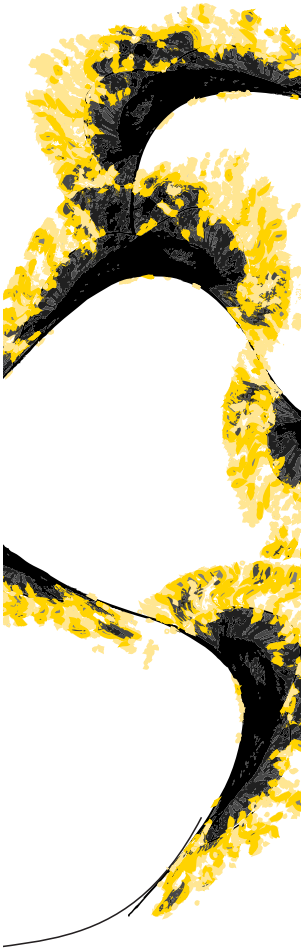


RAM ROBOTICS AND MECHATRONICS

AUTOMATIC CHARACTERIZATION OF A DC MOTOR

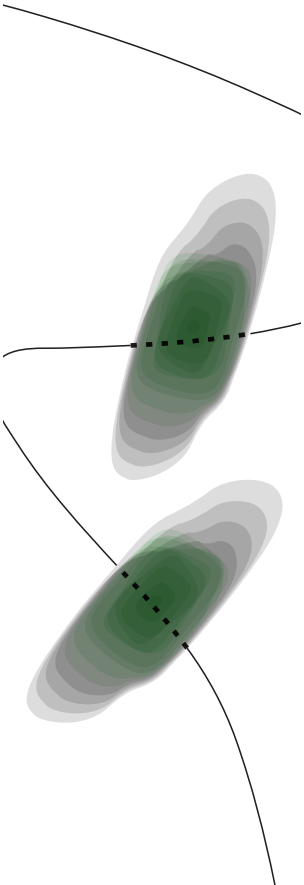
G.J.H.J. (Guus) Branderhorst

BSC ASSIGNMENT

Committee:
dr. ir. D. Dresscher
ing. M.H. Schwirtz
dr. ir. J.J. de Jong

July, 2024

026RaM2024
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands



UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Abstract

This project aimed to identify the characteristics of an unknown PMDC motor through automated measurements. Using a current sensor, encoder, and load cell, the measurements were realized. Despite time constraints, we successfully analyzed velocity and current step responses, determining their corresponding time constants and armature resistance. Although the motor constant, inertia, and friction could not be determined, the inductance was inferred using the electrical time constant and resistance. Issues in the RPM vs torque measurements were traced to mechanical setup instabilities, notably an unbalanced axle and misalignment, causing irregular torque data. Validation through RPM vs time and torque vs motor angle plots confirmed these mechanical issues. Our findings show the motor's resistance and inductance values to be higher than expected, attributed to sensor noise and measurement limitations. The final results offer a comprehensive understanding of the motor's electrical parameters, laying the groundwork for future improvements in mechanical stability and measurement accuracy.

Contents

List of Figures

Chapter 1

Introduction

DC-motors are ubiquitous in robotics, finding applications in a wide range of systems from industrial automation to hobbyist projects. These motors offer great flexibility and precise control but require a thorough understanding of their characteristics and behavior.

Before one can control such a motor, it is imperative to characterize its electrical and mechanical properties. Parameters such as electrical resistance, inductance, motor constants, and inertia vary between motors, even if they are of the same model. Therefore, the process of parameterization becomes essential to ensure accurate control and optimal performance.

However, manually characterizing a DC motor can be quite time-consuming. Creating an automated system for DC motor characterization would not only speed up the process but also reduce the risk of errors.

To identify the motor parameters, sensors such as a rotary encoder, current sensor, and load cell are utilized to obtain data, which leads to the motor parameters.

Step responses and ramp signals are used to analyze data in both the electrical and mechanical domains to not only find motor parameters but also the torque versus RPM curve of a motor under test (MUT).

The following chapters delve into the methodologies and tools utilized in this project, discussing the design, implementation, and evaluation of an automated system for DC motor characterization.

Chapter 2

Literature Review

2.1 DC-motors

2.1.1 What are the most commonly used types of DC motors within the field of robotics and how do they differ in design and functionality?

Nowadays, deciding on a DC-motor topology can be quite cumbersome as DC motors can be categorized in two ways: there exists brushed DC motors and brushless DC motors; similarly, the DC-motor type must be selected as well. Different DC-motor types include servo, permanent magnet, and gear motors.

Brushed DC-motors

The first step in selecting the right DC motor involves choosing between brushed and brushless DC motors. A brushed motor consists of two main parts, a stator, the stationary body of the motor, and a rotor, the inner part of the motor that rotates and generates kinetic energy (refer to fig. 2.1).

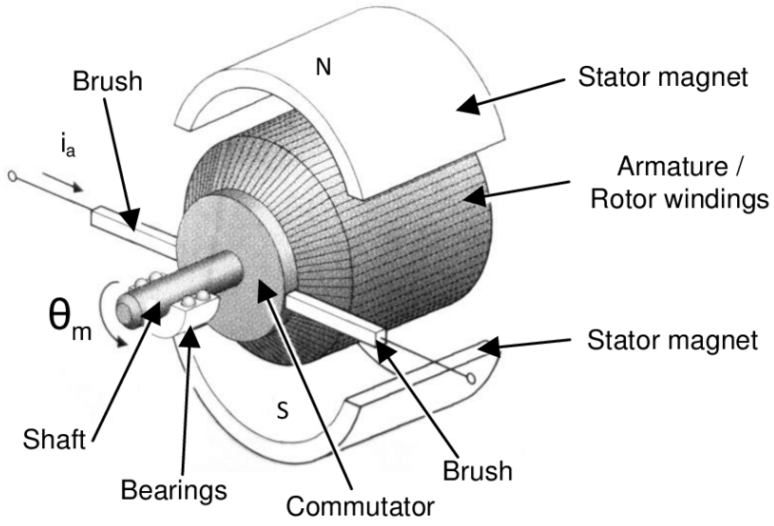


Figure 2.1: This figure shows the inside of a brushed DC-motor[15].

The carbon brushes slide along a copper layer attached to the motor's axle (the commutator). These brushes are used to supply voltage to the motor. As the brushes slide over different segments of the commutator, different rotor windings will sense the applied voltage, creating a dynamic magnetic field[5].

Brushless DC-motors

Over the years, brushless DC motors (BLDC motors) have grown more popular. The internal mechanics of such motor are displayed in fig. 2.2.

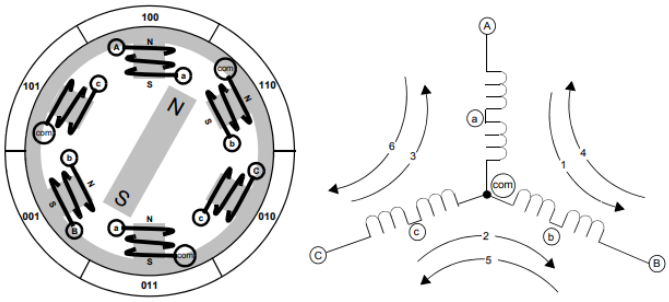


Figure 2.2: This figure shows the inside of a BLDC motor[4].

The stator of a BLDC motor consists of stacked-up laminated steel to carry windings. These windings come in two different forms: a star pattern as shown in fig. 2.2 or in a triangular configuration. The main difference between the two patterns is that the star configuration provides a high torque output at a low RPM (rounds per minute). Whereas the triangular pattern produces a low torque at a low RPM. The rotor of the BLDC motor is comprised of permanent magnets. There is no fixed amount of poles attached to the rotor of a BLDC motor, but increasing the poles does lead to a higher torque output at the expense of angular velocity[9].

Permanent magnet motors

Permanent magnet direct current motors (PMDC motors) are the simplest kind of motors on the market. They typically operate at higher speeds compared to other kinds of DC motors. On the other hand, this is at the expense of the torque which the motor supplies.

The dynamics of a PMDC-motor are illustrated in fig. 2.3.

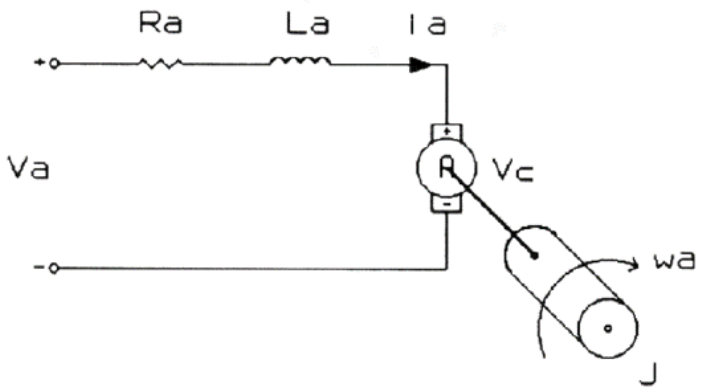


Figure 2.3: The schematic of a PMDC-motor[3].

The dynamic model of a PMDC-motor is highly compact (please refer to fig. 2.3), typically comprising fewer elements compared to other types of DC motors. Consequently, the parameterization process for a PMDC motor is relatively simple.

Gearmotors

Another variant of DC motors is the gearmotor. A DC-Gearmotor is comprised of a PMDC motor and a gearbox. The reason for producing these types of motors is to provide an alternative to the high-speed, low-torque PMDC motors. The gear motor, as opposed to the PMDC motor, can provide high torque at the cost of speed due to its gearbox. The dynamic model of a DC-gearmotor is shown below.

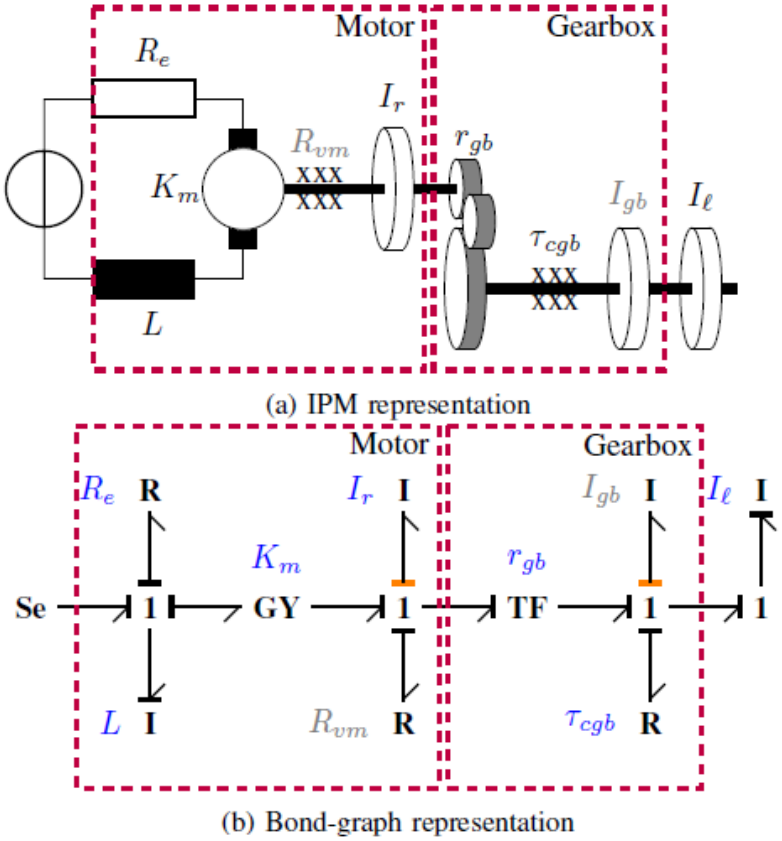


Figure 2.4: The schematic of a DC-gearmotor[7].

The gearbox from fig. 2.4 has the following relations:

$$\tau_r = n \cdot \tau_{out} \tag{2.1}$$

$$\omega_{out} = n \cdot \omega_r \tag{2.2}$$

In this context, τ_r and ω_r represent the rotor torque and velocity, while τ_{out} and ω_{out} denote the output torque and velocity, respectively. From eqs. (2.1) and (2.2), we conclude that a low gearbox ratio (n) leads to a high torque output and low output velocity of the motor shaft. Vice versa, a high gearbox ratio leads to a low torque output and high velocity.

Servo motors

Servo motors are one of many kinds of DC motors that are widely used within the robotics industry. They have proved essential in industrial motion control systems due to features such as reduced noise, increased energy efficiency, response time, and low manufacturing cost[2].

The schematic of a DC servo motor is shown in fig. 2.5.

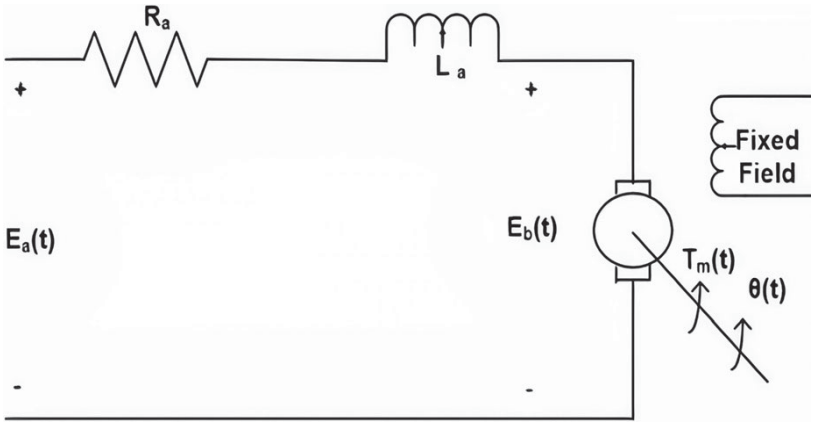


Figure 2.5: The schematic of a DC servo motor[2].

The dynamic model in fig. 2.5 however, does not cover every mechanical aspect of a servomotor. Missing parts are the gear train and position feedback. Also, the current model only exhibits that the electrical domain drives the mechanical domain of the motor, without any frictional components and inertia. To complete our dynamic servomotor model, please refer to fig. 2.6.

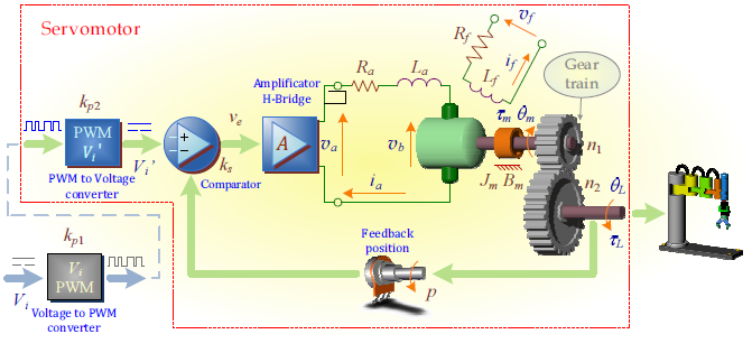


Figure 2.6: This figure displays the whole dynamic system of a DC-servomotor.[20]

From the final diagram, we conclude that the servomotor already has a feedback system implemented. Therefore, we can directly control the motor by sending a PWM signal that corresponds to a desired angle. This makes a servomotor an appealing choice in industry as it is easy to implement and very precise.

2.1.2 Which trade-offs must be considered when selecting a DC-motor design for a specific application?

When choosing between either a brushed DC motor or a brushless DC motor, various trade-offs must be considered. Brushed motors for example require maintenance due to rapid wear of the brushes [9]. As a consequence, brushed motors might turn out costly in the long run. On the other hand, brushed DC motors are cheaper to pick up. Meaning that they are more accessible to numerous clients. Advantages offered by a BLDC motor are higher speed ranges, less weight and volume compared to a brushed motor, and a higher lifetime[5][12]. This means that BLDC motors are ideal for concealing motors in smaller products and reaching higher speeds without the need of maintenance for longer periods. Disadvantages of the BLDC motor are the initial costs of the motor itself and the higher complexity that is required for the controller.

Having understood the trade-offs between brushless and brushed DC motors, let's now delve into the comparisons between permanent magnet, gearbox, and servo DC motors. PMDC motors tend to be easier to characterize and operate at higher speeds compared to other motors as we have discussed in the previous section. However, if a higher torque is demanded for the designated application, one might consider using DC-gearbox motors instead. These motors have an increased characterization complexity due to the added gearbox feature within the model, please

refer to fig. 2.4. Something that must be considered when comparing the gearbox motors with the PMDC motors, is their sizes. Due to the added gearbox, gearbox motors tend to be bulkier in size, which might be undesirable. In addition, gearbox motors are less efficient due to extra frictional losses in the gearbox component[21]. If a very high accuracy is demanded of the robot, servo motors might be the key to success. Servo motors typically cover the middle ground between the permanent magnet motor and gearbox motors. Servo motors are very precise due to their built-in encoder and feedback system, as seen in fig. 2.6. Depending on the desired specifications of the motor, servo motors can vary a lot in cost.

2.2 Motor parameters

From the previous section, we obtained enough background information to decide which type of motor must be selected for the requested tasks. However, once a motor has been selected, it must be characterized as well. Since the DC servo motors already have a built-in feedback loop for position control, we disregarded this motor type for analysis in the report. Therefore we will solely focus on the brushed PMDC-motors and brushed DC gearbox motors.

2.2.1 Which motor parameters are necessary for characterizing the performance and behavior of a DC motor, and how do they contribute to the dynamics?

The dynamics of a PMDC motor

The schematic of a PMDC motor is shown in fig. 2.3. The dynamic model is comprised of various parameters, both electrical and mechanical. These parameters are listed in table 2.1.

Table 2.1: List of Parameters for PMDC Motors[2]

Parameter Name	**Abbreviation**	**Units**
Moment of Inertia	J_m	kg·m^2
Back EMF Constant	K_b	V/(rad/s)
Torque Constant	K_t	N·m/A
Frictional Constant	D_m	N·m/rad/s
Electric Resistance	R_a	Ω
Electric Inductance	L_a	H
Armature Current	I_a	A
Angle of Motor Shaft	θ	rad
Developed Torque	τ_m	N·m
Load Torque	τ_l	N·m
Back EMF	E_b	V
Armature Voltage	E_a	V

The constitutive relations of a gyrator are used to obtain an equation for the motor constants. These are defined in eqs. (2.3) and (2.4).

$$e_1 = K_b \cdot f_1 \tag{2.3}$$
$$e_2 = K_t \cdot f_2 \tag{2.4}$$

Where,

$$\begin{aligned} e_1 &= E_a \\ e_2 &= \tau_m \\ f_1 &= I_a \\ f_2 &= \omega \end{aligned}$$

Through rearrangement and substitution of the flows and efforts, the following equations were obtained for the motor constants.

$$K_b = \frac{E_b}{\omega} \tag{2.5}$$
$$K_t = \frac{\tau_m}{I_a} \tag{2.6}$$

These equations are consistent with findings reported in [3].

If we assume that the electromagnetic losses are equal to zero, the power dissipated in the armature of the electrical domain by the back EMF is equal to the power dissipation in the mechanical domain. From this assumption, a relation between the motor torque constant and motor back EMF constant can be derived. Starting with the mechanical power:

$$P_m = \tau_m \cdot \omega \tag{2.7}$$

Vice versa, the electrical power is given by,

$$P_e = E_b \cdot I_a \tag{2.8}$$

Since we assume that the power dissipation in both domains is equal,

$$\tau_m \cdot \omega = E_b \cdot I_a \tag{2.9}$$

Where rearrangements lead to,

$$\frac{\tau_m}{I_a} = \frac{E_b}{\omega} \tag{2.10}$$

Substituting eqs. (2.5) and (2.6),

$$K_t = K_b = K \tag{2.11}$$

From this, we conclude that both constants are now equal to each other, which matches with the findings in [3][8].

The dynamic variables: armature voltage (E_a), developed torque (T_m) and the back EMF (E_b) can be modeled using the following equations below[2].

$$E_a(s) = R_a I_a(s) + L_a s I_a(s) + E_b(s) \tag{2.12}$$
$$\tau_m = K_t I_a(s) \tag{2.13}$$
$$\tau_m = \left(J_m s^2 + D_m s\right)\theta(s) + \tau_{load} \tag{2.14}$$
$$E_b(s) = K_b s \theta(s) \tag{2.15}$$

The dynamics of a DC-gearbox-motor

A gearbox motor can be modeled as a regular DC motor with a gearbox connected to the axle of the motor and an output shaft. The gearbox is not modeled as solely a transformer but has additional friction effects and brings extra inertia to the model of a motor[7].

The DC gearbox motor can be modeled with an IPM representation and bond-graph representation. The models are displayed in fig. 2.4. From fig. 2.4, the additional friction effects and inertia elements can easily be identified as well.

When it comes to DC gearbox motors, it becomes evident that next to the parameters stated in table 2.1, the added friction, transmission ratio, and the extra inertia must be taken into account as well when parameterizing the motor itself. It must be mentioned that the friction caused by the gearbox can be considered coulomb friction, as the friction within such a transformer is often dominated by coulomb friction. Therefore, the following constitutive relationship can be used for the friction within the gearbox.

$$\tau_m = \tau_{cgb} \cdot sgn(\omega) \tag{2.16}$$

Where, τ_{cgb} is the coulomb friction coefficient and, τ_m the torque and ω the angular velocity [7].

2.3 Existing methodologies

2.3.1 Which methods already exist to characterize a DC-motor and how do they differ in terms of complexity and accuracy?

Least Squares Method

The nonlinear least-square method can be applied to any model of the basic form [17]:

$$y = f(x;\beta) + \epsilon \tag{2.17}$$

In eq. (2.17), y represents the observed data, β the unknown parameters, and x the input variables. Finally, ϵ represents the difference between the observed data and model prediction [11]. The main goal of the least squares

method is to minimize the sum of squared errors. In this summation, the squared error is the squared difference between the observed data (y_i), and the predicted data ($f(x_i, \beta)$, please refer to eq. (2.18).

$$\text{minimize} \sum_{i=1}^{n} (y_i - f(x_i; \beta))^2 \tag{2.18}$$

The minimum error is found by iteratively changing the parameter values until the predicted data is a good representative of the observed data.

In [8], comparisons were made between the nonlinear least squares method, pattern search, and acceleration method. From their results, they concluded that the nonlinear least squares method is the most reliable way of parametrizing a DC motor. This conclusion was drawn by looking at their simulated results and the measured results. From this, it appeared that the nonlinear least squares method most accurately represents the measured response of the motor. In the upcoming sections, the ideas behind the acceleration method will be discussed. However, due to the lack of precision, only the general concept will be discussed.

Acceleration Method

When a DC motor is subjected to a pulse with a given time duration, its dynamic response can be analyzed. Information such as the rise time, overshoot, and oscillation can be obtained, see fig. 2.7.

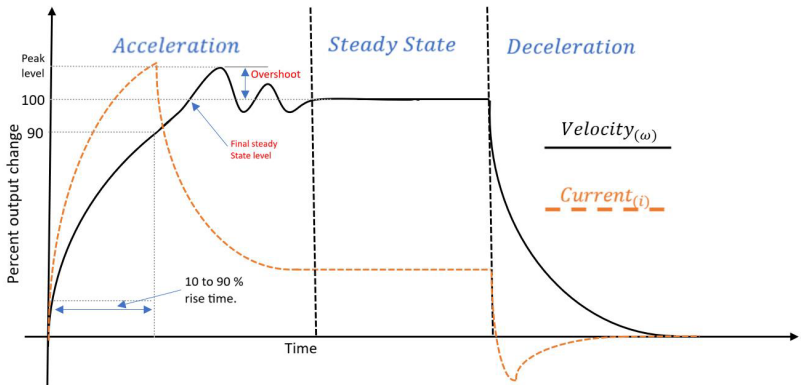


Figure 2.7: This figure displays the dynamic response of a DC motor, which is subjected to a pulse [8].

From the graph, we observe that the dynamic response can be divided into three stages: acceleration, steady state, and deceleration. During the acceleration phase, the rise time is found together with the oscillation of the motor. The next state is the steady-state response of the motor, which is recognized by the constant velocity. Finally, the deceleration phase shows the dynamic behavior of the motor once no current is present in the electrical domain of the device.

The dynamic response of a DC motor is closely related to the acceleration state. The motor's electrical and mechanical time constants, inertia, and load torque all play a role in the dynamic response of a motor.

The steady state of a motor refers to the operating point at which both motor torque and angular velocity are stabilized [8].

Based on the properties of the DC motor when subjected to a pulse, the acceleration method manages to find some motor parameters (K_e,K_t,D,J, and T_L) with limited accuracy.

Pattern Search

The pattern search method is another method that is attractive to implement due to the existence of the Genetic Algorithm and Direct Search Toolboxes[17]. Both of these toolboxes already contain pattern search methods capable of finding unknown parameters.

The global idea behind pattern search is to find a set of points on a lattice, which are called a pattern.

Pattern search consists of two moves: the exploratory move and the pattern move. The exploratory move looks for an improved lattice, leading to a closer minimum objective function, which is a function that determines a measure of discrepancy between the actual recorded data and simulated data. Finding an improved lattice is accomplished

by taking a small step (p) from the current point in the lattice (x_0) as follows [3][19]:

$$F_+ = F(x_0 + p) \tag{2.19}$$
$$F = F(x_0) \tag{2.20}$$
$$F_- = F(x_0 - p) \tag{2.21}$$

The pattern search method keeps trying to minimize the objective. The pattern move is defined by:

$$F_{min} = min(F_+, F, F_-) \tag{2.22}$$

If it was found that $F_{min} < F(x_0)$, than $F(x_0)$ is replaced by F_{min} and the current lattice is updated. Otherwise, the step size is decreased and the process repeats itself. This loop of doing an exploratory and pattern move repeats itself until the minimum objective is found. From this, the unknown parameters can be taken as proper estimates[3][19][17]. The process for pattern search is shown in a flow chart below, please refer to fig. 2.8.

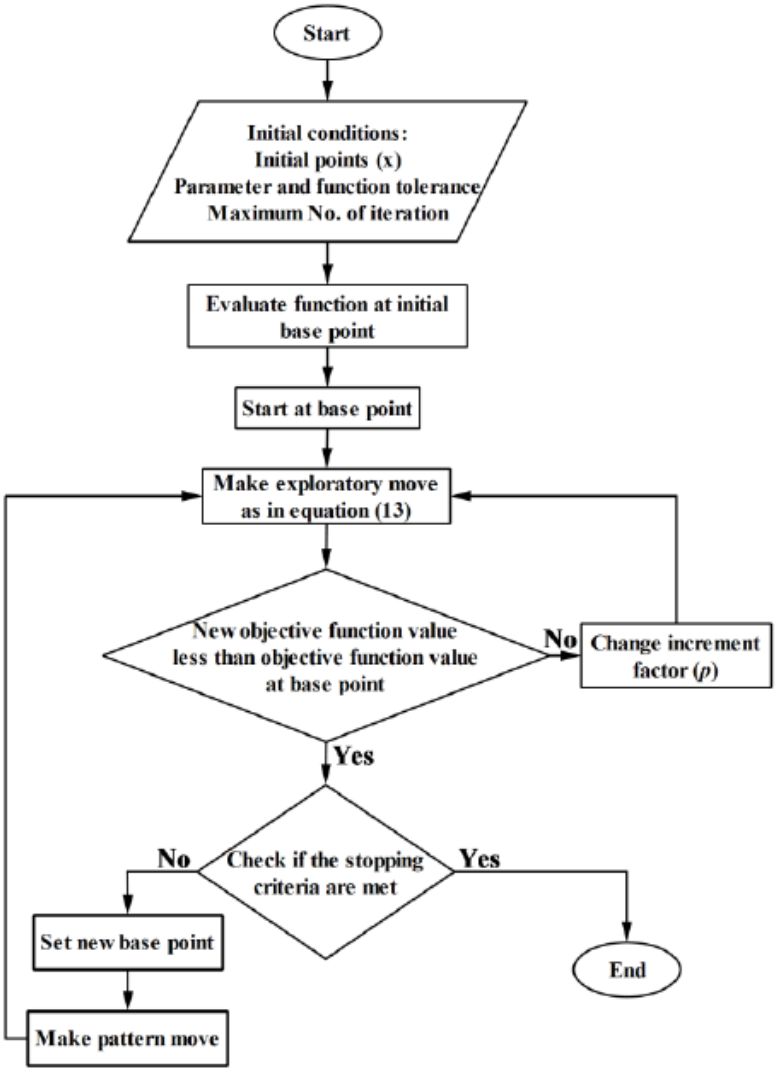


Figure 2.8: This figure shows the flow chart of the pattern search method[3].

RAPID

RAPID is an open-source project, which is based on a dynamometer setup[14]. Dynamometers are setups in which a motor under test (MUT) is connected to an active load. The active load is used to impose characteristics such as torque on the MUT to analyze the MUT under different operating points. From a dynamometer setup, motor

characteristics such as the no-load speed and stall current can be obtained[10]. The no-load speed describes the velocity at which the motor operates when no load is applied. The stall current, on the other hand, shows the amount of load torque required to prevent the motor from moving.

Through the use of a power analyzer, voltage and currents can be measured as well. Since not all variables can be measured at once, some need experimental estimation. Knowing that the current, voltage, and velocity have a linear relationship with the torque, the torque can still be obtained [10].

Just like other dynamometer setups, RAPID exerts a known load on the MUT. Here, this is achieved through an inertial disc and an active or passive braking system to provide friction.

To measure data, RAPID included electronics such as a position, voltage, and current sensor. In addition, to power drive the motor and acquire the data, a motor driver data acquisition tool was utilized [14].

2.3.2 What are the biggest challenges and limitations of the existing methods and techniques to characterize DC motors?

The dynamometer setup proposed by RAPID is an attractive way of characterizing a DC motor. Since this project aims to characterize a DC motor through another DC motor, an upcoming challenge is to include this active load within the model. Decisions have to be made whether the load (predetermined DC motor) will be modeled as mere mass or as another DC motor. This complicates the design as the system will become a higher order.

From previous studies, we have seen that the nonlinear least-square method is most suitable for analyzing measured data and estimating unknown parameters with high precision. However, to optimize the algorithm, proper initial values have to be given which might be a challenge as the parameters are still unknown and the range of each variable differs amongst various motor models.

Chapter 3

Analysis

To create a testbench that is capable of automating parametrization for DC motors, choices have to be made regarding the setup and the software side of the project. During previous years at the RaM group from UTwente, another bachelor assignment was executed. Within this assignment, a student created a dynamometer setup, which is able to collect the torque versus RPM curve for a wide range of DC-motors[16]. Due to time constraints, I opt to further develop this setup in order to perform automated parametrization for a DC motor. For the full setup, please refer to fig. 3.1

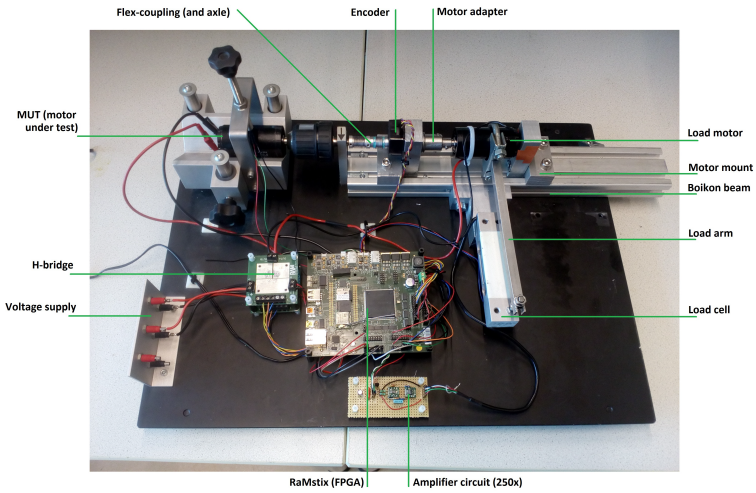


Figure 3.1: This figure shows the current dynamometer setup.

Next to the physical setup, an algorithm has to be selected that estimates the parameters of the MUT. From the literature review, we found that the nonlinear least square method is the most accurate. Therefore this will be the heart of the software side for this project.

3.1 The current setup

As previously stated, a dynamometer setup will be utilized to perform measurements. Dynamometers are known to be creatable with different kinds of active loads. However, the assignment is to create a setup using two DC motors. Therefore, a dynamometer setup must be chosen which satisfies this physical requirement. After talking to staff at RaM UTwente, I discovered that an existing setup already meets this requirement.

The current dynamometer setup measures an RPM range of 0-5000 [RPM] and a torque range of 0-5 [Nm]. The RPM measurements are performed through an encoder and the torque measurement through strain gauges.

3.1.1 Encoder functionality

Within the current setup, an optical incremental encoder (Heds-5605E06) is used to measure the shaft position of the MUT. This encoder works at 5 [V] and has three channels: A, B and I. Channels A and B are offset by 90 degrees, please refer to fig. 3.2[1]. Effectively, this allows the user to count how many degrees the motor axis turned. If the motor spins in a clockwise direction, the counter goes up if there is a rising edge on channel A and if channel B is already high. Vice versa, if the motor spins in a counterclockwise direction, the counter is decremented if there is a rising edge on channel A and if channel B is low. This also implies that the direction of the motor can be checked by looking at the value of channel B every time channel A encounters a rising edge.

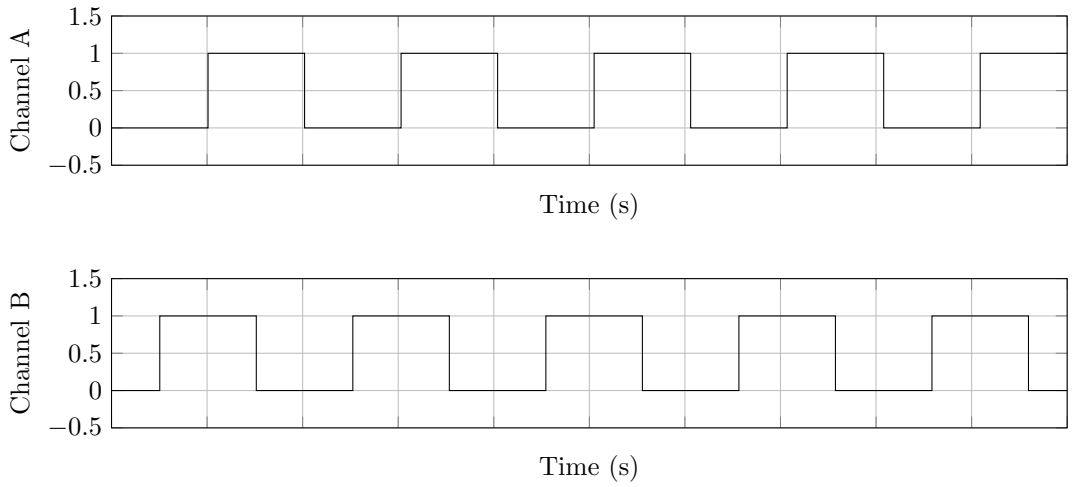


Figure 3.2: This figure shows a graph of channels A and B from the encoder.

Since this encoder has a resolution of up to 800 counts per revolution[1], the angle by which the encoder turned can be found by:

$$\text{Axle angle} = \frac{counts}{800} \cdot 360^\circ \tag{3.1}$$

To ease angle tracking even further, channel I can be used. Channel I becomes high every time the encoder reaches one whole revolution[1]. This allows the user to reset the encoder counter, therefore allowing to track angles between 0 - 360 degrees.

Knowing how the encoder output can be read and computed to an axle angle, the RPM (rounds per minute) value can be computed as well. We will use the angle-to-RPM conversion to obtain the torque vs RPM plot. Going from an angle to a velocity requires differentiation with respect to time. Currently, we measure the angle in degrees and the time in seconds. Therefore the following equation can be used to obtain RPM:

$$\text{RPM} = \frac{\Delta\theta}{\Delta t} \cdot \frac{60}{360} \tag{3.2}$$

Using eq. (3.2), the velocity of the motor axle can be found and used for characterization purposes. The next section will zoom in on how strain gauges are utilized to measure the torque within the dynamometer setup.

3.1.2 Load cell functionality

Within the dynamometer setup, an arm was added as a lever to the load motor. The motor is connected on one end of the arm, and strain gauges are integrated on the other end, this can be seen in fig. 3.1.

The strain gauges are used to measure the torque that is provided by the MUT. This can be done by mounting an arm horizontally onto the active load and load cell. As previously mentioned, it was opted that the maximum torque should be 5 [Nm]. The current setup has an arm with a length of 0.2 [m]. To measure the desired torque range of 0 to 5 [Nm], the maximum force that should be measurable by the load cell is:

$$F_{max} = \frac{\tau_{max}}{L} = \frac{5}{0.2} \tag{3.3}$$

Which is equivalent to 25 [N].

To find an eligible load cell for the setup, we must look at the required capacity of the load cell. Since the load cell should be able to measure up to at least 25 [N], the capacity must be greater than:

$$Capacity \geq \frac{F_{max}}{g} \tag{3.4}$$

From this, we find that the minimum capacity should be 2.55 [Kg]. For this project, the Tedea Huntleigh, model 1006, load cell was chosen. This load cell has a capacity of 5 [Kg], a rated output of 2 [mV/V], and operates at a maximum input voltage of 15 [V]. However, we will use an excitation voltage of 5 [V]. This means that the maximum output of the load cell is 10 [mV]. The load cell is connected to a differential voltage amplifier to boost the output of the load cell to a range that is specified for an ADC (Analog-to-Digital Converter), which filters the signal and converts the analog signal to a digital signal. The bit sequence must first be converted to a voltage by the microcontroller to find the force:

$$V = \frac{D}{2^n - 1} \cdot (2 \cdot V_{max}) - V_{max} \tag{3.5}$$

Where D is the digital output of the ADC, which is represented by an integer number. Once the voltage is determined, the force is found using eq. (3.6):

$$F = \frac{V}{V_{max}} \cdot F_{max} \tag{3.6}$$

Where F_{max} is:

$$F_{max} = 5 \cdot 9.81 = 49.05[\text{N}] \tag{3.7}$$

Finally, the torque is computed with the following equation:

$$\tau = F \cdot L \tag{3.8}$$

This concludes the analysis of the load cell implementation.

3.1.3 Measuring the torque vs RPM graph

To measure the torque versus RPM graph, the MUT is constantly supplied with its maximum voltage input, while the load motor's input is incremented with steps. This way, the MUT was subjected to different operating points, allowing it to measure its torque at different RPM values.

3.2 Characterizing the DC-motor

With a comprehensive understanding of the current sensors employed in the existing setup, the focus now shifts towards outlining the methodology for characterizing the DC motors. This section will explore how measurement data is analyzed to find the parameters of the MUT.

3.2.1 Considerations

For this project, several considerations must be made due to time constraints. First of all, only brushed PMDC motors will be focused on this project. This means that gearbox motors are out of the scope of this project but would be a good improvement for future research. Secondly, the friction will be modeled as simple as possible, namely as a viscous-coulomb friction. This will be done through a viscous damping constant (D_m) and a coulomb friction torque (τ_L), please refer to eq. (3.12).

3.3 The MUT

Before characterizing the MUT, a proper model must be created to identify the parameters of interest. The model of a PMDC motor is shown in fig. 3.3.

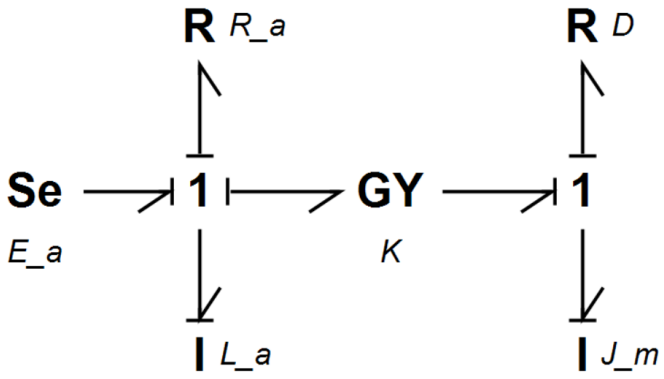


Figure 3.3: This figure shows the bondgraph implementation of a PMDC motor.

From the figure, we conclude that the parameters of interest are: the armature resistance, inductance, the motor constants, the mechanical friction and the inertia of the motor. The upcoming sections will again discuss both the electrical and mechanical domains and which methods are used to find the parameters of the MUT.

3.3.1 Electrical Domain Characterization of the MUT

For the characterization of the MUT, the parameters of interest are the armature resistance and inductance. A current sensor is integrated into the setup to monitor the armature current of the MUT continuously.

The armature resistance of the MUT

The resistance is determined through eq. (3.9).

$$R_a = \frac{E_a}{I_{a_{\text{sensor}}}} \tag{3.9}$$

The following steps explain the process of determining the armature resistance:

1. Feed a ramp input to the MUT.
2. During the ramp response, continuously compute and store the armature resistance using eq. (3.9).
3. Once the ramp response is completed, calculate the average value of the stored resistance.
4. The average value obtained is the armature resistance value for the MUT.

The inductance of the MUT

To measure the armature inductance, we supply the MUT with a low enough input voltage, in the form of a step, such that only the electrical domain is excited, which means that the rotor is not spinning. This way, the electrical domain of the motor can be seen as an RL circuit, which comes with a solution in the form of:

$$I(t) = \frac{E_a}{R_a}\left(1 - e^{-\frac{R_a}{L_a}t}\right) \tag{3.10}$$

Once the step response is measured, eq. (3.11) is used to find the inductance:

$$L_a = R_a \cdot \tau \tag{3.11}$$

Where the time constant (τ) is found by evaluating the current at 63.2% of the maximum amplitude in the current versus time step-response plot.

3.3.2 Mechanical Domain Characterization of the MUT

For the mechanical parameters, strain gauges will be used to determine the motor torque of the MUT, and the current sensor for the armature current.

The motor constants of the MUT

The mechanical domain of the MUT is characterized by starting with the determination of the motor constant. This is done using the following steps:

1. Feed the nominal voltage as input to the MUT.
2. Feed a ramp as input to the active load. This way, the motor torque is put on the stator of the active load, allowing to read the motor torque through the strain gauges.
3. During the ramp response, continuously compute and store the mechanical motor constant using eq. (2.6).
4. Once the ramp response is completed, calculate the average value of the stored motor constants.
5. The average value obtained is the final motor constant for the MUT.

Assuming that the electromagnetic losses are negligible, the motor constants are equal to each other. Thereby, we did not only discover the mechanical, but also the electrical motor constant.

The friction of the MUT

Next, the friction of the MUT must be characterized. During this experiment, the nominal voltage is inputted to the MUT, while incrementing steps are inputted to the active load. Every time the input of the load is increased, we wait until the angular velocity reaches a steady state. Once a steady state is reached, the torque and angular velocity are stored. Once the active load reaches the final input voltage, a motor torque versus angular velocity curve can be created with the data points. Finally, curve fitting is applied to estimate the curve with an equation in the form of:

$$\tau_m = D_m \omega_m + \tau_L \tag{3.12}$$

It's important to note that steady-state data is used to eliminate the acceleration component of eq. (2.14). However, not only the friction of the MUT is measured but also the friction from the active load and coupling mechanism of the dynamometer. Therefore, eq. (3.12) must be rewritten to:

$$\tau_m = \left(D_{MUT} + D_{ActiveLoad} + D_{coupling}\right) \omega_m + \tau_L \tag{3.13}$$

From which $D_{ActiveLoad}$ is already known. The equation can be generalized to the following format:

$$\tau_m = D_{total} \omega_m + \tau_L \tag{3.14}$$

Where the friction constant (D_{total}) and load torque (τ_L) are found by curve fitting the torque versus angular velocity curve, created through steady-state data at different voltage inputs. Next, the viscous friction constant related to the MUT is determined through eq. (3.15)

$$D_{MUT} = D_{total} - D_{ActiveLoad} - D_{coupling} \tag{3.15}$$

By observing eq. (3.15), it becomes obvious that the friction component due to the coupling mechanism is also yet to be determined.

The friction component of the coupling mechanism can be determined prior to characterizing the MUT. One way is to use two maxon motors, from which all parameters are already known. Then, by performing a similar experiment with steady state data collection, The viscous friction component of the coupling mechanism is determined by using:

$$D_{coupling} = D_{total} - D_{motor1} - D_{motor2} \tag{3.16}$$

Where D_{total} is again determined through curve fitting the steady state data. Once the friction of the coupling mechanism is determined, the MUT friction can be determined.

The inertia of the MUT

Finally, the inertia of the MUT is computed. The motor's inertia can be determined by analyzing the step response of the angular velocity (ω). This way, the solution to an angular mechanical system which is comprised of a motor shaft with viscous and/or coulomb friction can be used, please refer to eq. (3.17)[13]:

$$\omega(t) = \frac{\tau_m}{D_m}\left(1 - e^{-\frac{J_m}{D_m}t}\right) \tag{3.17}$$

By examining the time constant (τ),

$$\tau = \frac{D_m}{J_m} \tag{3.18}$$

The inertia can be found through eq. (3.19).

$$J_m = \frac{D_m}{\tau} \tag{3.19}$$

However, since the MUT is now connected to the active load through the coupling mechanism of the dynamometer setup, both equations need to be rewritten. Starting with eq. (3.17):

$$\omega(t) = \frac{\tau_m}{D_{total}}\left(1 - e^{-\frac{J_{total}}{D_{total}}t}\right) \tag{3.20}$$

By evaluating eq. (3.20), it becomes evident that the inertia also contains multiple components. Namely, the MUT inertia, the coupling inertia, and the load inertia. To determine the MUT inertia, eq. (3.21) is used:

$$J_{MUT} = J_{total} - J_{ActiveLoad} - J_{coupling} \tag{3.21}$$

Where,

$$J_{total} = \frac{D_{total}}{\tau} \tag{3.22}$$

Therefore, eq. (3.21) becomes:

$$J_{MUT} = \frac{D_{total}}{\tau} - J_{ActiveLoad} - J_{coupling} \tag{3.23}$$

From eq. (3.23), we observe that the coupling inertia must be determined prior to characterizing the MUT.

To obtain the inertia of the coupling mechanism, the same dynamometer setup is used, but with two Maxon motors. From these motors, we already know the inertia. This implies that performing the same step-reponse analysis as described above, the coupling inertia can be found through:

$$J_{coupling} = J_{total} - J_{motor1} - J_{motor2} \tag{3.24}$$

This concludes the analysis of the motor parameters.

3.3.3 Fine tuning

Once the electrical and mechanical parameters are determined experimentally, they must be validated. Oftentimes, it can be observed that experimentally determined parameters lead to a simulation model, which still requires fine tuning[6]. Therefore, the nonlinear least square method will be used to improve the model. This can be done through the Parameter-Estimation tool from Matlab[6]. One of the challenges we found earlier within such a toolbox is the need for an initial value. However, since we've already determined the parameter values experimentally, these parameters can be utilized as ranges. This allows to improve the accuracy of the toolbox.

Chapter 4

Design

4.1 The general overview

Having a systematic approach to automating the characterization process for brushed PMDC motors, a design must be considered. The design of the motor test bench will be based on the existing setup made by [16]. The electrical domain will be different compared to the prior setup. Therefore, a new system overview is created, please refer to fig. 4.1.

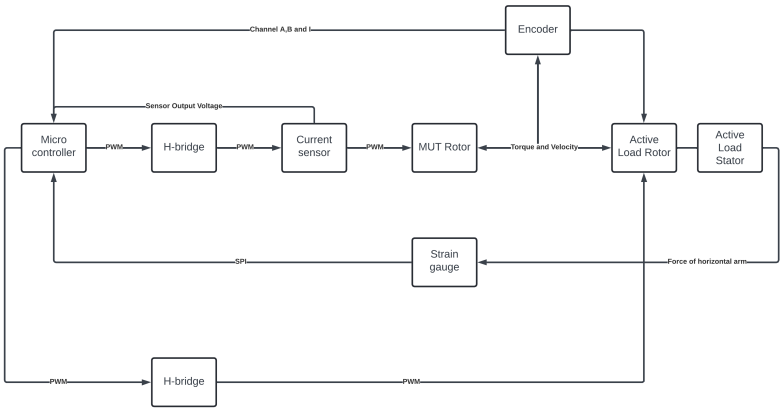


Figure 4.1: This figure shows the system overview.

The microcontroller is responsible for controlling the signals that are sent to the MUT and active load. These signals, however, will first be parsed through H-bridges. The H-bridges enable control over the direction and velocity of both motors.

Within the system, three sensors are integrated: a current sensor, an encoder, and a strain gauge. As shown in fig. 4.1, the current sensor is connected between an H-bridge and the MUT. This placement allows for the measurement of the armature current, which is then sent to the microcontroller for further processing.

The encoder is attached to the rotor of the MUT, with its outputs connected to the microcontroller. Using the working principle of such an encoder, as described in chapter 3, the angle of the motor axle can be tracked within the microcontroller, which allows us to determine the velocity of the rotor.

The final sensor is the strain gauge. The strain gauge is positioned next to the load motor. By attaching an arm horizontally to the stator (the body of the motor), torque can be sensed by the strain gauge once the MUT starts spinning the axle of the load. Newton's third law applies, meaning that the stator starts to exert an equal but opposite torque, which is the torque that is sensed by the strain gauge while the arm is deforming the strain gauges.

4.2 Data Flow

Having established a general concept for the test setup, the next step is to think about the connections between different components in the system. For that, a high-level data flow diagram was created as shown in fig. 4.2.

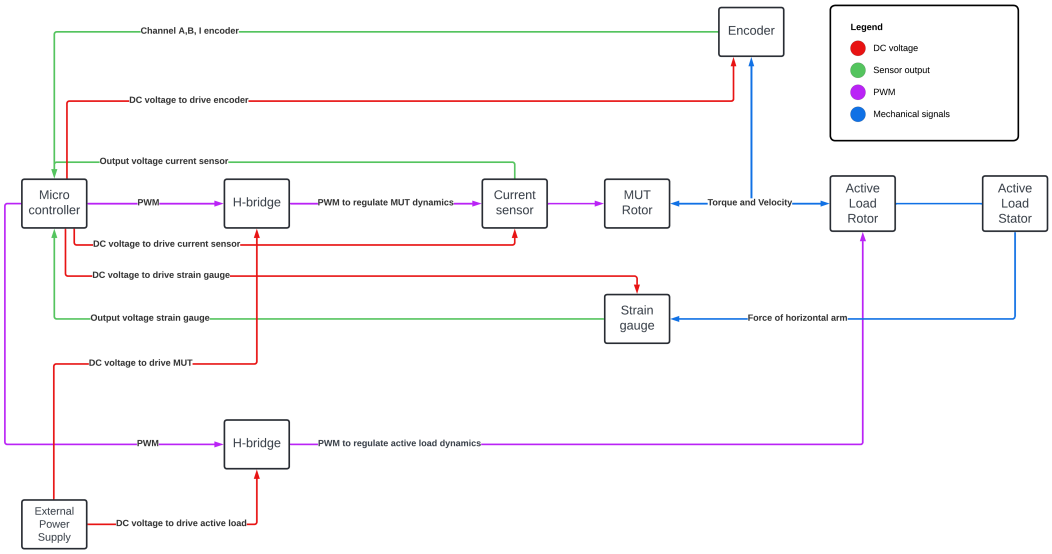


Figure 4.2: This figure shows the high-level data flow diagram of the system.

The high-level diagram provides great insight into the general signals that flow from one component to another. However, a more detailed diagram is required to build the setup. Therefore, fig. 4.3 was created.

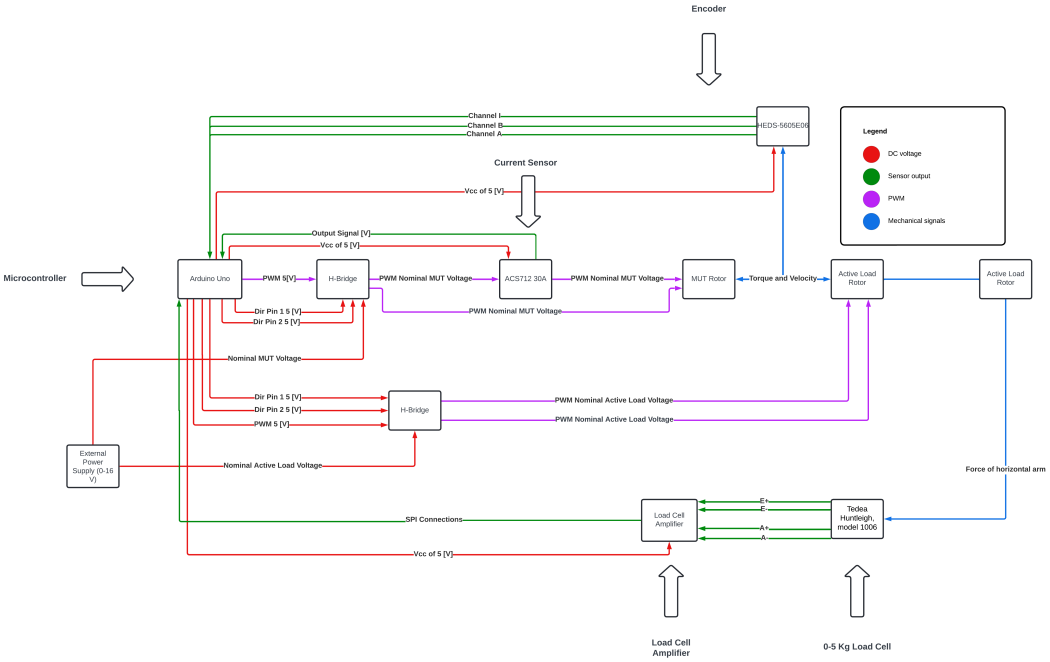


Figure 4.3: This figure shows the low-level data flow diagram of the system.

From fig. 4.3, it becomes clear how the Arduino Uno will communicate with the different sensors and how the motors are controlled. Therefore, the low-level diagram provides great insight into the wiring between the various components.

4.3 The Load Cell

4.3.1 Load Cell Support Structure

Since a new load cell was ordered, the old setup (see fig. 3.1) had to be modified. The newer load cell is much smaller than the old load cell. Ultimately, the choice between either decreasing the arm attached to the load motor or exchanging the current supporting structure for the load cell must be made.

Having a longer arm empowers the user to measure a higher torque range because a longer arm allows to measure the same torque while applying less force on the load cell. Ultimately, a longer arm allows higher torque measurements without exceeding the capacity of the load cell. This can also be deducted from eq. (3.6).

Based on the prior reasoning, the choice was made to build a new supporting structure for the load cell. The support was created in Solidworks, after which it was printed. The final structure is shown in fig. 4.4.

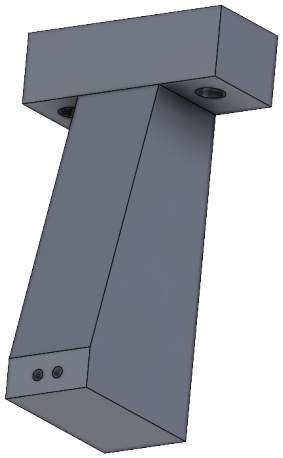


Figure 4.4: This figure shows the Solidworks part which will support the load cell.

4.3.2 Measurements

Once we integrate the load cell into the complete setup, we can perform the torque versus RPM measurement and plot both RPM versus time and the torque versus time.

Torque RPM Measurement Procedure

1. **Input Signals MUT And Load Motor:** We provide the MUT with its nominal voltage at a PWM of 255. Once the MUT spins for 2 seconds (to ensure a steady state), the load motor is provided with a ramp input, where the PWM starts at 0 and increments every 500 ms by 2.

2. **Recording Torque And RPM:** During every 500 ms period, we record the average torque and RPM over the last 400 ms, which are used within the plot.

3. **Stop Condition:** The recording stops if the MUT reaches a velocity of 0 RPM or if the load motor reaches a PWM of 255 (which is the maximum value).

RPM

To check the functionality of the load cell, we performed two measurements with the torque RPM measurement procedure as described in the previous section. First, we checked whether the RPM of the MUT decreases linearly over time. Since the load motor will be spinning in the opposite direction of the MUT and the load motor's input increases over time, we expect that the velocity of the MUT will decrease linearly. After the first measurement, the following graph was obtained:

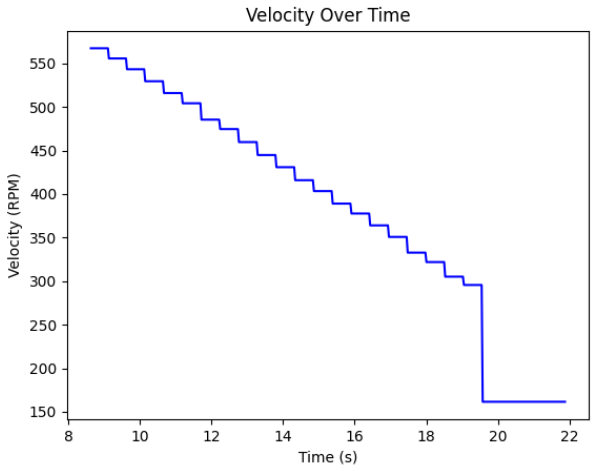


Figure 4.5: This figure shows a plot of the RPM vs time while running the RPM versus torque script.

From fig. 4.5, we conclude that our hypothesis matches with our findings after measuring the velocity over time.

4.3.3 Torque

After we validated that the RPM plot was as expected, we performed the same experiment, but with the motor torque. Since the load motor is spinning in the opposite direction from the MUT and due to its increasing input, we expect the motor torque to increase linearly over time. This also explains why a linearly decreasing RPM graph is obtained, because a bigger opposing force is required to decrease the velocity of the motor. This force must be the motor torque. After performing the measurements, fig. 4.6 was obtained.

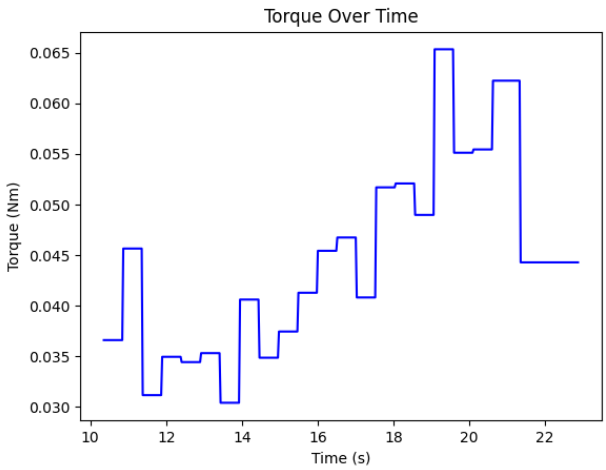


Figure 4.6: This figure shows a plot of the torque vs time while running the RPM versus torque script.

Clearly, the graph does not show the expected linear behavior. In general, we can say that the torque is increasing over time as expected, but not linearly.

4.4 The Current Sensor

Within the setup, as mentioned in a previous section, we implemented the ACS712 30A current sensor to measure the armature current the MUT draws. The ACS712 is a hall-effect current sensor. These types of current sensors output a voltage that depends on the magnetic field.

4.4.1 Interference and Calibration

A disadvantage of such a sensor is the interference of magnetic fields caused by neighboring wires. Therefore, calibration must be performed to increase the accuracy of the sensor readings.

The datasheet specifies that the current sensor output is half the voltage of V_{cc} when nothing is connected to the current sensor, meaning 2.5 [V] for our setup. However, due to the interference of external magnetic fields, this value of 2.5 [V] tends to change over time.

4.4.2 Calibration Process

To address this, we perform a calibration process:

1. **Recording Baseline Voltage:** We record the output voltage when no motor is powered over a set period of samples. This simulates the situation in which the sensor is not connected to any circuitry.
2. **Calculating the Offset:** After the recording period, we take the average voltage value and set this as our offset. This offset accounts for the variation in the baseline voltage caused by external magnetic fields.

4.4.3 Computing the Current

Knowing the offset of the current sensor, the current can be computed. The ADC of the Arduino Uno has a resolution of 10 bits, so the sensor's output voltage will be mapped to a digital value D between 0 and 1023. The output voltage of the current sensor V_{out} is then computed using:

$$V_{out} = D \cdot \frac{V_{cc}}{1023} \tag{4.1}$$

With $V_{cc} = 5[V]$ in our setup, we can then determine the current I drawn by the motor using the sensor's sensitivity, 66.0 [mV/A]:

$$I_{\text{sensor}} = 1000 \cdot \frac{V_{out} - \text{offset}}{66} \tag{4.2}$$

Where the factor 1000 is used to compute the current in amperes.

4.4.4 Noise Reduction and Signal Analysis

The integration of a current sensor aims to record the current step response and determine the resistance of the motor. However, we are recording the current of a PWM signal, which causes noise in the measurement as illustrated in fig. 4.7.

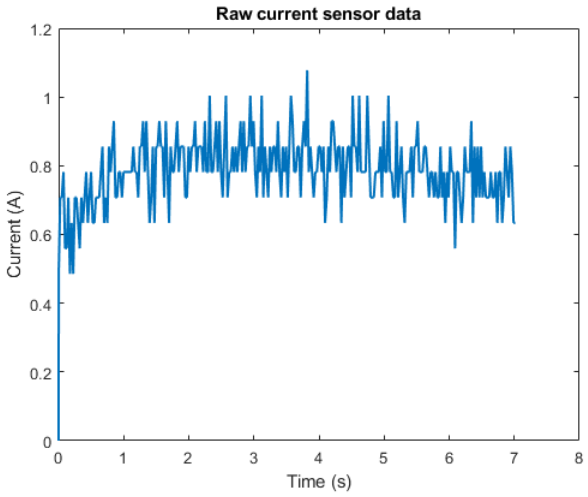


Figure 4.7: This figure shows a plot of the raw current sensor data after recording a step response.

As observed in the graph, the step response is discernible but contains significant noise. The noise can be attributed to the interference and the resolution limits of the sensor, causing spikes and fluctuations in the current

readings. While we intended to analyze each step response using a curve fit to directly correlate the data with eq. (3.10), the current noise level makes a curve fit unreliable. Consequently, we created an FFT to determine a suitable cutoff frequency for a filter, enabling noise reduction. The FFT is presented in fig. 4.8.

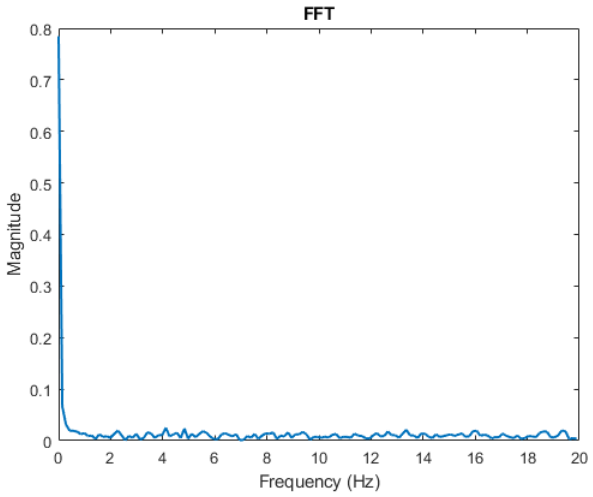


Figure 4.8: This figure shows an FFT of the raw current sensor data.

After analyzing the FFT, we decided to use a lowpass filter with a cutoff frequency of 1 Hz. The resulting filtered data is shown in fig. 4.9.

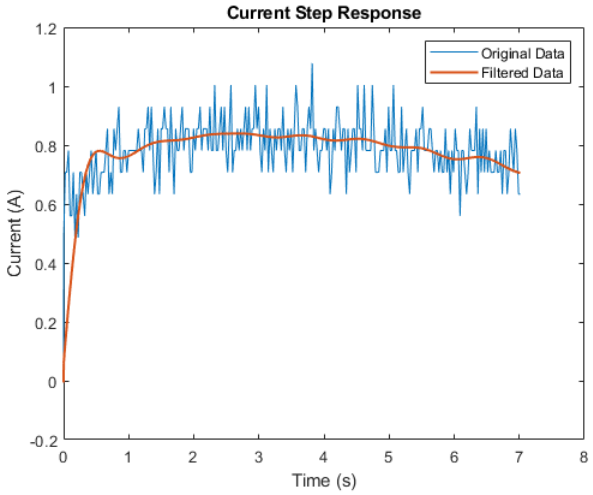


Figure 4.9: This figure shows a plot of the filtered current sensor data.

Figure 4.9 clearly shows that the step response is significantly clearer. We applied a curve fit to the filtered data to facilitate step response analysis and link the data to eq. (3.10). The resulting curve fit is displayed in fig. 4.10.

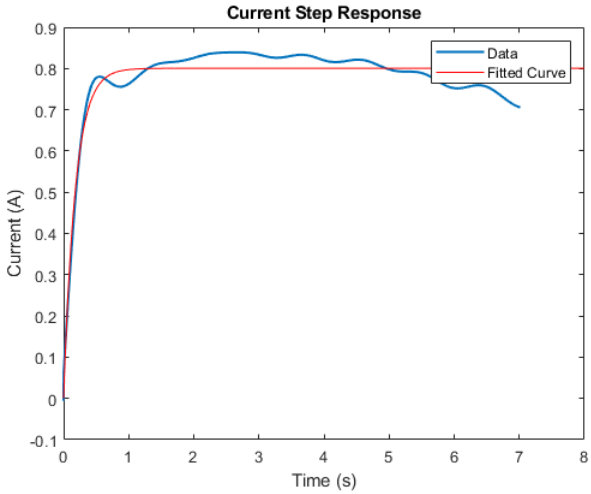


Figure 4.10: This figure shows a plot of the curve fitted, filtered, raw sensor data.

This curve can utilized to compute the time constant, which is identified at 63.2% of the maximum amplitude.

4.4.5 Resistance Measurement

In addition to recording a step response, a ramp response must also be recorded by the current sensor to compute the armature resistance of the MUT. Since we know the PWM during the ramp, the voltage can be estimated using the following equation:

$$V = \frac{\text{PWM}}{255} \cdot V_{supply} \tag{4.3}$$

Where V_{supply} is the voltage provided by the power supply. This equation allows us to estimate the voltage because the PWM (Pulse Width Modulation) signal is a method of reducing the average power delivered by an electrical signal using an on-and-off period [22]. The PWM value ranges from 0 to 255 in the Arduino Uno. This represents the duty cycle fo the signal as a fraction of 255. By scaling this value with V_{supply}, we can determine the effective voltage being applied to the motor. This way, both the current and voltage are known, enabling the calculation of the armature resistance can be calculated.

As previously stated, a ramp input is fed to the MUT to record the resistance. Throughout the measurement, the PWM values are converted to voltages and the current is read by the current sensor, after which the resistance is computed. After recording such ramp input, the following resistance graph can be created:

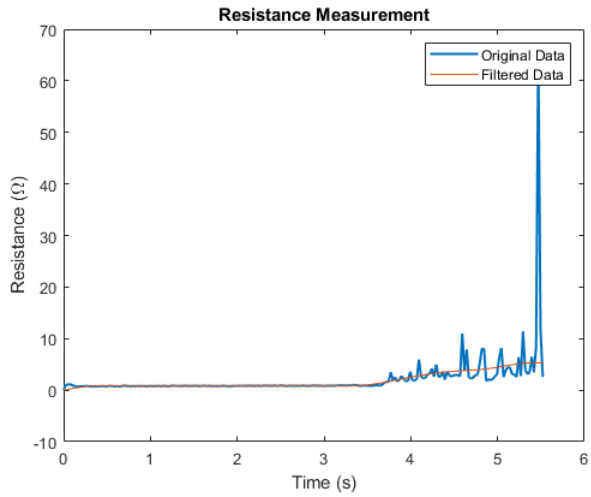


Figure 4.11: This figure shows a plot of the armature resistance over time.

From the figure, it is evident that filtering the data enhances the quality of the signal, reducing noise. However, there is still a significant deviation in resistance, which the thermal noise of the motor could cause, as higher temperatures lead to an increase in resistance. By taking the average value of the resistance, the armature motor resistance is determined.

4.5 The Encoder

The encoder within the setup is used to measure the velocity at which the motor axle is spinning. This allows us to measure the velocity step response of the motor, which can be used later for the mechanical properties of the motor. After providing a step to the motor, the raw data is curve-fitted. Leading to fig. 4.12

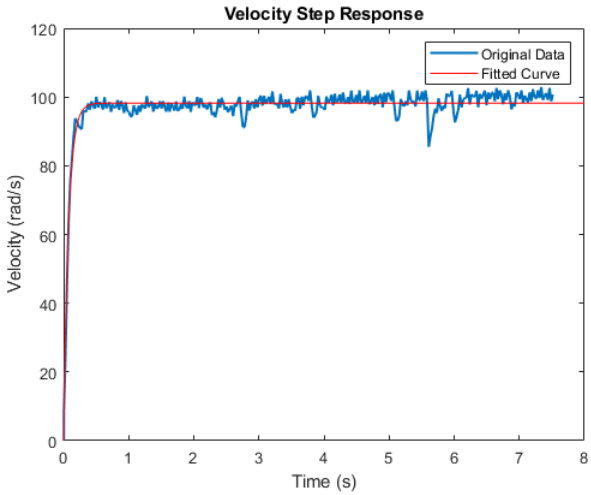


Figure 4.12: This figure shows a plot of the raw and curve-fitted encoder velocity data.

From the graph, it becomes obvious that the raw encoder data is already enough to create a proper curve fit. The curve fit can then be used to compute the time constant of the mechanical domain by taking 63.2% of the maximum value and extracting the corresponding time value.

4.6 Data Capturing

Having an idea on how the setup is going to be built physically, it is time to delve into data capturing. To successfully capture the data from each measurement, the Arduino Uno will communicate with a Python script, which runs in the background of an arbitrary laptop. This communication is done through a serial connection. The downside of this is that the Arduino must be connected to the laptop at all times. On the other hand, this does imply that the connection is as direct as possible without the added delays, which wireless connections cause frequently.

By sending flags and the variables of interest over the serial connection, the Python script knows when to start a new subplot and when to start and stop recording data from the Arduino. As a safety feature, the data will also be stored in a .xlsx file from Excel. This way, whenever the final plot is closed on accident, the user can rerun another script called "curvefit.py" to re-obtain the plots.

Within "curvefit.py", the user can select the plots that need to be curve-fitted and filtered at a selected frequency. Here, I opted to curve fit both velocity and current step responses to remove noise and match the data points to the expected theoretical signal. In addition, before curve fitting the current step response, the measurement is first parsed through a low pass filter at a cut-off frequency of 1 [Hz] to clean the data and remove noise. Finally, this Python script also indicates the time constants and their corresponding values within the plots of the step responses.

4.7 The Final Setup

Knowing how everything will be put together, the setup can be built. The final setup is shown in fig. 4.13.

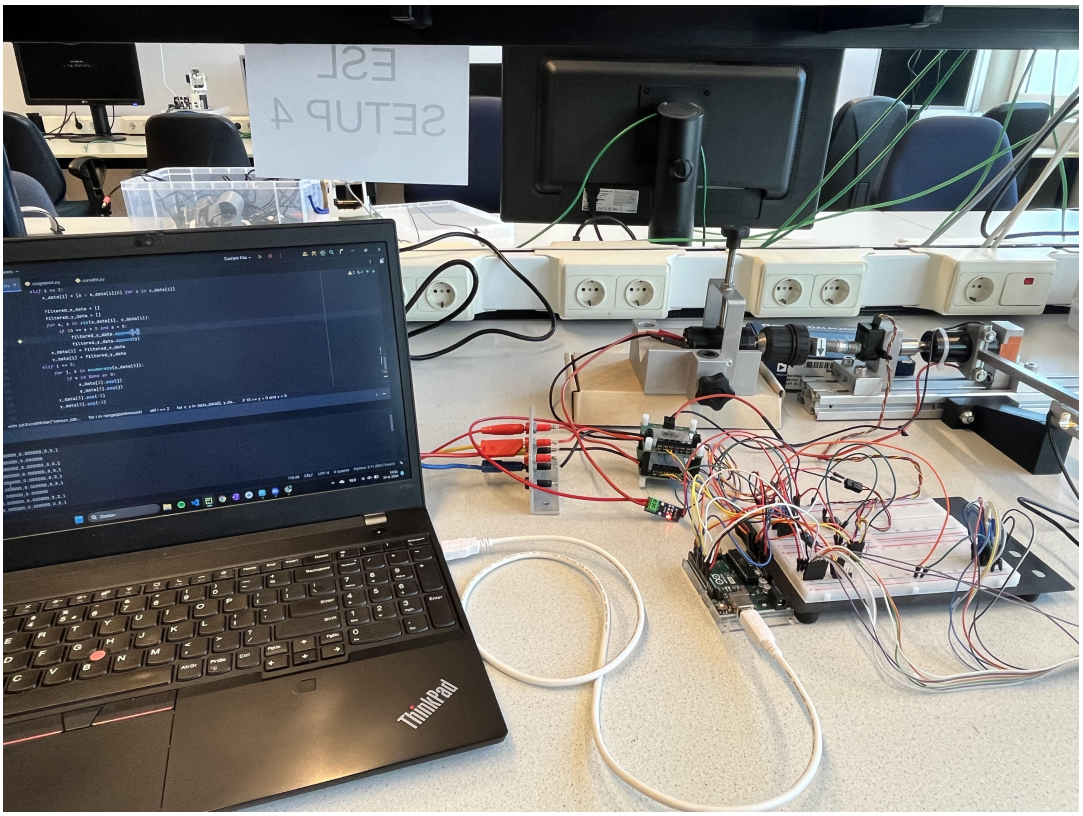


Figure 4.13: This figure shows the final setup.

As can be seen from fig. 4.13, the plate to which the old setup was mounted is missing. Due to time constraints, such a foundation could no longer be created, causing instability in the mechanical construction.

4.7.1 Data flow

To summarise the functionality of the setup, a flow chart is created (please refer to fig. 4.14.

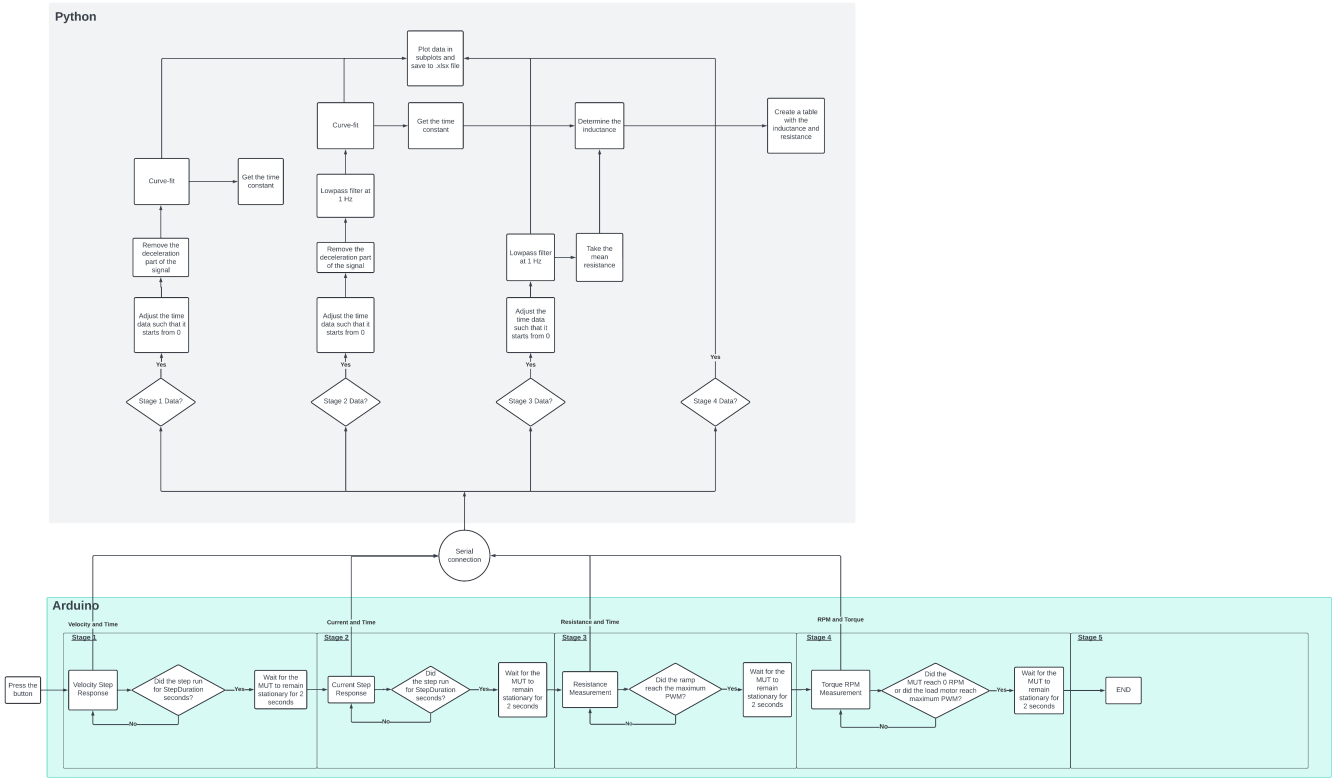


Figure 4.14: This figure shows the flow chart.

As shown in the graph, we start the characterization process by pressing a button. This button press initiates stage one of the measurement. Here, we record a velocity step response, after which the data is transmitted over a serial connection to the Python script. After every stage, we wait until the MUT has been stationary for two seconds such that previously executed measurements do not interfere with the current measurement.

After the delay of two seconds, the stage number is incremented to stage two. Here, we record the current step response and transmit the corresponding data through a serial connection to the Python script. After the measurement, we wait for two seconds once the MUT is stationary and proceed to stage three.

In stage three, we feed a ramp to the MUT, during which we record and transmit the resistance value. If the ramp reaches the maximum PWM, we stop the measurement and wait for two seconds again once the MUT is stationary.

During the final stage, stage four, we measure the torque versus RPM curve by feeding the nominal input voltage, a maximum PWM input, to the MUT, and a ramp to the load motor as described in previous sections. The torque and RPM are transmitted over the serial connection to the Python script during this measurement.

Chapter 5

Evaluation

After building the complete setup, it is time to write the software that allows automatic motor characterization through a single button press. Due to time constraints, unfortunately, the motor constant, inertia, and friction could not be determined. However, both velocity and current step responses are analyzed and their corresponding time constants are plotted as well. In addition, the armature resistance is recorded, meaning that the inductance could be found through the mechanical time constant. Finally, the RPM vs torque graph is plotted as well. On that note, future addition of viscous and coulomb friction estimation will empower to complete of the project because when the friction is known and the time constant, the inertia can be determined as well. The upcoming sections will discuss the measurements and the data processing tools that are utilized within this project.

5.1 Velocity Accuracy

After performing the velocity step response, we obtained the results shown in fig. 4.12. Upon examining the steady state of the step response, we observed minor spikes. These small spikes result from taking the discrete derivative during each iteration, which introduces quantization error due to the encoder's resolution of 800 counts per revolution.

More significant spikes were observed at approximately three and six seconds. These spikes occurred because the Arduino Uno missed a few encoder counts. This issue arose because the loop frequency was lower than the frequency at which new encoder counts were being registered, causing the microcontroller to occasionally miss updates from the encoder. To clarify, the encoder generates pulses at a rate that the Arduino's loop could not always keep up with, leading to occasional missed counts and resulting in larger spikes in the velocity data.

5.2 Current Accuracy

In Section 4.4.4, we discussed that we opted to filter the current sensor data through a low-pass filter set at 1 Hz to reduce noise. Since we are using an ACS712 30A Hall-effect current sensor, external magnetic fields can affect our measurements. The types of noise present in our current data are:

- Electromagnetic Interference (EMI)
- Quantization Noise

5.2.1 Electromagnetic Interference (EMI)

Hall-effect current sensors, such as the ACS712, operate by measuring the magnetic field generated by the current flowing through a conductor. However, these sensors are also susceptible to interference from external magnetic fields. Such interference can originate from nearby electrical devices, power lines, and other components in the circuit, inducing unwanted voltages and causing inaccuracies in the current measurements.

5.2.2 Quantization Noise

To determine the quantization noise, we need to consider the 10-bit ADC resolution of the Arduino Uno. The 10-bit ADC resolution constrains the lowest read voltage change by the Arduino to:

$$\Delta V = \frac{5 \text{ V}}{1024} \approx 4.88 \text{ mV} \tag{5.1}$$

We can use this voltage step, ΔV, and the ACS712's sensitivity of 66 mV/A to determine the lowest measurable current:

$$\Delta I = \frac{4.88 \text{ mV}}{66 \text{ mV/A}} \approx 0.074 \text{ A} \tag{5.2}$$

Thus, the quantization noise introduces a minimum current measurement step of approximately 0.074 A. This quantization noise is inherent to the ADC's resolution and affects the precision of the current measurements obtained from the ACS712 sensor.

5.3 Force Accuracy

To measure the force applied to the load cell, the output of the Wheatstone bridge is first passed into a 24-bit ADC converter, which sends its signals over an SPI connection to the Arduino Uno. This high-resolution ADC allows for precise measurements of the load cell's output voltage, converting it into a digital number between 0 and $2^{24} - 1$.

5.3.1 ADC Resolution and Quantization Noise

The 24-bit ADC provides a significant increase in resolution compared to the 10-bit ADC of the Arduino Uno. The voltage resolution ΔV of the ADC is given by:

$$\Delta V = \frac{V_{ref}}{2^{24}} \tag{5.3}$$

Where V_{ref} is the reference voltage used by the ADC. In our case, we use a reference voltage of 5 [V]. This means that the smallest voltage change the ADC can detect is approximately 0.298 μV. Using eq. (3.6), we conclude that the minimum change in force is 0.00146169 [N].

5.3.2 Validation of the resolution

To verify the precision of the load cell, we used a dynamometer to apply known pressures and then recorded the measured values from the Arduino. Multiple measurements were taken, and the results are plotted below.

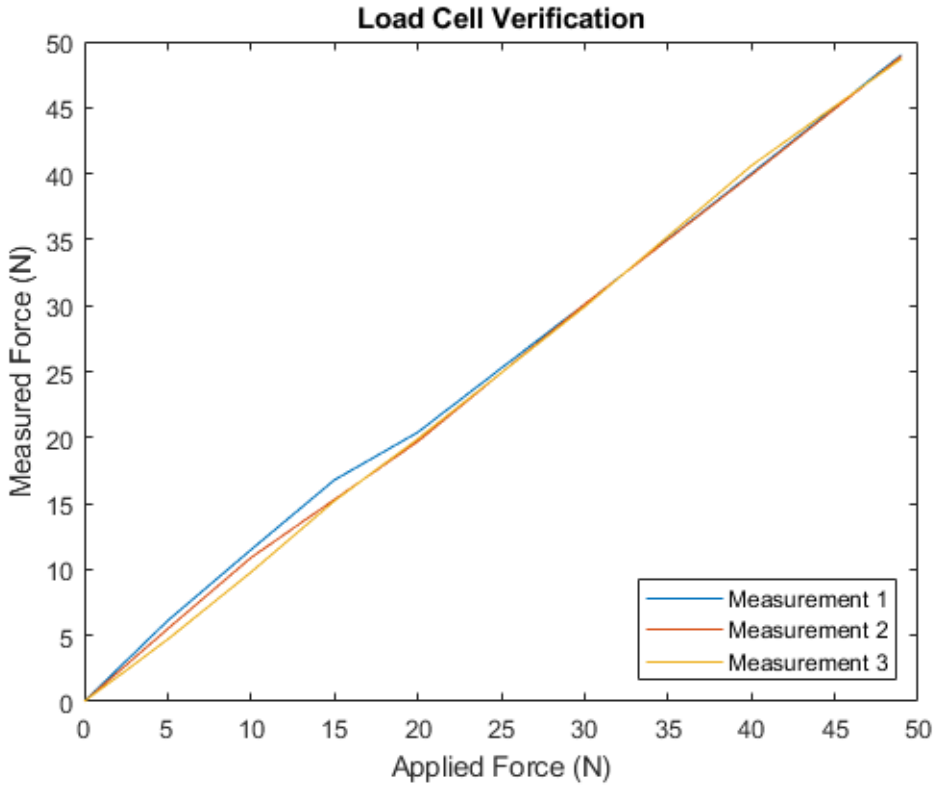


Figure 5.1: This figure shows the obtained graph after recording the measured force while the applied force is known.

Observations and conclusion:

Upon analyzing the plotted data and considering the load cell's specifications from the datasheet [18], several observations can be made:

- **Accuracy Threshold:** The discrepancies observed, particularly below 20 N, align with the load cell's specified sensitivity range. As indicated by the datasheet, the load cell operates optimally between 19.7 N and 49.05 N due to its rated capacity of 2-5 kg.
- **Resolution Consideration:** The 24-bit ADC used in our setup provides a high resolution, enabling precise measurement of the load cell's output voltage. With the smallest detectable voltage change of approximately 0.298 µV, corresponding to a minimum force change of 0.00146169 N (as calculated from Equation 3.6), minute variations in force can be accurately captured. This explains why at certain applied forces, such as 30 N, 40 N, and 49 N, the recorded differences were minimal and may not be visually apparent without careful analysis.
- **Human Error:** It is important to note that human error in reading the applied force could contribute to discrepancies, especially when the differences are subtle. The use of precise instrumentation and careful calibration procedures mitigates such potential errors.

In conclusion, despite minor deviations observed at lower forces, the load cell and measurement system demonstrate accurate and reliable performance within their designed operational range. The high-resolution ADC effectively captures fine force variations, affirming the load cell's functionality and confirming its suitability for precise force measurement applications.

5.4 The Results

The goal of this project was to identify the characteristics of an unknown PMDC motor automatically. Within the given time, as previously mentioned, the step responses and their corresponding time constants are identified just as the resistance and inductance. Also, it was attempted to record a proper RPM vs torque graph. After running the testbench, the following graph is created:

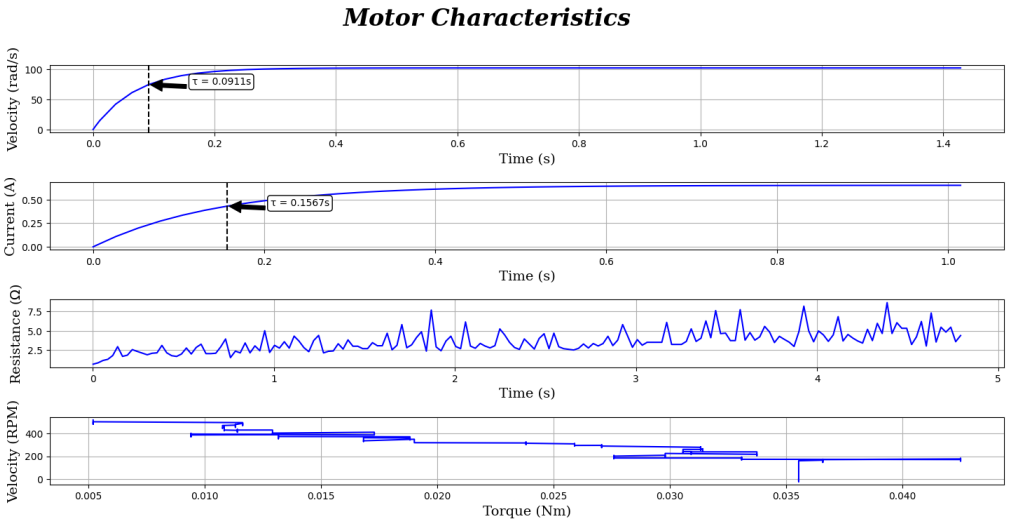


Figure 5.2: This figure shows the graphs that are obtained after running the final testbench.

5.4.1 The Step Responses

As can be seen in fig. 5.2, the first two plots are the velocity and current step responses. Within the step responses, a vertical dashed line indicates the location of the time constant, and the corresponding value is shown on the right of the dashed line.

As discussed within the literature review section, the inductance of the motor is often mitigated while modeling a DC motor as the electrical domain is much slower than the mechanical domain. We also observe this in fig. 5.2.

5.4.2 The Resistance

Within the third plot of fig. 5.2, significant noise is observed in the resistance measurements, which fluctuates noticeably. As discussed earlier, this noise primarily originates from two sources: the limited resolution of the Arduino's 10-bit ADC and electromagnetic interference (EMI).

Additionally, it is important to note that during the measurement process, the motor undergoes heating. As the motor temperature increases, the resistance of its windings also rises gradually. This thermal effect is reflected in the plot, where the resistance values show an upward trend over time.

5.4.3 Torque versus RPM

The final plot in fig. 5.2 illustrates the measured torque versus RPM graph. In section 4.3.3, we learned that the torque measurement was not working as expected. Therefore, validation had to be done to identify potential sources of error

Suspected Sources Of Error

The decreasing torque can be caused by vibrations of the physical setup, meaning that the arm, to which the load motor is attached, moves up at certain points in time. During the measurements, it is often observed that the arm indeed vibrates if not enough pressure was applied to the setup (to reduce vibrations as much as possible).

Another observation was that the load motor moves during the measurements. the axle of this motor is attached through a bearing, allowing to keep the stator free such that the stator is able to apply force on the load cell. However, there is quite some play for the motor to move back and forth, which could affect the measurements as well.

The next observation is that the bold which is inserted in the motor coupling mechanism as shown in fig. 5.3 causes an unbalanced axle.

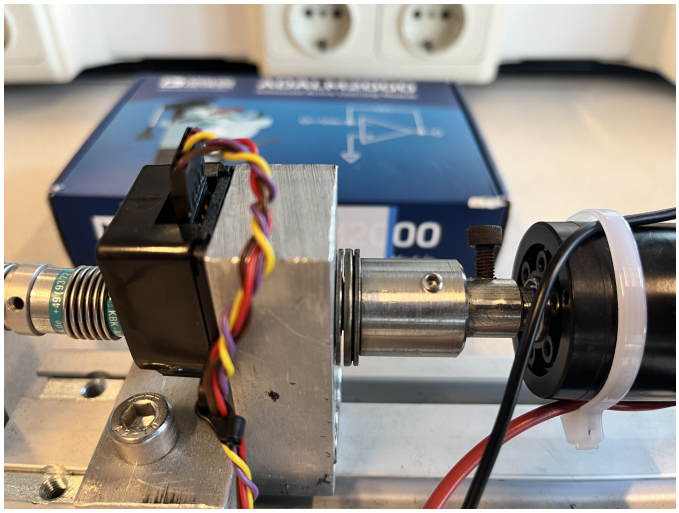


Figure 5.3: This figure shows the motor coupling of the setup.

Having an unbalanced axle, the center of mass is no longer in the middle of the axle. Therefore, when the MUT starts spinning and the bold approaches the top of the axle, the MUT must exert more force than it would when the bold is on the bottom of the axle. The extra weight brought by the bold also makes it so that the motor axle accelerates slightly at the point where the bold goes towards the bottom of the axle, and decelerates when the bold goes from the bottom of the axle to the top.

Finally, the flexible coupling mechanism between both motor axles, which is shown in fig. 5.4, could add extra friction if the axle of the MUT and load motor are misaligned.

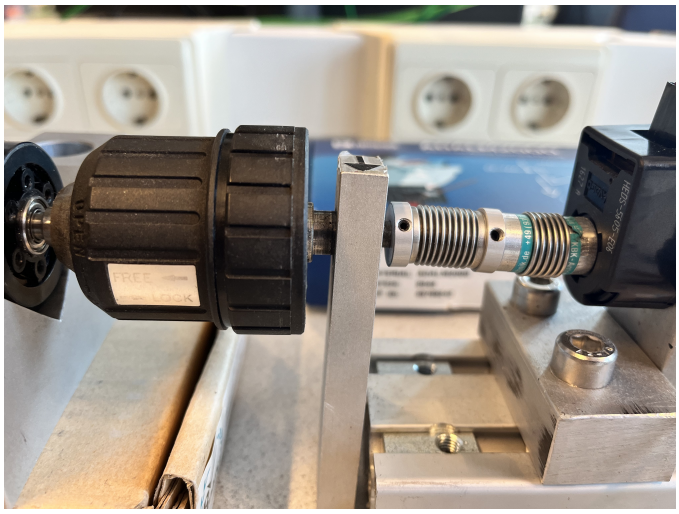


Figure 5.4: This figure shows the flexible coupling mechanism of the setup.

Confirming The Error

To confirm that the mechanical side of the setup is indeed the main cause of our RPM vs torque plot problem, another measurement was performed. For this experiment, a step was fed to both MUT and active load. However, the MUT received a step with a higher PWM value such that the MUT can spin at a constant speed and torque. Next, a torque versus motor angle plot was created, please refer to fig. 5.5.

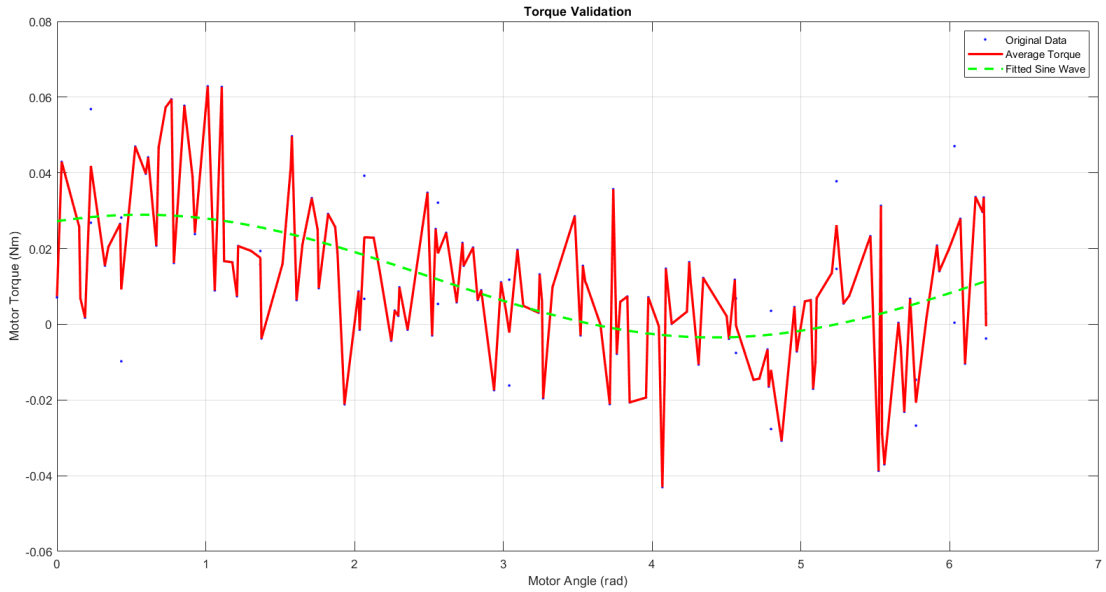


Figure 5.5: This figure shows the torque versus motor angle.

During this measurement, the motor went through multiple revolutions, implying that torque data was collected for multiple cycles to enhance the accuracy of the plot. The blue dots in the figure represent raw data points, the red line is the average torque for every uniquely measured motor angle. Finally, a curve fit was performed to show the sinusoidal relationship that is exhibited by the mechanical construction.

As can be observed from fig. 5.5, the torque has a sinusoidal relationship with the motor angle. This implies that the axle is not balanced in a manner in which it can rotate freely. This could be due to the added mass from the bold, or misalignment of both motor axles as discussed previously. Ultimately, due to the sinusoidal behavior of the torque, we conclude that the mechanical construction, in its current status, causes vibrations at the load cell.

5.4.4 Motor Characteristics

We use the armature resistance and electrical time constant to compute the inductance through eq. (3.11). The table with the measured resistance and inductance is shown in table 5.1.

Parameter	Value
Resistance (R)	3.6416 Ω
Inductance (L)	0.5707 H

Table 5.1: This table shows the found motor parameters.

From the table, it becomes evident that the resistance has a value of 3.64 [Ω] and the inductance a value of 0.57 [H]. If we relate these values to the enlisted 1.11 [Ω] and 0.2 [H], we observe that the found values are higher than expected.

The higher resistance can be explained by the noise in the current sensor and the thermal heating of the motor throughout the measurement, as we discussed in a previous section.

The method used for the inductance measurement assumes an ideal situation in which we deal with a first-order RL circuit. From the current step response, we then used the time constant to determine the inductance. However, this proposed method assumes no noise or transient effects such as on and off switching of the motor, which we do cause due to the inserted PWM signal. Therefore, the found resistance and inductance values are higher than expected.

5.5 Future Recommendations

5.5.1 Mechanical Redesign for Stability

- **Issue:** The current mechanical setup exhibits vibrations and an unbalanced axle, leading to inaccuracies in torque measurements and potentially affecting other measurements like RPM vs torque.
- **Recommendation:** Redesign the mechanical structure to minimize vibrations and ensure the motor axle is balanced. Consider improving the coupling mechanisms (e.g., flexible coupling) to reduce misalignment issues.

5.5.2 Improved Current Sensing

- **Issue:** The ACS712 30A current sensor, coupled with the Arduino Uno's 10-bit ADC, introduces quantization noise and limited resolution.
- **Recommendation:** Upgrade the current sensing system by either:
  - **Higher Resolution ADC:** Implementing a higher-resolution external ADC for the ACS712 sensor to enhance precision and reduce noise.
  - **Alternative Current Sensing Method:** Exploring alternative current sensing methods with higher accuracy and noise immunity, such as dedicated current measurement ICs.

5.5.3 Transmitting Data to Python

- **Issue:** Continuous data transmission over a serial connection from Arduino to Python leads to a reduced loop frequency in the Arduino code, causing missed encoder counts and other timing-related issues.
- **Recommendation:** Enhance data transmission efficiency by implementing the following strategies:
  - **Buffer Implementation:** Modify the Arduino software to include a buffer that accumulates data in memory during each measurement cycle. Once the measurement is complete, transmit the accumulated data in a single batch rather than continuously. This approach minimizes the number of serial prints to only four per cycle, reducing the impact on loop frequency and improving timing accuracy.
  - **Alternative Microcontroller:** Consider migrating to a more powerful microcontroller with a higher processing speed than the Arduino Uno. This upgrade can handle data transmission tasks more efficiently while maintaining adequate loop frequency for precise encoder count tracking and other real-time operations.

5.5.4 Complete Motor Parameter Characterization

- **Issue:** Due to time constraints, parameters like motor constant (K), inertia, and friction were not fully characterized.
- **Recommendation:** Complete the motor characterization by:
  - **Determine Motor Constant and Inertia:** Conducting thorough testing to determine the motor constant (K) and inertia using methods such as load torque versus acceleration or deceleration.
  - **Determine Friction:** Implementing additional tests or simulations to accurately quantify friction and its impact on motor performance.

5.5.5 Add Fine Tuning

- **Issue:** Due to time constraints, the addition of the non-linear least squares method was not added to the automated system. The absence of this tool leads to less accurate parameters.
- **Recommendation:** Integrate the non-linear least squares algorithm such that the parameters can be fine-tuned once all motor parameters are obtained from the automated system.

Chapter 6

Conclusion

In this study, we addressed the primary question driving our investigation: which parameters are crucial in the electro-mechanical model of a DC-motor, and how can they be identified experimentally? By delving into this question, we aimed to contribute to a more efficient and reliable method for parameterizing DC-motors.

Throughout the paper, we discussed various parameters that are important to DC motor characterization. Parameters revolving around the motor constants, the armature properties of the electrical domain, and mechanical characteristics were discussed and their relations to one another were enlisted.

In addition, we reviewed various methods of parameterization, including the least squares method, the acceleration method, and the pattern search method. We compared their accuracy and complexity, ultimately identifying the least squares method as the most reliable method for PMDC motor characterization.

Furthermore, we designed an automated system using step responses to record the electrical and mechanical time constants and a ramp response to determine the armature resistance of the MUT. From these measurements, we calculated the inductance, leveraging the resistance and electrical time constant.

Next, our analysis of sensor accuracy revealed that velocity measurements were highly accurate, despite occasional missed encoder counts by the Arduino due to a low loop frequency, which introduced random spikes in the data. The current sensor readings exhibited significant noise, attributed to quantization noise and electromagnetic interference (EMI). For the force sensor, we observed a sinusoidal relationship between torque and motor angle, indicating mechanical vibrations and added friction. Despite these challenges, additional filtering and curve fitting allowed us to obtain usable data for the time constants and resistance.

In addition to the time constants, resistance, and inductance of the MUT, the torque versus RPM curve was plotted as well. Due to mechanical vibrations and additional friction, we discovered that the torque measurements were malfunctioning. Hence, we concluded that the current mechanical setup could not be used to perform torque measurement, leaving us unable to measure friction, inertia, and motor constants.

For future work, we recommend redesigning the mechanical setup for improved stability. This should allow the user to obtain precise torque readings and a torque versus RPM graph. In addition, the performance of the current sensor can be upgraded by either incorporating an external ADC with a higher resolution than the 10-bit ADC from the Arduino Uno or by using an alternative current sensing method. Next, we recommend increasing the loop frequency of the Arduino Uno by reducing the number of serial prints during operation or using a more powerful microcontroller. Finally, implement methods to accurately measure inertia, friction, and motor constants, followed by a fine-tuning process to refine the parameter estimates. These improvements will ensure precise motor parameter estimation and complete the project, providing a robust and automated system for DC motor characterization. Even though the motor parameters found during this project were higher than expected, this study has laid the groundwork for future enhancements and serves as a significant step toward achieving efficient and accurate motor control in various applications.

Appendix A

Appendix

A.1 AI statement

During the preparation of this work the author(s) used the chatbot ChatGPT based on GPT 3.5 to resolve bugs in the code and to improve the grammar in the report. During the preparation of this work the author(s) used Grammarly to improve the grammar used in the report. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the work.

Bibliography

[1] HEDM-55xx/560x & HEDS-55xx/56xx Quick Assembly Two and Three Channel Optical Encoders Data Sheet. Datasheet. Available online: `https://nl.mouser.com/ProductDetail/Broadcom-Avago/HEDS-5605E06?qs=RuhU64sK2%252Bt2DYaQX1Xelw%3D%3D`.

[2] Arslan Ahmed Amin Muhammad Adnan Ashna Batool, Noor ul Ain and Muhammad Hamza Shahbaz. A comparative study of dc servo motor parameter estimation using various techniques. *Automatika*, 63(2):303–312, 2022.

[3] Murtadha Awoda and Ramzy Ali. Parameter estimation of a permanent magnetic dc motor. *Iraqi Journal for Electrical and Electronic Engineering*, 15(1):28–36, Jun 2019.

[4] Ward Brown. Brushless dc motor control made easy. *Microchip Technology Inc*, 1, 2002.

[5] Stijn Derammelaere, Michiel Haemers, Jasper De Viaene, Florian Verbelen, and Kurt Stockman. A quantitative comparison between bldc, pmsm, brushed dc and stepping motor technologies. pages 1–5, 2016.

[6] Barış Doğan, Ali Buldu, and Hasan Erdal. Realtime parameter estimation, calibration and simulation of a dc motor. *Technics Technologies Education Management*, 6:606–614, 01 2011.

[7] Douwe Dresscher, Theo J. A. de Vries, and Stefano Stramigioli. Motor-gearbox selection for energy efficiency. pages 669–675, 2016.

[8] Mohamad Farid Fazdi and Po-Wen Hsueh. Parameters identification of a permanent magnet dc motor: A review. *Electronics*, 12:x, 2023.

[9] Rithvik Gambhir and Akshay Kumar Jha. Brushless dc motor: Construction and applications. *Int. J. Eng. Sci*, 2(5):72–77, 2013.

[10] Aaron M Harrington and Christopher Kroninger. Characterization of small dc brushed and brushless motors. *ARMY RESEARCH LAB ABERDEEN PROVING GROUND MD VEHICLE TECHNOLOGY DIRECTORATE, Tech. Rep*, 2013.

[11] H. O. Hartley and Aaron Booker. Nonlinear Least Squares Estimation. *The Annals of Mathematical Statistics*, 36(2):638 – 650, 1965.

[12] Swaraj Ravindra Jape and Archana Thosar. Comparison of electric motors for electric vehicle application. *international Journal of Research in Engineering and Technology*, 6(09):12–17, 2017.

[13] Massachusetts Institute of Technology. 2.004 dynamics and control ii: Lecture 2 - solving the equation of motion, fall 2007. Lecture notes.

[14] Nicholas Morozovsky, Robert Moroto, and Thomas Bewley. Rapid: An inexpensive open source dynamometer for robotics applications. *IEEE/ASME Transactions on Mechatronics*, 18(6):1855–1860, 2013.

[15] Jacob Pedersen. *Model Based and Robust Control Techniques for Internal Combustion Engine Throttle Valves*. PhD thesis, 12 2013.

[16] Sander Roodink, J.F. Broenink, T.G. Broenink, and N. Meratnia. Large-range active load for dynamic motor testing. BSc Report 024RAM2017, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands, July 2017. Supervised by Dr.ir. J.F. Broenink and Dr. N. Meratnia.

[17] Mohammed SZ Salah. Parameters identification of a permanent magnet dc motor. *The Islamic University of Gaza*, 94, 2009.

[18] Tedea-Huntleigh. Model 1006 aluminum single-point load cell, February 2018. Document No.: 12003, Revision: 23-Feb-2018.

[19] Virginia Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1):1–25, 1997.

[20] Claudio Urrea and John Kern. A new model for analog servo motors. simulations and experimental results. *Canadian Journal on Automation, Control and Intelligent Systems*, 2(2):29–38, 2011.

[21] L. L. Wang, J. X. Shen, P. C. K. Luk, W. Z. Fei, C. F. Wang, and H. Hao. Development of a magnetic-geared permanent-magnet brushless motor. *IEEE Transactions on Magnetics*, 45(10):4578–4581, 2009.

[22] Z. Yu, A. Mohammed, and I. Panahi. A review of three pwm techniques. 1:257–261 vol.1, June 1997.