Implementation 9

This chapter describes some core aspects regarding the implementation of the data filtering and control. In **section 9.1**, this is considered for the Simulink-model, while **section 9.2** focuses on the Arduino-implementation and verification of the EKF.

9.1 Simulink

This section describes implementation of the EKF and developed control laws in Simulink. The content of the block *MCU loop* from **figure 4.12**, page **22**, is shown in **figure 9.1**.

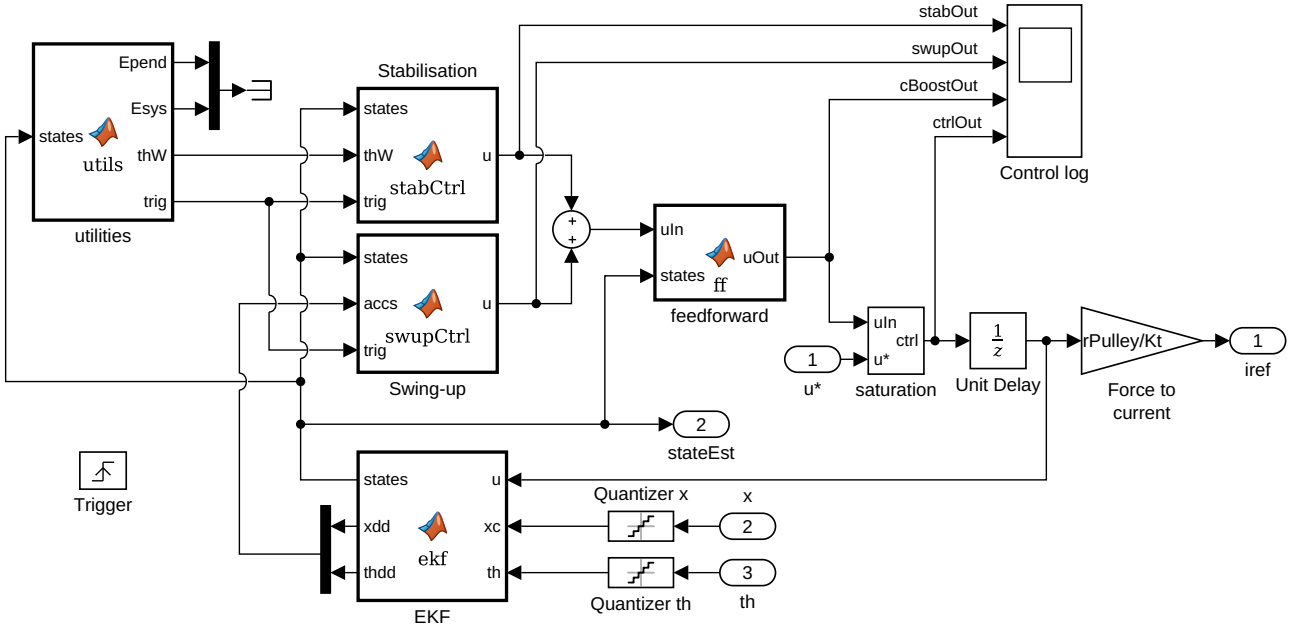


Figure 9.1: *Simulink model file swingUpModel.slx, MCU loop subsystem.*

As mentioned in **chapter 6**, only one controller is active at a time; either the swing-up or the stabilising controller. This is important to emphasise since their outputs in **figure 9.1** are summed before entering the friction feedforward block, *feedforward*.

The two controller blocks, *Stabilisation* and *Swing-up*, only perform a simple action which is a call to an `.m`-file which calculates the desired control signal. With this structure, changes can be made in the control calculations without changing the Simulink model, which is useful when multiple people develop using e.g. git version control and need to make changes in the same file. These are the only blocks with this configuration.

The *EKF* block is self-explanatory; it receives the quantised measurements of the cart position and pendulum angle, and outputs the state estimates along with the model-based accelerations.

The block *utilities* computes the energies in the system, and the catch trigger algorithm, thus forward-

ing a signal, *trig*, which activates either the swing-up or stabilising controller.

Another important feature within the utilities block is the calculation of a wrapped pendulum angle, θ_w, used by the stabilising controllers. This angle is wrapped between $\pm\pi$ to compensate for the relative optical encoding. If the angle is not wrapped and the pendulum happens to do a full revolution before the catch is triggered, the stabilising controller will attempt to correct this very large error, relative to zero, by producing a large control signal. This results in rapid acceleration until the physical limit of the rail is hit.

9.2 Arduino

The implementation on the Arduino is based on the structure of the Simulink implementation. The code written for the Arduino consists of several files and libraries:

- `main.ino`
  The main file of the Arduino program which uses a switch-case structure as depicted in **figure 9.2**.
- `ctrl.c` / `ctrl.h`
  A C library which calculates the control action of both the swing-up and stabilising control laws derived in **chapters 7** and **8**. It also manages the friction feedforward, the catch trigger algorithm and calculating θ_w.
- `EKF.cpp` / `EKF.h`
  A C++-library to compute state estimations, implemented using the `Eigen` template library which supports matrix computations[18].
- `Joint.cpp` / `Joint.h`
  A provided C++ library in which low-level interaction between the hardware and the control etc. is implemented. It has been modified to operate in SI units.
- `params.h`
  A header file containing system-wide parameters.

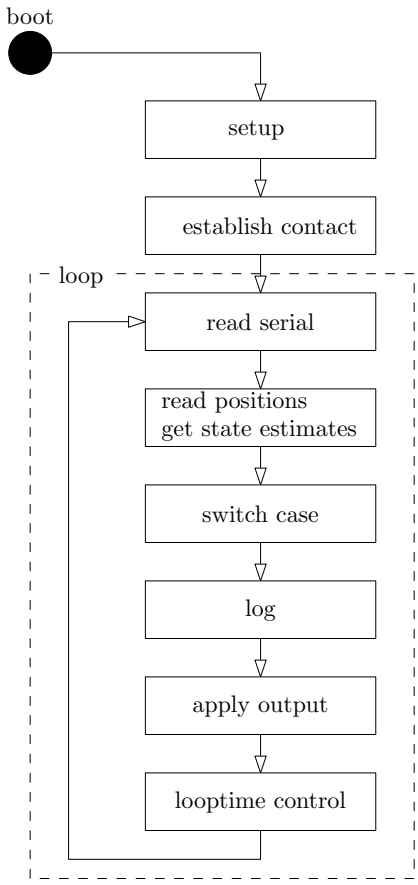


Figure 9.2: *State diagram of the Arduino code.*

When booted, the Arduino runs a setup protocol which creates and initializes two `Joint` C++ objects, *sled* and *pendulum*, and an `EKF` object, using the `Joint` and `EKF` library, respectively. It also initializes its peripherals, and enables the motor drivers. Then the Arduino goes into a waiting state, *establish contact*, where it listens on the serial line for a command.

When any command is received, it goes into the main loop, which is structured as follows:

1) In state *read serial*, any incoming command is interpreted.
   The available commands are defined as follows:

`L`	Enable/disable data logging.
`swup=x`	Set swing-up controller type to x.
`stab=x`	Set stabilising controller type to x.
`1/0`	Enable/disable control.
`r`	Reset encoders and EKF.

2) Positions are read using the `Joint.cpp`-library and states are estimated.
3) In the switch case, commands `0/1`, `L` and `r` are carried out.
   If the control action is enabled, a control signal is computed.
4) If the log enabled, a string is created and written to the serial line.
5) The control signal is applied to the motor, if control action is enabled.

6) A function ensures that each loop has a looptime of 5 ms.

7) Loops back to reading the serial.

9.2.1 Parameter tuning

The directional Coulomb friction for the cart, $c_{c,R}$ and $c_{c,L}$, have been considered as tuning parameters which is justified by their inconsistency. The implemented values are $c_{c,R} = 3.3$ and $c_{c,L} = 2.8$.

The weights of $\mathbf{Q}$ and $\mathbf{R}$ of the EKF has been tuned to obtain satisfactory results. Since measurement noise is lower than the quantization of the measurement signal, the weights have been tuned such that the estimate and measurement of x and θ are identical, as can be seen in **figure 9.3**. The new weights are

$$\mathbf{R} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.1 \end{bmatrix} \tag{9.1}$$

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{bmatrix} \tag{9.2}$$

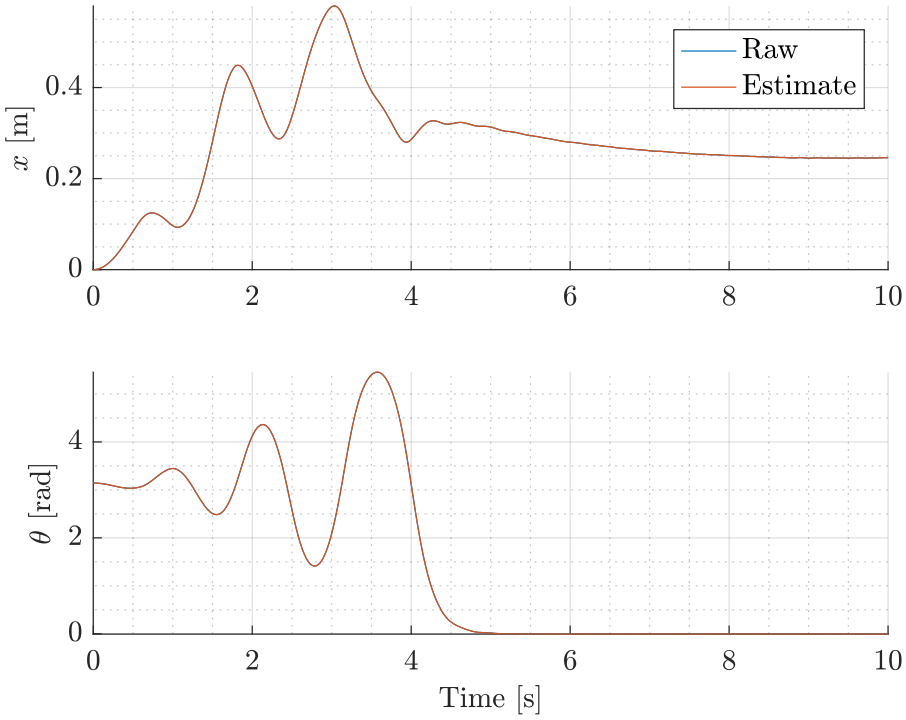


Figure 9.3: *Comparison between position estimates from the EKF and raw measurements.*

The estimates of the velocities are compared to a numerical derivative of the positions in **figure 9.4**.

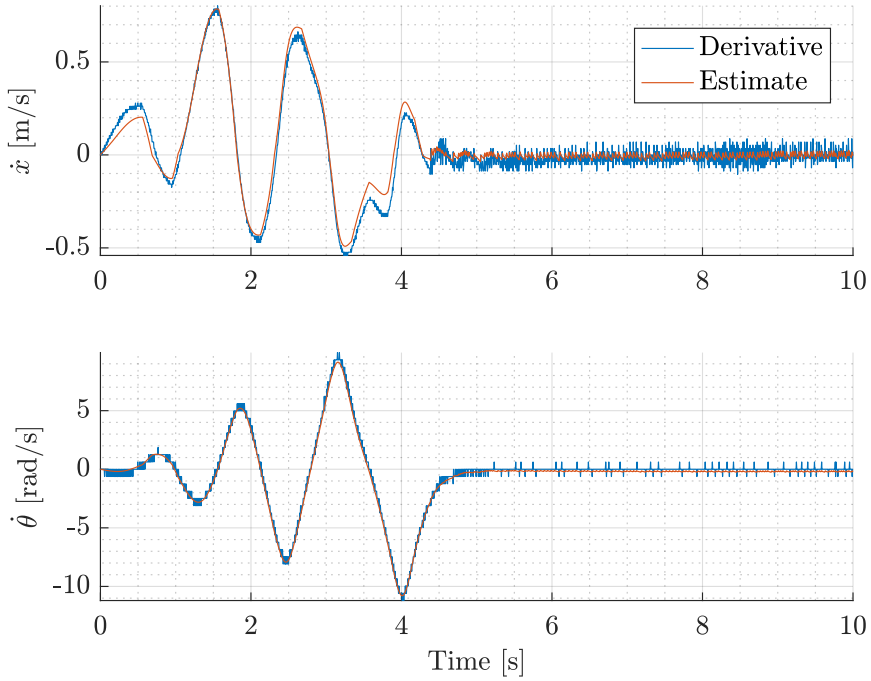


Figure 9.4: *Comparison between velocity estimates from the EKF and the numerical derivative of the measured positions.*

Finally, **figure 9.5** shows a comparison between a system test and an equivalent simulation using the same control configuration, namely the pendulum energy method for swing-up and LQG for stabilisation.

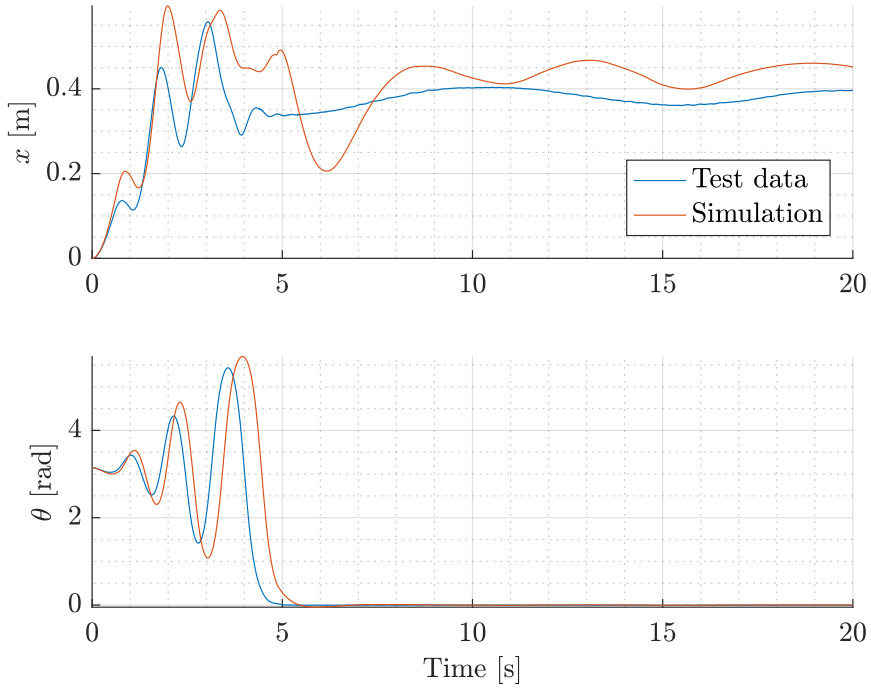


Figure 9.5: *Comparison between the implementation of the pendulum energy swing-up method using the LQG for stabilisation, and a simulation using the same control methods.*

Results 10

In this chapter, the results of this project will be presented. The chapter will be divided into three sections, each of which will deal with the results from a test related to the swing-up phase, the stabilisation and robustness, respectively.

10.1 Swing-up control

Tests are conducted to compare the three proposed methods for swinging up the pendulum, namely the *full energy method* described in **section 7.3.1**, the *pendulum energy method* described in **section 7.3.2** and finally the *sign-based method* described in **section 7.3.3**. **Figure 10.1** shows the resulting data from the tests.

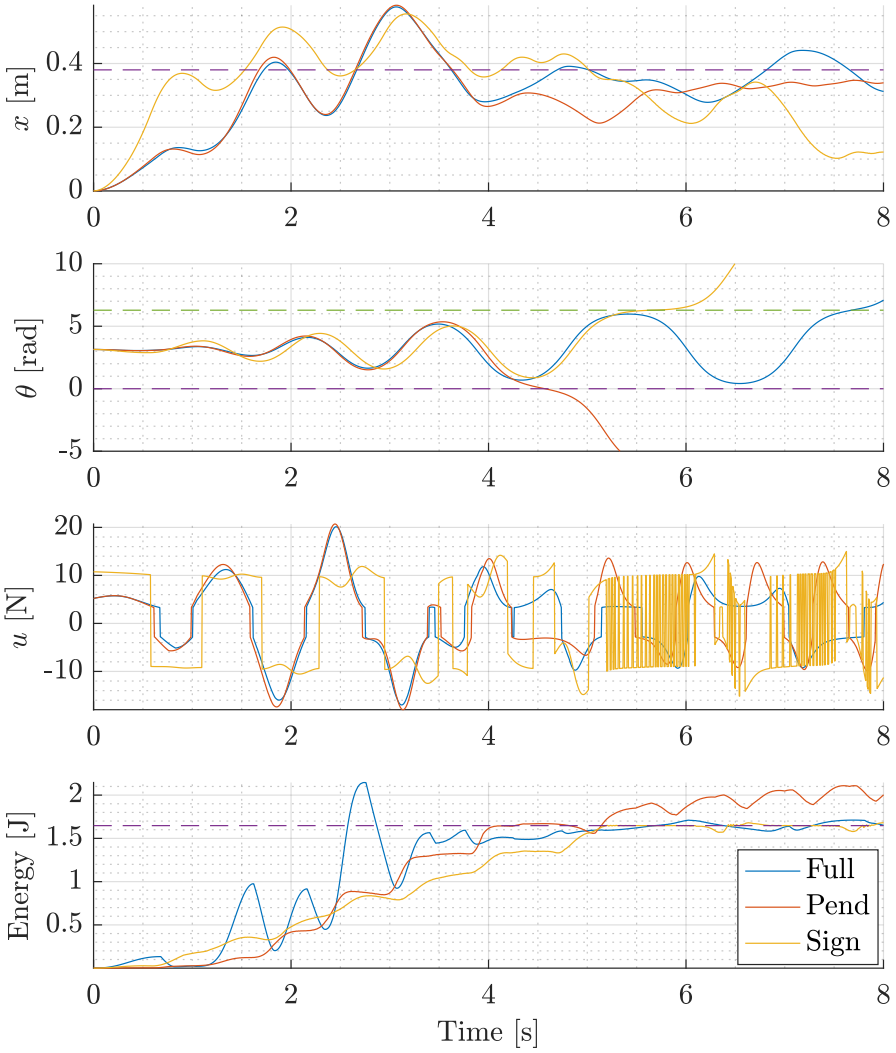


Figure 10.1: *Results from the swing-up control tests. Dashed lines indicate reference values. The subplots from top to bottom are: Cart position, pendulum angle, controller output, system/pendulum energy.*

The inverted pendulum is initialised in the leftmost position of the rail, hanging straight down, and the stabilising controller has been deactivated.

In terms of the cart position, all three methods keep the cart within the physical limits of the rail, though not exactly at the reference.

Additionally, all three methods manage to reach the unstable equilibria at angles zero and 2π, but due to the integral action they exactly swing beyond these points and perform a full revolution.

Considering the energy, all three methods manage to increase the energy in question to the reference value, although the pendulum energy method does overshoot. However, it also maintains an energy very close to the reference for about a second at $t = 4$ to $t = 5$, which is sufficient for the stabilising controller to catch the pendulum.

Finally, from the controller outputs u, it is evident how the sign-based method switches between ± 10, in contrast to the two other methods.

10.2 Stabilising control

For stabilisation, a PID-controller, and LQG-regulator and a SMC have been derived in **sections 8.1** to **8.3**, respectively.

In order to test them, the inverted pendulum has been started with the same method for swing-up, namely the pendulum energy method. When the catch-trigger criteria described in **section 6.4** are met, the stabilising control is used.

The results from this test are shown in **figures 10.2** and **10.3**.

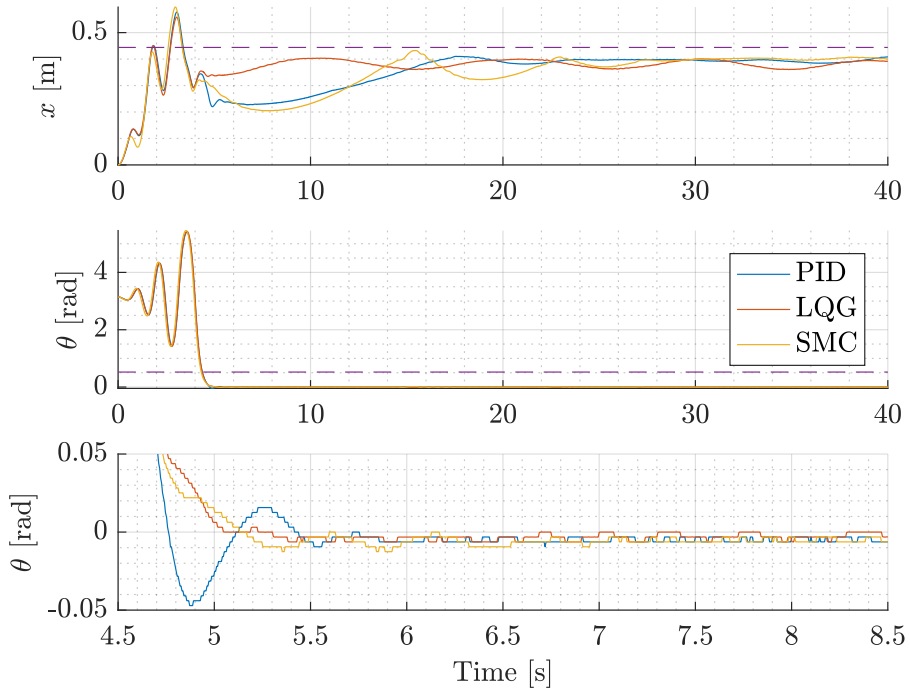


Figure 10.2: *Results from testing the stabilising controllers, all using the pendulum swing-up methods. The subplots from top top bottom are: Cart position, pendulum angle, pendulum angle at stable state.*

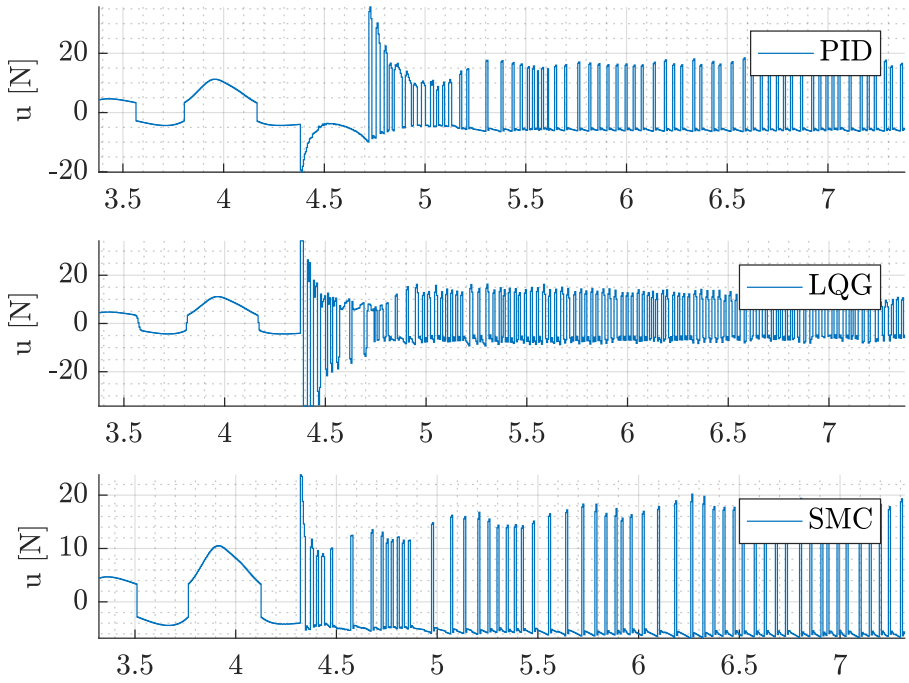


Figure 10.3: *Controller outputs from the test of the stabilising controllers, with zoom on the switch between the swing-up and stabilising phases.*

This test shows that all three stabilising controllers are capable of keeping the pendulum in the upright position. However, the PID tends to oscillate more than the two others. The cart is kept near the reference position, though with a small steady-state error and some oscillation.

In the controller outputs in **figure 10.3**, it is seen that the controllers all tend to chatter. This is due to the fact that the controllers must be aggressive to keep the pendulum upright, and because of the feedforward, as described in **section 6.3**. The sliding-mode control has larger peaks in the chattering, but also for short durations at a time.

Similar to the procedure in the respective ends of **sections 8.1** to **8.3**, the performance of the stabilising control at steady-state is quantified by calculating the MSEs over the final ten seconds of the data:

	PID	LQG	SMC	Unit
MSE_c	$2.640 \cdot 10^{-3}$	$4.120 \cdot 10^{-3}$	$1.811 \cdot 10^{-3}$	m
MSE_p	$20.99 \cdot 10^{-6}$	$14.00 \cdot 10^{-6}$	$16.29 \cdot 10^{-6}$	rad

10.3 Robustness

Finally, as stated in **section 3.1.2** (Additional requirements), the stabilising controllers must ensure some robustness with respect to variations in the pendulum tip-mass.

In order to test this, the tip-mass has been reduced by 26 g and 50 g, which corresponds to a parameter variation of approximately 10 and 20 percent.

The test is conducted using the pendulum energy method for swing-up. The results are shown in

figures 10.4 and **10.5**.

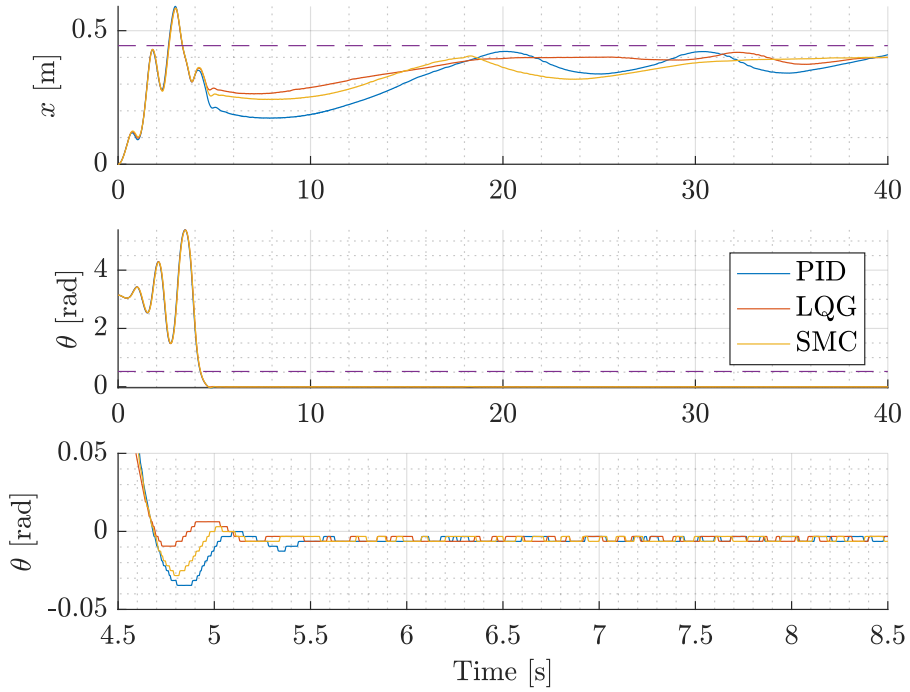


Figure 10.4: *Results of robustness test with* 26 g *removed from the tip mass. The pendulum energy methods is used for swing-up. Dashed lines indicates references.*

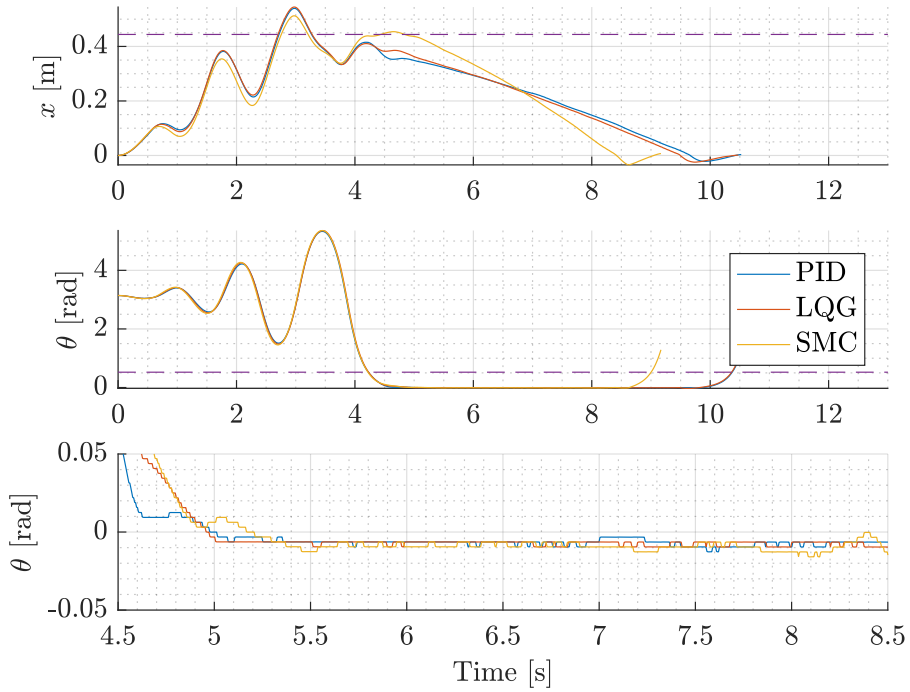


Figure 10.5: *Results of robustness test with* 50 g *removed from the tip mass. The pendulum energy methods is used for swing-up. Dashed lines indicates references.*

From this, it is seen that all controllers manage to stabilise the system when just 26 g is removed, though the LQG and SMC do oscillate more in the process, when compared to **figure 10.2**. The PID, on the other hand, seems to oscillate less in θ, but more in x.

Finally, when $50\ \mathrm{g}$ is removed, all of the controllers manage to stabilise the pendulum, but in doing so cause the cart to accelerate and hit the end of the rail.

Discussion 11

In this chapter, several topics presented throughout the thesis are discussed.

Cart friction

As presented in **section 4.3** (Cart and pendulum frictions), the final model of the friction in the system consists of a Coulomb model for the cart, and a Coulomb and viscous model for the pendulum. Several attempts to derive a more precise model of the cart friction are described in **appendix B** and **appendix C**, but discarded due to inconclusive results. These tests and analyses were made in the beginning of the project period, and measures to minimise friction in the system were attempted afterwards, as described in **section 2.4** (Rail and cart observations).

Better results may have been obtained by deriving a friction model based on e.g. the Dahl Model, or the Bristle Model. The Dahl Model has been used for adaptive friction compensation in servo system with ball bearings, and models the stress-strain curve by a differential equation. The Bristle Model attempts to model the behaviour of microscopical contact points between two surfaces as flexible bristles with a stiffness. When the surfaces move relative to each other, the strain in the bristles increases, which makes them act springs, giving rise to a friction force. [19]

In terms of minimising friction, a mechanical reconstruction of the system has been considered. The system is built with an intention of adding another pendulum to the cart. Because of this, the cart is larger than necessary when only supporting one pendulum. Minimising the cart size, thus mass, will theoretically decrease the friction between the cart and rail. Decreasing the number of points at which the cart is fastened to the rail decreases the contact surface and thus friction.

Addressing the way the cart is actuated through a toothed belt, it may be beneficial in terms of friction to implement another type of power transmission, e.g. based on a wire or pneumatics.

Aerodynamic drag

In the model derivation it is assumed that torque produced by aerodynamic drag on the pendulum is negligible. This is addressed in **appendix F**, which shows that the produced torque is low compared to gravitational torque. However, at e.g. peak velocity of approximately $11\ \mathrm{rad/s}$, this drag torque is larger than the friction torque. Including this may improve the accuracy of the pendulum model, and thus the control can be designed to compensate for it.

Control design

As presented in **section 7.1** (Pendulum energy method) and **section 7.2** (Pendulum energy method), two energy based swing-up methods are considered. These are based on a simplified system, that is, without frictions. The results in the respective sections indicate that the heteroclinic orbits and thus reference energy is not reached. This can be due omitting friction in the design, hence the control laws are not able to compensate for energy lost to friction.

Since heuristics are applied to the swing-up methods, the application of **theorem 6.1.2** (LaSalle's theorem) invalidated, and it is thus possible that the pendulum may exit the heteroclinic orbits or equivalently overshoot on the energy. However, the catch trigger algorithm provides a smooth transition between the swing-up and stabilising controllers, as seen in **chapter 10** (Results). In general, when experimenting with the catch angle during implementation, the stabilising controllers have been seen to struggle when the catch angle is less than ± 10 degrees.

Finally, during implementation of the stabilising controllers, the tuning process of the PID and LQG required more time to achieve satisfactory results. As opposed to this, the SMC required less, but it was more cumbersome to design.

Sensors

During implementation, an uncertainty has been found when initialising the optical encoder measuring the pendulum angle, as described in **appendix H**. When initialising the pendulum, resting at $\theta = \pi$, the encoder tends to come to rest at small values from a true, vertical position, presumably due to the Coulomb friction in the pendulum.

This could be optimized by using an absolute encoder, rather than the relative one, and calibrate it carefully, or by adding another sensor to the setup, e.g. an accelerometer. Having an extra sensor for the same measurement allows calibration of the encoder. Another solution could be to dismount the unused motor to which the pendulum is attached, and install a stand alone encoder directly on the pendulum, assuming the friction originates from this.

In addition to this, an accelerometer on the cart could be used to verify the assumption that the belt is non-elastic.

Implementation

The model is implemented using MATLAB's Simulink environment, as is described in **section 4.5** (Simulink implementation). The development during this project, either documenting or coding, has been managed with git version control. Unfortunately, Simulink's .slx files are not compatible with this, causing files to be overwritten is changed simultaneously by multiple developers, since the files cannot be merged. This could have been avoided by using Simulink Projects, a team collaborative design platform integrated with source control. [20]

The code written for the Arduino Due has been structured in C and C++ libraries as mentioned in **chapter 9** (Implementation). Implementation of the EKF computations requires matrix calculations, for which the Eigen-library is used, as this is not natively supported on the Arduino. These calculations require roughly half of the available processor time. If more efficient EKF computations are desired, alternatives such as C-code generation in MATLAB, using the MATLAB Coder, can be considered.[21]

Conclusion 12

In this thesis, the topic of nonlinear control is applied to an inverted pendulum on a cart, and used to swing up and stabilise the pendulum, with the purpose of comparing different methods, both linear and nonlinear, and their performance.

The control scheme has been based on a two-step strategy, containing a swinging phase and stabilising phase. The transistion between these phases are determined by a *catch trigger* algorithm with requirements on the pendulum angle and mechanical energy.

Three different nonlinear swing-up controllers have been developed, implemented, and compared. Two of the swing-up controllers are based on the energy of the pendulum, named *pendulum energy method* and *sign-based method*, and the third is based on the energy of the entire system, named *full energy method*. All of them are able to swing up the pendulum to its upright position, reaching the heteroclinic orbits of the pendulum, and meet the requirements stated in **section 3.1.2** (Additional requirements).

After implementation and testing, it has been found that the pendulum energy method provided best results, and is therefore used for further testing with the stabilising controllers. However, it is also found that the sign-based swing-up holds an advantage, as it does not need integral action on the energy error to reach the heteroclinic orbits, as the two other methods do. The integral action, while correcting the energy error, does also pose a risk to overshoot the energy reference, and thus exceed the heteroclinic orbits.

For stabilisation, three control schemes have been developed, implemented and compared: A nonlinear Sliding-Mode Controller, a Linear-Quadratic-Gaussian regulator, and a Proportional-Integral-Derivative controller. All of them are able to catch and stabilise the pendulum upright while keeping the cart close to its given reference. The results indicate minimal difference in performance in terms of stabilising the pendulum, and all the controllers meet the requirements stated in **section 3.1.2** (Additional requirements).

Finally, the stabilising controllers provide robustness with respect to 10% changes in the pendulum tip-mass.

Conclusively, system is able to autonomously swing up and stabilise the pendulum while keeping the cart safely within the physical limits of the test setup, and thus fulfils the requirements in **section 3.1.1** (Functional requirements).

Bibliography

[1] Stav Ziv. "Timeline: A Brief History of SpaceX's Reusable Rocket Launches". In: *Newsweek* (Jan. 20, 2016). URL: Timeline:%20A%20Brief%20History%20of%20SpaceX%E2%80%99s%20Reusable%20Rocket%20Launches (visited on 06/07/2016).

[2] SpaceX. *Official SpaceX Photos*. URL: https://www.flickr.com/photos/spacex/ (visited on 06/05/2017).

[3] Rhett Allain. "Why is it so difficult to land a rocket?" In: *Wired Magazine* (Jan. 19, 2015). URL: https://www.wired.com/2015/01/difficult-land-rocket/ (visited on 06/07/2017).

[4] H.K. Khalil. *Nonlinear Systems*. Third edition. Pearson Education. Prentice Hall, 2002.

[5] Nahum Shimkin. *Nonlinear Control Systems*. Lecture notes. Department of Electrical Engineering, Technion – Israel Institute of Technology, Haifa, Israel, 2009.

[6] *maxon motor control*. 4-Q-DC Servoamplifier ADS 50/10. maxon motor. July 2009.

[7] Arduino. *Arduino Due*. URL: https://www.arduino.cc/en/Main/arduinoBoardDue (visited on 02/21/2017).

[8] *maxon DC motor*. RE 50. maxon motor. Apr. 2014.

[9] *HEDM-55xx/560x and HEDS-55xx/56xx*. AV02-1046EN. Avago Technologies. Nov. 2014.

[10] *HCTL-2001-A00, HCTL-2017-A00/PLC, HCTL-2021-A00 / PLC*. AV01-0041EN. Avago Technologies. Feb. 2006.

[11] Simon J.A. Malham. *An introduction to Lagrangian and Hamiltonian mechanics*. Lecture notes. Aug. 2016.

[12] Ivan Virgala and Michal Kelemen. "Experimental Friction Identification of a DC Motor". In: *International Journal of M echanics and Applications* 3 (1 Mar. 1, 2013), pp. 26–30.

[13] Gautam Vallabha. *Real-Time Pacer for Simulink*. 2016. URL: https://mathworks.com/matlabcentral/fileexchange/29107-real-time-pacer-for-simulink.

[14] Torben Knudsen. *Kalman filter - for nonlinear systems*. Lecture notes. Aalborg University, 2016.

[15] K. J. Åstrøm and K. Furuta. "Swinging Up a Pendulum by Energy Control". In: *IFAC World Congress* (13 June 30, 1996), p. 42.

[16] Isabelle Fantoni Rogelio Lozano and Dan J. Block. "Stabilization of the inverted pendulum around its homoclinic orbit". In: *Systems and Control Letters* (40 Jan. 31, 2000), pp. 197–204.

[17] Y Nishioka M Ishitobi Y Ohta and H Kinoshita. "Swing-up of a cart-pendulum system with friction by energy control". In: *Proceedings of the Institution of Mechanical Engineers* (218 Mar. 11, 2004), Part I:J. Systems and Control Engineering.

[18] Christian Seiler Christoph Hertzberg Jitse Niesen. *Eigen*. URL: http://eigen.tuxfamily.org/index.php?title=Main_Page (visited on 06/02/2017).

[19] K. J. Åstrøm, H. Olsson, C. Canudas de Wit, M. Gäfvert, and P. Lischinsky. "Friction Models and Friction Compensation". In: *Nonlinear and Adaptive Control: Towards a design methodology for Physical Systems* (Nov. 28, 1997), p. 37.

[20] MathWorks. *What Are Simulink Projects?* URL: https://se.mathworks.com/help/simulink/ug/what-are-simulink-projects.html (visited on 06/01/2017).

[21] MathWorks. *C Code Generation from Matlab*. URL: https://se.mathworks.com/help/comm/ug/code-generation-from-matlab.html (visited on 06/01/2017).

[22] Carl Rod Nave. *Damped Harmonic Oscillator*. Feb. 28, 2017. URL: http://hyperphysics.phy-astr.gsu.edu/hbase/oscda.html.

[23] Yunus A. Çengel. *Fundamentals of Thermal-Fluid Sciences*. 4th.
ISBN-13: 978-0077422400, ISBN-10: 0077422406. McGraw-Hill, Jan. 1, 2012.

Integer Value to Current A

This appendix describes the test conducted to determine the relation between the Arduino setpoint to a current running through the motor, i_a. This is done by utilizing the provided library *Joint.cpp*, and applying the motor several different step inputs.

A.1 System overview

As described in **chapter 2**, the circuit powering the motor is defined as in **figure A.1**

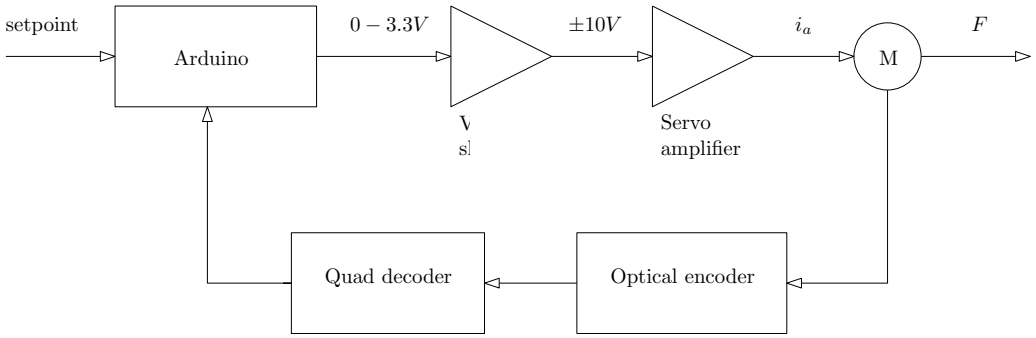


Figure A.1: *The circuit powering the motor. The Arduino output is amplified to ± 10 V, which is fed into the maxon ServoAmplifier providing current to the motor.*

The maxon servo amplifier allows for direct measurement of the current supplied to the motor it is running. It is therefore possible to construct a direct relationship between the setpoint in the Arduino to the motor current i_a, by simplifying **figure A.1** as depicted in **figure A.2**.

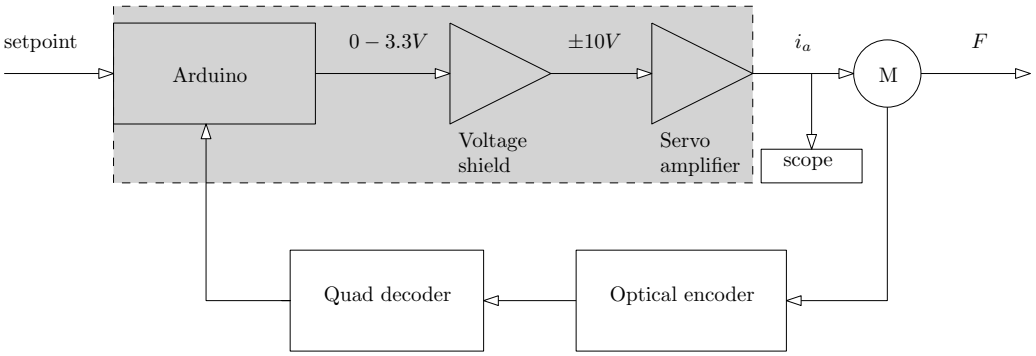


Figure A.2: *Simplified electrical powering circuit. The voltage amplification stages are ignored, and the current is measured directly from the servo amplifier.*

The output from the servo amplifier is a voltage with a gradient of approximately 0.4 V/A, thus a small conversion loss much be expected. **Figure A.3** displays the obtained results.

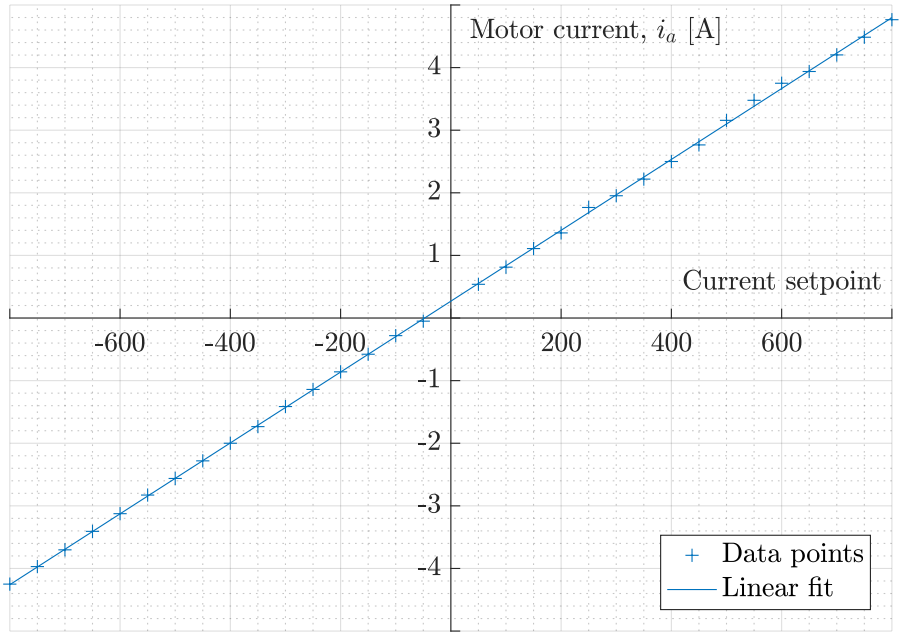


Figure A.3: *Test results*

The actual setpoint output in the Arduino is an 12-bit integer value between 0 and 4095, representing a voltage between 0 and 3.3 V, written on DAC0. However, to confine with an output voltage of $\pm$10 V after the voltage shield, an offset of 2000 is used to define 0 V out. This is incorporated in the *Joint.cpp* library. This means a negative setpoint drives the motor in one direction, and a positive setpoint drives it in the other direction. It should be noted, when inspecting the results, an integer value of 0 does not provide 0 current. The input corresponding to a zero current is $\sim$ -47, and thus the offset in the *Joint.cpp* library have been shifted to 1953.

A.2 Relation between Arduino setpoint and force

The force applied to the cart is defined by the motor torque divided by the length from the motor's rotor till the belt.

$$F = \frac{i_a \cdot K_M}{r} = i_a \frac{0.0934}{0.03} = i_a \cdot 3.1133 \tag{A.1}$$

where K_M is the motor torque constant, and r is the distance from the motor's rotor to the belt driving the cart. The torque constant is found in the datasheet, and is defined as 93.4 mNm/A, and the distance r is measured to 3 cm. The current i_a is defined as the Arduino setpoint times the gradient found from a linear fitting of the data presented in **figure A.3**.

$$i_a = setpoint \cdot 0.0056605 \tag{A.2}$$

Replacing this into **equation (A.1)** provides

$$F = setpoint \cdot 0.0056605 \cdot 3.1133 = setpoint \cdot 0.017623 \tag{A.3}$$

Thus a constant of $1/0.017623 = 56.744$ can the be used to convert a calculated force to a setpoint integer value.

Distributed Cart Coulomb Friction B

This appendix describes the tests conducted to determine the coulomb and viscous frictions of the motor and cart. The term "distributed" refers to how – in order to obtain precise estimations – the rail have been divided into four parts, such that frictions have been determined for each part of the rail.

B.1 System inspection

By first inspection of the system, it became clear that the coulomb friction differs depending on the placement of the cart. When moving the cart from the right-hand side towards the left, the cart experiences different sizes of friction becoming more rigid at the left-hand side. Furthermore, inspections also indicate that the viscous friction differs depending on the direction of the cart. Therefore a thorough cleaning of the rails was initiated, and after the cleaning they were also greased with MegaGlide grease, with the purpose of minimizing the frictions.

For the test, the rail have been divided into parts as depicted in **figure B.1**

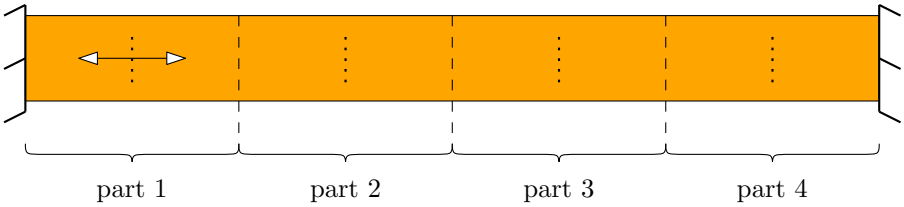


Figure B.1: *The rail divided by four sections.*

The cart have then been placed in the center of each part, represented by the dotted lines. In order to determine the coulomb friction, three tests have been conducted in both directions in each part using a ramp signal. In order to determine the viscous friction, a hand-tuned PI velocity controller have been developed to maintain a constant velocity. In this case, the pendulum have been placed at either end, and moved with a constant velocity across the rail.

The input signal used for testing is a force ramp scaled by the coefficient found in **appendix A**.

B.2 Test results - coulomb friction

The test results are presented as a figure for each part of the rail, and a combined table below with determined frictions. Each figure then contains six tests, three in each direction. The coulomb frictions have been estimated by selectively averaging the amount of force needed to create movement of the cart.

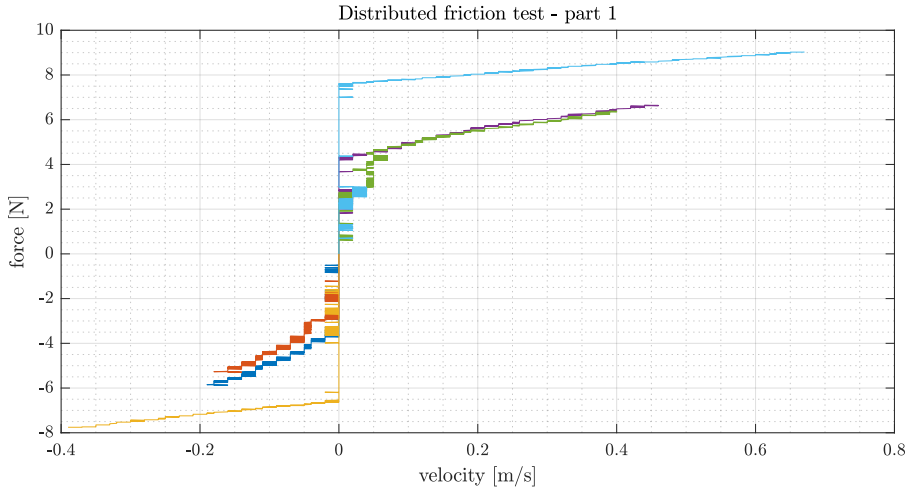


Figure B.2: *Friction test results of part 1 of the rail.*

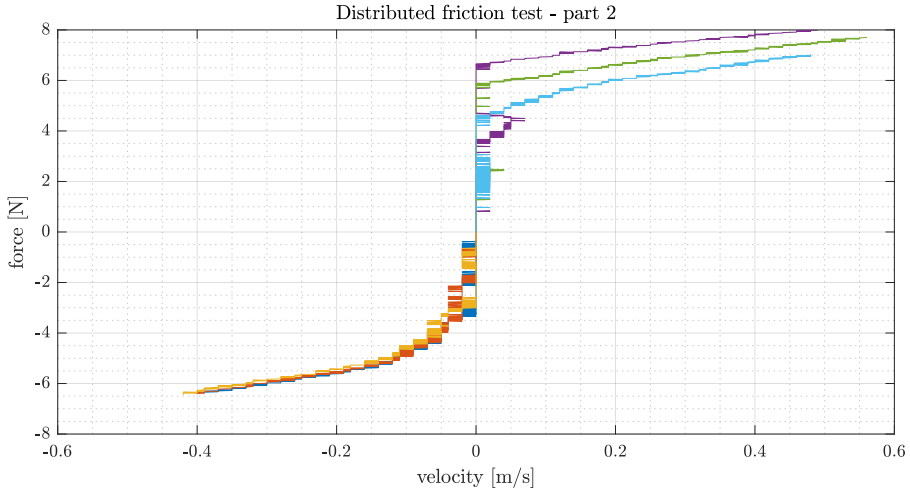


Figure B.3: *Friction test results of part 2 of the rail.*

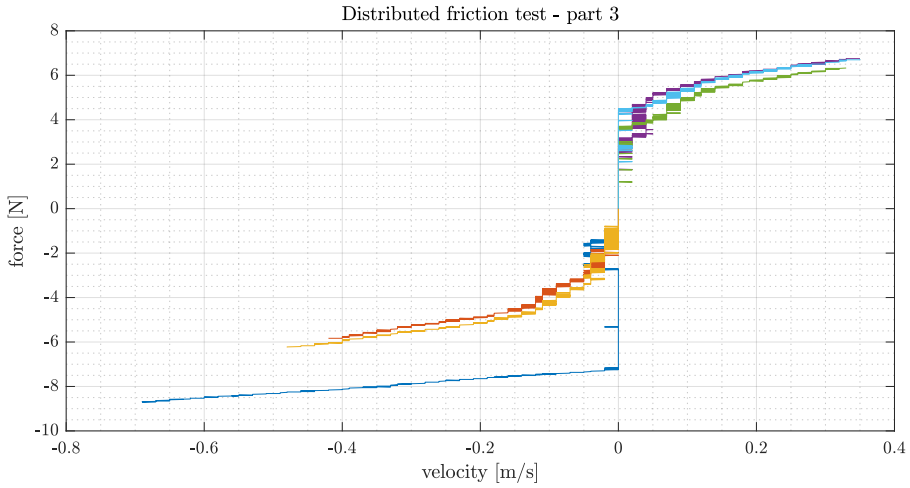


Figure B.4: *Friction test results of part 3 of the rail.*

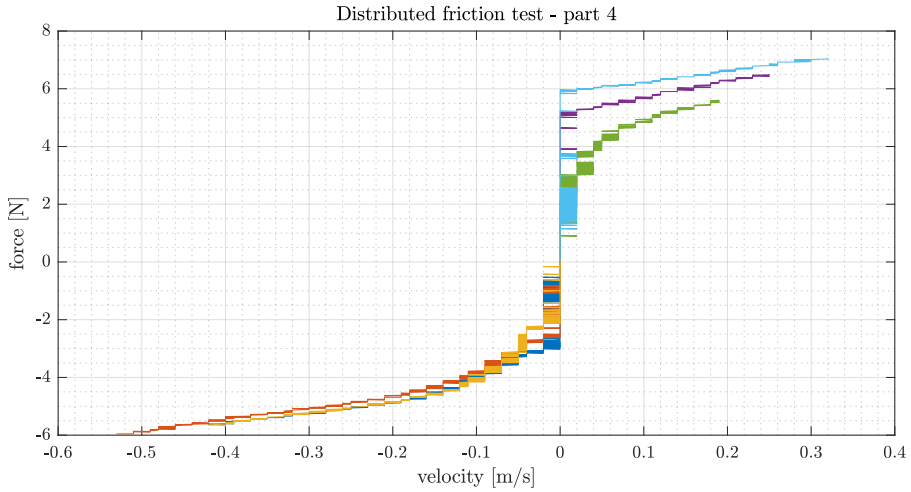


Figure B.5: *Friction test results of part 4 of the rail.*

Table B.1 summarizes coulomb frictions of each part and in both directions which have been determined from the presented results.

	part 1	part 2	part 3	part 4
$F_{c,r}$	2.943	2.467	1.692	1.128
$F_{c,l}$	-3.463	-2.855	-2.309	-1.956

Table B.1: *Coulomb frictions in newtons [N]*

The results conform well with the initial inspection of the system. Even though a thorough cleaning of rails was done, the cart still experience the prescribed frictions. The coulomb friction it is significantly higher when moving the cart the to right than moving the cart left, and is it generally lowest in the right hand side of the rail – part 3 and 4 – as mentioned.

Furthermore, as it can be seen in every figure, some tests have a larger coulomb friction than others. In **figure B.2** the yellow, and light blue lines have larger coulomb frictions and a very linear viscous friction. However, they do have velocity spikes in close vicinity of the other tests, but then return to zero velocity before enough force have been applied to overcome the coulomb friction. In **figure B.3** the purple line also have a velocity spike near the light blue line, before again returning to zero velocity. The same can also be seen for the blue line in **figure B.4**, and light blue line in **figure B.5**.
These discontinuities have been determined to be caused by the ball bearing driving the cart. This have been concluded by disassembling the belt from the cart to inspect the parts individually. The motor actuating the belt wells runs smoothly, and the ball bearings of the cart runs fairly smooth, but some discontinuities can be identified. The transitions between the gears of belt contains a surprising amount of friction, which must amplify the friction from the ball bearings, creating the extra coulomb friction seen in the figures. The friction between the belt and the wheels can be experienced by mounting the belt on the wheels, and manually moving it back and forth.
After consultation with the smith Jesper, and electronic technologist Simon, it was decided to order a new belt with teflon treated surface in hopes of minimizing the friction between gear transitions.

B.3 Attempt to model Coulomb friction position dependency

This section attempts to determine if there exists a position dependency of the Coulomb friction.

For the Coulomb friction coefficient, c_x, a ramping current input has been used to increase the torque produced by the motor and thus the force applied to the cart. The Coulomb friction is then the current required to set the cart in motion. This has been repeated at several places along the rail and in both directions in order to check for consistency, as depicted in **figure B.6**.

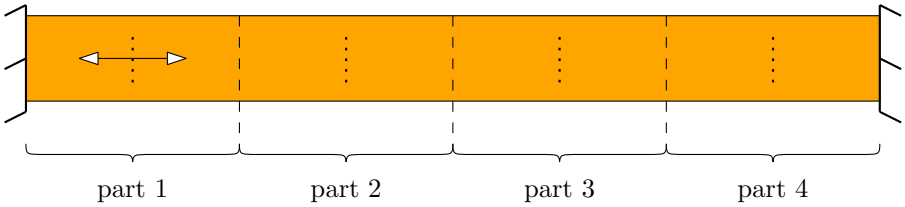


Figure B.6: *Segments which the rail has been divided into.*

The results of the experiment is shown in **figure B.7**.

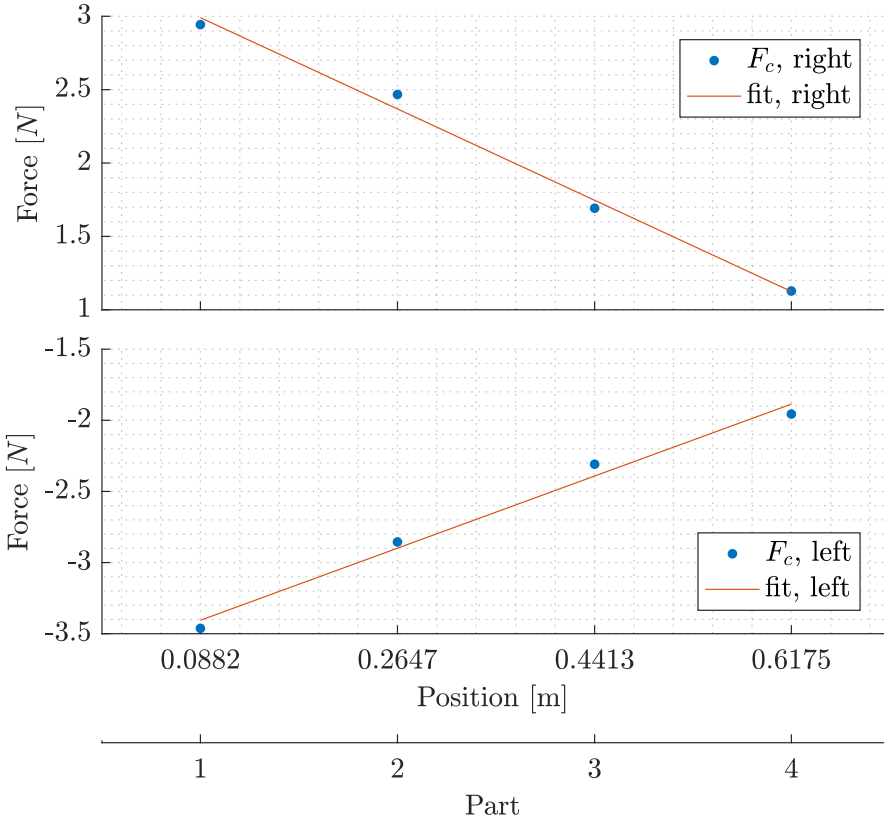


Figure B.7: *Coulomb frictions determined for different locations of the rail and in both directions.*

The data is fitted to the following linear functions:

$$c_{x,l}(x) = 2.8709x - 3.6588 \tag{B.1}$$

$$c_{x,r}(x) = -3.5249x + 3.3015 \tag{B.2}$$

Cart Viscous Friction C

This appendix describes the tests conducted in order to determine the viscous friction of the cart. A hand-tuned PI velocity controller have been developed, and used in two different test setups. The first test was conducted using MATLAB for logging purposes, and one using a Tektronix TDS2004B Oscilloscope.

A velocity controller have been chosen since a constant constant velocity equals zero acceleration, and thus the friction of the system is isolated as follows

$$F_{in} - F_{fric} = m\ddot{x} \tag{C.1}$$

$$F_{in} = F_{fric} \tag{C.2}$$

Thus the input force needed to move the cart is expressed by the combined friction force which is the coulomb and viscous friction. Since the coulomb friction is known at this point, the viscous friction can be isolated by doing tests at different velocities.

In order to make sure the coulomb friction is overcome, each test have been conducted at 0.4 m/s, 0.6 m/s, 0.8 m/s, and 1 m/s in both directions. Thus the cart have been placed in each end of the rail, and driven across its entire range.

The control action of the PI controller is calculated as follows

```
setpoint = -(V_ref - velSled) * V_Kp - V_acc_err * V_Ki * SAMPLINGTIME
```

where V_{ref} is the velocity reference, $velSled$ is the calculated velocity of the sled, V_{Kp} is the proportional gain, and V_{Ki} is the integral gain. The accumulated error, V_{acc_err} is calculated as

```
V_acc_err += V_ref - velSled;
```

Since the Arduino, using the library Joint.cpp, measures a position in mm, and converts this to a speed in m/s, the controller contains large gains where $V_{Kp} = 3000$, and $V_{Ki} = 4500$.

C.1 Test logged with MATLAB

When using MATLAB to log the data from the Arduino, the test setup is no different than the setup described in **chapter 2**. MATLAB sends a start command to the Arduino, and it responds with measured data on the serial line. Using the coefficients described in **appendix A**, the input signal have been directly converted to the motor current, i_a.
To present the results several figures have been created, and comments will follow below.

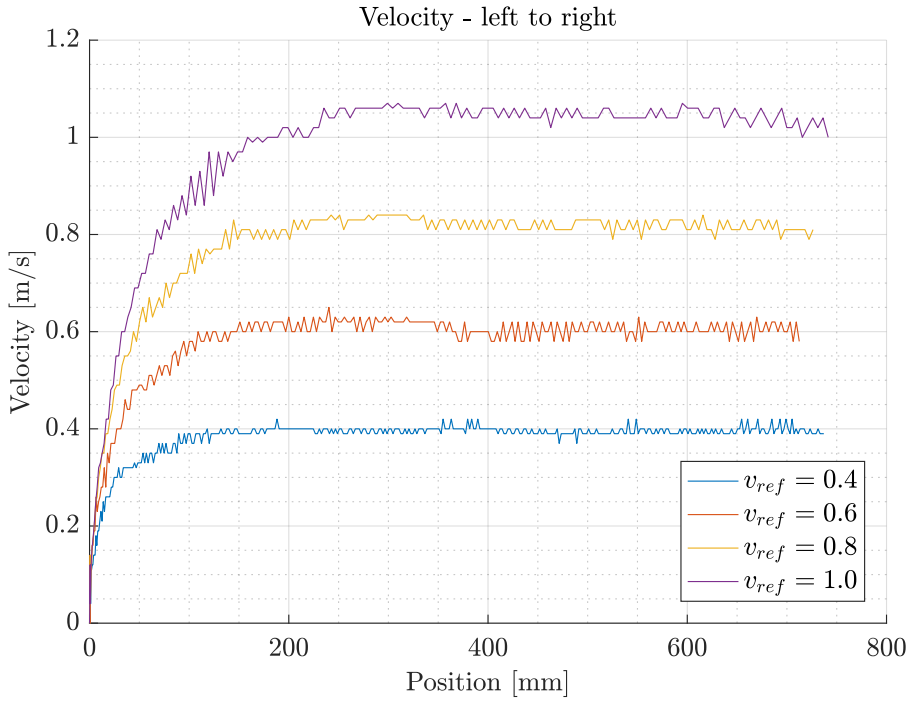


Figure C.1: *Velocity references when the cart moves from the left-hand side to the right-hand side, or from part 1 to part 4.*

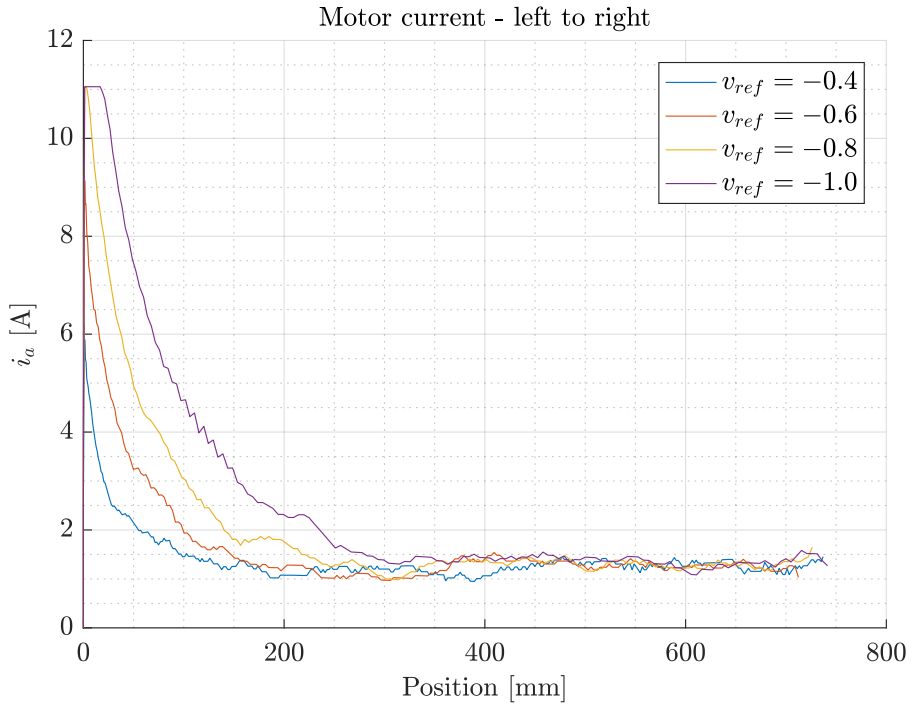


Figure C.2: *Motor current used to achieve the velocities shown in* ***figure C.1***

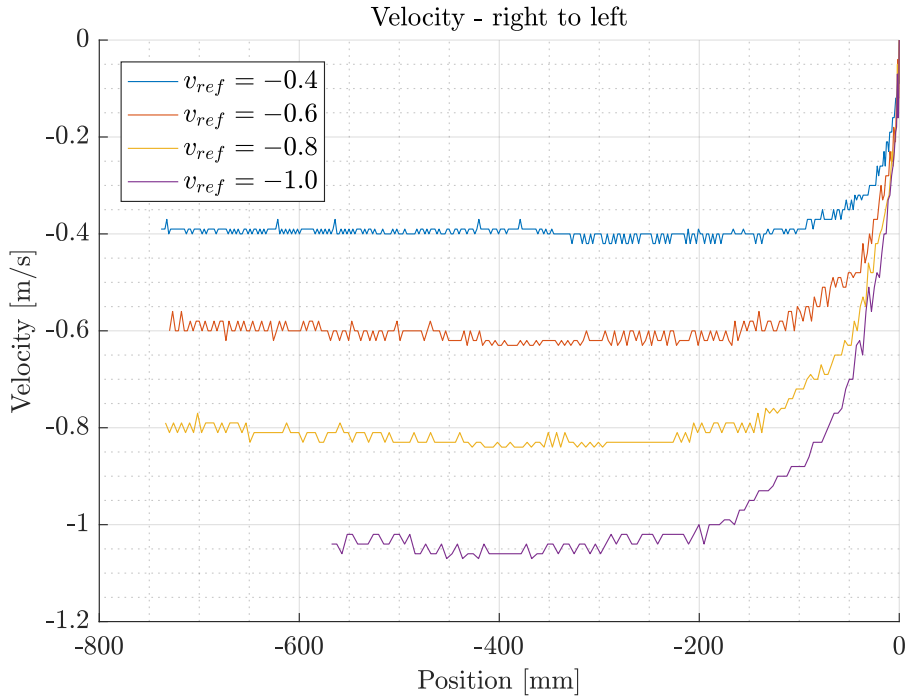


Figure C.3: *Velocity references when the car moves from the right-hand side to the left-hand side, or from part 4 to part 1.*

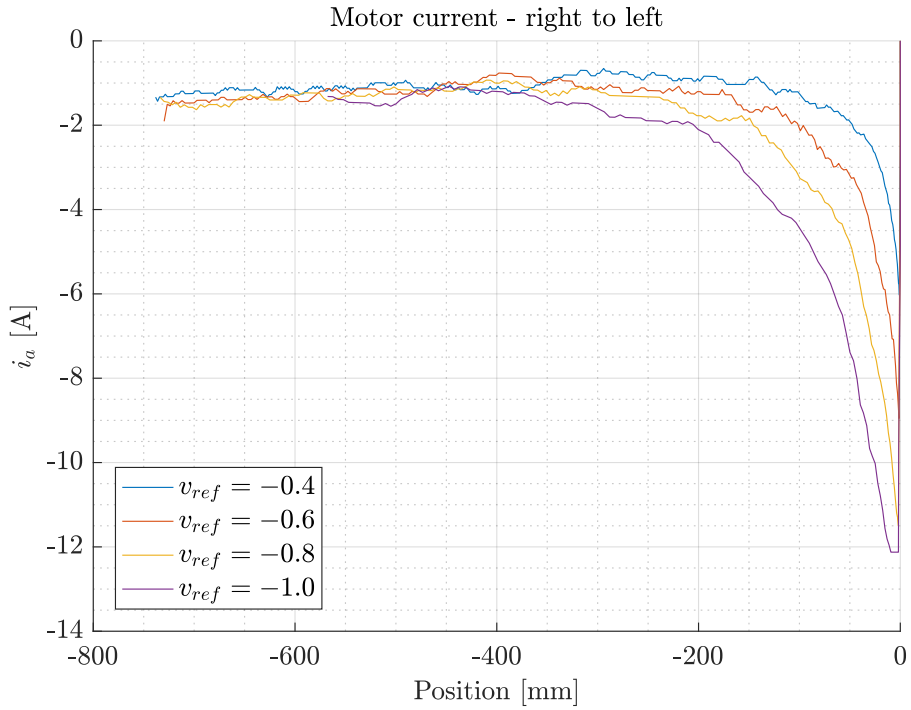


Figure C.4: *Motor current used to achieve the velocities shown in* ***figure C.3***

It can be seen in **figure C.1** and **figure C.3** that the controller needs some time to obtain the reference velocity, thus the viscous friction cannot be determined in the first parts of the tests. In both figures the velocity is constant after the sled have moved roughly 400 mm. At a reference velocity of ± 1 m/s, the calculated current saturates at 11 A and -12 A respectively. This means the servo amplifier

provides maximum outputs, and tests with faster velocity references have not been conducted.

The motor current in **figure C.2** for each velocity reference becomes identical after the sled have moved 400 mm, thus giving an indication of that the viscous friction is very small, and maybe not significant. Ideally the current for each reference should settle at larger values for faster references. Inspecting the figure closely, from 450 mm to 700 mm, it can be seen that the average tendency of the current is declining, however at a small rate. This confines well with the experienced Coulomb friction described in **appendix B**, where the Coulomb friction decreases as the cart moves from part 1 to part 4.
The current in **figure C.4** for the cart moving from part 4 towards part 1 also conforms well with the described Coulomb friction of the cart. Even though the current for all references are nearly identical, they all share the same tendency. As described in **appendix B**, the Coulomb friction rises when moving from part 4 to part 1. This is verified by this test, since the velocity in **figure C.3** is held constant – but the current increases as the cart moves on the rail.

Summarizing this test, the viscous friction could not be determined using MATLAB for logging data. The difference between the currents at constant velocities is too small. This implies that the viscous friction is very small, and could be neglected.

C.2 Test logged with oscilloscope

Since the servo amplifier saturates the output in the previous tests, it would be interesting to measure the actual output, and compare it to the calculated, thus verifying the approximation done in **appendix A**. With this test, it is also hoped to achieve a higher resolution of the current and thereby being able to identify the viscous friction. Using an oscilloscope to measure the current, the test setup have changed a bit, as depicted in **figure C.5**

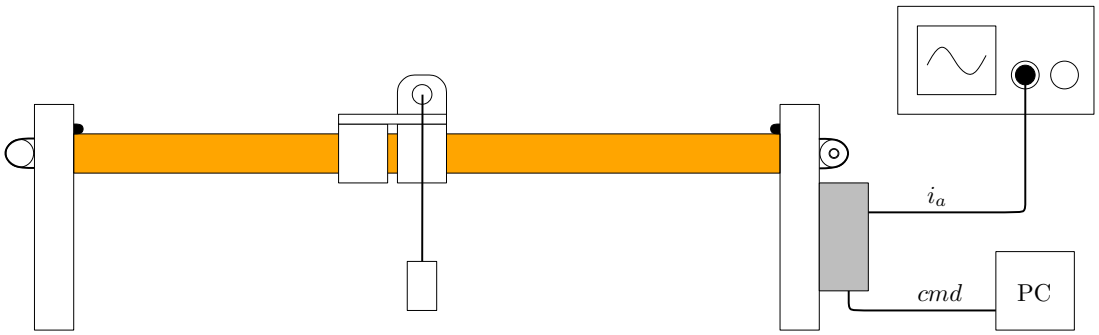
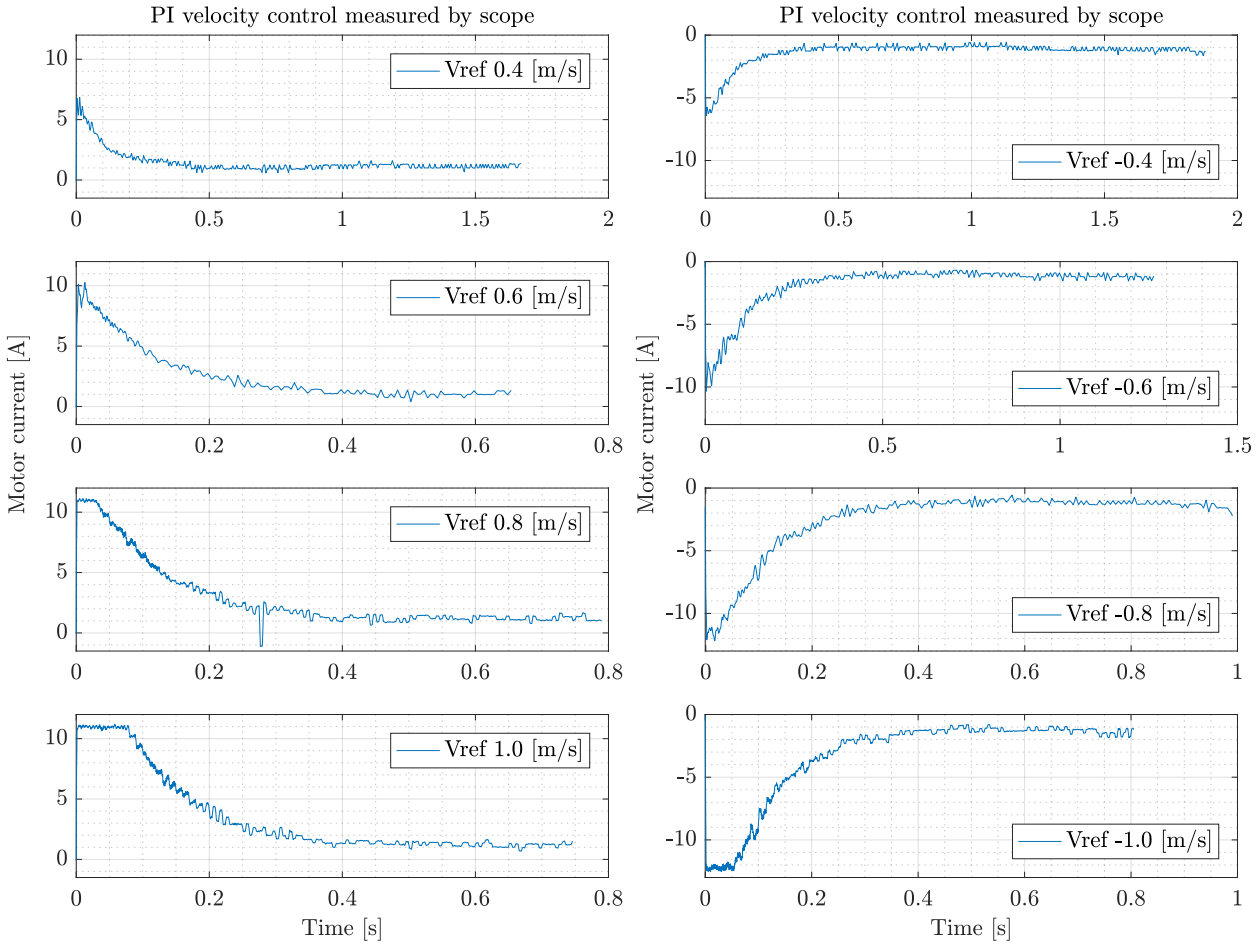


Figure C.5: *Test setup using an oscilloscope to measure to motor current directly from the servo amplifier*

The oscilloscope is connected directly to the servo amplifier, which outputs the motor current and needs to be scaled with a gradient of approximately 0.4 V/A. Since MATLAB is not used, the Arduino program is started by writing directly on the serial line using the Arduino IDE's Serial Monitor.
The tests have been conducted using the same PI controller as in **appendix C.1**, and it is therefore known that the velocity becomes constant after the sled have moved roughly half the length of the rail – thus the current measurements stands alone.



(a) *Positive velocity reference* **(b)** *Negative velocity reference*

Figure C.6

Inspecting **figure C.6a** closely, when the current is constant, only the tests with a reference of 0.6 m/s and 1 m/s includes a decreasing rate of the current, implying correspondence with the estimated Coulomb friction. However, in all four tests, the current settles in too close proximity of each other – indicating again, that the viscous friction is very small.
In **figure C.6b** the current in all tests settles too close to each other, preventing an identification of the viscous friction. Inspecting the individual subplots, it can be seen, that the current increases over time confining with the identified Coulomb friction, which increases when the cart moves from part 4 to part 1.

As it can be seen in both figures, the current saturates at approximately 11 A and -12 A, which corresponds nicely to the calculated maximum and minimum current depicted in **figure C.1** and **figure C.3** respectively. Thus concluding that the conversion from an integer value to a current, described in **appendix A**, is linear in the entire spectrum. The current resolution have not improved significantly enough, concluding that the calculated motor current will suffice for further use in this project.

Test of Teflon-coated Belt D

When the new belt was installed, some viscous friction tests was conducted, to see whether the new belt have minimized the friction or not. **Figure D.1** and **figure D.2** displays the results of tests done with a velocity reference of -0.5 m/s, and 0.5 m/s, hence a test in either direction.

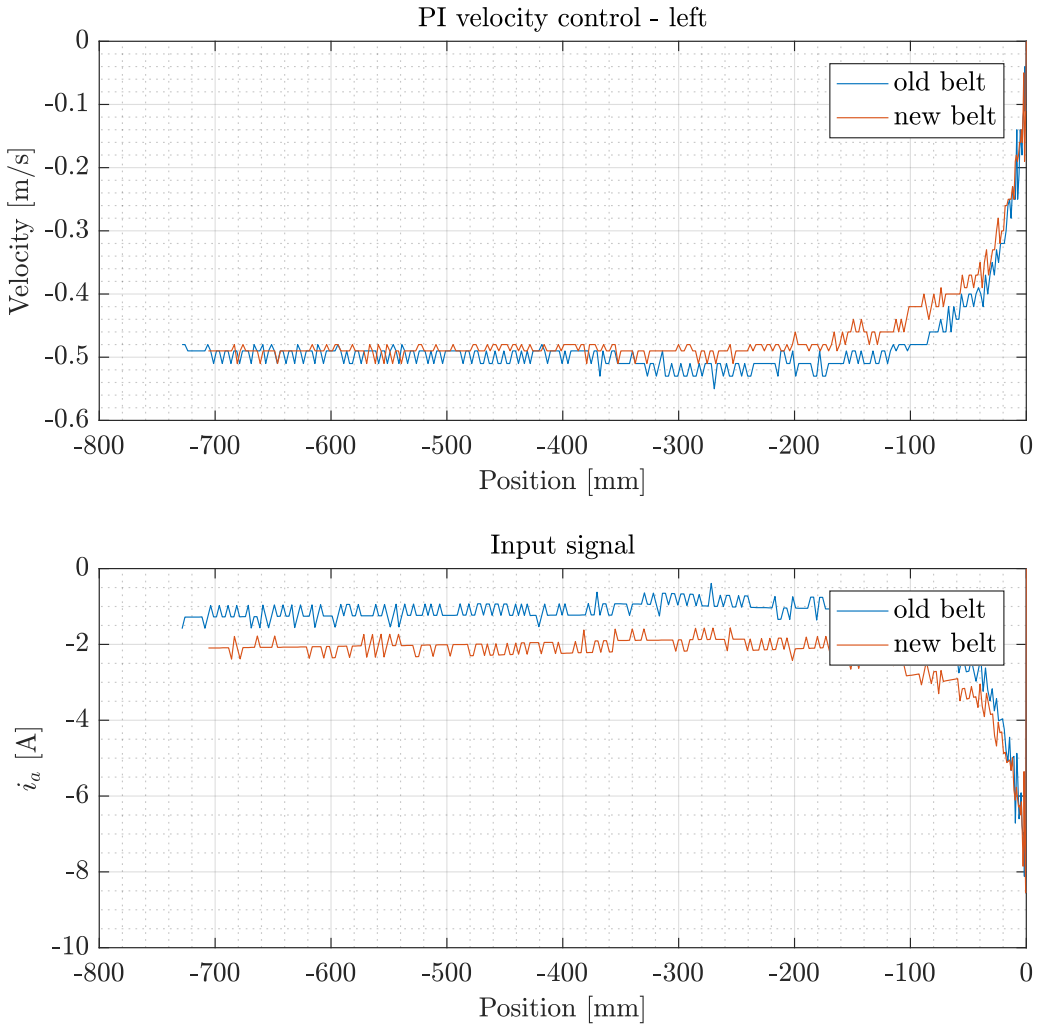


Figure D.1: *PI velocity test. Notice how the red graph(new belt) have a slightly slower rise time than the blue(old belt) on the top figure. Bottom figure displays motor current, and the red curve is stable around -2 [A], where the blue is stable around -1 [A].*

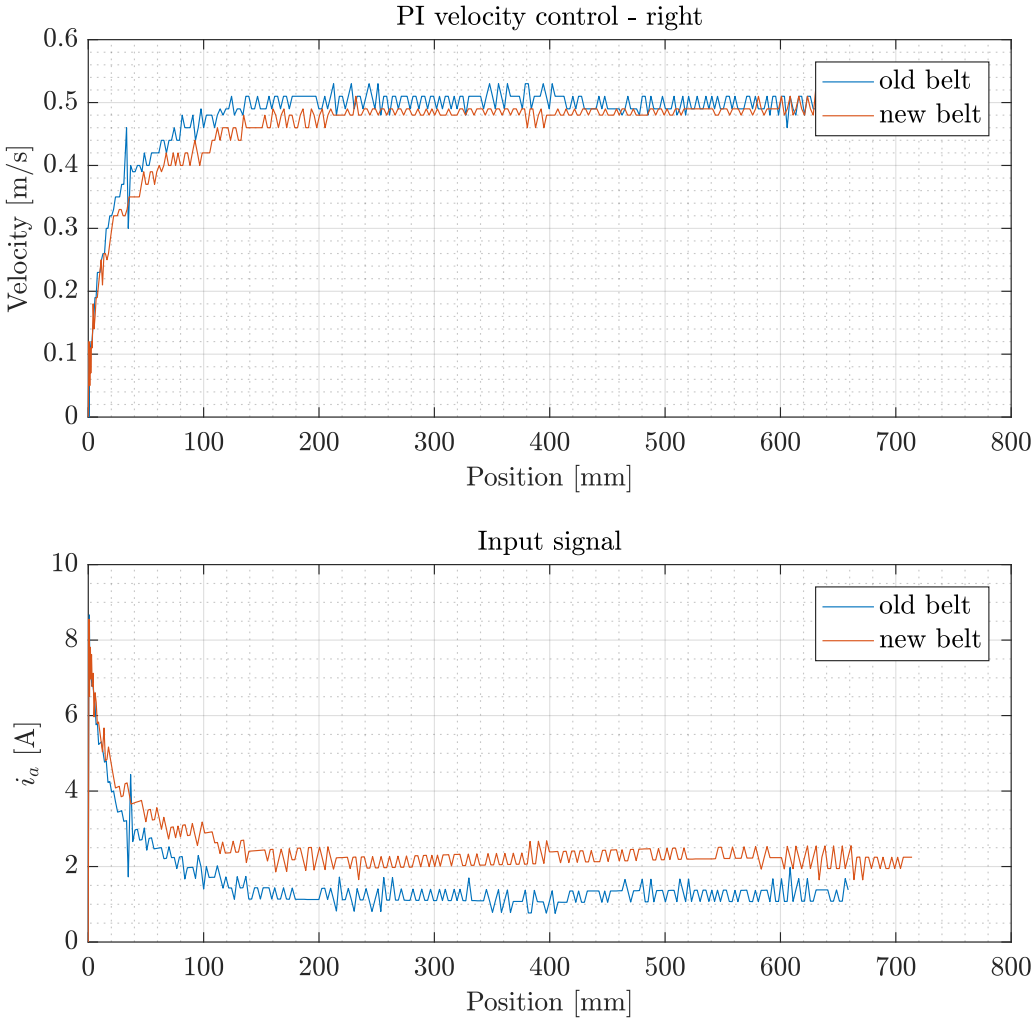


Figure D.2: *Rise times on the top figure are almost identical, but from the bottom figure, it can be seen that the new belt introduces a large friction due to an increase in motor current.*

As it can be seen in both figures, the new belt causes the controller to deliver more current to the motor, hence more friction is introduced using the new belt. After some considerations, and more tests, all with the same outcome, it was decided to return to the belt used in the first place.

Pendulum Friction Estimation E

This appendix describes the tests conducted in order to find the necessary coefficients for the pendulum friction model.

E.1 Coulomb coefficient

In order to determine the coulomb coefficient, c_θ, the pendulum alone is considered, as shown in **figure E.1**.

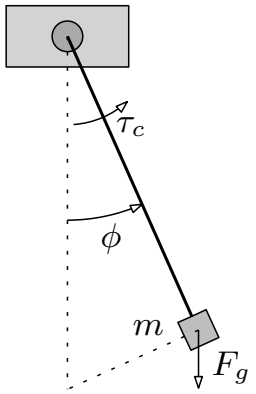


Figure E.1: *Pendulum with parameters used for coulomb friction estimation.*

Using a small end-mass, the pendulum is manually lifted very carefully to either the right or left side of the bottom equilibrium point, until largest possible angle at which the pendulum can hold still.

At this point, the the gravitational pull is not large enough to overcome the Coulomb friction, thus the position is maintained.

Then, the angle is measured, and the procedure repeated several times. Then, for the known mass m_p, the torque produced by the Coulomb friction is:

$$\tau_c = -\tau_g = -F_g l \sin(\phi) = m_p g l \sin(\phi) \tag{E.1}$$

Using parameters $m_p = 0.026$ kg, $g = 9.81$ m/s^2 and $l = l_{rod} + 0.5 \cdot l_{m26g} = 0.2875$ m, the results are:

ϕ	-0.0408	-0.0377	-0.0440	0.0377	0.0408	0.0408
τ_c	-0.0030	-0.0028	-0.0032	0.0028	0.0030	0.0030

Taking the mean of the absolute values of τ_c yields the Coulomb coefficient $c_p = 0.0030$ for the pendulum.

E.2 Viscous coefficient

In order to determine the viscous friction coefficient, v_p, the pendulum is considered like in **figure E.1**, but it is now raised manually to the vicinity of the top equilibrium point at $\phi = \pi$, and then released while the cart is kept at a fixed position.

The pendulum will start oscillating and is expected to behave like a underdamped harmonic oscillator, for which the position will vary with a exponentially declining cosine curve, i.e.:

$$\theta(t) = \theta_0 \exp^{-\gamma t} \cos(\omega t - \alpha), \qquad \theta_0 = \theta(t = 0) \tag{E.2}$$

Considering only the amplitude of the oscillation, **equation (E.2)** becomes:

$$|\theta| = \theta_0 \exp^{-\gamma t} \tag{E.3}$$

$$\gamma = \frac{v_p}{2m_p} \tag{E.4}$$

Using MATLAB-function `findpeaks` to find the peaks of the oscillation, and then fitting them to **equation (E.3)**, yields the results shown in **figure E.2** for three different masses.

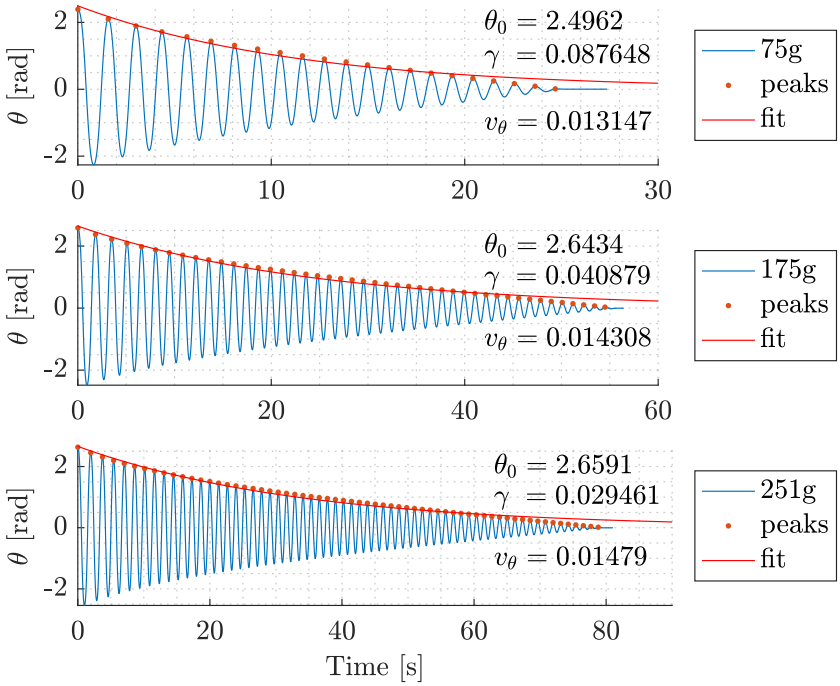


Figure E.2: *Measurements for free, underdamped, harmonic oscillation with peaks, fit and fit results θ_0, γ, v_p.*

Figure E.2 shows that v_p varies slightly with the three test scenarios. However, as $m_p = 251g$ is the scenario mainly considered in this thesis, the found value for v_p for that case will be used.

[22]

Aerodynamic Drag F

In order to verify that effects from aerodynamic drag on the pendulum are negligible, an approximate torque produced by drag at various angular velocities will be presented in this appendix.

For a linear motion with v through air with density ρ, the force exerted on the object in motion is given by:

$$F_d = \frac{1}{2}\rho C_d S v^2 \tag{F.1}$$

where C_d is the drag coefficient determined by the shape of the object and S is the area projected towards the direction of motion.

Using the relation between linear and angular velocity, $v = \omega l$ and considering both the pendulum rod and tip-mass as cylinders ($S = w \cdot l$), **equation (F.1)** becomes:

$$F_d = \frac{1}{2}\rho C_d w(\omega l)^2 \tag{F.2}$$

$$= \frac{1}{2}\rho C_d w \omega^2 l^3 \tag{F.3}$$

The differential of the aerodynamic drag is then:

$$\frac{dF_d}{dl} = \frac{1}{2}\rho C_d w \omega^2 \frac{d\, l^3}{dl} \tag{F.4}$$

$$= \frac{1}{2}\rho C_d w \omega^2 3l^2 \tag{F.5}$$

and thus

$$dF_d = \frac{1}{2}\rho C_d w \omega^2 3l^2 dl \tag{F.6}$$

The torque exerted by the aerodynamic drag in the pendulum joint is then found by:

$$\tau_d = \int_{l_{min}}^{l_{max}} l \cdot dF_d \tag{F.7}$$

$$= \int_{l_{min}}^{l_{max}} \frac{3}{2}\rho C_d w \omega^2 l^3 dl \tag{F.8}$$

$$= \frac{3}{2}\rho C_d w \omega^2 \int_{l_{min}}^{l_{max}} l^3 dl \tag{F.9}$$

$$= \frac{3}{2}\rho C_d w \omega^2 \left[\frac{1}{4}l^4\right]_{l=l_{min}}^{l_{max}} \tag{F.10}$$

$$\tag{F.11}$$

For the rod with integral limits 0 and l_{rod}, the torque is evaluated to:

$$\tau_{d,rod} = \frac{3}{8}\rho C_d w \omega^2 l_{rod}^4 \tag{F.12}$$

and for the end-mass with limits l_{rod} and $l_{rod} + l_{mass}$

$$\tau_{d,mass} = \frac{3}{8}\rho C_d w \omega^2 \left((l_{rod} + l_{mass})^4 - l_{rod}^4 \right) \tag{F.13}$$

Finally, the total torque produced by aerodynamic drag is:

$$\tau_{d,total} = \tau_{d,rod} + \tau_{d,mass} \tag{F.14}$$

An evaluation of **equations (F.12)** to **(F.14)**, with $C_d = 0.96$, $\rho = 1.2041\ \frac{\text{kg}}{\text{m}^3}$ and the remaining parameters as given in **table 2.1**, is shown in **figure F.1**.

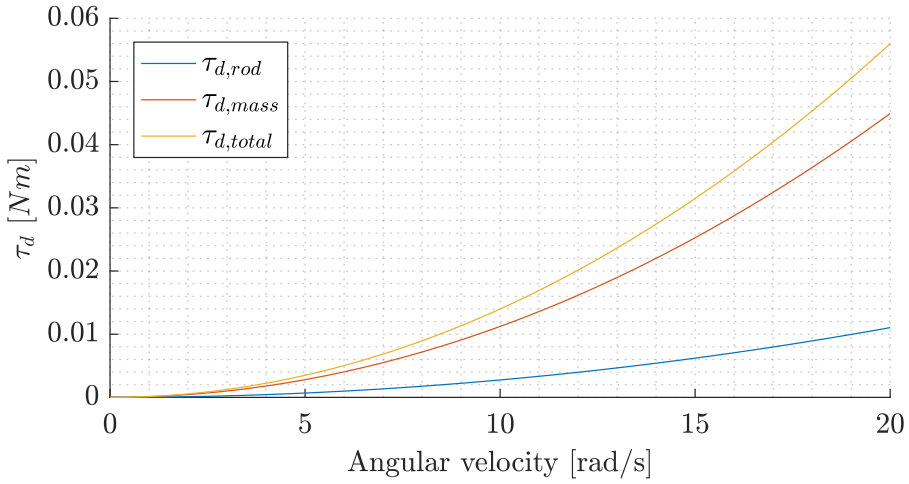


Figure F.1: *Torque produced by aerodynamic drag on the pendulum.*

For comparison, the torque produced by gravity at $\theta = \frac{\pi}{2}$ is:

$$\tau_g = (l_{rod} + 0.5 l_{mass}) m_p g \tag{F.15}$$

$$= 0.8236 \text{ Nm} \tag{F.16}$$

[23]

Determination of Measurement Noise G

This appendix serves as documentation of the measurement noise. The theory behind this test is, that when maintaining a constant output, then, if any measurement noise exists, it would be visible when subtracting the mean of the output from itself.

A simple test was conducted where the cart was manually moved into the middle of the rail, and then allowed to settle. This resulted in the following results

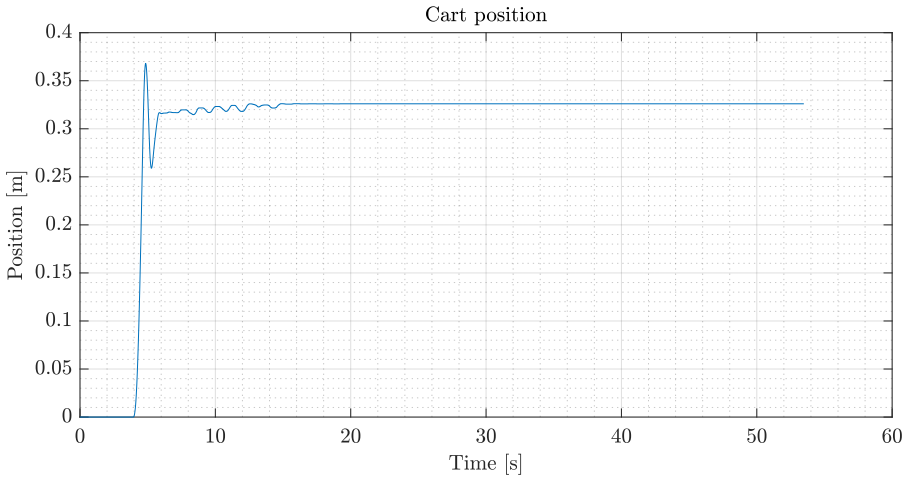


Figure G.1: *Cart position*

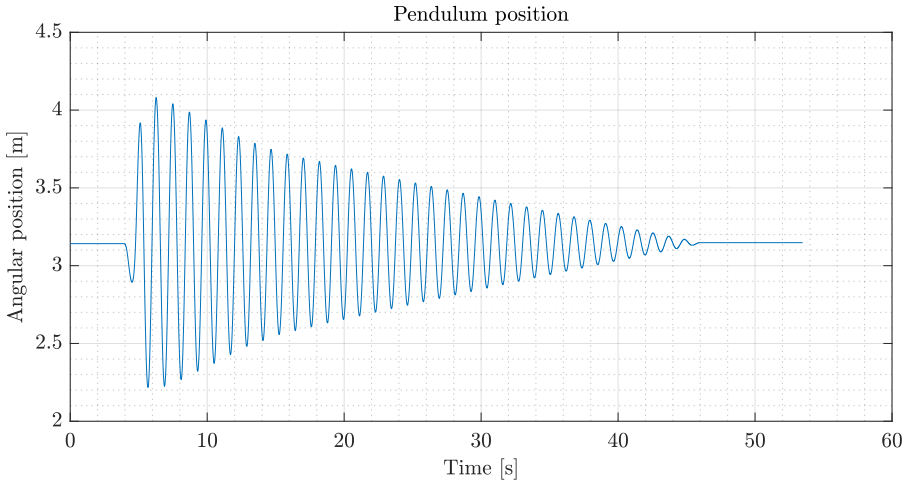


Figure G.2: *Pendulum position*

As it can be seen on both figures, when the system has settled, there is no deviation on the output. Subtracting the mean is therefore omitted. Since there is no deviation on the output at the end of both measurements it can be concluded that the measurement noise is smaller than the quantization, and in effect the measurement noise matrix R_k can be set to zero.

Initialisation Error Test H

The purpose of this appendix is to document the outcome when the pendulum is not carefully initialised. Upon initialisation, before activating the control, the cart must be taken to the leftmost position on the rail and the pendulum must be hanging exactly downwards.

Due to the presence of Coulomb friction in the pendulum joint, although small, the pendulum may not be hanging straight down when initialised (i.e. θ is reset to π). This will result in a response as shown in **figure H.1**.

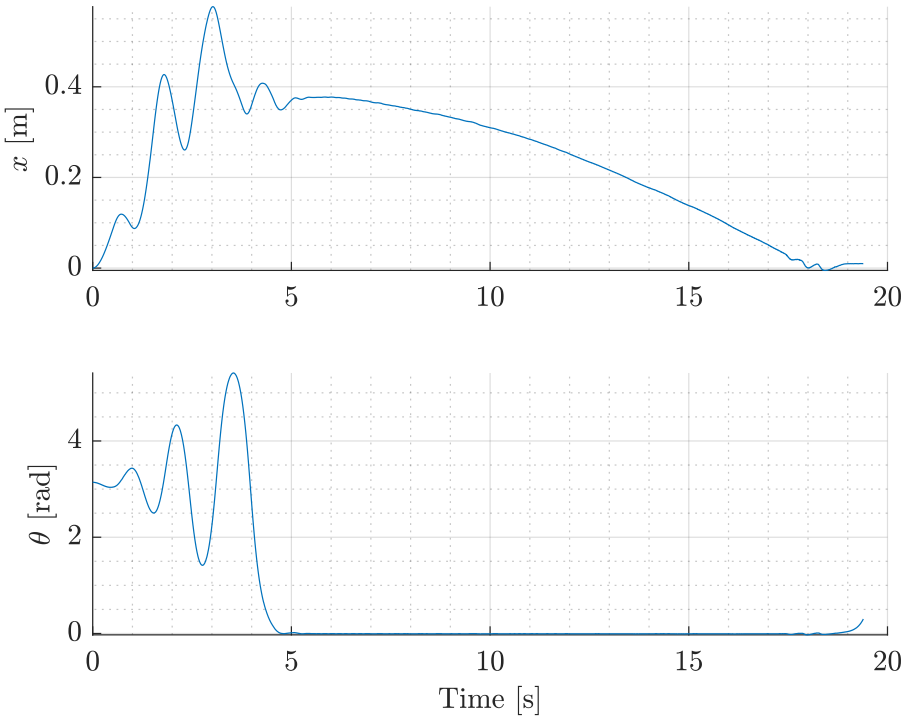


Figure H.1: *Outcome of swing-up and stabilisation when pendulum is not properly initialised.*

It is seen that the swing-up and catch is executed successfully, but some error persists, causing the cart to drift until it hits the end of the rail.

Carefully flicking the pendulum or pushing the table on which it is mounted tends to help prevent this.