# Applied Robotics - FRTF20
# Project Group 10 - Fall 2020

Crane Damping
ABB IRB 140 Industrial Robot

Hicham Mohamad (hi8826mo-s)
Ola Nilsson (elt15oni, 19960517-9630)

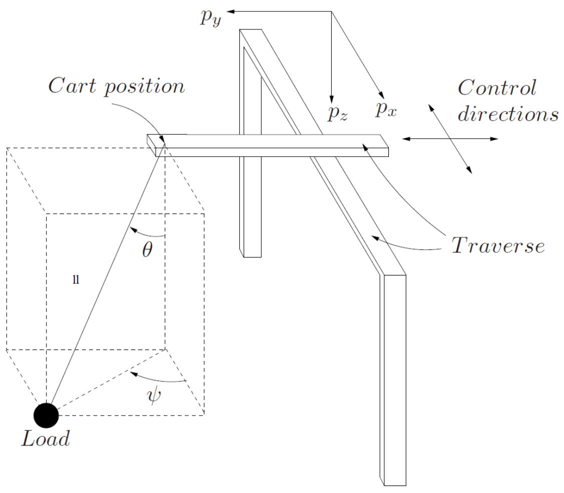
Supervisors
Anders Robertsson
Julian Salt Decaju

20 October 2020

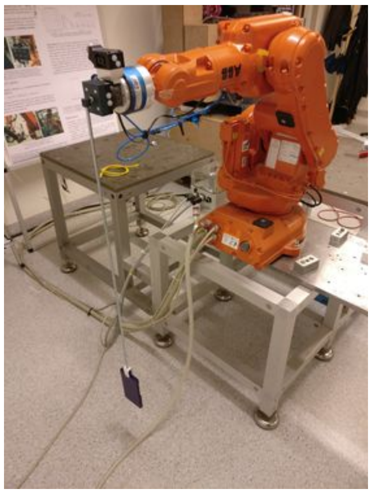
# Contents

## 1 Introduction

The purpose of this project is to design a controller for damping a crane with a swinging load. The system consists of the robot `ABB IRB 140` and a free swinging pendulum attached to the robot. The controller will be implemented and evaluated using MATLAB/Simulink, then the satisfying design will be tried on the real robot using the ABB controller `IRC5` and the LTH-made sensor software interface `ExtCtrl`, based on the networked architecture running on `Linux/Xenomai` platforms. By using an Inertial Measurement Unit IMU app called `IMU+GPS-Stream` together with a mobile phone, we extract information from both the accelerometer and the gyro to estimate and track the angle of the pendulum.



(a)



(b)

*Figure 1: a) Schematic overview of the crane layout and coordinates, where The process can be modelled as a hanging pendulum mounted to a motorized cart. The robot arm position, i.e., the pivot point of the crane load, can be moved in the $(p_x, p_y)$-plane. The control objective is to damp the load swinging in a planar/circular orbit while having the robot arm at a certain position. b) Crane robot ABB IRB 140 with the mobile phone attached to the pendulum.*

## 2 Physical Modeling

The system consists of a **robot arm** that is driven by ABB robot IRB140 and a free **swinging pendulum** that is attached to the pivot point of link 6 of the ABB robot, i.e robot arm. The process can be modelled as a hanging pendulum mounted to a **motorized cart**, as illustrated in Figure 1a.

A model of a system is a tool we use to answer questions about the system without having to do an experiment. All the relationships between quantities that can be observed in the system are described as mathematical relations in the model.

Modeling is performed to help with the development of the crane control. If the nonlinear model is somewhat similar to the real process and a good theoretical control strategy is found, then it is likely that the strategy works well on the real process . Then a linear model is needed because linear control theory is rich in results, e.g. pole placement or LQR which we need in this project. This model has to be as close to the nonlinear model as possible.

### 2.1 Model Description

As illustrated in the intuitive picture of the system in Figure 1a, the process can be modelled as a hanging pendulum mounted to a motorized cart. Hanging pendulums, like a crane payload and the oscillating arm inside clock, swing forth and back in a stable manner with **limited amplitude**. It is actually an example commonly found in control textbook and research literature. We can summarize the characteristics of the **cart-pendulum system**:

- The cart motion tries to keep the leaning angle, $\theta$, at a small level and damp the swinging pendulum.
- The dynamics of the pendulum system is nonlinear.

## 2.2 Robot arm dynamics - Motorized cart

As explained in [3], the robot arm/cart dynamics can be modeled as a **double integrator** from the acceleration reference $u$, which is our desired control signal, to the position $p$ of the robot arm, i.e.

$$\ddot{p} = a_{ref} = u \tag{1}$$

where $a$ is the horizontal acceleration of the pendulum pivot point. In fact, the cart is driven by a DC-motor which is controlled in a **cascaded structure**. Here we can summarize the characteristics of this cascaded structure of the cart controller:

- There is an **inner loop** that controls the current through the DC-motor.
- The current reference, $i_r$, is set by the **outer loop** that controls the cart velocity.
- The reference to the velocity control loop, $v_r$, is integrated from an **acceleration reference**, $u$, since acceleration is our desired control signal.
- The **current dynamics** are fast in comparison to the **velocity dynamics**, which makes $i_r \approx i$ a good approximation.
- When **fast closed loop dynamics** from $v_r$ to $v$ is desired, $v_r \approx v$ is a good approximation as well.
- Thus, in this way we have **double integrator dynamics** from control signal $u$ to cart position $p$.

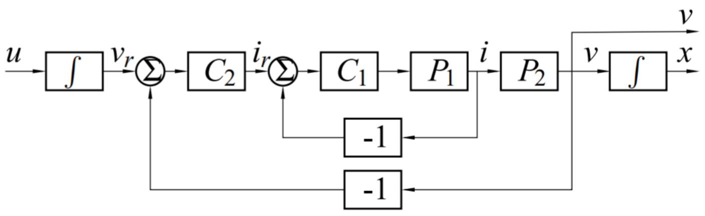


*Figure 2: Schematic view of the cascaded control structure for the cart control, as described in [3].*

#### Cart state space model

We introduce the state variables $p$ and $\dot{p}$ to create the state space model of the cart $z_c = (p \quad \dot{p})^T$

$$\dot{z}_c = A_c z_c + B_c a_{ref} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} z_c + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a_{ref} \tag{2}$$

#### Poles of the cart process

By using the characteristic polynomial, we are going to find the poles of the cart system. As we know, the poles of a system are equal to the eigenvalues of the A matrix, if the system is **controllable** and **observable**. For the cart system, the characteristic equation is computed for obtaining the corresponding poles

$$det(\lambda I - A_c) = 0 \implies det \begin{pmatrix} \lambda & -1 \\ 0 & \lambda \end{pmatrix} = 0 \implies \lambda^2 + 1 = 0 \implies \lambda = \pm i \tag{3}$$

## 2.3 The nonlinear pendulum model

For the **planar swinging process**, we consider a two dimensional problem where the pendulum is constrained to move around the vertical plane. For this system, the control **input** is the DC motors driving voltage represented by the **force** that moves the cart pivot horizontally whereas the **outputs** are the angular position of the pendulum $\theta$ and the horizontal position of the cart $p$, i.e. the x-direction as shown in Figure 3a.

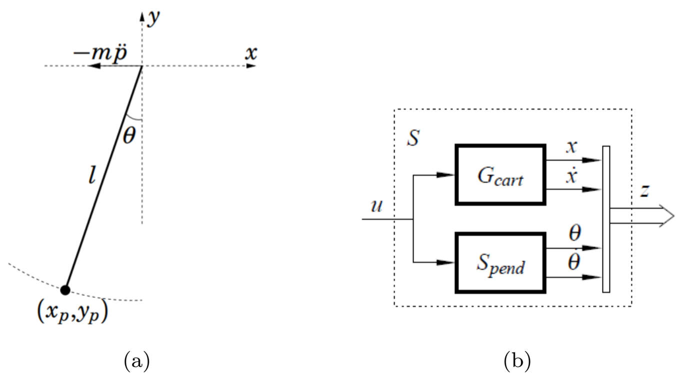


*Figure 3: a) The two dimentional pendulum problem. b) Schematic view of the system cart-pendulum. [3]*

**Dynamics: Euler-Lagrange Equations**

Here we assume that the reactions forces which describe the interaction between the pendulum and the cart are neglected since the load is of small dimension and they can be attenuated by the inner controller. To get the dynamic equations of the system, it is often convenient to rewrite the second Newton law using Lagrange mechanics. First, we compute the equation for the **kinetic energy** K

$$K = \frac{1}{2}\, m\, (\dot{p_x}, \dot{p_y})^2 \tag{4}$$

Then, the one for the **potential energy**

$$V = m\ g\ y_p \tag{5}$$

Here by considering the pendulum angle $\theta$ as the **generalized coordinate**, we can express $x_p$ and $y_p$ as function $r$ of the only one generalized coordinate $\theta$. Thus, we can write

$$\begin{aligned} x_p &= r_x(\theta) = -l\sin(\theta) \\ y_p &= r_y(\theta) = -l\cos(\theta) \end{aligned} \tag{6}$$

Now we can rewrite kinetic and potential energy in the generalized coordinates

$$\begin{aligned} K &= \frac{1}{2} m l^2 \dot{\theta}^2 \\ P &= -mgl\cos(\theta) \end{aligned} \tag{7}$$

we now introduce a scalar function called the **Lagrangian** $L(\theta, \dot{\theta}) = K - P$ and write the Euler-Lagrangian equation to get the **equation of motion** in compact form given by

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = \tau = F_x \frac{\partial r_x(\theta)}{\partial \theta} = m\ddot{p}l\cos\theta \tag{8}$$

Substituting K and P in Euler-Langrange equation we get

$$\frac{d}{dt}(ml^2\dot{\theta}) + mgl\sin\theta = m\ddot{p}l\cos\theta \tag{9}$$

Solving this equation, we get the governing equation for the pendulum system

$$\ddot{\theta} = \frac{-g}{l}\sin\theta + \frac{\ddot{p}}{l}\cos\theta \tag{10}$$

## Complete system dynamics - Notations

The approximate model for the complete system dynamics can now be written as

$$\begin{aligned} \ddot{p} &= u = a \\ \ddot{\theta} &= \frac{-g}{l}\sin\theta + \frac{u}{l}\cos\theta \end{aligned} \tag{11}$$

Here it is nice to define the **notations** used in the model above

- $\theta$ the pendulum angle, where $\theta = 0$ is defined to be the pendulum downward position.
- $l$ is the pendulum length.
- $g$ is the gravitational acceleration
- $a$ is the **horizontal acceleration** of the pendulum pivot point.
- $u$ the horizontal force/torque exerted by the DC motor on the robot arm. The horizontal acceleration $a$ is equal to the cart acceleration $\ddot{p}$ and consequently equal to the control signal $u$.

## 3 Linearisation

A schematic view of the full system is illustrated in Figure 3b, where $S_{pend}$ represents the non-linear pendulum dynamics in 10 and $G_{cart}$ represents the cart dynamics in 1, $z$ is a vector containing the states, and S represents the full system. We can linearize our pendulum equation in two different methods:

1. The pendulum equation determined above can be linearised about the vertically downward equilibrium position, $\theta = 0$, by computing the **partial derivatives** (Jacobians).
2. However, assuming that the system stays within a small neighborhood of the equillbrium point, we can use the following **small angle approximations** of the nonlinear functions in our pendulum equation to linearize it, i.e.

$$\cos\theta \approx 1 \tag{12}$$
$$\sin\theta \approx \theta \tag{13}$$

   This assumption can be reasonably valid if we design the controller such that the pendulum do not deviate too much from the vertically downward equilibrium position. By substituting the approximations mentioned above we get the **linearized equations of motion**:

$$\ddot{\theta} = -\frac{g}{l}\theta + \frac{u}{l} \tag{14}$$

### State space model of the pendulum

To create the state space model of the pendulum, we introduce the state variables $z_p = (\theta \quad \dot{\theta})^T$. Then the state space model becomes

$$\dot{z_p} = \begin{pmatrix} 0 & 1 \\ -g/l & 0 \end{pmatrix} z_p + \begin{pmatrix} 0 \\ 1/l \end{pmatrix} u \tag{15}$$

### Poles of the pendulum

After computing the linearised equation of the pendulum system in 14, we can find the **complex conjugated eigenvalues/poles** of the system. For solving this differential equation, we need the **homogeneous equation**

$$\theta_{hom} = \ddot{\theta} + \frac{g}{l}\theta = 0 \tag{16}$$

By setting $\theta = e^{\lambda t}$, we can write

$$(\lambda^2 + \frac{g}{l})e^{\lambda t} = 0 \implies \lambda^2 + \frac{g}{l} = 0 \implies \lambda_{1,2} = \pm i\sqrt{\frac{g}{l}} \tag{17}$$

This results in an **undamped oscillation** because we have complex conjugated poles on the imaginary axis ($Re\{\lambda_i\}$ are not negative).

### Pendulum length extraction

In the modeling of the pendulum a stiff and undampened pendulum is assumed. In reality, the phone mounted to the pendulum shifts the mass center and thus changes the frequency of which the pendulum oscillates at. In order to compute the "equivalent length" of the pendulum an experiment was performed. The pendulum was let go from an angle and the IMU-data collected. From the measurements the frequency of the pendulum could be extracted by measuring the time between two peaks in the measured acceleration. Since the pendulum is assumed to be undampened the frequency of the oscillations directly correspond to the locations of the complex pole pair on the imaginary axis in the s-plane.

In this way, after extracting the frequency of the oscillatory pendulum in Matlab, we use the resulting values of the poles in 17 to find the length $l$ of the pendulum

$$w = 2\pi f = \lambda = \sqrt{\frac{g}{l}} \implies l = \frac{g}{w^2} \approx 0.85m \tag{18}$$

### The full state Space Model

Linearisation of the complete system dynamics, results in the following state space model

$$\dot{z} = Az + Bu = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -g/l & 0 \end{pmatrix} z + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1/l \end{bmatrix} u \tag{19}$$

and the output equations are as follows:

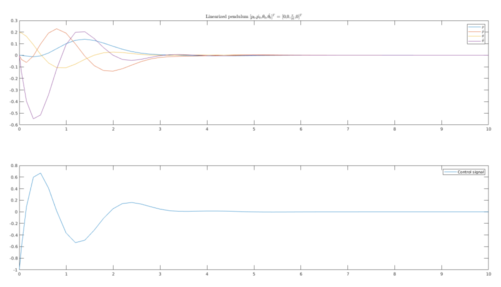
$$y = \begin{pmatrix} p \\ \theta \end{pmatrix} = Cz + Du = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} z + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \tag{20}$$

by introducing the following four states

$$z = (z_1, z_2, z_3, z_4)^T = (p, \dot{p}, \theta, \dot{\theta})^T \tag{21}$$

## 4 Motivations for Modeling and Process Characteristics

The models designed for this project was first an exact nonlinear model of the cart-pendulum system based on the Lagrange mechanics. This model was then linearised around the downward position of the pendulum ($\theta = 0$) to create another model that would be the base of the control design. To verify our linearised model, we implement the controller for both models, linearised and nonlinear, to see if they react the same. The more their response will match, the more the linearized system will be a good approximation of our nonlinear one.


(a)

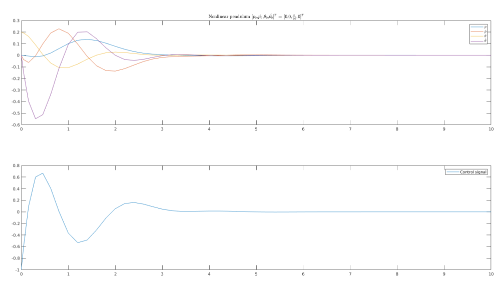

(b)

*Figure 4: LQ control of the cart-pendulum system. a) Linear simulation. b) Nonlinear simulation.*

### Process Characteristics - Matlab Simulink

As in all control problems, to get familiar with the process dynamics before attempting to control it, we need to study the characteristics of the built model by simulating it in Matlab/Simulink.

- Open-loop response

## 5 Network Communication - Orca/ExtCtrl

In this section we need to describe the setup used to communicate with the robot and how to send data over the network to the robot control system `ExtCtrl` or receive back, as illustrated in Figure 5. **Three parts** are necessary to set up a proper communication to the robot control system:

1. ***.lc-file** to define the variables to send and receive on network.
2. **Matlab/Simulink model** to set up the connection between Orca and ExtCtrl.
3. ***.c-file** to manipulate signals or to introduce measurement devices.

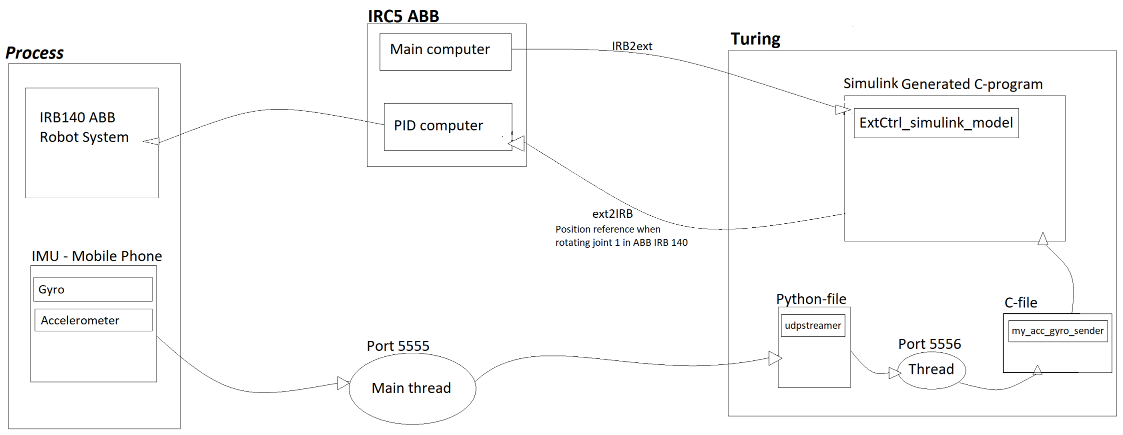


*Figure 5: Diagram showing the main hardware components and communication set-up to the robot in the project.*

We can summarize the important communication, illustrated in Figure 5, between the robot and the other hardware components in the following steps:

- The mobile phone is connected to the WiFi-network `Robotlab_local` and uses the app **IMU+GPS-Streamer** to send the IMU data over User Datagram Protocol **UDP**.
- The data is sent to the computer **Turing** which is running a python program that receives the data on port 5555.
- The python program extracts the accelerometer- and gyro-data and transmit the extracted data through port 5556 over UDP to a C-program running on the same machine.
- The C-program receives the data and writes it to a vector defined in the *.lc file.
- The data vector is read in the `ExtCtrl_simulink_model` and fed to the implemented controller.
- The controller sends the tracked **position reference** to the robot that is used to stabilize the pendulum.

## 6 Angle estimation using IMU (gyro and accelerometer)

In this project, we are interested in tracking the tilt-angle of the pendulum relative to the vertical downward position. For this reason, we use a mobile phone which is usually equipped with an accelerometer for measuring **linear acceleration** in the same axes, and a gyro for measuring **angular velocity** around the three axes x, y, z. An Inertial Measurement Unit (IMU) app is installed on the mobile in order to gather these measurements.

**Accelerometer** The way the magnitude of the acceleration $\sqrt{a_x^2 + a_y^2 + a_z^2} = 9.8m/s^2$ is distributed on the three body-fixed accelerations $a_x$, $a_y$, $a_z$ will give us information about the absolute orientation. When the mobile phone device, attached to the pendulum, is kept still in $0°$ position, the measurements is $a_x = 0$, $a_y = 9.8$, $z_z = 0$, since the acceleration is measured relative to free-fall. Thus, our tilt angle $\theta$ should be related to the relative size of the components $a_x$ and $a_y$ when the pendulum is oscillating in planar orbit.

**Gyro** Since the gyro give us the angular velocity we need to integrate to obtain the rotation angle. In this way, the bias errors in the sensor will lead to a **growing error** in the angle estimate, compared to a constant error from the accelerometer. in our problem, the gyro signal will regard the rotation around z-axis because we have oscillation in the x-y plane.

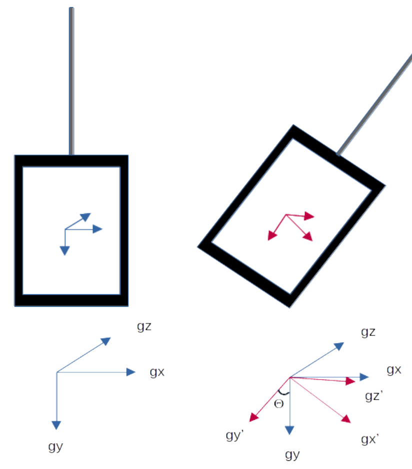


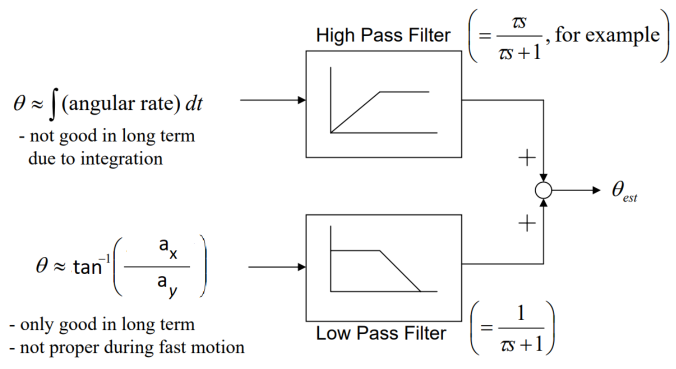
*Figure 6: Mounting of phone on pendulum.*

### Complementary filter

Often, it is usefull to use a complementary filter in the case when there are two different measurement sources for estimating one variable and the noise properties of these two measurements are such that one source gives good information only in **low frequency region** while the other is good only in **high frequency region**.

As mentioned in [6], We need to combine the merits from the two sensors and avoid the drawbacks of them using the so called complementary filter. Thus, by extracting information from both the accelerometer and the gyro and combining these using low-pass and high-pass filters, we will create a better estimate of the angle of the pendulum, compared to a naive approach of using only the gyro, or only the accelerometer.

As shown in Figure 7, the accelerometer solution is low-pass filtered, while the gyro solution is high-pass filtered. The name complementary filter comes from the fact that the two filters sum to 1.



High Pass Filter $\left(= \frac{\tau s}{\tau s + 1}, \text{for example}\right)$
$\theta \approx \int (\text{angular rate})\, dt$
- not good in long term due to integration
+
$\theta_{est}$
+
$\theta \approx \tan^{-1}\left(\frac{a_x}{a_y}\right)$
- only good in long term
- not proper during fast motion
Low Pass Filter $\left(= \frac{1}{\tau s + 1}\right)$

*Figure 7: The complementary filter design here is to take slow moving signals from accelerometer (Static conditions) and fast moving signals from a gyroscope (Dynamic conditions) and combine them.*

# 7 Control Strategy

There are many ways to determine a linear controller for the system. In this project, it is suggested to try LQ-design and pole placement, illustrated in Figure 10. However, because of lack of time we could only study and implement LQR design. In the following subsections, we have a short description of both design approaches. The simulations will serve to debug the implementation of the controller and to tune the design parameters so that satisfactory performance is achieved.

To suppress a possible model error, it is often desired to introduce an **integral action** in the controller. Consequently, in such case, we need to take into account an **anti-windup** as the solution to an eventual problem of integral windup by preventing the integral part from growing when the control signal is saturated.

## 7.1 LQ-design

As we know, in LQ/LQG control we need to minimize the familiar cost function:

$$J = \int_0^\infty (x^T(t)Qx(t) + u^T(t)Ru(t))dt \tag{22}$$

Since in our problem, we use complimentary filtering to estimate the tilt angle such that all states are measurable, we don't need to compute the Kalman filter but it is enough to compute the **optimal feedback law**. Thus, the regulator is then **pure state feedback**

$$u = -L\ \mathbf{x} \tag{23}$$

When the results are not satisfactory, the problem formulation has to be tuned, i.e. we try other diagonal penalty matrices $Q_1$ when $R = I$.

In this project we implement optimal linear state feedback for stabilizing the pendulum. This is done by computing the optimal gain vector L that minimizes the cost function J given the weights $Q$ and $R$ where $Q$ corresponds to penalty of state deviations and $R$ corresponds to penalty of the use of control actuation. The resulting L that minimizes the cost function J is used as linear state feedback in the controller.

$$J = \int_0^\infty (x^T(t)Qx(t) + u^T(t)Ru(t))dt = \int_0^\infty (x^T(t)Qx(t) + u^2(t))dt \tag{24}$$

## 7.2 LQ computations

The $Q$ matrix and $R$ was chosen as follows:

$$Q = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}, R = 1$$

The reasoning behind the chosen weights is that it is desirable to keep the pendulum close to the downward equilibrium, however the controller should also be able to follow a reference trajectory when moving the crane. In order to compute $L$ the MATLAB function `lqr()` was used which takes the dynamics of the system (A and B matrices) and the weights $Q$ and $R$ and then computes $L$. The resulting optimal gain vector L then becomes $L = \begin{bmatrix} 2.2361 & 2.6999 & 5.0890 & 1.8570 \end{bmatrix}$ which places the poles of the system in:

$$p = \begin{bmatrix} -1.3642 + 3.1309i \\ -1.3642 - 3.1309i \\ -1.0764 + 1.0260i \\ -1.0764 - 1.0260i \end{bmatrix}.$$

Simulations of both the linear and nonlinear dynamics with the same linear state feedback law can be seen in figure 8 and 9 respectively. In both simulations the controller is able to stabilize the pendulum and drive all states to zero.

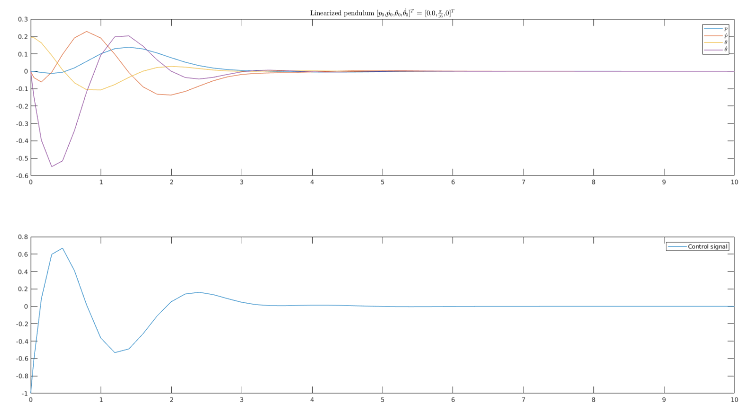


*Figure 8: Simulation of the linearized pendulum with state feedback.*

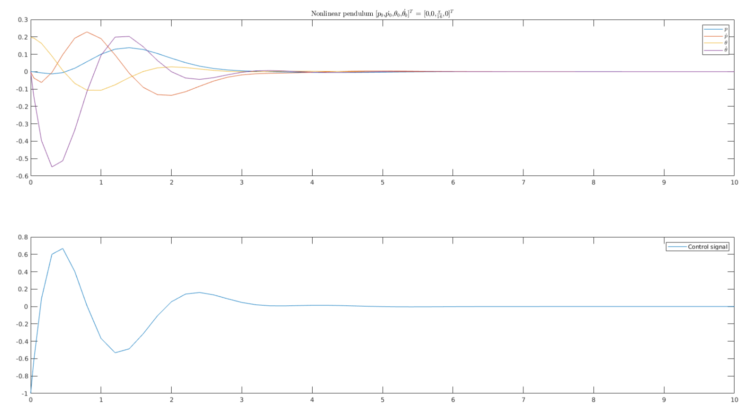


*Figure 9: Simulation of the nonlinear pendulum with state feedback.*

## 7.3 Pole placement design

Having measurements of all states available allow for state feedback control. The state feedback can be designed to meet different performance criteria. Here the task is to design the feedback controller so as to obtain desired poles in the closed-loop system, i.e., pole placement design, as illustrated in Figure 10b. It is often useful to keep the following pole placement specifications:

1. The system should be as fast as possible.
2. The overshoot should be less than 5%.
3. When reference signal $r$ is a unit step the control signal must not exceed $u_{max}$.
4. The static gain of the closed loop system $G_c(0)$ should be equal to 1.

In this way, we use the state feedback, illustrated in Figure 10b, and pole placement to find a control law

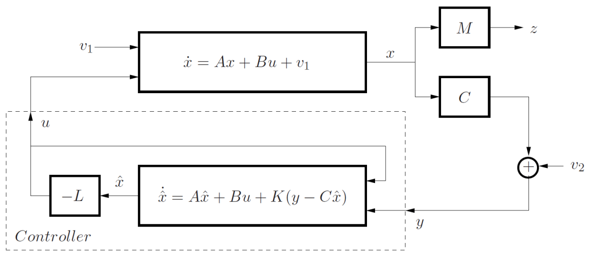
$$u = -Lx + l_0 r \tag{25}$$

which gives a closed loop system that satisfies the above requirements. Here, $l_0$ will be chosen such that the static gain of the closed loop system $G_c(0) = 1$.

Consequently, the system matrix for the closed loop system of the pendulum becomes

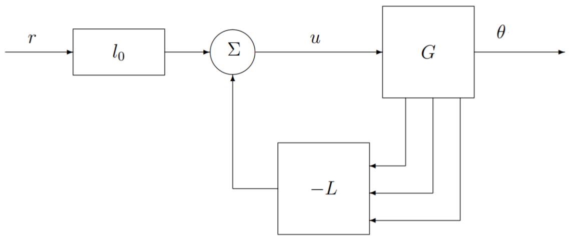
$$\dot{z_p} = A_p z_p + B_p u = A_p z_p + B_p(-L z_p + l_0 r) = (A_p - B_p L) z_p + B_p l_0 r \tag{26}$$

This can be computed in Matlab using the function `L = place(A,B,p)` which places the desired closed-loop poles $p$ by computing a state-feedback gain matrix L.



$v_1$
$\dot{x} = Ax + Bu + v_1$
$x$
$M$
$z$
$C$
$u$
$v_2$
$\hat{x}$
$-L$
$\dot{\hat{x}} = A\hat{x} + Bu + K(y - C\hat{x})$
$y$
Controller

(a)

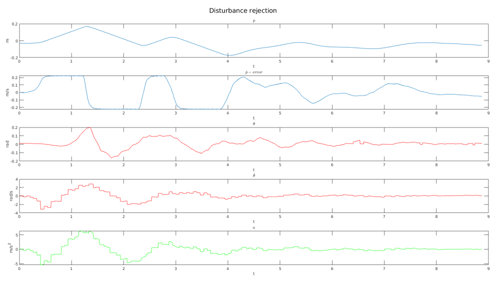


(b)

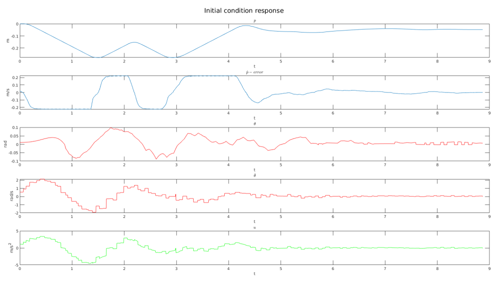
*Figure 10: a) Control structure when using LQG controller on a process. The variable $y$ is the measurements, $z$ controlled variables, $x$ states of the process, $\hat{x}$ estimated states, $u$ control signal, $v_1$ process noise and $v_2$ measurement noise. b) Control system with state feedback design controller used to obtain desired poles in the closed-loop system, i.e., pole placement design.*

## 8 Results

In this section the results of implementing the controller and conducting experiments on the ABB IRB140 robot is presented. The experiments consist of measuring the performance of the controller when it handles disturbance rejection and oscillations of the load. The **disturbance experiments** consist of pushing the pendulum at the downward equilibrium. The **oscillation experiments** consist instead of raising the pendulum and letting the pendulum fall from an angle. While the pendulum was swinging freely the controller was activated. The results of these experiments can be seen in figure 11a and figure 11b. The graphs shows the state deviations from the reference values and the control signal $u$.



(a)



(b)

*Figure 11: a) Disturbance rejection. b) Damping of oscillations, Initial value response.*

## 9 Conclusion

From the results of the experiments the controller succeeded to stabilize the pendulum and drive the error between reference values and the states $(p, \dot{p}, \theta, \dot{\theta})$ to zero in a satisfying manner.

The models designed for this project was first an exact nonlinear model of the cart-pendulum system based on the Lagrange mechanics. This model was then linearised around the downward equilibrium position of the pendulum ($\theta = 0$) to create another model that would be the base of the control design. To evaluate the performance of our linearised model, we implemented the controller for both models, linearised and nonlinear, and we verified that they react the same. In this way, we could conclude that the linearized system is a good approximation of the nonlinear real process.

### Further development

This project could be further improved on. The system has been modeled as a **planar** swinging pendulum attached to a cart. An extension of this project could be instead to model **spherical** pendulum movement when the cart can move in 2 dimensions. In our model some assumptions have been made about the pendulum, for instance that the pendulum is stiff. In reality the load is attached to a cable that is flexible.

## References

[1] Anders Robertsson, Applied Robotics - FRTN20, Lectures notes, LTH

[2] Karl J. Åström, Richard M. Murray, Feedback Systems: An Introduction for Scientists and Engineers, Second Edition, Princeton University Press 2020

[3] Pontus Giselsson, Johan Åkesson, Anders Robertsson. *Optimization of a Pendulum System using Optimica and Modelica*, Proceedings 7th Modelica Conference, Como, Italy, 2009.

[4] Laboratory exercise in Multivariable Control, Department of Automatic Control LTH.

[5] Laboratory Exercise in Nonlinear Control and Servo Systems by Pontus Giselsson, 2009

[6] Angle estimation using gyros and accelerometers. Lab in Dynamical systems and control TSRT21, Automatic Control, LiU

[7] S Park, J How, Complementary Filtering. Lecture Notes, MIT: `https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-333-aircraft-stability-and-control-fall-2004/lecture-notes/lecture_15.pdf`

[8] L. Freidovich, Control Methods for Robotic Applications, 2017

[9] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics — Modelling, Planning and Control, Springer Advanced Textbooks in Control and Signal Processing Series, London, UK, 2009. Italian edition: Robotica — Modellistica, Pianificazione e Controllo, McGraw-Hill Libri Italia, Milano, 2008

[10] P.I. Corke, Robotics, Vision and Control, Springer, ISBN 978-3-319-54413-7,

[11] ABB Robotics (2004): "Operating manual — IRC5 with FlexPendant", Document ID: 3HAC16590-1, Revision: P