ME 492

SENIOR PROJECT

Design and Implementation of A Dynamometer for Small DC Motors

Group E

Erkut Bahadır, Rabia Konuk

Ömer Faruk Seven

09 January 2023

Course Instructor | Assoc. Prof. Hasan Bedir

Abstract

This report discusses the implementation and testing process of a tabletop absorption-type dynamometer design dedicated to measuring the power characteristics of a DC motor with a rated power range of 0-50W and a torque range of 0.01-0.3 Nm. The report starts with defining a dynamometer and providing design limitations. Subsequently, the design configuration is explained, and the advantages and disadvantages of the selected design are listed. Moreover, the detail of purchased items and corresponding expenditure amounts are given. Then, critical activities carried out during the assembly process are highlighted. The following parts briefly explain the working mechanism and specifications of each mechanical or electrical component. On the other hand, written computer code is elaborated for the acquisition, assessment, and visualization of data from sensors. The results obtained from the testing are provided and discussed. Finally, potential application areas of the project are given an emphasis on possible future improvements is made.

Table of Contents

List of Figures

List of Tables

1. Introduction

A device that simultaneously measures a rotating object's torque and speed to calculate its power is known as a dynamometer. Two types of dynamometers exist: absorption dynamometers, as shown in *Figure 1*, and transmission dynamometers. [1]

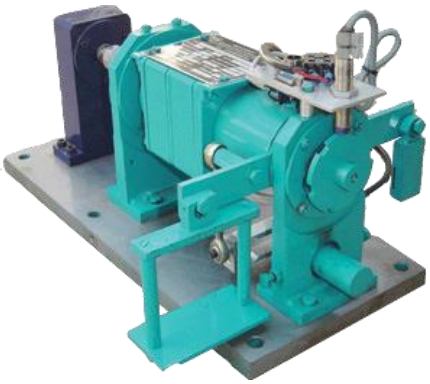


Figure 1: Example of absorption type dynamometer [2]

Absorption dynamometers measure torque through mechanical friction, fluid friction, or electromagnetic induction. In contrast, transmission dynamometers measure torque through the elastic twist of a shaft or by inserting a specific torque meter between shaft portions [1]. In various industries, dynamometers are utilized for measurement purposes, including the automotive industry and manufacturing. However, the acquisition of convenient and appropriate dynamometers and systems for testing small-scale motors can be both rare and expensive, leading to a waste of time and/or money for customers and designers in search of motors with specific specifications due to lack of information on the power and torque characteristics of small-scale motors. In order to address this problem, a tabletop absorption-type dynamometer, dedicated explicitly to measuring the power characteristics of DC motors with a rated power range of 0-50 W and a torque range of 0.05-0.3 Nm, was designed and implemented. During the design process, four significant goals were determined to be met. Firstly, ease of installation and storage is to be achieved by designing a tabletop dynamometer. Secondly, the design must allow for easy mounting of the testing motor, enabling the testing of various motors with ease. Thirdly, compatibility with various shaft diameters must be incorporated into the design, allowing for testing motors with different shaft diameters. Lastly, the design should allow for cost minimization in order for the feasibility of prototype construction. The cost of the dynamometer was determined to be kept below 3000₺, and the length and width were determined to be 60 cm and 45 cm, respectively. The torque range of the

test motors was determined to be 0-0.3 Nm. In literature, several similar studies, such as "Design and Implementation of a Small Electric Motor Dynamometer for Mechanical Engineering Undergraduate Laboratory" by Aaron Farley [3] and "Design of a Small Electric Motor Dynamometer" by William A. Black Jr [4], can be found and have been used as references. However, the approach taken in this project differs from those in the literature in terms of using a DC motor as a brake mechanism and utilizing an Arduino for data acquisition.

2. Design & Cost

This section involves overall design configuration and cost analysis, respectively. In the configuration part, a simple schematic of the overall design is provided to illustrate the working mechanism and make it more comprehensible, in addition to a pros and cons table in which the design's advantages and disadvantages are highlighted. Subsequently, another table is provided, listing all expenditure items in the cost analysis section.

2.1 Overall Design Configuration

The block diagram of the system is provided in *Appendix A.* As shown in the diagram, there are two power units, each dedicated to supplying energy to the test and brake motor, respectively. The speeds of the motors are controlled via PWM dc motor speed controllers, and the information about the electrical power applied to the motors is acquired with the help of power meters. There are two shafts, namely Shaft 1 and Shaft 2, extending from each motor. The shafts are connected with a coupler that works as an intermediary to transfer energy. The magnet that is attached to the coupler is utilized to measure the rotational speed of the test motor. The measurement is done by so using a linear hall effect sensor. The magnetic field originating from the magnet attached to the coupler is detected by the Hall effect sensor in a certain vicinity. Each time the magnet passes near the hall effect sensor, a signal is generated. Utilizing the time spanned between consecutive signals; the rotational speed is derived. The torque information of the test motor is obtained by utilizing a torque arm attached to the outer case of the brake motor. The force exerted at the tip of the torque arm is measured by a load cell, and then the signals are transferred to the microcontroller. The signals acquired both from the hall effect sensor and the load cell is interpreted and transferred to the MATLAB interface set up on a PC. Then, the measured rpm, torque, and power values of the test motor are plotted in MATLAB and displayed on the PC panel.

The advantages and disadvantages of the overall design configuration are listed in *Table 1.*

Table 1: Pros and cons of the selected design

PROS	CONS
Data acquisition via Arduino is viable.	The setup is harder to build.
Generating rpm/torque/power plots on PC is possible.	The measurement is less precise.
Power, rpm, and torque values are displayed all together on a single OLED screen.	
It takes up less space due to the sizes of the load cell and the Hall effect sensor	
It is cheaper due to the price of sensors	
The signals sent out by sensors are sufficiently clear.	

2.2 Cost

A market analysis was conducted in order to determine the most cost-effective parts that meet the design criteria for the overall system. Aluminum sigma profiles were ordered from a local manufacturer that provides additional after-sale services such as cutting. The purchased profiles were cut in pre-selected lengths by the manufacturer for free. On the other hand, the DC motors, both brake, and test, were purchased from a second-hand store in Karaköy, which sells mechanic parts at substantially low prices compared to brand-new motors with comparable specs. In addition, medium-density fiberboard (MDF) as a support base was cut in desired shape and dimensions and supplied for free by a furniture manufacturer that we have acquaintance with. Also, the parts at our disposal, such as the microcontroller and adjustable DC power supply, were utilized to minimize the cost. All the other parts, including the PWM controller, hall effect sensor, load cell, multimeter, coupling (drill chuck), shaft adapters, bolts, screws, nuts, washers, and constant DC power supply, were purchased from relevant electronics and mechanics stores online. In the attempt to manage the cost, equipment presents in the lab and at our disposal was used for various applications such as milling, cutting, drilling, sanding, soldering, etc. So, no money was spent on any application whatsoever.

All expenditure items are listed in *Table 2.*

Table 2. Cost Analysis

Part List	Purchase Detail	
	Piece (#)	**Cost (৳)**
Brake Motor	1	450
Test Motor	1	250
Constant DC Power Supply	1	260
Load Cell	1	155
Multimeter	1	65
Neodymium Magnet	1	10
Hall Effect Sensor	1	25
PWM Controller	1	85
Cables	1	90
Cable Connectors	1	100
Drill Chuck	1	90
Shaft Adapters	1	120
Mounted Bearing	2	210
Motor Holder	1	80
Torque Arm	1	70
Sigma Profile	1.2 (m)	170
Corner Bracket	10	80
T-slot Nut	22	70
Bolt	10	60
Screw	30	70
Washer	24	50
MDF Wood (18mm)	0.27 (m^2)	N/A
Adjustable Power Supply	1	N/A
Arduino Uno	1	N/A
	Total Cost	**2560**

3. Implementation

This section involves step by step explanation of the assembly process and elaboration on the overall system outlook, mechanical/electronic components that make up the system, and data acquisition method.

3.1 Assembly

Overall assembly of the system is supported by the frame. The frame consists of 6 aluminum sigma profiles which are connected to each other via corner brackets and T-slot nuts. Both of these connection elements are fully compatible with the selected type of sigma profile. While connecting these elements together, it was important to obtain perfect alignment to prevent undesired vibration and wobbling of the system, which could damage the system itself and the integrity of measurements. Therefore, the frame parts are connected to each other on a flat

surface. A set of M6 bolts are used and tightened gradually respectively to prevent tilting of the frame while tightening the bolts. The general frame structure can be seen in *Appendix B.* Once the frame parts are fastened, and alignment is acquired, the bearings are connected to the framework. The selected bearings from previous design steps needed to be changed during the final assembly process since the shaft diameters of the brake motor were different on both sides. The bearings with housings are manufactured within standards and have different housing dimensions with respect to the bore diameter of the bearing. The usage of bearings with different housing dimensions would create a vertical alignment problem, resulting in the brake motor shafts being non-parallel to the ground. Furthermore, the shaft of the brake motor, which is connected to the test motor shaft via coupling, does not have enough length to support both bearing and coupling (drill chuck) adapters simultaneously. As a result, we decided to mount the bearing over the coupling adapter. Thus, the length of the shaft is used more efficiently, providing enough space for coupling the adapter and bearing at the same time. This created a different situation from previous design steps; we used 15 mm bore diameter bearings with housings on both sides of the brake motor. The adapters also have different outer diameters; as a solution, we applied the lathing process to obtain two adapters with 15 mm outer diameters, as can be seen in *Figure 2.* Both the adapters and bearings have set screws to hold the shafts. We created several notches on the surface of the motor shafts to fix the coupling adapters and used the notches on the surface of the adapters to fix the bearings again with set screws.

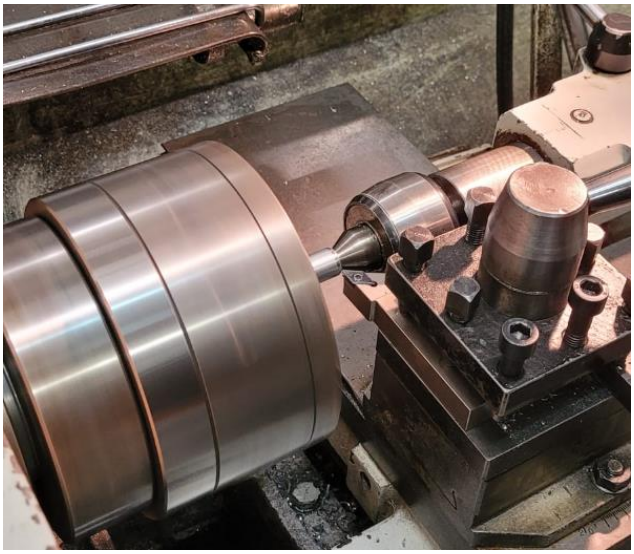


Figure 2: Lathing operation of coupling adapters

As mentioned earlier, we used a drill chuck as coupling. The brake motor side of the drill chuck is mounted by coupling adapter with threads. The location of the load cell and the direction of

the rotation is designed such that it provides self-tightening condition. The adjustable side of the drill is connected to the test motor. Thus, test motors with different shaft diameters become compatible with the system. Undesired slipping motion of the shaft is prohibited by the triple-jaw structure of the adjustable drill chuck.

There are two components that we had to design and get manufactured: The test motor holder and the torque arm. SolidWorks 2021 is used as CAD software. Both components are produced from sheet metal with a 3 mm thickness. The dimensions of the test motor holder can be seen in *Appendix C*. The motor holder is attached to the frame with 2 x M6 bolts and T-slot nuts. Due to T-slot nuts, the subassembly could be shifted and fixed at the desired position. At the upper part of the holder, the slot for the test motor is placed. The slots are designed to provide the ability of vertical movement, so the user can slide the test motor up-down and avoid vertical misalignments between the test and brake motors. Likewise, the bolt holes are designed with 2 mm of diameter and 2 mm of vertical sliding slots. However, there had been slight manufacturing issues resulting in misaligned bolt holes. Also, the casing of the brake motor mildly interfered with the framework. In return, we had to raise the motor assembly from the frame to ensure that the interference was eliminated, using a set of washers as spacers, providing 3 mm of elevation. This raising process strengthened the motor holder bolt-hole misalignment. To compensate, we had to revise the slots of bolt holes via milling, as shown in *Figure 3*.

Figure 3: Milling process of the motor holder

The torque arm is the other special part of this project. Unlike the motor holder, the torque arm did not require further machining process after delivery. The design of the torque arm has changed since another brake motor is used within the project. The dimensions and geometry of

the torque arm can be found in *Appendix D.* The effective length of the torque arm was designed to be 0.18 m for the calculation of the torque by multiplying the force. However, slight deformations on the torque arm needle due to manufacturing processes caused nearly 4,5 mm of deviation; thus, the effective length of the torque arm became 0,1845 m and was used within MATLAB. The torque arm should be leveled with respect to the ground for the integrity of the calculations. Any diversions of the angle between the torque arm and the ground would cause the effective length of the torque arm and, as a result, inaccuracies in torque calculations. While attaching the torque arm to the brake motor casing with 2 x 4 mm diameter bolts, we checked with a water gauge to determine whether the torque arm was positioned correctly.

The load cell is calibrated and used for force calculation applied by the torque arm. The torque arm and load cell combination are shown in *Figure 4.*

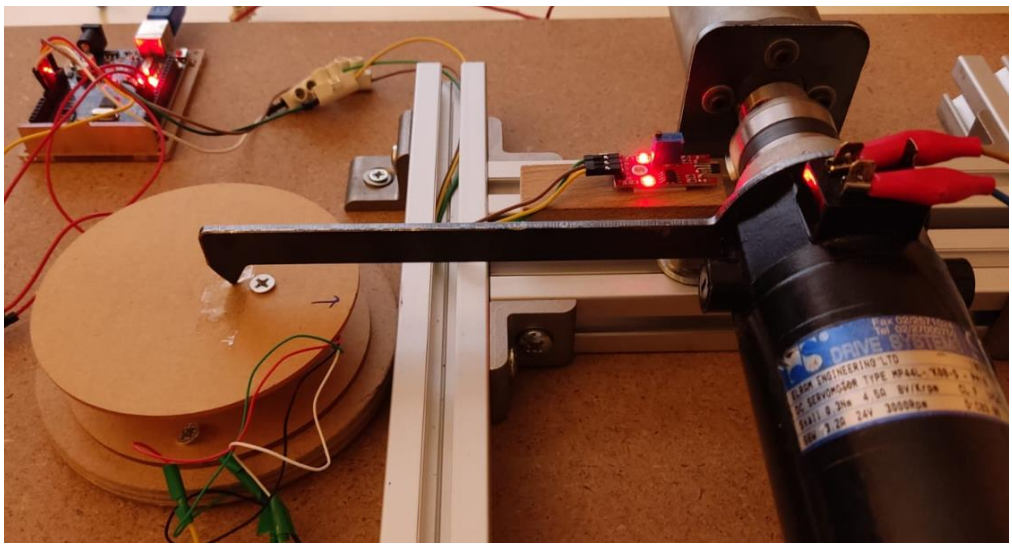


Figure 4: Torque arm, load cell structure, and magnetic hall effect sensor

The load cell is delivered with its own structure, flat plates at the top and below, and a strain gauge is located between them. While designing the overall system, we considered the height of the load cell structure. However, a mandatory raising process of the motor assembly from the framework resulted in increasing the height of the load cell structure. To obtain the correct height and level of the torque arm, we used four wooden discs placed under the load cell structure, providing 12 mm of elevation in total.

The magnetic hall effect sensor can again be seen in *Figure 4,* indicated with a white elliptical. A magnetic hall effect sensor is used to measure the current RPM of the combined shafts of the brake and test motors. A neodymium magnet is attached to the coupling for the hall effect sensor

to detect each time the magnet passes. The sensor board is placed on top of a block between two motors to achieve the height required for healthy measurement conditions.

To power the brake motor, a constant power supply is used and adjusted for 12 V output voltage, as can be seen in *Figure 6*, indicated with number 6. 12 V output voltage is transferred to the brake motor via a PWM controller and digital multimeter, illustrated in *Figure 5*.

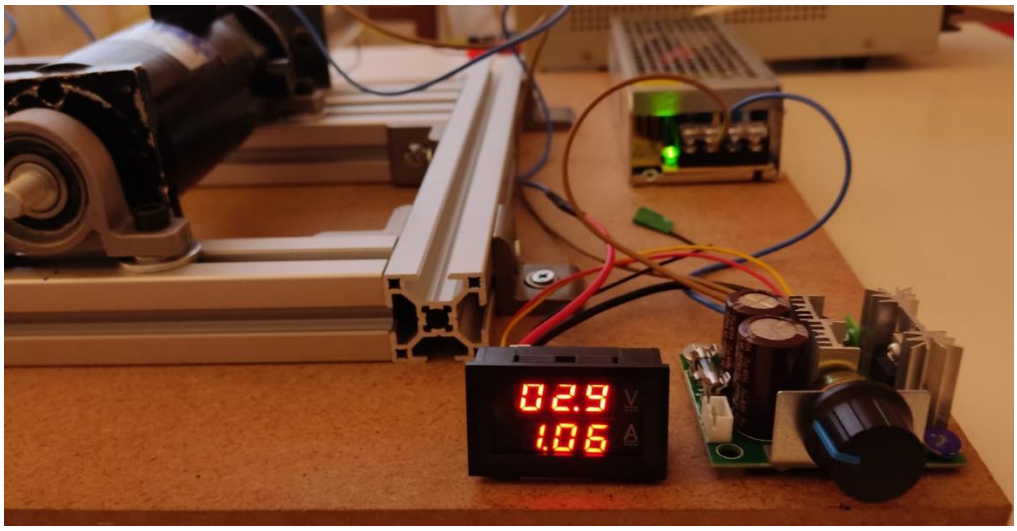


Figure 5: PWM controller in conjunction with a digital multimeter

The PWM controller allows the user to transfer energy to the brake motor at the desired level with a knob, thereby breaking the test motor gradually. A digital multimeter is used to check the current voltage and current values that appear on the brake motor and be able to avoid the high currents that can damage the wiring harness and the motor itself while braking.

An adjustable power supply is used to energize the test motor for given voltage levels, depicted as number 7 in *Figure 6.* An adjustable power supply ensures that a variety of test motors can be powered with the system, and different power levels can be used within the same test motor to see the effect of the input voltage on operating characteristics. The whole system is attached to a support base of MDF wooden board to ensure that the system is rigid, and the tabletop design is portable.

3.2 Overall System Outlook

The final state of the overall system after the completion of the assembly phase is shown in *Figure 6.* For additional pictures of the overall system, please refer to *Appendix E.*

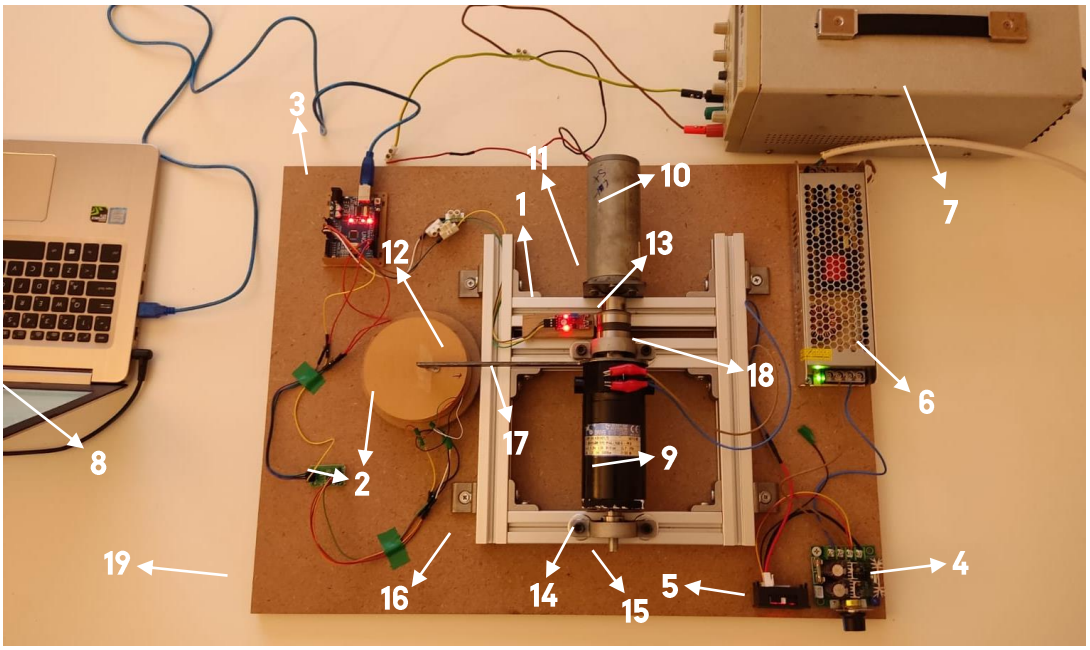


Figure 6: Overall System Outlook

In *Figure 6*, the key components that make up the system are represented by numbers and are listed and explained below:

1. *Hall Effect Sensor:* A Hall effect sensor is a type of sensor that utilizes the Hall effect to detect the presence and magnitude of a magnetic field. The Hall sensor's output voltage is proportional to the intensity of the field. The phenomenon is named after an American scientist, Edwin Hall. The working principle is the following: A current is applied to a thin strip of metal in a Hall sensor. In the presence of a magnetic field perpendicular to the direction of the current, the charge carriers are deflected by the Lorentz force – resulting in a difference in voltage between the two sides of the strip [5]. There are several ways of measuring the rotational speed of a shaft, such as using an IR optocoupler, IR line tracking sensor, and industrial tachometer; however, the Hall effect sensor is the most optimum way due to budget concerns and minimum signal noise.
2. *Load Cell:* A load cell is a force transducer that converts a force such as tension, compression, pressure, or torque into an electrical signal that can be measured and standardized. As the force applied to the load cell increases, the electrical signal changes

proportionally [6]. The load cell used in this project has a maximum load capacity of 5 kg. The maximum load capacity of the cell is sufficient since the maximum force exerted on the cell is around 0.5 kg. In addition, the cell has a percentage accuracy of 0.5. The load cell is also integrated with a signal amplifier, HX711 so that signals can be read from Arduino. Although there are several ways of measuring the torque of a shaft, such as using a torque meter, due to budget concerns and high compatibility with Arduino, a load cell was implemented for the measurement.

3. *Microcontroller:* A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. The microcontroller used in the project that is responsible for data collection and data process is an Arduino Uno. The reason why an Arduino was selected is due to its high compatibility with sensors, user-friendly interface, and simplicity of its programming language.
4. *PWM Controller:* PWM, pulse-width modulation, is a technique of breaking an electrical signal effectively into discrete parts to reduce the average power delivered by the signal. By turning the switch between supply and load 'on' and 'off' at a steady rate, the average value of voltage fed to the load is controlled. The shorter the switch is 'on' compared to 'off' periods, the lower the total power supplied to the load and vice versa [7]. The module selected for the project has an operating voltage of 12-40 V and an output current of up to 10 A, which is sufficient for the brake motor used in this project. Although there are several ways of controlling the rotational speed of a DC motor, the most effective and precise way to do this is by using the PWM method.
5. *Multimeter:* A multimeter is a tool used to detect the amount of electrical current and voltage difference in a circuit. The digital multimeter being used in this project is able to measure voltage up to 30V and current up to 10A and was connected to the PWM controller output to measure the current and voltage of the power being supplied to the brake motor.
6. *Constant Power Supply:* A power supply is an electrical device that supplies electric power to an electrical load. The main purpose of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. The constant power supply selected for the project is a DC power supply, which converts an AC input voltage to a DC output voltage. It is capable of supplying up to 360W of electrical power at 24V & 15A and was deployed to supply energy to the brake motor.
7. *Adjustable Power Supply:* The adjustable power supply selected for the project is a DC power supply with an integrated potentiometer to control the output voltage and current. It is capable of supplying electric power up to 300W at 30V & 10A.

8. *PC:* Data collected from sensors are transferred to the PC for data assessment and visualization. Acquired data is assessed in MATLAB environment through some form of mathematical functions and physics equations to generate meaningful results such as torque, mechanical power, and rpm. Generated results are plotted live on a graph, and the resulting chart is displayed on the PC screen.
9. *Brake Motor:* The motor that is used to brake the rotating shaft through reverse rotation is referred to as "brake motor" throughout the report. The brake motor selected for the project is a DC servo motor with the following specs: maximum mechanical power output of 66W at 24V & 3.2A, a maximum rotational speed of 3000Rpm, and a stall torque of 0.3 Nm.
10. *Test Motor:* The motor to be tested is referred to as the "test motor" throughout the report. The test motor selected for the project is a permanent magnet brushed DC motor with the following specs: maximum mechanical power output of 50W at 40V, a maximum rotational speed of 3300Rpm, and a stall torque of 0.17 Nm.
11. *Motor Holder:* The motor holder used to fix the test motor in space was produced from iron sheet metal.
12. *Torque Arm:* The torque arm used to transfer the absorbed energy to the load cell was produced from iron sheet metal.
13. *Coupling:* A coupler is a link or rod transmitting power between two rotating mechanisms or a rotating part and a reciprocating part. Due to adjustability concerns with a range of shaft diameters, a drill chuck with an adjustable mouth of 3-13 mm was preferred as a coupling in this application.
14. *Bearing:* A bearing is a machine element that constrains relative motion to only the desired motion and reduces friction between moving parts. Mounted bearings with a bore diameter of 15 mm were used to prevent the translational motion of the brake motor while allowing rotational motion. The housing is made out of zinc, whereas the bearing is of chrome steel.
15. *Shaft Adapter:* Shaft adapters were used to adjust the thickness of the motor shafts to allow for an assembly with bearings and the drill chuck. The adapters are made out of steel.
16. *Sigma Profile:* Aluminum sigma profiles were used as a supporting structure to fix the mechanical system and also to allow for an easy assembly of bearings and the motor holder.
17. *Corner Bracket:* Aluminum corner brackets were used to fasten sigma profiles together.
18. *T-slot Nut:* T-slot nuts were slid into profile channels and were used to position brackets and bearings.
19. *Support Base:* Medium-density fibreboard was used as a support base to fix the overall system.

3.3 Code

This project involves the use of an Arduino microcontroller to collect real-time data from the motors' shaft and the load cell, which measure the motor's rotational speed (rpm) and the force it applies, respectively. The Arduino program reads the data from these sensors and sends it to a MATLAB program via a serial connection. The MATLAB program processes the data as it is received and plots it on a graph in real-time, with two y-axes, one for rpm and one for torque. The MATLAB program also calculates and plots the power output of the motor based on the rpm and force data. This allows for the analysis and visualization of the motor's performance in real time, enabling the identification of patterns, trends, and potential issues. While the relevant codes are presented in *Appendix F* and *Appendix* , the logic and working mechanisms of the codes are explained below.

The Arduino code is designed to read and report the rpm and force data from a KY-024 linear magnetic hall effect sensor and HX711 load cell connected to an Arduino Uno microcontroller. The load cell is used to measure the break motor's force applied to it, while the magnetic hall effect sensor is used to measure the rpm of a rotating object.

The code begins by including the necessary libraries, "HX711.h" and "Arduino_JSON.h". The "HX711.h" library is used to interface with the load cell, and the "Arduino_JSON.h" library is used to create and manipulate a JSON object to store and transmit the data.

Next, the code defines constants for the pins to which the load cell and Hall sensor are connected, as well as a calibration factor for the load cell. The code then creates variables for the HX711 scale object, Val1 and Val2 (which are used to store the sensor data), and an array for the sensor data. The code also creates variables for RPM calculation, including a float variable called "rps" and a volatile float variable called "rpm". The "pulses" variable is a volatile byte variable used to store the number of pulses from the Hall sensor, and "timeold" is an unsigned long variable used to store the previous time that the Hall sensor was triggered. The "s" variable is a float variable used to store the current time, and "refsig" is an int variable used to convert the analog signal from the Hall sensor to a digital signal. The "val" variable is a float variable used to store the digital value of the incoming analog signals, and "prev_val" is an int variable used to store the previous value of "val". The "t" and "cur_t" variables are volatile unsigned long variables used to store the current time and the previous time, respectively. The "counter" and "limit" variables are used to track the number of times that the current time is the same as the previous time and to set the rpm to 0 if the count exceeds a certain limit.

The code then defines a function called "rpm_calculator" to calculate the rpm based on the time between successive pulses from the Hall sensor. This function is triggered every time the Hall sensor sends a pulse (rising edge).

In the setup function, the code initializes serial communication and sets the Hall sensor pin to INPUT_PULLUP mode. It then initializes the scale object and tares it (resets it to 0), and attaches an interrupt to the Hall sensor pin to trigger the "rpm_calculator" function. The code also initializes the variables used in the RPM calculation and delays for 2 seconds.

In the main loop, the code creates a JSON object to store the data. It then reads the values from the load cell and Hall sensor and stores them in Val1 and Val2, respectively. The code then checks if the current time is the same as the previous time and increments the "counter" variable if it is. If the current time is not the same as the previous time, the "counter" variable is reset to 0. If the "counter" variable exceeds the defined "limit", the rpm is set to 0, which means the motors are stopped by the user. Finally, using the stored variables, the code adds the rpm and force data to the JSON object and converts it to a string for transmission. In this way, the Arduino code was completed, and the data could be presented to MATLAB in a format that MATLAB could read.

The MATLAB code is designed to work in conjunction with an Arduino program, which is responsible for collecting data from the motor and sending it to the MATLAB program over a serial connection.

The code begins by setting up a serial connection to the Arduino program and initializing several variables that will be used to store and plot the data. The code then enters a while loop, which will continue to run as long as the plot is active. Within the loop, the code reads a string of data from the serial connection and attempts to decode it using the jsondecode function. If the data cannot be decoded, it is displayed on the screen.

If the data can be decoded, it is stored in a variable called "data". The code then extracts two elements from the data variable, "rpm" and "force", which correspond to the rotational speed of the motor in revolutions per minute and the force applied to the motor, respectively. The code also increments two counter variables, "count_1" and "count_2", and stores the current elapsed time in two arrays, "time_1" and "time_2". The values of "rpm" and "force" are then stored in two additional arrays, "data_1" and "data_2", respectively.

The code then updates the plots on the screen to show the data collected so far. The x-axis of the plots is set to the elapsed time, while the y-axis of the first plot is set to the rpm of the motor, and the y-axis of the second plot is set to the torque applied to the motor. The code also sets the limits of the y-axes to fixed values, which are chosen based on the expected range of values for the rpm and torque data.

After updating the plots, the code waits for a short period of time before repeating the loop. This delay is controlled by a variable called "delay", which is set to a value of 0.01 seconds. The purpose of the delay is to give the plots a chance to update and to allow the Arduino program to send new data to the MATLAB program.

After the while loop ends, the code calculates the power output of the motor by multiplying the rpm data by the torque data and dividing it by 1000 to convert the result to watts. The power data is then plotted on a new graph, with the x-axis set to the elapsed time and the y-axis set to the power output of the motor. Finally, the code includes an optional line for users who want to save the collected data to a CSV file. Using this line, time, rpm, torque, and power values collected throughout the program can be presented to users as a CSV file after use.

Overall, the Arduino and MATLAB codes provided in *Appendix F* and *Appendix* , respectively, are designed to collect and analyze data from a small-size DC motor in real time, using a combination of an Arduino program and MATLAB. The Arduino code measures and calculates the rotational speed of the motors and the force applied by a brake motor utilizing a magnetic hall effect sensor and a load cell, respectively. On the other hand, the MATLAB code reads data from the Arduino program over a serial connection, processes the data to calculate additional quantities of interest, and plots the data in real time for easy visualization. The code also saves the data to a file for later analysis. Therefore this project demonstrates the ability to collect and analyze real-time data from sensors using an Arduino microcontroller and a MATLAB program. It provides a useful tool for monitoring and studying the performance of a small-scale DC motors.

4. Results

The project collects data from an Arduino microcontroller using a serial connection to a Matlab program. The Arduino program reads data from a hall sensor and load cell, creates a JSON object with the data, and sends it to the Matlab program via the serial port. The Matlab program decodes the data as a JSON object, extracts the rpm and force data, and stores it in variables. The data collection interval is determined by the speed of the main loop in the Arduino program and the delay variable in the Matlab program, which is currently set to 0.01 seconds or 100 milliseconds. This interval can be adjusted by modifying the delay variable in the Matlab code.

After we finished the project, we carried out two tests. Since we have only one test motor, we wanted to see the characteristics of the motor with two different input voltages. In the first test, we supplied the test motor with 12V.

Figure 7 shows the RPM and Torque values obtained during the test. The test started with the test motor running with 12V. Then, the input voltage of the brake motor was increased slowly until the stall condition was obtained when RPM = 0. After the stall condition, the input voltage of the brake motor had been decreased again gradually until the brake motor stopped. The maximum torque value is obtained when the stall condition is achieved as proposed, which is equal to 459.03 mN.m.

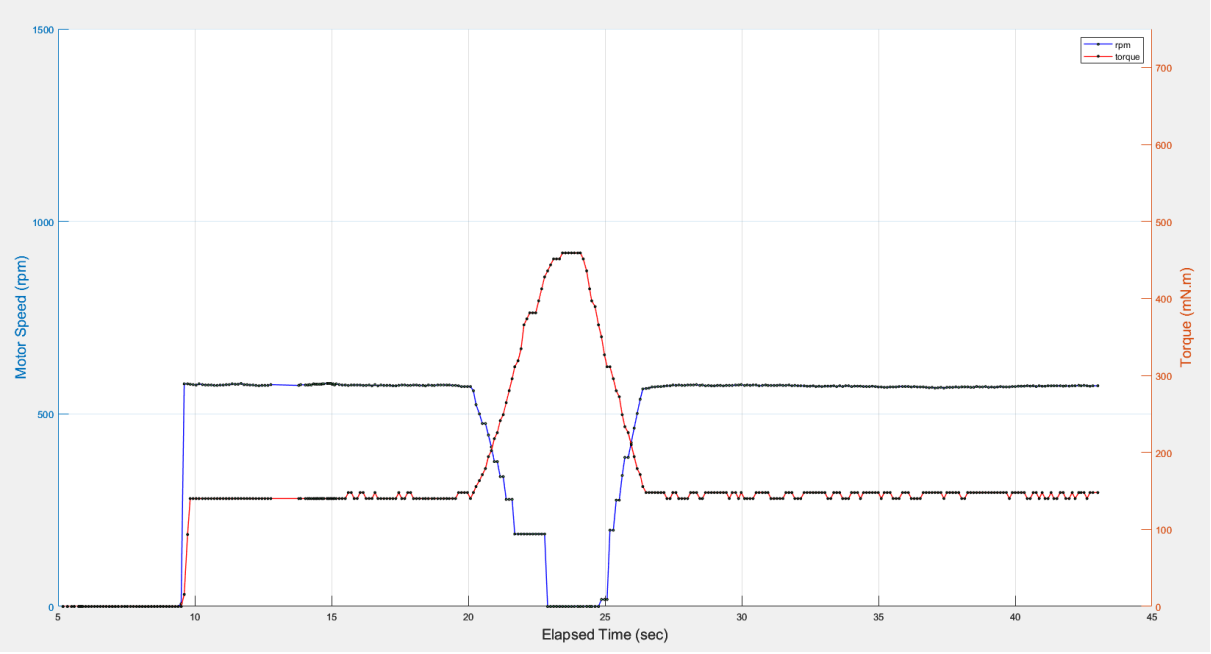


Figure 7: Torque and RPM Plot with 12V of Test Motor Input

Figure 8 shows the power plot obtained during the test, again with a 12V input voltage of the test motor. As expected, the power value during stall conditions is zero since the power is a

product of torque and RPM. The maximum power value obtained after the motor stall, while decreasing the input voltage of the brake motor, equals 9.6432 W. At the same timestamp, the obtained values of torque and RPM are 171.16 mN.m and 538 RPM. Based on the results, we can say that the maximum power of the test motor fed with 12V is 9.6432 W, with corresponding torque of 171.16 mN.m and 538 RPM.

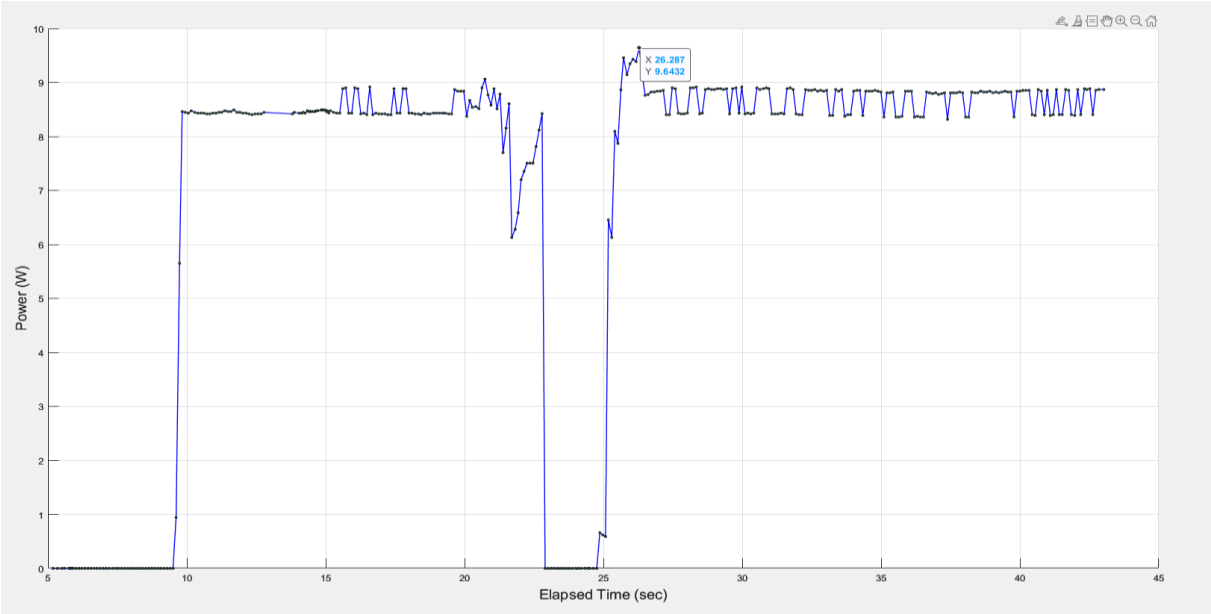


Figure 8: Power Plot with 12V of Test Motor Input

The second test was carried out similarly. The only difference was that the test motor was fed with 18V instead of 12V. *Figure 9* shows the data obtained from the 18V test. The maximum torque is obtained similarly at the stall condition when RPM = 0 and equals 599,07 mN.m.

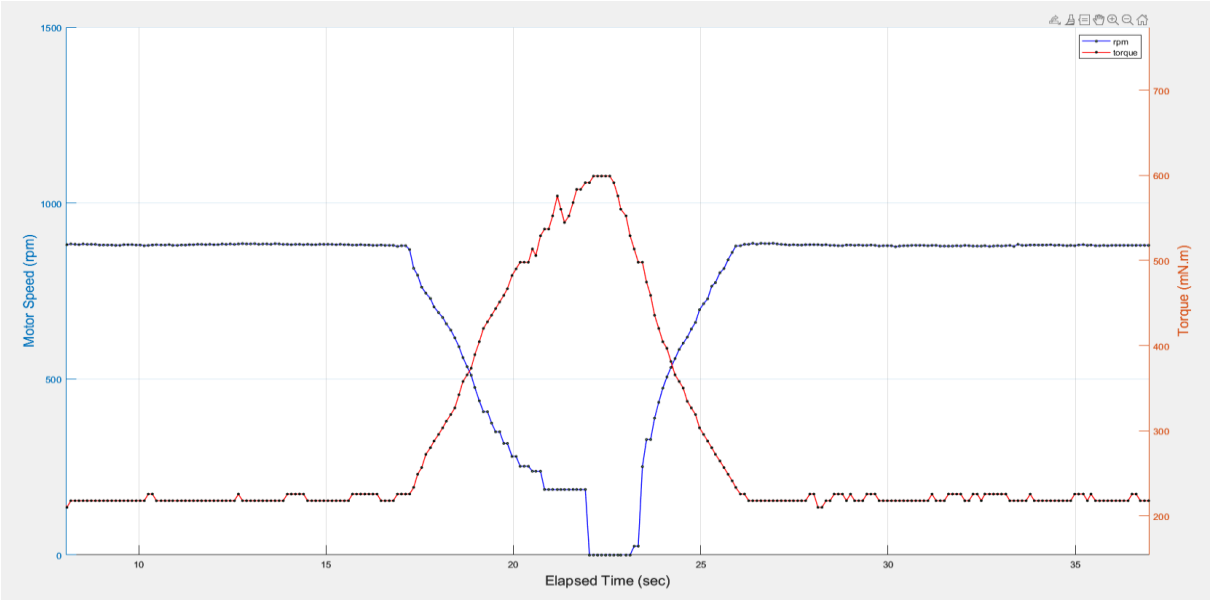


Figure 9: Torque and RPM Plot with 18V of Test Motor Input

The power plot obtained during the 18V test is shown in *Figure 10.* The maximum power value was again obtained after the motor stall, near the point that the brake motor input voltage is 0V. During the 18V test, the maximum power was measured as 22.408 W, at t = 25.296 s. At the maximum power point, the corresponding torque value was 280.08 mN.m, and the RPM = 764.

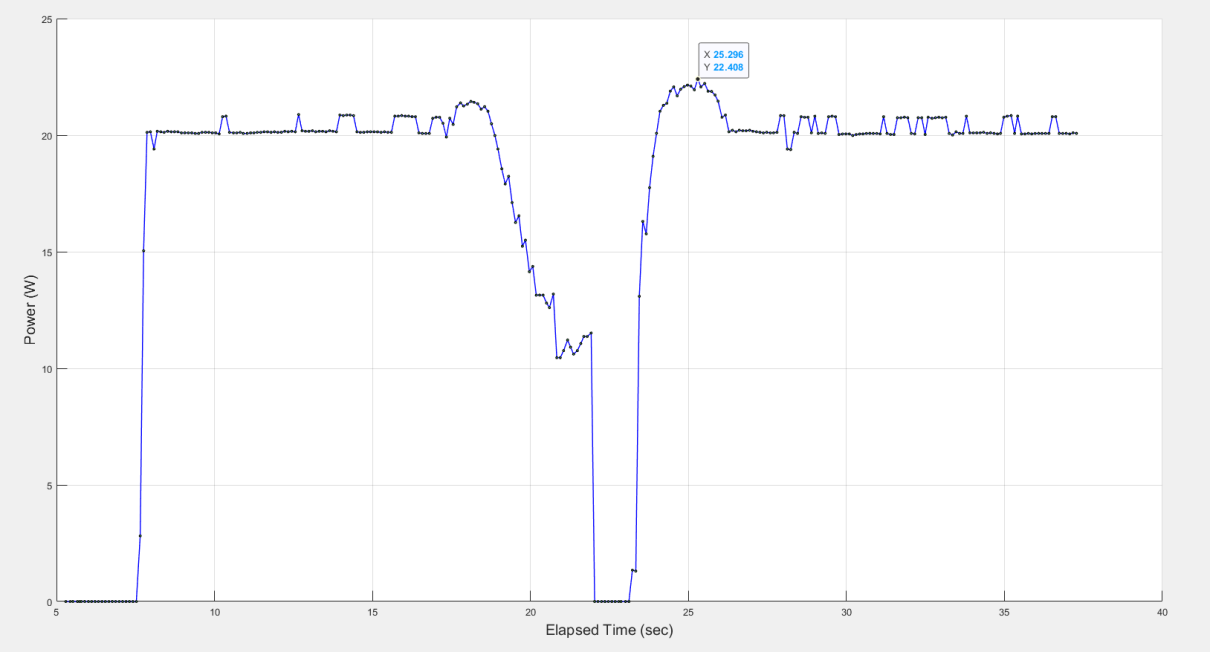


Figure 10: Power Plot with 18V of Test Motor Input

It has been observed that, in both of these tests, the maximum power value is obtained after the stalling process and near the point that the brake motor voltage input is 0. At this point, the power plot showed a sudden increase pattern, similar to a spike. This situation could be caused by measurement errors and instrumentation of the system, both of which are argued further in the discussion section.

5. Discussion

One of the goals of this project is to define the power characteristics of a DC motor. We obtained the power value for the test motor of 9.6432 W at 12V operation and 22.408 W at 18V operation. The power output of the test motor is indicated by the manufacturer as 50W at 40V input voltage. When the obtained data is compared with the value indicated by the manufacturer, the measured power characteristics are not contradictory to the manufacturer's data, and the system operates with an acceptable error margin. The potential measurement errors may depend on various reasons, and these sources of error are discussed in this section.

The accuracy and consistency of the data collected in the project may be affected by several sources of error, including sensor error, noise, calibration error, and serial communication error. It is important to minimize these sources of error to ensure that the data is as accurate and consistent as possible.

One possible source of error leading to inconsistency or inaccuracy in this project is sensor error. The sensors used to collect data, such as the hall sensor and the load cell, may not be perfectly accurate or may drift over time, leading to errors in the data. For example, the hall sensor may not always produce consistent output for a given rpm, or the load cell may drift over time and produce inaccurate force readings. To minimize sensor error, it is important to use high-quality sensors that are designed to be accurate and stable over time and to periodically recalibrate the sensors to ensure that they are producing accurate readings.

Another possible source of error is noise, which can be caused by external factors such as electrical interference or mechanical vibrations. Noise can introduce errors into the data, making it less accurate and consistent. To minimize noise, it is important to eliminate sources of interference and reduce mechanical vibrations. This can be done by using shielded cables, filtering the sensor signals, and isolating the sensors from external sources of noise and vibration.

Calibration error is another potential source of error that can lead to inconsistency or inaccuracy in the data. The load cell may not be perfectly calibrated, leading to errors in the force data. The load cell has a calibration factor that is used to convert the raw sensor readings to force values, and if this factor is not accurate, the force data will be incorrect. To minimize calibration error, it is important to carefully follow the instructions for calibrating the load cell and to use a known weight for calibration. It may also be useful to periodically recalibrate the load cell to ensure that it is producing accurate readings.

Finally, a serial communication error may also lead to errors in the data. There may be errors in the serial communication between the Arduino and MATLAB programs, such as data loss or corruption during transmission or improper synchronization of the programs. To minimize serial communication error, it is important to use a reliable serial connection and to properly synchronize the Arduino and MATLAB programs. This can be done by using a high-quality serial cable, ensuring that the baud rate and other communication parameters are correctly configured, and using error-checking and correction protocols to ensure the integrity of the data.

During the studies with the project, it has been noticed that the RPM of the test motor differs depending on the brake motor's electrical connection. When the brake motor is not running, PWM = 0, but plugged into the power supply, the RPM of the test motor is lower than the case that the brake motor is not connected to the power supply at all. This situation may be caused by the Back-EMF phenomenon, which is present within the brake motor while plugged into the power source, and the shaft of the brake motor is rotating due to the test motor rotation. Another possible reason for this situation is the integrity of the PWM controller module. The PWM controller might be 'leaking' unwanted voltage through the brake motor, which creates a braking condition even though the brake motor is not working. Both of these situations may disrupt the accuracy of the testing process.

6. Conclusion

The output of this project for individuals who participated was the successful creation of a system capable of measuring the specifications of a small-size DC motor using real-time data acquisition. This required the use of various mechanical engineering principles, such as the selection and use of motors, bearings, and other components, as well as the design and assembly of the system. The project also involved the use of data acquisition tools and software, such as Matlab and Arduino, as well as the creation of cable connections. Working in a group of three individuals, we were able to effectively manage our time and budget to complete the project. In addition to gaining technical skills, we also developed teamwork and project management skills through this project. Overall, the output of this project was a functional system that demonstrated our understanding of mechanical engineering principles and our ability to apply these principles in a practical setting.

The purpose of this project is to develop a real-time data collection and analysis system for small-size DC motors. By using sensors and software, this system is able to measure and record various performance parameters of the motor, including rpm, torque, and power. The data collected by the system can be used for a variety of purposes, including quality control, research and development, motor control, and education.

One potential application of this system is quality control. During the manufacturing process, it is important to ensure that small-size DC motors meet certain performance specifications. The real-time data collection and analysis capabilities of this project can be used to monitor and test the motors during production to ensure that they are operating within the specified range of

rpm and torque. This can help to identify and correct defects in the motors before they are shipped to customers, improving the overall quality of the product.

Another potential application of this system is research and development. By conducting experiments and gathering data on the performance of small-size DC motors in a variety of conditions, researchers can study and improve the performance of these motors. For example, the motors can be tested under different load conditions, temperatures, and operating voltages to understand how these factors affect the motor's performance. This data can be used to optimize the design of the motors for specific applications or to develop new technologies for improving motor performance.

In addition to quality control and research and development, this system can also be used for motor control in a variety of applications, such as robotics, drones, and other small-scale systems. By collecting real-time data on the performance of the motor, it is possible to adjust the power supplied to the motor in real-time based on the load, temperature, or other factors. This can improve the efficiency and reliability of the motor in these applications.

Finally, this project can also be used as an educational tool to introduce students to the principles of real-time data collection and analysis. The ability to collect and analyze data in real-time can be a valuable skill in a variety of fields, including engineering, science, and technology. By using this project to teach students about real-time data collection and analysis, educators can help to prepare students for careers in these fields and give them the skills they need to be successful. In addition, students who learn about real-time data collection and analysis through this project may be more likely to pursue further studies in these areas, potentially leading to more advanced research and development efforts in the future.

There are several ways in which the current project, which uses a hall sensor and a load cell to collect data on a motor, could be improved. One possibility is to add more sensors to the system. For example, temperature sensors could be used to monitor the temperature of the motor and its ambient environment, which could provide valuable information on the performance of the motor and help to detect overheating or other problems. Voltage sensors, on the other hand, could be used to monitor the power supplied to the motor and the voltage across it, which could be useful for understanding the electrical characteristics of the motor and for optimizing the power supply.

Another option would be to use wireless communication to collect data from the motor. Currently, the Arduino and MATLAB programs communicate through a serial connection, but

using technologies like Bluetooth or WiFi could make it easier to collect data from the motor and allow the project to be used in a wider range of applications. For instance, the project could be used to test motors that are located far from the data collection equipment or to test motors that are in motion or are otherwise difficult to access. Additionally, wireless communication could facilitate the sharing of data with other devices or systems, such as cloud servers or mobile devices.

Increasing the data collection rate is another avenue for improvement. Currently, the rate at which data is collected is determined by the speed of the main loop in the Arduino program and the delay variable in the MATLAB program. By using a faster microcontroller or by reducing the delay in the MATLAB program, it would be possible to obtain a more detailed and accurate picture of the motor's performance, which could be particularly useful for studying the dynamics of the motor.

The use of machine learning algorithms could also enhance the project by enabling it to automatically identify patterns and trends in the data that are not immediately apparent. This could enable the project to detect and predict problems with the motor, such as impending failures, and provide recommendations for addressing these issues. For example, the project could be trained to recognize patterns in the data associated with different types of failures, such as bearing failure or winding failure and take preventative action to avoid or minimize the impact of such failures.

Finally, expanding the range of tests carried out on the motor could provide a more comprehensive understanding of its capabilities and limitations. By testing the motor under a wider range of rpm, torque, and load conditions, it may be possible to optimize its performance for specific applications and better understand its behavior under different conditions. This could be particularly useful for identifying and addressing problems that may not be apparent under normal operating conditions.

References

[1] J. B. Winther, Dynamometer Handbook of Basic Theory Applications, Cleveland, Ohio: Eaton Corporation, 1975.

[2] Elektrodyne, "Elektrodyne," Associated Elektrodyne Industries Pvt. Ltd., [Online]. Available: http://www.elektrodyne.com/fhp_low_speed_dyn.html. [Accessed 2 June 2022].

[3] A. Farley, "Design and Implementation of a Small Electric Motor Dynamometer for Mechanical Engineering Undergraduate Laboratory," University of Arkansas, North Carolina, 2012.

[4] W. A. B. Jr., "Design of a Small Electric Motor Dynamometer," Massachusetts Institute of Technology, Cambridge, 1951.

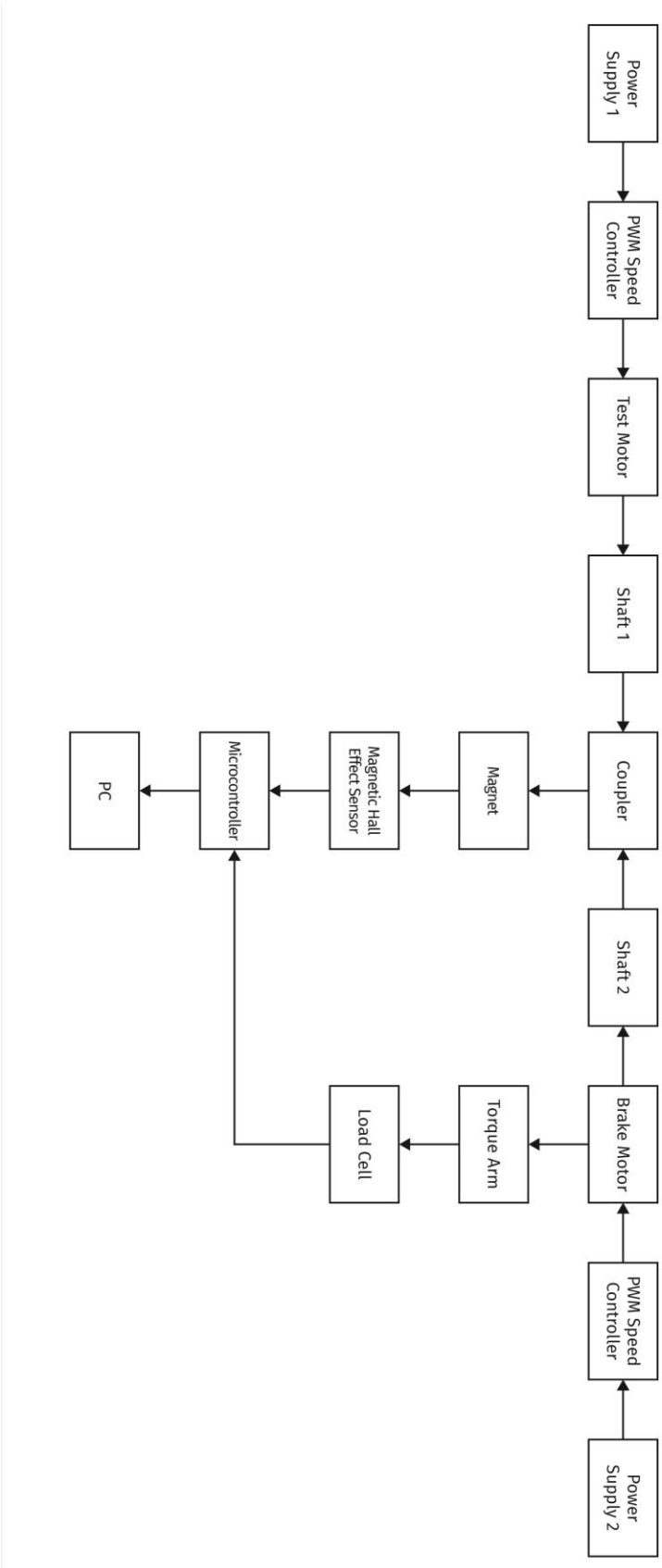
[5] "The Guide to Hall Effect Sensors," [Online]. Available: https://ie.rs-online.com/web/generalDisplay.html?id=ideas-and-advice/hall-effect-sensors-guide. [Accessed 07 06 2022].

[6] "Load Cell and Strain Gauge Basics | Load Cell Central," [Online]. Available: www.800loadcel.com. [Accessed 29 07 2019].

[7] M. Barr, "Introduction to Pulse Width Modulation (PWM)," 01 09 2001. [Online]. Available: https://barrgroup.com/Embedded-Systems/How-To/PWM-Pulse-Width-Modulation. [Accessed 09 01 2023].

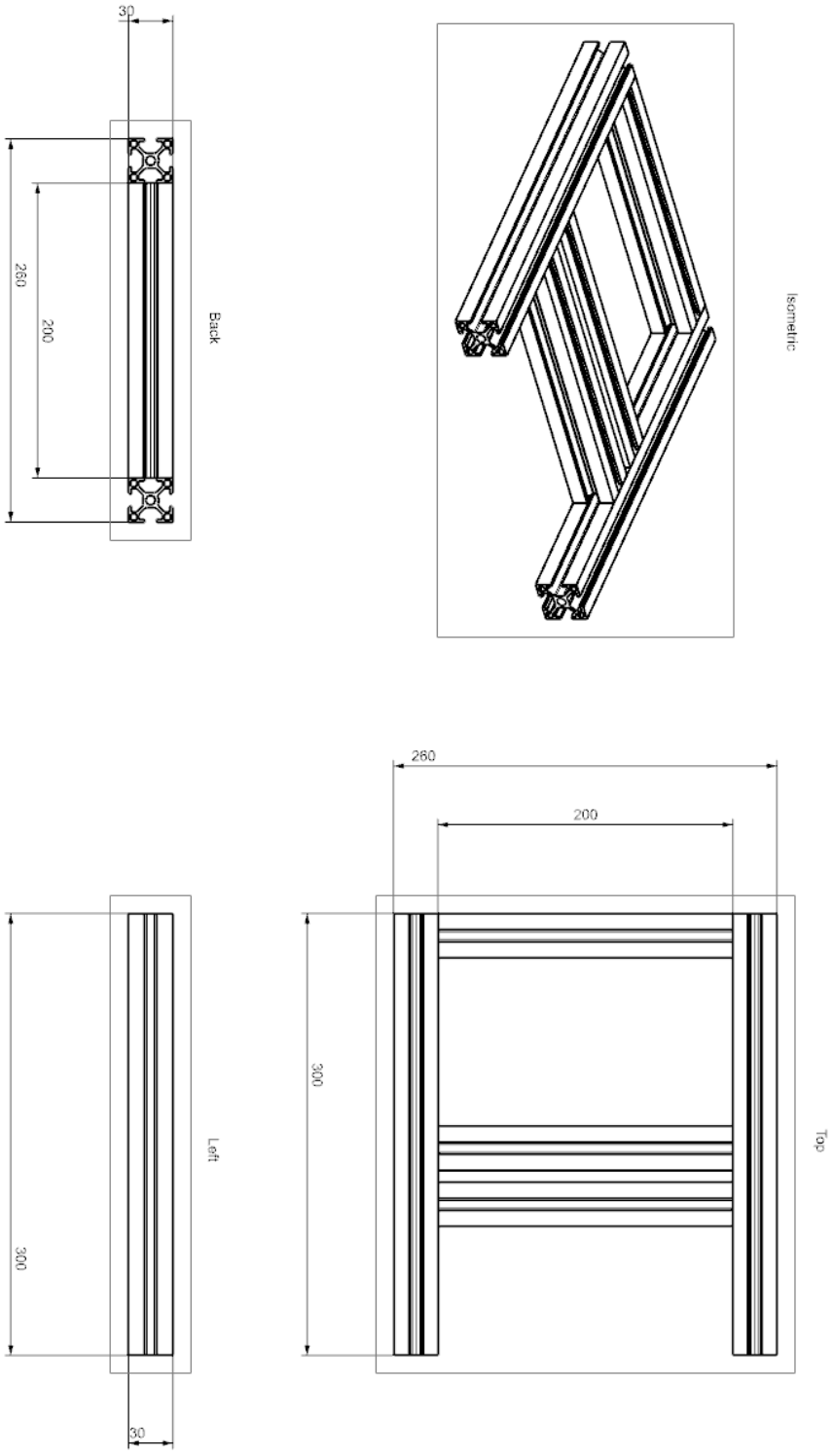
Appendix A

Schematic of Design Configuration



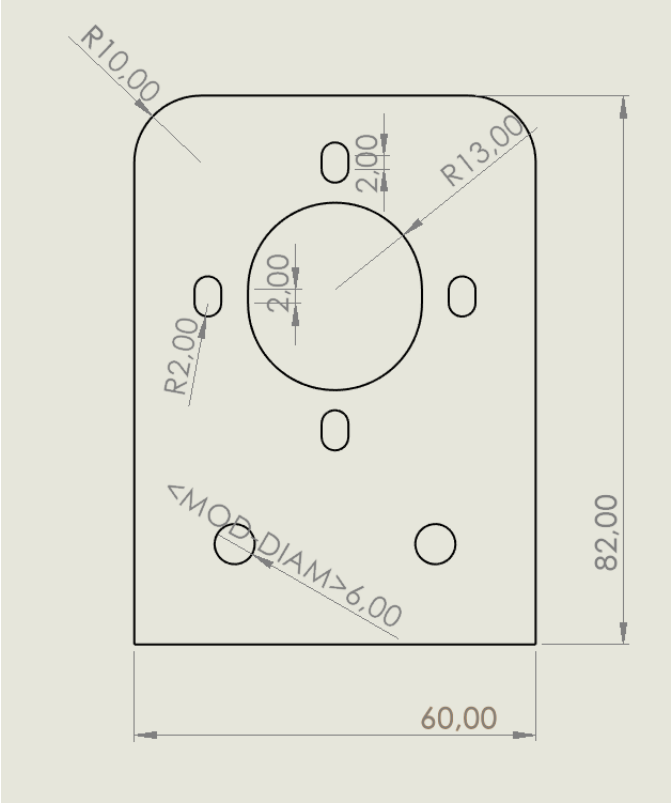
Appendix B

Technical Drawing of Sigma Profiles



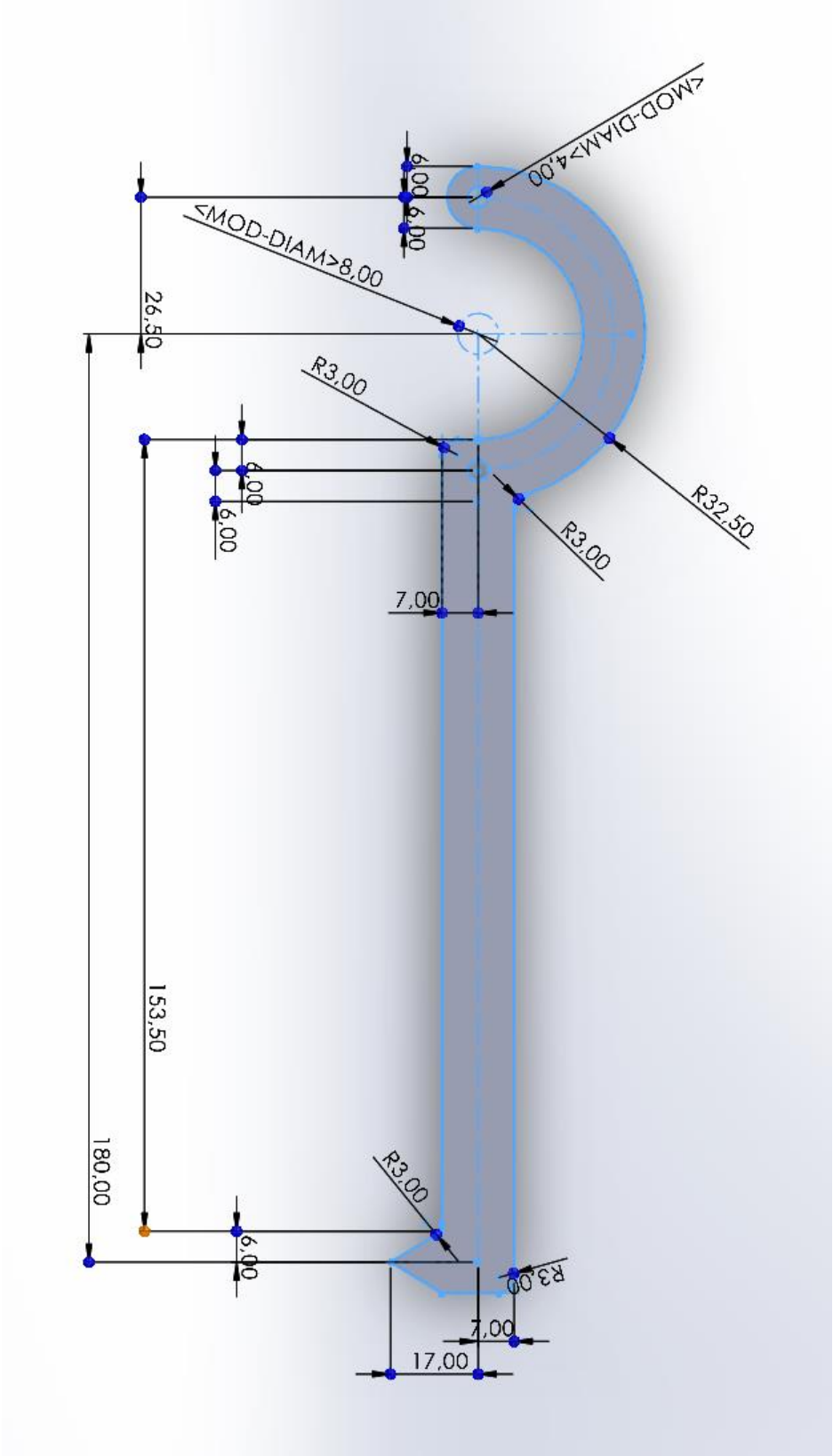
Appendix C

Technical Drawing of Motor Holder



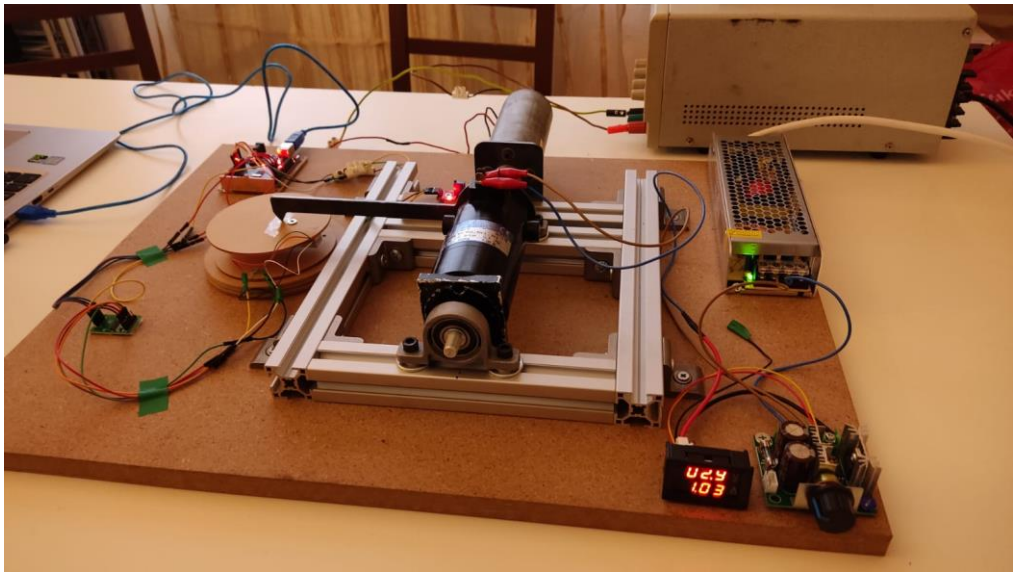
Appendix D

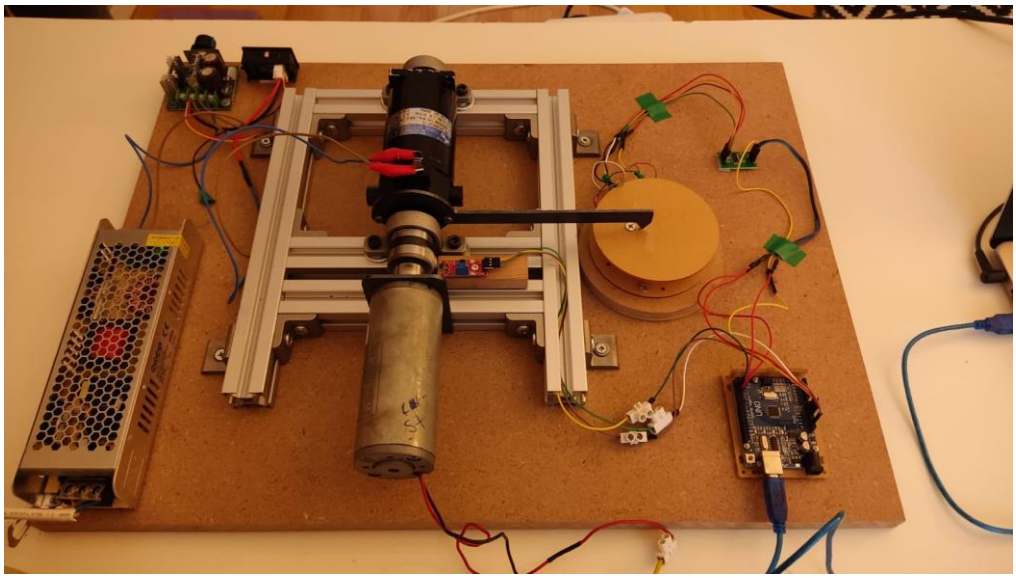
Technical Drawing of Torque Arm

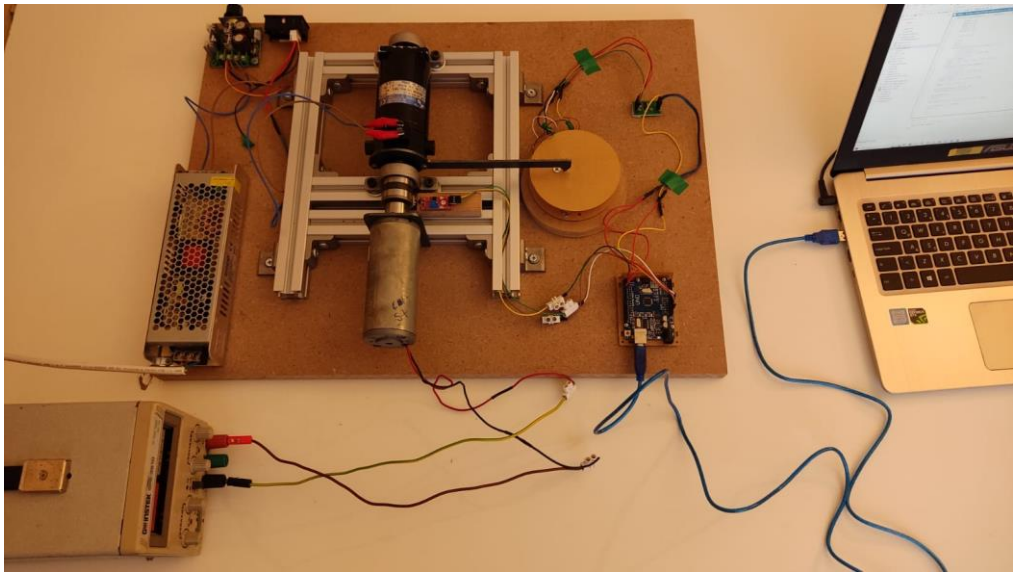


Appendix E

Additional Pictures of Final System







Appendix F

Arduino Code

```
/*
ME 492 - Engine Dynamometer
Group E Senior Project
*/

// include necessary libraries
#include "HX711.h"
#include "Arduino_JSON.h"

// define constants for load cell and Hall sensor pins and calibration factor
#define LOADCELL_DOUT_PIN  3
#define LOADCELL_SCK_PIN  4
#define Hall_Sensor_D 2
#define calibration_factor  -1000

// create variables for HX711 scale, Val1 and Val2, and array for sensor data
HX711 scale;
int Val1=0;
int Val2=0;
int i = 1;
const int numsensor = 2;
int data[numsensor];


// create variables for RPM calculation
float rps;
volatile float rpm;
volatile byte pulses;
unsigned long timeold;
float s;
int refsig=0;//for converting the analog signal coming from hall sensor to digital
through arduino code
float val;//the digital value of the incoming analog signals
int prev_val=0;
volatile unsigned long t,cur_t;//time variables
int counter = 0;
int limit = 10;
unsigned long prev_t = 0;

// function to calculate RPM
void rpm_calculator()  {
    cur_t=micros();
    rpm = 1000000*60/(cur_t-t);
    t=cur_t;
}

// setup function
void setup() {

  // initialize serial communication and set Hall sensor pin to INPUT_PULLUP mode
  Serial.begin(9600);
  pinMode(Hall_Sensor_D,INPUT_PULLUP);
  // initialize scale and tare it
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
```

```
  scale.set_scale(calibration_factor); //This value is obtained by using the
SparkFun_HX711_Calibration sketch
  scale.tare(); //Assuming there is no weight on the scale at start up, reset the
scale to 0

  // attach interrupt and initialize variables for RPM calculation
  attachInterrupt(digitalPinToInterrupt(2),rpm_calculator,RISING);  //attaching
the interrupt and declaring the variables, one of the interrupt pins on Nano is
D2, and has to be declared as 0 here
    pulses=0;
    rps=0;
    rpm=0;
    timeold=0;
    s=0;

  // delay for 2 seconds
  delay(2000);

}

// main loop
void loop() {
  // create JSON object to store data
  JSONVar json;

  // read values from load cell and Hall sensor
  Val2=(scale.get_units(1));
  Val1=digitalRead(Hall_Sensor_D);

  // store sensor data in array
  data[0] = Val1;
  data[1] = Val2;

  // check if current time is the same as previous time
  // if it is, increment counter
  // if it is not, reset counter
  if(t==prev_t){
    counter ++;
  }
  else{
    counter = 0;
  }
  prev_t = t;

  // if counter exceeds limit, set rpm to 0
  if(counter > limit){
    rpm = 0;
  }

  // add rpm and force data to JSON object
  json["rpm"] = rpm;
  json["force"] = Val2*4.3;
  String tmp = JSON.stringify(json);
  Serial.println(tmp);

}
```

Appendix G

MATLAB Code

```
% ME 492 - Engine Dynamometer
% Group E Senior Project
% real time data logger

% clear all variables and close any open serial port connections
clear all
clc
delete(instrfind({'Port'},{'COM10'}));

% define plot properties and function variables
plotTitle_1 = '';
xLabel_1 = 'Elapsed Time (sec)';
plotGrid = 'on';
delay = 0.01;
l_torque_arm = 0.1845; % m
g = 9.80665;
% function variables
time_1 = 0;
time_2 = 0;
data_1 = 0;
data_2 = 0;
count_1 = 0;
count_2 = 0;

% set up plot with two y-axes and legend
hold on

yyaxis left;
plotGraph_1 = plot(time_1,data_1,'-bo',...
    'LineWidth',1,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[.49 1 .63],...
    'MarkerSize',2);
ylim([0 1500]);
ylabel('Motor Speed (rpm)','FontSize',15);

yyaxis right;
plotGraph_2 = plot(time_2,data_2,'-ro',...
    'LineWidth',1,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[.17 .36 .21],...
    'MarkerSize',2);
ylim([0 750]);
ylabel('Torque (mN.m)','FontSize',15);

legend('rpm','torque')
title(plotTitle_1,'FontSize',25);
xlabel(xLabel_1,'FontSize',15);
grid(plotGrid);

hold off

% open serial port
s = serialport('COM10',9600);
fopen(s);
```

```
% start timer
tic
i = 1;

% loop until plot is active
while ishandle(plotGraph_1)

    % read data from serial as string and decode JSON object
    data = fscanf(s, '%s');
    try
    data = jsondecode(data);
    catch
        disp(data);
    end

    % extract rpm and force data from JSON object
    dat_1 = data.rpm;
    dat_2 = data.force;

    % extract elapsed time and update time and data arrays for rpm
    count_1 = count_1 + 1;
    time_1(count_1) = toc;
    data_1(count_1) = dat_1;

    % extract elapsed time and update time and data arrays for torque
    count_2 = count_2 + 1;
    time_2(count_2) = toc;
    % multiply cell data with torque arm length and g to find torque
    data_2(count_2) = dat_2*l_torque_arm*g;

    % update x-axis according to minimum and maximum elapsed time for rpm and
torque
    try
        set(plotGraph_1,'XData',time_1,'YData',data_1);
        set(plotGraph_2,'XData',time_2,'YData',data_2);
        xlim([min(time_1(1),time_2(1)) max(time_1(count_1),time_2(count_2))]);
    catch
        disp("Interrrupted");
    end
    % MATLAB to Update Plot, creates pause for specified delay and update plot
    pause(delay);

end

% close serial port and delete unnecessary variables
fclose(s);

% calculate power from rpm and torque data and plot on new graph
data_1 = data_1 .* pi/30; % rpm to rad/s unit conversion
power = (data_1 .* data_2 ).* 10^-3; % watt
hold on
grid on
plot(time_1,power,'-bo',...
    'LineWidth',1,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[.49 1 .63],...
    'MarkerSize',2);
ylabel('Power (W)','FontSize',15);
```

```
xlabel(xLabel_1,'FontSize',15);
hold off

% convert rotational speed data back to rpm and combine time, rpm, torque, and
% power data into a single array, this feature is for the users want to save the
% data in a CSV file
% data_1 = data_1 .* 30/pi ;
% last_data = [time_1' data_1' data_2' power'];
% write combined data array to a CSV file
% csvwrite('data_18V.csv', last_data)


% clear unnecessary variables at the end
clear count_1 count_2 dat_2 delay max_1 max_2 min_1 min_2...
    plotGraph_1 plotGraph_2 plotGrid plotTitle_1 plotTitle_2 s s_2 ...
    scrollWidth serialPort serialPort_2 xLabel_1 yLabel_1...
    xLabel_2 yLabel_2;

disp('End of the Session');
```