SC42035 Integration Project

Container Crane

19 June 2019

TU Delft
Coordinated by Sander Bregman, Jens Kober and Jan-Willem van Wingerden

Matthijs Bekendam 4725751 J.M.Bekendam@student.tudelft.nl

Evan Remmerswaal 4387872 J.W.G.Remmerswaal@student.tudelft.nl

Contents

1 Introduction

This report is a summary of the controller design for SC42035 Integration Project, performed in the last quarter of 2019 at Delft University of Technology. The report is built up in such a way that each section builds upon the previous one, taking the reader through every consecutive step taken by the team to get to the final design.

All code used to generate the results and control the setup is given in a separate .zip-file. Note that not all code is our intellectual property, some of it was already written to start the project with but edited by us to serve our project best, some essential snippets were provided to get us started with the project and some was written completely from scratch by us.

Section 2 takes the reader through the derivation of the dynamics of the system step-by-step. Using Euler-Lagrange the equations of motion will be formulated and the system will be linearised and decoupled. Section 3 shows how the sensors in the setup were properly calibrated. Section 4 shows how unknown parameters were estimated using a Greybox model. Section 5 denotes the observer design and why a Kalman Filter was implemented. Section 6 is about the two controllers designed for this project. Section 7 will denote the achieved results and modification applied to improve on these results. In section 8 we will conclude on our project. Section 9 will be a reflection on our work and on future recommendations.
We introduce the Container Crane setup and the control objective below.

1.1 The setup

An schematic drawing of the container crane setup can be found in Figure 1. A trolley with mass M is driven by a DC motor with input u such that it travels in the lateral direction on a rail approximately 3 meters long. The trolley carries a container of mass m. The container is attached to the trolley by a cable of variable length l. A second DC motor drives the hoist system. However due to time constraints, we only actuate the DC motor that drives the trolley, keeping the cable length constant.

The system has one control input u, which is the current that runs trough the DC motor. The system has an internal control loop that actively regulates this current, due to time constraints this will be ignored in the model dynamics. This input u is commanded from the computer and is scaled between -1 and +1. There are two measured outputs: x– the position of the trolley and θ– the angle between the cable and the vertical.

1.2 Control Objective

We will be designing an anti-sway controller for the container crane setup assuming constant cable length. The controlled system should be able to quickly follow a reference for the trolley position (given by the joystick or using a predefined signal), while minimizing the container swing. The team will be aiming for the below denoted control design specifications:

- A reference point should be reached as quickly as possible within the operation region of the controller and using realistic inputs.
- The swing of the container during acceleration of the trolley should be kept below 0.5 radians.
- After reaching the reference point the swing of the container should decay to zero within 2 seconds.

2 Dynamics

In this section the dynamics behind the system will be derived using Euler Lagrange (EL) [1]. In the first part of this section, we will formulate the EL problem. Secondly, the EL will be worked out for our system and placed into a familiar and convenient state space form. Thirdly, the state space model will be linearised and decoupled.

2.1 Euler Lagrange

To derive the dynamics of the container crane, an Euler Lagrange with dissipative term and exogenous input has been applied to the system.

$$\frac{d}{dt}(\frac{\delta L}{\delta \dot{q}}) - \frac{\delta L}{\delta q} + \frac{\delta D}{\delta \dot{q}} = Q \tag{2.1}$$

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix}$$

Where L consists of:

$$L = \underbrace{T}_{\text{Kinetic Energy}} - \underbrace{V}_{\text{Potential Energy}} . \tag{2.2}$$

and D denotes the dissipative energy of the system:

$$D = \underbrace{\frac{1}{2} b_1 \dot{x}^2}_{\text{Trolley}} + \underbrace{\frac{1}{2} b_2 \dot{\theta}^2}_{\text{Container}} . \tag{2.3}$$

Q is the exogenous input into the system:

$$Q = \begin{bmatrix} F \\ 0 \end{bmatrix} \tag{2.4}$$

and for our setup can be approximated in the two manners. The first approach equates F to a motor constant (g_m $[\frac{T \cdot m}{A}]$) multiplied with the input. This is done since F scales linearly with u.

$$F = g_m \cdot u \tag{2.5}$$

The second, approach is a bit more advanced and takes into account the internal closed loop, from which we obtain:

$$T = k_m \frac{v_{in} - v_{emf}}{R} \tag{2.6}$$

$$v_{in} = k_p(u - \dot{x}) + k_d \ddot{x} + \int_t^0 k_i(u - \dot{x}) d\tau$$

$$V_{emf} = k_m \cdot \dot{x}$$

$$F = T \cdot r$$

In the following sections, we will start working with the former for simplicity.

2.2 Derivation Euler Lagrange

Going back to Equation 2.2, the kinetic energy and potential energy can be derived from the simple schematic denoted in Figure 1

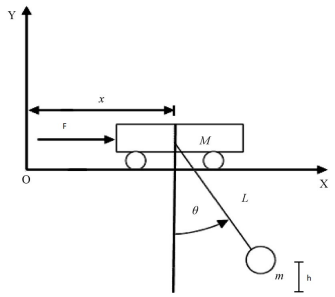


Figure 1: Illustration of the Container Crane system, containing a trolley and a pendulum.

The kinetic energy is obtained through:

$$T = \underbrace{\frac{1}{2}M\dot{x}^2}_{Trolley} + \underbrace{\frac{1}{2}m\Big((\dot{x} + l\dot{\theta}\cos\theta)^2 + (l\dot{\theta}\sin\theta)^2\Big)}_{Container} \tag{2.7}$$

$$T = \frac{1}{2}(M+m)\dot{x}^2 + \frac{1}{2}m(2\dot{x}\dot{\theta}l\cos\theta + l^2\dot{\theta}) \tag{2.8}$$

The potential energy is obtained through:

$$V = \underbrace{Mg0}_{Trolley} + \underbrace{Mgh}_{Container} = Mgh \tag{2.9}$$

Where the coordinate system is chosen such that the height of the trolley is zero. Hence, the potential energy is determined only by the container.

$$V = mgl(1 - \cos\theta) \tag{2.10}$$

Putting V and T together forms L:

$$L = \frac{1}{2}(M+m)\dot{x}^2 + \frac{1}{2}m(2\dot{x}\dot{\theta}l\cos\theta + l^2\dot{\theta}^2) - mgl(1-\cos\theta) \tag{2.11}$$

2.2.1 Differentiating L and D

Now that L and D are determined, we perform the appropriate differentiation's according to Equation 2.1.

$$\frac{\delta L}{\delta q} = \begin{bmatrix} \frac{\delta L}{\delta x} \\ \\ \frac{\delta L}{\delta \theta} \end{bmatrix} = \begin{bmatrix} 0 \\ \\ -m\dot{x}\dot{\theta}l\sin\theta - mgl\sin\theta \end{bmatrix}$$

$$\frac{\delta L}{\delta \dot{q}} = \begin{bmatrix} \frac{\delta L}{\delta \dot{x}} \\ \\ \frac{\delta L}{\delta \dot{\theta}} \end{bmatrix} = \begin{bmatrix} (M+m)\dot{x} + m\dot{\theta}l\cos\theta \\ \\ m\dot{x}l\cos\theta + ml^2\dot{\theta} \end{bmatrix}$$

$$\frac{\delta D}{\delta \dot{q}} = \begin{bmatrix} \frac{\delta D}{\delta \dot{x}} \\ \\ \frac{\delta D}{\delta \dot{\theta}} \end{bmatrix} = \begin{bmatrix} b_1\dot{x} \\ \\ b_2\dot{\theta} \end{bmatrix}$$

$$\frac{d}{dt}(\frac{\delta L}{\delta \dot{q}}) = \begin{bmatrix} (M+m)\ddot{x} + m\ddot{\theta}l\cos\theta - m\dot{\theta}^2\sin\theta l \\ \\ m\ddot{x}l\cos\theta - m\dot{x}l\sin\theta\dot{\theta} + ml^2\ddot{\theta} \end{bmatrix}$$

Inserting these into Equation 2.1 yields:

$$\begin{bmatrix} (M+m)\ddot{x} + m\ddot{\theta}l\cos\theta - m\dot{\theta}^2\sin\theta l + b_1\dot{x} \\ \\ m\ddot{x}l\cos\theta + ml^2\ddot{\theta} + mgl\sin\theta \end{bmatrix} = \begin{bmatrix} g_m u \\ \\ 0 \end{bmatrix} \tag{2.12}$$

From Equation 2.12 we can extract and rewrite for $\ddot{x}$ and $\ddot{\theta}$, which describe the nonlinear dynamics of the system.

$$\ddot{x} = \frac{ml\dot{\theta}\sin\theta + mg\frac{1}{2}\sin 2\theta + \cos\theta b_2\dot{\theta}\frac{1}{l} - b_1\dot{x} + g_m}{M + m(1-(\cos\theta)^2)} \tag{2.13}$$

$$\ddot{\theta} = \frac{-ml\dot{\theta}^2\frac{1}{2}\sin 2\theta - mg\frac{1}{2}\sin 2\theta\cos\theta - \cos\theta^2 b_2\dot{\theta}\frac{1}{l} + \cos\theta b_1\dot{x} - \cos\theta g_m u}{l\big(M + m(1-(\cos\theta)^2)\big)} - \frac{mgl\sin\theta + b_2\dot{\theta}}{l^2 m} \tag{2.14}$$

2.3 Linearization

The linearization will be done according to [2]. First we will create the Jacobian, secondly we insert the equilibrium points $x = 0$ and $\theta = 0$ into the Jacobian to retrieve an 'A' and 'B' matrix. We will denote the dynamics of our system in the follow state-space form:

$$f(q) = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \\ \dot{q_4} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \tag{2.15}$$

According to [2], the general Jacobian form is:

$$Jf\left(q_1, q_2, q_3, q_4\right) = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \frac{\partial f_1}{\partial q_3} & \frac{\partial f_1}{\partial q_4} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} & \frac{\partial f_2}{\partial q_3} & \frac{\partial f_2}{\partial q_4} \\ \frac{\partial f_3}{\partial q_1} & \frac{\partial f_3}{\partial q_2} & \frac{\partial f_3}{\partial q_3} & \frac{\partial f_3}{\partial q_3} \\ \frac{\partial f_4}{\partial q_1} & \frac{\partial f_4}{\partial q_2} & \frac{\partial f_4}{\partial q_3} & \frac{\partial f_4}{\partial q_4} \end{bmatrix} \tag{2.16}$$

The f_1- and f_3 row are straight forward easy computations, the f_2- and f_4 row however require some more extensive calculations, which are shown in the next subsections.

2.3.1 Calculation of the Jacobian f_2-row

For convenience, we will denote Equation 2.13 ($\ddot{x}$) as $f_2 = \frac{h}{p}$

Deriving Jacobian for f_2

$$\frac{\partial f_2}{\partial q_1} = 0 \qquad \frac{\partial f_2}{\partial q_2} = \frac{-b_1}{M + m(1-(\cos\theta)^2)}$$

$$\frac{\partial f_2}{\partial q_3} = \frac{h\dot{p} - \dot{h}p}{p^2} \qquad \frac{\partial f_2}{\partial q_4} = \frac{2ml\dot{\theta}\sin\theta + \cos\theta b_2\frac{1}{l}}{M + m(1-(\cos\theta)^2)}$$

$\frac{\partial f_2}{\partial q_3}$ was split up into three parts due to its complexity and was solved using the coefficient rule.

$$h\dot{p} = (ml\dot{\theta}^2\sin\theta + \frac{1}{2}mg\sin 2\theta + \cos\theta b_2\dot{\theta}\frac{1}{l})(2m\cos\theta\sin\theta)$$

$$\dot{h}p = (ml\dot{\theta}^2\cos\theta + mg\cos 2\theta - \sin\theta b_2\dot{\theta}\frac{1}{l})(M + m(1-(\cos\theta)^2))$$

$$p^2 = (M + m(1-\cos\theta))^2$$

2.3.2 Calculation of the Jacobian f_4-row

For convenience, we will denote Equation 2.14 ($\ddot{\theta}$) as:

$$f_4 = \frac{h}{p} - \frac{mgl\cos\theta}{l^2 m} \tag{2.17}$$

Deriving Jacobian for f_4

$$\frac{\partial f_4}{\partial q_1} = 0 \qquad \frac{\partial f_4}{\partial q_2} = \frac{\cos\theta b_1}{l(M + m(1-(\cos\theta)^2))}$$
$$\frac{\partial f_4}{\partial q_3} = \frac{h\dot{p} - \dot{h}p}{p^2} + \frac{mgl\sin\theta}{l^2 m} \qquad \frac{\partial f_4}{\partial q_4} = \frac{-ml\dot{\theta}\sin 2\theta - (\cos\theta)^2 b_2 \frac{1}{l}}{l\big(M + m(1-(\cos\theta)^2)\big)} - \frac{b_2}{l^2 m}$$

Where $\frac{\partial f_4}{\partial q_3}$ consists of the following three parts:

$$h\dot{p} = h \cdot (2ml\cos\theta\sin\theta)$$
$$\dot{h}p = \big(-ml\dot{\theta}^2\cos 2\theta + \frac{1}{2}mg\sin 2\theta\sin\theta - mg\cos 2\theta\cos\theta + 2\cos\theta\sin\theta b_2\dot{\theta}\frac{1}{l} - \sin\theta b_1\dot{x}\big)\cdot p$$
$$p^2 = \Big(l\big(M + m(1-(\cos\theta)^2)\big)\Big)^2$$

2.3.3 Linearization around equilibrium

From [2] we know that inserting the results from the previous sections and the equilibrium points into the Jacobian matrix (Equation 2.16) yields a linearised state space representation of the following:

$$A := \left.\frac{\partial f}{\partial q}\right|_{\substack{x=0\\u=0}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-b_1}{M} & \frac{mg}{M} & \frac{b_2}{Ml} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{b_1}{lM} & \frac{-g(m+M)}{Ml} & \frac{-b_2(m+M)}{l^2 mM} \end{bmatrix} \tag{2.18}$$

$$B := \left.\frac{\partial f}{\partial u}\right|_{\substack{x=0\\u=0}} = \begin{bmatrix} 0 \\ \frac{g}{M} \\ 0 \\ \frac{-g}{Ml} \end{bmatrix} \tag{2.19}$$

2.4 Decoupling dynamics

To create a better parameter estimation we will decouple the system in two parts: a trolley part and a container/pendulum part.

2.4.1 Equations of motion of the trolley

The trolley only has a kinetic energy, dissipative energy and an exogenous input. Looking back at Section 2.1, we can create a new EL using the trolley dynamics only.

Euler Lagrange Trolley system

$$\frac{\delta L}{\delta x} = 0 \qquad \frac{\delta L}{\delta \dot{x}} = M\dot{x}$$
$$\frac{d}{dt}(\frac{\delta L}{\delta \dot{x}}) = M\ddot{x} \qquad \frac{\delta D}{\delta \dot{x}} = b_1\dot{x}$$

Inserting these into Equation 2.1 yields

$$M\ddot{x} + b_1\dot{x} = g_m u \tag{2.20}$$

Rewriting for $\ddot{x}$ gives

$$\ddot{x} = -\frac{b_1}{M}\dot{x} + \frac{g_m}{M}u \tag{2.21}$$

Formulating the state space capturing the dynamics:

$$f(q,u) = \begin{bmatrix} \dot{x} \\ -\frac{b_1}{M}\dot{x} + \frac{g_m}{M}u \end{bmatrix} \qquad q = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{2.22}$$

Linearizing the state space model (2.22) gives:

$$A := \left.\frac{\partial f(q,u)}{\partial q}\right|_{q(0,0)} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{b_1}{M} \end{bmatrix} \tag{2.23}$$

$$B := \left.\frac{\partial f(q,u)}{\partial u}\right|_{q(0,0)} = \begin{bmatrix} 0 \\ \frac{g_m}{M} \end{bmatrix} \tag{2.24}$$

2.4.2 Equations of motion of the container

Again, we will refer back to Section 2.1 to construct a new EL formulating for the decoupled container. In contrast to the trolley, the container or pendulum has no exogenous input. As such, the EL depends only on the kinetic energy, potential energy and dissipative energy. Illustration 2 was used to formulate the equations of energy.

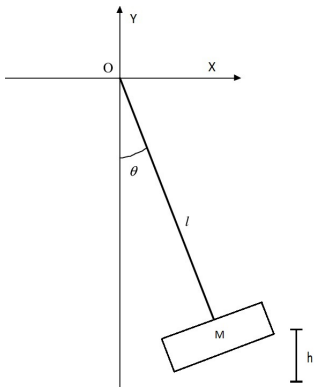


Figure 2: Illustration of the decoupled container from the Container Crane system.

Starting with the kinetic energy:

$$T = \frac{1}{2} m \underbrace{\dot{\theta}^2 l^2}_{v^2} \tag{2.25}$$

v^2 formulation

$$x = l\cos\theta \qquad y = \sin\theta$$
$$\dot{x} = -l\sin\theta\dot{\theta} \qquad \dot{y} = l\cos\theta\dot{\theta}$$
$$v^2 = x^2 + y^2 \qquad v^2 = \dot{\theta}^2 l^2 \underbrace{((\cos\theta)^2 + (\sin\theta)^2)}_{1}$$

The potential energy remains unchanged from the original system and is displayed in Section 2.1 as Equation 2.10. With the potential energy and kinetic energy, the L can be constructed for the EL formulation.

$$L = T - V = \frac{1}{2} m\dot{\theta}^2 l^2 - mgl(1 - \cos\theta)$$

The dissipative energy equals:

$$D = \frac{1}{2} b_2 \dot{\theta}^2$$

And formulating the EU:

Euler Lagrange Pendulum system

$$\frac{\delta L}{\delta\theta} = -mgl\sin\theta \qquad \frac{\delta L}{\delta\dot{\theta}} = ml^2\dot{\theta}$$
$$\frac{d}{dt}(\frac{\delta L}{\delta\dot{\theta}}) = ml^2\ddot{\theta} \qquad \frac{\delta D}{\delta\dot{\theta}} = b_2\dot{\theta}$$

Inserting these into Equation 2.1 gives:

$$ml^2\ddot{\theta} + mlg\sin\theta + b_2\dot{\theta} = 0$$

Rewriting for $\ddot{\theta}$

$$\ddot{\theta} = -\frac{g}{l}\sin\theta - \frac{b_2}{ml^2}\dot{\theta} \tag{2.26}$$

Formulating the state space capturing the dynamics:

$$f(q) = \begin{bmatrix} \dot{\theta} \\ -\frac{g}{l}\sin\theta - \frac{b_2}{ml^2}\dot{\theta} \end{bmatrix} \qquad q = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{2.27}$$

If we linearize we obtain the following results.

$$A := \left.\frac{\partial f(q)}{\partial q}\right|_{q(0)} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\frac{b_2}{Mml^2} \end{bmatrix} \tag{2.28}$$

Since the system contains no exogenous inputs

$$B = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.29}$$

3 Sensor Calibration

Now that we have derived a model for our system, the sensors of the system should be calibrated such that we can gather data and start working with the setup.

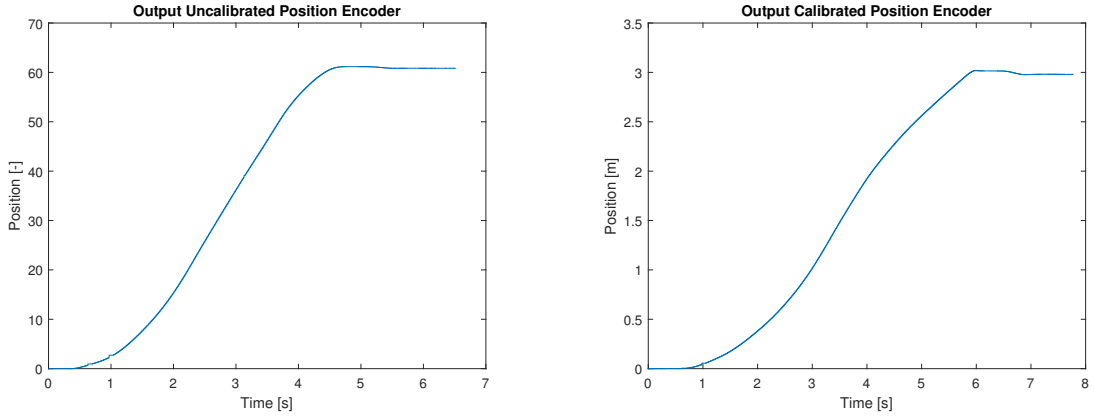
The Container Crane setup has three encoders (sensors) for generating digital signals in response to mechanical motion. One encoder for the motion of the trolley and two for the angular displacement of the pendulum (container). The digital pulse train generated by these encoders have to be translated (calibrated) into standard SI units of measurement. This can be done by applying an addition gain or an additive offset to the final signal output.

3.1 Calibration Trolley Encoder

The units of measurement of interested for the Trolley are in Meters. Initially, the encoder would give a total displacement of 60 when the trolley moved along the total length of its track. In reality, the track is only 3 meters long. The existing encoder gain had to be changed to:

$$\text{Calibrated Gain} = \frac{\text{Default Gain}}{60} \cdot 3 \tag{3.1}$$

Figure 5 shows the effect on the output when the encoder is calibrated.



(a) Uncalibrated Trolley Encoder

(b) Calibrated Trolley Encoder

Figure 3: Calibrated- and uncalibrated trolley encoder. The figures show a trolley trajectory going along the entire track of approximately 3 meters.

3.2 Calibration Pendulum Encoder

The units of measurement of interested for the pendulum are in Radians. The total angular displacement of the pendulum was initially determined by the difference in angular displacement of both ends of the container. Placing the container at an angle of 30 degrees (measured with a drafting triangle) would give values around 0.23 radians, where ≈ 0.5 radians is to be expected. After trying to solve this calibration error by applying a gain (like in Section 3.1), there was still no correct response observed by the encoder. The cause of the problem was found when looking at both angles separately (see Figure 4). One of the encoders saturates at -1. Having no success removing this saturation from the encoder it was decided to not use this sensor and work only with the secondary encoder, applying twice the gain onto it.

$$\text{Calibrated Gain} = -2 \cdot \text{Default Gain} \cdot \frac{0.517}{0.44} \tag{3.2}$$

The -2 is due to the turning one sensor off, the fraction is the value the sensor should put out divided by the value of the angle which is currently being displayed. Furthermore, the encoder initially had an offset of 0.02

radians whenever the container was stationary. This was remedied by a simple addition of -0.02 radians to the signal.

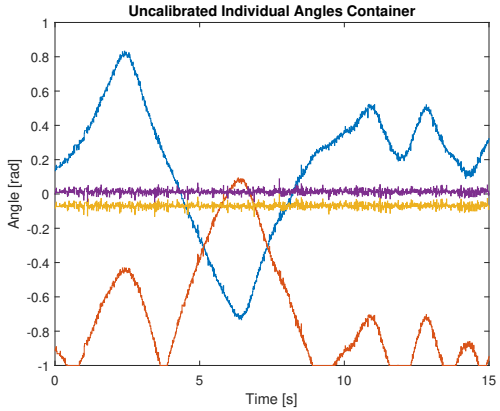
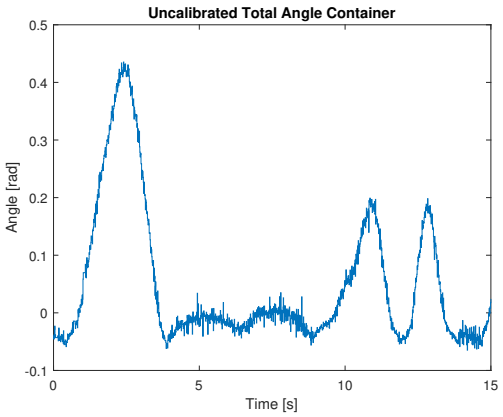
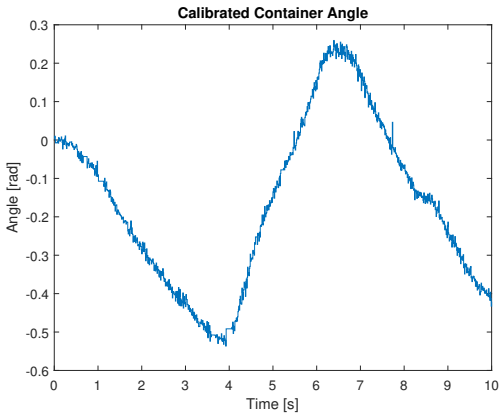


Figure 4: Uncalibrated angles of the container. One encoder (orange) saturates at -1 and flips direction. This particular encoder was turned off.



(a) Uncalibrated Trolley (total) Angle



(b) Calibrated Trolley (total) Angle

Figure 5: Calibrated- and uncalibrated container angles. The figures show the container being forced into an angle of 30 degrees (0.52 rad). (a) shows a general offset of approximately -0.2 and a deviation of approximately 0.1 radians from the 30 degree mark. (b) shows the calibrated angle, having a negligible offset and goes to the 30 degree mark.

4 System Identification

Before we can implement a controller, we need to estimate some unknown constant parameters from the setup. This is done by an optimisation routine which uses our decoupled models derived in chapter 2 and measured data. Measured data is now available since we calibrated the system in Chapter 3.

In the first part of this section we will show the method used for our system identification. A deliberate choice is made to use 'Greybox estimation', Greybox estimation allows for accurate parameter estimation. In the second part we will discuss our parameter estimation results and its performance will be evaluated.

4.1 Greybox Estimation

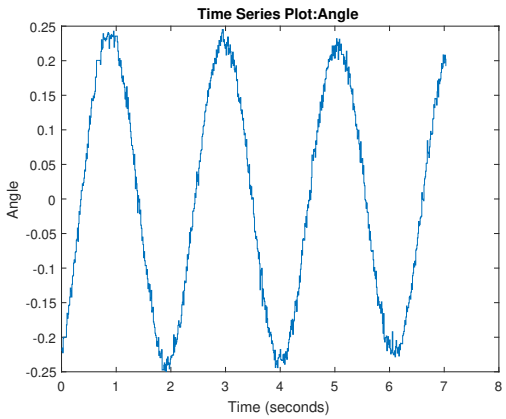
In this subsection we will give a broad overview as to how Greybox estimation works. Grey box estimation is one among many system modeling approaches. One should use Greybox estimation when the equations for the system are known and measured data is available. Related to the Greybox is the Black box estimation, which was also considered for our system identification. However, Black box models only use empirical data to estimate the entire model. Since the equations could all be derived (Section 2), Greybox was the obvious choice and proved to be very accurate.

Parameters are estimated through a prediction error minimization method. It tries to enforce minimum error between the actual data and the predicted curve by adjusting the unknown parameters in the model. The parameter values which yield the best fit on the empirical data, are chosen. The method used for this optimization process was Levenberg-Marquardt least squares search. This is the most efficient optimizer when the Hessian and Δf are easy to compute. Which for our decoupled systems is definitely the case.

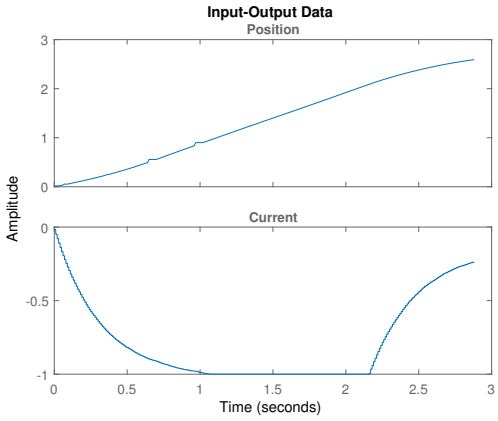
4.2 Estimation results

In this subsection we will discuss the results obtained from the Greybox estimation. To simplify the estimation, the system was decoupled into two subsystems as discussed in Section 2.4.

The motor and trolley damping (b_1) and motor constant (g_m) can be estimated by using the decoupled trolley system. The cable hinge damping (b_2) and the length of the cable (l) can be determined by the decoupled container system. Figures 6-10 show the estimation results of the decoupled container system (left) and the decoupled trolley system (right). Table 1 shows the final estimated parameter values and the corresponding initial guesses (required for the search method). For gathering the measured data we used minimal excitation of the system (see Figure 6). This represents our operating region and will yield the best estimations for our design.



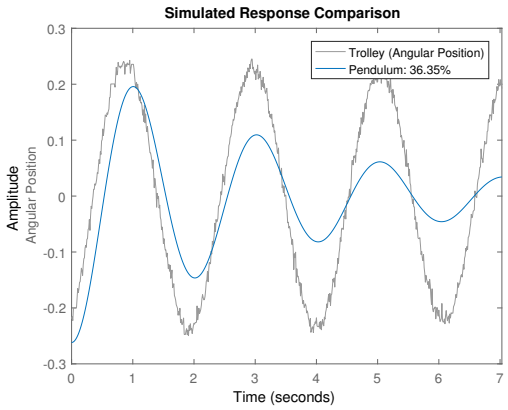
(a) Measured angle (rad) of container vs time. Initial angle of 15 degrees and no input.



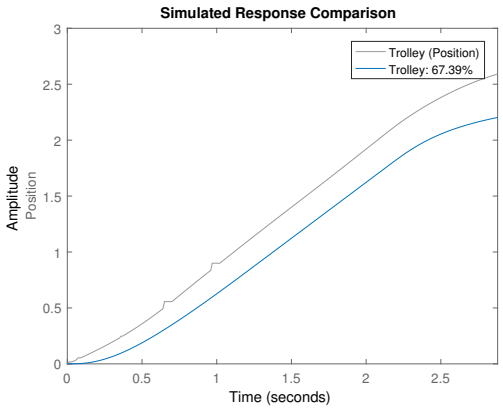
(b) Measured position (m) of trolley and input vs time. Trolley going from the start of the track till end.

Figure 6: Gathering measurements for parameter estimation.

Before estimating all the parameters, the output of the system is simulated using the initial guesses for the parameters.

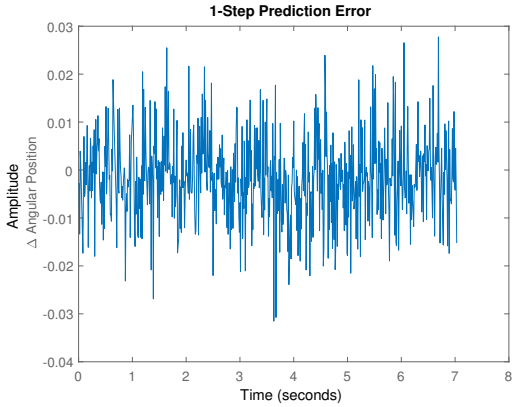


(a)

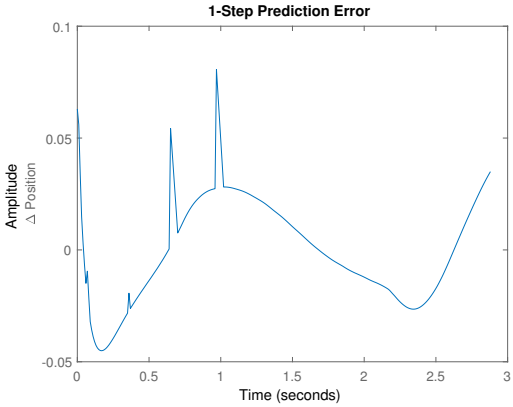


(b)

Figure 7: Simulated response using initial guessed parameter values. The simulation yields a better fit for the trolley system than for the container system. It seems like the initial guess for the cable hinge damping is a bit to high since the oscillations in the simulation more rapidly decreases. Furthermore, it seems like either the motor constant is a bit too low or the motor and trolley damping is a bit too high.

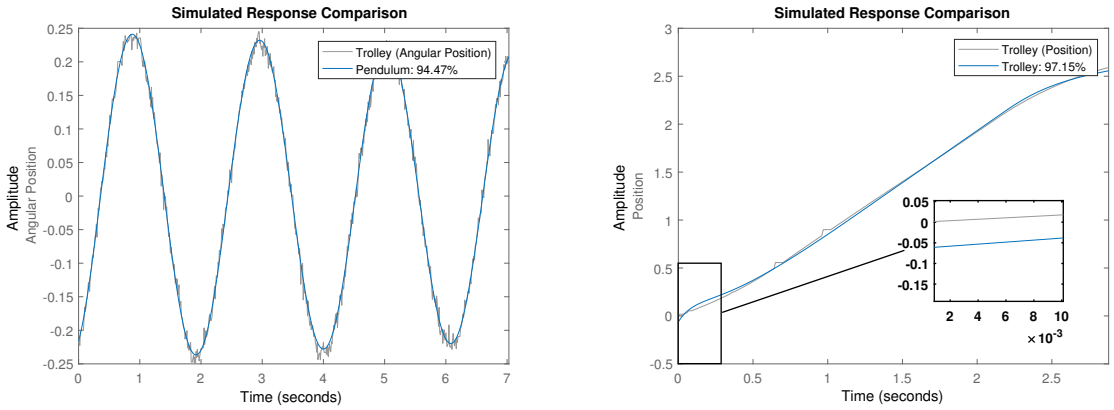


(a) The prediction errors obtained are small and are centered around zero (non-biased).



(b) The prediction error of the trolley shows some undesired features due to an optimisation artifact and a encoder defect (two spikes) in the trajectory of the trolley (see Figure 9b). The error seems to be centred around zero and has no bias.

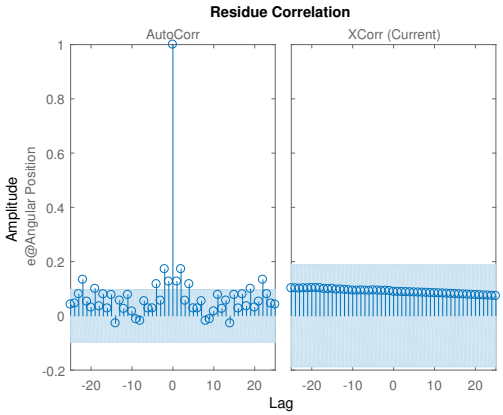
Figure 8: 1-step prediction errors. The two spikes in (b) are due to the sensor failing to output a signal. This is not uncommon and the amount of information lost is tracked using ticks. When 500 ticks are registered the simulation stops. However, when less than 500 consecutive ticks are registered the simulation keeps going. The lost information is filled in with a step-like line, hence the spikes.



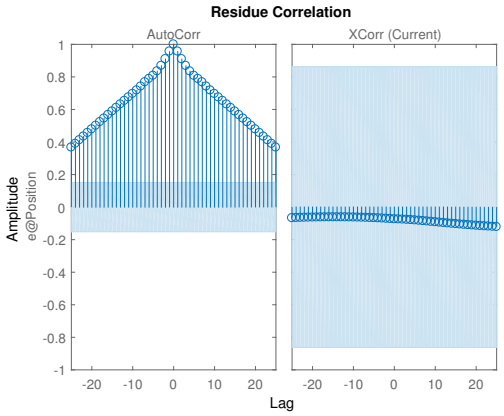
(a) (b)

Figure 9: Simulated response using optimal parameter values. Both estimation shows sufficient results to be confident about the correctness of the parameter values. Although the trolley has a better total fit, its error is less consistent than the pendulum. This is due to the two spikes in the trajectory (disturbance) and do to an optimization artifact shown in the magnified section of (b). For an unknown reason the initial state would not start at 0 but at -0.06.

Residuals indicate what is left unexplained by the model and are small for good model quality. The first column of plots shows the auto correlations of the residuals for the two outputs. The second column shows the cross-correlation of these residuals with the input "Ampere". The correlations are within acceptable bounds (blue region) [5].



(a) Since almost all residuals are within the confidence interval, the decoupled pendulum model is of good quality. This was expected since the container was easy to decouple from the trolley (using our hands we could keep the trolley stationary).The cross-correlation of this particular plot is meaningless since there was no actual input. The system on default always registers a DC, its traces are shown in this plot.



(b) The large residuals in the auto correlation of the trolley means there is still information left uncaptured. This was to be expected since it was not possible to decouple the system properly. Due to setup restrictions we were unable to detach the container from the trolley. To negate the effects of the container on the trolley as much as possible, the container was moved to its highest vertical position. Although this approach certainly helps, it is sub-optimal. The effect described in Figure 8 probably will also effect the residuals.

Figure 10: Correlogram: auto correlation and cross correlation. The residuals are the differences between the fitted model and the data. In essence it shows how an error at one point in time travels to a subsequent point in time. If the auto-correlation sequence of the residuals looks like the auto-correlation of a white noise process, you are confident that none of the signal has escaped your fit and ended up in the residuals [8]. This can be checked through a confidence interval (the blue boundary in the plots). If the residuals lie within these bounds, we are confident that the residuals are white noise. For both (a) and (b) the cross-correlation lies within the bounds. This is an indication that the model properly describes how part of the output relates to the corresponding input.

		Estimated Value	**Initial Guessed Value**	**Units**
1	b_1	10.8671	10	$[\frac{kg}{s}]$
2	b_2	0.05415	0.75	$[\frac{kg}{s}]$
3	g_m	-11.7944	-10	$[\frac{F}{A}]$
4	l	1.0763	1	[m]

Table 1: Estimated parameter values and initial guess. Only b_2, the cable hinge damping, differs significantly from its initial guessed value.

5 Observer

Since there is no full-state information, our design requires the use of an observer.

In this section we will discuss the use of an observer in our system and why we extended this to a Kalman Filter. First, we will talk about the Luenberger Observer and review its results. Next, we talk about the Kalman Filter and compare its results to the Luenberger Observer.

5.1 Luenberger Observer

A Luenberger Observer, or just observer, is a filter which approximates the state vector $\hat{x}$ of a system from the measurements of the input and output sequences.

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L(y(k) - \hat{y}(k)) \qquad \hat{y}(k) = C\hat{x}(k)$$

If the initial state $\hat{x}(0)$ equals $x(0)$, then the state sequences $\hat{x}(k)$ and $x(k)$ will be equal. If the initial conditions are not the same and the matrix $A-LC$ is Hurwitz, $x(k)$ and $\hat{x}(k)$ will converge after a period of time.

The position of the eigenvalues of the A-LC matrix determine the speed of convergence between $\hat{x}(k)$ and $x(k)$. Placing the eigenvalues closer to the origin will result in a smaller interval in which the true system and the observer differ. When there is no noise in the measurements and the true system model is known, one could suggest placing the eigenvalues at zero. This is also known as a dead-beat observer. In practise however, the measurements almost always contain noise so this will most often not be a solution.

To be able to place these eigenvalues freely (using the observer gain L), the matrix pair (A,C) has to be observable. This can be checked via MATLAB and is the case for our system [2]. The observer poles should be placed such that they are a bit faster than the state feed-back poles. This way, the error converges faster to zero than the system can react to any dynamics. This means placing the observer poles such that they lie closer to the origin (in the z-domain) than the state feedback poles. The observer poles were given the following location:

$$[\text{Discrete Observer Poles }] = \frac{2}{3} \cdot [\text{Discrete Feedback Poles}]$$

More information on discretization and state feedback can be found in Section 6.

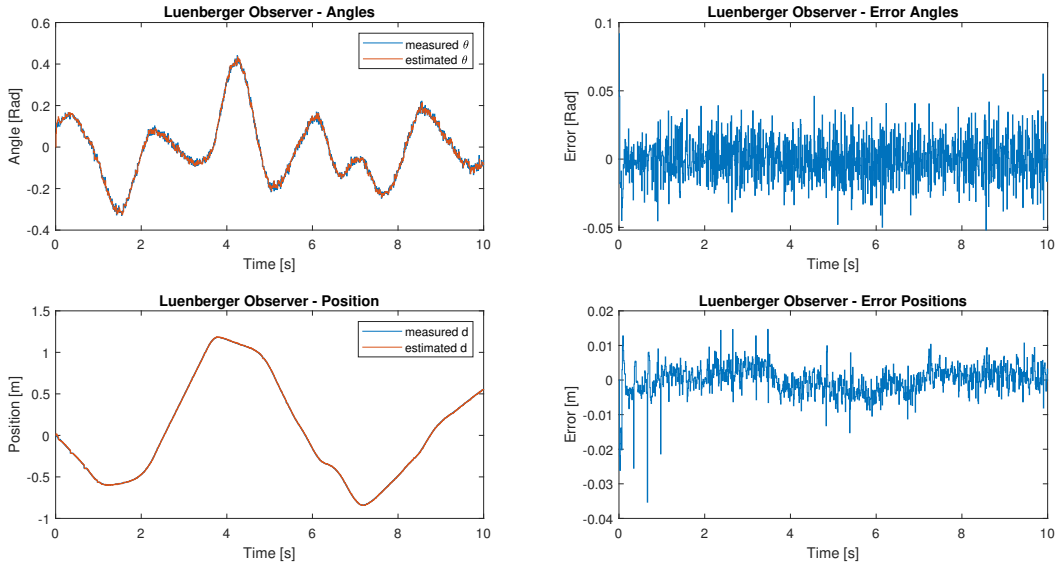


Figure 11: Estimated output of the Luenberger observer is compared to the measured output of the system. The plots on the left show the trajectories of both measured- and estimated angles/position. Plots on the right show the error between the estimation and measured values.

5.2 Kalman Filter

A Kalman Filter is a framework in which states are reconstructed in a statistically optimal manner [3]. Generally, this is expressed as the unbiased, minimum variance state-reconstruction error. The main difference

between the Kalman Filter and an Leunberg observer is that the Kalman Filter takes into account the process- and measurement noise. A steady-state Kalman filter implementation is used if the state-space model and the noise covariance matrices are all time-invariant, if not a time-varying implementation is available. Given the plant:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Gw(k) \\ y(k) &= Cx(k) + Du(k) + Hw(k) + v(k) \end{aligned}$$

with known inputs u, white process noise w and white measurement noise v satisfying

$$\begin{aligned} E[w(k)] = E[v(k)] &= 0 \\ E\left[w(k)w^T(k)\right] &= Q \\ E\left[v(k)v^T(k)\right] &= R \\ E\left[w(k)v^T(k)\right] &= N \end{aligned}$$

The estimator has the following state equation

$$\hat{x}(k+1|k) = A\hat{x}(k|k-1) + Bu(k) + L(y(k) - C\hat{x}(k|k-1)) \tag{5.1}$$

where the observer gain L is calculated through solving the discrete Riccati equation [4]:

$$L = (APC^T + \overline{N})(CPC^T + \overline{R})^{-1}$$

where,

$$\begin{aligned} \overline{R} &= R + HN + N^T H^T + HQH^T \\ \overline{N} &= G\left(QH^T + N\right) \end{aligned}$$

The interested reader can refer to [3] chapter 5 for a more detailed mathematical derivation of the Kalman Filter.

5.3 Creation of the Time-Invariant Kalman Filter

To create a Time-Invariant Kalman filter, it is of great importance to follow the theory as stated above. Hence, determining the process- and measurement noise covariance matrices will be the initial step in the creation of the Kalman Filter. A covariance matrix of a random variable with itself can also be thought of as the variance:

$$\mathrm{Var}(X) = \mathrm{Cov}(X, X) \tag{5.2}$$

$$\mathrm{Var}(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2 \tag{5.3}$$

where μ is the expected value, i.e.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{5.4}$$

In the case of the gantry crane setup, it was only possible to measure the measurement noise. Therefore, these formulas only applied for the creation of the measurement noise covariance matrix. This was done by starting the setup, but giving the system no input. It was observed that when applying the formulas to the

noise measurements, that for the distance sensor of the trolley the variance was zero. Since R needs to be a positive definite matrix, a choice was made to place 1e-10 in the top left entry.

For the process noise covariance matrix a $n \times n$ matrix was created with very small positive number on the diagonal. This was done because it was not able to create any measurements to calculate the process noise. It was assumed that the process noise is very small, and thus a small Q matrix was chosen. The two covariance matrices can be found below. The process and measurement noise cross-covariance matrix, N, is assumed to be zero.

$$R = \begin{bmatrix} 1e-10 & 0 \\ 0 & 0.8210 \end{bmatrix} \qquad Q = 1e-4 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad N = \begin{bmatrix} 0 \end{bmatrix}$$

After implementing the three different covariance matrices, the state estimates could be created. The following results were obtained.

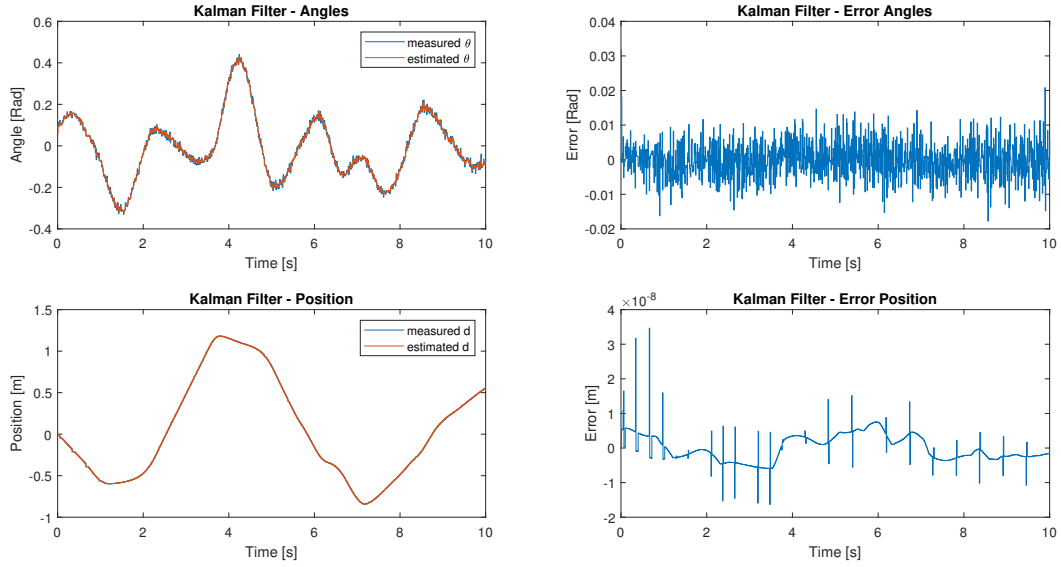


Figure 12: Estimated output of the Kalman Filter is compared to the measured output of the system. The plots on the left show the trajectories of both measured- and estimated angles/position. Plots on the right show the error between the estimation and measured values.

5.4 Comparison

A comparison can be made made between the two observers. When looking at the error figures, given in Figure 11 and 12, it can be observed that the Kalman observer completely outperforms the Luenberger observer. The error in the angles is decreased by a factor of roughly five, and the error in the position roughly by a factor of 10^6.

6 Controllers

In this section we will discuss the two controllers designed for our control objective, LQR and PID. We will give a general description of the controllers and review their results. Also we will briefly go over discretization of the plant.

6.1 Discretization

When discretizing a system, choosing a reasonable sample time (h) is essential due to aliasing, computational effort and equipment cost. A suitable sample time will yield about 10 samples per rise time [6]. A continuous time state space model can be transformed into a discrete state space model using the sample time h and the zero-order hold approximation:

$$\Phi(h) = e^{Ah} \qquad \Gamma(h) = \int_0^h e^{A\eta} B d\eta \tag{6.1}$$

It should be noted that the discretization method used for PID controllers mostly is 'Tustin', although a bit more complex, it better preserves the phase curve (i.e less phase warping [6]) which was carefully tuned for this specific plant. The discretized plant takes the form:

$$x(kh+h) = \Phi x(kh) + \Gamma u(kh) \tag{6.2}$$
$$y(kh) = Cx(kh) \tag{6.3}$$

6.2 LQR

Linear Quadratic control problem is an extension of pole placement techniques where the design parameter was the locations of the closed-loop poles. For a successful implementation, it is required that (Φ, Γ), is an controllable pair. In the case of the container crane setup this is indeed the case. LQR however aims to minimize a criterion, which is a quadratic function of the states and control signals. The solution of LQ criterion leads to a optimal control law which can be used in a standard state-feedback design. An advantage of the LQ-controller is that it is easy to compromise between speed of the system and the magnitude of the control input. Control signals and states can be penalized by tuning the Q and R matrices accordingly. These matrices arise in the loss function to be minimized:

$$J = \int_0^\infty \left(x^T Q x + u^T R u + 2x^T N u\right) dt \tag{6.4}$$

The feedback gain which minimizes this cost function is determined using the MATLAB function *lqr*, which outputs [K,S]. K being the feedback gain en S the associated Ricatti equation. A step by step solution to optimal control law can be found in chapter 11 of [6]. But some theoretical background is given below.

$$u(hk) = -Kx(hk) \tag{6.5}$$

$$x(kh+h) = \Phi x(kh) + \Gamma u(kh) \tag{6.6}$$
$$x(kh+h) = \Phi x(kh) - \Gamma K x(hk) \tag{6.7}$$
$$x(kh+h) = (\Phi - \Gamma K) x(kh) \tag{6.8}$$

Higher Q values will result in a fast system, increasing the overshoot and reducing rise time. As R increases, an optimal K is found by reducing the control action. Hence, a higher value of R will make the system slower. Tuning a LQ controller is deciding an appropriate trade off between response time and control action.

6.2.1 Design Decisions

Before starting to create a simulation model, a few design decisions need to be taken into account. Firstly, using state feedback in a simulation requires the use of a feed forward gain, which can be represented as the inverse of the DC-gain.

$$FF = \frac{1}{\text{DC-Gain}} \tag{6.9}$$

Secondly, the choice of the weighing matrices needs to be made. When observing the output signal it is observed that the output purely depends on the first and third state, which represents the position of the trolley and the angular position of the container. Therefore, it would only make sense to penalize these states. Since no (direct) restrictions on the control input to the system are given yet, there is no need to penalize the control action. Hence, the following Q and R are chosen.

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 \end{bmatrix}$$

Next, a method for the conversion of the continuous-time poles to the discrete-time poles is proposed. The MATLAB function *eig* was used find to pole positions using the optimal gain for continuous-time. Since we are interested in the discrete time location (z-domain) of the poles, the s-domain (continuous time) poles have to transformed using:

$$[\text{z-domain poles}] = e^{[\text{s-domain poles}] \cdot h} \tag{6.10}$$

The MATLAB function *place* was used to receive the new gain required for the discretized plant. For this conversion of eigenvalues a sampling time of 0.01 seconds was chosen. This resulted in a over-sampled but stable system. The decision for taking 0.01 seconds as sampling time also arises from the fact that in the real world system 0.01 shows smooth results regrading the settling of to the reference point. Lastly, it is important to notice that the system only outputs 2 out of the 4 state variables. Hence it is important to include an observer in the simulation model, and test if this works.

6.2.2 Simulation

Now that all design choices have been made a discrete time domain simulation can be create. In the simulation a Luenberger observer was used. Below a depiction of the simulation is found.

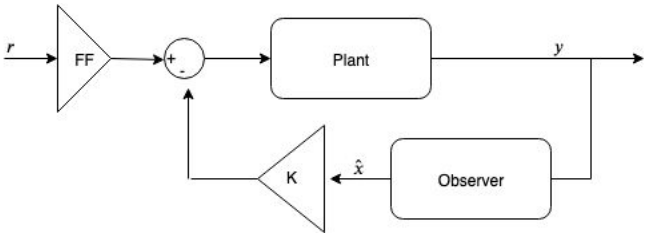
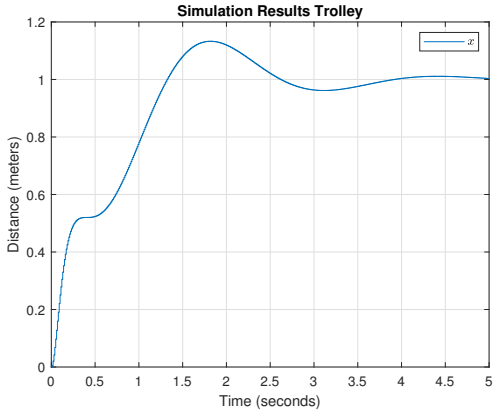


Figure 13: State feed-back with observer.

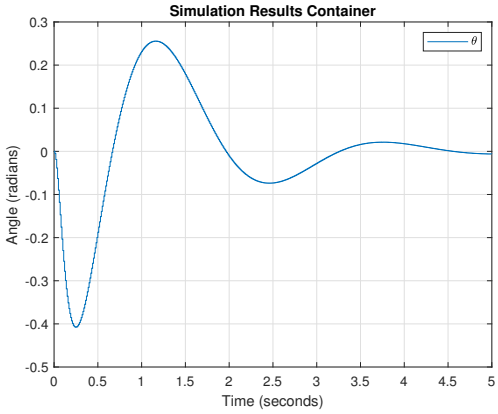
As can be seen above the reference input enters the system and is directly multiplied with the Feed Forward gain. After this the summation with the control input generates the signal that goes into the plant. The output of the plant is fed as input to the Luenberg observer and gives the state estimates. These state-estimates

are then multiplied with the optimal state-feedback gain, K.

After the implementation the simulation is run, this gave the following simulation results, whom are depicted below. The results of the simulation are sufficient, the trolley moves towards the reference position and the angle of the container settles to zero after roughly 5 seconds. It should be noted that in the simulated response of the trolley, a velocity decrease is observed after 0.5 seconds, this is due to the second and higher order terms in the closed loop transfer function.



(a) Step response of the Trolley, with a state-feedback controller.



(b) Angular response of container with state-feedback.

Figure 14: Step response plots using state-feedback.

6.3 PID

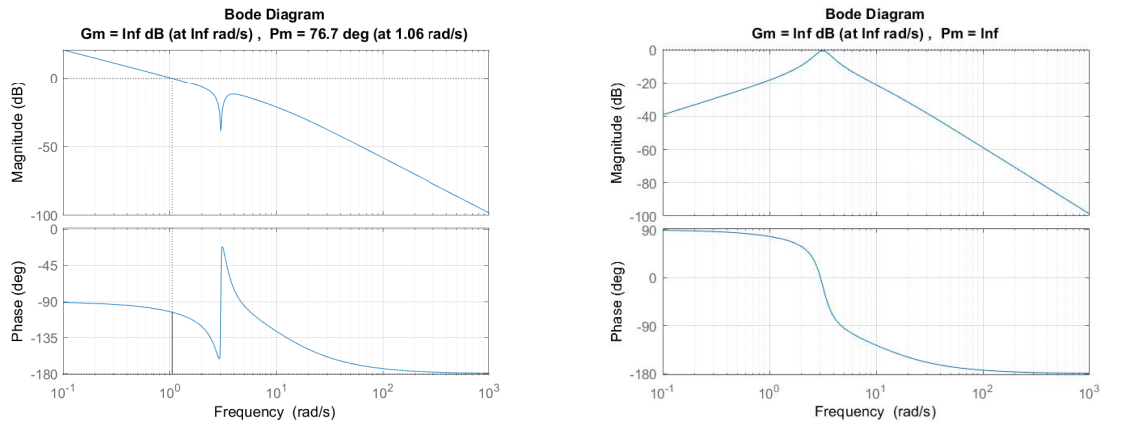
In this section the design of a PID controller for the container crane is discussed. The design method for tuning the PID controller is called "Loop-Shaping", which is a method for shaping the PID based on the Open-Loop Bode plot and the step response. Since the container crane system is a Single Input, Multiple Output system (SIMO), there are two transfer function going from the reference input to the system's output. A control system will be designed with the transfer functions that corresponds to the container crane. All designs will be executed within the continuous time domain and will later on be transferred to discrete time. Both these transfer function are analysed separately as if they were Single Input, Single Output (SISO) systems to gain better insight in the behaviour of the system. Eventually the system is put in a Cascade form, where the system contains an inner- and outer-loop.

6.3.1 System Analysis

The system analysis is a crucial step, which is needed for a proper PID controller design. Prior to the start of the design of the controller, a thorough inspection of the transfer functions, whom are given below, is required. Here, $G_{real1}(s)$ represents the transfer function from the reference position of the trolley to the output position, and $G_2(s)$ represents the transfer function from the reference angle of the container to the output angle. Please note that there is a $G_{real1}(s)$, which has a minus sign in front of it. This minus sign causes some trouble regarding the analysis of the open- and closed-loop (without control). Hence, $G_1(s)$ is created. $G_1(s)$ is actually $G_{real1}(s)$, but multiplied with minus one. This can be viewed as multiplying the $G_{real1}(s)$ system with a proportional gain of minus one. For further theoretical analysis $G_1(s)$ will be used.

$$G_{real1}(s) = \frac{-12.45(s^2 + 0.03596s + 9.114)}{s(s + 10.37)(s^2 + 1.152s + 10.05)}$$
$$G_1(s) = \frac{12.45(s^2 + 0.03596s + 9.114)}{s(s + 10.37)(s^2 + 1.152s + 10.05)}$$
$$G_2(s) = \frac{11.535s}{(s + 10.37)(s^2 + 1.152s + 10.05)}$$

The open loop bode plots of the plants are given below.



(a) Bode diagram of the open-loop system, $G_1(s)$ (b) Bode diagram of the open-loop system, $G_2(s)$

Figure 15: The bode plots corresponding to $G_1(s)$ and $G_2(s)$.

It should be noted that the bode plot corresponding to $G_1(s)$ has infinite Gain Margin and 76.7 degrees of Phase Margin, with a cross-over frequency at 1.06 rad/s. The bode plot of $G_2(s)$ lies completely below the 0 dB line, this makes sense since it represents a pendulum in a downwards position. Having a bode plot completely beneath the 0 dB line ensures that all incoming inputs will be reduced, when the system is put in a closed-loop.

Now that the open-loop bode plots have been analyzed, the next step will be placing the system in a closed loop structure. After placing the plant in a closed loop, it is possible to explore the closed-loop step response (without control). Here the standard closed loop transfer function of a system with negative feedback is given:

$$\frac{Y(s)}{R(s)} = \frac{G(s)}{1 + G(s)} \tag{6.11}$$

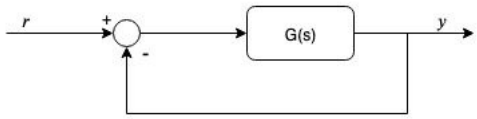
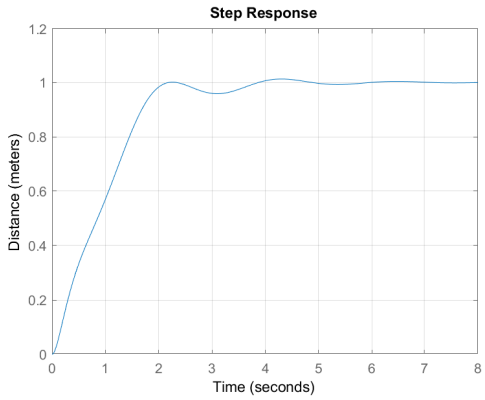


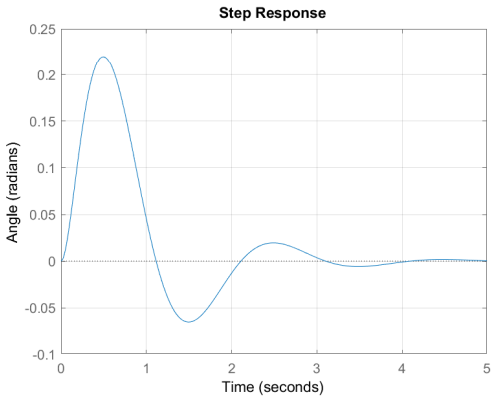
Figure 16: Overview of the system containing a negative feedback loop, where r represents the reference input signal, $G(s)$ represents the plant's transfer function and lastly, y represents the output of the system..

The closed loop step responses of the plants are given below. It can be observed that both system are stable, the trolley has a satisfactory response and settles very well. The step response of the pendulum converges to

zero. This was expected since the bode magnitude plot completely lies beneath the 0dB line. As mentioned before, this entails that all inputs will eventually die out and converge to zero.



(a) Step response of the closed-loop system, with $G_1(s)$ as plant



(b) Step response of the closed-loop system, with $G_2(s)$ as plant

Figure 17: The step response plots corresponding to the closed loop systems of $G_1(s)$ and $G_2(s)$.

6.3.2 Steady-state error

After analyzing the system which only contains the plant, it is now required to asses the steady-state error of the uncontrolled system. This is done to decide what type of controller is desired for this system, it is required to analyze the steady-state error of the system. Because, when the system contains a steady state error an Integral action is required to remove this steady-state error. Since the transfer function of the trolley represents a type 1 system, from the book of Franklin [7] it is know that the steady state error will be zero for the uncontrolled loop. This can be proven by application of the Final Value Theorem, where $E(s)$ represents the transfer function of the error signal and $R(s)$ represents the transfer function of the reference input. This $R(s)$ transforms into the following form $R(s) = \frac{1}{s^{k+1}}$. This form can only be used when taking either step, ramp or parabola inputs where k represents a 0, 1 or 2 respectively. For the analysis a step reference is preferred and hence, k is equal to zero.

The transfer function of the pendulum is not a type one system, but a type 0 system. It will be proven that for a step input the system will have a steady state error of 1. Now at first sight this might seem not wished for, but on the contrary this is actually exactly what is needed and what was expected. The system is a downwards pendulum, hence it will always converge to its own equilibrium position, which is zero. Later on in this chapter it will be shown that this is highly beneficial for a anti-sway container controller.

$$E(s) = \frac{1}{1 + G(s)} R(s) \tag{6.12}$$

$$e_{ss} = \lim_{t \to \infty} e(t) = \lim_{s \to 0} sE(s) \tag{6.13}$$

$$= \lim_{s \to 0} s \frac{1}{1 + G(s)} \frac{1}{s^{k+1}} \tag{6.14}$$

By working out the Final Value Theorem given in equation 6.13, the steady-state error of the uncontrolled system can be discovered. The example start of with analyzing the trolley and next the pendulum will be analyzed. In this example it is easier to calculate the limit of s going to zero for $G(s)$.

$$= \lim_{s\to 0} G_1(s) \Rightarrow \lim_{s\to 0} \frac{12.45(s^2 + 0.03596s + 9.114)}{s(s + 10.37)(s^2 + 1.152s + 10.05)} = \frac{12.45 \cdot 9.114}{0} = \infty$$

$$e_{ss} = \frac{1}{1 + \infty} = 0$$

After dividing by infinity it is easily observed that this type 1 system has a zero steady-state error without using any form of integral action. Therefore, it can be concluded that no integral action is needed for $G_1(s)$. For the G_2 the same example is worked out below.

$$= \lim_{s\to 0} G_2(s) \Rightarrow \lim_{s\to 0} \frac{11.535s}{(s + 10.37)(s^2 + 1.152s + 10.05)} = \frac{0}{0} = 0$$

$$e_{ss} = \frac{1}{1 + 0} = 1$$

After dividing by zero it is shown that this type 0 system has a steady-state error of one. As mentioned before, this might seem like a bad thing, but later on it will be shown that this is exactly what the system requires to operate at its full potential.

6.3.3 Cascade Control Loop

Now that the system analysis is finalized and we discovered the system types, it is possible to start creating the actual control system and controllers. The final control system is a cascade system, where two separate controllers are used to control the trolley and the container to a desired reference location.

Cascade system
Until now all analysis was done as if the system can be decoupled and represented as two SISO systems. Obviously this will not work in practice, since the system is not decoupled. Hence a solution for this problem is required. A viable solution is putting the system is a cascade control loop. The cascade control loop is depicted below.

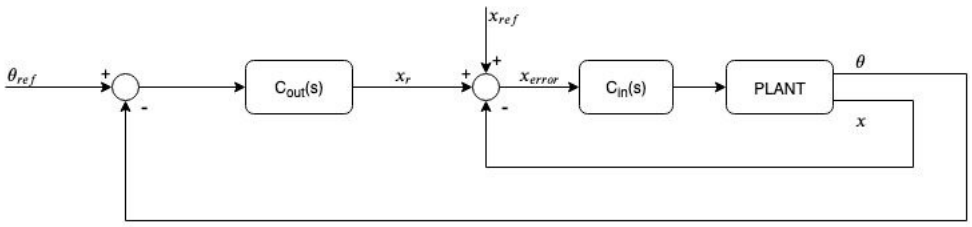


Figure 18: Cascade control loop, where there is an inner- and outer-loop.

The cascade loop consist out of an inner- and outer-loop. The inner loop controls the position of the trolley and generates a control input that goes into the plant. The plant generates the two measured outputs; the position of the trolley, x, and the angular position of the container, θ. The output of the position comes out of the plant it is brought back to a summation of an externally specified reference position, x_{ref} and the output of the angular controller form the outer loop, x_r. These three elements generate an error signal, x_{error}, which is fed to the inner loop controller. The outer loop of the cascaded system is less complicated. The angular position of the container is fed back to a summation with the reference position of the container angle. This angular reference position is always zero, since the objective is to create an anti-sway controller. Just like a regular control structure, the error is fed to the outer loop controller and this generates x_r, which is a part of the input for the inner loop. A good thought experiment to gain more insight in the cascade control structure is as follows. When the simulation is run the position of the trolley is not yet at its desired reference location, this will generate a control input that is fed into the plant. As the trolley starts moving

towards its reference position, the container will start oscillating. As the trolley moves closer and closer to its reference position the x_{error} becomes smaller. When the trolley has finally reached its desired location, but the container is still swinging, the outer loop will generate a nonzero x_r. This leads to a nonzero x_{error}, which will lead to a control action in such a manner that the container will stop swinging and the trolley will end up in the reference position.

To complete the design theory of the cascade system a more mathematical description of it is required. The control input of the plant has changed, as described above. Hence it can be put in the following form, here $C_{in}(s)$ represents the inner loop controller.

$$u = C_{in}(s)(x_r + x_{ref} - x) \tag{6.15}$$

Creating this new control input enables for a new way of describing the system in a simplified version where the inner-loop is seen as one big block, P_{in}. This new structure can be seen in the figure below.

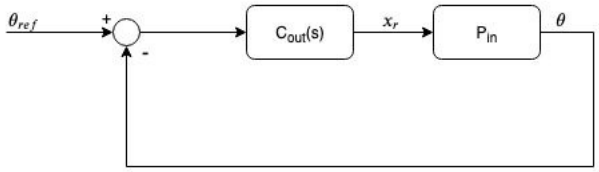


Figure 19: Cascade control loop, where the plant and the inner control loop have been merged into P_{in}

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{41} \\ a_{12} & a_{22} & a_{32} & a_{42} \\ a_{13} & a_{23} & a_{33} & a_{43} \\ a_{14} & a_{24} & a_{34} & a_{44} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ b_2 \\ 0 \\ b_4 \end{bmatrix} u \tag{6.16}$$

$$\begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{6.17}$$

If equation 6.15 is substituted in the equation above, the following state space model is obtained.

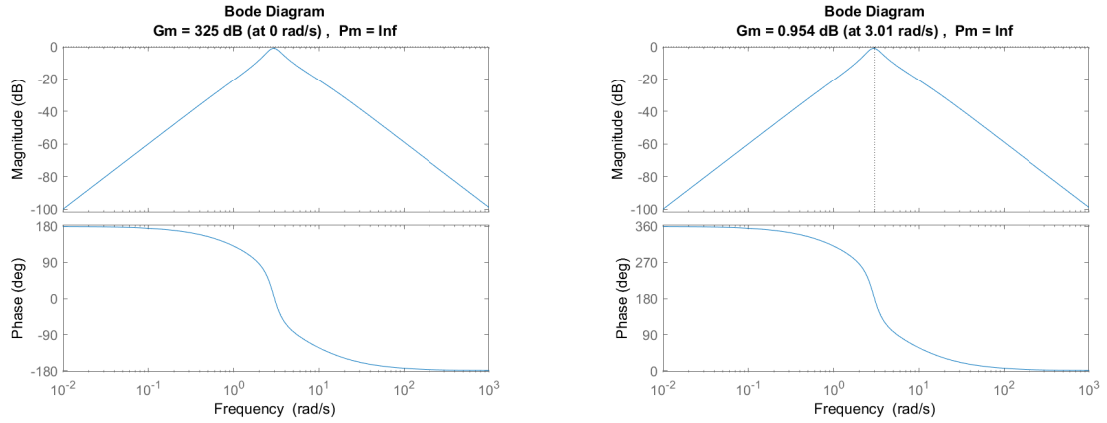
$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{41} \\ a_{12} - b_2 C_{in}(s) & a_{22} & a_{32} & a_{42} \\ a_{13} & a_{23} & a_{33} & a_{43} \\ a_{14} - b_4 C_{in}(s) & a_{24} & a_{34} & a_{44} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ b_2 \\ 0 \\ b_4 \end{bmatrix} (x_r + x_{ref}) \tag{6.18}$$

$$\theta = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{6.19}$$

Now that all theory regarding the PID controller has been completed. The tuning itself can start, for the tuning of the first controller, $C_{in}(s)$, because it is possible to treat the inner loop as a SISO system, it is wise to look at the bode plot made for $G_1(s)$. As discussed previously, $G_1(s)$ is actually the real transfer function, $G_{real}(s)$, multiplied with a proportional gain of minus one. Hence the choice for $C_{in}(s)$ will be a proportional gain of minus one, because as shown before this gave a satisfactory response. It should also be mentioned that whenever the proportional gain is chosen too low (e.g. -30), the trolley would be uncontrollable, a sinusoidal like control input signal would be generated and the trolley would start moving from left to right in a very aggressive manner. Hence, minus one is a proper choice for the controller. Now that the first controller is chosen, we can create the system as described in equation 6.18 and equation 6.19. After doing so a new transfer function, G_{out}, is created from the angle reference position to the measured angle output.

$$G_{out}(s) = \frac{11.535(s + 2.343e - 08)(s - 2.343e - 08))}{(s + 8.8)(s + 1.495)(s^2 + 1.229s + 8.599)}$$

With this transfer function the corresponding open loop bode plot can be plotted, as shown below.The bode plot shows that the system has a Gain Margin of 325 dB at 0 rad/s and infinite Gain Margin. This 325 dB at 0 rad/s is definitely not ideal and will cause a slow converge of the pendulum. By choosing a proportional gain of minus one the following open loop bode plot is obtained. Now that the crossover frequency is at 3.01 rad/s, it can already be predicted that with the controller the anti sway response will be a lot faster when comparing it to the response without any controller.



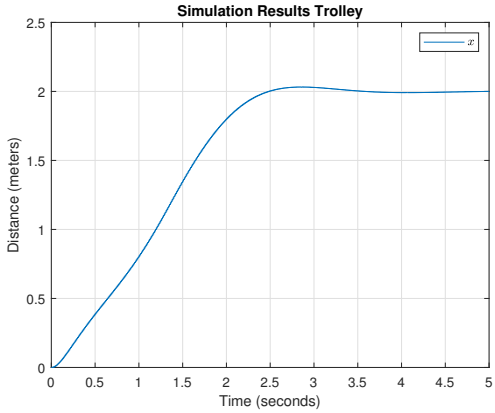
(a) The bode plot of the open loop, of G_{out}, without control

(b) The bode plot of the open loop,$-G_{out}$, with control where a proportional gain is chosen of minus 1

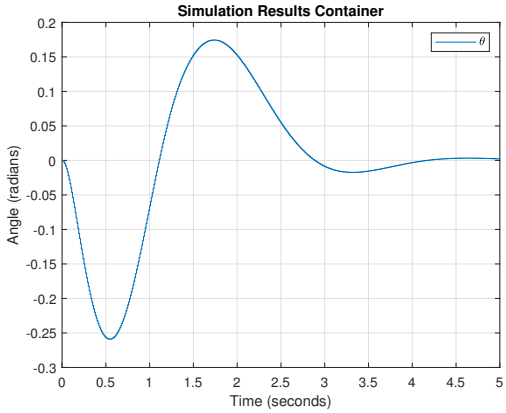
Figure 20: The bode plots of the open loop without and with control

6.3.4 Simulation

Now that the final design is finished and the gains of the controllers ($C(z)_{out}$ = -1, $C(z)_{in}$ = -1) have been found, the simulation can be created and tested. A system similar to the one as depicted in Figure 18 was created. The only difference is that until now all the modeling and design has been done in continuous-time. To make a conversion to discrete-time a sampling time of 0.01 is chosen. A sampling time of 0.01 will create an over-sampled system, but the system will remain stable, later on in the results choosing a sampling time of 0.01 creates a smooth and good response. In the simulation x_{ref} was equal to 2 and θ_{ref}, obviously equal to 0. Below are the depictions of the simulation results.



(a) Step response of the trolley, containing two proportional controllers in a cascade system



(b) plot of the angle of the container, containing two proportional controllers in a cascade system

Figure 21: Container crane system results when a cascade system is implemented and two proportional controllers are chosen ($C(z)_{in} = -1$ and $C(z)_{out}$)

When observing the results shown above, it can be seen that the two proportional controllers in a cascade system perform very well. The trolley is able to to reach the reference point in a fast manner and decrease the sway of the container when the trolley is (almost) settled. In comparison to the results that were obtained from the simulated response when using state-feedback, the cascade system outperforms the state-feedback controller. It generates a quicker settling time and slightly faster converges of the container to zero.

7 Results and Modifications

In this section the results achieved by the LQR- and PID controller will be discussed. During the testing of the controller it was observed that the control input was too high for both controllers. Hence, the controller or control structure was modified to satisfy the control input constraint of [-1, 1].

7.1 Results LQR

In the final tests of the real world container crane system, almost the exact same design decisions were chosen regarding the sampling time, weighing matrices and conversion of the poles. The difference in the real world system is that a Kalman Filter was used instead of a Lueunberg. Also no Feed Forward gain was required in this system. A depiction of the system is shown below.

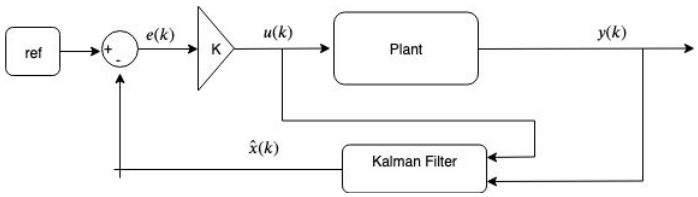


Figure 22: Schematic of system with a Kalman Filter.

After implementing the state feedback controller in the system, and determining that the reference position is 2 meters. The following results were obtained.

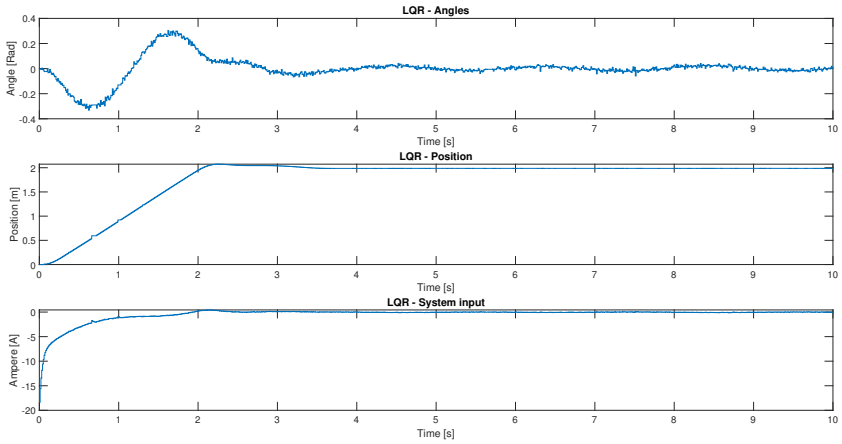


Figure 23: Measured angle, position and control input of the container crane system using LQR reference tracking. A reference of 2 was given to the position whilst the control input had no saturation bound.

As mentioned before the control input for the system is too high. So, some slight modifications regarding the design were required. By increasing the R matrix, which is a weight for penalizing high control inputs, the system is able to reduce the control input while still creating a satisfactory response. It was also required to alter the Q matrix slightly.

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 500 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 400 \end{bmatrix}$$

It should be observed that the angle of the container takes roughly one second more to converge to its equilibrium position of zero, this obviously comes from the limitations put on the control input. Furthermore, the sway in the container angle seems to have increased, this is due to the trolley staying longer and the overshoot location (roughly 2.3 meters).

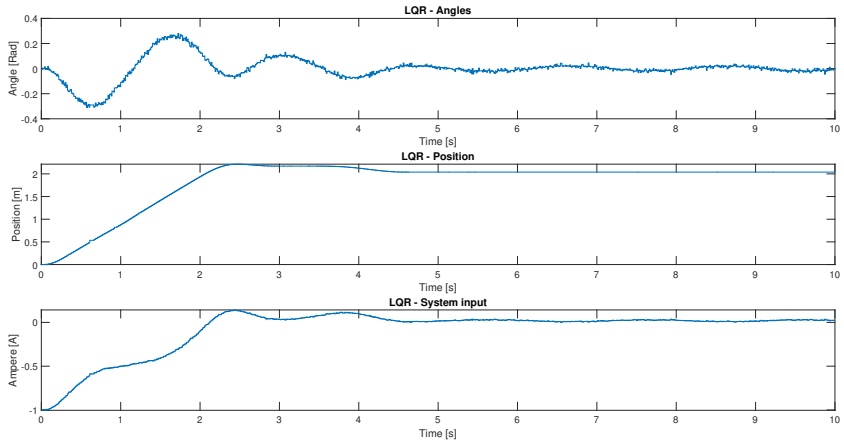


Figure 24: Measured angle, position and control input of the container crane system using LQR reference tracking. A reference of 2 was given to the position whilst the control input had a saturation bound of [-1,1].

7.2 Results PID

The final controller for this assignment is the PID structured controllers in a cascade setting. In the simulation is was observed that both, $C_{in}(z) = -1$ and $C_{out}(z) = -1$, would give satisfactory results. For the real world container crane system, it was decided to use the estimates of the output for subtraction with the references, this choice was made because the estimates contain less or no noise. If the real output measurements were chosen, there is a potential risk of creating a noisy control input signal. A depiction of the system is shown below.

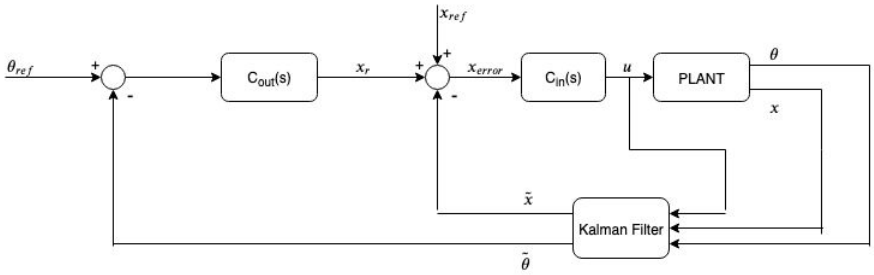


Figure 25

After the implementation of the controllers the following results were obtained when a reference position for the trolley was chosen to be 2.

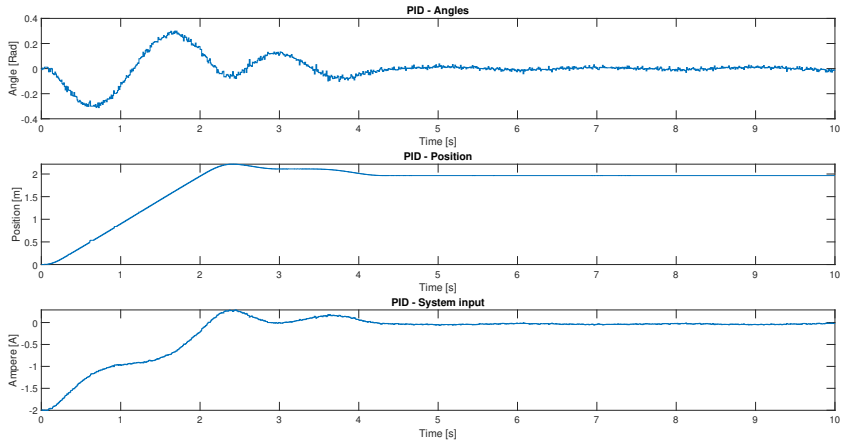


Figure 26: Measured angle, position and control input of the container crane system using PID reference tracking. A reference of 2 was given to the position whilst the control input had no saturation bound.

As can be seen above the control input to the system is out of the set control input bounds of [-1, 1]. Hence some small modifications are required. It should be noticed that as the simulation starts the control input is -2. This is due to the fact that the $x_{ref} = 2$, is multiplied with -1. This entails that the reference position of the trolley directly influences the control input to the plant. A solution for this problem is changing $C_{in}(z)$. It is possible to create a formula for $C_{in}(z)$ in such a manner that the initial control input is always -1, and thus overreaches the control input boundary.

$$C_{in}(z) = -\frac{1}{x_{ref}} \tag{7.1}$$

After introducing this new $C_{in}(z)$, the following results are obtained. As can be seen below, the input constraints are met. It should be noted that the performance of the system has not decreased much due to the input constraints. The trolley is about as fast as it was without the constraints. Regarding the sway motion, it is observed that the constraints are of little influence.

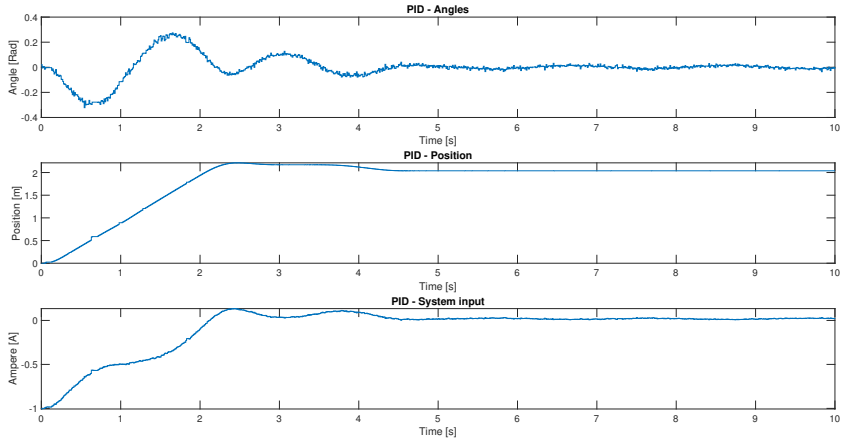


Figure 27: Measured angle, position and control input of the container crane system using PID reference tracking. A reference of 2 was given to the position whilst the control input had a saturation bound of [-1,1].

7.3 Results Comparison

A small comparison between the two controllers is given in this subsection. The two controllers, with no input constraints are compared to each other as well as the controller with input constraints, this is done to ensure a fair comparison. Both unsaturated controllers give satisfactory results, they are both capable of reaching their reference points within reasonable time, and the swing of the container decays greatly within approximately 2 seconds. Also, both controllers are capable of staying within the swing bounds of [-0.5, 0.5] rad. But, although it is not very clearly visible in Figure 23, the control system containing state-feedback has a very small oscillation after the trolley has stopped moving, the PID controllers in the cascade system does not have this problem as can be seen in Figure 26.

When viewing at the saturated controllers in Figure 24 and 27, again the same observation can be made as for the unsaturated controllers. Both controllers are capable of reaching the trolley's reference position in approximately 2 seconds, staying within the [-0.5, 0.5] rad swing bound and it is also observed that the state-feedback controller gives rise to really small oscillations when the trolley is settled. Thus, the PID in cascade system structure is better capable of anti-sway.

The optimal controller for this assignment would be the PID controller in a cascade system with input constraints. This controller shows the best performance, while taking into account the system's input constraints. Therefore, this control structure is the optimal choice.

8 Conclusion

As the final design has come to an end, two controllers have been implemented and tested. In this section we will look back and give some final concluding remarks. The project started of with the creation of the control objective. The controller had to be capable of reference tracking, not overreach a certain amount of sway and have a anti-sway policy within two seconds after the trolley has reached its reference position. Next, the system's dynamics needed to be created. This was done via the Euler Lagrange method, where kinetic- and potential energies are used to gain insight in the model dynamics. Some dynamics were left out to simplify the model (like the internal control loop), however that did not have a significant effect on our controlled system.

After the completion of the dynamic model it was possible to start testing for the very first time, this immediately gave rise to problems regarding the sensor calibration. The setup was calibrated such that the sensors outputs logical values coherent within the SI metric. It was noticed that there were missing calibrated gains and offsets to be compensated for. By making the decision of only measuring the container angle with one sensor, instead of two, and compensating for the gains and offsets the measurements became correctly calibrated. Succeeding the calibrated sensors was the parameter estimation, creating the model dynamics also gave insight in the parameters of the dynamic model. Via Greybox estimation four different parameters were estimated and implemented in the dynamic model. The System Identification process gave realistic values for the estimated parameters but revealed an expected difference in our decoupled trolley model and the actual system. This was because it was not possible to detach the container from the trolley in the setup, whilst in the decoupled model this was not taken into account for.

Next, was the implementation of the first controller, but this was hindered due to the absence of full-state information. Hence, our design required the use of an observer. After creating the Luenberger observer it was noticed that the estimated states were good but not accurate enough. This was because of the measurement noise of the sensors. The solution to this problem was introducing a Kalman observer, the Kalman observer takes into account this noise and filters it out. This full state reconstruction, allowed for the creation of the first controller. A LQR controller was created, by first discretizing the whole system and choosing the weighing matrices. Choosing these weighing matrices correctly ensure a appropriate trade off between response time and control action. After this step the first simulation results were created and analyzed. Next, the second controller had to be created. The choice for a PID type controller was made. A thorough system analysis was performed, which gave rise to many insights and a lot of information. Eventually, after finalizing

the system analysis the choice for a cascade control loop was made. This was done since the system is a SIMO system, and the regulation of two outputs was quite difficult. The cascade control loop structured solved this problem relatively quickly by choosing two appropriate proportional gains for the inner- and outer-loop respectively. Choosing the gains correctly allowed us to create a new simulation where the cascade system was tested and it was found that the system was stable and able to satisfy the control objectives.

Because all simulations were done and yielded satisfactory results, it was time to implement the controller into the real container crane system. After gathering the data of the real system it was noticed that the controllers did not yet satisfy the control input constraints, and hence some minor modifications were needed. After implementing these modifications the results were re-analyzed and it was concluded that both controller perform well. As mentioned before, the optimal controller for this assignment is the PID controller with input constraints in a cascade system. This controller is the best fit when looking at the real life size container, due to the fact that it is able to satisfy all control objectives and not overextend on the control input which is fed to the plant.

9 Reflection and Recommendations

During the span of SC4035 we were finally able to apply our theoretical, control related knowledge to a real setup. This is an essential educational part of the Masters program since you quickly learn that controlling a real time system is a lot more difficult than merely running a simulation. A lot of time was spend on the derivation of the model and finding a proper way to calibrate the system. This was unfortunate because this gave us less time to design more sophisticated controllers.

There are a lot of possibilities regarding future work. The derived model can always be extended and improved. Some dynamics were left out of the model, like the internal control loop of the system that actively regulates the current. Or drag forces. With an improved model a better parameter estimation can be performed. With better estimated parameters and a better (more complex) model other controllers can be considered for the system. Due to our somewhat simplified model we were unable to properly tune a Model-Predictive Controller, which could be a good asset to the controlled system. MPC is very efficient in handling input and state constraints. Another controller considered but not realised due to time constraints is the $\mathscr{H}_\infty$-controller. Such a controller ensures the system suffers less from missing dynamics in the derived model. This can be done by tuning the sensitivity- and complimentary sensitivity function properly. Synthesising a controller in this fashion guarantees robust performance.

10 References

[1] David Morin, The Lagrangian Method, Harvard, 2007, http://www.people.fas.harvard.edu/ djmorin/chap6.pdf.
[2] B. Friedland, Control System Design: An Introduction to Statespace Methods. Dover Publications, 2005
[3] M.Verhaegen, Vicent Verdult, "Filtering and System Identification; A least squares approach", Cambridge.
[4] Mathworks: Kalman Filter, 2014, https://se.mathworks.com/help/control/ref/kalmanfilter.html
[5] Mathworks: Estimate Nonlinear Grey-Box Models, 2019, https://se.mathworks.com/help/ident/ug/estimating-nonlinear-grey-box-models.html
[6] K.J. Åström, B. Wittenmark, 2012 February, Computer-Controlled Systems: Theory and Design, Dover-Publications Inc
[7] Franklin, Gene F., et al. Feedback control of dynamic systems. Vol. 3. Reading, MA: Addison-Wesley, 1994.
[8] Mathworks: Autocorrelation, 2014,
https://nl.mathworks.com/help/signal/ug/residual-analysis-with-autocorrelation.html