Inverted Pendulum

Control Theory MT5104

Katrine Hansen
Per Henrik Hardeberg

University of Agder - 2023
Faculty of Engineering and Science

Contents

List of Figures

1 Revision Table

Report revision	**Date**	**Change**	**Signature**
A	02.12.2023	First publishment	PHH & KAH

2 Introduction and Objectives

The task given in this project is to get the inverted pendulum showed in figure 1 to swing up from an initial position and balance in an upright position. The setup is located in the mechatronics Lab at Mahindra University.

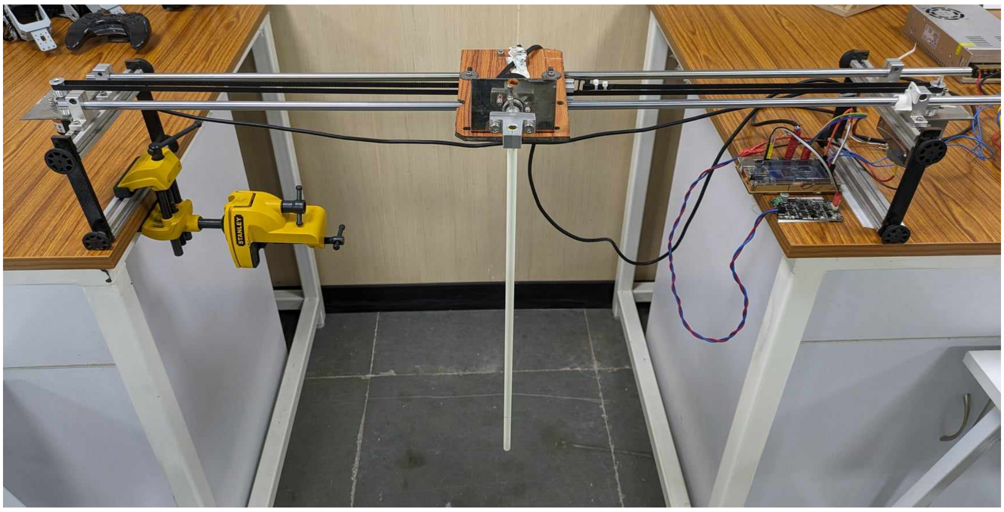
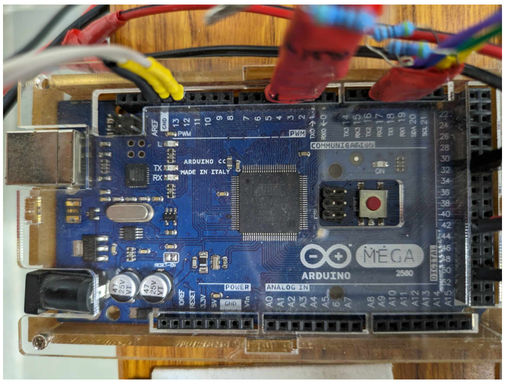


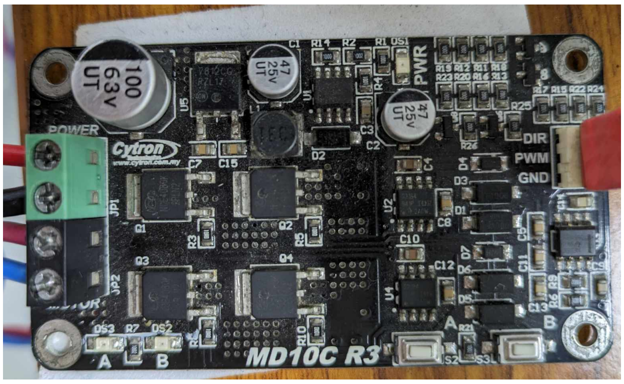
Figure 1: Pendulum Setup

This project is highly inspired by the inverted pendulum project of Jitendra Singh. A lot of inspiration and equations are found in his reports and models from his GitHub [1]. All equations mentioned in his projects will not be mentioned here, however some calculations are adapted for this physical system and calculated with the same equations. For a deeper understanding of energy based control and LQR control, it is advised to go through his project as well.

3 Hardware



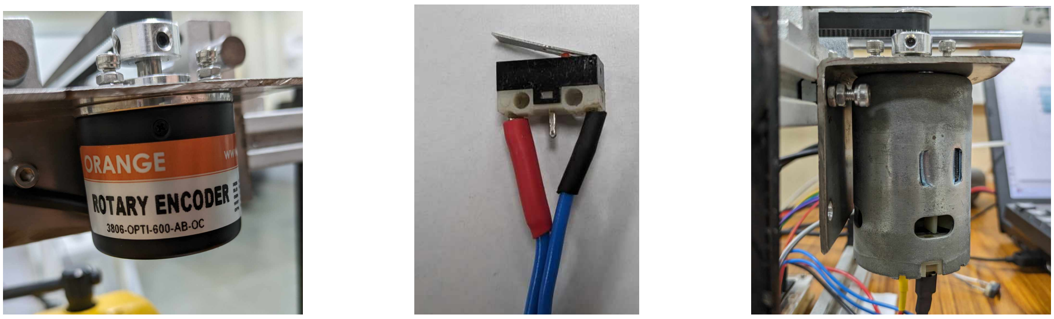
(a) Arduino Mega



(b) Motor Controller

Figure 2: Boards

The microcontroller used is the Arduino Mega, showed in figure 2a. The motor controller is a MD10C R3 controller, showed in figure 2b.



(a) Rotary Encoder (b) Enable (c) DC motor

Figure 3: Other Components

There are two encoders. Both are 3806-opti encoders. However, one of them are 1000 PPR and the other is 600 PPR. Both of them look the same, as showed in figure 3a, however, they are labeled with 1000 or 600 accordingly. The limit switch presented in figure 3b is used as a safety switch for the system. The use of this is explained more in detail in the demo chapter 4. The DC motor is presented in figure 3c and the power supply is presented in figure 4.

Figure 4: Power Supply

Arduino Pin	**Connected to**
12	Motor Driver pin: DIR
13	Motor Driver pin: PWM
GND	Motor Driver pin: GND
16	Angle Encoder: VCC
17	Angle Encoder: GND
18	Angle Encoder: Phase A (Green) Pull-up Resistor Soldered to cable
19	Angle Encoder: Phase B (White) Pull-up Resistor Soldered to cable
5	Position Encoder: VCC
4	Position Encoder: GND
3	Position Encoder: Phase A (Green) Pull-up Resistor Soldered to cable
2	Position Encoder: Phase B (White) Pull-up Resistor Soldered to cable
42	Enable Switch: Input
43	Enable Switch: Output

Table 1: Connection Table Arduino Pins

The connections from the Arduino to the different components are presented in table 1. The supply voltages for the encoders are connected to different digital pins. Then, each of them are enabled in the software. This is to make a cleaner connection without a breadboard, as well as be able to enable them individually if necessary.

The motor controller is also connected to the 12 V power supply, and to the DC motor.

4 Start Up for Demo: Work Instruction

Read all the steps in this section carefully before starting the simulink model.

4.1 Step 1

Download folder from github:

https://github.com/PerHenrikHardeberg/PendulumProjectMU_public

4.2 Step 3

Open simulink model, filename PendulumProjectMU.slx:

Figure 5: Simulink File

4.3 Step 4

Make sure to have the Arduino Support Package for simulink installed on your computer.

Connect the USB cable from the Arduino to your computer. If problems finding the Arduino, check the following:

- Modeling (top menu) » Model Settings (ctrl + E) » Hardware Implementation » Hoast-Board Connections » Host COM Port
- Set the correct step time: Modeling (top menu) » Model Settings (ctrl + E) » Solver (side menu). Set the solver to *Auto*, type to *fixed-step*, and step time to 0.01.

4.4 Step 5

Make sure POWER IS OFF ALL THE TIME DURING UPLOAD.

Figure 6: POWER SUPPLY OFF

4.5 Step 6

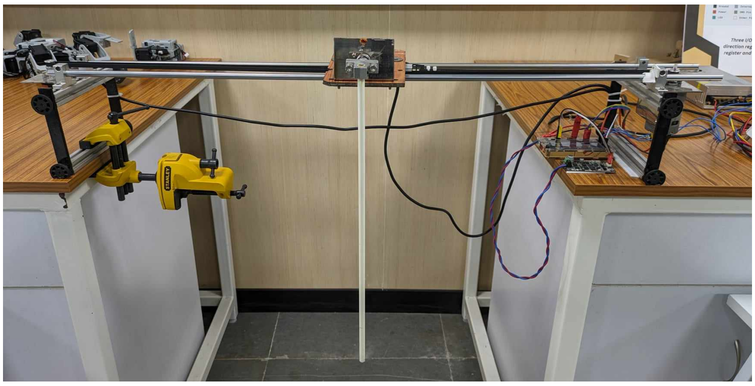


Figure 7: Pendulum in center

Make sure the pendulum is in the center of the work area, and the pendulum is hanging down as presented in figure 7. This is because the position x and θ is set to zero when the program is uploaded.

Be sure the power supply is turned off, like presented in the last step. Then the monitor and tune button is pressed to upload the program. The program will run on your computer and gets communicated to the microcontroller. Therefore, the computer must remain connected.

If there is a problem during upload, read the error messages. Also, make sure to check step 4 (Section 4.3).

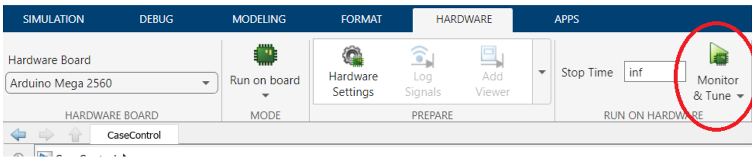


Figure 8: Monitor and Tune

4.6 Step 7

Make sure POWER IS OFF ALL THE TIME DURING UPLOAD.

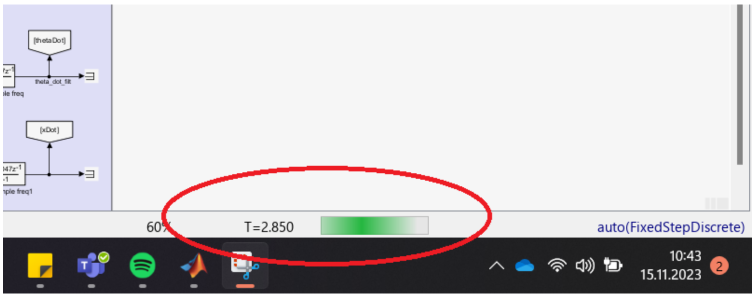


Figure 9: Program Running

After the program starts to run, the program running time will start to count seconds like presented in figure 9. Now, the power supply can be turned on, see figure 10.

Figure 10: POWER SUPPLY ON

4.7 Step 8

Press and hold the Enable button all the time to get the motor running. When the button is released, the motor PWM is set to 0 and the motor will stop immediately. This is a safety function.

4.8 Step 9

ALWAYS TURN SUPPLY OFF WHEN CHANGES ARE DONE IN THE MODEL BEFORE NEXT UPLOAD. The reason for this is that when the new script is uploaded, all Arduino pins go to high value, and then back to low. If the power supply is connected, the motor will run with maximum voltage and crash.

5 System Modeling

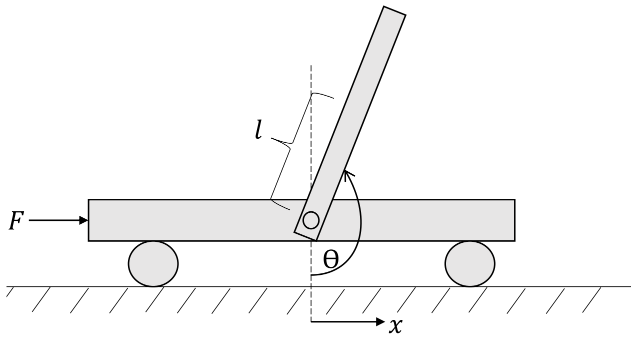


Figure 11: Free Body Diagram

The system can be modeled like a cart. The parameters from the model are used to calculate the gains for the different controllers. Because there was little to no information about the motor parameters, some parameters are roughly estimated. The most important observations from the system in figure 11 is that the angle $\theta = 0$ is where the pendulum is when the program is started. Also, the position $x = 0$ is where the cart is located when the program is started, and should be in the center of the work area of the cart.

Figure 12: Physical Plant

The input and output for the physical system is presented in figure 12. This is because the only input to the physical system is the motor voltage. This has a direct relation to the force applied to the cart. The only outputs from the system are the values from the two encoders. One attached to the belt, measuring the position of the cart x, and the other attached to the pendulum, measuring the angle θ. By differentiating x, $\dot{x}$ is found, and by differentiating θ, $\dot{\theta}$ is found. These values are used in the controllers.

6 LQR Control

The Linear Quadratic Controller (LQR) controller is used for this project because of the complexity of the system. First a proportional controller was tested, only controlling the force based on the angle error. That was not successful, two examples are presented on why it is not sufficient to control the pendulum without respect to the rest of the system.

6.1 P — controller example 1

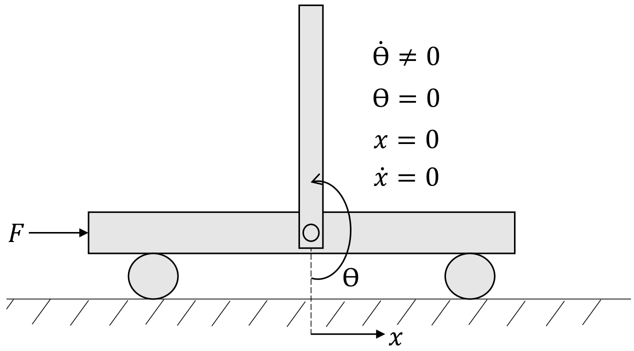


Figure 13: P controller example 2

Figure 13 shows the example of a situation where the pendulum is at the desired angle, and the P controller will remove the applied force. However, the angular velocity is not zero, and the pendulum will continue in the direction of velocity.

6.2 P — controller example 2

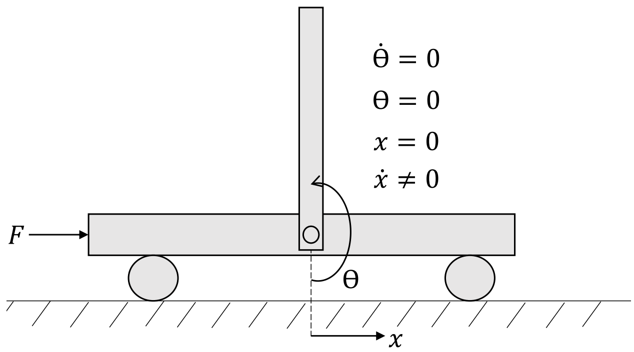


Figure 14: P controller example 1

Figure 14 shows a situation where the pendulum is in the desired position. However, the cart is not standing still. When the p-controller removes the force, due to no angle error, the cart, and thus also the pendulum, carry some kinetic energy. Therefore, when the force will be set to 0, the pendulum will continue in the direction the cart was moving towards, and thus it will never stabilize. This experiment is illustrated in 15.

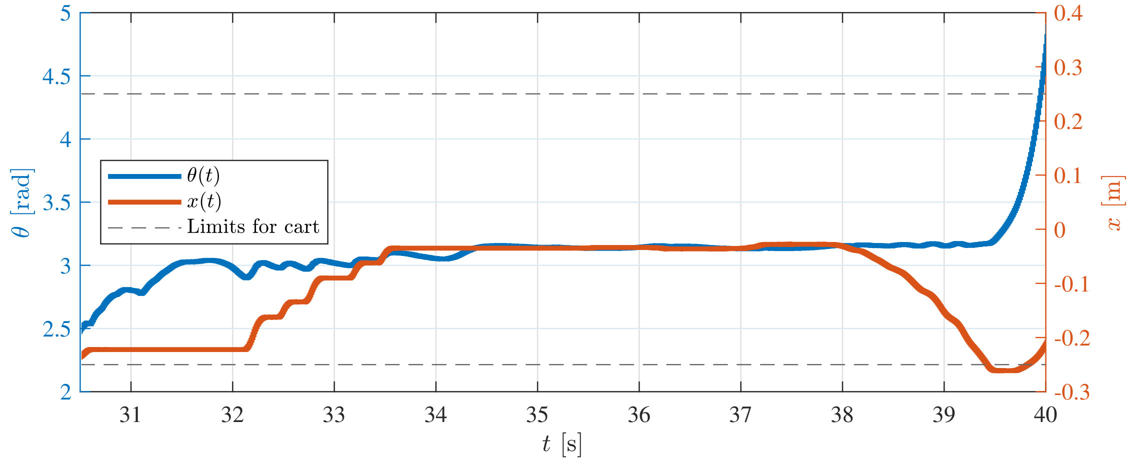


Figure 15: P-controller challenge

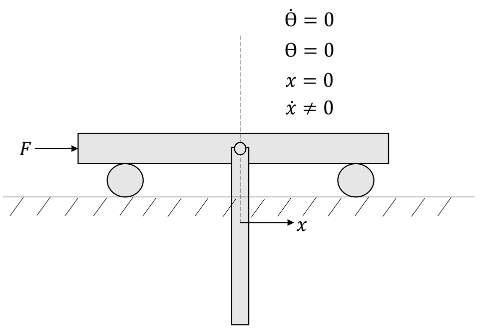
The mathematics to calculate the gains of the LQR controller was done based on the calculations used by Jitendra Singh [1]. In the next chapter, where the model is explained, observe that the gains in the LQR controller have opposite signs, like shown in equation 1

$$KK = [K_x, K_\theta, K_{\dot{x}}, K_{\dot{\theta}}] = [-20,\ 120,\ -20,\ 5] \tag{1}$$

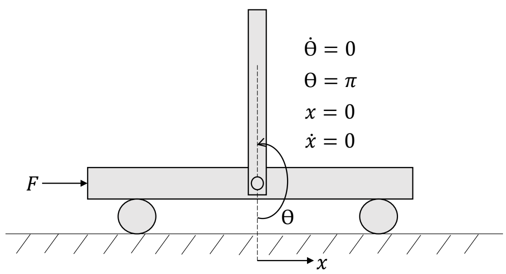
This is because when the angular velocity or linear velocity is present, the cart needs to go further to stop the momentum before balancing.

7 Swing up Control

For the project, an energy based swing up controller is used. The idea is the cart should start from rest with the initial conditions shown in figure 16a. Mathematically, an initial speed is necessary however, because of the resolution of the angle encoder, the velocity of the cart is never absolutely zero. The desired position is shown in figure 16b with an angle of 180 degrees.



(a) Initial position



(b) Desired position

Figure 16: Initial position and desired position of the pendulum for the energy swing up control

The potential energy of the pendulum is used for the energy controller. The concept of the controller is to add energy to the system if the actual potential energy is higher than the desired energy and remove energy if the actual potential energy is lesser than desired energy.

8 Simulink Program

The Simulink model consists of four main parts; the energy controller, the LQR controller, the control system and case controller that ensures the switching between controllers and, the real time data block that gets the pendulum angle and cart position from the encoders. Additionally, there is a separate block for encoder power supply. The whole system is shown in figure 17.

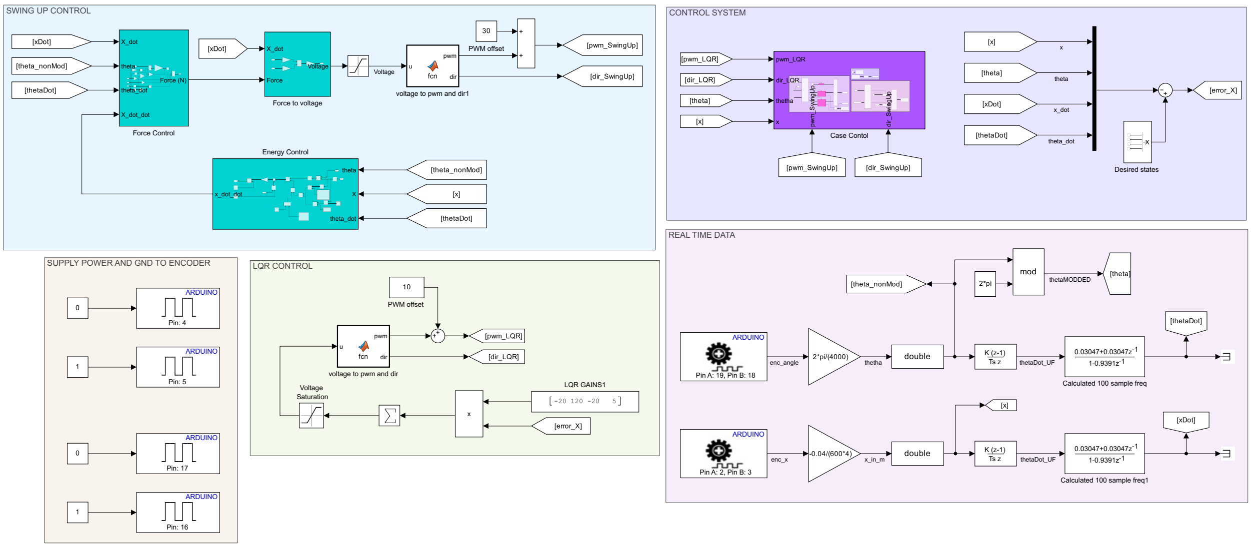


Figure 17: The main parts of the Simulink model

8.1 Energy Swing Up Controller

The swing up energy controller part consists of three separate blocks. The energy controller compares the desired potential energy with the actual potential energy and provides a desired acceleration. This acceleration, or desired control signal, is used to calculate a control force. This force is then transformed to voltage and given as input to the system. The blocks are shown in figure 18. The PWM offset is added to overcome the friction in of the cart. However, due to inconsistent friction, the offset may need changes in the future.

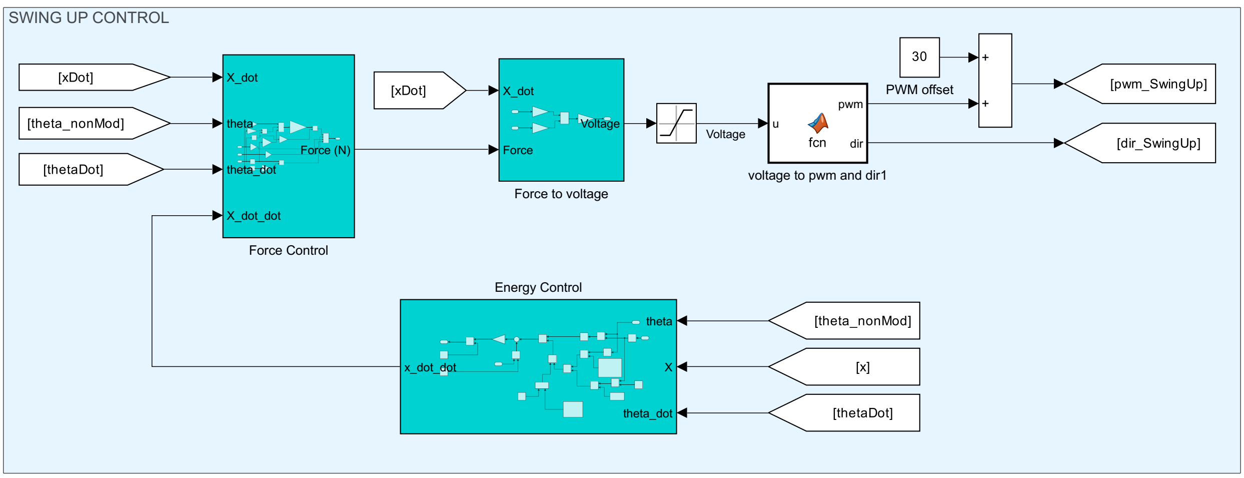


Figure 18: Energy swing up controller

8.2 LQR Controller

The LQR controller multiplies the gains calculated from the Matlab script with the error of the pendulum angle, angular velocity, cart position, and cart velocity accordingly. See figure 19. The summation of this is directly translated to voltage and provided to the system as PWM. The PWM offset constant may need to change, as explained for the swing up control in section 8.1

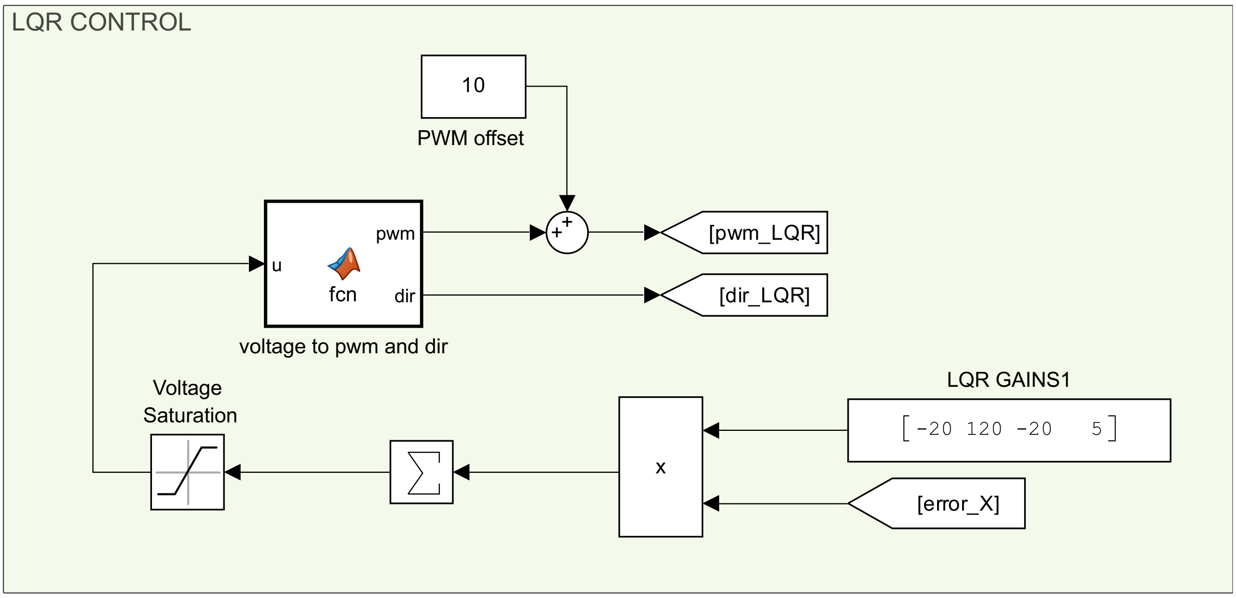


Figure 19: LQR controller

8.3 Real Time Data

This part of the Simulink program gets feedback data from the system or, the two encoders. The pulses from the encoders are transformed to angle in radians and distance in meters respectfully, see figure 20. To get the angular velocity and the cart velocity, differentiation of the signals is necessary. A filter is added to the derivatives to remove unnecessary peaks or, noise.

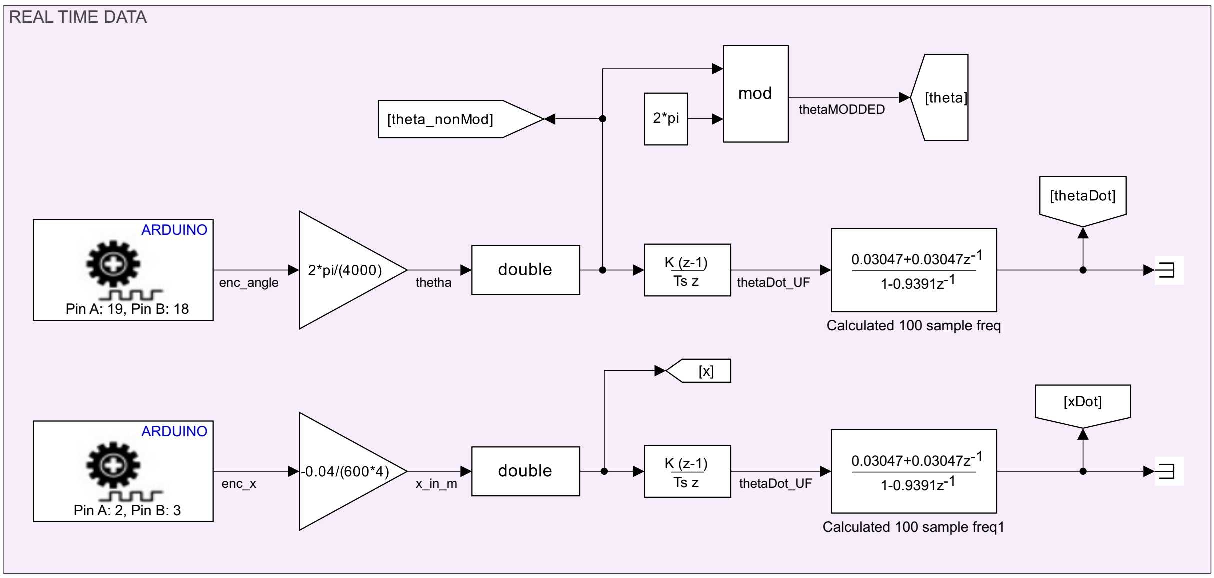


Figure 20: Real time data

8.4 Control System - Case Controller

Finally, the control system part, or the case controller part is responsible for choosing the correct control system, calculating the errors and also, some safety features. The system is shown in figure 21.

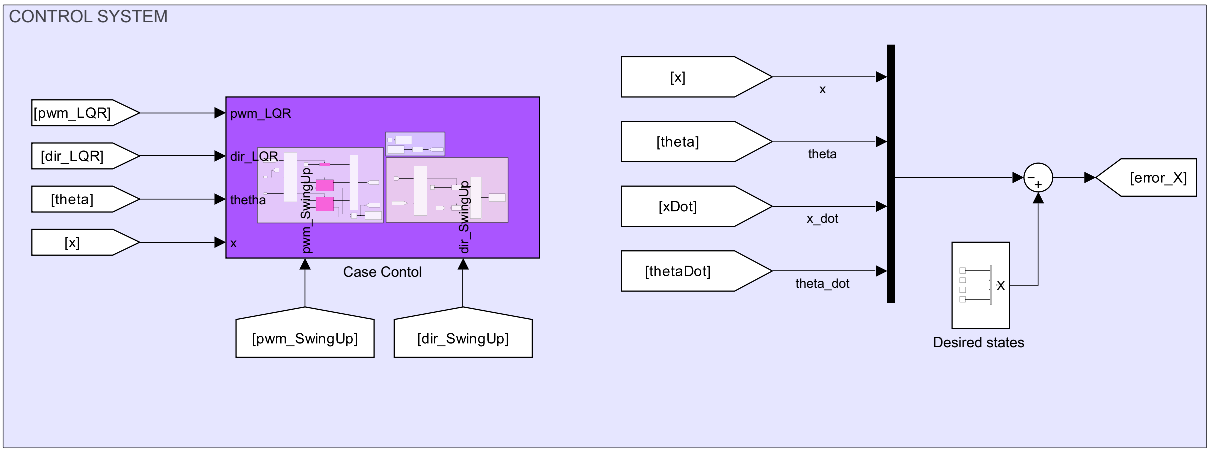


Figure 21: Control system - case controller

8.4.1 Choose Controller And Limit Switch Safety

Inside the case control block in the control system part there are different parts with different purposes. The first one decides which controller to use depending on the angle of the pendulum, see figure 22. Also, there is a safety function added that sets the pwm to zero if a limit switch is not pressed continuously.

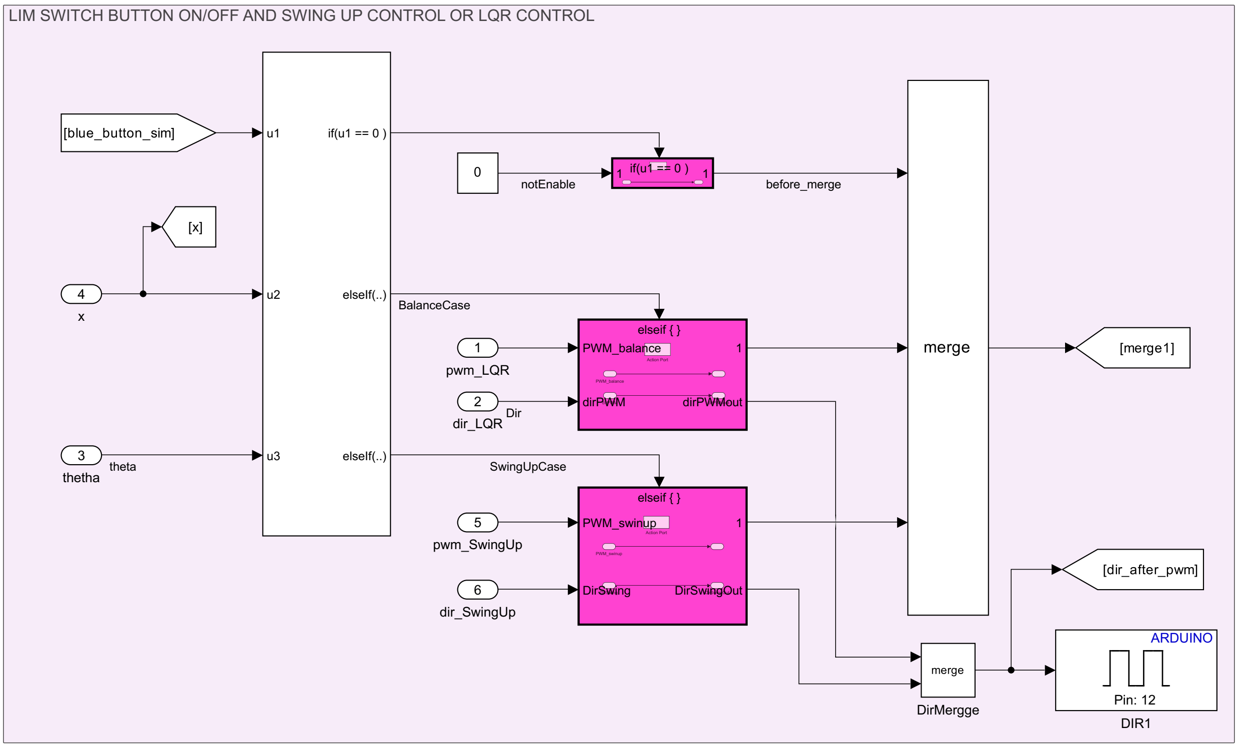


Figure 22: Controller chooser and limit switch safety

8.4.2 Crash Protection

One other part is responsible for the crash protection. In short, it will stop the motor if it is within some threshold and the velocity is going towards the end stop. However, if the cart is within the maximum allowable threshold position wise, but the velocity is in the opposite direction, the cart is allowed to move. Since, it will move away from the end-stop. The crash protection part is shown in figure 23.

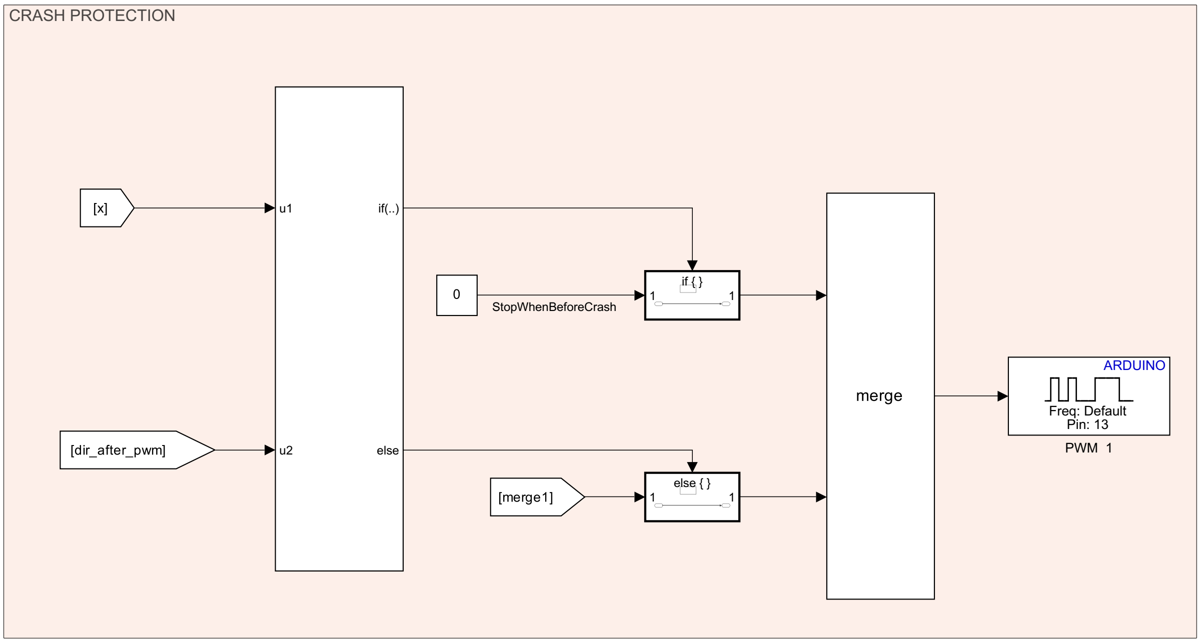


Figure 23: Crash protection

9 Outcome and Conclusion

9.1 Results

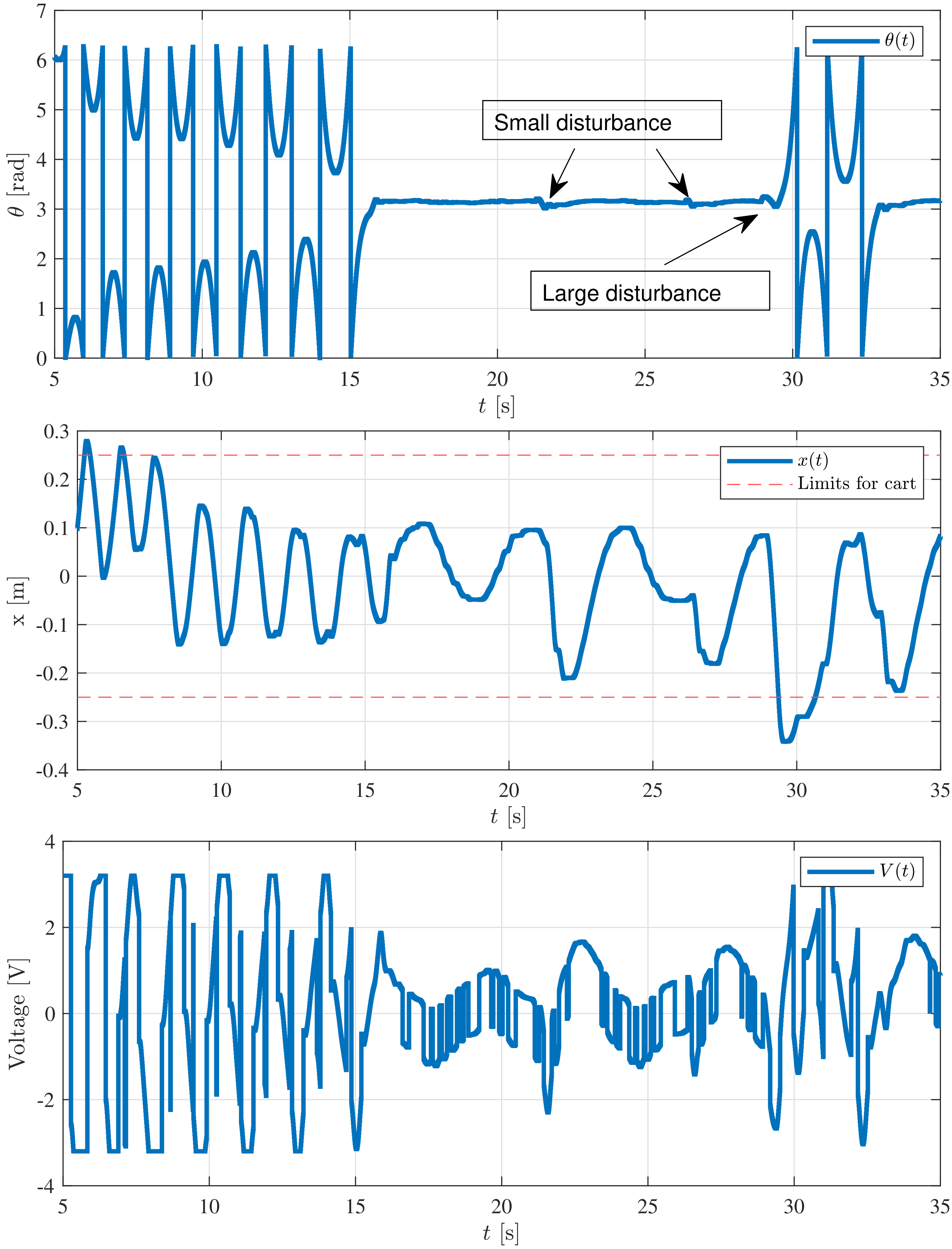


Figure 24: Angle, Cart Position and Voltage Swing Up and Balance

Figure 24 shows the cart position x, angle θ, and voltage V. Observe the time it takes for the swing up, as well as how the cart reacts to the disturbance. The voltage is saturated at around 3 V.

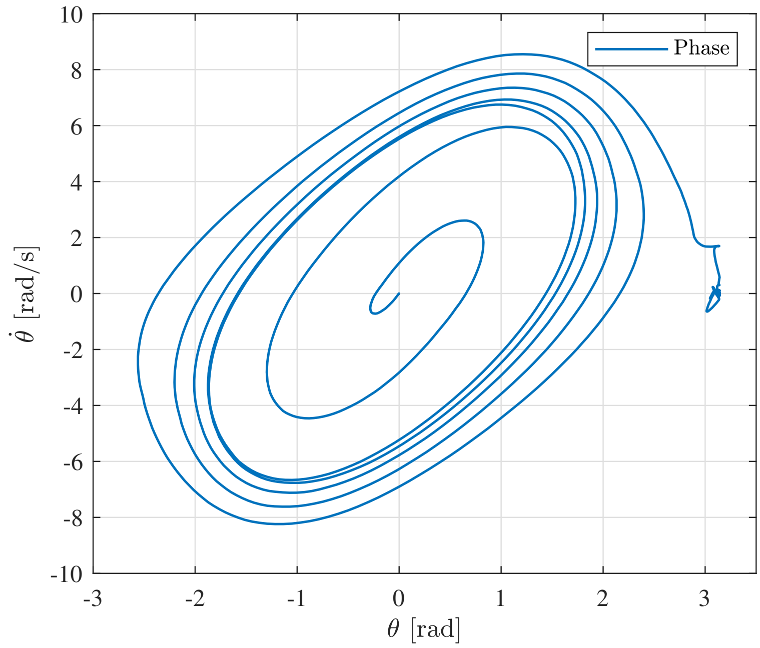


Figure 25: Phase Plot

Figure 25 is the phase plot illustrating the energy in the pendulum, by plotting the angle θ and the angular velocity $\dot{\theta}$. These are the two components of energy in the energy equation of the pendulum. Observe that from the third swing of the pendulum, the energy is not rising for some swings, before it rises again. This is because the cart is returning to oscillate around x = 0. This can also be seen in figure 24.

9.2 Conclusion

The hardware setup was partly built, but the students finished the setup and, created a Simulink model for controlling the system. Two control systems was designed separately, a Linear Quadratic Controller (LQR) for the balancing part, and an energy based controller for the swing up of the pendulum. Additionally, other features was added and programmed such as a safety limit switch and crash protection.

10 Challenges and Further Work

10.1 Hardware

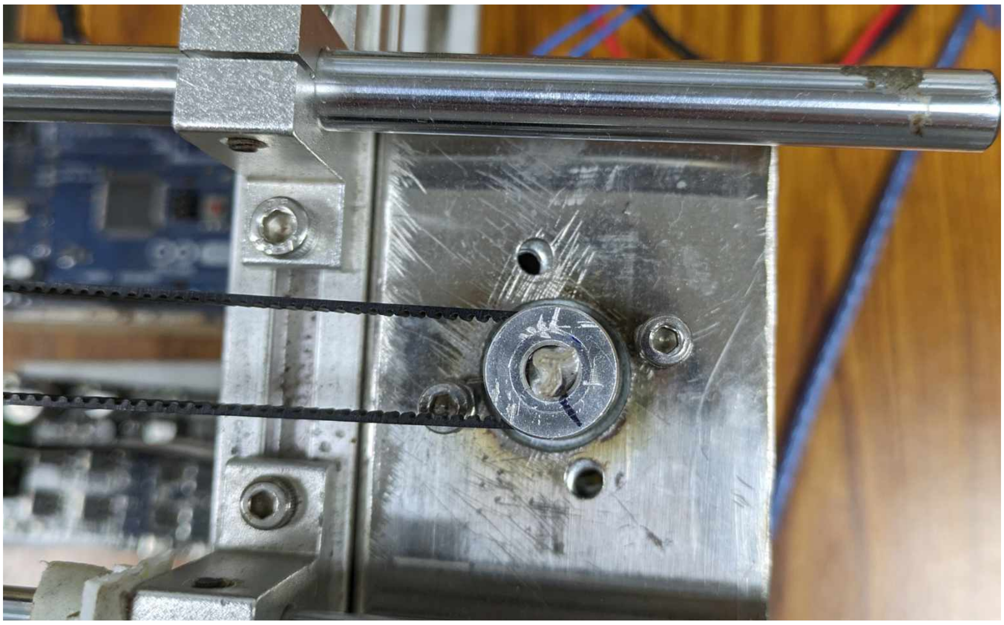


Figure 26: Pulley on Motor

The pulley shown in figure 26 is too wide big for the axle of the motor. This results in a nonlinear torque for the motor, depending on if the motor needs to tight the belt, as well as move the cart.

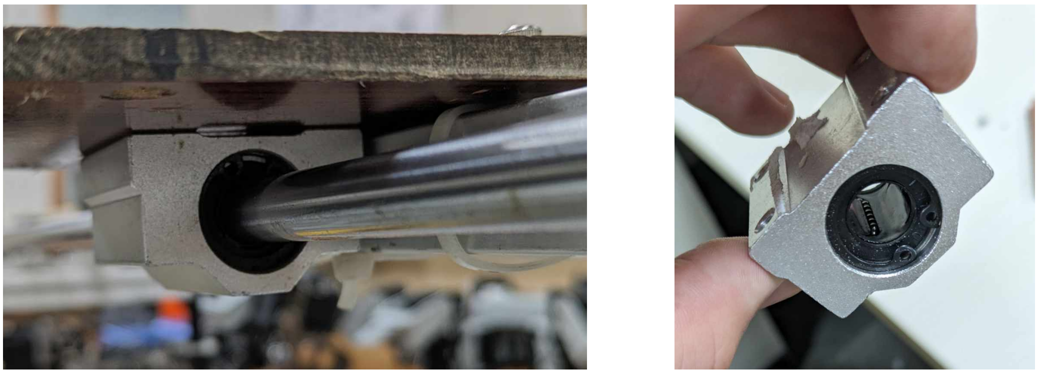


Figure 27: Friction Challenge

Figure 27 shows one of two bearings carrying the cart. These bearing, together with beams, have rust and dust inside the bearing. The axle was lubricated with some intervals during the project. However, that resulted in nonlinear friction for the cart.

10.2 Software

Many parameters, such as the friction mentioned above, motor parameters used to calculate gains for the different controllers, are not accurate. However, there could be other methods than LQR and Energy Based control that could give better results such as, a more robust balance and a decreased swing up time.

Bibliography

[1] Jitendra Singh. *jitendra825*. URL: https://github.com/jitendra825. (accessed: 15.11.2023).