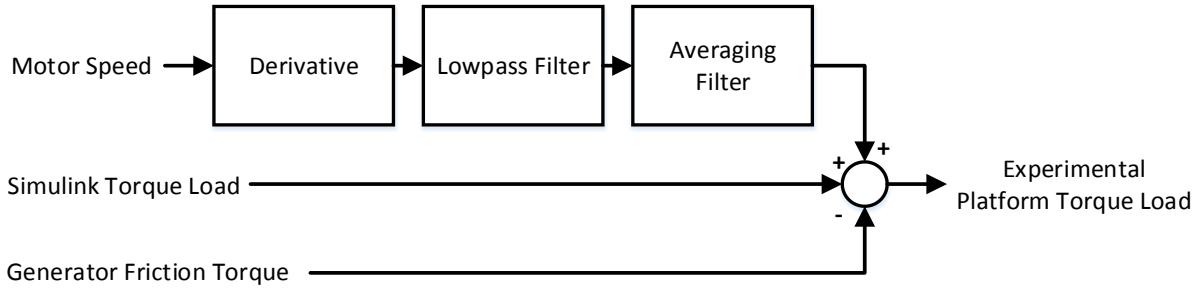


Figure M-1: Block Diagram for Torque Correction and Acceleration Matching Code in the Experimental Platform Microcontroller

Figure M-1 shows the net torque reduction algorithm running on the experimental platform microcontroller. The derivative of the motor speed, or motor acceleration is sampled every 2 milliseconds. It is filtered with a first order lowpass filter with a cutoff frequency of 100 Hz and a 4 sample backwards averaging filter. The acceleration, which is proportional to the net torque, is scaled and added to the Simulink torque load corresponding to the current vehicle condition. The total torque load in the experimental platform motor is therefore scaled up, and the net torque is correspondingly reduced. Friction torque from the generator coupled to the motor is subtracted from the final torque command to avoid torque distortion from this extra opposition torque source.

N. *Detailed I^2C Design*

I^2C communication is used to send the disturbance commands to the left and right current sources. Once the disturbance torques that are calculated from the dynamic model are summed they are scaled to 12-bits and communicated to the digital-to-analog converters (DAC) via the fast mode write command protocol defined in the DAC datasheet. The ATmega128 datasheet provides driver software for toggling the clock and data lines of the I^2C interface. By using the drivers provided by Atmel in conjunction with the

protocol defined in the DAC datasheet 12-bit values can be communicated to the DACs at a speed of 100 kHz. The microcontroller needs to be set to output on the clock and data lines at that speed so the TWPS and TWBR registers need to be modified in such a way that the I^2C clock line is toggled at 100 kHz. By utilizing Eq. [N-1] the values of the registers can be determined. By setting TWBR to 72 and TWPS to 0 an f_{scl} of 100 kHz can be achieved. The oscillator frequency on the ATmega128 is 16 MHz. It is important to note that the I^2C communication happens inside the interrupt for the experimental platform; furthermore, because there are two I^2C communication cycles that need to occur if they were to be done back-to-back then the execution time for the interrupt would exceed 1 millisecond. In order to avoid this timing issue the I^2C communication cycles are split and run on alternating interrupts. This change reduces the execution time of the interrupt by approximately 300 microseconds. This massive difference is due to the fact that the I^2C hardware execution time is approximately 300 microseconds and the I^2C software execution time is only about 20 microseconds.

$$f_{scl} = \frac{f_{osc}}{16 + (2 * TWBR) * 4^{TWPS}} \qquad [N-1]$$

O. *Detailed Serial Communication*

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) drivers provided by Atmel in the ATmega128 datasheet were imperative to establishing serial communication between MATLAB®. By using [O-1] the value of the UBRR register can be calculated in order to set the desired BAUD rate. A BAUD rate of 38.4 kbps is desired so the value of the UBRR register is calculated to be 25. It is also necessary to modify USART control registers in order to set the communication characters to 8-bits with a single start bit and a single stop bit. The BAUD rate of 38.4 kbps was decided upon because it is the highest possible BAUD rate that is compatible with both MATLAB® and the ATmega128 microcontroller.

$$UBRR = \frac{f_{osc}}{16 * BAUD} - 1 \qquad [O-1]$$

In order to automate the serial communication process a communication protocol had to be developed. Fig. O-1 depicts the communication protocol that was designed. It is important to note that the microcontroller must acknowledge commands from MATLAB® because the microcontroller software is interrupt based and therefore the serial communication cycle could be interrupted. The communication protocol defined in Fig. O-1 prevents any data from being lost due to that interruption. MATLAB® does not need to acknowledge any information being sent to it by the microcontroller because the MATLAB® software is never being interrupted.

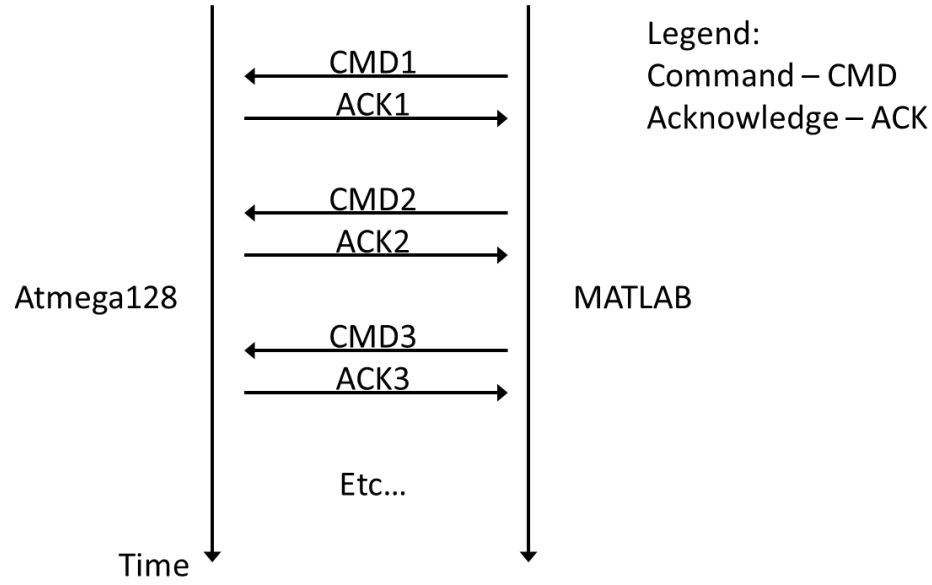


Figure O-1: Serial Communication Protocol for Microcontroller to GUI Communication

P. *Detailed Interrupt Timing*

The interrupt was designed to be 1 millisecond and was generated by using Time/Counter unit 0 in the ATmega128. The interrupt was required to be 1 millisecond in order to keep the controller gain a reasonable magnitude. By using [P-1] where N is the clock divider, X is the compare match value, T_i is the interrupt period, and f_{osc} is the microcontroller clock the interrupt time can be determined. By setting N equal to 64 and X equal to 250 by modifying the Timer/Counter unit 0 control registers a 1 millisecond interrupt is generated. It is important to note that in order for the interrupt to be called the compare match enable bit for Timer/Counter unit 0 must be set in the TIMSK register and global interrupts must be enabled.

The interrupt duration is between 600 microseconds and 900 microseconds. This variation is primarily due to the execution time of various calculations that may or may not be executed depending on the velocities and dynamic model parameters commanded by the user. In order for the interrupt software to fit in the 1 millisecond time required the dynamic model had to use a Taylor series expansion to perform the sine and cosine functions necessary to generate the opposition torques for the generators. It should also be noted that the left and right current sources had to be updated every other interrupt. It was necessary to update them this often in order for the torque matching controller to function properly; however, the I^2C hardware execution time is 300 microseconds whereas the software execution time is only 20 microseconds. This disparity requires that the left and right current sources be updated on alternating interrupts in order for the interrupt software to take less than 1 millisecond.

$$T_i = \frac{N}{16\ MHz} * X \qquad [P-1]$$

Q. *Detailed Controller Design*

$$G_P H(s) = \frac{0.0001606}{1.389 \cdot 10^{-6} s^2 + 0.001315 s + 0.001877} \quad [Q-1]$$
$$with\ poles\ at\ s = -1.43,\ -945.4\ rad/s$$

The vehicle plant shown in [Q-1] includes linearized motor characteristics as well as the vehicle inertia. The dominant pole is very close to DC. Thus, the main goal of controller development is to speed up the system response.

$$G_{PID}(s) = \frac{K_P s + K_I + K_D s^2}{s} = 19.5k\left(\frac{\frac{s}{19.5}+1}{s}\right) \quad [Q-2]$$

$$k = 500$$
$$K_P = 500$$
$$K_I = 9750$$
$$K_D = 0$$

Equation [Q-2] shows the Laplace domain model of the PID velocity feedback controller.

$$G_c(z) = k\left(\frac{1.01z - 0.9902}{z - 1}\right) \quad [Q-3]$$
$$where\ k = 500$$

Equation [Q-3] shows the discrete feedback controller converted with the Tustin method and pre-warped at 63.8 rad/s.

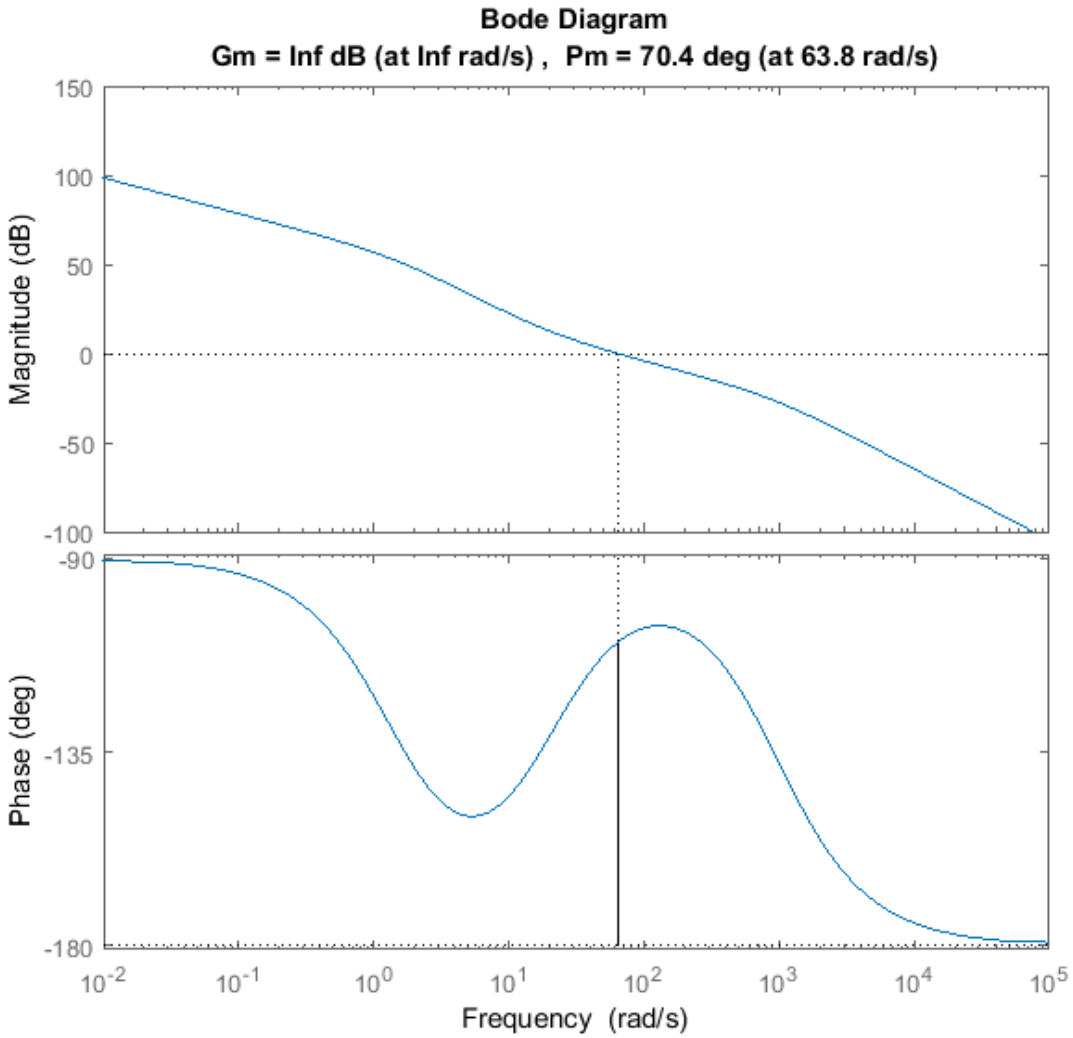


Figure Q-1: Compensated Frequency Response of the Open Loop Theoretical Vehicle

The vehicle controller mainly consists of an integrator to meet zero steady-state error for step commands and a lead compensator to add phase margin to the system near the crossover frequency. The proportional, integral, and derivative component gains can be seen in Equation Q-2, and the discrete controller can be seen in Equation Q-3. The derivative component of the controller is not needed as the system speed is sufficient enough to meet controller specifications without it. The overall gain was chosen as a tradeoff between meeting optimal phase margin in the system and avoiding constant saturation of the 10 bit PWM driving the motors.

The anti-windup system sets the integral component gain to zero if the PWM is saturated and the feedback error signal is still positive, meaning the system response is still lagging behind the issued command. The integral gain is restored once either of these conditions change.

R. *Detailed Experimental Results*

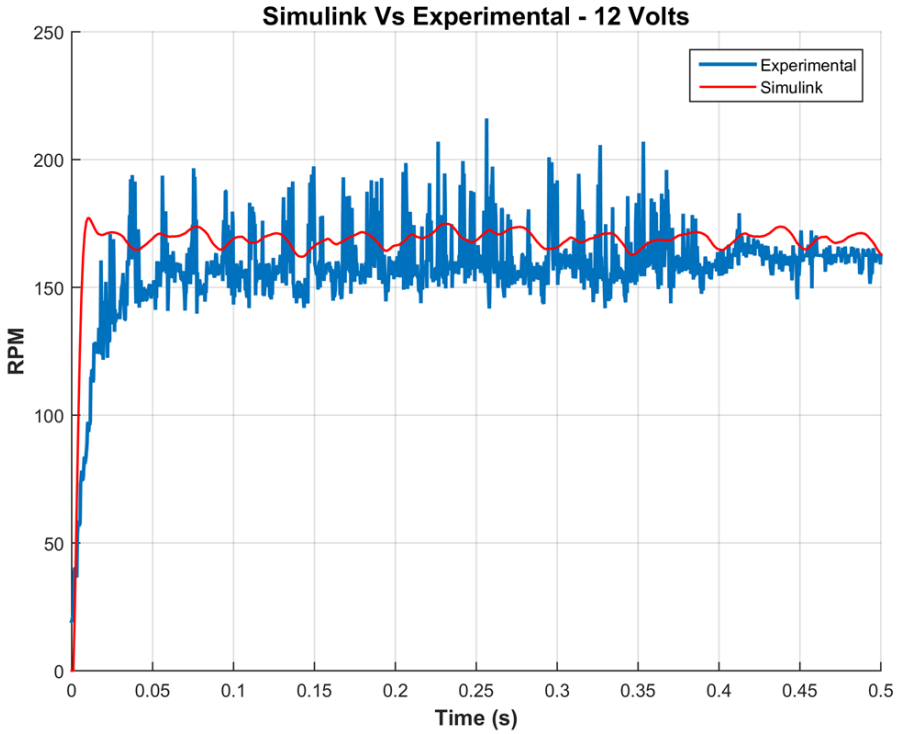


Fig. R-1. Simulink vs. Experimental Transient Responses at 12 volts

TABLE R-I. EXPERIMENTAL VS. SIMULINK TRANSIENT RESPONSE PERCENT ERROR

Voltage	Settling Time Percent Error	Overshoot Percent Error	Steady-State Error Percent Error
1	97.202	216.897	87.994
2	97.151	83.945	66.095
3	97.198	63.427	75.704
4	97.186	50.843	46.750
5	97.188	47.559	48.249
7	97.205	34.249	40.421
10	97.183	103.757	42.061
12	94.469	130.277	40.609
24	95.500	1321.972	42.602
Average Values =	**96.70%**	**228.10%**	**54.50%**

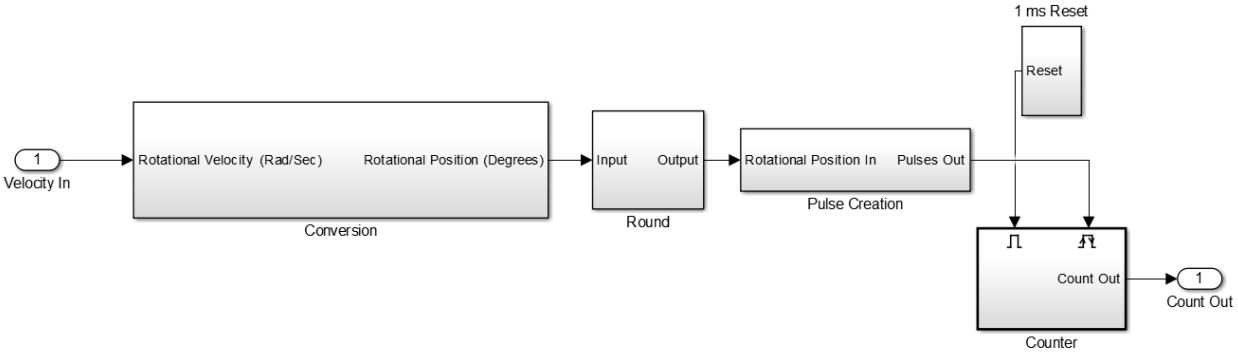


Fig. R-2. Rotary encoder model in Simulink

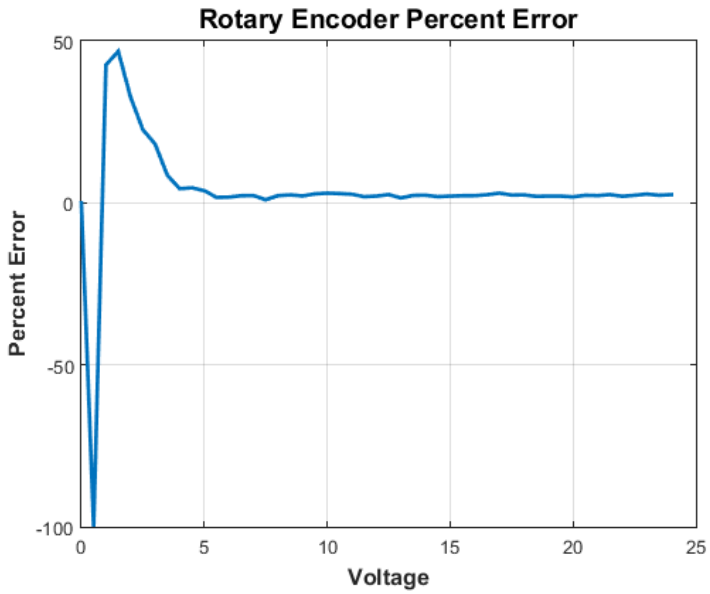


Fig. R-3. Rotary Encoder Error Plot

TABLE R-II. COGGING TORQUE PEAK TO PEAK PERCENT ERRORS

Voltage	Cogging Torque Percent Error
1	61.098
2	11.037
3	24.063
4	3.621
5	4.780
7	9.125
10	0.605
12	1.182
24	15.283
Average Percent Error =	**14.53%**

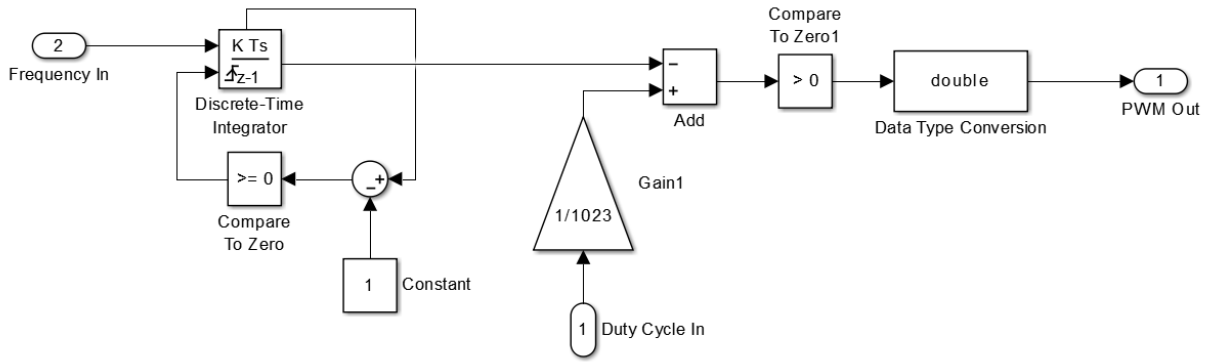


Fig. R-4. PWM Model in Simulink

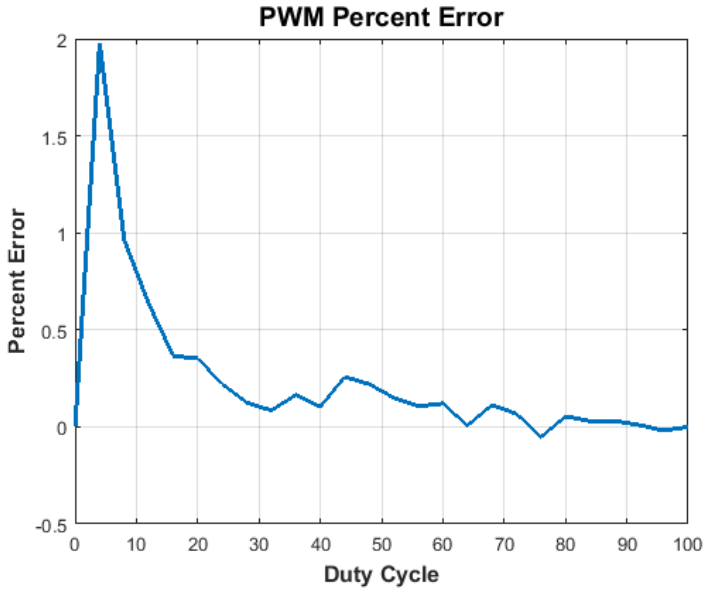


Fig. R-5. PWM Error Plot

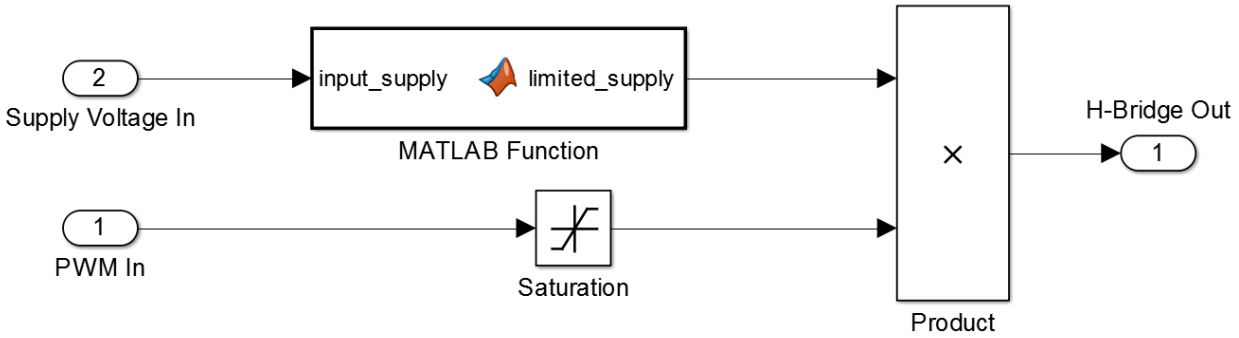


Fig. R-6. H-Bridge Model in Simulink

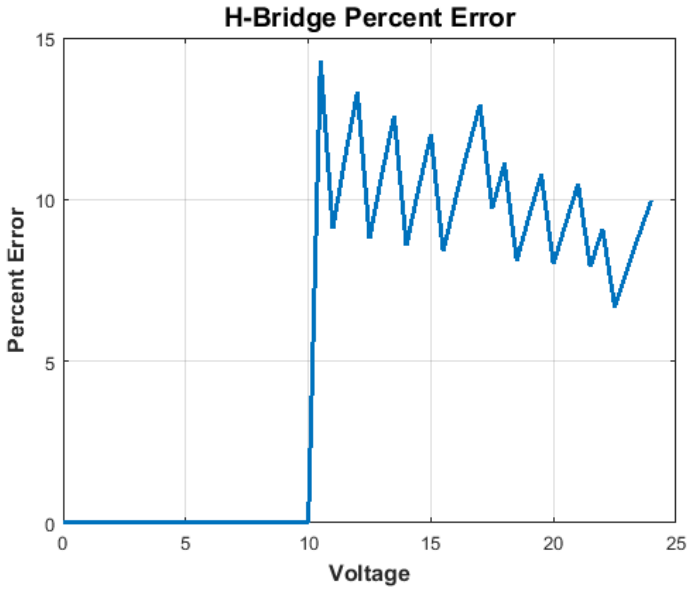


Fig. R-7. H-Bridge Error Plot

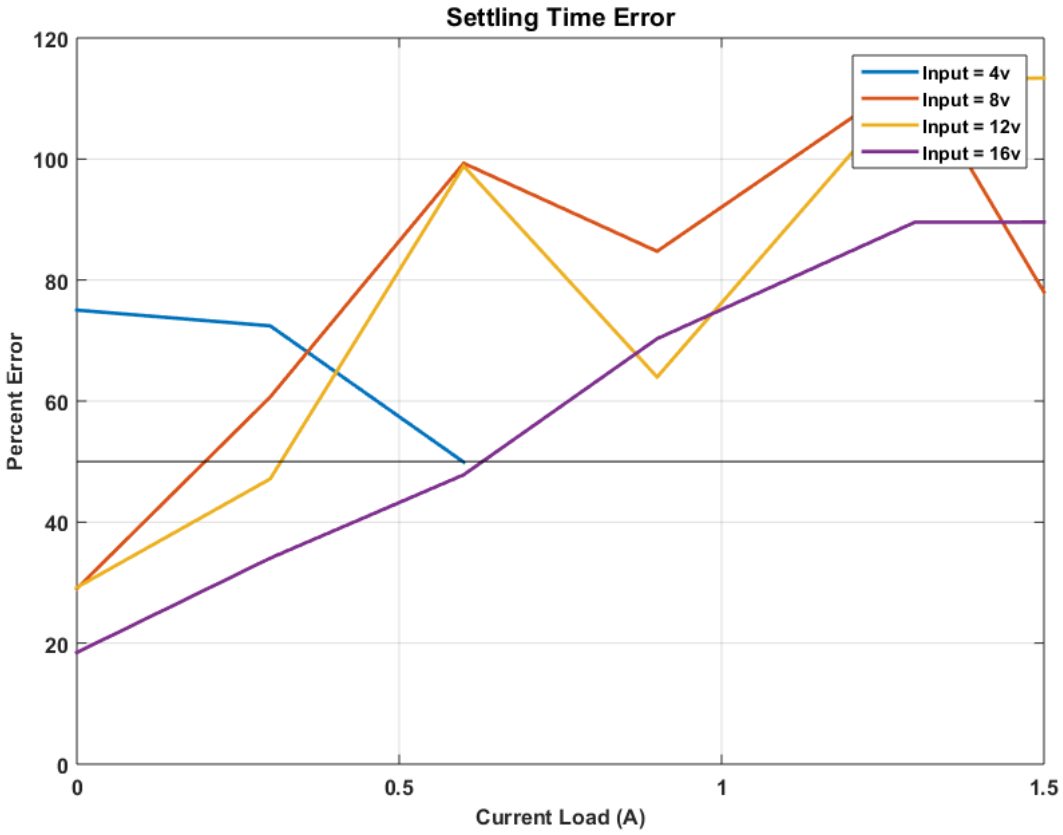


Fig. R-8. Settling Time Percent Error as a Function of Open Loop Input Voltage and Generator Current Load

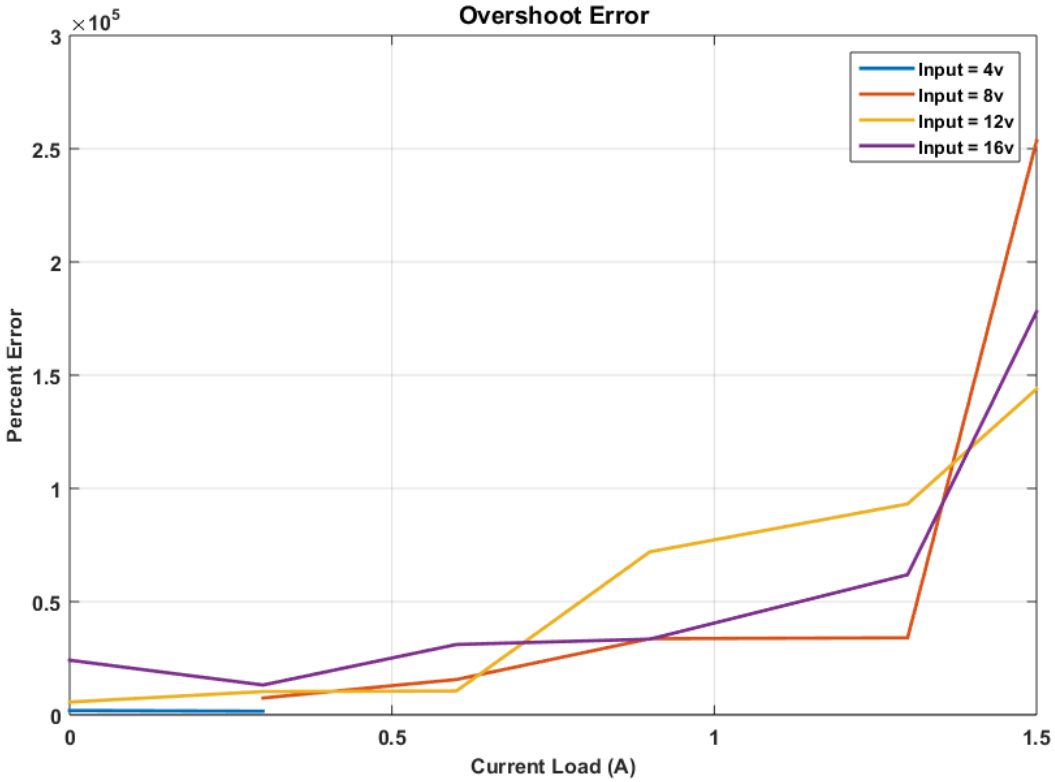


Fig. R-9. Overshoot Percent Error as a Function of Open Loop Input Voltage and Generator Current Load

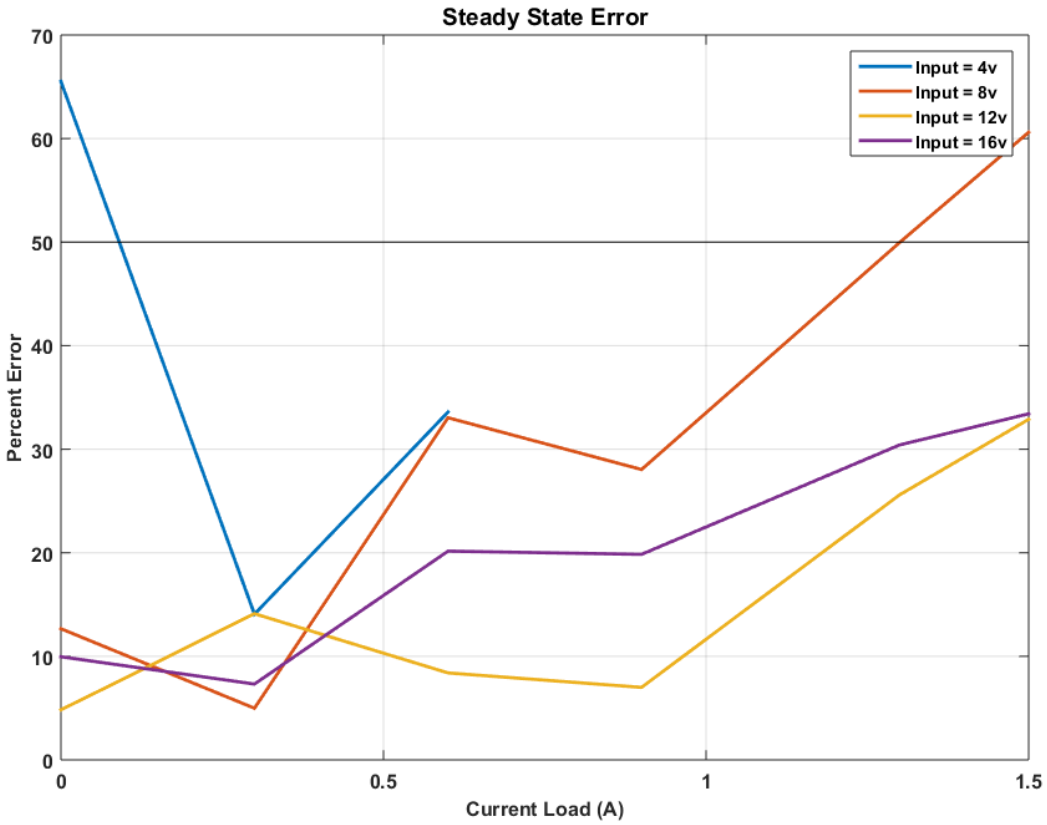


Fig. R-10. Steady-State Percent Error as a Function of Open Loop Input Voltage and Generator Current Load

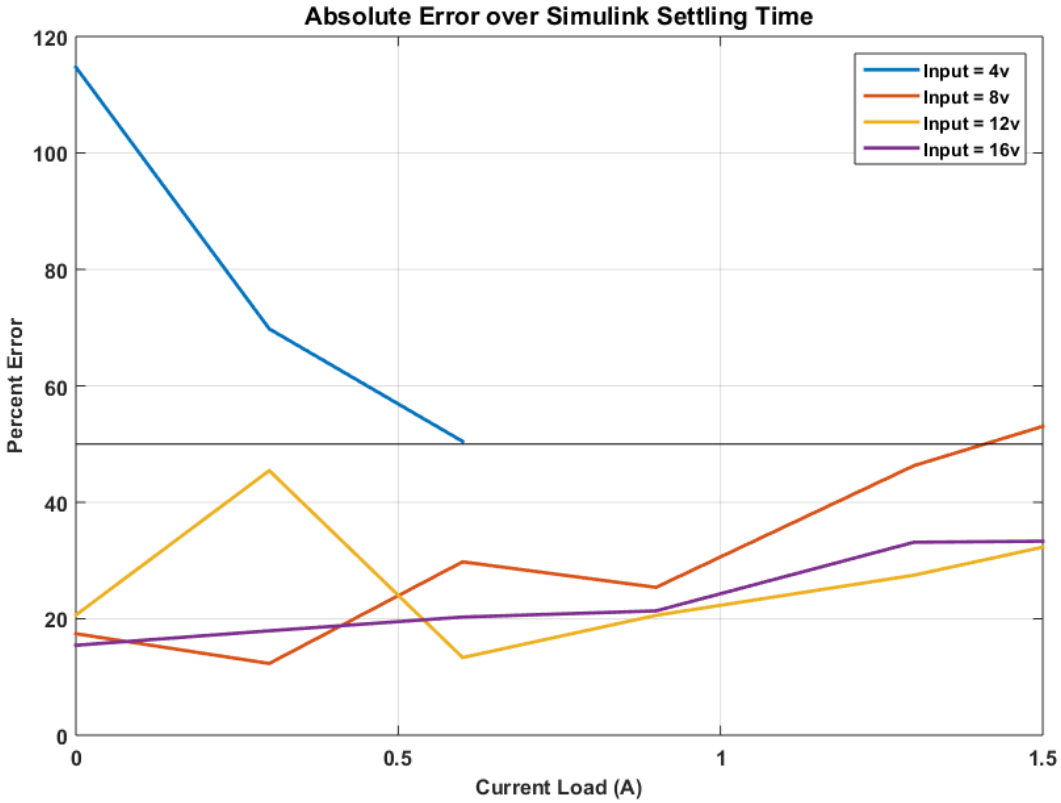


Fig. R-11. Absolute Percent Error as a Function of Open Loop Input Voltage and Generator Current Load

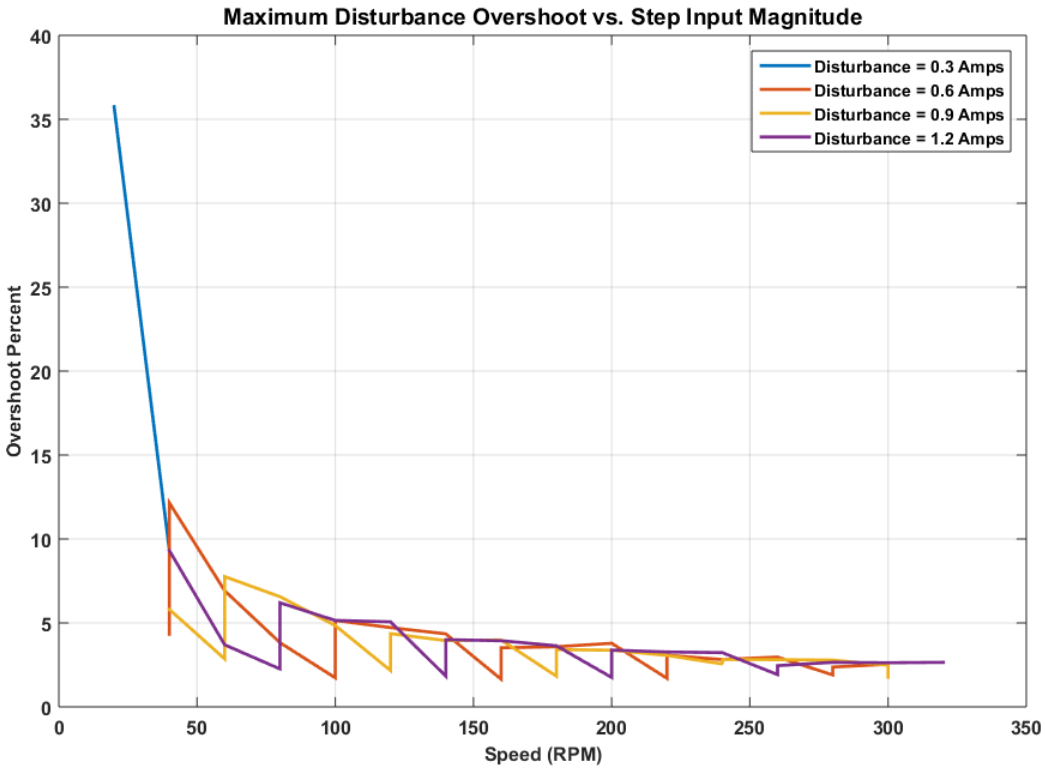


Fig. R-12. Disturbance Overshoot Percent as a Function of Steady-State Motor Speed and Disturbance Step Input Magnitude in the Simulink Model

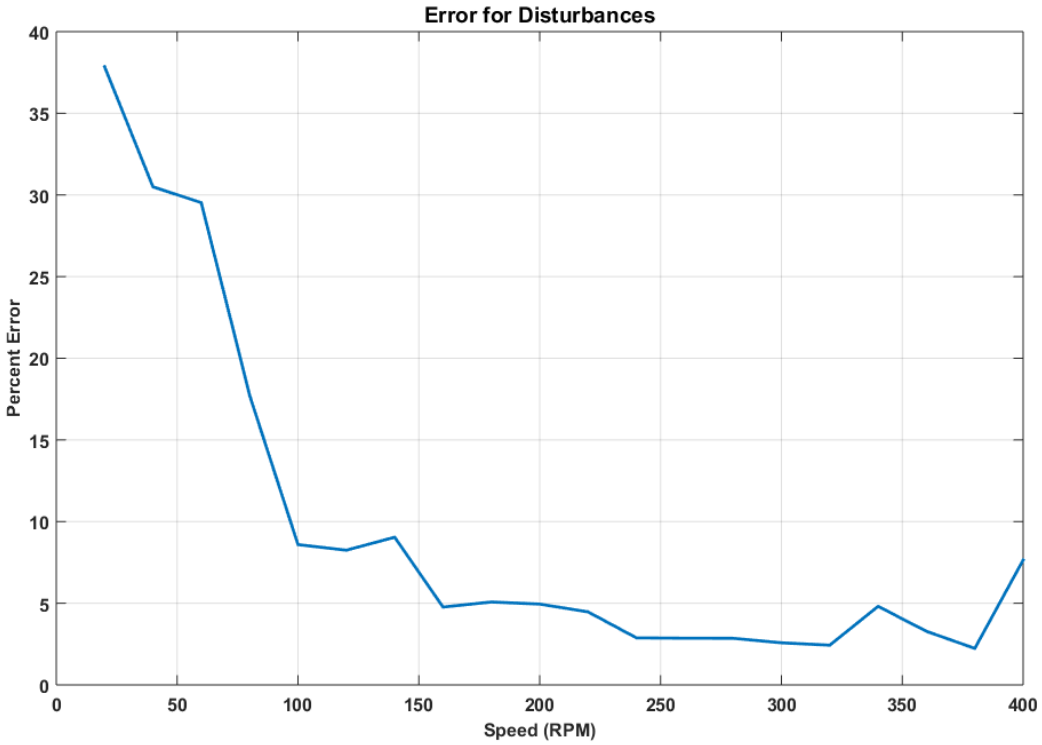


Fig. R-13. Disturbance Overshoot Percent as a Function of Steady-State Motor Speed in the Experimental Platform. The disturbance command is equal to 1.2 A, or the worst case.

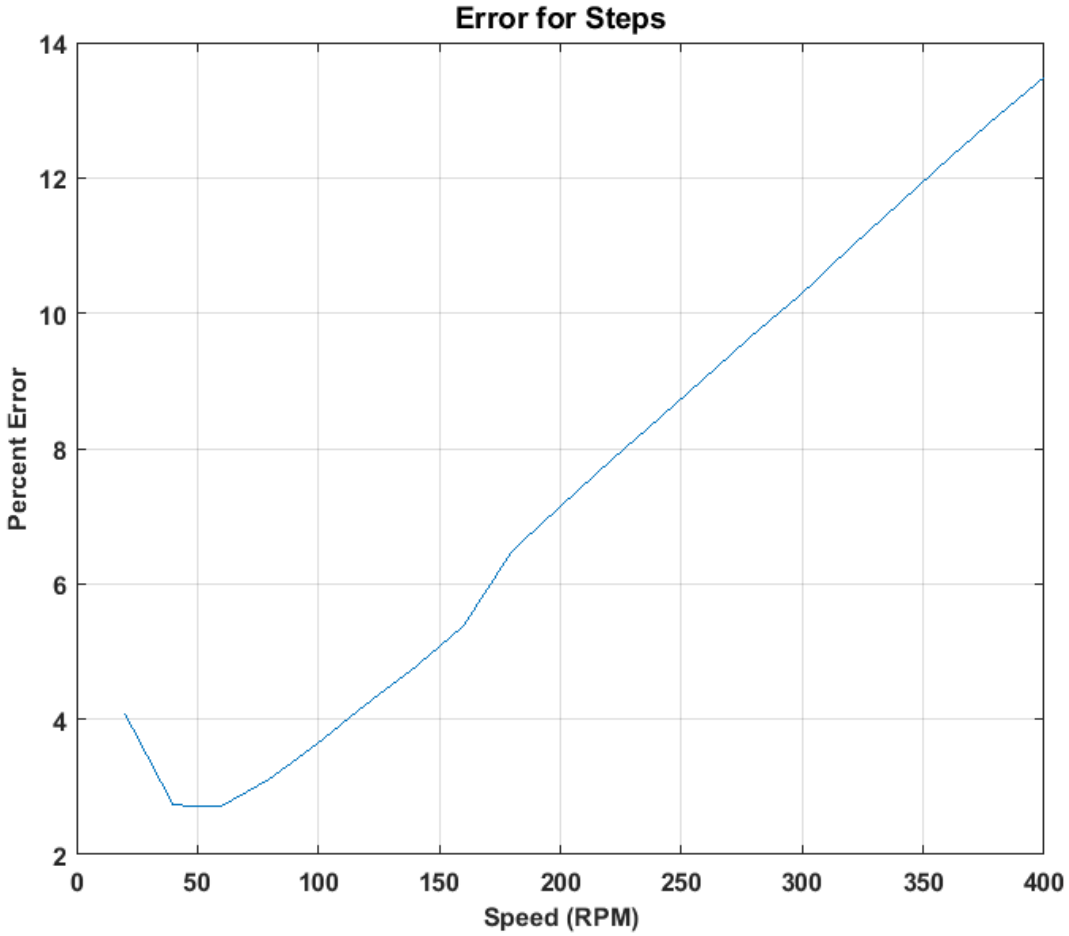


Fig. R-14. Step Tracking Percent Error as a Function of Step Input Magnitude in the Simulink Model

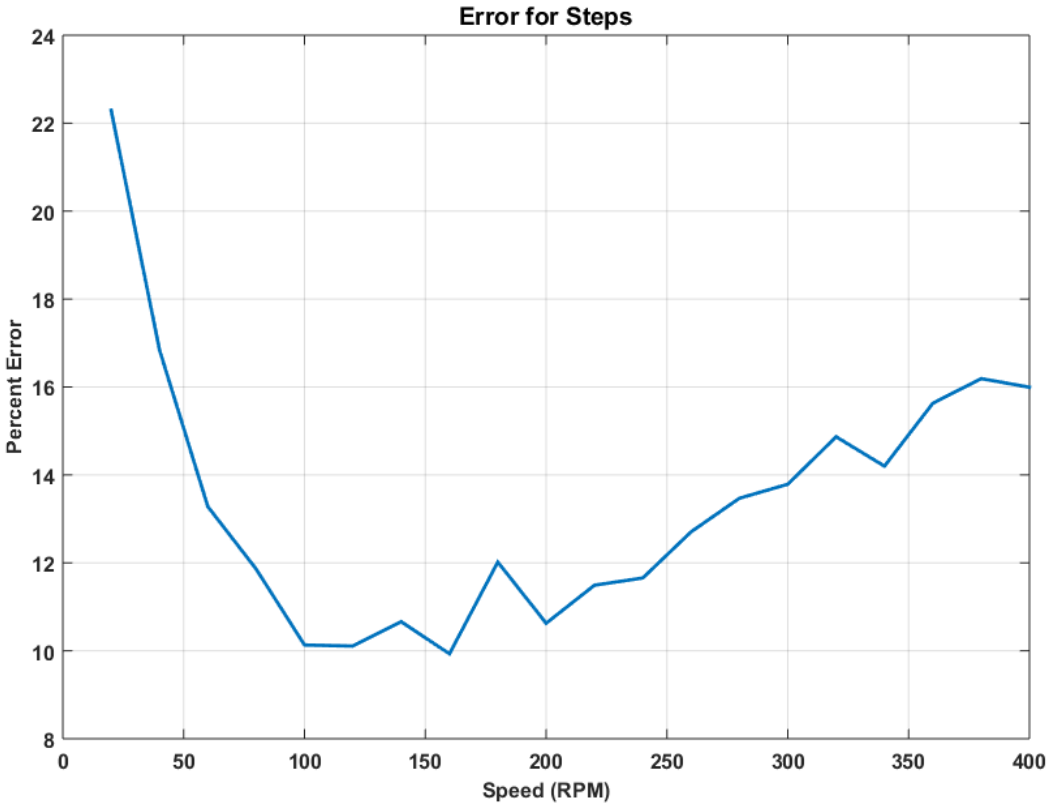


Fig. R-15. Step Tracking Percent Error as a Function of Step Input Magnitude in the Experimental Platform

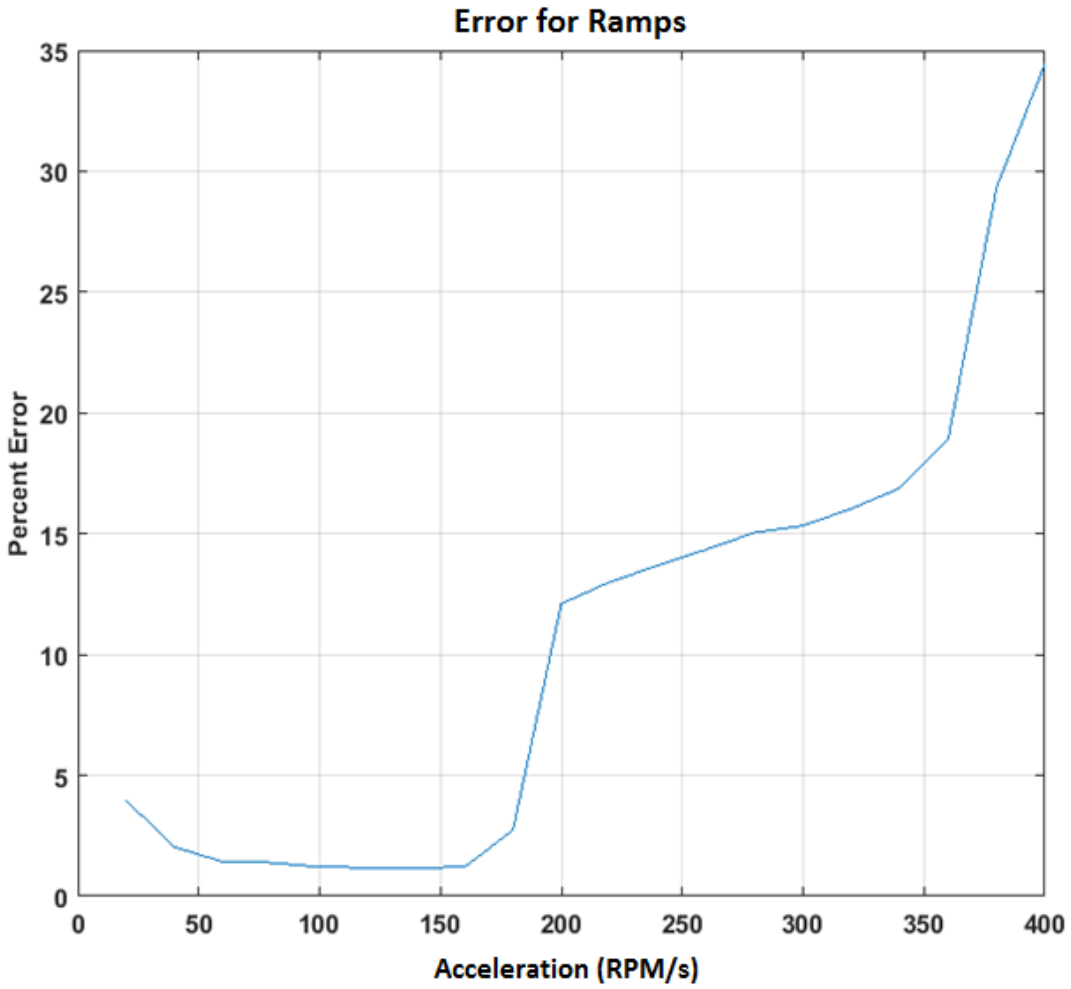


Fig. R-16. Ramp Tracking Percent Error as a Function of Ramp Input Magnitude in the Simulink Model

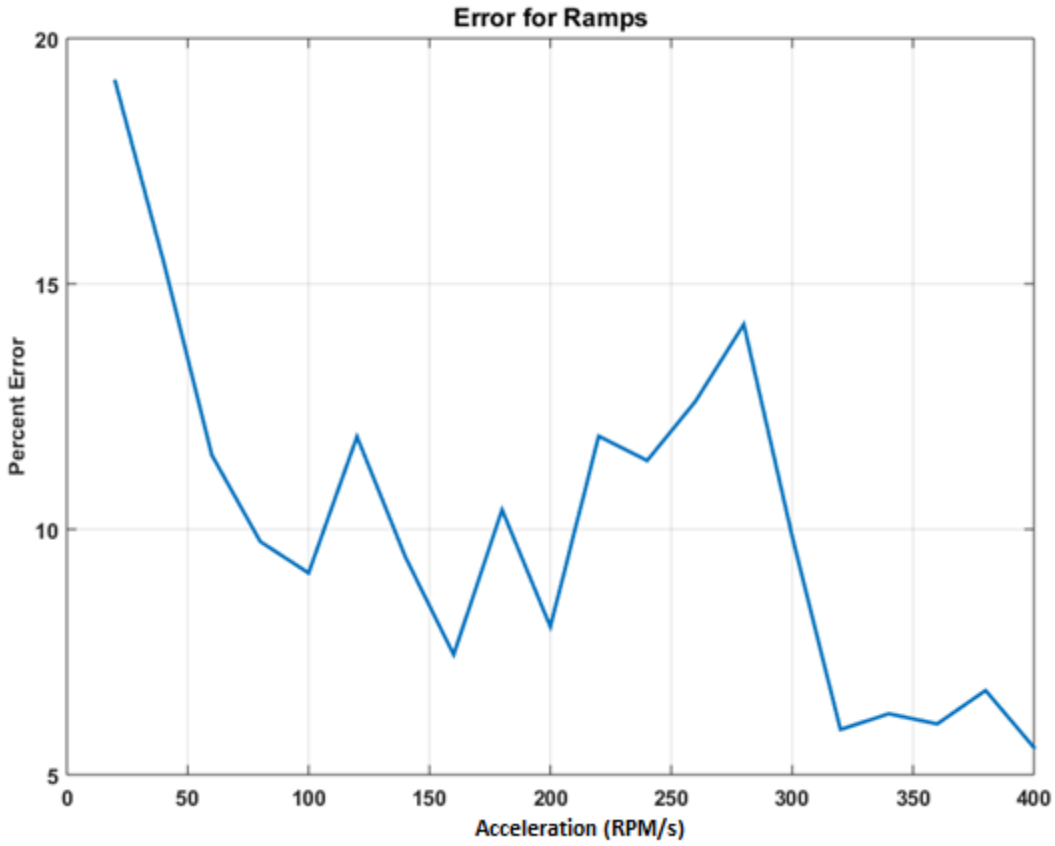


Fig. R-17. Ramp Tracking Percent Error as a Function of Ramp Input Magnitude in the Experimental Platform

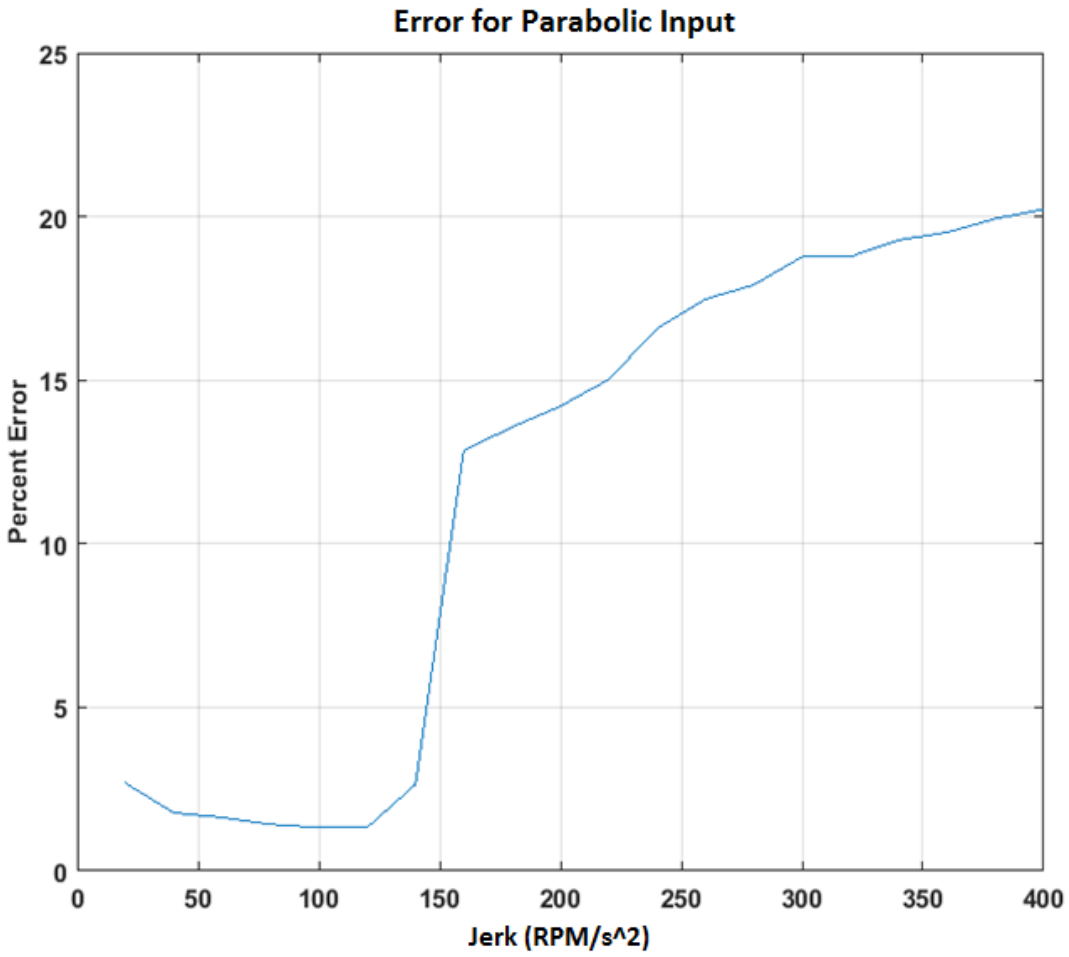


Fig. R-18. Parabolic Tracking Percent Error as a Function of Parabolic Input Magnitude in the Simulink Model

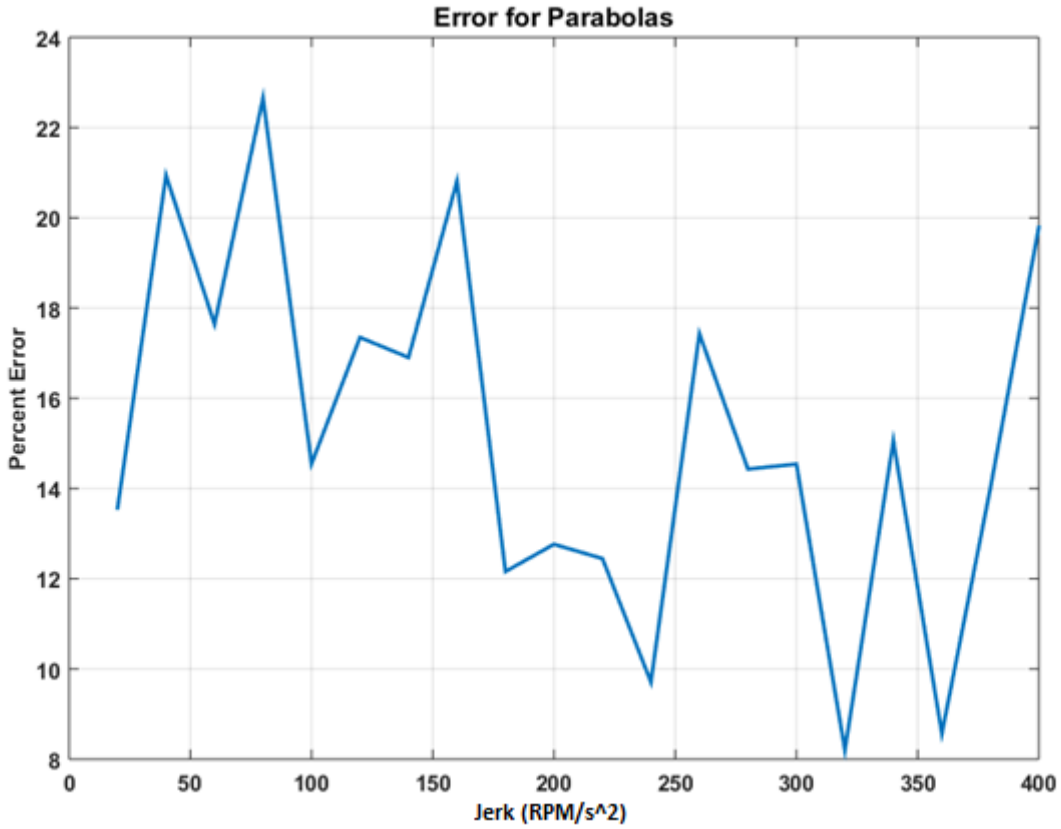


Fig. R-19. Parabolic Tracking Percent Error as a Function of Parabolic Input Magnitude in the Experimental Platform

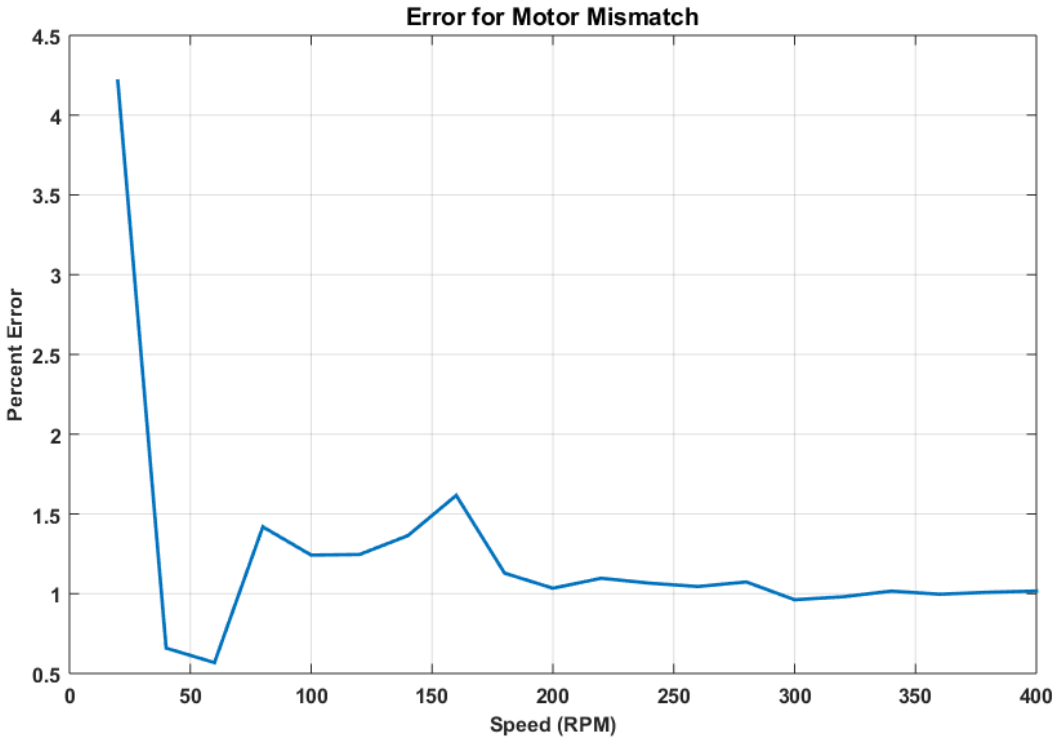


Fig. R-20. Motor Response Difference Percent as a Function of Step Input Commands.

S. *Video, Web Links*

Senior Project Website:
http://ee.bradley.edu/projects/proj2016/venom/
Run-through of the GUI for the Simulink system:
https://www.youtube.com/watch?v=vuGQLxFuk8A
Run-through of the GUI for the experimental platform:
https://www.youtube.com/watch?v=zawAYN9LUPA
Experimental Platform demonstration:
https://www.youtube.com/watch?v=ao5LDD65wgI

T. *Future Recommendations*

Future adaptations of the project could involve the following:

- Bidirectional current source for both theoretical vehicle acceleration and deceleration matching
- Bidirectional H-bridge and motor functionality to simulate braking and vehicle reversal
- Faster microcontroller for more accurate and complex control methods
- More modular controller code in both Simulink and Experimental Platform for easier testing of multiple controllers
- Closed-loop position control