FH Joanneum

Graz

Model Based Design

Balanbot

Training Unit 05

Authors
David B. Heer
Jakob Soukup
Graz, February 1, 2019

Lecturer
Alfred Steinhuber

Contents

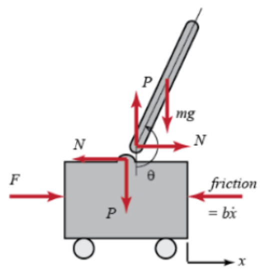
Part I
Laboratory Session 06

Introduction

In this laboratory unit the model of an inverse pendulum on a moving cart will be implemented snd simulated in simulink. In a first step the non linear model will be implemented and then discretized. After that the non linear model shall be linearized and discretized again. The differences between the two models are to be investigated. The two models shall be controlled with a PID controller. If the simulation works the model shall be deployed onto an actual moving robot to see if it holds up in real life.

1 Description of the Model

The model consists of a moving part with a hinged pendulum atop. The goal for the controller is to accelerate the cart in the right direction depending on the angle of the pendulum in order to keep it upright at all times.



Where:

x : cart's position	b : coefficient of friction for cart
$\dot{x}$: cart's velocity	l : length to pendulum center of mass
$\ddot{x}$: cart's acceleration	J : moment of inertia of the pendulum
θ : pendulum's position (angle)	F : external force applied (by motors)
$\dot{\theta}$: angular velocity	N : interaction force between cart and pendulum in x direction
$\ddot{\theta}$: angular acceleration	P : interaction force between cart and pendulum in y direction
m : mass of pendulum	
M : mass of cart	
g : gravitational constant	

Figure 1: *graphical description of the model*

The equations of the model are given by:

$$\ddot{x} = \frac{1}{M}\sum_{cart} F_x = \frac{1}{M}\left(F - N - b\dot{x}\right) \tag{1}$$

$$\ddot{\Theta} = \frac{1}{I}\sum_{pend} \tau = \frac{1}{I}\left(-Nlcos\Theta - Plsin\Theta\right) \tag{2}$$

$$N = m\left(\ddot{x} - l\dot{\Theta}^2 sin\Theta + l\ddot{\Theta}cos\Theta\right) \tag{3}$$

$$P = m\left(l\ddot{\Theta}^2 cos\Theta + l\ddot{\Theta}sin\Theta\right) \tag{4}$$

1.1 Implementation in simulink

The non linear model can be implemented using the equations above, this was already done in a previous lectore in the third semester. The resulting model can be seen in Figure 5.

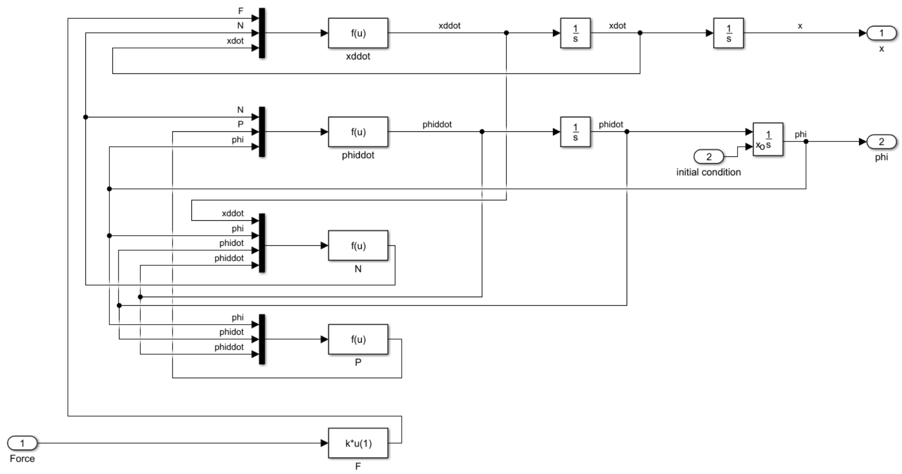


Figure 2: *Non linear continuous model in simulink*

2 Discretization from non-linear model

Since the model will later be used on an actual hardware, it is important to sicretize the system. This is done by simply replacing the continuous time integrators with discrete time integrators. The settings of the integrators are shown in Figure 3.

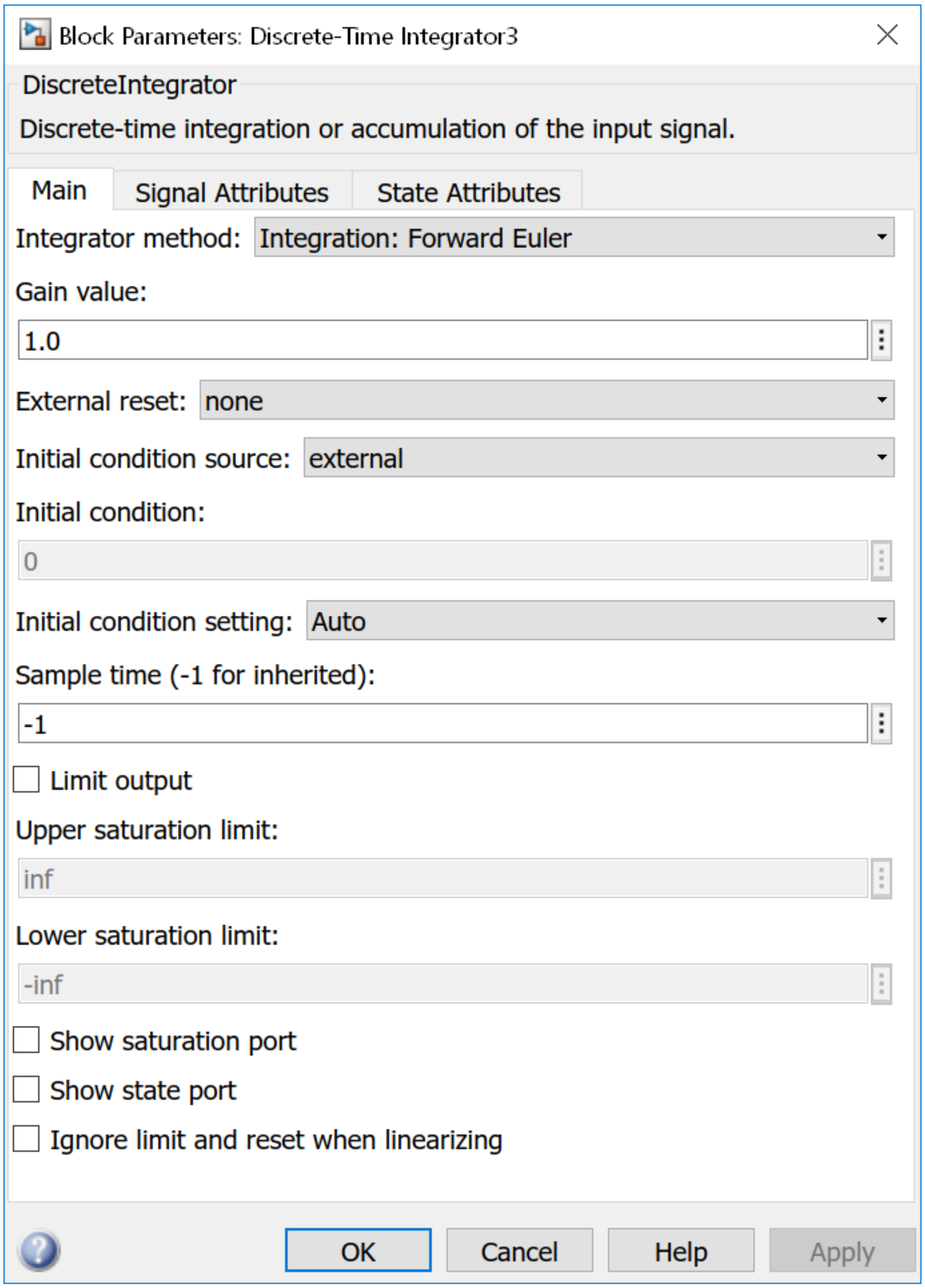


Figure 3: *discrete time integrator settings*

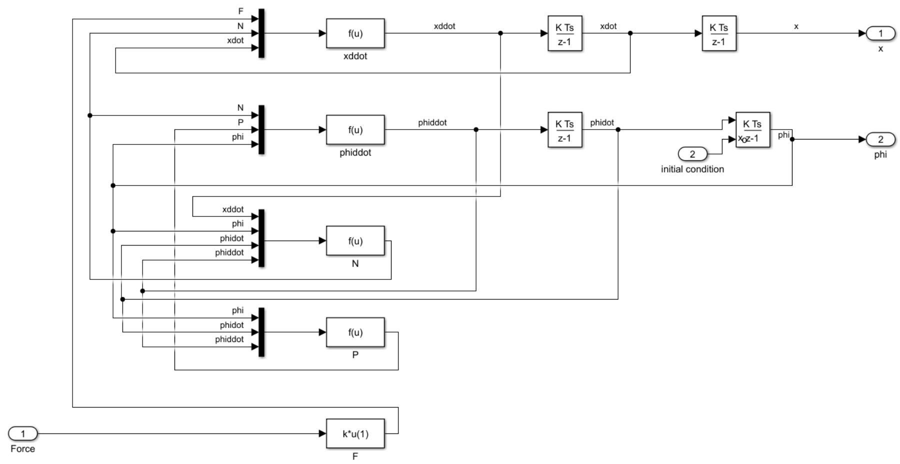


Figure 4: *Non linear discrete model in simulink*

2.1 Applying zero force to the system

As a first test the model was tested with a constant of zero at its input. It would be expected to do nothing but stay upright since there are no external forces applied to the pendulum in the horizontal axis.

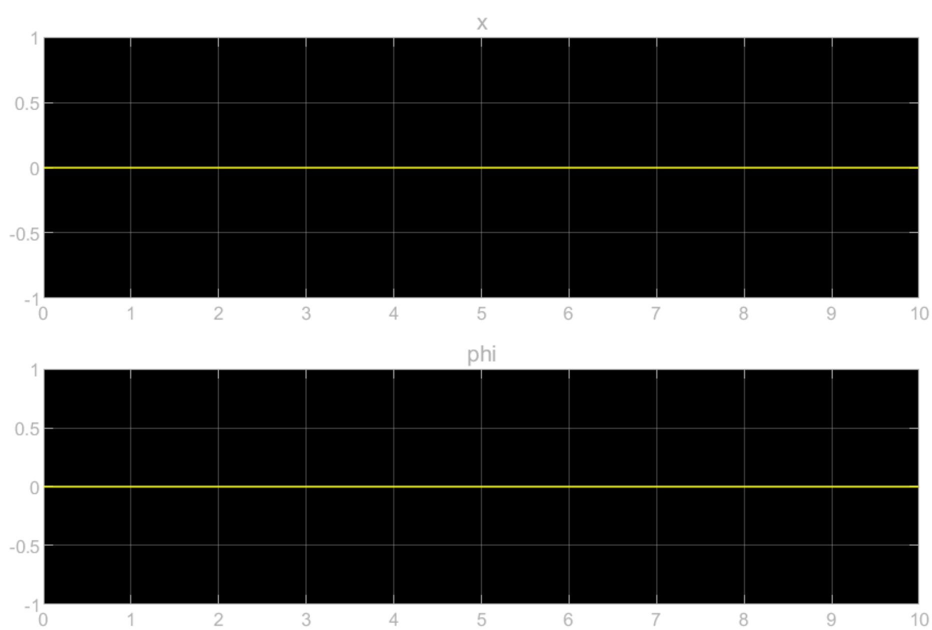


Figure 5: *Non linear discrete model simulation with zero force applied*

As a small test an initial step was applied to the system to see if it behaves correctly. The disturbance of the angle was accomplished by using the step function of simulink with an initial value of $10 \cdot \frac{\pi}{180}$, which is $10°$ in radians. As shown in Figure 6 the pendulum swings left and right and slowly loses height, so the model seems to be behaving correctly.

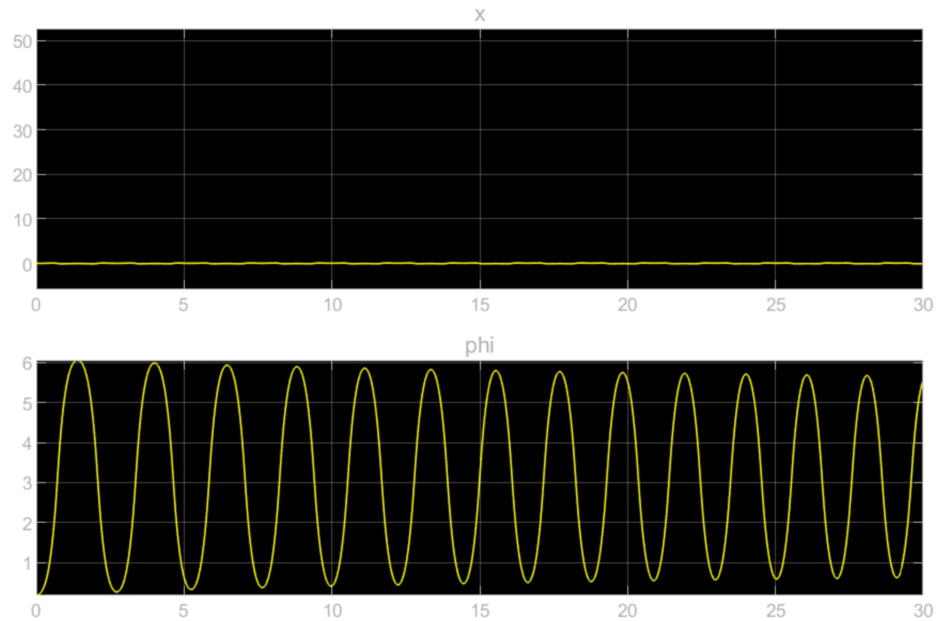


Figure 6: *Non linear discrete model simulation with an offset step in the angle*

3 Linearization

In order to further investigate the system for stability to make tuning the controller easier, it is mandatory to linearize the model. This is done by assuming that the slope of a sine wave is linear which of course is not the case but it is a valid approximation. The linearized equations are given by

$$X(s)\,s^2 = \frac{1}{M}\left(F(s) - mX(s)\,s^2 + ml\Phi(s)\,s^2 - bX(s)\,s\right) \tag{5}$$

$$\Phi(s)\,s^2 = \frac{1}{I}\left(mlX(s)\,s^2 + mlg\Phi(s) - ml^2\Phi(s)\,s^2\right) \tag{6}$$

From these equations the two transfer functions $G_1(s) = \frac{\Phi(s)}{F(s)}$ and $G_2(s) = \frac{X(s)}{F(s)}$ are to be found. This is simply a fact of rearranging the equations. $G_1(s)$ Solving equation 5 for $X(s)$:

$$\begin{aligned}
X(s)\ s^2\ M &= F(s) - mX(s)\ s^2 + ml\Phi(s)\ s^2 - bX(s)s && (7)\\
X\left(s\right)s^2M &= F\left(s\right) + ml\Phi\left(s\right)s^2 + X\left(s\right)\left[-ms^2 - bs\right] && (8)\\
X\left(s\right)s^2M - X\left(s\right)\left[-ms^2 - bs\right] &= F\left(s\right) + ml\Phi\left(s\right)s^2 && (9)\\
X\left(s\right)\left[s^2M + ms^2 + bs\right] &= F\left(s\right) + ml\Phi\left(s\right)s^2 && (10)\\
X\left(s\right) &= \frac{F\left(s\right) + ml\Phi\left(s\right)s^2}{s^2M + ms^2 + bs} && (11)
\end{aligned}$$

Inserting into equation 6 we get

$$\Phi\left(s\right)s^2 = \frac{1}{I}\left[mls^2 \cdot \frac{F\left(s\right) + ml\Phi\left(s\right)s^2}{s^2M + ms^2 + bs} + mlg\Phi\left(s\right) - ml^2\Phi\left(s\right)s^2\right] \tag{12}$$

$$\Phi\left(s\right)Is^2 = mls^2 \cdot \frac{F\left(s\right) + ml\Phi\left(s\right)s^2}{s^2M + ms^2 + bs} + mlg\Phi\left(s\right) - ml^2\Phi\left(s\right)s^2 \tag{13}$$

$$Is^2 = mls^2 \cdot \frac{\frac{F(s)}{\Phi(s)} + mls^2}{Ms^2 + ms^2 + bs} + mlg - ml^2s^2 \tag{14}$$

$$Is^2 = mls^2 \cdot \frac{\frac{F(s)}{\Phi(s)} + mls^2}{Ms^2 + ms^2 + bs} + mlg - ml^2s^2 \tag{15}$$

$$\frac{Is^2}{ml} = s^2 \cdot \frac{\frac{F(s)}{\Phi(s)} + mls^2}{Ms^2 + ms^2 + bs} + g - ls^2 \tag{16}$$

$$\frac{Is^2}{ml} = \frac{\frac{F(s)}{\Phi(s)} + mls^2}{M + m + \frac{b}{s}} + g - ls^2 \tag{17}$$

$$\frac{F\left(s\right)}{\Phi\left(s\right)} = \left[\frac{I_s^2}{ml} - g + ls^2\right]\left[M + m + \frac{b}{s}\right] - mls^2 \tag{18}$$

$$\frac{F\left(s\right)}{\Phi\left(s\right)} = \frac{MI}{ml}s^2 + \frac{I}{l}s^2 + \frac{Ib}{ml}s - gM - gm - \frac{gb}{s} + Mls^2 + mls^2 - mls^2 \tag{19}$$

$$\frac{F\left(s\right)}{\Phi\left(s\right)} = s^2\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s\left(\frac{Ib}{ml} + lb\right) - gM - gm - \frac{gb}{s} \tag{20}$$

$$\frac{\Phi\left(s\right)}{F\left(s\right)} = \frac{1}{s^2\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s\left(\frac{Ib}{ml} + lb\right) - gM - gm - \frac{gb}{s}} \tag{21}$$

$$\frac{\Phi\left(s\right)}{F\left(s\right)} = \frac{s}{s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s(-gM - gm) - gb} \tag{22}$$

$$\frac{\Phi\left(s\right)}{F\left(s\right)} = \frac{s}{s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s\left(-gM - gm\right) - gb} \tag{23}$$

$G_2(s)$ Solving equation 6 for $\Phi(s)$:

$$\Phi\left(s\right)s^2 = \frac{1}{I}\ \left(mlX\left(s\right)s^2 + mlg\Phi\left(s\right) - ml^2\Phi\left(s\right)s^2\right) \tag{24}$$

$$I\Phi\left(s\right)s^2 = mlX\left(s\right)s^2 + mlg\Phi\left(s\right) - ml^2\Phi\left(s\right)s^2 \tag{25}$$

$$\Phi\left(s\right) = \frac{mlX\left(s\right)s^2}{Is^2 + ml^2s^2 - mlg} \tag{26}$$

Inserting into the previously calculated transfer function:

$$\frac{\frac{mlX(s)s^2}{Is^2+ml^2s^2-mlg}}{F\left(s\right)} = \frac{s}{s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s(-gM - gm) - gb} \tag{27}$$

$$\frac{X\left(s\right)}{F\left(s\right)} = \frac{Is^2 + ml^2s^2 - mlg}{mls^2} \cdot \frac{s}{s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s(-gM - gm) - gb} \tag{28}$$

$$\frac{X\left(s\right)}{F\left(s\right)} = \frac{Is^2 + ml^2s^2 - mlg}{mls \cdot \left[s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s\left(-gM - gm\right) - gb\right]} \tag{29}$$

$$\frac{X\left(s\right)}{F\left(s\right)} = \frac{s^2\left(I + ml^2\right) - mlg}{s^4\left(MI + Im + Mml^2\right) + s^3\left(Ib + mbl^2\right) + s^2\left(-gml\left[M + m\right]\right) - s\left(gbml\right)} \tag{30}$$

$$\tag{31}$$

Our two transfer functions are therefore

$$G_1\left(s\right) = \frac{\Phi\left(s\right)}{F\left(s\right)} = \frac{s}{s^3\left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + s^2\left(\frac{Ib}{ml} + lb\right) + s(-gM - gm) - gb} \tag{32}$$

$$G_2\left(s\right) = \frac{X\left(s\right)}{F\left(s\right)} = \frac{s^2\left(I + ml^2\right) - mlg}{s^4\left(MI + Im + Mml^2\right) + s^3\left(Ib + mbl^2\right) + s^2\left(-gml\left[M + m\right]\right) - s\left(gbml\right)} \tag{33}$$

To validate the calculations, the results were compared to the ones yielded in the online documentation[1]. The poles and zeros matched and therefore it can be assumed that the calculations are correct.

[1]http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=SystemModeling

4 Discretization linear model

4.1 Forward Euler

$$z = e^{sT} \approx 1 + sT \rightarrow s \approx \frac{z-1}{T} \quad (34)$$

$$G_1(z) \approx \frac{\frac{z-1}{T}}{\left(\frac{z-1}{T}\right)^3 \left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + \left(\frac{z-1}{T}\right)^2 \left(\frac{Ib}{ml} + lb\right) + \frac{z-1}{T}(-gM - gm) - gb} \quad (35)$$

$$G_2(z) = \frac{\left(\frac{z-1}{T}\right)^2 (I + ml^2) - mlg}{\left(\frac{z-1}{T}\right)^4 (MI + Im + Mml^2) + \left(\frac{z-1}{T}\right)^3 \left(Ib + mbl^2\right) + \left(\frac{z-1}{T}\right)^2 (-gml\,[M+m]) - \frac{z-1}{T}\,(gbml)} \quad (36)$$

4.2 Backward Euler

$$z = e^{sT} \approx \frac{1}{1+sT} \rightarrow s \approx \frac{z-1}{Tz} \quad (37)$$

$$G_1(z) = \frac{\frac{z-1}{Tz}}{\left(\frac{z-1}{Tz}\right)^3 \left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + \left(\frac{z-1}{Tz}\right)^2 \left(\frac{Ib}{ml} + lb\right) + \frac{z-1}{Tz}(-gM - gm) - gb} \quad (38)$$

$$G_2(z) = \frac{\left(\frac{z-1}{Tz}\right)^2 (I + ml^2) - mlg}{\left(\frac{z-1}{Tz}\right)^4 (MI + Im + Mml^2) + s^3 \left(Ib + mbl^2\right) + \left(\frac{z-1}{Tz}\right)^2 (-gml\,[M+m]) - \frac{z-1}{Tz}\,(gbml)} \quad (39)$$

4.3 Trapezoidal or Tustin

$$z = e^{sT} \approx \frac{1 + sT/2}{1 - sT/2} \rightarrow s \approx \frac{2\,(z-1)}{T\,(z+1)} \quad (40)$$

$$G_1(z) = \frac{\frac{2(z-1)}{T(z+1)}}{\left(\frac{2(z-1)}{T(z+1)}\right)^3 \left(\frac{MI}{ml} + \frac{I}{l} + Ml\right) + \left(\frac{2(z-1)}{T(z+1)}\right)^2 \left(\frac{Ib}{ml} + lb\right) + s(-gM - gm) - gb} \quad (41)$$

$$G_2(z) = \frac{\left(\frac{2(z-1)}{T(z+1)}\right)^2 (I + ml^2) - mlg}{\left(\frac{2(z-1)}{T(z+1)}\right)^4 (MI + Im + Mml^2) + \left(\frac{2(z-1)}{T(z+1)}\right)^3 \left(Ib + mbl^2\right) + \left(\frac{2(z-1)}{T(z+1)}\right)^2 (-gml\,[M+m]) - \frac{2(z-1)}{T(z+1)}\,(gbml)} \quad (42)$$

4.4 Discretizing using Matlab

Matlab has a built in function $c2d()$ that can discretize a continuous time transfer function. It only requires the transfer function and the sample time as an input. We using 0.0001 seconds as the sampling time. The Matlab code is shown below:

```
cart_n2 = (I+m*l^2)/q;
cart_n1 = 0;
cart_n0 = -g*m*l/q;
cart_d4 = 1;
cart_d3 = b*(I+m*l^2)/q;
cart_d2 = ((M + m)*m*g*l)/q;
cart_d1 = - b*m*g*l/q;
cart_d0 = 0;

pend_n1 = m*l/q;
pend_n0 = 0;
pend_d3 = 1;
pend_d2 = (b*(I + m*l^2))/q;
pend_d1 = -((M + m)*m*g*l)/q;
pend_d0 = -b*m*g*l/q;

P_cart = (cart_n2*s^2 + cart_n1*s + cart_n0)/(cart_d4*s^4 + cart_d3*s^3 + cart_d2*s
    ^2 + cart_d1*s + cart_d0)
P_pend = (pend_n1*s + pend_n0)/(pend_d3*s^3 + pend_d2*s^2 + pend_d1*s + pend_d0)


%% discretizing the transfer functions
d_P_cart = c2d(P_cart, 0.0001)
d_P_pend = c2d(P_pend, 0.0001)
```

This script puts out the discrete transfer function

$$\frac{2.273 \cdot 10^{-8} z^2 - 1.377 \cdot 10^{-13} z - 2.273 \cdot 10^{-8}}{z^3 - 3z^2 + 3z - 1} \tag{43}$$

[TODO - put in both equations]

5 System analysis

As a next step the transfer function of the systems shall be analysed using Matlab. This can be done using the command *pzmap()*.

```
%Plotting poles and zeros
figure
pzmap(P_cart, P_pend)
legend('cart','pendulum');

figure
pzmap(d_P_cart, d_P_pend)
legend('cart','pendulum');
```

This results in the following two figures.

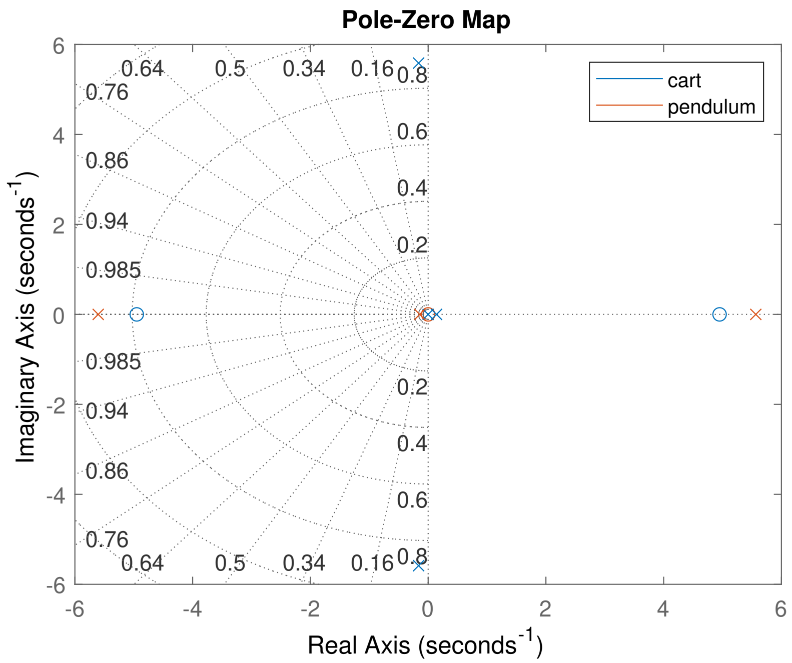


Figure 7: *pole and zero map of the continuous systems*

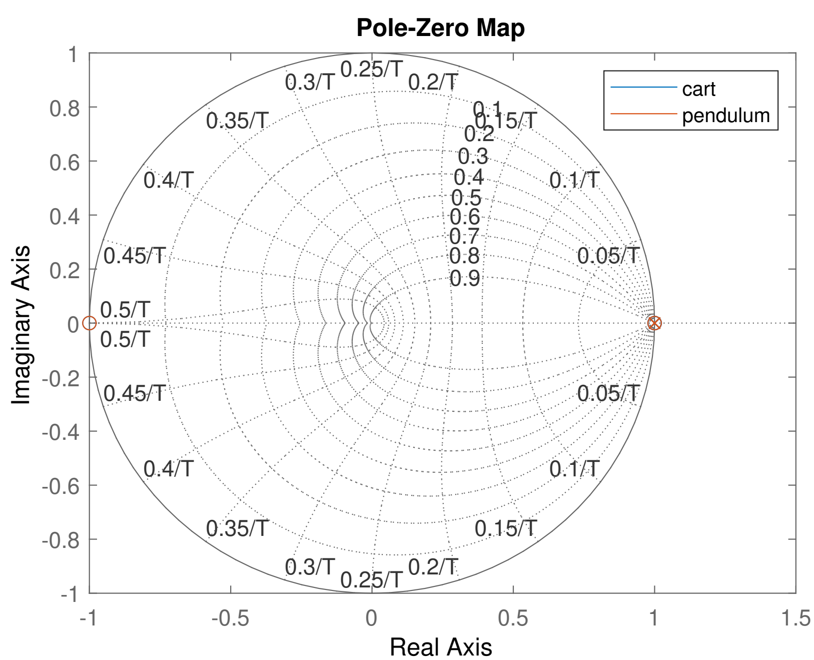


Figure 8: *pole and zero map of the discrete systems*

As figure 7 and 8 show both the cart and the pendulum are unstable no matter if discretized or not. For the continuous system this can be detected because the poles do not all have a negative imaginary part. Looking at the discrete system at first glance it looks like all poles are within or at last at the unit circle. However after zooming in (see Figure 9) it appears that a pole is outside of the unit circle which results in an unstable system.

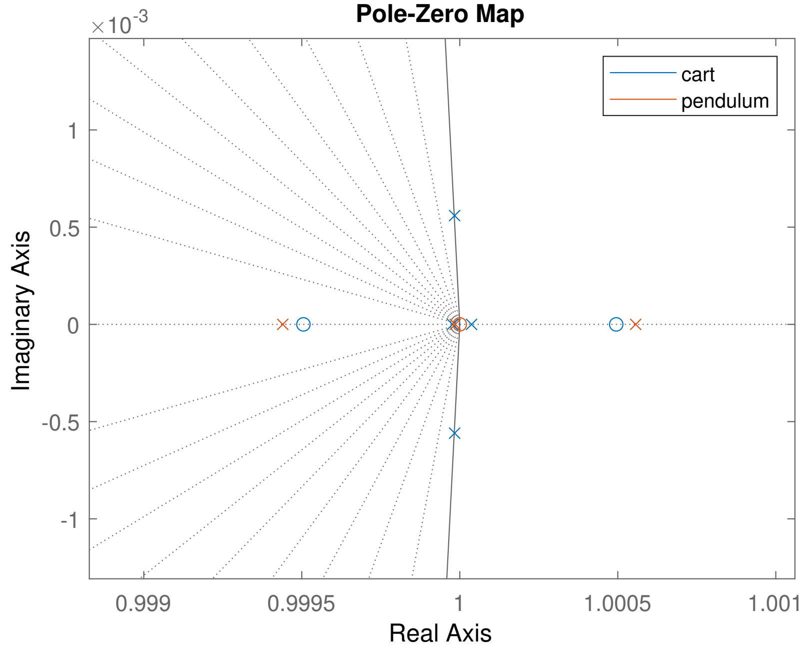


Figure 9: *zoomed in pole and zero map of the discrete systems*

Next the both the discrete and the continuous model of the linearized model were implemented in simulink and tested next to each other. In order to get the denominator and the numerator of the transfer function for simulink, Matlabs *tfdata* function was used.

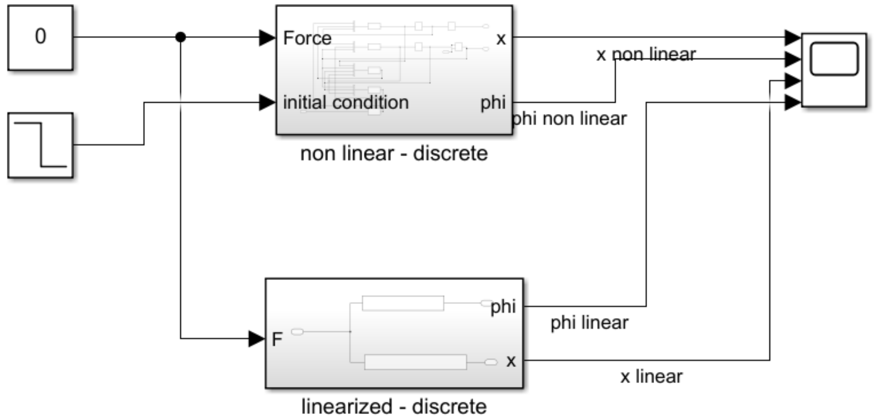


Figure 10: *linear and non linear version of the discrete system*

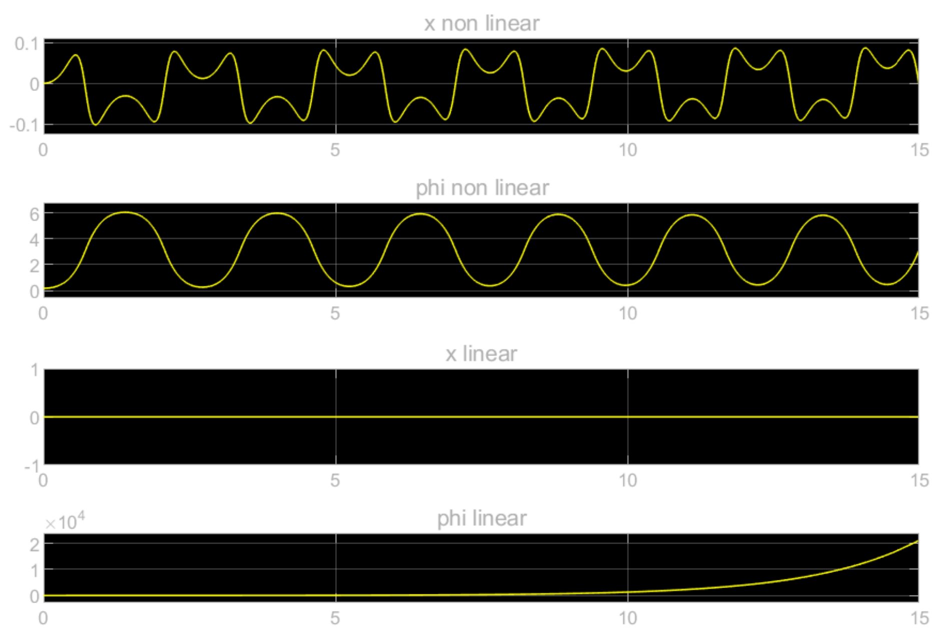


Figure 11: *simulation result of linear and non linear version of the discrete system*

As figure 11 shows, the non linear system behaves the way it should. The pendulum starts swinging and slowly decreases in height. Due to the inertia of the system the cart moves a bit. At first sight the linearized model looks to be wonrg. However, this is not the case as it shows an exponential function which is the solution for the differential equation we're trying to solve. the linearized model only works with small deviations and small time slots, so in order for it to behave correctly we need to implement a controller that gets executed regularly.

6 Control function

Now that we have a working model of the pendulum we are going to have to control the cart in such a way, that it always keeps the pendulum upwards. For this purpose two models are going to be developed: one using the continuous plant and the other using the discrete one. The controller we'll be using is a simply PID controller that simulink offers. In addition a Kalman filter will be implemented in order to provide a more plausible vertical position coming form the plant. Firstly, the provided Kalman filter was implemented using the Matlab function block and to test it the following model was built and simulated:

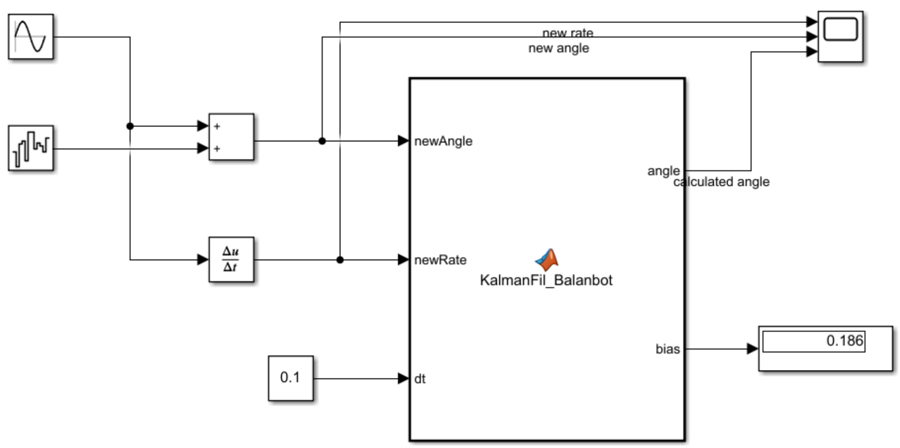


Figure 12: *Kalman filter test model*

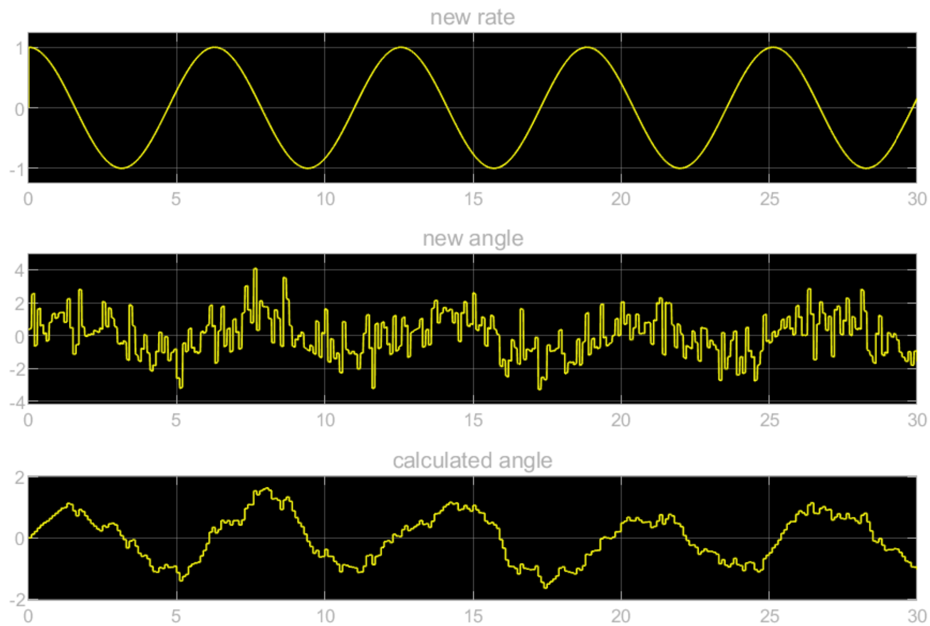


Figure 13: *Kalman filter test results*

As Figure 12 shows, the filter works quite well, so we can move onto implementing the PID controllers. Now two simulink models were developed. One with the discrete plant and one with the continuous one. The controller was in both cases discrete. For simulation purposes the Kalman filter was implemented using the deviation o fthe angle in order to gain the angular velocity.

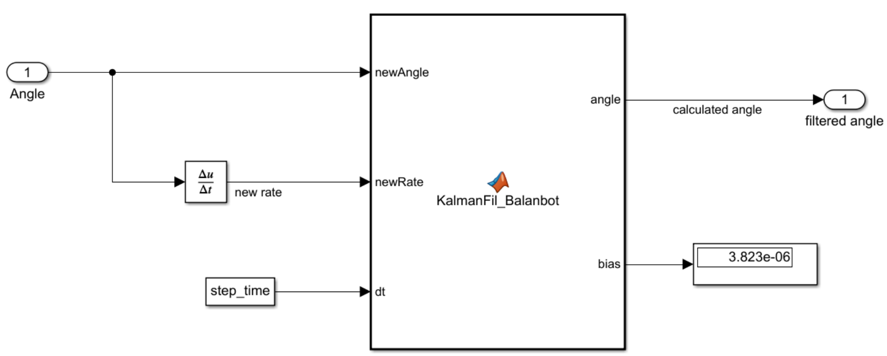


Figure 14: *Kalman filter for simulation*

Now the continuous system was built and simulated using a variable step solver.

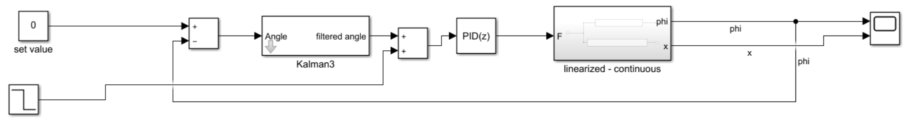


Figure 15: *linearized continuous system simulation setup*

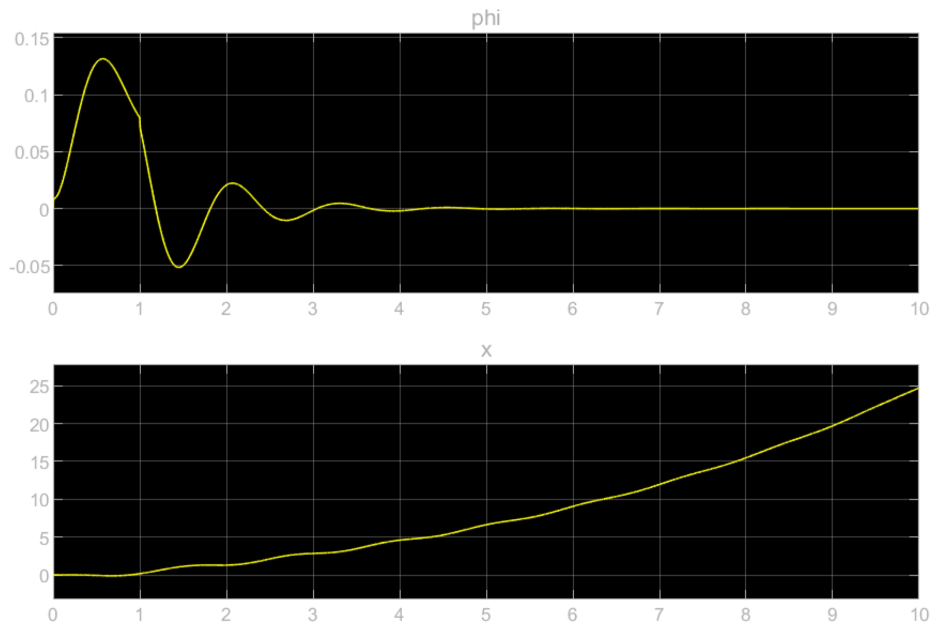


Figure 16: *linearized continuous system simulation results*

After that the same was done for the discrete system. This time a fixed step discrete solver was used.

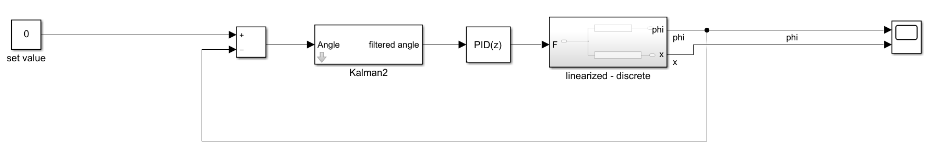


Figure 17: *linearized discrete system simulation setup*

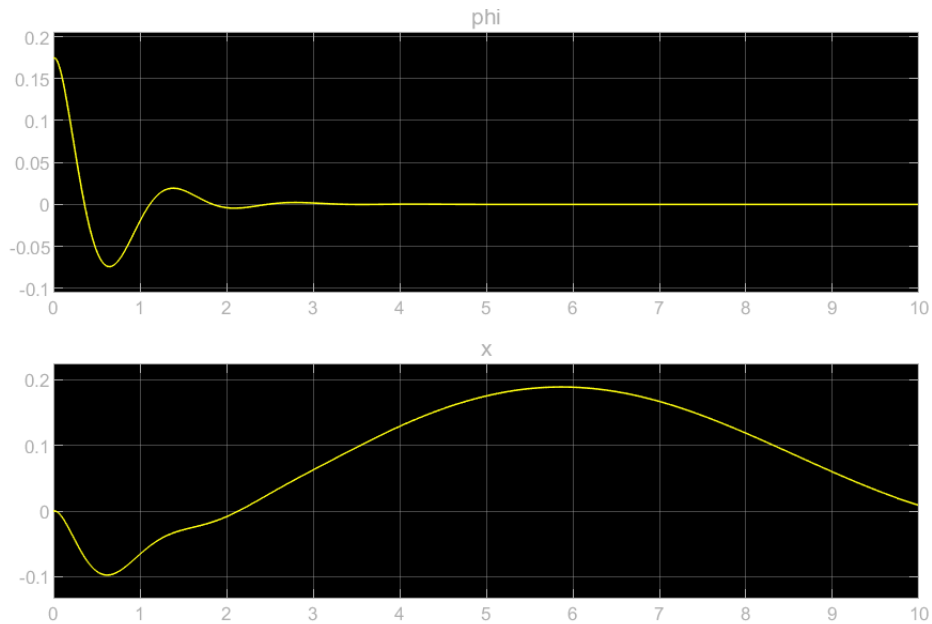


Figure 18: *linearized discrete system simulation results*

In both cases the models seem to behave the way they should, the controllers also seem to be working well so we can move on to the next step which is deploying the whole thing onto the actual hardware.

Part II

Laboratory Session 07

List of Figures