AALBORG UNIVERSITY

STUDENT REPORT

ED4-1-F16

Modelling and Control for a Ball and Beam System

Students:
Alexandra Dorina Török
Allan Gjerlevsen
Andrius Kulšinskas
Emil Már Einarsson
Thomas Thuesen Enevoldsen

Supervisors:
Christian Mai
Kasper Jepsen

May 24, 2016

AALBORG UNIVERSITY

STUDENT REPORT

School of Information and Communication Technology
Niels Bohrs Vej 8
DK-6700 Esbjerg
http://sict.aau.dk

Title:
Modelling and Control for a Ball and Beam System

Theme:
Scientific Theme

Project Period:
Spring Semester 2016

Project Group:
ED4-1-F16

Participant(s):
Alexandra Dorina Török
Allan Gjerlevsen
Andrius Kulšinskas
Emil Már Einarsson
Thomas Thuesen Enevoldsen

Supervisor(s):
Christian Mai
Kasper Jepsen

Copies: 3

Page Numbers: 93

Date of Completion:
May 24, 2016

Abstract:

This report goes into detail about the implementation of a classical control solution for a ball and beam system. The main objective of the control system is to stabilize a ball around a given set point, where the controller is also able to handle disturbances. The system is a SISO (Single Input - Single Output) system and uses the mathematical modelling of the physical setup in order to theoretically design the controllers. The system consists of a single DC-motor as an actuator and the National Instruments PCI-6229 data acquisition card, through which the controller will be implemented and used together with Simulink Desktop Real-Time. Different experiments were performed to gather the motor coefficients and ball positional data in order to be able to use the mathematical models. The final chapters of the report cover the comparison between the developed theoretical control solution and the real life implementation.

Contents

Preface

The project entitled *Modelling and Control for a Ball and Beam System* was made by five students from the Electronics and Computer Engineering programme at Aalborg University Esbjerg, for the P4 project during the fourth semester.

From hereby on, every mention of 'we' refers to the five co-authors listed below.

Aalborg University, May 24, 2016.

Allan Gjerlevsen
<agjerl13@student.aau.dk>

Emil Már Einarsson
<eeinar14@student.aau.dk>

Andrius Kulšinskas
<akulsi14@student.aau.dk>

Alexandra Dorina Török
<atarak14@student.aau.dk>

Thomas Thuesen Enevoldsen
<tten14@student.aau.dk>

Chapter 1

Introduction

1.1 Introduction

A control system is a mechanism that alters the future behaviour of a system. In order for the system to be considered a control system, its behaviour must tend towards a desired state. All control systems consist of two basic components: the system to be controlled, also known as the plant, and an input for the plant. The input acts on the plant, which responds over time to produce an output. This type of control system is called an open loop system because the input does not depend on the output of the plant. A closed loop system uses the concept of an open loop system, but it has an added path between the output and the input. Figure 1.1 gives a visual understanding on the working principle of a closed loop control system. The output is measured with a sensor, the result of this measurement is then compared with the desired state, also known as the reference. This comparison generates an error term, which is then used as an input value for the controller. [1]

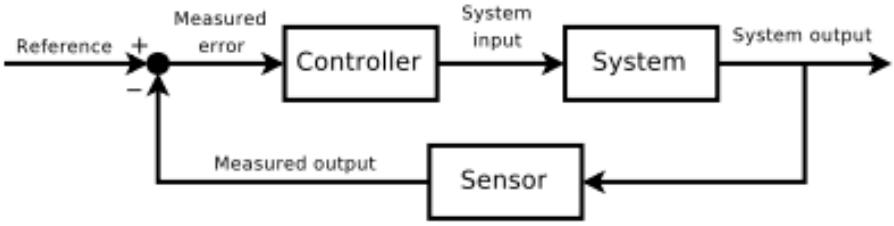


Figure 1.1: Block diagram of closed loop control system [2]

In order to fulfil the semester's requirements, we had to choose a project theme within the field of control theory and implement a control system. The initial project ideas were the following:

- Drone stabilization
- 3 axis camera gimbal
- Ball balancing on a two-dimensional surface

- Ball and beam system

The titles of all the project ideas mentioned above are self-explanatory. Each one of them would involve working on a closed loop control system, which would be enough to fulfil the requirements. As far as the Drona stabilization idea goes, the reason why we chose notto proceed with it was due to the high risk of damaging the prototype during the testing phase. The camera gimbal stabilisation idea seemed to be too challenging from both hardware and theoretical point of view, taking into consideration the fact that we do not have any previous knowledge on control theory. Balancing a ball on a two-dimensional surface was also disregarded because building the hardware setup would have taken a lot of time and the lack of resources would have slowed down the project significantly. Finally, our decision was to choose the ball and beam system due to its capability in terms of control implementation. Another benefit of this choice was that the university had an old setup from a previous project, which meant that we could spend more time focusing on applying control theory instead of building the hardware platform.

The goal of the project is to be able to control the position of the ball on the beam, by using the angle of the beam, controlled by a single motor. Powering the motor would change the angle of the inclined plane that the beam creates and as a result, the ball will roll. Stopping the ball from rolling in order to get it in the desired position would mean having to change the beam angle again. The system we plan to design is SISO. However, the setup we are using has two crank arms mounted at each end of the beam, this means system can be modified to become MISO (Multiple Input - Single Output) if necessary.

Chapter 2

Problem Description

In order to create and implement a control system, it is necessary to figure out the scope and determine the initial approach towards creating the solution. Therefore, this chapter will present the different methods in which a controller for a ball and beam system can be designed and implemented. It will take a look at how to manipulate and change the angle of the beam, how to determine the position of the ball and also what hardware and software components are required in order to create a complete solution.

2.1 Physical setup

Since the setup we plan on using already has two motors attached to it, there are two possibilities of controlling the angle of the beam.

- **Single motor system** As its name suggests, using this type of system implies having only one rotating motor. This motor will be connected to one end of the beam, while the other end will be fixed, so that when the motor rotates, the position of the beam changes based on just one side of the system.
- **Dual motor system** Using this system implies having both ends of the beam connected to an individual rotating motor. While this system would allow us to increase the speed of the ball movement, it would also make the control part more challenging.

Taking into account our yet limited experience within control theory, we have decided to use just one motor, creating a single motor system.

2.2 Ball position determination

The position of the ball can be determined in several ways. Some possibilities were discussed such as using ultrasonic and infrared sensors, but none of these ideas seemed worth trying since they present some major drawbacks. An ultrasonic sensor emits sound waves, these sound waves are reflected by the detected object and received back to the sensor. The problem with the ultrasonic sensor is that it has a wide detection range, especially if the object is placed at a greater distance. Therefore, if used, the ultrasonic sensor would have had a high chance of detecting objects other than the ball. As for the infrared sensor, it has a similar working principle as the ultrasonic one, having an infrared LED as an emitter and an infrared photodiode as a receiver. The receiver is sensitive to infrared light of the same wavelength as the one emitted by the infrared LED. This means that in certain light circumstances, the sensor could be inaccurate in its readings. Also, its accuracy in terms of reflecting the light depends on the surface of the ball, where in this case a shiny metal ball is used.

The most suitable choice for position detection of the ball is to use the beam as a resistive sensor. Because the beam is made out of two conductive rails, if a current is passed through one of them, the metal ball acts as a connector. Moving the ball across the beam results in a proportional change of voltage compared to the distance. Knowing the mathematical relationship between the voltage and the distance allows us to determine the position of the ball.

2.3 Control methods

Control theory can be divided into two branches: classical control and modern control. Classical control often deals with SISO systems using frequency domain tools. This means taking the Laplace transform of the differential equations defining the system and designing the controller based on certain performance specifications. The most popular type of designed controller using classical control is PID (Proportional - Integral - Derivative). Modern control deals with MIMO systems using the differential equation in the time domain. Modern control theory is also known as state-space based and it utilizes vectors and matrices in order to monitor the states of the control system. In modern control, controllers have good stability characteristics without using too much actuation [3].

Since its already been established that the project will use just one motor, our system can be labelled as SISO. Therefore, the chosen theory will be the classical control theory.

2.4 Controller implementation

There are different ways in which the implementation of the controller can be achieved. During the initial discussions of the project, using an Arduino or a MSP430 microprocessor was suggested, due to previous working experience and familiarity with the platforms. Both microprocessors have the required ports - analog I/O and Pulse Width Modulation (PWM) output. By using the hardware packages in Simulink, it is possible to create a control system and interface with Arduino or manually write a controller using C for the MSP430. The constraint to be taken into consideration when thinking of using one of the two is their 10-bit ADC, which is limited in terms of bits. This would work for some slow processes, but, for the purpose of our project, a better alternative was available in the lab on campus. The final decision was to use the data acquisition card from National Instruments (PCI-6229) for faster sampling rates and higher accuracy. A benefit of this particular card is that it allows direct interfacing with Simulink Desktop Real-Time.

Chapter 3

Physical Setup

The physical setup used for this ball and beam stabilization project is from an old bachelor project from 2005. Since the platform has the capabilities to allow dual motor stabilization, the one moving end has been fixed to a single position, so that it can be used as a single motor ball and beam setup. It utilizes a power drill motor as an actuator for the control system. The motors require 12V and runs at 450RPM (revolutions per minute). In order to determine the current angular position of the motor, a 5V 250Ω potentiometer is connected to the end of the motor shaft. This potentiometer has a working area of 270° and will break if rotated further than it allows. The potentiometer acts as a variable voltage divider, where the output voltage will be converted to an angular position dependant on the equivalent radian value of the output voltage.

$$4.9907 - 0.007 = 4.9837 \tag{3.1}$$

In order to convert the voltage values of the potentiometer to radians, it is necessary to perform a small set of calculations. The maximum and the minimum voltage values measured from the potentiometer can be seen in equation 3.1. These values will serve as the voltage range for the angular positions.

$$270° = 4.712rad \tag{3.2}$$

In order for the system to function, the degree range of the potentiometer has to be converted to radians, since the mathematical modelling will be done using radians instead of degrees.

$$\frac{4.7124}{4.9837} = 0.9456 \tag{3.3}$$

Equation 3.3 shows the radians per volt and will be used to calculate the current motor position. When the beam is at a level position, the measured voltage from the potentiometer will indicate 0 radians position. This means that the measured

voltages in the positive and negative directions around this point need to indicate positive and negative radian values.

$$potVoltToRad(u) = (cal - u) \cdot 0.9456 \tag{3.4}$$

The value *cal* in equation 3.4 is the voltage measured from the potentiometer when the beam is at a level position. The value u is the current output voltage from the potentiometer. This results in negative angles when the beam moves downwards past the levelled position of the beam and positive angles when moving upwards.

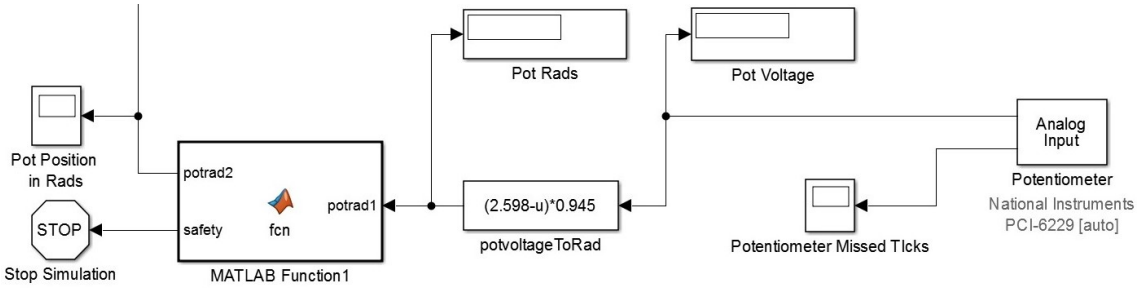


Figure 3.1: Software implementation of safety limits for the potentiometer

Once the beam is at a levelled position, it has been decided that the motor is allowed to move ±1.5 radians. In order to ensure that the system does not damage the potentiometer, a safety system is implemented using software. As shown in figure 3.1, after the potentiometer voltage is calculated to a radians value, using equation 3.4, it is fed into a Matlab Function block, the code contained within the block can be seen in the appendix 11.2. This block checks if the converted voltage exceeds the safety limits of ±1.5 radians and if the threshold is exceeded, the system is shut off in order to prevent any possible damage to the potentiometer.

3.1 Motor controller

In order to drive the motor which controls the angle of the beam, it has been decided that we will build a motor controller. To fulfil the requirements, this motor controller needs to be able to adjust the speed and the direction of the motor with ease. A typical H-bridge is capable of achieving both of the requirements. See figure 3.2 for an H-bridge design.

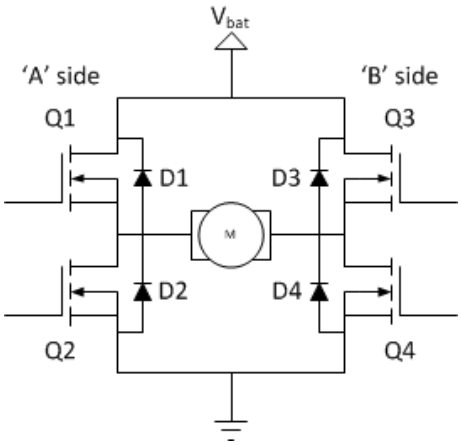


Figure 3.2: H-bridge [4]

An H-bridge uses four transistors as gates and two input signals to turn on each pair of the gates. In each of the transistor pairs, one of them allows the current from the supply to flow while the second one connects it to the ground. An example of how the current flows when the gates are opened can be seen in figure 3.3.

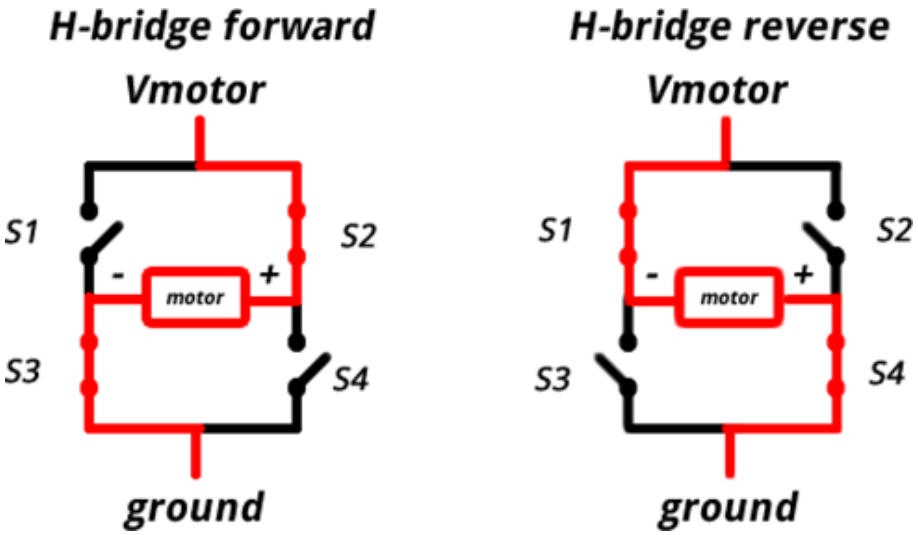


Figure 3.3: H-bridge with voltage flowing through [5]

For this motor controller design two P-channel and two N-channel MOSFETs (metal–oxide–semiconductor field-effect transistors) are used to act as the gates of the H-bridge. By turning on one pair of the gates, it will allow the motor to run in one direction. If the second pair of the gates were powered on, the motor would run in the opposite direction due to opposite flow of the current. See table 3.1 for logic levels of the h-bridge.

Input 1	Input 2	Result
0	0	Slowing
1	0	Forward
0	1	Reverse
1	1	Slowing

Table 3.1: H-bridge logic levels

This way, with an H-bridge it is possible to control the direction of the motor. Typically the speed adjustment is achieved by applying PWM to the input signals. PWM changes the width of the signal sent to the to gate and decides for how long it is open at a time, thus changing how long it is being sent over a certain period of time. For example, if the duty cycle is set to 80% on one of the input signals, the motor would run at only 80% of efficiency. Therefore, by applying PWM on the input signals, it is possible to adjust the speed of the motor.

3.1.1 Choosing components

The choice of components for the use in the motor controller began with selecting the right MOSFETs. Tests have revealed that the motor runs at around 3 amps, therefore, the MOSFETs need to be able to withstand that amount of current, or else they will heat up and break. The upper pair of the MOSFETs were chosen to be IRF9530N P-channel MOSFETs [6]. They can handle a continuous drain current of about -14 amps. The lower ones, IRFZ44E N-channel [7], are also sufficient enough, with ability to handle a continuous drain current of 48 amps when powered on. In the end, it is safe to say that both types of MOSFETs are more than capable of withstanding the current range the motor works within. Additionally, both types of MOSFETs come with diodes built in for circuit protection. Due to parasitic capacitance between gate-drain and gate-source in the MOSFETs, it was necessary to include some resistors. The choices were 100 ohm resistors in series with the N-channel gates and 10k ohm resistors in series with the P-channel gates.

In order to simplify the control logic of the H-bridge, it was decided that the design had to be changed to use only two input signals, one for each side, instead of using four signals for each gate. This was accomplished by adding transistors to control how the gate voltage flows. The BC547B [8] NPN transistors were chosen due to their capabilities of interaction with the data acquisition card signals. In order to protect the I/O two non-inverting op-amps were added to the PCI6229 data acquisition card to be used as a buffer for protection.

The final notable component was the passive heat sink on which the MOSFETs were mounted. An aluminium-finned heatsink can dissipate the heat through convection. Since the MOSFETs tend to heat up after prolonged use, it is crucial to have some way to dissipate the heat in order to protect the health of the MOSFETs and the entire motor controller.

The finalized list of materials used for the motor controller:

- 2 x IRFZ44E MOSFET
- 2 x IRF9530N MOSFET
- 2 x 100 ohm resistors
- 2 x 10k ohm resistors
- 2 x BC547B NPN transistors
- 2 x Op-Amps
- Heat sink

3.1.2 Simulation

Before building the motor controller, it is important to make sure that the theoretical solution is in working order. Therefore, circuit simulation software was used, in this case LTSpice, to simulate the H-bridge logic with the added simplifications. While using LTSpice, some issues were encountered, it is worth mentioning that there was no built-in motor model available to use in the simulation. Instead, it was decided to use a simple resistive load of 1k ohm, to simulate a load for the H-bridge. In the end, the simulation goal was to prove that the behaviour was consistent with the expectations. The design which was simulated can be seen in figure 3.4.

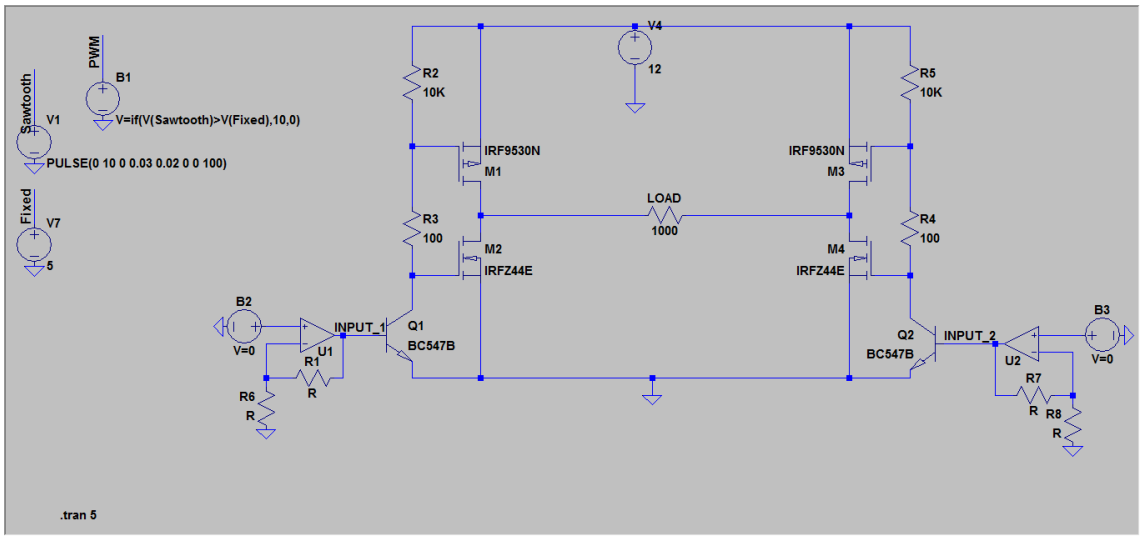


Figure 3.4: H-bridge simulation design

The simulation was run with both inputs set to 0V and whilst measuring the current across the load. The measured signals can be seen in figure 3.5.

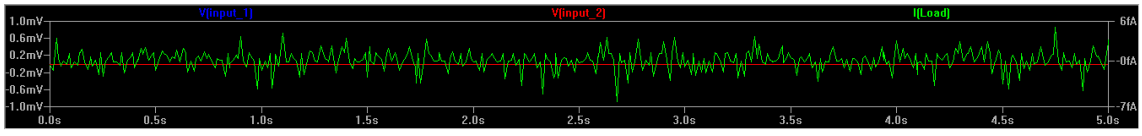


Figure 3.5: H-bridge measured signals

A very small current of ±6fA can be seen mostly due to the design of the H-bridge and the conditions of simulation. Even if such current can be seen in a physical model, it is simply too small to have any noticeable effect on the attached load.
There was then applied a 10V signal to one side of the H-bridge to produce a current across the resistor. The results can be seen in figure 3.6.

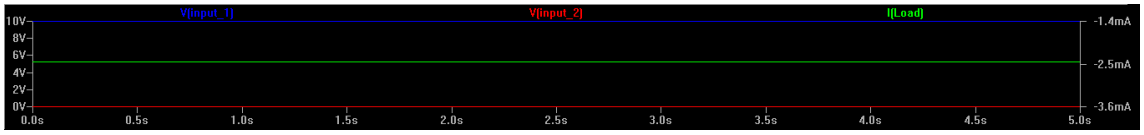


Figure 3.6: H-bridge running on a 10V fixed signal in one direction

The first input was then set to 0V and the other input to 10V to change the direction of the current. See figure 3.7 for the measured signals.

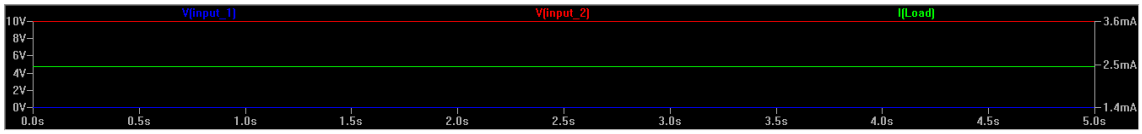


Figure 3.7: H-bridge running on a 10V fixed signal in other direction

Finally, a PWM signal with a duty cycle of 50% was created and applied to one side of the controller in order to simulate if the current logic can handle a change of speed. The results we got can be seen in figure 3.8.

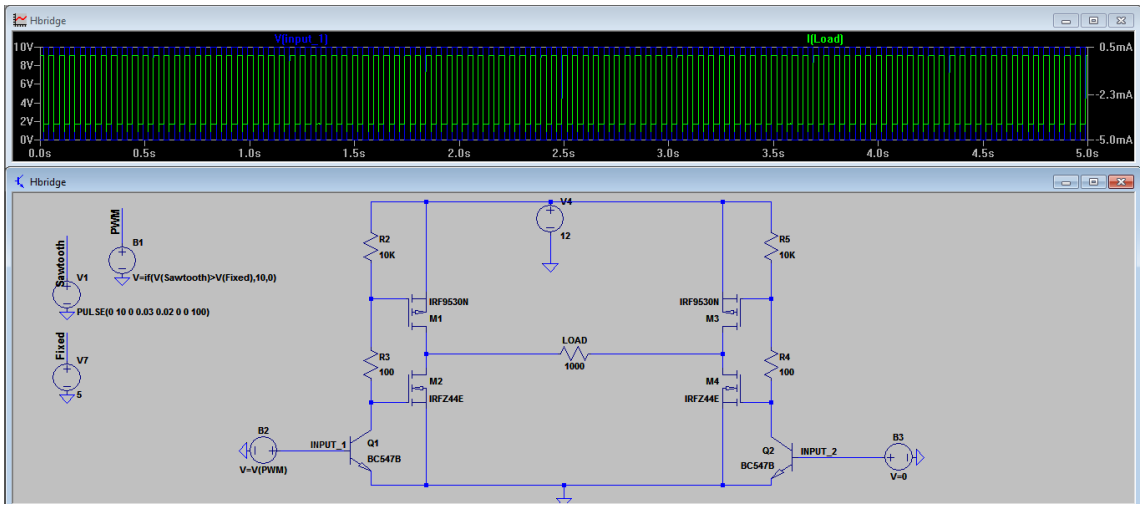


Figure 3.8: H-bridge running on a PWM signal

A close up of the measured signals can be seen in figure 3.9.

Figure 3.9: Closeup of the PWM-run motor controller signals

3.1.3 Motorcontroller testing

Once the simulations proved to be successful, the components were placed on a breadboard in order to perform some testing. This can be seen in figure 3.10.

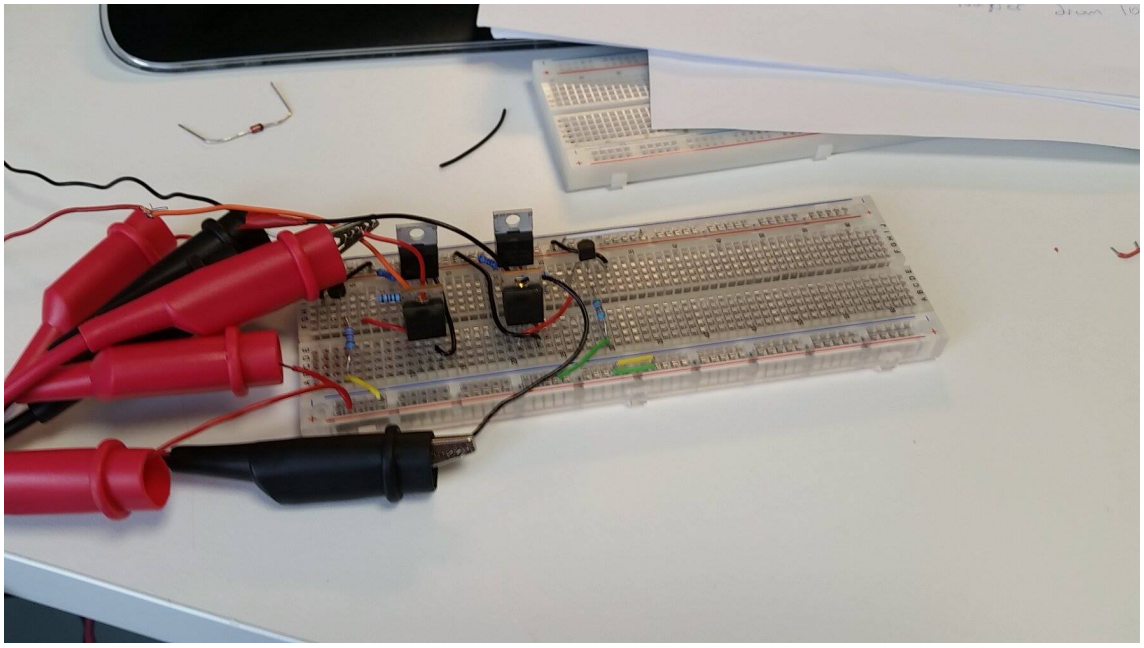


Figure 3.10: Physical motor controller

To test it, a 12V power supply was attached to either produce 10 or 0 volts applied to the NPN transistors. By swapping the voltages between the two inputs, the motor reacted by turning in different directions depending on the state of the pins, proving that the design was successful.

Chapter 4

Mathematical Modelling

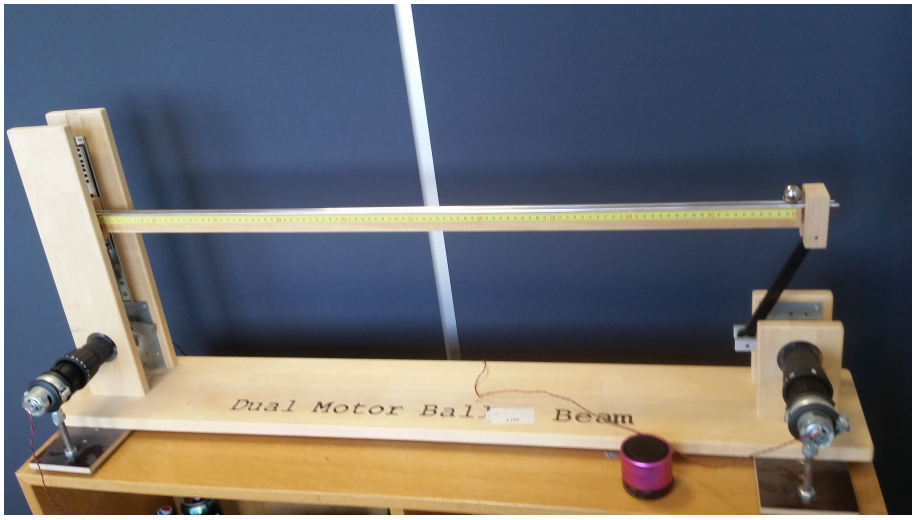


Figure 4.1: The setup of the rig

In order to create a set of mathematical models for the ball and beam system, it is necessary to investigate each part of the system individually. Figure 4.1 displays the platform where the control system will be implemented. The actuator for the system is the motor, where a certain voltage results in angular movement and position. The angular position of the motor is then translated into the equivalent beam angle. Based on the beam angle the ball placed on the beam will be displaced.

The relationship between the input voltage and angular position of the motor will be modelled using the electrical and mechanical representations of the motor circuit and mechanical rotor. A transfer function will then be derived explaining the input and output relationship in terms of the voltage and angle.
In order to translate the motor angle into the beam angle, the gearing ratio between the two will be calculated and used for the equations related to the ball movement. The displacement of the ball will be modelled using Newtonian physics, where a transfer function also will be retrieved representing the input and output relationship of the beam angle and ball displacement.

4.1 Modelling the DC motor

We start the mathematical modelling with the DC motor. The model consists of an electrical part and a mechanical part. In order to determine the transfer function describing the motor, it is necessary to utilize the electrical circuit diagram of the motor as well as the free-body diagram of the mechanical rotor. The diagram of the motor is shown in figure 4.2 [9].

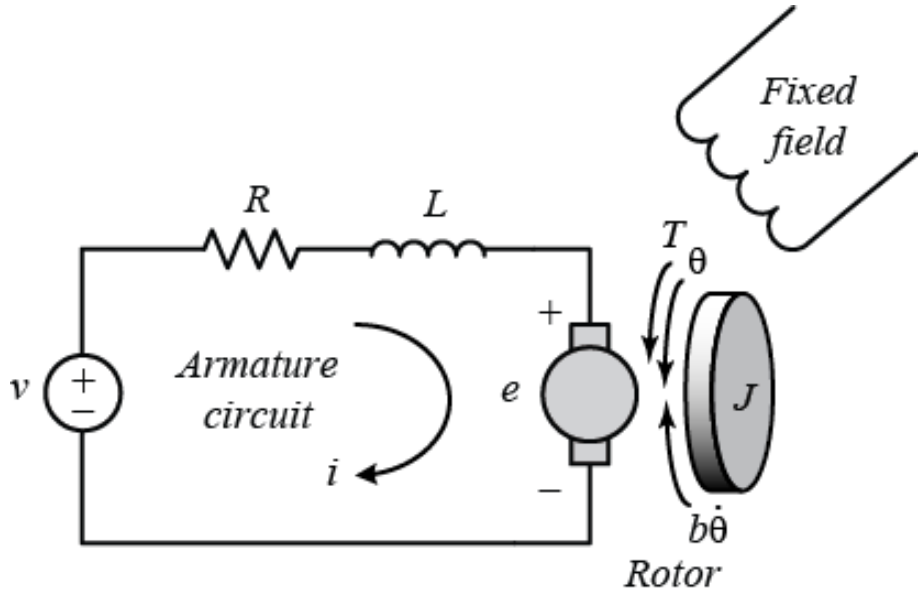


Figure 4.2: Diagram of the electrical and mechanical part of the DC motor [10]

Name	Description	Unit
T_m	Motor torque	Nm
T_L	Load torque	Nm
K_t	Motor constant	
K_e	Electromotive force constant	
$i_a(t)$	Armature current	A
R_a	Armature resistance	Ω
L_a	Armature inductance	H
J	Moment of inertia of the rotor	$kg \cdot m^2$
B	Viscous friction	Nms
$\theta(t)$	Motor angle	rad
$V(t)$	Input voltage	V

Table 4.1: Table of nomenclature for the DC motor

4.1.1 Mechanical Part

Figure 4.3: Diagram of the mechanical part of the DC motor [10]

The equations describing the torque of the DC motor are the following as seen in equations 4.1a and 4.1b.

$$T_m = k_t i_a(t) \tag{4.1a}$$

$$T_m = J\ddot{\theta}(t) + B\dot{\theta}(t) \tag{4.1b}$$

Equation 4.1a shows the relationship between the torque produced from an induced armature current, whereas equation 4.1b shows the torque expressed in terms of Newton's second law for rotational movement. [11]

$$J\ddot{\theta}(t) + B\dot{\theta}(t) + T_L(t) = k_t i_a(t) \tag{4.2}$$

By using the free-body diagram for the rotating armature seen in figure 4.3, equation 4.2 is formed by applying Newton's law for rotational motion, where T_L is the load torque. $B\dot{\theta}(t)$ is an expression for the viscous friction.

4.1.2 Determining the Load Torque

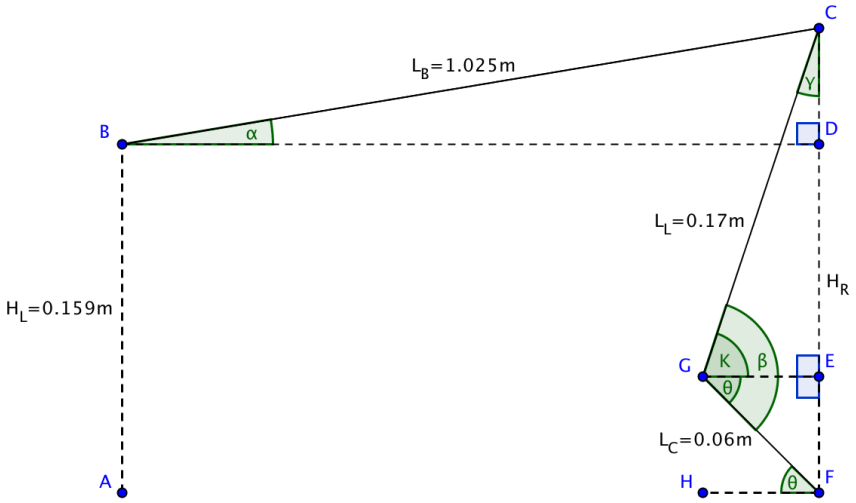


Figure 4.4: Model illustration

The load torque(T_L) is the beam's effect on the rotation of the motor shaft (θ). It can be calculated using the vector cross product between the length of the crank arm and the applied force, as seen in equation 4.3a.

$$T_L = F \times L_C = F \cdot L_C \cdot \sin(\beta) \tag{4.3a}$$

L_C is an already known constant, which will be inserted into the following equations. Moving on, in order to find an expression for the angle β, it is necessary to calculate the length of either one of the segments GE or HF. Since they are equal, it is then possible to determine γ and κ as shown in equations 4.4.

$$GE = L_C \cdot cos(\theta) = 0.06 \cdot cos(\theta) \tag{4.4a}$$

$$\gamma = \sin^{-1}\left(\frac{GE}{L_L}\right) = \sin^{-1}\left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right) \tag{4.4b}$$

$$\kappa = \frac{\pi}{2} - \gamma \ = \frac{\pi}{2} - \sin^{-1}\left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right) \tag{4.4c}$$

$$\beta = \theta + \kappa = \theta + \frac{\pi}{2} - \sin^{-1}\left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right) \tag{4.4d}$$

The force from the beam pressing onto the lever arm can then be achieved as seen in equations 4.5, where equation 4.6 is the final expression for the torque applied to

the motor shaft.

$$F_1 = m \cdot a \cdot cos(\gamma) = 0.4 \cdot 9.82 \cdot \cos\left(\sin^{-1}\left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right)\right) \tag{4.5a}$$

$$= 3.928 \cdot \cos\left(\sin^{-1}\left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right)\right) \tag{4.5b}$$

$$= 3.928 \cdot \sqrt{1 - \left(\frac{0.06 \cdot \cos(\theta)}{0.17}\right)^2} \tag{4.5c}$$

$$= 3.928 \cdot \sqrt{1 - 0.125 \cdot \cos^2(\theta)} \tag{4.5d}$$

The final load torque, T_L, can then be calculated using equations in 4.6.

$$T_L = 3.928 \cdot \sqrt{1 - 0.125 \cdot \cos^2(\theta)} \cdot 0.06 \cdot \sin(\beta) \tag{4.6a}$$

$$= 0.23568 \cdot \sqrt{1 - 0.125 \cdot \cos^2(\theta)} \cdot \sin\left(\theta + \frac{\pi}{2} - \sin^{-1}\left(\frac{0.06 \cdot cos(\theta)}{0.17}\right)\right) \tag{4.6b}$$

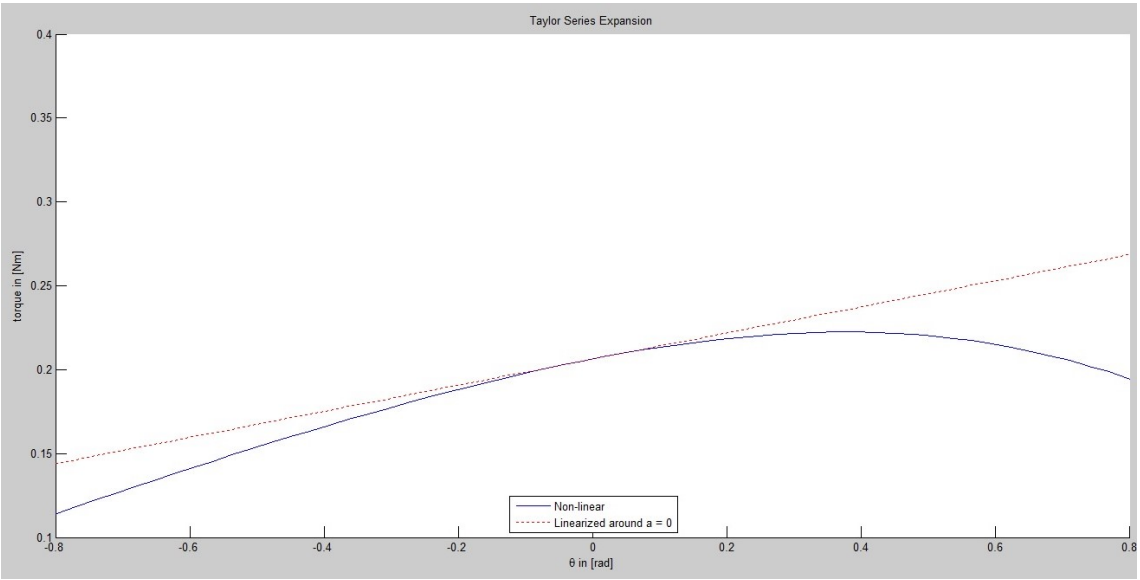


Figure 4.5: Taylor expansion in θ=0 rad

Since the relationship between θ and T_L is non-linear the equation is then linearised using Taylor Expansion. Figure 4.5 indicates that the linearised model becomes less accurate as the motor angle moves away from zero.

The linearised result from equation 4.6a is given in equation 4.7. The linearisation was done around an expansion point of 0 radians.

$$T_L = 0.0778\theta + 0.206 \tag{4.7}$$

4.1.3 Electrical Part

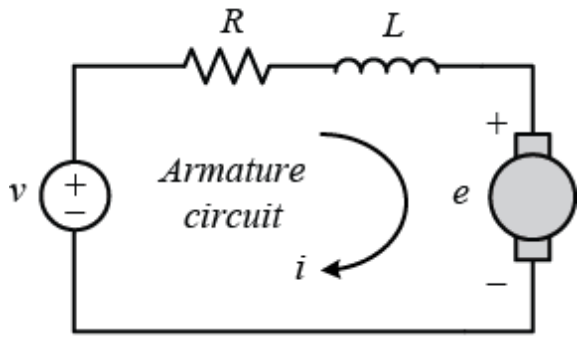


Figure 4.6: Diagram of the electrical part of the DC motor [10]

Analysing the circuit seen in figure 4.6 and using Kirchhoff's voltage law (KVL), the expression found at equation 4.8 is formed.

$$L_a\dot{i}(t) + R_a i(t) = V(t) - k_e\dot{\theta}(t) \tag{4.8}$$

$k_e\dot{\theta}$ describes the voltage that appears as a result of the motor's coils moving relative to a magnetic field. This voltage is often referred to as back electromotive force, or back EMF for short. It opposes the supply voltage and therefore reduces the current while the motor is let to run freely.

To manipulate equation 4.8, the Laplace transformation is performed. By isolating in terms of the current, which can be seen in equation 4.9, it further simplifies the differential equation related to the electrical part of the DC motor. Simplified equation will, later in the chapter, be combined with the mechanical part of the DC motor.

$$L_a I(s)s + R_a I(s) = V(s) - k_e\Theta(s)s \tag{4.9a}$$
$$(L_a s + R_a)I(s) = V(s) - k_e\Theta(s)s \tag{4.9b}$$
$$I(s) = \frac{V(s) - k_e\Theta(s)s}{L_a s + R_a} \tag{4.9c}$$

4.1.4 Combination

Earlier in the report equation 4.2 was found, this equation describes the mechanical part of the system which was derived from the free-body diagram. By inserting the linearised T_L (found in equation 4.7) it is possible to form the entire expression for the mechanical part of the DC motor (seen in equation 4.10).

$$J\ddot{\theta}(t) + B\dot{\theta}(t) + 0.0778\theta(t) + 0.206 = k_t i_a(t) \tag{4.10}$$

In order to take care of the constant part of equation 4.10, it is necessary to calculate the shift in coordinates. This is done around the equilibrium point of the system where all dynamics are equal to zero. This will be revisited later in the report, once the motor coefficients have been determined, but at this current stage the constant is disregarded.

$$J\ddot{\theta}(t) + B\dot{\theta}(t) + 0.0778\theta(t) = k_t i_a(t) \tag{4.11}$$

Equation 4.11 denotes the mechanical part of the DC motor, where the constant from the linearisation has been removed. By taking the Laplace transformation of the equation it is then possible to obtain the equations shown in 4.12, which allow easier combination with the electrical part later on.

$$J\Theta(s)s^2 + B\Theta(s)s + 0.0778\Theta(s) = k_t I(s) \tag{4.12a}$$
$$(Js^2 + Bs + 0.0778)\Theta(s) = k_t I(s) \tag{4.12b}$$

By inserting equation 4.9c into equation 4.12b, it is possible to cancel out the terms involving the current. This then results in equation 4.13.

$$(Js^2 + Bs + 0.0778)\Theta(s) = k_t\left(\frac{V(s) - k_e\Theta(s)s}{L_a s + R_a}\right) \tag{4.13}$$

Through simplification and by gathering like terms, the equation for the DC motor will become as shown in equation 4.14b

$$\frac{k_t k_e \Theta(s)s}{L_a s + R_a} + (Js^2 + Bs + 0.0778)\Theta(s) = \frac{k_t V(s)}{L_a s + R_a} \tag{4.14a}$$
$$\left(\frac{k_t k_e s}{L_a s + R_a} + Js^2 + Bs + 0.0778\right)\Theta(s) = \frac{k_t}{L_a s + R_a}V(s) \tag{4.14b}$$

To properly express the input-output relationship of the DC motor, it is necessary to simplify and factor equation 4.14b, which is the expression related to the output angle depending on an input voltage. The transfer function for the DC motor is then as shown in equation 4.15.

$$\frac{\Theta(s)}{V(s)} = \frac{\frac{k_t}{L_a s + R_a}}{\frac{k_t k_e s}{L_a s + R_a} + Js^2 + Bs + 0.0778} \tag{4.15}$$

By further simplifying equation 4.15 and by ignoring the terms related to inductance, the final transfer function for the the input voltage and the output angle is shown in equation 4.16b.

$$\frac{\Theta(s)}{V(s)} = \frac{k_t}{JL_a s^3 + JR_a s^2 + L_a B s^2 + R_a B s + k_t k_e s + 0.0778 R_a + 0.778 L_a s} \tag{4.16a}$$

$$\frac{\Theta(s)}{V(s)} = \frac{k_t}{JR_a s^2 + (R_a B + k_t k_e) s + 0.0778 R_a} \tag{4.16b}$$

4.2 Modelling the Beam Angle

The next step of the mathematical modelling is related to the beam angle. In order to control the beam's position, it is necessary to find a relationship between the angle of the beam α and the angle of the motor θ, which can be seen in figure 4.7.

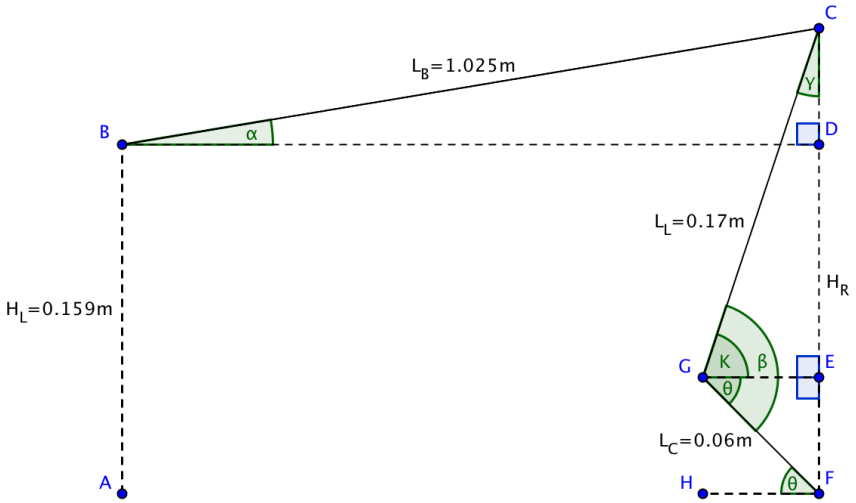


Figure 4.7: Model illustration

By applying the sine definition to the BCD triangle, the following equation is obtained.

$$\alpha = \sin^{-1}\left(\frac{CD}{L_B}\right) = \sin^{-1}\left(\frac{H_R - H_L}{L_B}\right) \tag{4.17}$$

Since two terms of the the equation above have fixed values, H_L and L_B, its only needed to find a way to express H_R in terms of θ. As seen in equation 4.18, H_R can be described as the sum of CE and EF. An expression for CE can be found by applying the cosine definition in the GEF triangle and Pythagoras' formula in the CGE triangle:

$$CE = \sqrt{L_L^2 - (L_C cos(\theta) - L_B(1 - cos(\alpha)))^2} \tag{4.18}$$

By applying the sine definition in the GEF triangle, equation 4.19 is obtained.

$$EF = L_C sin(\theta) \tag{4.19}$$

Therefore, the final equation for H_R is equation 4.20.

$$H_R = L_C sin(\theta) + \sqrt{L_L^2 - (L_C cos(\theta) - L_B(1 - cos(\alpha)))^2} \tag{4.20}$$

Inserting equation 4.20 into 4.17 results in what can be found in equation 4.21.

$$\alpha = \sin^{-1}\left(\frac{L_C sin(\theta) + \sqrt{L_L^2 - (L_C cos(\theta) - L_B(1 - cos(\alpha)))^2} - H_L}{L_B}\right) \tag{4.21}$$

The $L_B(1 - cos(\alpha))$ term describes the horizontal measurement change determined by the vertical movement of the beam and it can be omitted from the above equation due to the fact that α is a small angle. Therefore, its assumed that it tends to 0 and computes $\lim_{\alpha \to 0} cos(\alpha) = 1$, which proves that the whole term cancels out.

The final equation to express α is then:

$$\alpha = \sin^{-1}\left(\frac{L_C sin(\theta) + \sqrt{L_L^2 - (L_C cos(\theta))^2} - H_L}{L_B}\right) \tag{4.22}$$

Replacing the terms which are constants measured from the setup - H_L, L_B, L_L, L_C [m] with their actual values (as shown in figure 4.7), results in the following particular equation 4.23.

$$\alpha = \sin^{-1}\left(\frac{0.06 sin(\theta) + \sqrt{0.17^2 - (0.06 cos(\theta))^2} - 0.159}{1.025}\right) \tag{4.23}$$

Since the relationship between α and θ is non-linear, the equation will be linearised using the Taylor expansion. The accuracy of the linearised equation depends on the point where it is linearised. See figure 4.8 and figure 4.9.

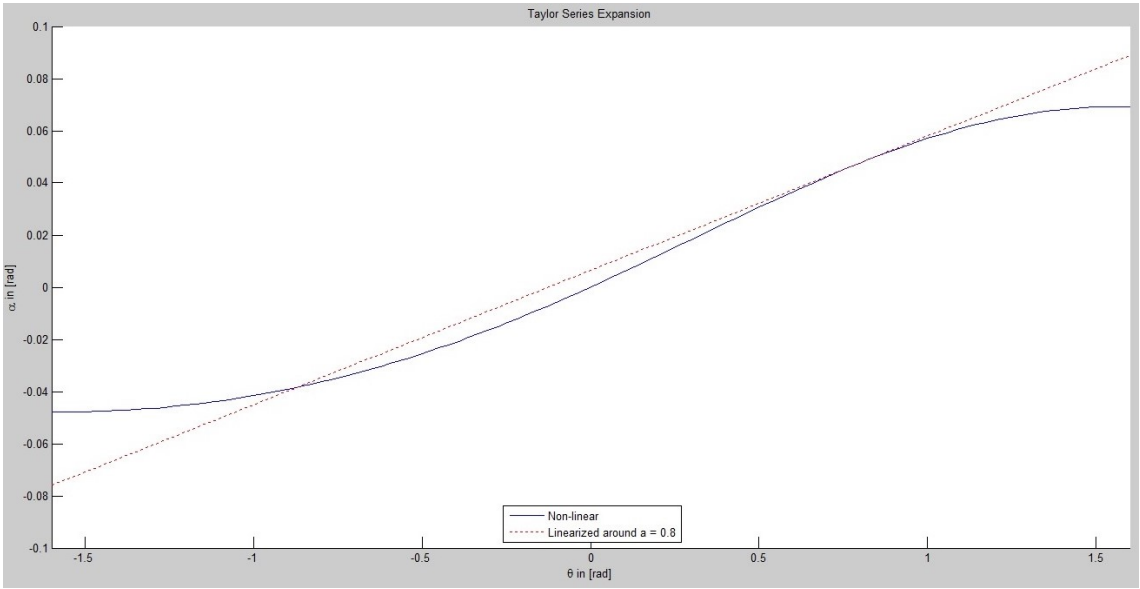


Figure 4.8: Taylor expansion in θ=0.8 rad

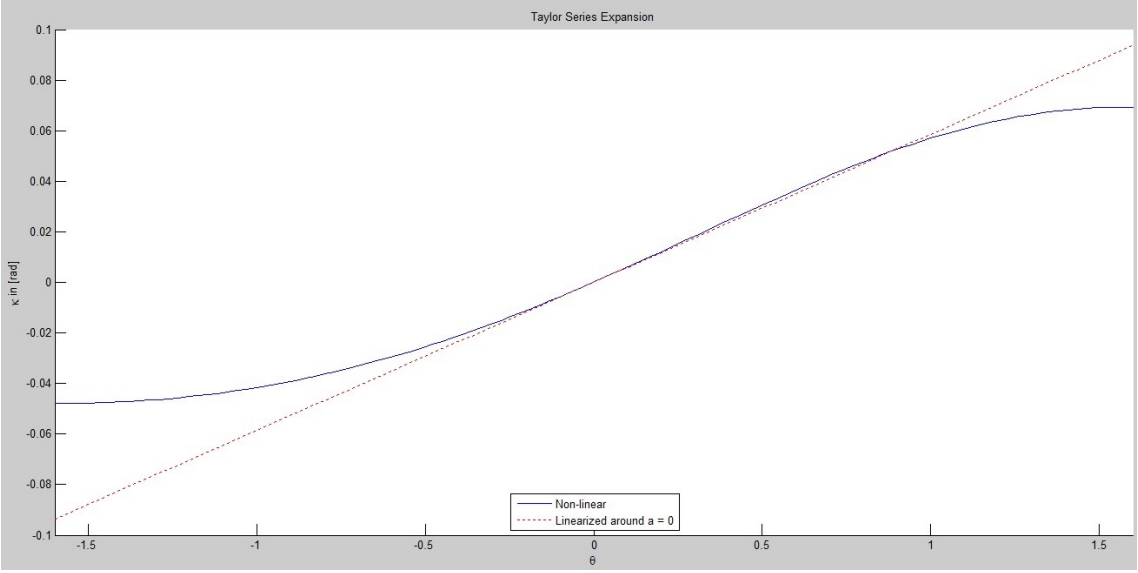


Figure 4.9: Taylor expansion in θ=0 rad

The result by linearising the function in $\theta = 0$ can be seen in equation 4.24a.

$$\alpha = 0.0585\theta - 0.0033 \tag{4.24a}$$

The constant part of equation 4.24a is ignored, since it has almost no influence on the final result. Therefore, the angular relationship between the beam and the motor, also known as the gear ratio, can be found in equation 4.25.

$$\alpha = 0.0585\theta \tag{4.25}$$

4.3 Modelling Ball Movement

The last part of the mathematical modelling is determining the ball movement.

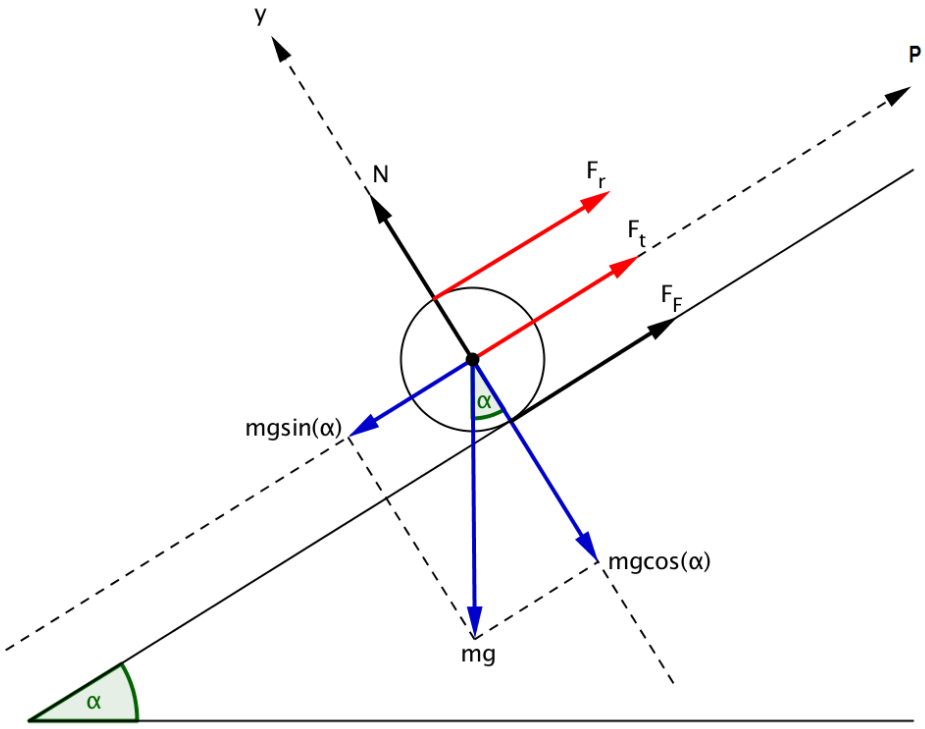


Figure 4.10: Free-body diagram of the ball's movement

Since the gravitational force is neither perpendicular nor parallel to the inclined plane, this being the beam, it needs to be decomposed into an x-component and a y-component. To do so, parallels are drawn to both the x and y axis through the end point of the gravitational force vector. The result will be a rectangle which has the x-component and y-component as its sides. The resulting decomposition can be seen in figure 4.10. For the mathematical modelling of the ball movement the friction force F_F is ignored and the ball movement will be assumed to be friction less.

Name	Description	Unit
T_r	Ball rotational torque	Nm
F_r	Edge force of ball	N
F_t	Force from translational motion	N
R	Ball radius	m
J	Moment of inertia	$kg \cdot m^2$
J_b	Moment of inertia of the ball	$kg \cdot m^2$
p	Ball position	m
m	Ball mass	kg
g	Gravitational force	N
α	Beam angle	rad
ω	Angular velocity of the ball	rad/s
v	Ball speed along p-axis	m/s

Table 4.2: Table of nomenclature for the ball dynamics

The torque, generated from the ball rolling on the beam, can be expressed by the force vector F_r multiplied by the ball's radius, this force can been seen on the free-body diagram in figure 4.10. Through Newton's second law for rotational movement, it is possible to express the following relationship seen in equation 4.26.

$$T_r = F_r R = J\dot{\omega} = J\left(\frac{\dot{v}}{R}\right) = \frac{J}{R}\ddot{p} \tag{4.26}$$

Through simplification it is possible to create an expression in terms of the vector F_r, as seen in equations 4.27a and 4.27b.

$$F_r R = \frac{J_b}{R}\ddot{p} \tag{4.27a}$$

$$F_r = \frac{J_b}{R^2}\ddot{p} \tag{4.27b}$$

The moment of inertia for a solid ball is defined as $J_b = \frac{2}{5}mR^2$ and is inserted into equation 4.27b, resulting in equation 4.28.

$$F_r = \frac{2}{5}m\ddot{p} \tag{4.28}$$

Besides the rotational force, there is also a force generated through the translational motion, and is described as found in equation 4.29. This force can also be found on the ball movements free-body diagram in figure 4.10.

$$F_t = m\ddot{p} \tag{4.29}$$

By summing the influential forces from the free-body diagram in figure 4.10, equation 4.30 shows the force relationships.

$$F_r + F_t = mg\sin(\alpha) \tag{4.30}$$

Equations 4.28 and 4.29 are then inserted into equation 4.30 to allow simplification, which can been seen in equations 4.31. These simplifications allow the mathematical model to become independent from the ball's mass.

$$\frac{2}{5}m\ddot{p} + m\ddot{p} = mg\sin(\alpha) \tag{4.31a}$$

$$m\left(\frac{2}{5}\ddot{p} + \ddot{p}\right) = mg\sin(\alpha) \tag{4.31b}$$

$$\frac{2}{5}\ddot{p} + \frac{5}{5}\ddot{p} = g\sin(\alpha) \tag{4.31c}$$

$$\frac{7}{5}\ddot{p} = g\sin(\alpha) \tag{4.31d}$$

The ball's acceleration is then determined by equation 4.32 this describes a non-linear relationship between the beam angle and the ball acceleration.

$$\ddot{p} = \frac{5}{7}g\sin(\alpha) \tag{4.32}$$

To linearise equation 4.32 we can assume, that for angles less than 0.1 radians, $\sin(\alpha) = \alpha$, since $\sin(0.1) = 0.0998rad$. For smaller angles, the difference will be even smaller. In terms of angular movement of the motor, it has been decided that the motor angle will be able to move ± 1.5 radians. Previously calculated in the report, the gear ratio between the beam angle and motor angle was found in equation 4.25, this can then be utilized to determine the maximum beam angle can be found in equation 4.33

$$\alpha = 1.5rad \cdot 0.0585 \tag{4.33a}$$

$$\alpha = 0.0878rad \tag{4.33b}$$

$$\sin(\alpha) = \alpha \tag{4.33c}$$

$$-0.1rad < \alpha < 0.1rad \tag{4.33d}$$

Since the maximum beam angle is less than 0.1 radians, the linearised expression will be equation 4.34c after the gear ratio containing θ from equation 4.25 is added to replace the term including α.

$$\ddot{p} = 0.0585 \cdot \frac{5}{7} \cdot g \cdot \theta \tag{4.34a}$$

$$\ddot{p} = 0.0585 \cdot \frac{5}{7} \cdot 9.82 \cdot \theta \tag{4.34b}$$

$$\ddot{p} = 0.4097\theta \tag{4.34c}$$

By calculating the scalar in front of θ it is possible to retrieve equation 4.34c. Then, by taking the Laplace transformation on each side of the equal sign, the equation becomes the following as shown in 4.35.

$$P(s)s^2 = 0.4097\Theta(s) \tag{4.35}$$

The transfer function for the ball position dependant on the motor angle angle is expressed as the following in equation 4.36.

$$\frac{P(s)}{\Theta(s)} = \frac{0.4097}{s^2} \tag{4.36}$$

Equation 4.37 shows the ball transfer function with the gear ratio separated, since this will be used in some visual representations of system later in the report.

$$\frac{P(s)}{\Theta(s)} = 0.0585 \cdot \frac{7.01}{s^2} \tag{4.37}$$

Chapter 5

Experiments and Lab Work

5.1 Data Acquisition

The best suited interface for the project was the National Instruments PCI-6229 board. One of the main goals was to use an interface that allowed for real-time interaction with Matlab and Simulink. Initially, a possibility was to use an Arduino since the group is comfortable with our knowledge surrounding the platform and also because Arduino has a great amount of support available. If the Arduino was the chosen platform, it would have used Matlab's hardware support package created for it. After doing research and some initial tests, it was concluded that another DAQ (data acquisition) method was needed because of the slow communication between the Arduino and the computer for data acquisition, and the low refresh rate of the system. The university uses the National Instruments PCI-6229 boards, as seen in figure 5.1, for students' control system projects and for control related lab exercises. Since using Matlab and Simulink is part of the curriculum this semester, working with the PCI-6229 would provide the best learning experience in terms of gaining Simulink and Matlab knowledge.

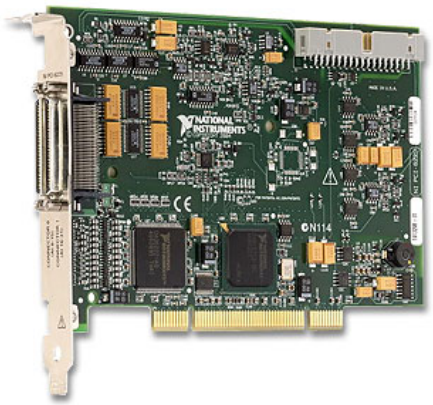


Figure 5.1: The National Instruments PCI-6229 board [12]

The PCI-6229 has 4 x 16bit analog inputs (ADCs) and 48 digital I/O's. Further

specifications can be seen in figure 5.2. It is possible to configure the board to have the following input ranges: $\pm 0.2V$,$\pm 1V$, $\pm 5V$ and $\pm 10V$. Each input and output channels can be individually configured for different ranges [13].

Analog Input

Number of channels	16 differential or 32 single ended
ADC resolution	16 bits
DNL	No missing codes guaranteed
INL	Refer to the *AI Absolute Accuracy* section
Sample rate	
Single channel maximum	250 kS/s
Multichannel maximum (aggregate)	250 kS/s
Minimum	No minimum
Timing accuracy	50 ppm of sample rate
Timing resolution	50 ns
Input coupling	DC
Input range	±0.2 V, ±1 V, ±5 V, ±10 V
Maximum working voltage for analog inputs (signal + common mode)	±11 V of AI GND

(a) Analog inputs

Analog Output

Number of channels	4
DAC resolution	16 bits
DNL	±1 LSB
Monotonicity	16 bit guaranteed
Maximum update rate	
1 channel	833 kS/s
2 channels	740 kS/s per channel
3 channels	666 kS/s per channel
4 channels	625 kS/s per channel
Timing accuracy	50 ppm of sample rate
Timing resolution	50 ns
Output range	±10 V
Output coupling	DC

(b) Analog outputs

Figure 5.2: Specification of the PCI-6229 [13]

Initializing the board with Matlab can be done either directly in the Matlab environment or using Simulink, which is a simulation platform inside Matlab. For our initial experiments, we decided to communicate with the board using only Matlab. To create a session in Matlab you need to give the command seen in code listing 5.1. The remaining code can be seen in the appendix 11.1.

```
1 % First needed is to create a matlab session with the board
  s = daq.createSession('ni');
```

Code Listing 5.1: Initializing the board

To be able to physically connect to the PCI-6229 board we use the CB-68LP board, which contains screw terminals to allow connection to different channels. The CB-68LP can be seen in figure 5.3 and the pinout for it in figure 5.4.

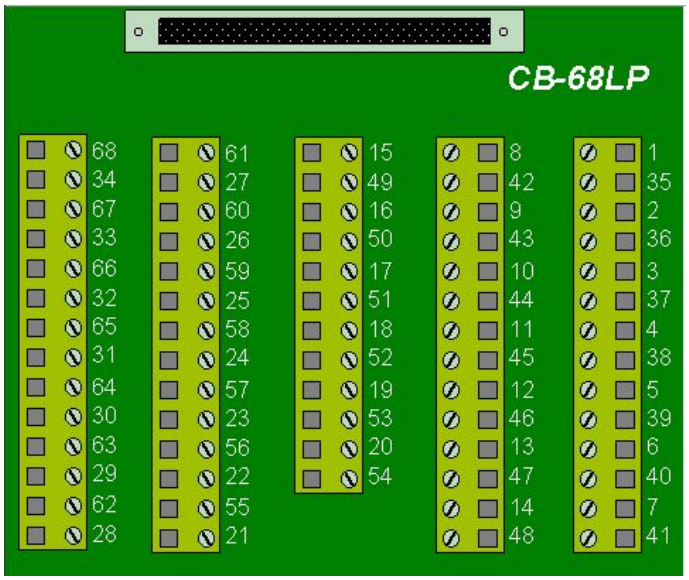


Figure 5.3: The National Instruments CB-68LP screw terminal board [14]

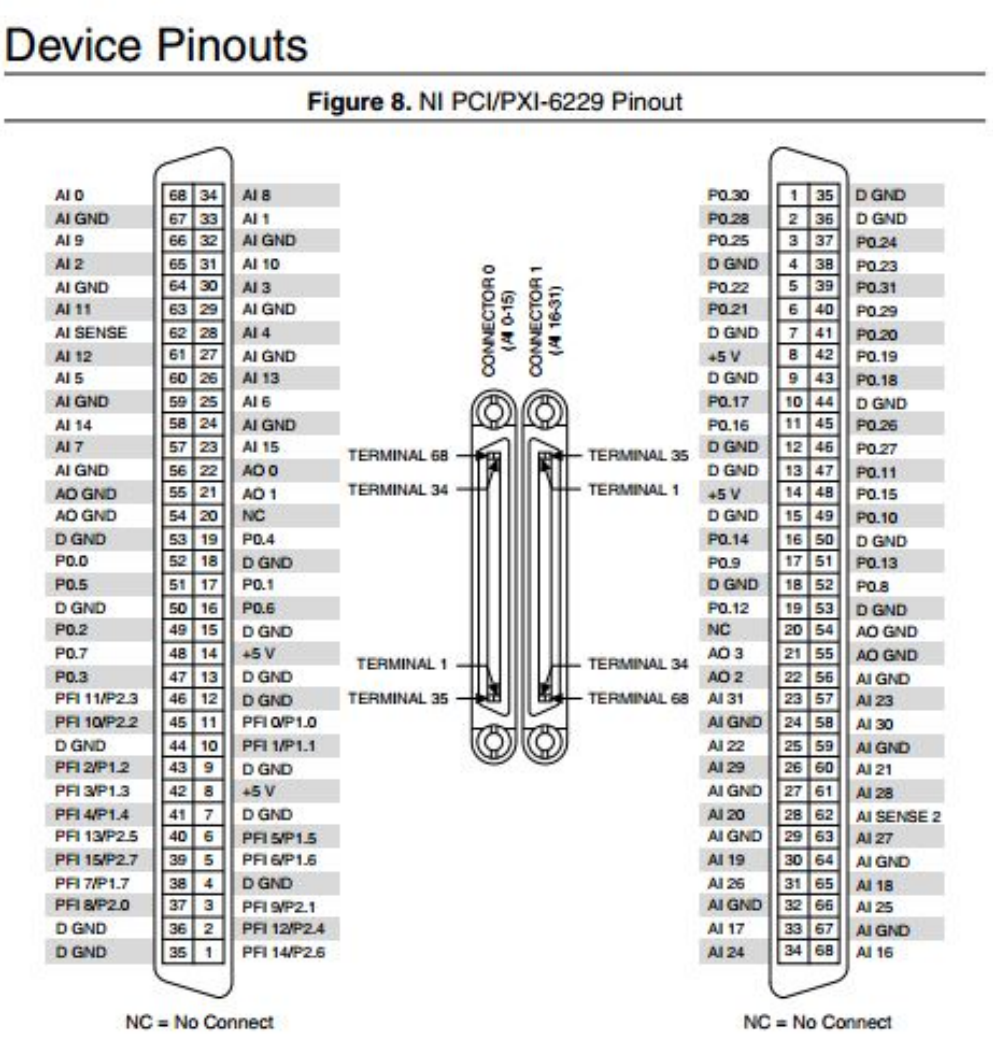


Figure 5.4: The National Instruments PCI-6229 pinout [13]

5.2 Determining the Ball Placement

To determine the position of the ball, we are using two metal rails and a metal ball. One rail is supplied with 2 amps and the voltages at rail 1 and 2 are then measured using an analog input on the PCI-6229 board. The two rails and the ball then acts as a variable resistor that affects the voltage reading on rail 2. Previous projects that used a similar system amplified the rail signals using a set of operational amplifiers with gains at around 20.

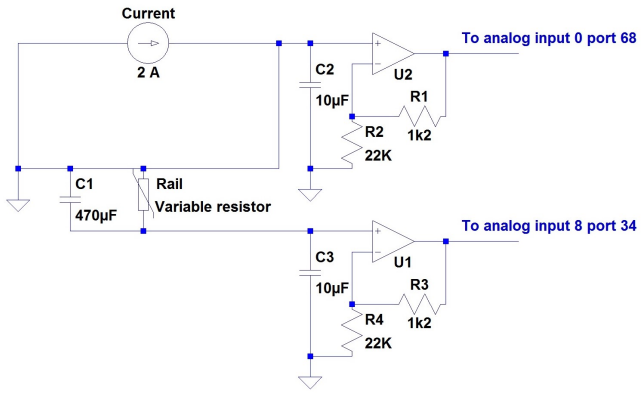


Figure 5.5: Rail setup with opamps

The reasoning behind the use of the OpAmps is that the measured voltage is really low, see figure 5.5. Using an OpAmps setup to amplify the rail outputs created too much noise in our case and caused instability between the measured values. The solution was then to remove the OpAmps and use the setup as shown in figure 5.6.

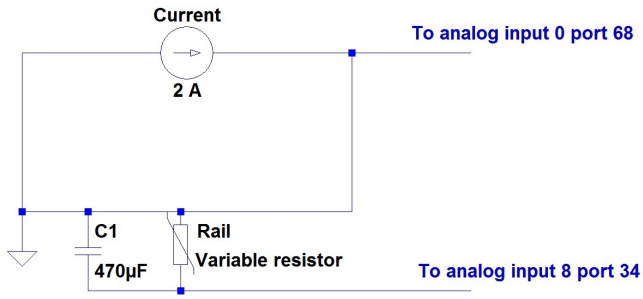


Figure 5.6: Rail setup without opamps

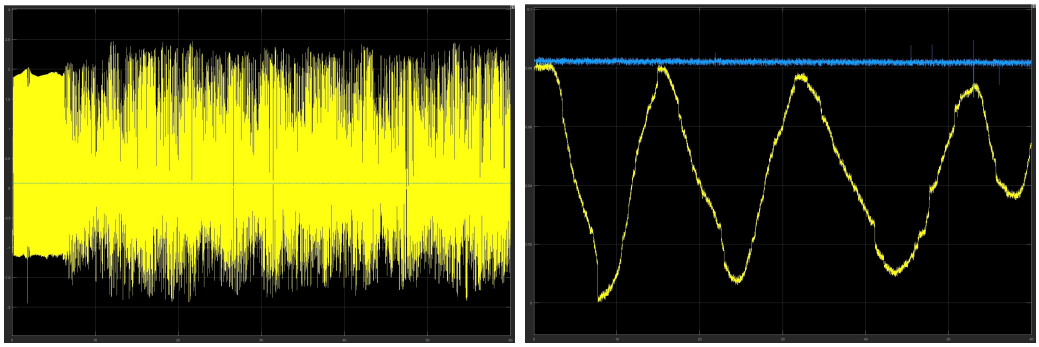
When connecting the outputs of the rails directly to the board it is necessary to be aware of the 10V limit for the PCI-6229. Before connecting the board to the output rails, the voltage output was measured with a multimeter. The multimeter measured a voltage at around 1V, which is within the operating voltages for the board. By default the PCI-6229 is configured to be in differential mode, meaning that it subtracts two inputs from each another to produce a single output, see code

listing 5.2. That configuration suits the criteria needed to experiment with the setup. Rail 1 was connected to port 68 (A0) and rail 2 to port 34 (A8).

```
% We want to have it set to reference single ended
% To change that you write
ch.TerminalConfig = 'SingleEnded';
```

Code Listing 5.2: Singel ended input

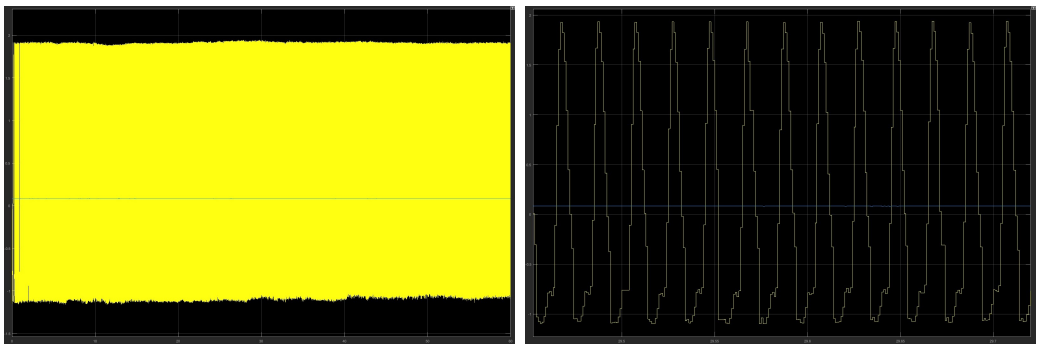
The physical setup came with a pre-mounted $470\mu F$ capacitor between the two rails. Experiments were performed with capacitor removed and letting the ball move back and forth for 60 seconds, this can be seen in figure 5.7a. Experiments were also performed with the capacitor on, see figure 5.7b. The noise generated without the capacitor was comparable to a sinus wave and was at a much higher voltage.



(a) Ball movement without a capacitor **(b)** Ball movement with a capacitor

Figure 5.7: Comparison of ball movement with and withour a capacitor

A close up of the noise without the capacitor while the ball was resting at the end of the rail for 60 seconds can be seen in figure 5.8a. The blue line indicates voltage on rail 1 and the yellow is the voltage on rail 2.



(a) Noise at the beginning of the rail **(b)** Zoomed in of the noise

Figure 5.8: Noise generated at the beginning of the rail

Based of the performance with and without the capacitor, it was decided to remount the capacitor and use the setup without op-amps. Using Matlab, measurements of

the ball placement were then acquired for 20 seconds at a sampling frequency of 10000Hz, the plotted data is shown in figure 5.9.

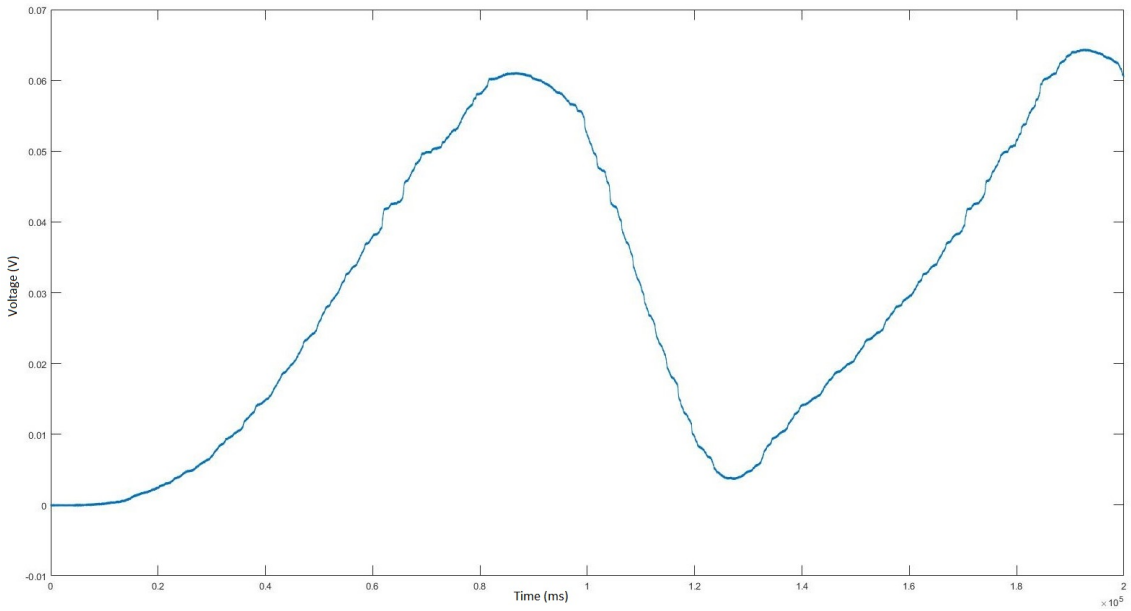


Figure 5.9: Ball moving back and forth for 20 seconds

During the 20 second duration shown in figure 5.9 the ball was moved back and forth to give an overview of the noise and voltage values at different positions. By comparing the ball movement data gathered with and without the OpAmps, it was possible to determine that the data whilst not using the OpAmp setup was far less noisy.

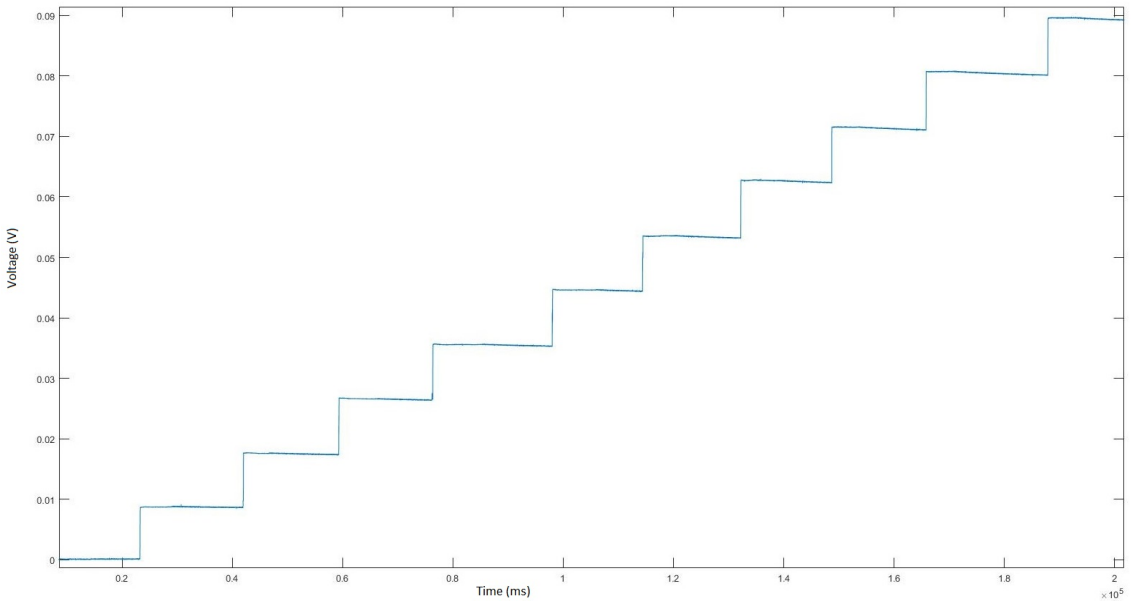


Figure 5.10: Ball removed and put on every 10 cm

The next test consisted of removing the ball from the rail and placing it in different positions every 10 cm, this resulted in 10 measurements since the beam length is 1 meter. Figure 5.10 displays the voltage values over time. The reason why the last positional value remains steady is due to the 470μF capacitor that is situated between the rails. If the ball is removed for 30 seconds and the capacitor is allowed to drain it produces the curve shown in figure 5.11.

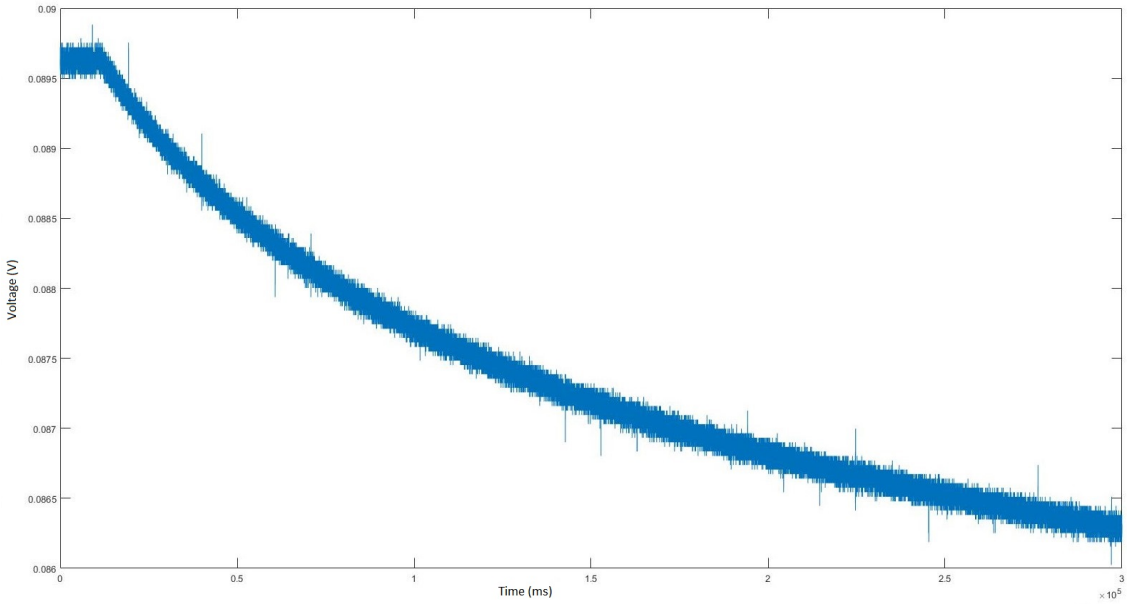


Figure 5.11: Ball taken off and capacitor drained

To determine the relation between the voltage readings from the rail and the actual position of the ball, the ball was placed at both ends for 10 seconds while collecting voltage readings. The mean value of these readings provided a voltage at 0 cm, which was 0.000034028V and at 100 cm the mean voltage was 0.0896V. Equations 5.1b shows how to calculate the voltage value of a single centimetre.

$$V_{difference} = 8.96 \times 10^{-2}V - 3.4 \times 10^{-5}V = 8.9565972 \times 10^{-2}V \tag{5.1a}$$

$$1cm = \frac{8.9565972 \times 10^{-2}V}{100cm} = 8.9579 \times 10^{-4}\frac{V}{cm} \tag{5.1b}$$

Equation 5.1b then provides the information that for every movement of 1 cm the rail voltage increases by approximately 0.00089579V.

In order to calculate the drain rate of the capacitor it is based on the total drain time. The capacitor drained for 29 seconds and went from 0.0899V to 0.0860V. It is then possible to calculate how many centimetres the ball moves during the course of over the 29 seconds see equation 5.2c.

$$V_{difference} = 8.99 \times 10^{-2}V - 8.6 \times 10^{-2}V = 3.9 \times 10^{-3}V \tag{5.2a}$$

$$Drain_{29s} = \frac{3.9 \times 10^{-3}V}{8.9579 \times 10^{-4}\frac{V}{cm}} = 43cm \tag{5.2b}$$

$$1s = \frac{43cm}{29s} = 1.49cm \tag{5.2c}$$

Equation 5.2c then indicates the amount of centimetre drift per seconds, once the ball is removed from the rails.

5.2.1 Filtering the Ball Position data

Even though the quality of the acquired ball placement data had less noise than when using the OpAmps, there was still a considerable amount of noise present, as shown in figure 5.12.

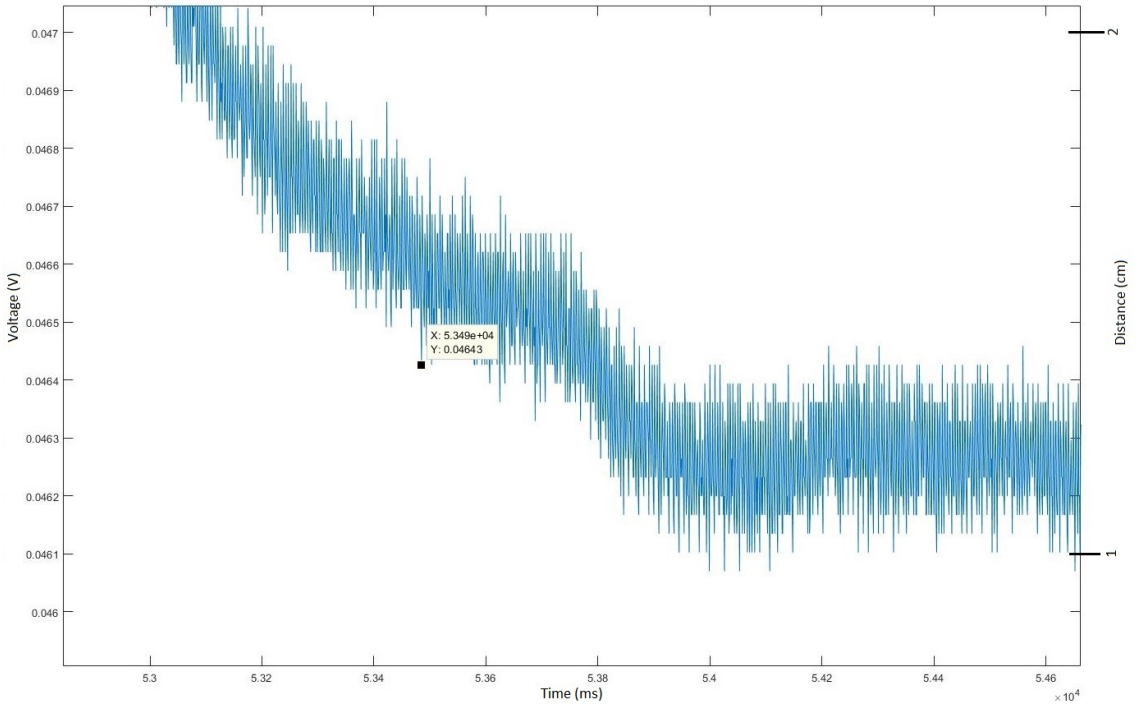


Figure 5.12: Close-up of the noise generated on the rail

Based on figure 5.12 it is possible to calculate the SNR (Signal to Noise Ratio). Since the centimetre movement in the figure is 1cm it will be used as the signal. The noise is roughly 1/3 of the signal and then the SNR can be calculated from those values as seen in equations 5.3b.

$$SNR_{voltage} = \frac{1}{\frac{1}{3}} = 3 \tag{5.3a}$$

$$dB = 20 \cdot log_{10}(3) = 9.5dB \tag{5.3b}$$

Having a signal to noise ratio of 3, which is 9.5 dB, tells us that the noise does not affect the readings much. A filter will also be implemented to deal with most of the noise. To remove the noise, a lowpass filter was designed using the signal processing toolbox (sptool) in Matlab. In order to create a decent filter the values were tweaked related to the pass and stop frequencies.

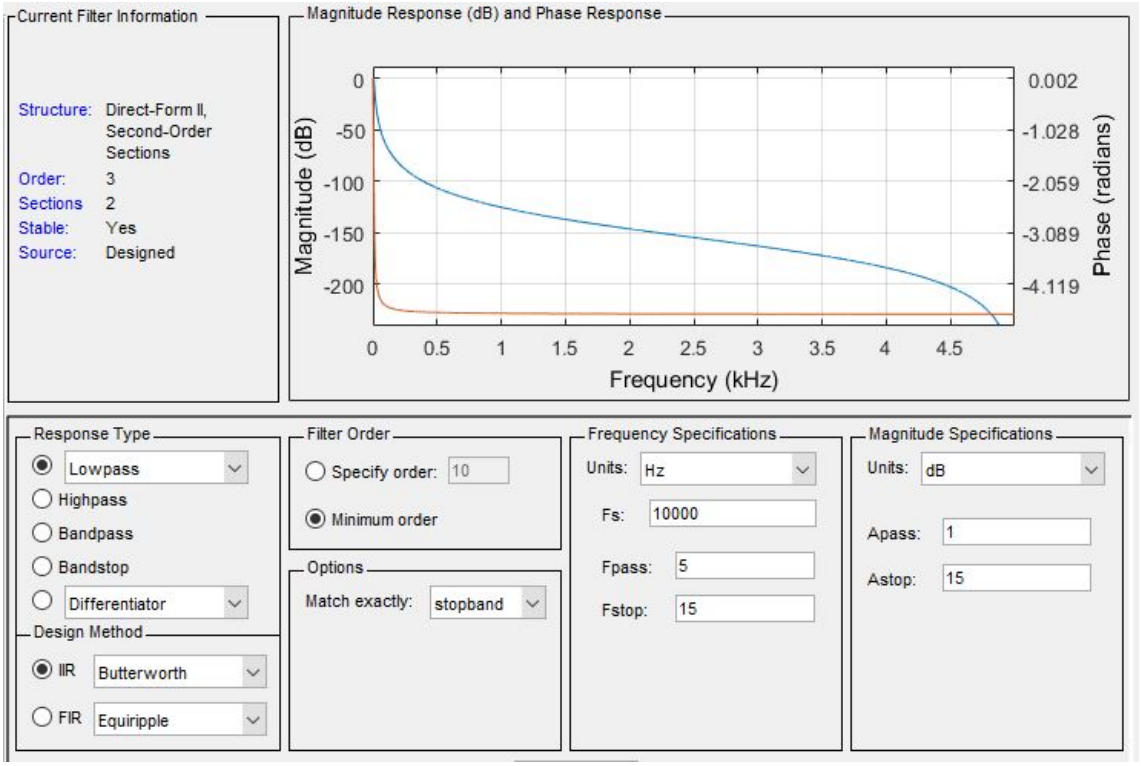


Figure 5.13: Filter 1 settings

The chosen filter values can be seen in figure 5.13 and figure 5.14 displays the group delay. If the collected data is less than 0.1 kHz the filter will have around 500 sample delay. The current filter setup will be validated once it is implemented and using real time data, where it can be determined if the group delay will have a noticeable impact.

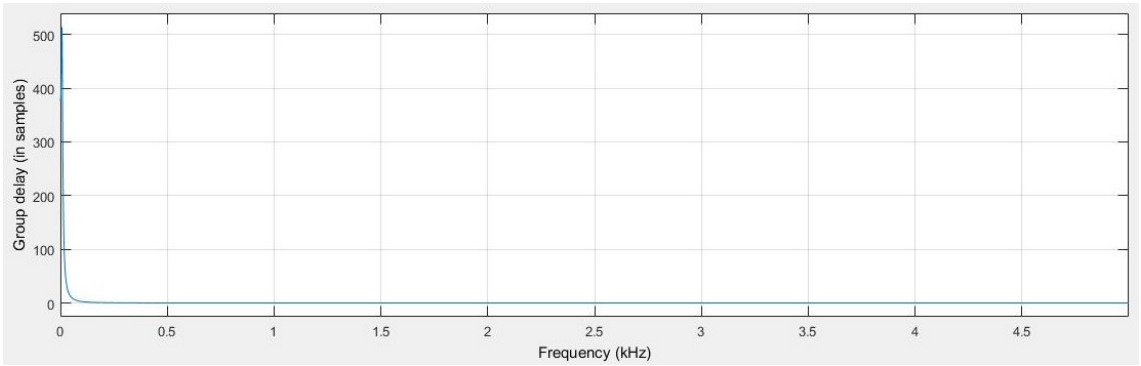


Figure 5.14: Filter 1 group delay

Figure 5.15 displays a comparison of the unfiltered and filtered rail data. The shift is the delay created while filtering the data, this reflects the shift Matlab creates while running the samples through the filter.

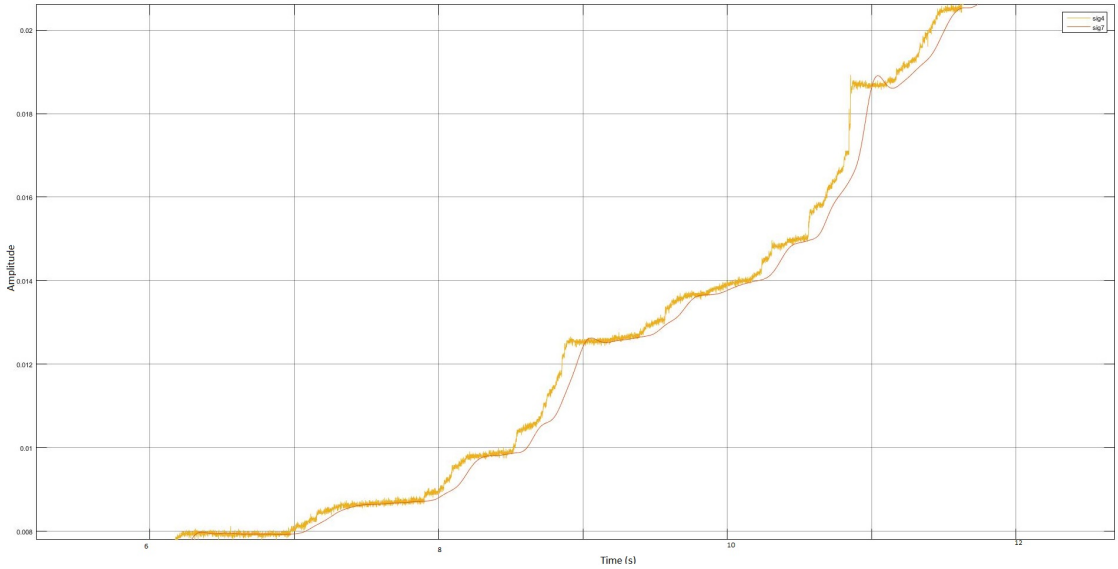


Figure 5.15: Comparison of filtered and unfiltered rail data

A test of the ball movement is shown in figure 5.16, after the raw rail data has been filtered by the designed lowpass filter from figure 5.13.

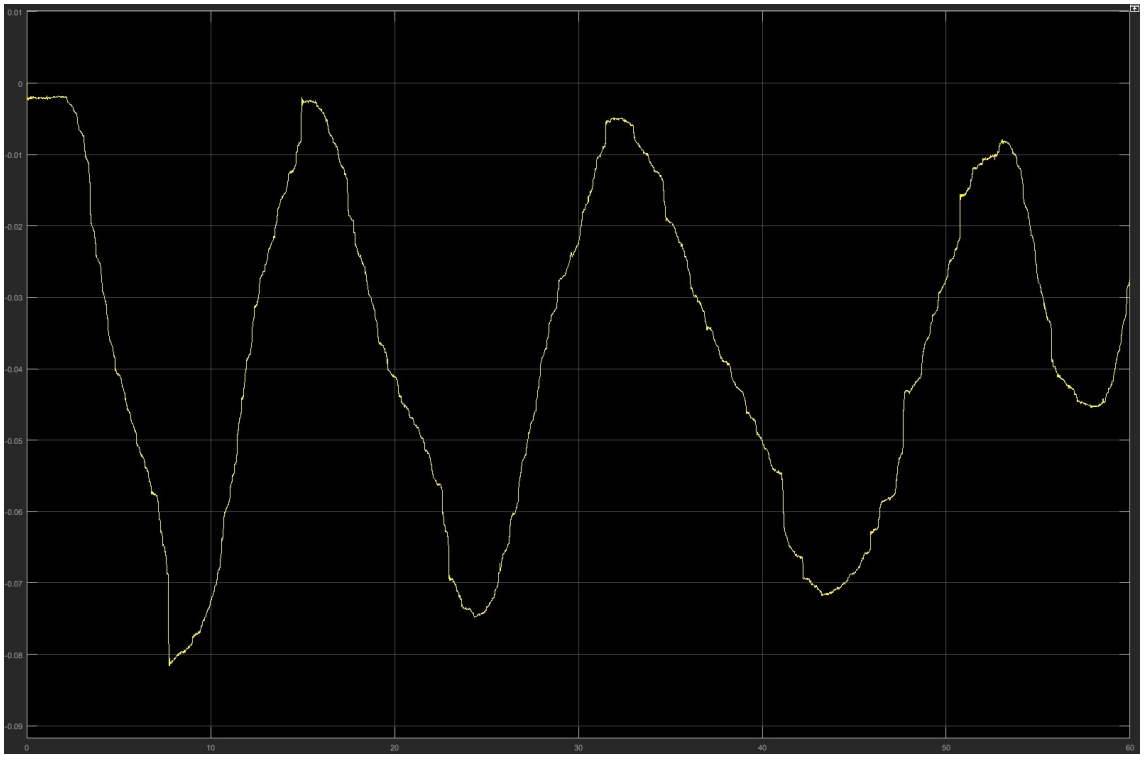


Figure 5.16: Filtered rail data

5.3 Determining Motor Coefficients

For the mathematical modelling of our beam and ball system it is necessary to know the different motor coefficients, this being the inductance, inertia and viscous friction of the given DC motor, since the motors used in the given setup had no documentation related to their specifications and there were no available datasheets.

5.3.1 Estimating Internal Resistance and Back EMF

In order to estimate the internal resistance of the given DC motor, the differential equation describing the motor, which was derived in equation 4.8, is rearranged in order to achieve a resistance value. The term $e_a(t)$ is for the back EMF that in equation 4.8 is denoted as $k_e\dot{\theta}(t)$.

$$V_a(t) = R_a i_a(t) + L_a \dot{i_a}(t) - e_a(t) \tag{5.4a}$$

$$R_a = \frac{V_a(t) - L_a \dot{i_a}(t) + e_a(t)}{i_a(t)} \tag{5.4b}$$

The ball and beam setup used has two identical unknown DC motors that can be used to an advantage, to determine the required values for equation 5.4b. The two motors were mechanically connected using a metal rod (See appendix 11.1b). Due to the motor running at a steady-state whilst calculating the resistance values, it is possible to ignore the voltage drop across the inductor.

V	e_a	$V - e_a$	i_a	R_a
10	9.34	0.66	2.81	0.23
10.5	9.88	0.62	2.83	0.22
11	10.46	0.54	2.85	0.19
11.5	10.87	0.63	2.87	0.22
12	11.31	0.69	2.98	0.23
12.5	11.66	0.84	3.07	0.27
13	12.1	0.9	3.17	0.28

Table 5.1: Mechanically connected motors

As shown in table 5.1 the value V was the supplied voltage to the one motor, whilst the back EMF (e_a) was measured on the secondary connected motor. This secondary motor only had a multimeter connected to it in order to determine e_a. The current (i_a) was measured in parallel with the primary motor, which was powered at the different voltages in order to determine the different current draw. By substituting the measured values into equation 5.4b and ignoring the inductance term, it was possible to determine different resistance values at the different voltage values. By finding the average it is concluded that the estimated armature resistance is 0.23Ω.

To determine the motor voltage constant k_e a second experiment was performed using the motors (See appendix 11.1a). This time, a single motor was left running without any load whilst the RPM was measured using SHIMPO DT-205 Hand Digital Tachometer. In order to determine the motor constant, some modifications to equation 5.4b were necessary.

$$k_e = \frac{e_a(t)}{\dot{\theta}(t)} = \frac{e_a(t)}{\omega(t)} \tag{5.5}$$

By rearranging and substituting 5.5 into equation 5.4b it is possible to create an expression for the constant at different voltage levels.

$$\omega(t)k_e = V_a(t) - R_a i_a(t) - L_a \dot{i_a}(t) \tag{5.6a}$$

$$k_e = \frac{V_a(t) - R_a i_a(t) - L_a \dot{i_a}(t)}{\omega(t)} \tag{5.6b}$$

Since this motor is also running at a steady-state, it possible to ignore the voltage drop across the inductor.

V	i_a	RPM	$\omega(\frac{rad}{s})$	K_e
10	1.303	450	47.12	0.2059
10.5	1.362	470	49.22	0.2069
11	1.371	494	51.73	0.2065
11.5	1.377	514	53.83	0.2078
12	1.474	535	56.03	0.2081
12.5	1.528	558	58.43	0.2079
13	1.566	580	60.74	0.2081

Table 5.2: Free-spinning motor

Then, by inserting the experimental data found within table 5.2 into equation 5.6b, it is possible to obtain the motor constant. The average value for K_e is then calculated to be 0.2073.

5.3.2 Internal Inductance

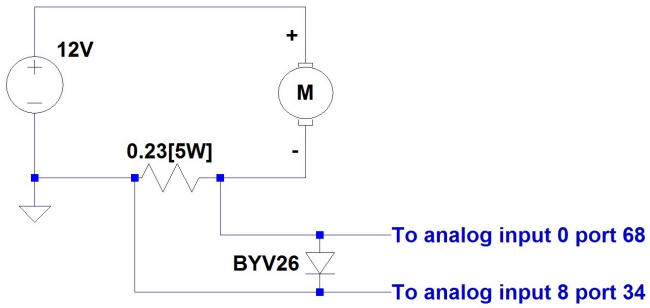


Figure 5.17: Setup for current measurement over a resistor

To determine the influence of the inductance on the motor, the setup shown in figure 5.17 was used to perform some initial experiments. The setup uses a zener diode to act as a cut-off switch, this is to protect the PCI-6229 card from voltage spikes that can possibly damage it.

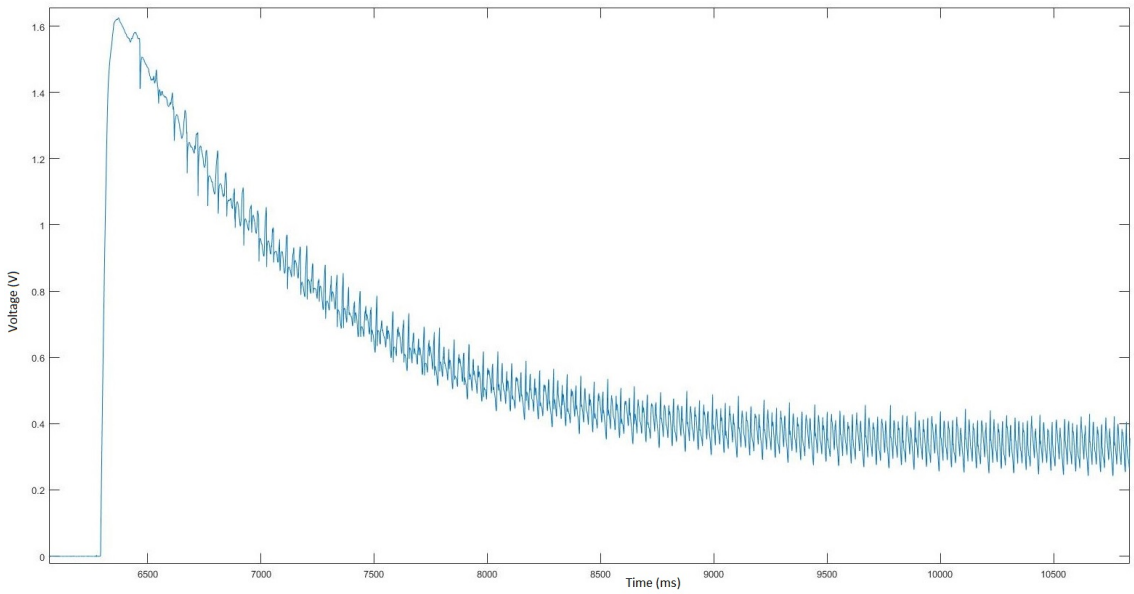


Figure 5.18: Graph of the motor inductance

The inductance can be estimated from figure 5.18. If the time constant for the electrical part of the system is really small, this results in the electrical part being much faster than the mechanical part and can therefore be removed without effecting the overall performance besides simplifying the final mathematical modelling. By looking at figure, it was determined that the effect of the inductance could be disregarded.

5.3.3 Friction effect analysis

Controlling angular velocity or angular position of a DC motor requires not only knowing the electrical coefficients for the mathematical model, but also knowing the mechanical coefficients and how to describe the friction in the motor. Friction is a tangential force that appears between two surfaces that are in contact. [15]

- **Static friction** acts between two surfaces that are sticking. To create motion, a breakaway force is required. Static friction is usually larger than the Coulomb friction.
- **Coulomb friction** is the friction that is always present. It does not depend on velocity, but on the direction of motion. It opposes velocity and has a magnitude that depends on the surfaces in contact and on the normal force.
- **Viscous friction** is directly proportional with velocity.
- **Stribeck friction** is indirectly proportional with velocity and it appears when fluid lubrification is being used.

Knowing other variables of the motor allows us to identify the friction in the motor. The viscous friction can be expressed with respect to the time constant τ and the moment of inertia J. Equation 5.7a is an expression for the torque produced by the mechanical part of the DC motor, this equation can also be found during the mathematical modelling in equation 4.1b.

$$T_m = J\ddot{\theta}(t) + b\dot{\theta}(t) \tag{5.7a}$$
$$T_m = J\dot{\omega}(t) + b\omega(t) \tag{5.7b}$$

Equation 5.7b expresses equation 5.7a in terms of angular velocity ($\omega(t)$) and angular acceleration ($\dot{\omega}(t)$). Taking the Laplace transformation of equation 5.7b results in the following first order system seen in equation 5.8d

$$T_m(s) = J\omega(s)s + B\omega(s) \tag{5.8a}$$
$$T_m(s) = (Js + B)\omega(s) \tag{5.8b}$$
$$\frac{\omega(s)}{T_m(s)} = \frac{1}{Js + B} \tag{5.8c}$$
$$\frac{\omega(s)}{T_m(s)} = \frac{\frac{1}{B}}{\frac{J}{B}s + 1} \tag{5.8d}$$

It is then possible to determine the time constant based on the first order system found in equation 5.8d. The time constant is then as following in equation 5.9, where through rearrangement it is possible to determine the viscous friction (B) or the inertia (J), depending on the known values.

$$\tau = \frac{J}{B} \tag{5.9}$$

5.3.4 Determining Moment of Inertia and Viscous Friction

To determine the moment of inertia of the DC motor, a load with a known inertia was attached to the DC motor. Then, by powering on the motor and allowing it to accelerate to constant speed whilst having the inertial load attached, then powering off, the deceleration time was recorded until the rotation came to a complete stop. By comparing an experiment with and without an inertial load attached, it is then possible to determine the moment of inertia and the viscous friction of the given DC motor. As described in section 5.3.3, the time constant for the first order system is described as shown in equation 5.9. By modifying this equation, it is possible to determine the inertia based on the two different time constants shown in equations 5.10a, 5.10b.

$$\tau_1 = \frac{J_m}{B} \tag{5.10a}$$

$$\tau_2 = \frac{J_m + J_{load}}{B} \tag{5.10b}$$

By combining these equations and cancelling out the viscous friction it is then possible, through experimentation, to retrieve the remaining values.

$$\tau_2 = \frac{J_m + J_{load}}{\frac{J_m}{\tau_1}} \tag{5.11a}$$

$$\tau_2 = \frac{(J_m + J_{load})\tau_1}{J_m} \tag{5.11b}$$

$$\tau_2 = \left(1 + \frac{J_{load}}{J_m}\right)\tau_1 \tag{5.11c}$$

$$\left(\frac{\tau_2}{\tau_1} - 1\right) = \frac{J_{load}}{J_m} \tag{5.11d}$$

The final and simplified equation for determining the inertia is shown in equation 5.12. Through experimentation, the values τ_1 and τ_2 are determined.

$$J_m = \frac{J_{load}}{\left(\frac{\tau_2}{\tau_1} - 1\right)} \tag{5.12}$$

Initially an experiment to determine the moment of inertia was performed using the same electrical setup for recording data as shown figure 5.17. But by plotting the current decay of the motor (without a load) does not display any useful information,

since it just shows the current whilst the motor is powered off. This can be seen in figure 5.18.

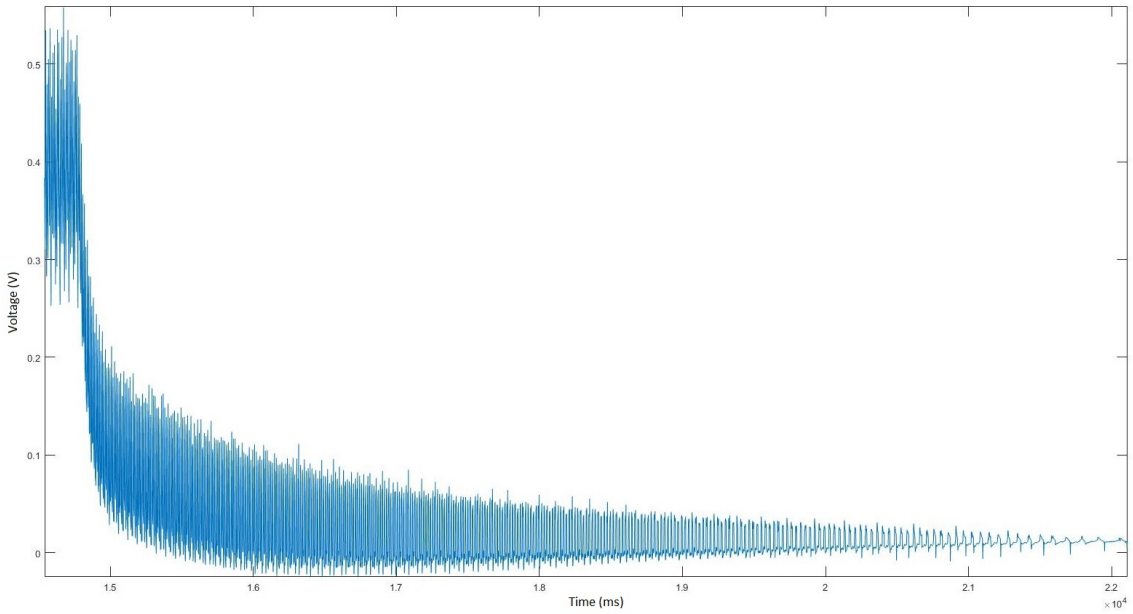


Figure 5.19: Drop in current when motor is turned off

Measuring the current change of the circuit did not provide data usable to calculate the inertia, therefore the electrical setup was altered to what can be seen in figure 5.20. This time it is the voltage across the motor being measured, instead of the current.

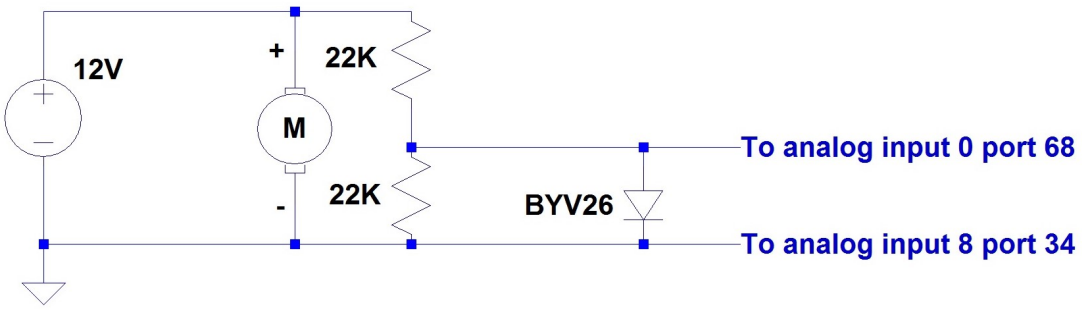


Figure 5.20: Setup for measuring voltage over the motor

To calculate the inertia, the motor is powered on till it reaches a steady state and then it is switched off. Measurements of the voltage were taken until the motor came to a complete stop.

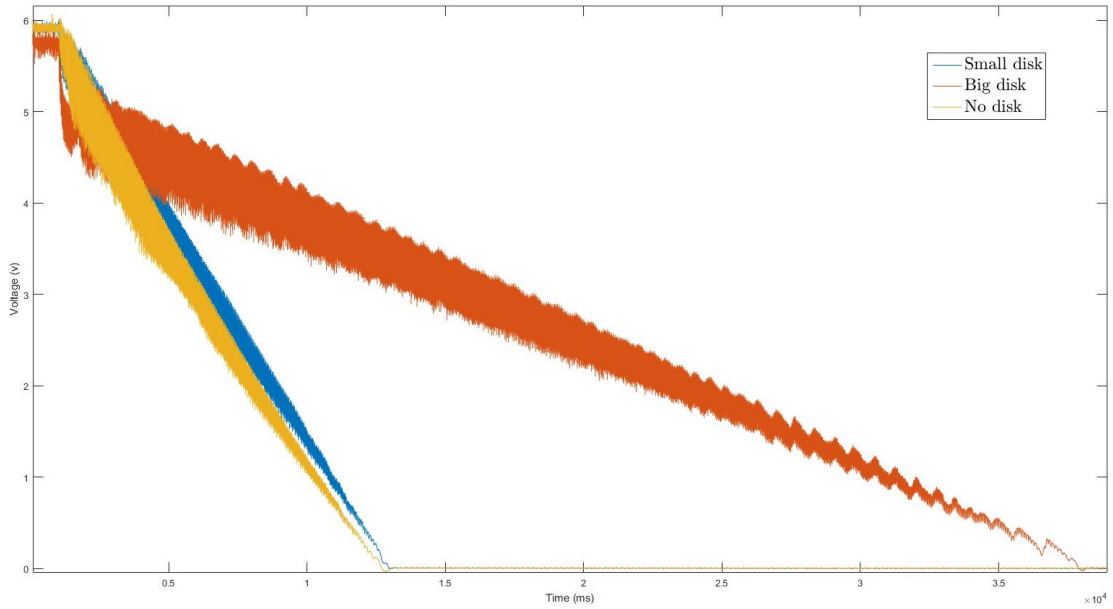


Figure 5.21: Motor inertia compared to small disk and big disk

Figure 5.21 shows three different experiments, in order to determine the moment of inertia. The yellow data is the motor itself with no load attached, where the blue is a small disk that was attached, but the attached disk did not have enough inertia to affect the measurements. Where finally the orange data is a big disk that increases the run-out of the motor significantly, due to having a large enough inertia to affect the motor. See appendix 11.2 and 11.3 to see the setup with the externally connected load.

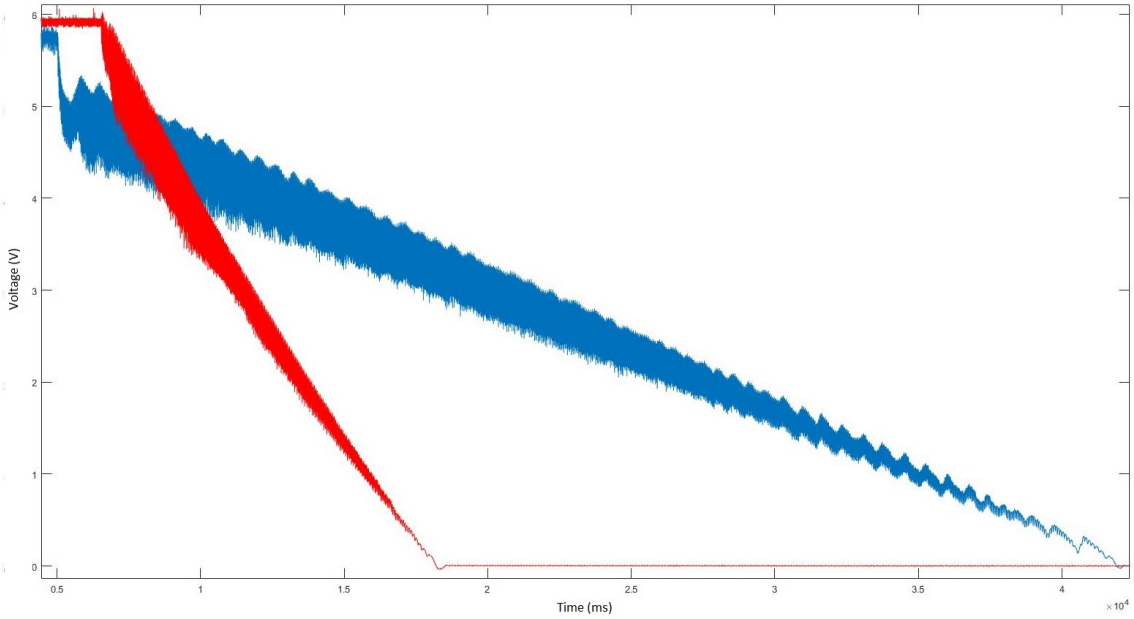


Figure 5.22: Comparison of no disk and big disk inertia

Since the smaller disk yielded insignificant results, the larger disk was instead used in order to perform the two experiments. Figure 5.22 shows the run-out tests with the big disk (blue) and without any external inertial load (red). The inertia of the big disk is determined by equation 5.13

$$J_m = \frac{1}{2} \cdot M \cdot r^2 \tag{5.13}$$

By inserting the measured values of the big disk, the calculated inertia is found in equation 5.14.

$$J_m = \frac{1}{2} \cdot 8.52\ kg \cdot 0.1175^2\ m = 0.0588\ kg \cdot m^2 \tag{5.14}$$

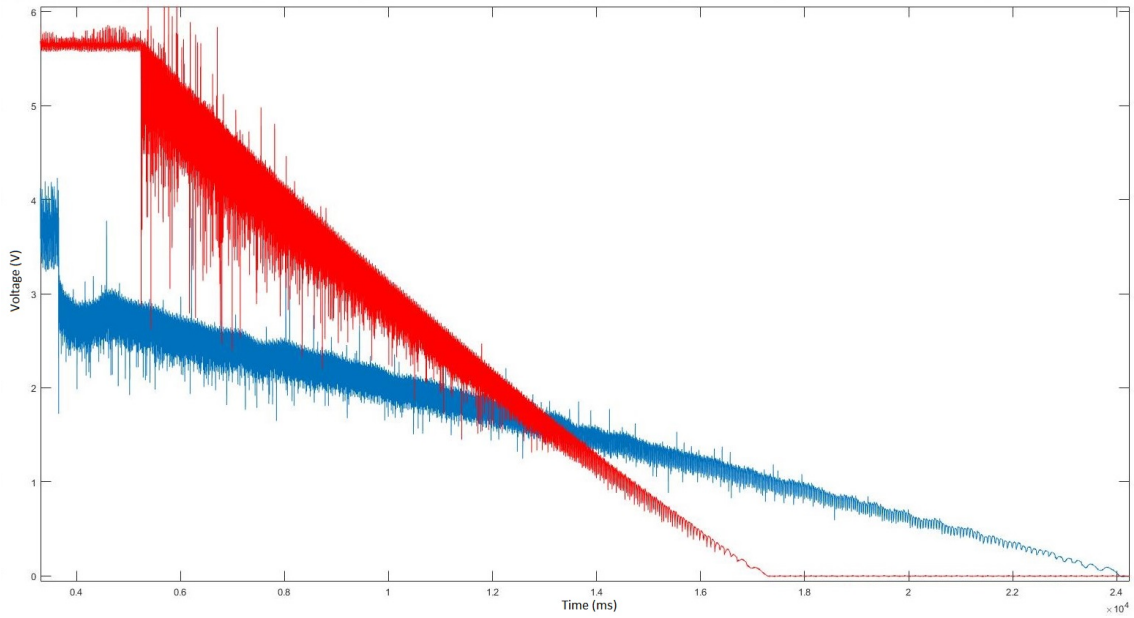
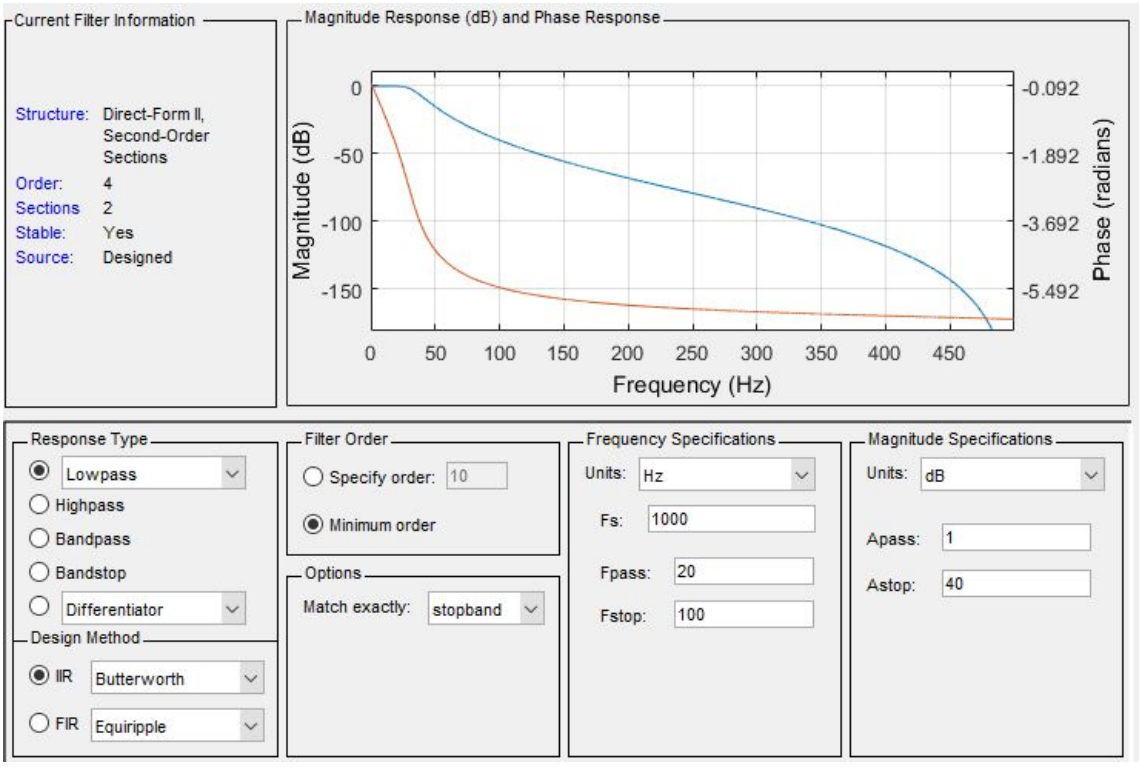


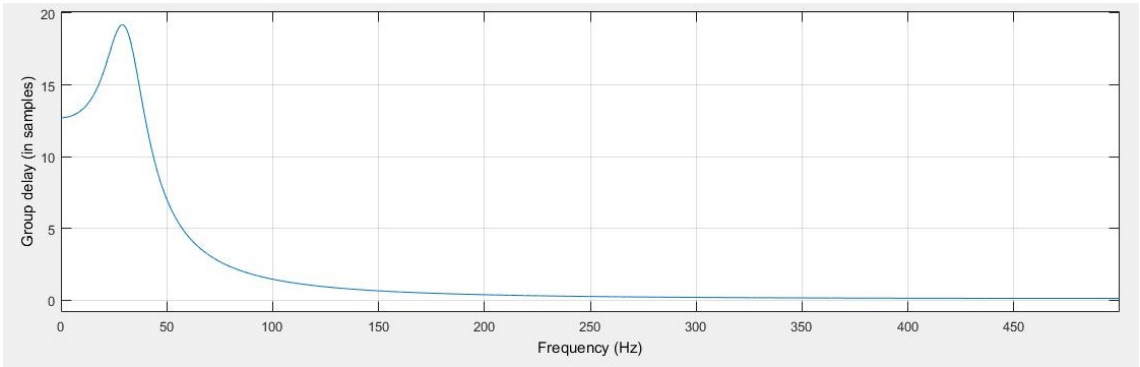
Figure 5.23: Effects of decoupling the circuit

As an attempt to reduce the capacitance effect from the power supply used to power the motor, instead of only switching off the supply, it was also decoupled, by removing the power source completely. These results can be seen in figure 5.23, which did not really make a difference besides adding additional noise to the measurements. Therefore the data collected as shown in figure 5.22 will be used for calculating the inertia.

In order to properly determine the time constants of the two experiments, it was necessary to perform some digital filtering in order to interpret the data. Filter values that resulted in good filtering without causing much delay and magnitude change can be seen in figure 5.24a. This current filter has a larger delay beneath a frequency of 100Hz, at most around 20 samples, as shown in figure 5.24b.



(a) Filter settings



(b) Filter group delay

Figure 5.24: Filter specification

A comparison between the filtered and unfiltered inertia test data can be seen in figure 5.25. The filtered data is indicated by the thick lines inside of the unfiltered data.

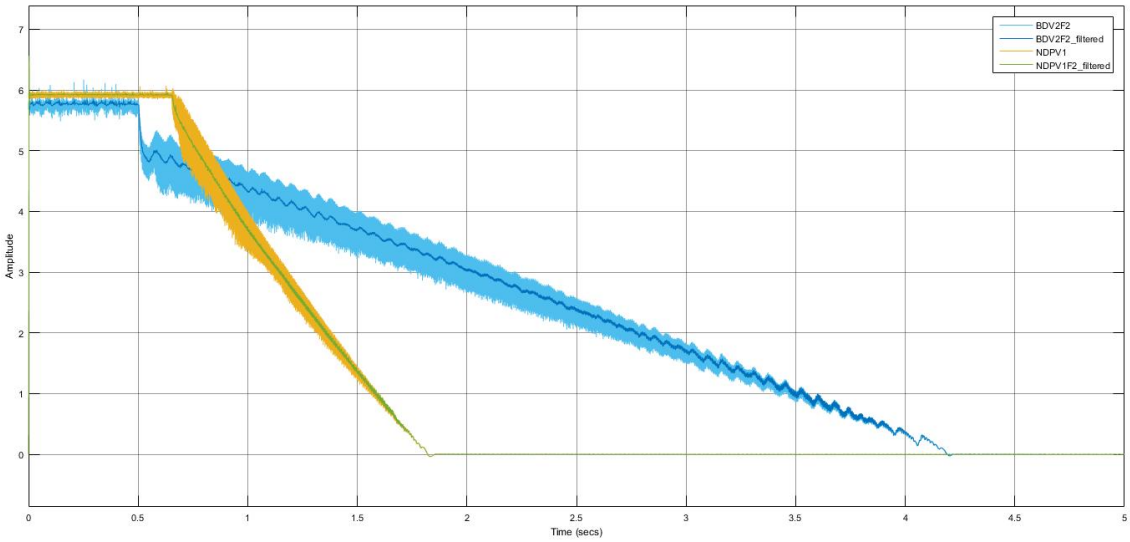


Figure 5.25: Comparison of unfiltered and filtered motor data

To determine the time constant, we are looking at the time it takes the steady state to reach $\frac{1}{e} \approx 36.8\%$. The time constant is then the Δt, between the steady-state value and the 36.8% mark. This is shown in figure 5.26 where the time constant for the motor without an attached disk is determined. [16]

$$\Delta T = \tau_1 = 1.327s - 0.647s = 0.679s \tag{5.15}$$

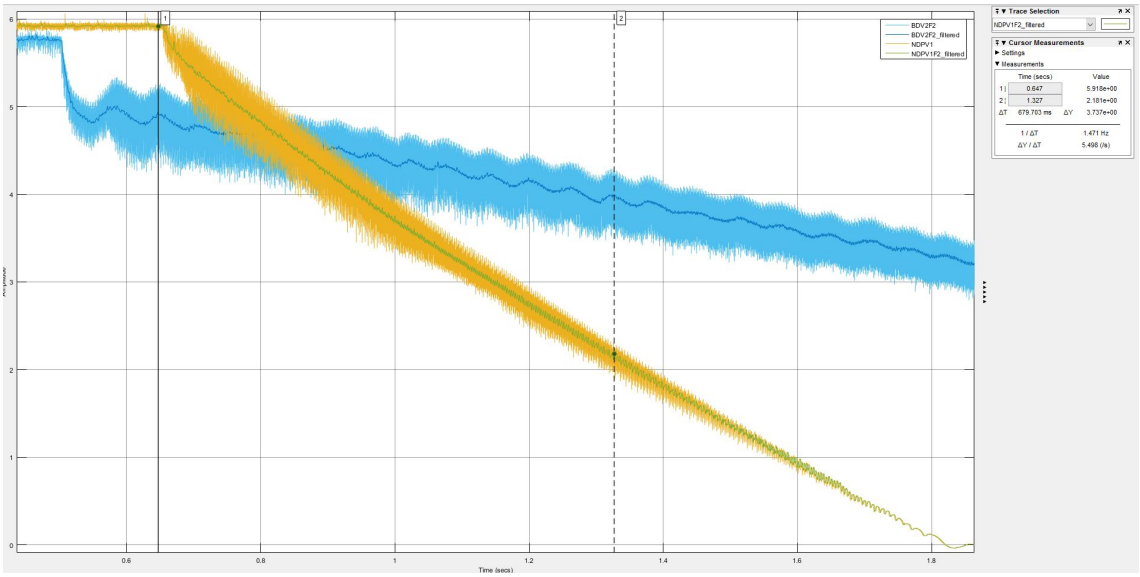


Figure 5.26: Motor inertia calculations

The time constant for the experiment with the externally connected inertial load is determined the same way, this can be seen in figure 5.27.

$$\Delta T = \tau_2 = 2.243s - 0.503s = 1.740s \tag{5.16}$$

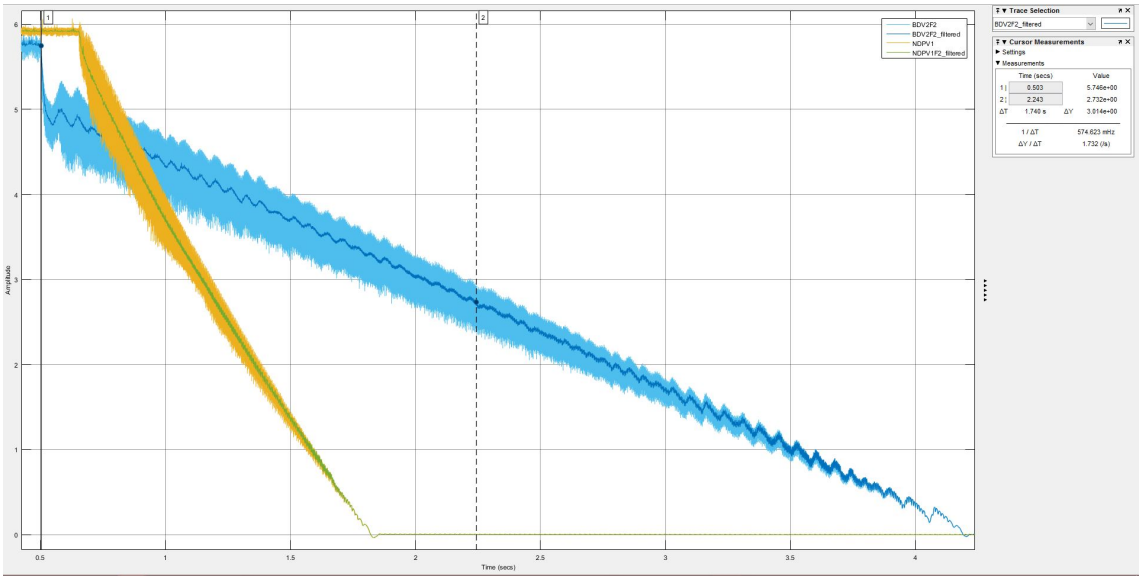


Figure 5.27: Motor and big disk inertia calculations

By inserting the two measured time constants into equation 5.12 and also by substituting the calculated inertia for the big disk from equation 5.14, it is possible to determine the moment of inertia for the DC motor, as shown in equation 5.17

$$J_m = \frac{0.0588\ kg \cdot m^2}{\frac{1.740s}{0.679s} - 1} = 0.0376\ kg \cdot m^2 \tag{5.17}$$

Since we now know the inertia of the motor, it is possible to determine the viscous friction using τ_1 and the calculated inertia from equation 5.17.

$$B = \frac{0.0376\ kg \cdot m^2}{0.679s} = 0.0554\ N \cdot m \cdot s \tag{5.18}$$

Chapter 6

Model Validation and Performance

6.1 Logical Block Diagrams

Name	Description	Unit
T_m	Motor torque	Nm
T_L	Load torque	Nm
K_t	Motor constant	
K_e	Electromotive force constant	
$i_a(t)$	Armature current	A
R_a	Armature resistance	Ω
L_a	Armature inductance	H
J	Moment of inertia of the rotor	$kg \cdot m^2$
B	Viscous friction	Nms
$\theta(t)$	Motor angle	rad
$V(t)$	Input voltage	V

Table 6.1: Table of nomenclature for the logical block diagrams

One of the initial steps in order to validate the calculated model is to compare it to the logical block diagram of the DC motor. The logical block diagram of the motor can be seen in figure 6.1.

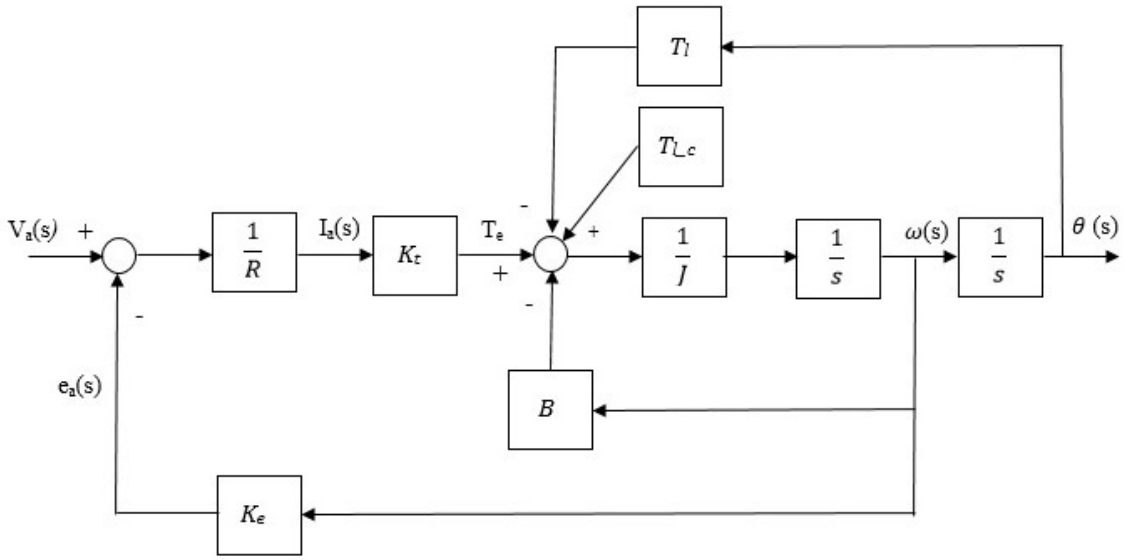


Figure 6.1: Block diagram of the motor part

In order to calculate the transfer function for the open loop system using the block diagram, we can use the feed back equation shown in 6.1.

$$\frac{Y(s)}{R(s)} = \frac{G1}{1 + G2G1} \tag{6.1}$$

To simplify one of the inner feedback loops, it is possible to remove the loop that involves the viscous friction (B) and make it part of the open loop sequence, this then results in equation 6.2.

$$\frac{\frac{1}{J \cdot s}}{1 + B \cdot \frac{1}{J \cdot s}} = \frac{1}{(J \cdot s + B)} \tag{6.2}$$

The altered logical block diagram for the system can then be seen in figure 6.2.

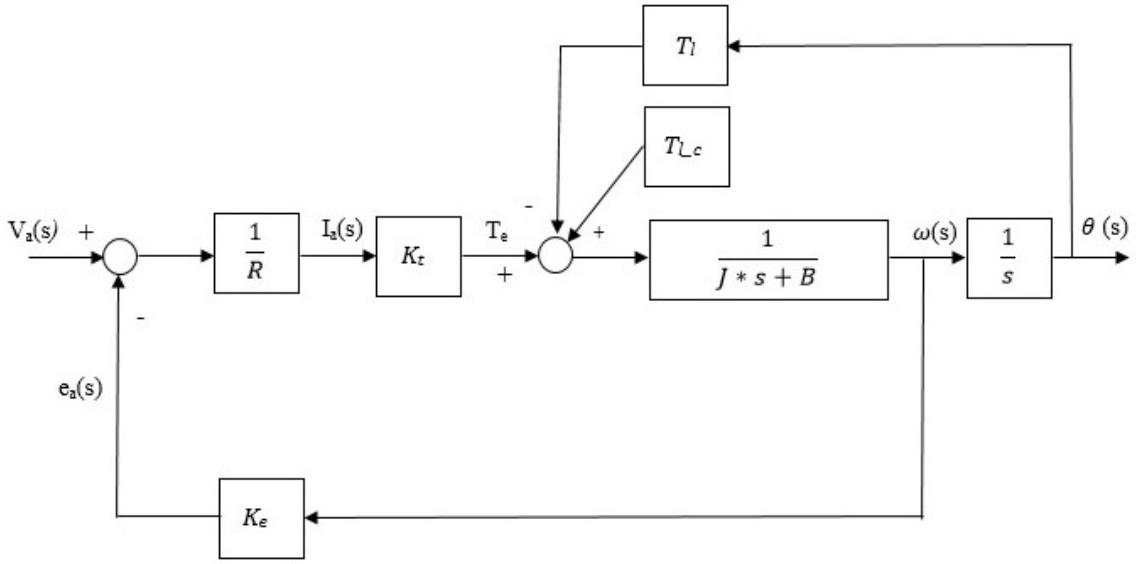


Figure 6.2: Block diagram after B has been moved

In a similar procedure T_L can be moved into the open loop as seen in equation 6.3.

$$\frac{\frac{1}{(J\cdot s+B)\cdot s}}{1+T_L\cdot\frac{1}{(J\cdot s+B)\cdot s}}=\frac{1}{J\cdot s^2+B\cdot s+T_L}=\frac{1}{(J\cdot s+B+\frac{T_L S}{s})}\cdot\frac{1}{s} \tag{6.3}$$

The system then looks as following in figure 6.3.

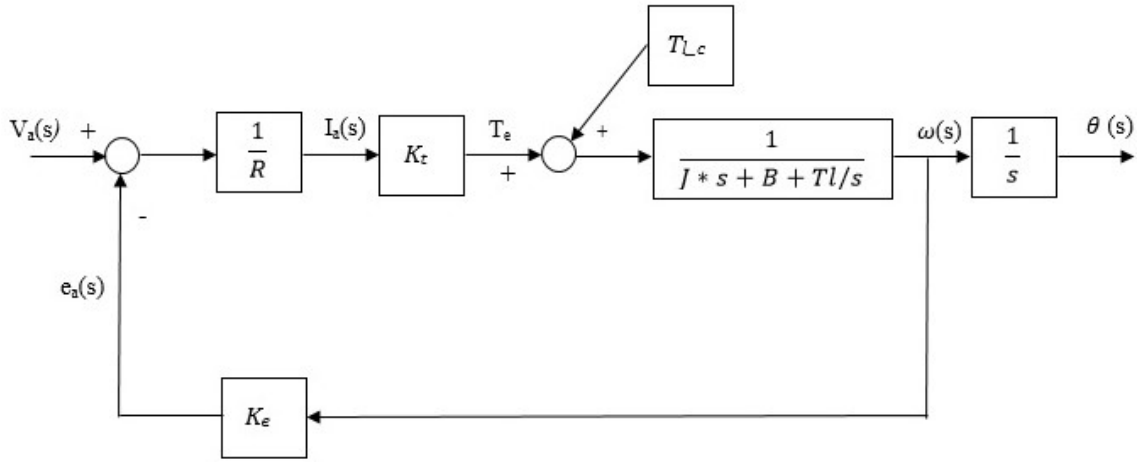


Figure 6.3: Block diagram after Tl has been moved

The final item that needs to be moved into the open loop system is the constant k_e. As with the previous simplifications, this is done in a similar manner and is shown in equation 6.4.

$$\frac{\frac{K_t}{(J\cdot s+B+\frac{T_L}{s})\cdot R}}{1+K_e\cdot\frac{K_t}{(J\cdot s+B+\frac{T_L}{s})\cdot R}}\cdot\frac{1}{s}=\frac{K_t}{(J\cdot s^2+B\cdot s+T_L)\cdot R+K_e\cdot K_t\cdot s} \tag{6.4}$$

Then finally, by disregarding the constant torque T_lc and handling it individually, it is possible to construct the open loop in a single transfer function block. The final and single block is equal to the theoretical calculations done previously in the report in equation 4.16b, which validates that the theoretical calculations are correct. By connecting the motor transfer function with the gear ratio from equation 4.25 and the ball transfer function from equation 4.37, we get the open loop block diagram of the ball and beam system seen in figure 6.4.

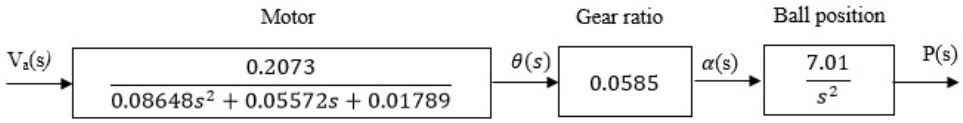


Figure 6.4: Block diagram of the ball and beam system

6.2 Linearised and Non-Linear Comparison

During the mathematical modelling, the torque from the beam at different angles was taken into consideration, even though it was non-linear. Therefore, in order to create our linear control solution, it had to be linearised and has therefore impacted the accuracy of the mathematical model. As shown previously in equation 4.5, the non-linear expression for the beam angle is repeated in equation 6.5.

$$T_L = 0.23568 \cdot \sqrt{1 - 0.125 \cdot \cos^2(\theta)} \cdot \sin\left(\theta + \frac{\pi}{2} - \sin^{-1}\left(\frac{0.06 \cdot cos(\theta)}{0.17}\right)\right) \tag{6.5}$$

In order to compare the linearised model and the non-linear one, it is necessary to perform step responses on both systems. The differential equations for the mechanical and electrical parts of the DC motor are rearranged so that it is possible to create a combined system. The initial equations can be found in equation 4.8 and equation 4.2.

$$L_a\dot{i}(t) + R_a i(t) = V(t) - k_e\dot{\theta}(t) \tag{6.6a}$$

$$i(t) = \frac{V(t) - k_e\dot{\theta}(t)}{R_a} \tag{6.6b}$$

By setting the inductance term equal to zero in equation 6.6, it simplifies the equation. It is then solved for the current, so that it is possible to use it as an input for the mechanical equations seen in equation 6.7.

$$J\ddot{\theta}(t) + b\dot{\theta}(t) + T_L = k_t i_a(t) \tag{6.7a}$$

$$\ddot{\theta}(t) = \frac{k_t i_a(t) - b\dot{\theta}(t) - T_L}{J} \tag{6.7b}$$

Equation 6.7 is solved for the highest order of derivative, since it is then possible to integrate the output in order to use the different derivatives as inputs.

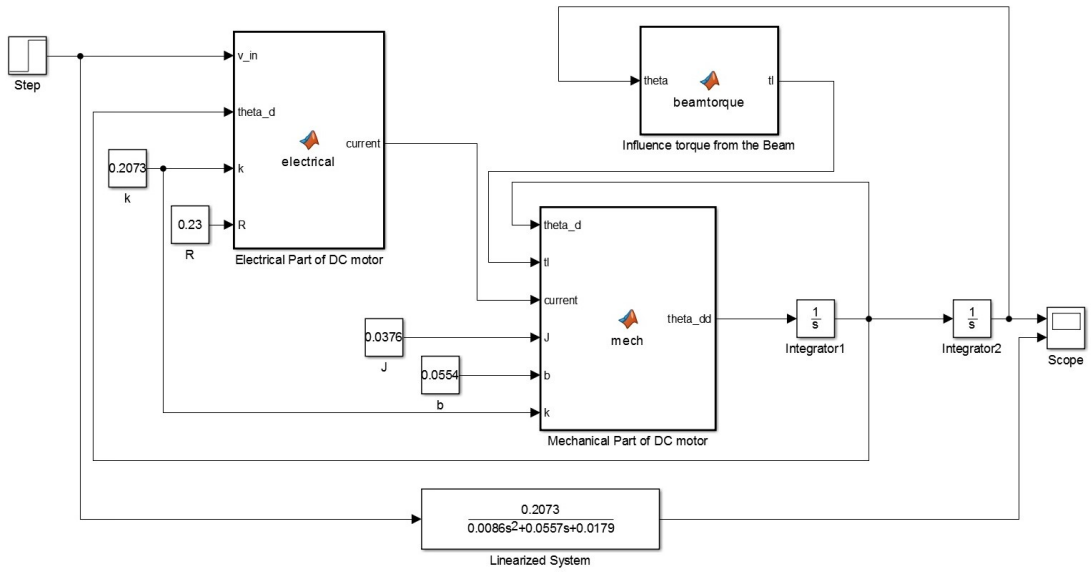


Figure 6.5: Simulink implementation of the linearised and non-linearised transfer function

Figure 6.5 displays the Simulink representation and comparison of the non-linearised system and the linearised transfers function. The output of the mechanical differential equation is integrated twice in order to retrieve the correct output of angular position and in order to ensure that both the linearised and non-linearised system has the same input and output relationship.

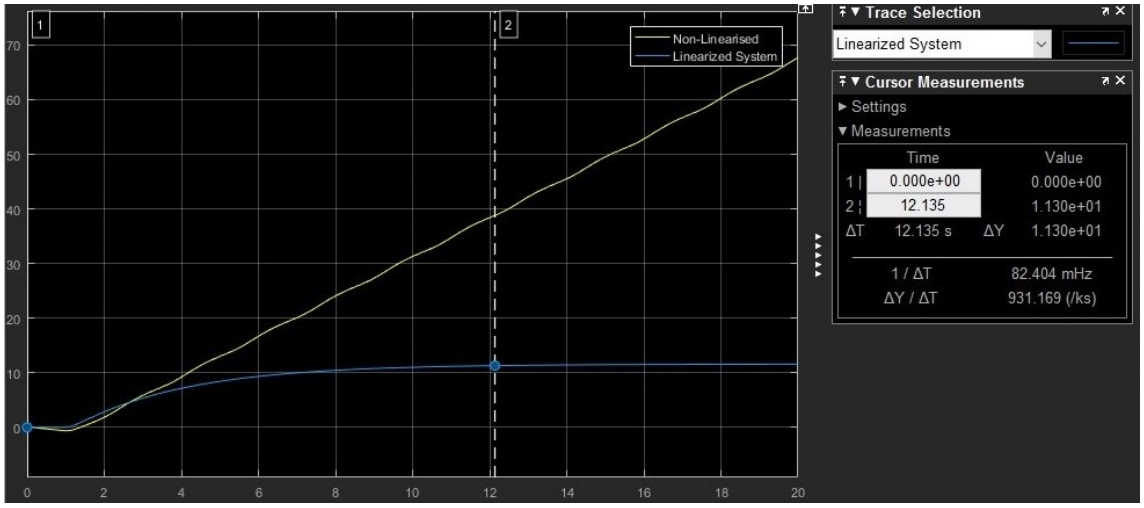


Figure 6.6: Comparison of the linearised and non-linearised motor model

At the intersection point of 4.5 radians on the plot of the linear and non-linear system, shown on figure 6.6, the linearised system begins to reach its maximum value of around 11 radians. At the equilibrium point of the system, the maximum allowed distance it can can move will be ±1.5 radians, which is a valid interval within this linearisation. Figure 6.7 shows how the linearised model acts similar to the non-linear one, which makes the linearised transfer function suitable to use as a replacement for the non-linear one.

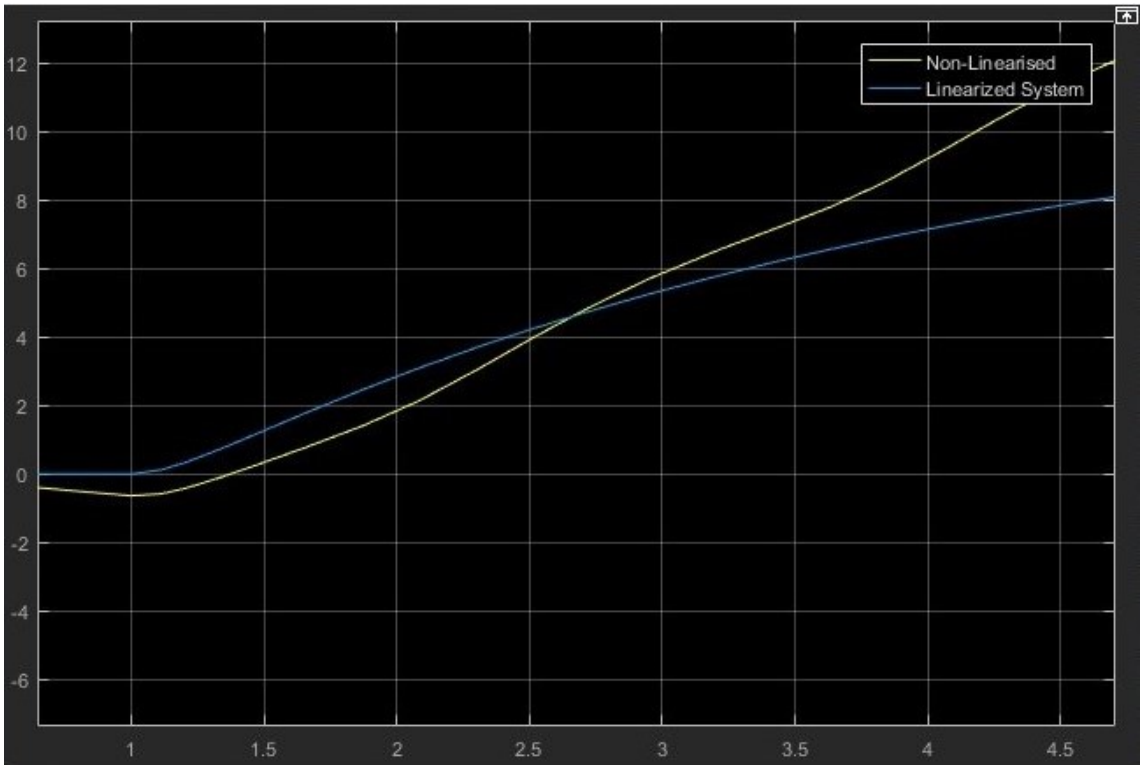


Figure 6.7: Comparison of the linearised and non-linearised motor model

For the real life implementation of the linearised model it is necessary to take the constant from the linearised torque load into account. In order to calculate the voltage needed to compensate for the constant, all dynamics of the differential equations must be set to zero. By inserting the constant values calculated previously in the report for T_L and k_t into equation 6.8b, it is possible to solve for the current.

$$J\ddot{\theta}(t) + b\dot{\theta}(t) + T_L = k_t i_a(t) \tag{6.8a}$$
$$T_L = k_t i_a(t) \tag{6.8b}$$
$$0.206 = i(t) \cdot 0.2073 \tag{6.8c}$$

The calculated value for i(t), 0,9937A, from equation 6.8c is then inserted into the electrical equation related to the DC motor.

$$i(t) = \frac{V(t) - k_e\dot{\theta(t)}}{R_a} \tag{6.9a}$$
$$i(t) = \frac{V(t)}{R_a} \tag{6.9b}$$
$$0.9937 = \frac{V(t)}{0.23} \tag{6.9c}$$
$$V(t) = 0.2286 \tag{6.9d}$$

This results in the voltage needed being 0.2286V, this will be added in the real-life implementation of the system.

Chapter 7

Controller Design

In order to develop a control solution for the entire ball and beam system, specific requirements for both the inner and outer loops of the system will be defined. The inner loop being a being the angular position of the motor and the outer loop the ball position.

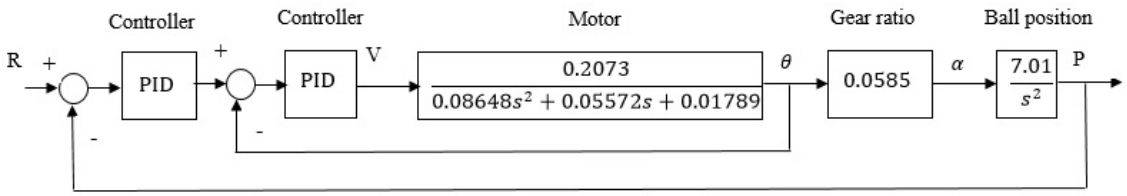


Figure 7.1: Overview of the desired system with two controllers

Figure 7.1 shows a possible control suggestion for the system. It contains two PID controllers, one for the inner loop and one for the outer loop. Each loop will be configured so that they meet their individual requirements.

In the coming sections there will be a brief run down of some controllers available to the classical control theory and thereafter sections defining the requirements for each loop and also the calculations for the specific controller values will be performed.

7.1 PID controller theory

PID stands for Proportional Integral Derivative, which are the names of the terms used in the calculation for the manipulation of the control variable. The PID controller continuously calculates the system input based on the error value, which is the difference between the setpoint and the measured value. The scope of such a controller is to minimize the error by adjusting a control variable. The equation for a PID controller is

$$D(s) = k_p + \frac{k_I}{s} + k_D s \tag{7.1}$$

where k_p is the proportional gain term, k_I is the integral gain term and k_D is the derivative gain term. The proportional term produces an output value which is proportional to the error value. Therefore, an expression for the proportional term a multiplication of the error and the proportional gain term k_p. The proportional term speeds up the system but also introduces overshoot and steady-state error. If the proportional term is too large, it will introduce oscillations making the system unstable if the value is too large. [17]

$$P_{out} = k_p e(t) \tag{7.2}$$

While the proportional term considers the error only at the time of the controller calculation, the integral term considers the history of the error, since integration accumulates over time. The integral term removes the steady-state error, but also introduces some overshoot. An expression for the integral term is the following

$$I_{out} = k_I \int_0^t e(\tau)dt \tag{7.3}$$

Derivative action predicts system behaviour and thus improves settling time and stability of the system. The derivative term helps combat the overshoot introduced by the proportional and integral terms. An expression for the derivative term is

$$D_{out} = k_D \frac{de(t)}{dt} \tag{7.4}$$

7.2 Lead and lag compensation

Lead and lag phase compensators are different types of controllers than PID. It is a popular opinion that controllers and compensators are the same and can be used to obtain the desired performance of a system. They can also be used to make a system more stable and minimize overshoot. They increase the steady state accuracy, the results of this being a less stable system. Using compensators implies changes in the transfer function of the system because of the poles and zeros introduced in the system, the result being the changes in performance. [18]

Choosing one of the two types of compensators is done by taking a look at the poles and zeros of the system. The difference between phase lead and lag compensator is that in a phase lead the zero is closer to the origin than the pole. And a phase lag compensator is when the zero is further away from the origin than the pole. [18]

The effects of using the two methods of compensation are being compared in the table below.

Lead compensator	**Lag compensator**
High pass	Low pass
Approximates derivative plus proportional control	Approximates integral plus proportional control
Contributes phase lead	Attenuation at high frequencies
Increases the gain crossover frequency	Decreases the gain crossover frequency
Increases bandwidth	Reduces bandwidth

Table 7.1: Effects of lead and lag compensation techniques [19]

7.3 Inner Loop Controller

The controller output for the inner loop will be for the angular position of the motor. The performance requirements have been chosen based upon estimations since we had no practical experience in this area. The requirements are as following

- Overshoot: 10%
- Settling time: 0.1 second
- Steady state error: 1%

To achieve these goals the system was simulated and calculations were done in Matlab. Firstly, the open loop step response performance of the system can be seen in figure 7.2.

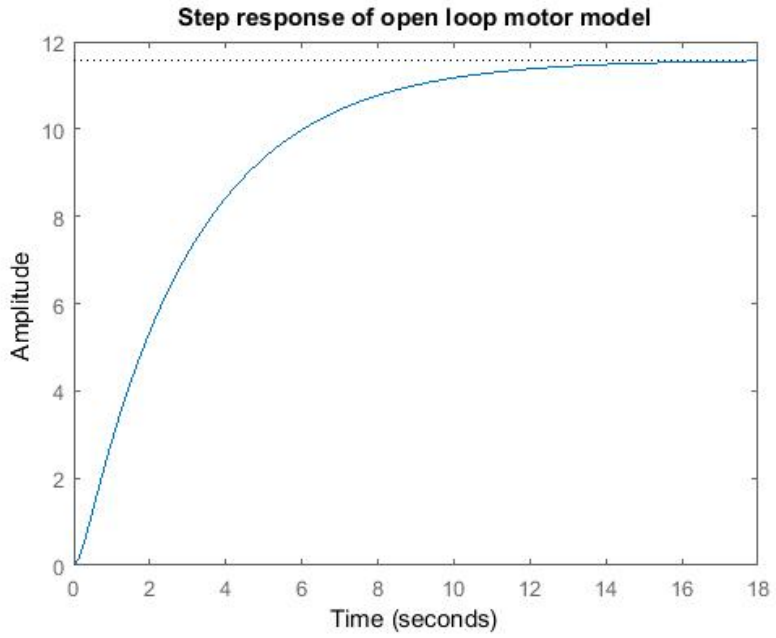


Figure 7.2: Step response of the open loop system

As expected, the motor angle changes over time, until it reaches a limit in terms of the counter-torque produced by the beam. Without the limitation of the counter-torque from the beam, the motor angle would continue to change indefinitely. After applying unity feedback, another test was performed and can be seen in figure 7.3.

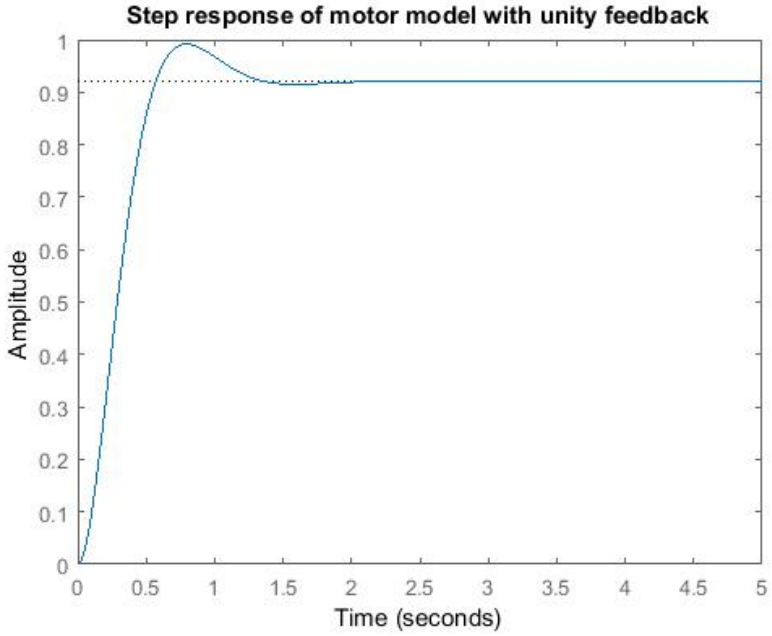


Figure 7.3: Step response of the system with unity feedback

Based on the simulation we can conclude that the system is now stable and able to settle at a desired value, but still, the settling time is too long and the steady state error is too high. Therefore, a controller needs to implemented. By finding the natural frequency of the system and damping ratio, Matlab can be used to configure a P-controller that matches our desired inner loop specifications. First, the damping ratio is calculated based on equation 7.5.

$$M_p = e^{-\pi \cdot \zeta / \sqrt{1-\zeta^2}} \tag{7.5a}$$

$$\ln 0.1 = -\pi \cdot \zeta / \sqrt{1-\zeta^2} \tag{7.5b}$$

$$\zeta = 0.5912 \tag{7.5c}$$

Then the undamped natural frequency can be calculated. see equation 7.6.

$$e^{-\zeta \cdot W_n \cdot T_s} = 0.01 \tag{7.6a}$$

$$-0.5912 \cdot W_n \cdot 0.1 = \ln 0.01 \tag{7.6b}$$

$$W_n = 77.84 \frac{rad}{sec} \tag{7.6c}$$

By creating a root locus plot and plotting the desired values for the natural frequency and damping ratio we can select a valid point from the root locus plot and use the specified gain for our P-controller. See figure 7.4.

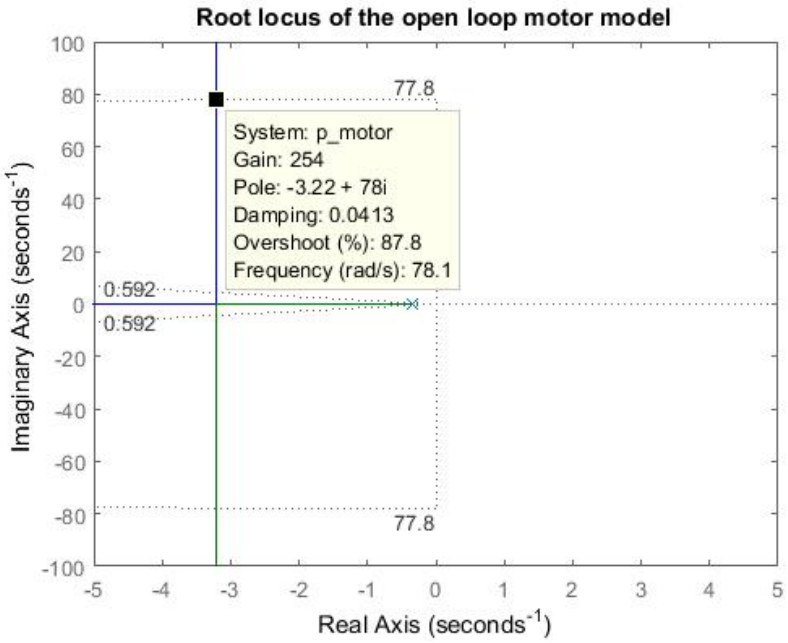


Figure 7.4: Root locus of inner loop

After adding the P-controller with a gain of 254, the overshot is higher than the 10% requirement. The simulation in Simulink shows that a p-controller is not sufficient. The overshoot and heavy oscillation of the system can be seen in figure 7.5.

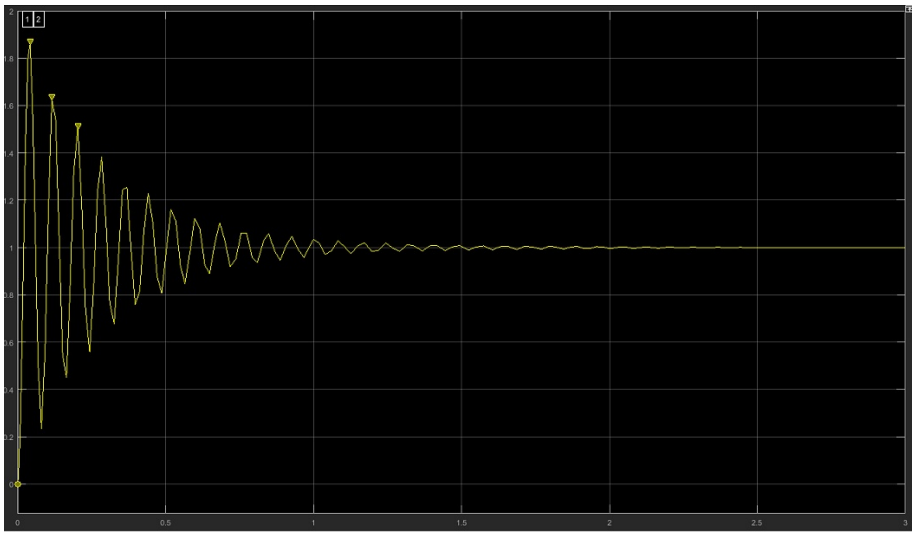


Figure 7.5: Simulation after adding a P-controller

To combat the overshoot, it is possible to add a derivative part to the control solution. It is then possible to minimize the overshoot as seen in figure 7.6.

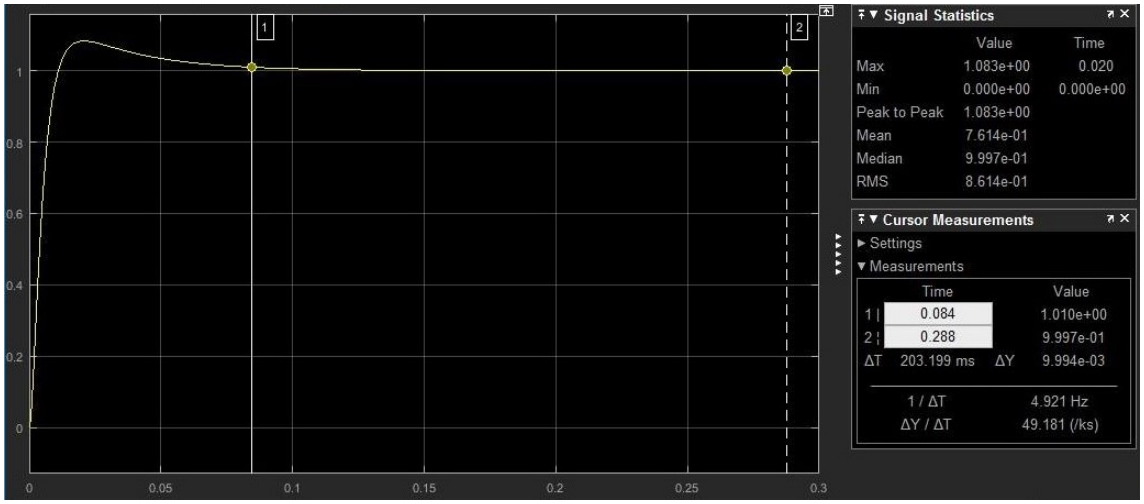


Figure 7.6: Simulation after adding a D-controller

The settling time is also below the 0.1 second limit, which satisfies the specified requirement for the inner loop. Once the model was simulated in Simulink it was discovered that it did not react the same way as a similar model in Matlab. In the real world implementation noisy measurements would cause problems related to the derivative part and a low-pass filter would in most cases be implemented to combat the noise. In our Simulink model it can be simulated by adding a filter as shown in equation 7.7, where N is the filter coefficient.

$$\left(\frac{s \cdot N}{s + N}\right) \tag{7.7a}$$

After adding the PD-controller with the added filtering for the derivative, the Simulink implementation is shown in figure 7.7. The step response of the system is stable and within the specified requirements for the inner loop.

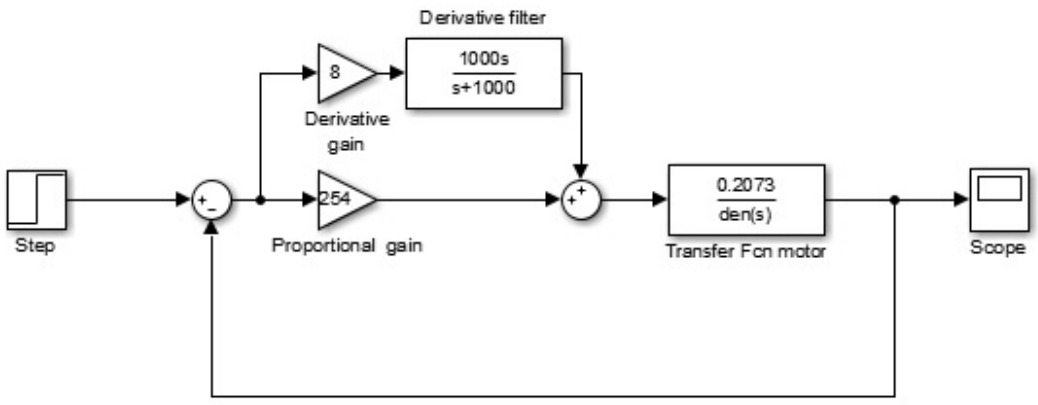


Figure 7.7: Motor model with a pd-controller and a filter

7.4 Outer Loop

The outer loop consists of a controller that takes the error of the ball position as an input, where the desired output is the ball position based on a given setpoint. The performance specifications for the outer loop, which is the entire system, is as follows

- Overshoot: 10%
- Settling time: 10 seconds
- Steady state error: 1 centimetre

To achieve these goals the system was simulated in Matlab. The open loop step response performance of the system can be seen in figure 7.8.

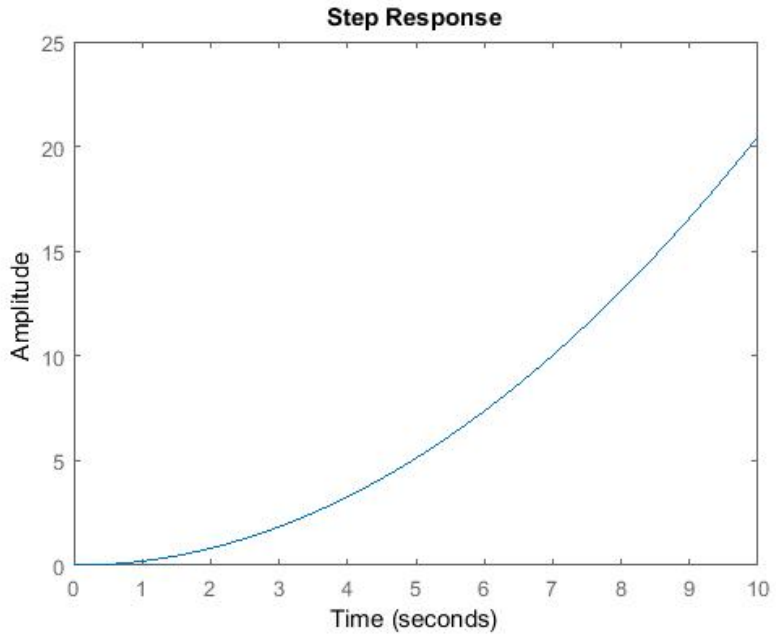


Figure 7.8: Step response of ball position.

Figure 7.8 clearly shows that the system is not stable and the ball would, after some time, just roll off the beam if it is not physically stopped. After adding unity feedback, the system response was the following seen in figure 7.9.

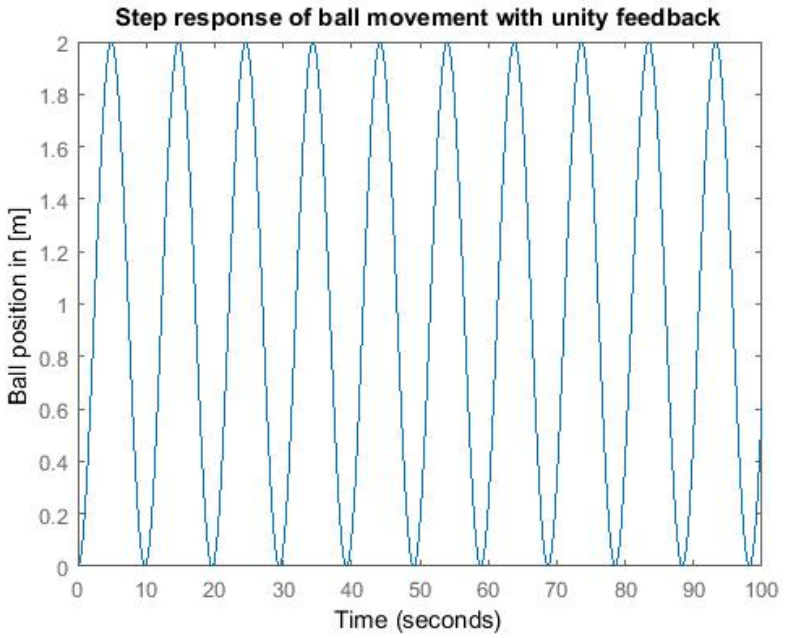


Figure 7.9: Step response of ball position with unity feedback

It shows that the system is only marginally stable and it will need a controller in order to deliver a usable result. Similar to the inner loop, the desired natural frequency and damping ratio of the system is calculated and using root locus, it will be attempted to create a P-controller that matches the specifications. The damping ratio is calculated in equation 7.8.

$$M_p = e^{-\pi \cdot \zeta / \sqrt{1-\zeta^2}} \tag{7.8a}$$

$$\ln 0.1 = -\pi \cdot \zeta / \sqrt{1-\zeta^2} \tag{7.8b}$$

$$\zeta = 0.5912 \tag{7.8c}$$

Then the undamped natural frequency can be calculated, as seen in equation 7.9.

$$e^{-\zeta \cdot W_n \cdot T_s} = 0.01 \tag{7.9a}$$

$$-0.5912 \cdot W_n \cdot 10 = \ln 0.01 \tag{7.9b}$$

$$W_n = 0.78 \frac{rad}{sec} \tag{7.9c}$$

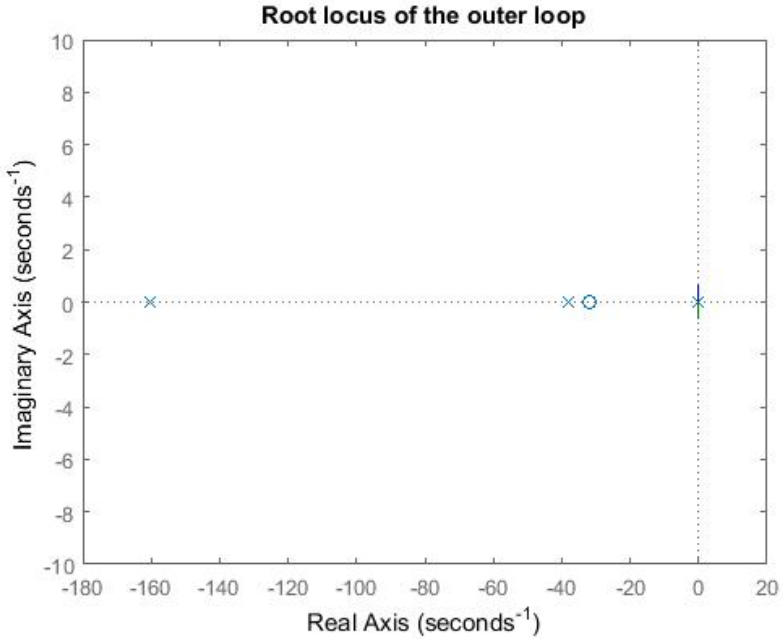


Figure 7.10: Root locus of the outer loop

Root locus of the whole outer loop is then drawn as seen in figure 7.10. By zooming in on the plot, we can see that it is not possible to move the poles at zero to a valid position on the negative side of the origin using only a P-controller since the highest valid location on the imaginary axis is 0.65 which is less than the minimum radius of 0.778. See figure 7.11.

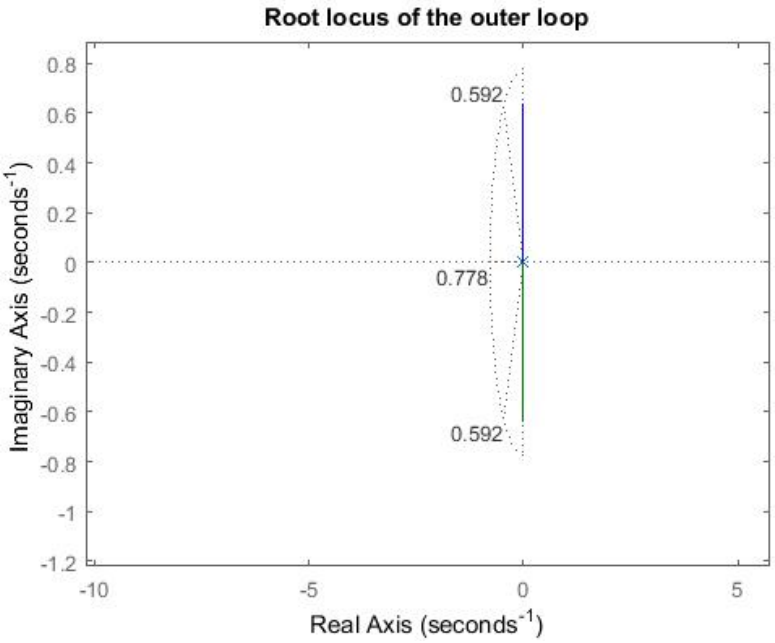


Figure 7.11: Root locus of the outer loop

Instead of attempting to implement a P-controller, another option is to add a lead compensator that can move the poles to the negative side, which results in making the system stable. To find the coefficients for the lead compensator, the phase margin is calculated as shown in equation 7.11.

$$P_m > \zeta \cdot 100 \tag{7.10a}$$
$$P_m > 0.778 \cdot 100 \tag{7.10b}$$
$$P_m > 77.8^\circ \tag{7.10c}$$

Using bode plots, the coefficients can be found by trial and error. It is possible to calculate how to change the phase, but since the amplitude will also change, the zero point will move and recalculations will have to be performed. In figure 7.12 it can be seen that the phase margin is -0.0384. The value indicates how far the phase margin is from -180 degrees, which produces the maximum noise and will effect the analog readings and make them inaccurate.

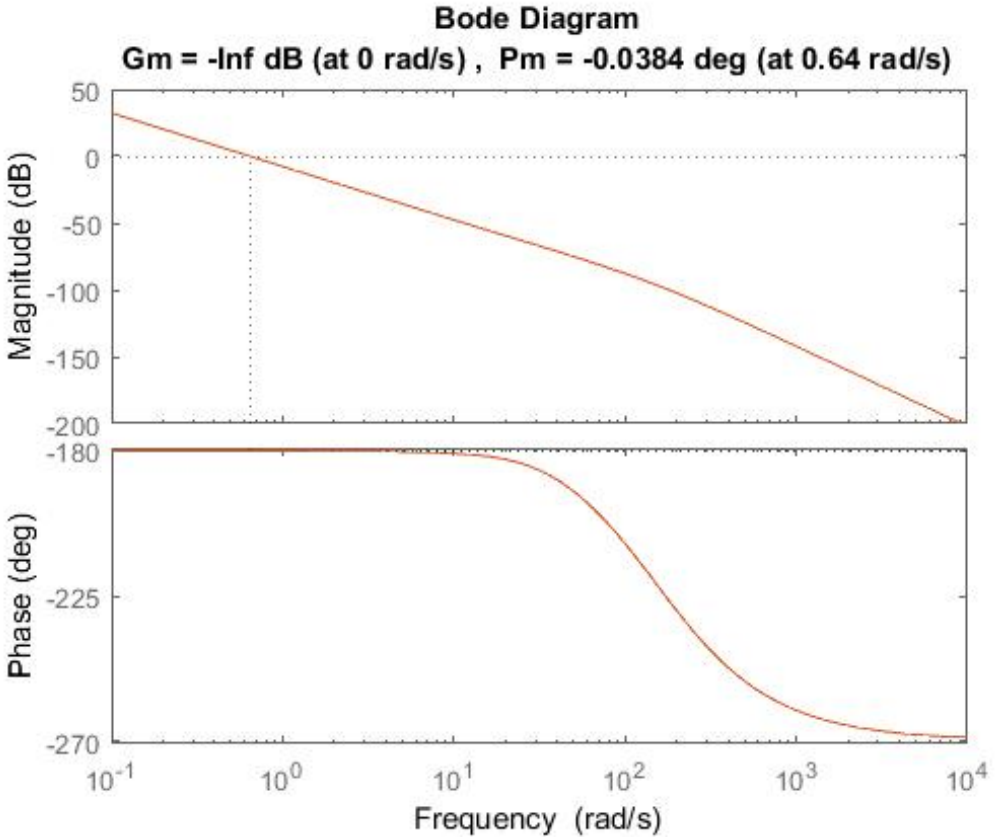


Figure 7.12: Bode plot of the system without a lead compensator

By adding a lead compensator the phase will increase up to a maximum of 90 degrees and therefore the phase margin will increase too. After some testing, the coefficient were found and the result can be seen in Figure 7.13 with the lead compensator:

$$\frac{s + 0.01}{s + 8.8} \tag{7.11a}$$

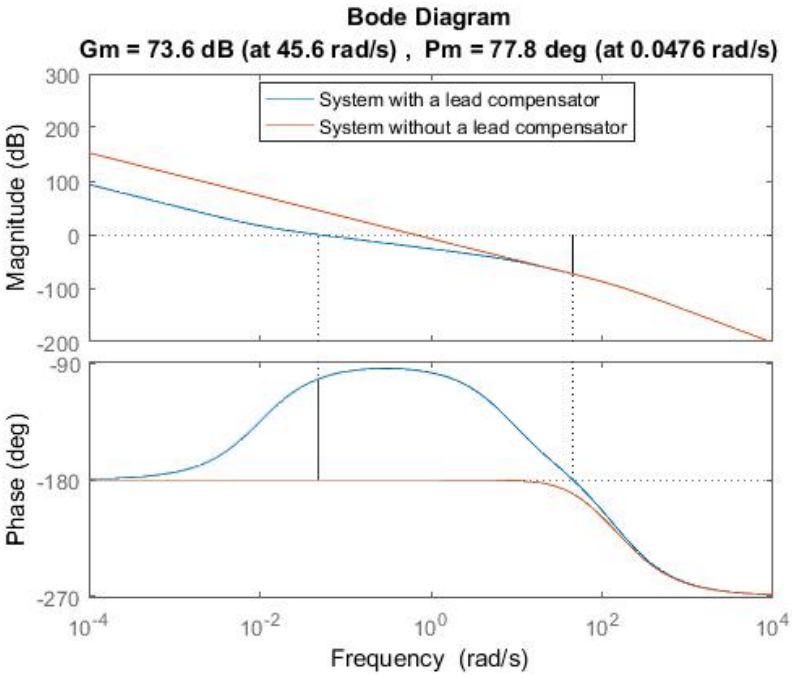


Figure 7.13: Bode plot of the system with a lead compensator

After adding the lead compensator, another root locus is drawn, in order to find a suitable gain value. This root locus can be seen in figure 7.14

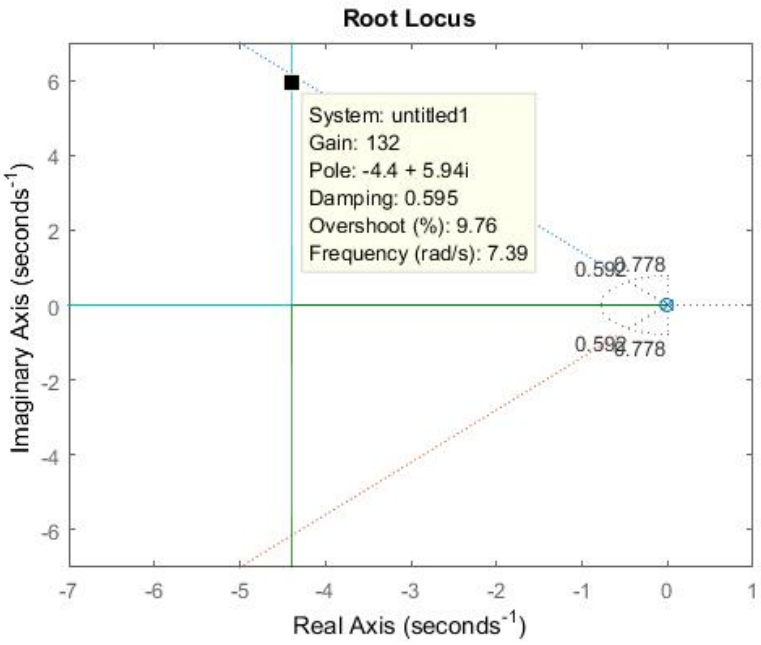


Figure 7.14: Root locus of transfer function with a lead compensator

If the pole at zero is moved to the marked point shown on figure 7.14, the system will be stable and have an acceptable overshoot at a suitable speed. A gain with a value of 132 was then added to the lead compensator, for the ball controller. The final model can be seen in figure 7.15.

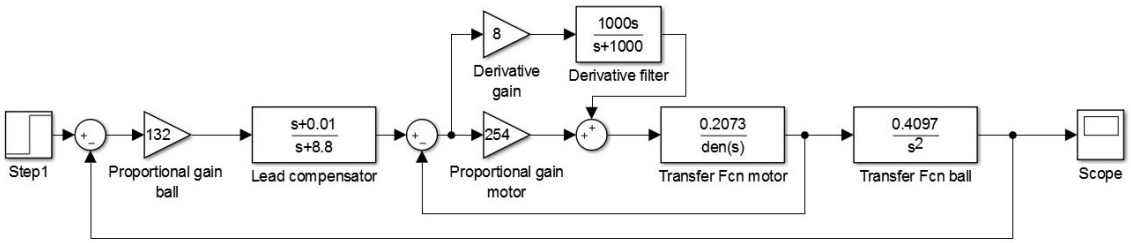


Figure 7.15: Simulink model of the system

The final result of the simulation had a steady state error equal 0.2%, an overshoot of 10% and a settling time of 1.1 seconds. All values are within the limits of the system performance specifications. See the simulation result in figure 7.16.

Figure 7.16: Result of simulation

Chapter 8

Controller Implementation

8.1 Initial Controller Implementation

During the initial implementation of the developed controller, there was some instability with the inner loop controller for the motor position, causing it not to work. Whenever the system was powered on, it would attempt to move towards the set point and then shortly after move in the opposite direction and stop due to the physical limitations of the potentiometer.

Figure 8.1: Position in radians from the potentiometer

Figure 8.1 shows how the inner loop attempts to reach and settle at the set point of 0 radians, where at this position the beam is level. The initial motor position is at negative 1.5 radians, and when powered on it is expected to move upwards towards the set point of 0 radians. When tested, it reaches and overshoots the set point by a small amount, the controller then rapidly shifts the motor direction back towards the negative direction, causing the system to stop due to its safety feature to protect the potentiometer.

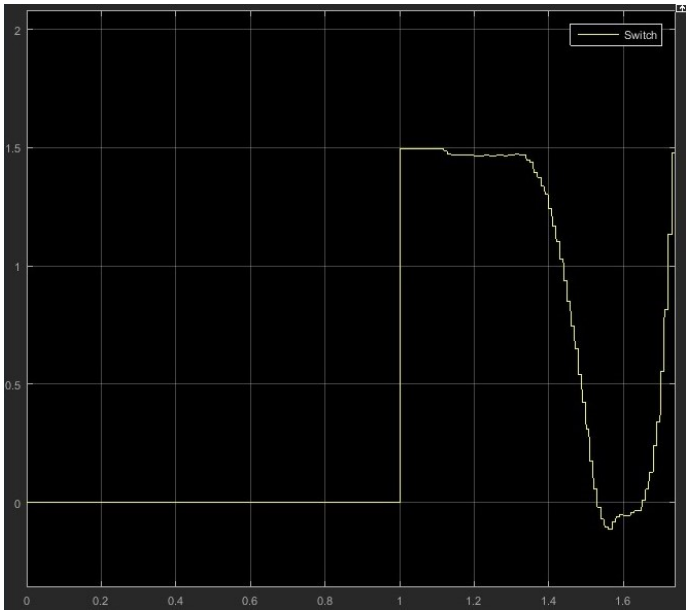


Figure 8.2: Error signal

After investigating the different outputs from the system, it was discovered that the error signal going into the controller had similarities to a step-like signal, this can be seen in figure 8.2. The derivative term of the developed PD controller handles these steps poorly, since taking derivatives of values with sudden changes creates a very large derivative output, known as derivative kicks. The step behaviour of the error signal originates from the noisy potentiometer readings. To decrease the effect of the derivative kick, the value of the derivative-term for the controller has to be decreased and filtered, in order to provide better results.

The theoretical values for the controller were P: 254 and D: 8 with a filter coefficient of 1000, these calculated values resulted in the system reacting too fast and would rapidly misbehave. The model would need some saturation, to provide some higher and lower limits for the voltage, since the unsaturated controller demands thousands of volts when turned on in order to react fast. This would result in damaging the motor and the other hardware the motor is connected to. In order to simulate the behaviour of the system with the current controller coefficients, a saturation was added to the output of the PD controller. Figure 8.3 shows the model behaviour after adding the saturation to the controller output.

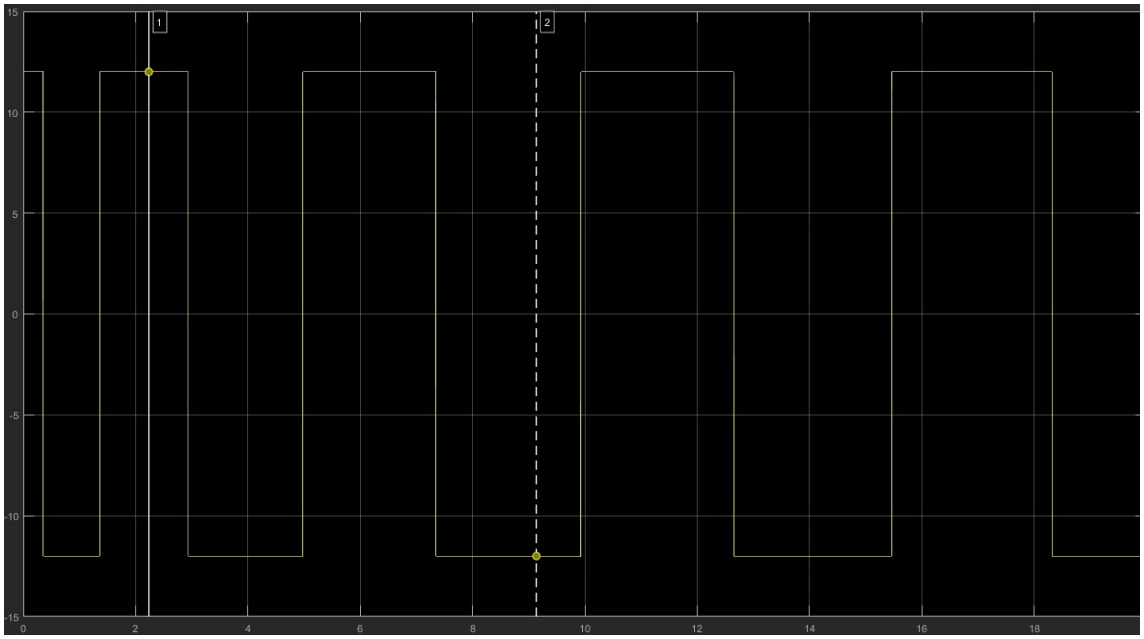


Figure 8.3: The voltage consumption over time.

It is clear that the system will never be stable and run smoothly using the calculated coefficients without taking the saturation limits into consideration. The system will either run forwards or backwards at the maximum possible speed, due to the high gain of 254.

Figure 8.4: Initial working step response of inner loop

Figure 8.4 shows the initial performance of the inner loop after decreasing the derivative term and manually tuning the P value, whilst also adding an Integral-term to reduce the stead-state error, which was introduced whilst lowering the other two terms. The values used in figure 8.4 are P: 10, I: 3.5, D: 0.5 and filter coefficients:

50. These coefficients resulted in a stable performance in the upward direction with an acceptable amount of steady-state error, since this can be corrected by the outer loop controller also.

8.1.1 Inner and Outer Loop combination

When attempting to combine the inner loop control with the outer loop, some issues were encountered in terms of sampling rate. Initially for the inner loop system a sampling rate of 1kHz was used, in order to gain a suitable performance in terms of the motor position. The inner loop utilizes a single analog input and a single analog output. Once combined with the outer loop further two analog inputs were added, which caused some issues in terms of missing ticks, this means that the Simulink model misses samples and lags in terms of real-time performance. This caused the outer loop implementation to be useless, since within a few seconds the maximum allowed missed ticks were exceeded and results in stopping the system.

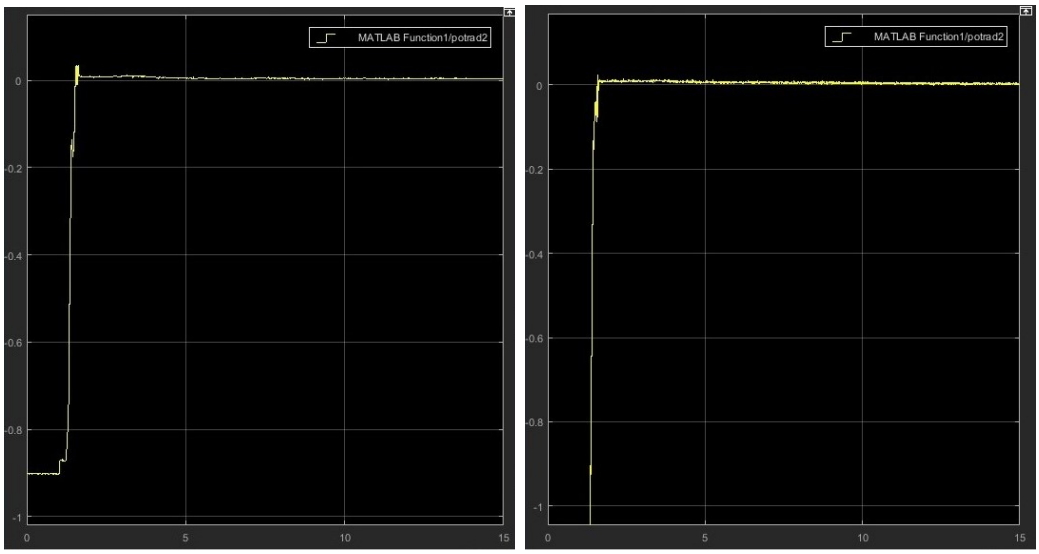


Figure 8.5: Step Response comparing sampling rates of 100Hz (Left) and 1kHz (Right)

In order to combat the issue of missing ticks, the sampling rate of the input connected to the potentiometer was reduced to 100Hz, whilst the sampling rate for the analog output controlling the motor also was reduced to 100Hz. As shown in figure 8.5, this only alters the performance slightly and creates a small overshoot in terms of moving the beam in an upwards direction to the level position.

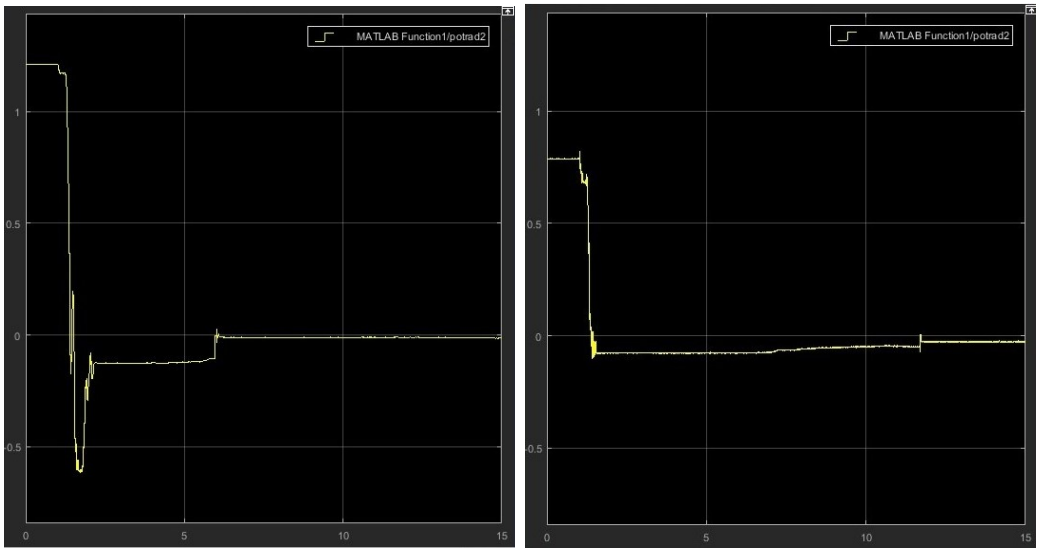


Figure 8.6: Downward Step Response comparing sampling rates of 100Hz (Left) and 1kHz (Right)

Moving from a position above the levelled position and downwards, is more noticeably influenced by the sampling rate changes, compared to the upward movement. The left plot in figure 8.6 shows how the overshoot is increased drastically, yet with a smaller amount of steady-state error, compared to the 1kHz sampling rate showed on the right plot. The downward motion is more prone to overshoot due to the added weight of the beam when moving downwards.

8.1.2 Motor Deadzone

Whilst implementing the inner and outer loop controllers, the power drill motor's performance was experienced in greater detail and the motor's deadzone was discovered. The implemented control solution would often take a long time to eliminate the steady-state error and in some cases, at slow intervals, oscillate around the set point.

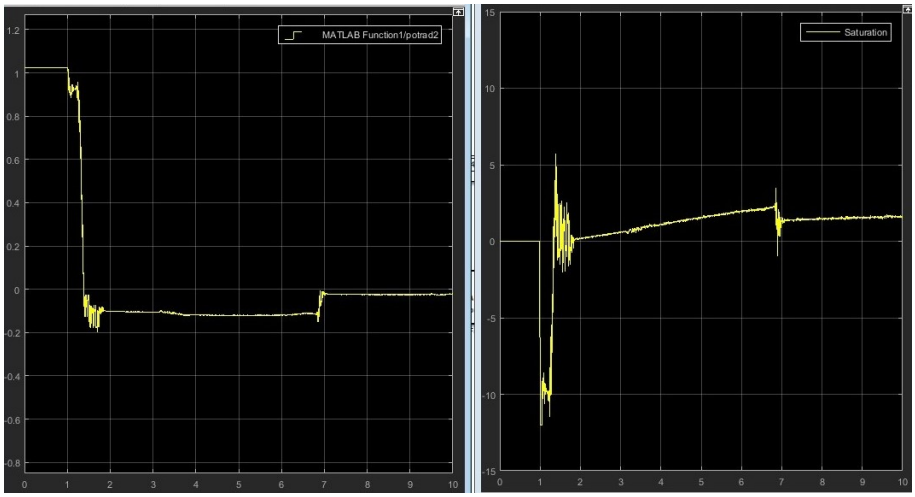


Figure 8.7: Deadzone of the power drill motor

The data from the motor position displayed how the power drill motor has a large deadzone. Figure 8.7 shows how the left plot is attempting to reach the set point of 0 radians and the right plot adjusts the input voltage accordingly. Since the motor has a deadzone of $\pm 2.8V$, the controller slowly over time attempts the build up a voltage high enough to move the motor closer to its desired position. Often the motor would overshoot its position by a little bit, due to the high RPM of the power drill motor, once it finally kicks in and begins to move.

8.2 Change of motorcontroller

The H-bridge that was built with the purpose of controlling the system worked well both in theory and under a low load. Once it was implemented to control the motors combined with the entire system, the controller experienced both current leakage and heat issues. The current leakage occurred due to the gates of the MOSFETs not being able to fully opened and closed by the given signals. Since the gates were not either in a completely closed or open state, a small amount of current was able to keep it partially open, leaving the MOSFETs turned on and allowed them to generate heat over time. After attempting to use the controller a couple of times the MOSFETs were getting damaged by the heat generated during the time they were not completely closed or open. Given more time, this issue would have been solved using some appropriate MOSFET drivers. The drivers would ensure that the voltages levels were either high or low enough to control the P- and N-channel MOSFETs accordingly.
In order to quickly progress with the control system implementation, it was decided to use a pre-built motor controller package. The chosen motor controller was the Sabertooth 2x5 motorcontroller [20], and to control the speed, the configuration of the Simulink model was changed from a PWM signal to an analog control signal.

8.3 Final controller implementation

8.3.1 Inner Loop

Due to the poor performance of the power drill motor, for the final implementation of the inner loop it was replaced by another motor. The motor replacing it was a E192-12-13 DC-motor with a planetary gearbox rated at 12V, with 300RPM and 450mNm of torque [21]. The main reasoning being that in order to achieve working control solution, the inner loop controller had to become more robust. This new motor has no noticeable backlash from the mounted gearing and it also does not have a deadzone as large as the power drill motor.

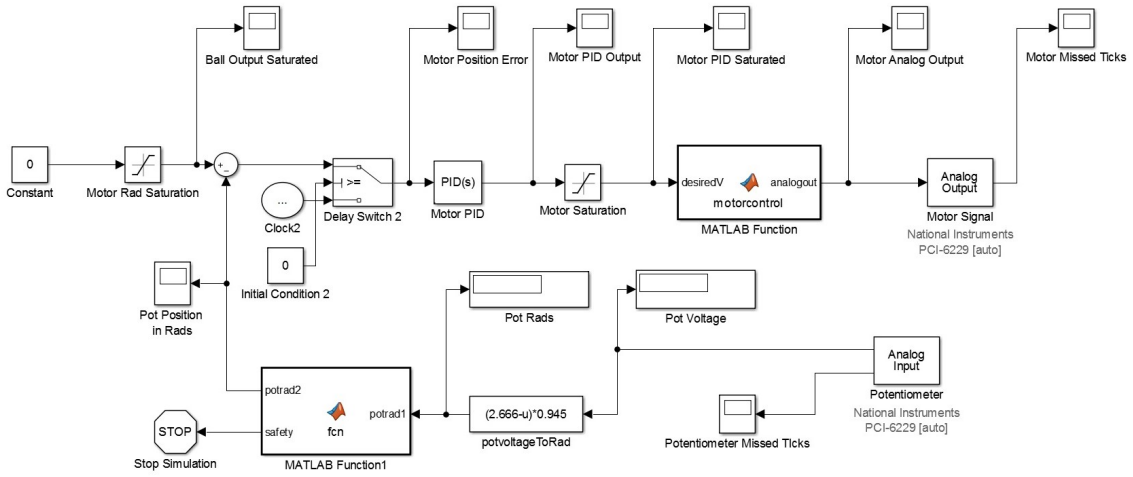


Figure 8.8: Simulink implementation of the inner loop

Figure 8.8 shows the final Simulink implementation of the inner loop. The analog output and input are both set to a sampling rate of 100Hz. The PID values for the inner loop control were then finally P: 5, I: 10, D: 0.75, with a filter coefficient of 50 for the derivative term. Unfortunately due to the potentiometer, the new motor's PID controller was also limited by the derivative term, since the performance of the potentiometer was not improved by a change of motor.

The inner loop takes in a desired angular position of the motor, which is saturated to ± 1.5 radians. During some further testing the safety feature for the potentiometer was raised from ± 1.5 radians to ± 2 radians, in order to allow a larger amount of overshoot during tuning. A 0.5 second delay is added to the controller in order to allow the National Instruments card to initialize and correctly configure the different I/O's before the control system begins.

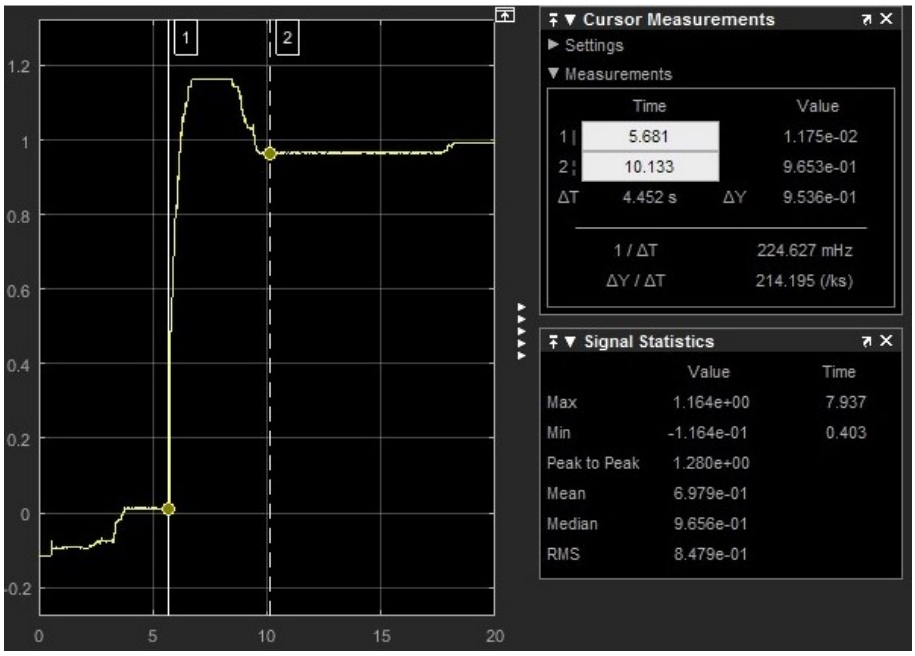


Figure 8.9: Step response of the inner loop

With the implementation of the new motor, the controller for the inner loop had to be re-tuned, in order to deliver suitable results. Figure 8.9 show the step response performance of the inner loop, which has 0.16 radians overshoot and a small amount of steady-state error.

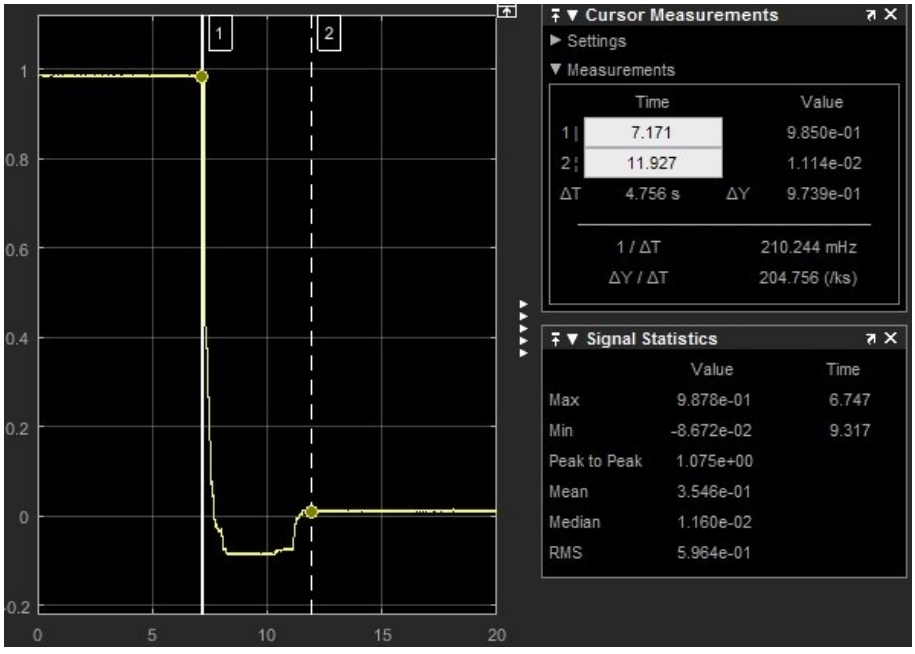


Figure 8.10: Downward Step response of the inner loop

Figure 8.10 shows the performance of the final inner loop controller in the downward direction.

8.3.2 Outer Loop

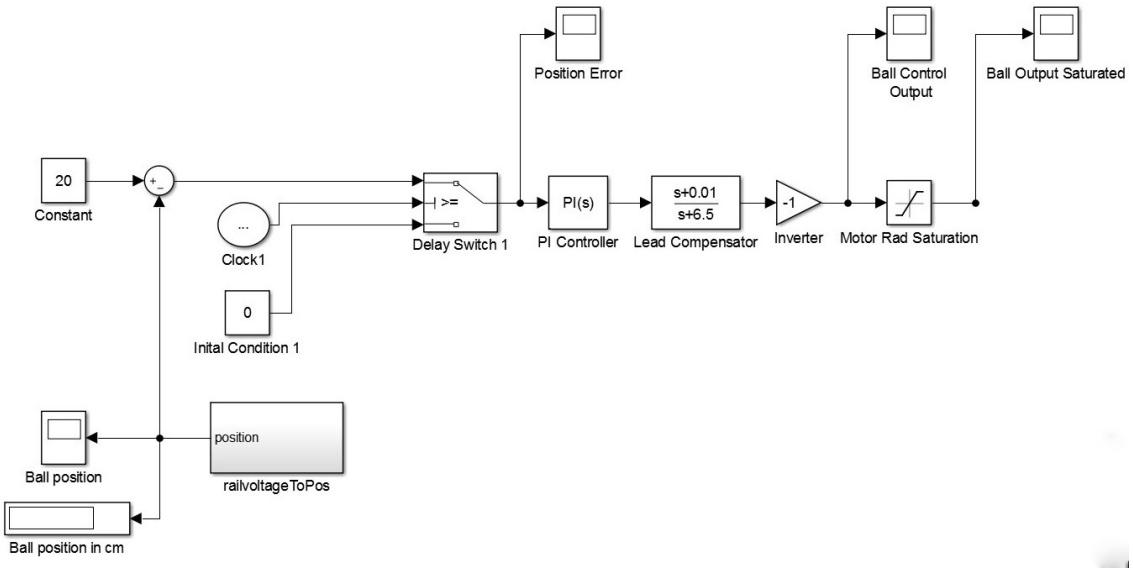


Figure 8.11: Simulink implementation of the outer loop

Figure 8.11 shows the implementation of the outer loop in Simulink. This model is connected to the constant input of the inner loop model, as shown in figure 8.8, where the complete Simulink representation can be found in appendix 11.3. Along side the lead compensator, a PI controller was added in order to eliminate the steady error of the system, since this was not handled entirely by the lead compensator in the final implementation.
Since the mathematical modelling was done so that ball position of 0cm was at the left end of the beam, the motor should lift the beam at the left side to make the motor angle and ball position proportional. However in the physical implementation the motor was mounted at the right end of the beam, which equalled to making the ball position and motor angle reverse proportional. When the motor lifted the beam from a levelled position, the ball position shifted from zero to negative values. To fix this, an inverter was added to the Simulink model. The proper ball position values were then retrieved with the 0cm mark at the end of the right side of the beam. Since the beam already had a tape measure glued on with a starting point to the left end of the beam, equation 8.1 was then added in our model to get the ball position with respect to the tape measure.

$$newballposition = 100 - ballposition \tag{8.1}$$

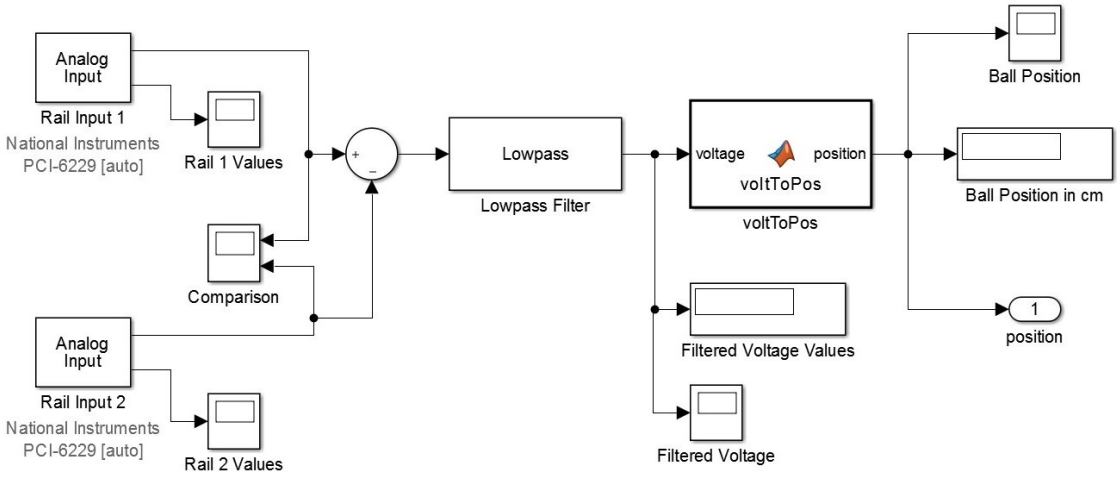


Figure 8.12: Subblock containing the rail position calculations

The contents of the railvoltageToPos block can seen in figure 8.12. This block contains the two analog inputs from each of the separate rails, where the differential is calculated in the sum block. The voltage readings are then passed through a low-pass filter in order to retrieve suitable voltages to calculate the current position.

Whilst reading the current ball position from the rails, it seemed a lot of noise had started to appear. This noise was only present once the new motor was powered on and was causing the low voltages of the wires coming from the rails to be distorted, this can be seen in figure 8.13. The noise initially caused the system to be uncontrollable, since the system experienced rapid and large positional changes within a short amount of time, due to the noise. The controller would then react by rapidly setting the motor angle to the maximum allowed positions, causing it to exceed the safety threshold and ultimately shutting off the system.

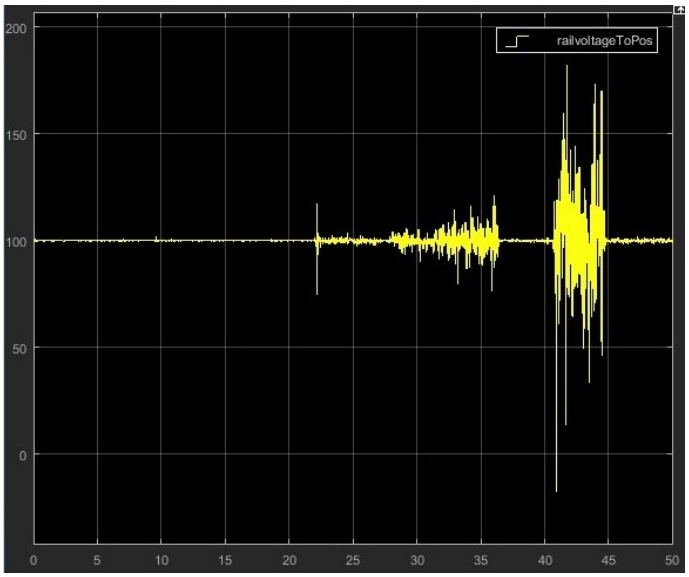


Figure 8.13: Ball position readings disturbed by the motor

The noise was caused due to Electromagnetic Interference (EMI) that can happen when wires containing a high current, in this instance power cables powering the motors with an analog signal, lies too close to or on top of cables containing low voltage signals. The disturbance was removed by re-routing the wiring. The cables for the motor were also shortened, in order to limit the amount of noise produced.

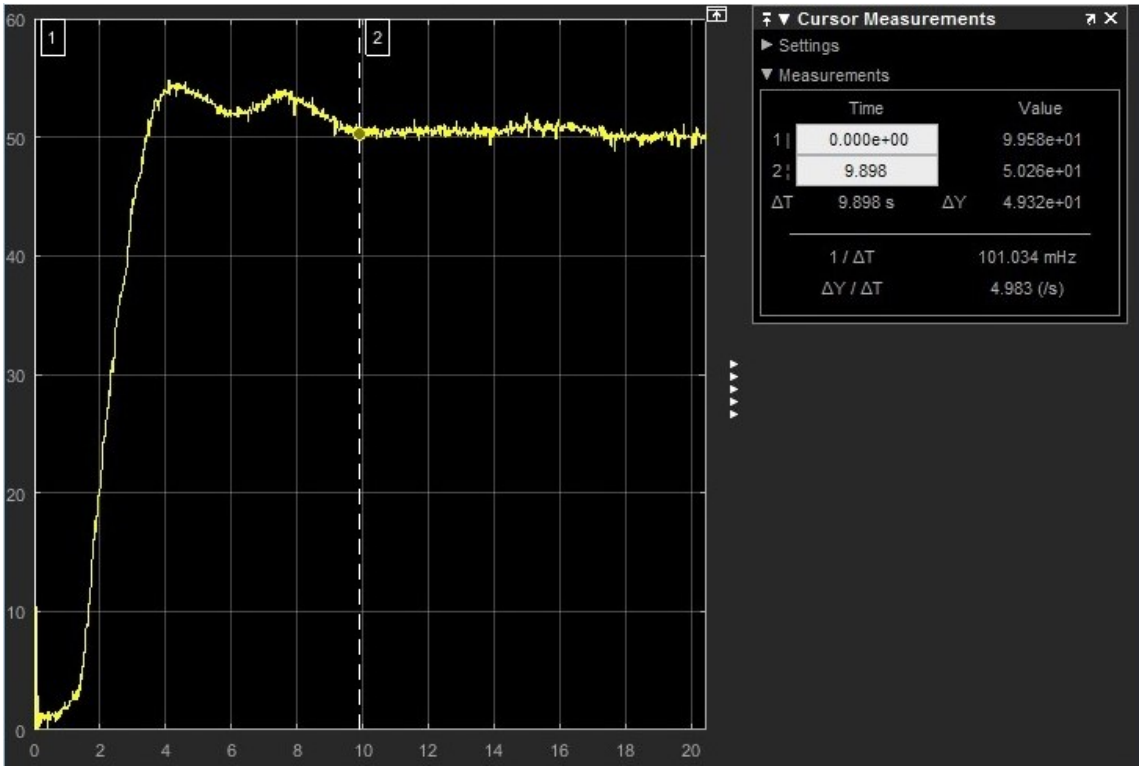


Figure 8.14: Ball movement from 0cm to 50cm (outerloop control implementation)

After manually tuning the outer loops lead compensator and PI-controller, the final performance can be seen on figure 8.14. The performance is based on a test, where the ball is positioned at the 0cm mark on the beam and then initiating the control system to move the ball to a set point of 50cm, this is to mimic a step response of the system. The ball location is recorded as the system attempts to stabilize the ball at the set point.
The current controller implementation has an overshoot of approximately 5cm and with a settling time of around 10 seconds. The final values for the PI-controller were P: 0.15 and I: 0.09. Due to some small and influential changes in the rail data, the lead compensator coefficients were limited in size due to the noise form the rails and therefore the overshoot of the outer loop could not be limited further. This is due to the lead compensator being similar to a PD-controller, where the derivative term is influenced more heavily by noise.

8.3.3 Setpoint changes and Disturbance handling

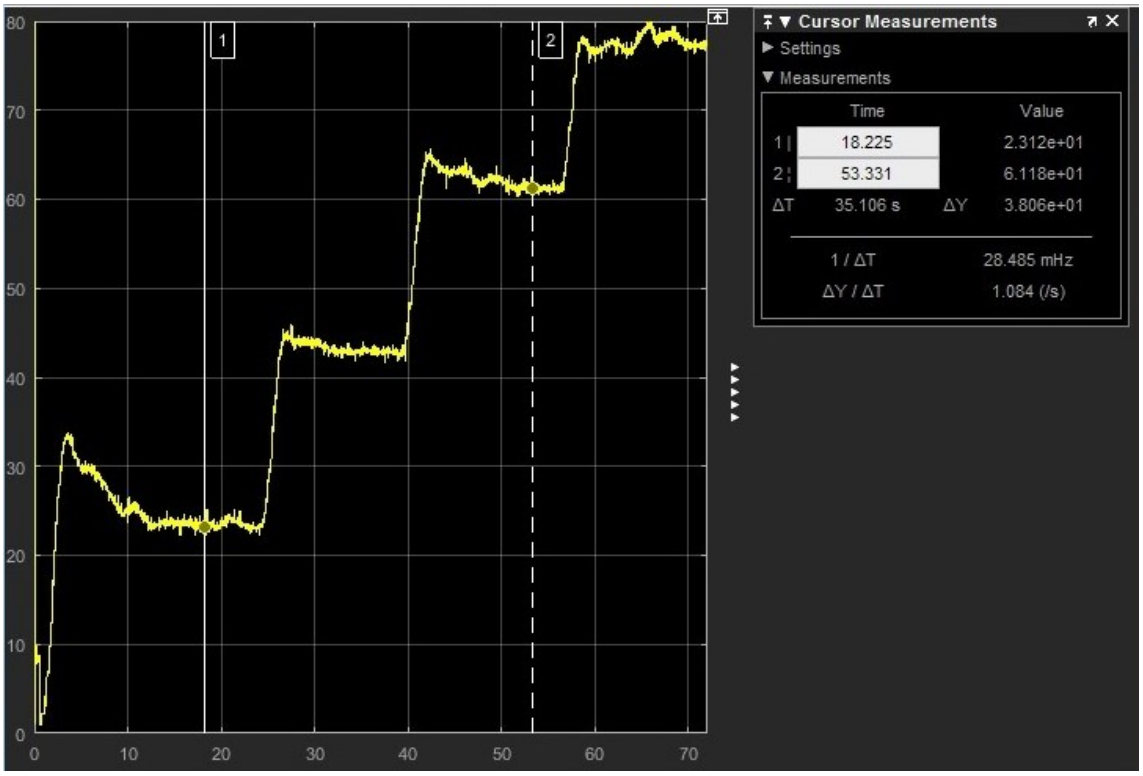


Figure 8.15: Setpoint changes in software and controller performance

In order to further test the robustness of the implemented controller, the set point was changed through software and updated real-time to track the performance of the system. Figure 8.15 shows the accuracy of the ball position, when at 10 seconds intervals the set point changes by 20cm. Initially at 20cm there is a considerable amount of overshoot due to the rapid starting movements of the controller, but it then settles at around 25cm, which is within a 5cm steady-state error of the 20cm set point. When the set point is relocated to 40cm, the overshoot is noticeably smaller and so is the steady-state error. This pattern continues also at 60cm and 80cm where the steady-state error less than previously at approximately 2cm.

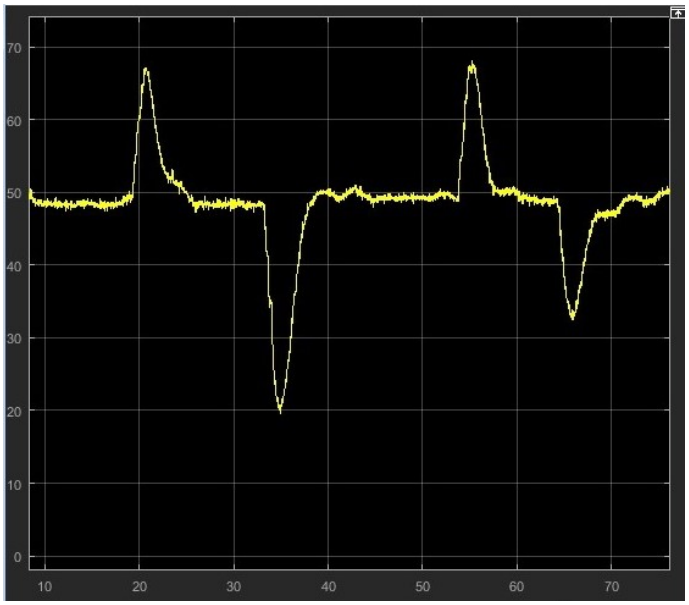


Figure 8.16: Manually causing disturbance

Figure 8.16 displays the performance of the system, when the ball is disturbed by pushing it away from its set point. The set point in this case is 50cm, since this is the center of the beam, allowing it to be disturbed equally in each direction.
When displaced, the system is able to return to the set point within 5 seconds or less, this time frame is also valid for quite large disturbances. As seen on figure 8.16 the system is able to deal with disturbance in both directions, in cases ±30cm.

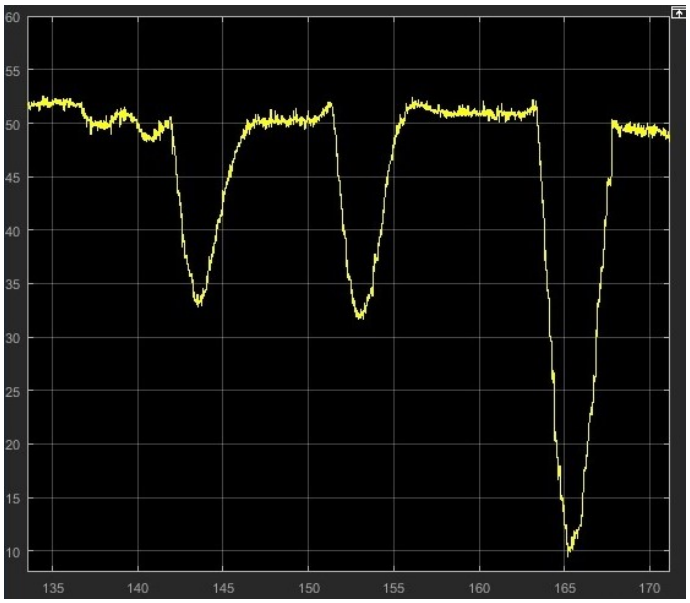


Figure 8.17: Manually causing disturbance

To further test the robustness, the system was disturbed to a greater extent in one of its directions. From the the setpoint of 50cm, it is able to be recover from a 40cm displacement within 5 seconds, this can be seen on figure 8.17.

Chapter 9

Discussion

During the initial implementation of the theoretical controller, preparations and efforts were being made in order to validate the theoretical controller based on its real life performance. Whilst implementing the control solution the performance of the drill motor was discovered to be a non-feasible actuator in terms of creating a well-functioning control solution, the motor was then replaced by an alternative motor with better specifications, which would aid in improving the control precision. Due to replacing the motor during the implementation phase of the project, the measured and calculated coefficients from the performed experiments used during the theoretical controller development no longer fits the new motor. Therefore, by performing the model validation would yield invalid results since the modelling done previously in the report is dependant on the coefficients for the drill motor, where the real life implementation uses the other motor with a different behaviour and set of coefficients. The drill motor was replaced during the late stages of the project and the focus was shifted towards creating a functioning control system. Therefore there was no available time to perform another set of experiments, in order to gain the new set of coefficients required for the new motor.

	P	I	D
Theoretical inner loop	254	0	8
Implemented inner loop (First motor)	10	3.5	0.5
Implemented inner loop (Second motor)	5	10	0.75

Table 9.1: Inner loop PID values

While developing the theoretical controller, the performance specifications were easily met after utilizing the root locus plot. Later when implementing the controller it was realised that the theoretical calculations were performed upon a non-saturated system. Further investigations showed that the PD-controller for the inner loop demanded voltage levels above 1000V, where the motor used was rated at 12V. This resulted in the theoretical system being over-saturated whilst attempting to implement it, causing it be uncontrollable. After reviewing the calculated values for the inner loop controller, it made sense that the implementation would react so

violently, with a proportional gain of 254, as seen in table 9.1. The values for the implemented controller were then manually tuned, starting with a slow system and then gradually speeding it up to approach the system requirements. Initially, when manually tuning the inner loop, a controller was created that had a fast reaction time, with small amounts of overshoot and also a small steady error. When combining the fast inner loop controller with the outer loop, some compatibility issues began to appear. Now, the inner loop was too fast, making it hard for the ball to remain on the beam whilst the rapid movements of the motor were taking place.

	Overshoot	Settling time	Steady-state error
Inner loop specifications	10%	0.1 seconds	1%
Theoretical inner loop	8.3%	0.084 seconds	0.3%
Implemented inner loop	16%	4.5 seconds	1.3%
Outer loop specifications	10%	10 seconds	1cm
Theoretical outer loop	10%	1.1 seconds	0.2cm
Implemented outer loop	5%	10 seconds	0.26cm

Table 9.2: Controller performance comparison

The inner loop controller was slowed down a considerable amount after performing manual tuning, resulting in the inner loop performance that can be seen in table 9.2. This performance resulted in more stable results in terms of the outer loop performance. Even though the inner loop performance does not match the specified performance requirement, the outer loop controller will compensate in order to meet the full system performance requirements.
The theoretical outer loop controller was also created without a saturated output from the controller. This once again resulted in some over-saturations issue and were solved by drastically slowing down the system and manually tuning it to meet the specified requirements. As shown in table 9.2, the performance of the outer loop meets the initial specified requirements for the outer loop.

Since the theoretical solutions were created based on another motor and also without saturated outputs, it justifies the large amount of manual tuning performed in order to create a working control system. When performing the manual tuning, the choice to start with a slow system and gradually speed it up was chosen in order to protect the hardware, to avoid damaging the physical setup by moving too fast and have an unstable controller. One of the major limiting factors in terms of the manual tuning, was how the noise from the various inputs from the system influenced the derivative part of our controllers. The coefficients for the PD-controller for the inner loop and the coefficients for the lead compensator used in the outer loop were limited due to the noisy measurements retrieved from the system. Due to the limitations on the derivative term, the overshoot of the system could only be limited to a certain degree, this also meant that the integral and proportional terms had to be scaled accordingly to avoid having oscillations around the set point.

Chapter 10

Conclusion

The overall goal of this project was to control the ball position on a beam using control theory, in this case consisting of an inner loop controller and an outer loop controller. We successfully created a control system within the desired performance specifications for the system, although the inner loop controller specifications were not achievable without causing instabilities. The inner loop performance did not matter in the end, as long as the outer loop performance matched the required performance. The theoretical controller was able to fulfil all of its performance specifications, but this solution was not acceptable once implemented due to the lack of voltage saturation whilst creating the theoretical controller. The speed of the theoretical solution was also too fast, which resulted in the ball losing contact with the beam and eventually falling off.
Because of the motor replacement during the late stages of the project, a realistic theoretical model could not be recreated within the available time frame. Therefore the final physical model's controllers were created by performing manual tuning on the system. We were able to successfully use a PID-controller for the inner loop, alongside a PI-controller together with a lead compensator for the outer loop in order to create a stable system, which could efficiently handle disturbances towards the ball and recover to the given set point.
Initially, we also built a functioning motor controller in order to control the motor speed and direction. During the implementation phase it stopped working correctly and due to the lack of time it was replaced with a pre-built motor controller.
As a final conclusion to the project, we succeeded in creating a functioning control system, even though improvements could have been made if there had been more time.

Bibliography

[1] Electronics Tutorials. Closed-loop systems. http://www.electronics-tutorials.ws/systems/closed-loop-system.html. (Accessed 02/04/16).

[2] Block diagram of a negative feedback control system. https://upload.wikimedia.org/wikipedia/commons/thumb/2/24/Feedback_loop_with_descriptions.svg/500px-Feedback_loop_with_descriptions.svg.png. (Accessed 02/04/16).

[3] Fred Thompson. Control systems. http://www.willamette.edu/~fthompso/MgmtCon/Control_Systems.html. (Accessed 02/04/16).

[4] Modular Circuits. H-bridges – the basics. http://modularcircuits.com/blog/wp-content/uploads/2011/10/image7.png. (Accessed 02/04/16).

[5] Lon Glazner. H bridge control of a motor – low side switch. http://blog.solutions-cubed.com/wp-content/uploads/2012/06/h-bridge.png. (Accessed 02/04/16).

[6] International Rectifier. IRF9530N HEXFET® Power MOSFET. http://www.irf.com/product-info/datasheets/data/irf9530n.pdf. (Accessed 24/04/16).

[7] International Rectifier. IRFZ44E HEXFET® Power MOSFET. http://www.irf.com/product-info/datasheets/data/irfz44e.pdf. (Accessed 24/04/16).

[8] Fairchild Semiconductor. BC546 / BC547 / BC548 / BC549 / BC550 NPN epitaxial silicon transistor. https://www.fairchildsemi.com/datasheets/BC/BC547.pdf. (Accessed 24/04/16).

[9] Gene F. Franklin, J. David Powell, Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Pearson Education, 7th edition, 2015. p.72-75.

[10] University of Michigan. Dc Motor Speed: System Modeling. http://ctms.engin.umich.edu/CTMS/Content/MotorSpeed/System/Modeling/figures/motor.png. (Accessed 02/03/16).

[11] Raymond A. Serway, John W. Jewett. *Physics for Scientist and Engineers with Modern Physics*. Pearson Education, 9th edition, 2014. p.302-303.

[12] National Instruments. Pci-6229 NI board. http://sine.ni.com/nips/cds/view/p/lang/da/nid/14136. Accessed 02/04/16.

[13] National Instruments. National instruments PCI manuals. http://www.ni.com/pdf/manuals/375204b.pdf. (Accessed 02/04/16).

[14] National Instruments. CB-68LP NI pinout board. http://digital.ni.com/public.nsf/websearch/5F6E0E5867D78C7D86256C4B007DB758/$FILE/CB-68LP.gif. (Accessed 02/04/16).

[15] Ivan Virgala. Friction effect analysis of a DC motor. http://pubs.sciepub.com/ajme/1/1/1/. (Accessed 15/05/16).

[16] Gene F. Franklin, J. David Powell, Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Pearson Education, 7th edition, 2015. p.145.

[17] Gene F. Franklin, J. David Powell, Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Pearson Education, 7th edition, 2015. p.216-226.

[18] Compensation in control system | lag lead compensation. http://www.electrical4u.com/compensation-in-control-system-lag-lead-compensation/. (Accessed 15/05/16).

[19] T. Aaron Gulliver. Compensation techniques. http://www.ece.uvic.ca/~agullive/trans/D_p1-20.pdf. (Accessed 15/05/16).

[20] Sabertooth 2x5. https://www.dimensionengineering.com/datasheets/Sabertooth2x5.pdf. (Accessed 18/05/16).

[21] Micromotors. E192 DC-motor with planetary gearbox. http://www.micromotors.eu/E192.pdf. (Accessed 15/05/16).

Chapter 11

Appendix

11.1 Appendix code

```
%
    ------------------------------------------------------------------------

% This is a setup for the National Instruments PCI-6229 board
% Work for P4 project in control theory in Aalborg university Esbjerg,
    2016
% Manual can be found here
% http://www.ni.com/pdf/manuals/371022k.pdf
%
    ------------------------------------------------------------------------


% First needed is to create a matlab session with the board
s = daq.createSession('ni');

% Next is needed to open a channel we use A0
ch = addAnalogInputChannel(s,'DEV1', 0, 'Voltage');

% When the channel is started it starts in the voltage difference
    setting
% We want to have it set to reference single ended
% To change that you write
ch.TerminalConfig = 'SingleEnded';

% To change the sample rate to 10.000 per secong, it starts default in
    5000, you write
s.Rate = 10000;

% To change the total sample time, it starts default in 1 s, you write
s.DurationInSeconds = 5;

% To change the range the board is measuring to -1V to 1V, the default
% is -10V and 10V you write
set(s.Channels,'Range',[-1 1])
% This will change all channels to that set range. You can also specify
```

```
% each channel range by specify them.
set(s.Channels(1,1),'Range',[-10 10])

% To remove a channel you type
removeChannel(s,'DEV1', 1, 'Voltage')

% To start a measurement you write
[BDPV1,time] = s.startForeground;
% There BDPV1 is the name of the sample, you can choose what ever name, and
% the second one specifies that you want to take the time down also.
```

Code Listing 11.1: Initializing the board

```
function [potrad2,safety] = fcn(potrad1)

potrad2 = potrad1;
if 2 < potrad1 || potrad1 < -2
    safety = 1;
else

    safety = 0;
end
```

Code Listing 11.2: Potentiometer to voltage

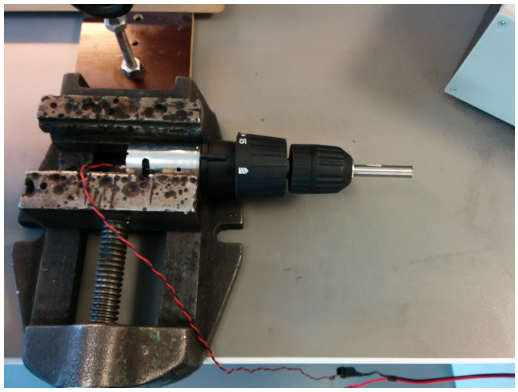
```
function position = voltToPos(voltage)
position = (voltage / (((-7.000*10^-4)-(-8.124*10^-2))/100))+100;
```

Code Listing 11.3: Rail voltage to position in cm

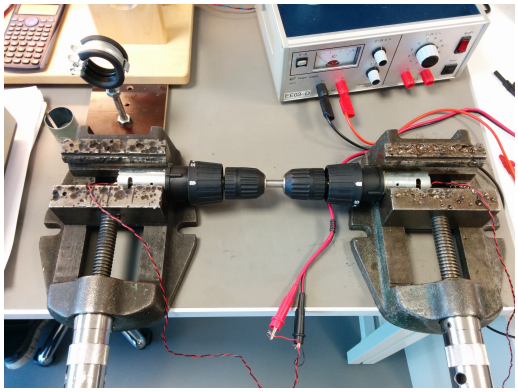
```
function analogout = motorcontrol(desiredV)
analogout = 2.5;
analogout = (2.5/12)*desiredV + 2.5;
```

Code Listing 11.4: Analog control to the motorcontroller

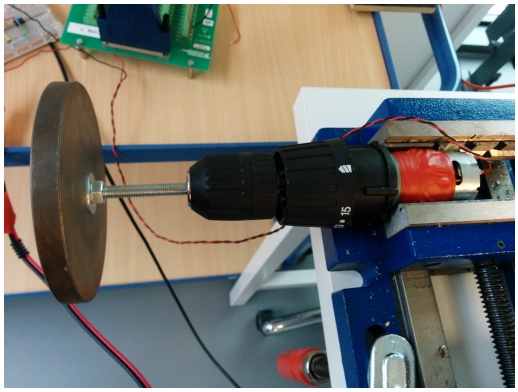
11.2 Experiment setup



(a) Free-spinning motor



(b) Mechanically connected motors



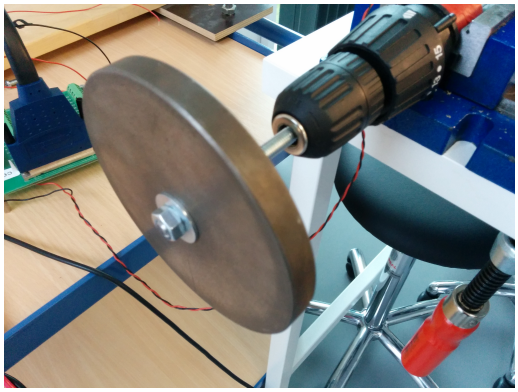
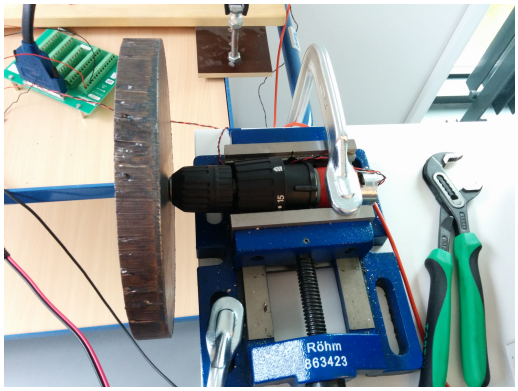


Figure 11.2: Initial small disk for inertia tests



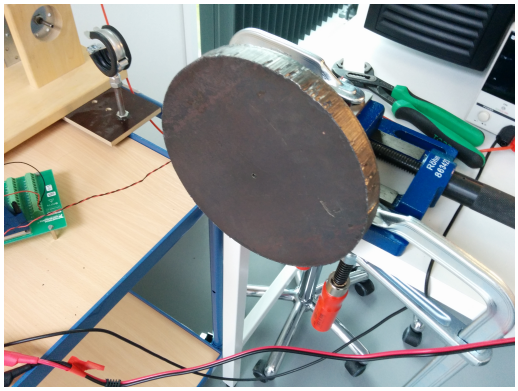


Figure 11.3: Second big disk for inertia tests

11.3 Full Simulink Implementation