# Identification and Control of the Furuta pendulum

Wesley Nijhuis, Geer Chen

## I. Introduction

The Furuta pendulum, invented by Katsuhisa Furuta, is an inverted pendulum that has been thoroughly studied since its conception in 1992. The pendulum is of academic interest due to its nonlinear and underactuated dynamics to develop and demonstrate linear and nonlinear control methods. The system consists of a rotary arm actuated by a motor and a pendulum mounted on the end of the arm. The pendulum is free to wing in a plane that is perpendicular to the arm. For this project a Qube Servo 2 Furuta pendulum was used.

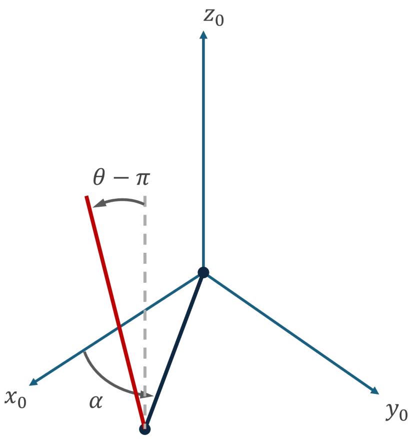


Fig. 1: Simplified representation of the Furuta pendulum.

The following objectives for the control of this Furuta pendulum were given:

1) The controller should be able to stabilize the system around it's unstable equilibrium, with minimal oscillatory behaviour.
2) The controller should be able to track a reference signal
3) The controller should be able to compensate for disturbances

As this project is intended to be an exercise in integrating all knowledge accumulated in the first year of the Master Systems & Control at TU Delft, two additional objectives were created to add generalizability to the experience gained.

In this project, an initial set of parameters was given to function as an initial guess for the identification of the system. Furthermore, the first principle model of the Furuta pendulum is well studied and relatively easy to construct. Both the initial set of parameters and the first principle model should always be used if these are available, but numerous cases can be thought of in industry or research where having this luxury cannot be assumed. Therefore we set the additional objective of not using the supplied initial parameter guesses, and utilizing an identification method that is not dependent on a near perfect first principle model.

In chapter II a the model of the Furuta pendulum is derived via the Euler-Lagrange method. In chapter III the calibration procedure is documented to validate sensor readings and account for motor dead-zone. Chapter IV will contain a description of the identification process and model validation. The framework that is used is then summarized in algorithm XII-A. Control synthesis will begin in chapter VI with an LQR controller. A swing-up controller will then be added to the LQR in chapter VII. To finalize controller synthesis, a Model Predictive Controller is implemented in chapter VIII.

Wesley Nijhuis and Geer Chen are master students at TU Delft, The Netherlands. E-mail addresses: {W.P.A.Nijhuis@student.tudelft.nl, G.Chen@student.tudelft.nl}.

## II. Model

The Furuta system consists of a rotary arm and an inverted pendulum. Define the angle of rotation of the rotary arm around a fixed vertical axis $z_1$ as $\alpha$. The pendulum is mounted at the end of the arm and rotates around a horizontal axis with angle $\theta$. $\theta = 0$ corresponds to the downwards position. $\theta = \pi$ corresponds to the upwards position. The center of mass of pendulum is located at its midpoint, at a distance $L_p/2$ from the pivot.

The moments of inertia of rotary arm and pendulum arm are defined as

$$\mathbf{J}_r = \begin{bmatrix} J_{rxx} & 0 & 0 \\ 0 & J_{ryy} & 0 \\ 0 & 0 & J_{rzz} \end{bmatrix}, \quad \mathbf{J}_p = \begin{bmatrix} J_{pxx} & 0 & 0 \\ 0 & J_{pyy} & 0 \\ 0 & 0 & J_{pzz} \end{bmatrix}.$$

Define a global coordinate system of the frame $\mathcal{F}_0 = (x_1, y_1, z_1)$, with $z_1$ pointing upwards. The following two rotation matrices are used to map the coordinate system of the rotary arm, frame $\mathcal{B}$, to the inertial frame $\mathcal{F}_0$. The matrix $\mathbf{R}_1$ represents the rotation of the rotary arm about the vertical axis $z_1$. It is given by

$$\mathbf{R}_1 = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The matrix $\mathbf{R}_2$ represents the rotation of the pendulum about its local axis $z_2$, and is multiplied by a fixed axis-mapping transformation matrix that reorients the local frame of the pendulum to align with the global coordinate system. The combined transformation is given by:

$$\mathbf{R}_2 = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & \sin(\theta) & -\cos(\theta) \\ 0 & \cos(\theta) & \sin(\theta) \\ 1 & 0 & 0 \end{bmatrix}.$$

Defining the angular velocity of the rotary arm as $\omega_r = \begin{bmatrix} 0 & 0 & \alpha \end{bmatrix}^\top$, the angular velocity of the pendulum arm is then given by

$$\boldsymbol{\omega}_p = \mathbf{R}_2 \boldsymbol{\omega}_r + \begin{bmatrix} 0 & 0 & \dot{\theta} \end{bmatrix}^T = \begin{bmatrix} -\cos(\theta)\dot{\alpha} & \sin(\theta)\dot{\alpha} & \dot{\theta} \end{bmatrix}^T. \tag{7}$$

The linear velocity of the center of mass of rotary arm is given by

$$v_r = \boldsymbol{\omega}_r \times \begin{bmatrix} L_r/2 & 0 & 0 \end{bmatrix}^T = \begin{bmatrix} 0 \\ \dot{\alpha} L_r/2 \\ 0 \end{bmatrix}$$

The linear velocity of the joint between rotary arm and pendulum arm is

$$\mathbf{v}_j = \mathbf{R}_2 \left( \boldsymbol{\omega}_r \times \begin{bmatrix} L_r & 0 & 0 \end{bmatrix}^T \right) = \begin{bmatrix} \dot{\alpha} L_r \sin(\theta) & \dot{\alpha} L_r \cos(\theta) & 0 \end{bmatrix}^T.$$

So the linear velocity of the center of mass of pendulum is given by

$$\mathbf{v}_p = \mathbf{v}_j + \boldsymbol{\omega}_p \times \begin{bmatrix} L_p/2 & 0 & 0 \end{bmatrix}^T = \begin{bmatrix} \dot{\alpha} L_r \sin(\theta) \\ \dot{\alpha} L_r \cos(\theta) + \dot{\theta} L_p/2 \\ -\dot{\alpha} \sin(\theta) L_p/2 \end{bmatrix}.$$

Thus, the translational kinetic energy of the rotatry arm and pendulum is given by:

$$T_{\text{trans}} = \frac{1}{2}\left( \mathbf{v}_r^T m_r \mathbf{v}_r + \mathbf{v}_p^T m_p \mathbf{v}_p \right) = \frac{1}{2}\left( \frac{1}{4}\dot{\alpha}^2 m_r L_r^2 + \dot{\alpha}^2 m_p L_r^2 + \frac{1}{4}\dot{\theta}^2 m_p L_p^2 + m_p L_r L_p \cos(\theta)\dot{\alpha}\dot{\theta} + \frac{1}{4}\dot{\alpha}^2 \sin^2(\theta) L_p^2 m_p \right)$$

The rotational kinetic energy is:

$$T_{\text{rot}} = \frac{1}{2}\left( \boldsymbol{\omega}_r^T \mathbf{J}_r \boldsymbol{\omega}_r + \boldsymbol{\omega}_p^T \mathbf{J}_p \boldsymbol{\omega}_p \right) = \frac{1}{2}\dot{\alpha}^2 \left( J_r zz + J_p yy \sin^2(\theta) + J_p xx \cos^2(\theta) \right) + \frac{1}{2}\dot{\theta}^2 J_p zz$$

Total kinetic energy is:

$$T = T_{\text{trans}} + T_{\text{rot}}$$

The potential energy of the system is given by

$$V = 0 + \frac{1}{2} L_p m_p g (1 - \cos(\theta))$$

To obtain the equations of motion for the pendulum, the Euler-Lagrange method is used. Resulting in the Lagrangian

$$\begin{aligned}\mathcal{L} := T - V = \frac{1}{8}\dot{\alpha}^2 m_r L_r^2 + \frac{1}{2}\dot{\alpha}^2 m_p L_p^2 + \frac{1}{8}\dot{\theta}^2 m_p L_p^2 + \frac{1}{2} m_p L_r L_p \cos(\theta)\dot{\alpha}\dot{\theta} + \frac{1}{8}\dot{\alpha}^2 \sin^2(\theta) L_p^2 m_p \\ + \frac{1}{2}\dot{\alpha}^2 \left(J_{rzz} + J_{pyy}\sin^2(\theta) + J_{pxx}\cos^2(\theta)\right) + \frac{1}{2}\dot{\theta}^2 J_{pzz} - \frac{1}{2} L_p m_p g(1-\cos(\theta))\end{aligned}$$

The Euler-Lagrange equation is represented as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) + b_i \dot{q}_i - \frac{\partial L}{\partial q_i} = Q_i$$

Here the generalized coordinates are defined as $q = [\theta \quad \alpha]^T$. The non-conservative forces acting on the generalized coordinates are given by

$$Q = [-\mu_\theta \dot{\theta} + \tau_m \quad -\mu_\alpha \dot{\alpha}]^T$$

After some algebraic manipulation, the equation is rewritten in matrix form as:

$$\begin{bmatrix} \frac{1}{4} m_p L_p^2 + J_{pzz} & \frac{1}{2} m_p L_r L_p \cos\theta \\ \frac{1}{2} m_p L_r L_p \cos\theta & \frac{1}{4} m_r L_r^2 + m_p L_r^2 + \frac{1}{4} m_p L_p^2 \sin^2\theta + J_{rzz} + J_{pyy}\sin^2\theta + J_{pxx}\cos^2\theta \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\alpha} \end{bmatrix}$$
$$+ \begin{bmatrix} \frac{1}{8} m_p L_p^2 \sin\theta\cos\theta\,\dot{\theta}\dot{\alpha} + \frac{1}{2}(J_{pyy} - J_{pxx})\sin\theta\cos\theta\,\dot{\theta}\dot{\alpha} \\ -\frac{1}{2} m_p L_r L_p \sin\theta\,\dot{\theta}^2 + \frac{1}{8} m_p L_p^2 \sin(2\theta)\,\dot{\alpha}^2 - (J_{pyy} - J_{pxx})\sin\theta\cos\theta\,\dot{\alpha}^2 \end{bmatrix} + \begin{bmatrix} \frac{1}{2} m_p g L_p \sin\theta \\ 0 \end{bmatrix} + \begin{bmatrix} \mu_\theta \dot{\theta} \\ \mu_\alpha \dot{\alpha} \end{bmatrix} = \begin{bmatrix} \tau_m \\ 0 \end{bmatrix}$$

### *A. DC motor modelling*

In order to model the input-output behavior of the system, the applied torque $\tau_m$ must be expressed as a function of the input voltage $u$.

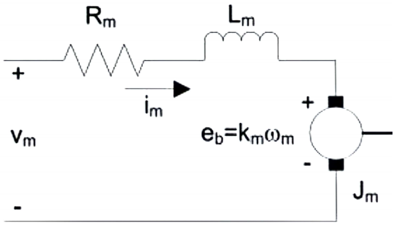


Fig. 2: Model of DC motor

Let $V_m$ be the applied motor voltage, $i_m$ the motor current, $R_m$ the armature resistance, $L_m$ the armature inductance, $e_b$ the back electromotive force $e_b = k_m \omega_m$, $k_m$ the motor back-emf constant, $\omega_m$ the motor angular velocity$\omega_m = \dot{\theta}$.

Using Kirchhoff's law, the differential equation describes DC motor can be expressed as:

$$V_m(t) - e_b(t) = R_m i_m(t) + L_m \frac{di_m}{dt},$$

As the electrical inductance of the servomotor is relatively very small, it can be neglected for simplification. The expression for current $i$ is obtained as:

$$i_m(t) = \frac{V_m(t) - e_b(t)}{R_m}$$

The torque produced by the DC motor is

$$\tau_m = K_t i = K_t \frac{V_m(t) - e_b(t)}{R_m} = K_t \frac{V_m(t) - k_m \dot{\theta}}{R_m}$$

Combining the voltage expression with the Euler-Lagrange equations, we obtain

$$\begin{bmatrix} \frac{1}{4} m_p L_p^2 + J_{pzz} & \frac{1}{2} m_p L_r L_p \cos\theta \\ \frac{1}{2} m_p L_r L_p \cos\theta & \frac{1}{4} m_r L_r^2 + m_p L_r^2 + \frac{1}{4} m_p L_p^2 \sin^2\theta + J_{rzz} + J_{pyy}\sin^2\theta + J_{pxx}\cos^2\theta \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\alpha} \end{bmatrix}$$
$$+ \begin{bmatrix} \frac{1}{8} m_p L_p^2 \sin\theta\cos\theta\,\dot{\theta}\dot{\alpha} + \frac{1}{2}(J_{pyy} - J_{pxx})\sin\theta\cos\theta\,\dot{\theta}\dot{\alpha} \\ -\frac{1}{2} m_p L_r L_p \sin\theta\,\dot{\theta}^2 + \frac{1}{8} m_p L_p^2 \sin(2\theta)\,\dot{\alpha}^2 - (J_{pyy} - J_{pxx})\sin\theta\cos\theta\,\dot{\alpha}^2 \end{bmatrix} + \begin{bmatrix} \frac{1}{2} m_p g L_p \sin\theta \\ 0 \end{bmatrix} + \begin{bmatrix} \mu_\theta \dot{\theta} + \frac{k_t k_m \dot{\theta}}{R_m} \\ \mu_\alpha \dot{\alpha} \end{bmatrix} = \begin{bmatrix} \frac{K_t V_m(t)}{R_m} \\ 0 \end{bmatrix}$$

### B. Simplification

Considering the slender structure of the pendulum arms and their rotational symmetry, the moments of inertia can be approximated as follows [1]

$$\mathbf{J}_r = \begin{bmatrix} J_{rxx} & 0 & 0 \\ 0 & J_{ryy} & 0 \\ 0 & 0 & J_{rzz} \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & J_r & 0 \\ 0 & 0 & J_r \end{bmatrix}, \quad \mathbf{J}_p = \begin{bmatrix} J_{pxx} & 0 & 0 \\ 0 & J_{pyy} & 0 \\ 0 & 0 & J_{pzz} \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & J_p & 0 \\ 0 & 0 & J_p \end{bmatrix}.$$

According to the parallel axis theorem, the total moment of inertia of rotary arm about its pivot point is

$$\hat{J}_r = \mathbf{J}_r + \frac{1}{4} m_r L_r^2.$$

The total moment of inertia of pendulum arm about its pivot point is

$$\hat{J}_p = \mathbf{J}_p + \frac{1}{4} m_p L_p^2.$$

In the global coordinate frame, when the pendulum arm is in its equilibrium position, the contribution of its mass to the moment of inertia about the motor axis can be approximated as that of a point mass located at a distance $L_1$. Therefore, the total moment of inertia of the system about the motor axis is given by:

$$\hat{J}_0 = \hat{J}_r + m_p L_r^2 = \mathbf{J}_r + \frac{1}{4} m_r L_r^2 + m_p L_r^2 = \mathbf{J}_h$$

The previous equation is substituted as

$$\begin{bmatrix} \ddot{\alpha}\left(\hat{J}_0 + \hat{J}_p \sin^2(\theta)\right) + \frac{1}{2}\ddot{\theta} m_p L_r L_p \cos(\theta) - \frac{1}{2} m_p L_r L_p \sin(\theta)\dot{\theta}^2 + \dot{\alpha}\dot{\theta}\hat{J}_p \sin(2\theta) + b_\alpha \dot{\alpha} \\ \frac{1}{2}\ddot{\alpha} m_p L_r L_p \cos(\theta) + \ddot{\theta}\hat{J}_p - \frac{1}{2}\dot{\alpha}^2 \hat{J}_2 \sin(2\theta) + \mu_\theta \dot{\theta} + \frac{1}{2} g m_p L_p \sin(\theta) \end{bmatrix} = \begin{bmatrix} \frac{K_t V_m(t)}{R_m} \\ 0 \end{bmatrix}$$

Then the angular acceleration of $\theta$ and $\alpha$ are expressed as:

$$\ddot{\theta} = \frac{\begin{bmatrix} \frac{1}{2} m_p L_r L_p \cos(\theta)\mu_\alpha \\ -(\mu_\theta + \frac{k_t k_m \dot{\theta}}{R_m})\left(\hat{J}_0 + \hat{J}_p \sin^2(\theta)\right) \\ \frac{1}{2} m_p L_r L_p \hat{J}_p \cos(\theta)\sin(2\theta) \\ -\frac{1}{2}\sin(2\theta)\left(\hat{J}_0\hat{J}_p + \hat{J}_p^2 \sin^2(\theta)\right) \\ -\frac{1}{8} m_p^2 L_r^2 L_p^2 \sin(2\theta) \end{bmatrix}^T \begin{bmatrix} \dot{\alpha} \\ \dot{\theta} \\ \dot{\alpha}\dot{\theta} \\ \dot{\alpha}^2 \\ \dot{\theta}^2 \end{bmatrix} + \begin{bmatrix} -\frac{K_t}{R_m}\frac{1}{2} m_p L_r L_p \cos(\theta) \\ \hat{J}_0 + \hat{J}_p \sin^2(\theta) \\ -\frac{1}{2} m_p L_p \sin(\theta)\left(\hat{J}_0 + \hat{J}_p \sin^2(\theta)\right) \end{bmatrix}^T \begin{bmatrix} V_m(t) \\ 0 \\ g \end{bmatrix}}{\hat{J}_0\hat{J}_p + \hat{J}_p^2 \sin^2(\theta) - \frac{1}{4} m_p^2 L_r^2 L_p^2 \cos^2(\theta)}$$

$$\ddot{\alpha} = \frac{\begin{bmatrix} -\hat{J}_p \mu_\alpha \\ \frac{1}{2} m_p L_r L_p \cos(\theta)\mu_\theta \\ -\hat{J}_p^2 \sin(2\theta) \\ -\frac{1}{4}\hat{J}_p m_p L_r L_p \cos(\theta)\sin(2\theta) \\ \frac{1}{2}\hat{J}_p m_p L_r L_p \sin(\theta) \end{bmatrix}^T \begin{bmatrix} \dot{\alpha} \\ \dot{\theta} \\ \dot{\alpha}\dot{\theta} \\ \dot{\alpha}^2 \\ \dot{\theta}^2 \end{bmatrix} + \begin{bmatrix} \frac{K_t}{R_m}\hat{J}_p \\ -\frac{1}{2} m_p L_r L_p \cos(\theta) \\ \frac{1}{8} m_p^2 L_p^2 L_r \sin(2\theta) \end{bmatrix}^T \begin{bmatrix} V_m(t) \\ 0 \\ g \end{bmatrix}}{\hat{J}_0\hat{J}_p + \hat{J}_p \sin^2(\theta) - \frac{1}{4} m_p^2 L_r^2 L_2^2 \cos^2(\theta)}$$

Considering the influence of the connecting cable on pendulum, we introduce a simple model where the cable acts as an external force. The cable is represented as a linear torque disturbance that is proportional to the pendulum angle:

$$\ddot{\alpha} = -K_c \alpha$$

Here, $K_c > 0$ represents the effect of the cable on the system behavior.

So the expression of $\ddot{\alpha}$ is

$$\ddot{\alpha} = \frac{\begin{bmatrix} -K_c \\ -\hat{J}_p \mu_\alpha \\ \frac{1}{2} m_p L_r L_p \cos(\theta)\mu_\theta \\ -\hat{J}_p^2 \sin(2\theta) \\ -\frac{1}{4}\hat{J}_p m_p L_r L_p \cos(\theta)\sin(2\theta) \\ \frac{1}{2}\hat{J}_p m_p L_r L_p \sin(\theta) \end{bmatrix}^T \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \dot{\theta} \\ \dot{\alpha}\dot{\theta} \\ \dot{\alpha}^2 \\ \dot{\theta}^2 \end{bmatrix} + \begin{bmatrix} \frac{K_t}{R_m}\hat{J}_p \\ -\frac{1}{2} m_p L_r L_p \cos(\theta) \\ \frac{1}{8} m_p^2 L_p^2 L_r \sin(2\theta) \end{bmatrix}^T \begin{bmatrix} V_m(t) \\ 0 \\ g \end{bmatrix}}{\hat{J}_0\hat{J}_p + \hat{J}_p \sin^2(\theta) - \frac{1}{4} m_p^2 L_r^2 L_2^2 \cos^2(\theta)}$$

### C. Linearized state space equation

The dynamic equation can be linearized around a desired equilibrium point $x^* \in \mathbb{X}^*$ where $\mathbb{X}^* = \{x \mid -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2},\ \dot{\theta}, \dot{\alpha} = 0,\ \alpha = \pi \cdot k, k \in \mathbb{Z}\}$. Here $x = [\alpha\ \theta\ \dot{\alpha}\ \dot{\theta}]^T$.

For the upright position, we then obtain$\alpha = 0, \theta = \pi, \dot{\alpha} = 0, \dot{\theta} = 0$, the linearized simplified state space equation for upright position can be expressed as:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{\theta} \\ \ddot{\alpha} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} \alpha \\ \theta \\ \dot{\alpha} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ B_3 \\ B_4 \end{bmatrix} V_m(t)$$

where

$$\begin{aligned}
A_{31} &= -K_c, \\
A_{32} &= \frac{\frac{1}{4} g m_p^2 L_p^2 L_r}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
A_{33} &= \frac{-b_\alpha \hat{J}_p}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
A_{34} &= \frac{-\frac{1}{2} b_\theta m_p L_p L_r}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
A_{41} &= 0, \\
A_{42} &= \frac{\frac{1}{2} g m_p L_p \hat{J}_0}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
A_{43} &= \frac{-\frac{1}{2} b_\alpha m_p L_p L_r}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
A_{44} &= \frac{-b_\theta \hat{J}_0}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
B_3 &= \frac{K_t}{R_m} \frac{\hat{J}_2}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}, \\
B_4 &= \frac{K_t}{R_m} \frac{\frac{1}{2} m_p L_r L_p}{\hat{J}_0 \hat{J}_p - \frac{1}{4} m_p^2 L_r^2 L_p^2}.
\end{aligned}$$

For the convenience of system identification, we compute the linearized state equation for downright position. For equilibrium point, $\alpha = 0, \theta = 0, \dot{\alpha} = 0, \dot{\theta} = 0$.

Then the state equation for downright position is expressed as:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{\theta} \\ \ddot{\alpha} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} \alpha \\ \theta \\ \dot{\alpha} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ B_3 \\ B_4 \end{bmatrix} V_m(t) \tag{1}$$

where $A_{31} = A_{31}, \quad A_{32} = A_{32}, \quad A_{33} = A_{33}, \quad A_{34} = -A_{34}, \quad A_{41} = 0, \quad A_{42} = -A_{42}, \quad A_{43} = -A_{43}, \quad A_{44} = A_{44},$ $B_3 = B_3, \quad B_4 = -B_4.$

## III. Calibration

To make sure obtained measurements are accurate and mapped to physical units in the right way, calibration is needed. For this project the motor dead-zone needed to be calibrated and the sensors were verified be already calibrated. In `Simulink`, the outputs signals have been first multiplied by -1 to conform to the rotational direction defined in the model.

### A. *Sensor calibration verification*

An optical encoder measures the angle $\alpha$ of the rotary arm. To validate the accuracy of the encoder, a set of experiments was conducted. In the experiments, the rotary arm moved from a starting point to a specific rotation. The experiments were performed by hand and are therefore not very accurate. They only serve to verify the assumed correctness of the sensor calibrations. The experiments consisted of initializing the sensors while keeping the arm or pendulum at a certain angle, and then bringing the arm or pendulum to a predefined set of angles. For the optical encoder, which measures the angle $\alpha$ of the rotary arm, results of the experiments are given in table I. For the pendulum, results are given in table II.

| Desired angle | Measurement (from 0°) | Measurement (from $-90°$) | Measurement (from 90°) |
|---|---|---|---|
| $-90°$ | -89.1 | -0.35 | -179.6 |
| $-45°$ | -45.7 | 44.5 | -134.6 |
| 0° | 0 | 89.8 | -90.7 |
| 45° | 44.5 | 133.9 | 44.12 |
| 90° | 89.3 | 178.6 | 0 |

TABLE I: Results for verification of measurements of $\alpha$

| Desired angle | Measurement (from 0°) | Measurement (from $-180°$) | Measurement (from 180°) |
|---|---|---|---|
| $-180°$ | 181.9 | 0 | 359.8 |
| $-90°$ | 91.2 | -89.8 | 268.6 |
| 0° | -0.2 | -179.6 | 179.1 |
| 90° | -88.9 | -269.4 | 90.9 |
| 180° | -180.4 | -360.2 | 0 |

TABLE II: Results for verification of measurements of $\theta$

To gain some insights in the validity of the calibration, the obtained histogram for the difference between expectations and measurements is given in figure 3. For this very limited and basic verification, no significant deviation was found of statistical significance for both sensors. We emphasize that this experiment was under the very reasonable assumption that the sensors were already calibrated (as was instructed to us), and any verification of calibration of sensors in for example industry should be much more extensive. Also the experiment setup was not designed for accuracy but rather to serve as a pragmatic way to quickly verify sensor calibration. Most of the errors are expected to come from the experimental setup.

### B. *Dead-zone calibration and the dead-zone compensator*

Due to static friction that is present in the actuator, a motor dead-zone is present in the setup. This dead-zone is problematic as the hysteresis originating from this can introduce a delay in the control input. This can be a significant limitation in controller synthesis later on. This motor dead-zone can be modeled in a number of ways. A smooth dead-zone model uses left and right breaking points $b_l$ and $b_r$ as can be seen in figure 4 [2]. In the range $[b_l, b_r]$ the effective input is effectively zero while the supplied input $u(t)$ might be non-zero. To account for this, the input needs to be offset by these breaking points. This motivates the construction of a dead-zone compensator.

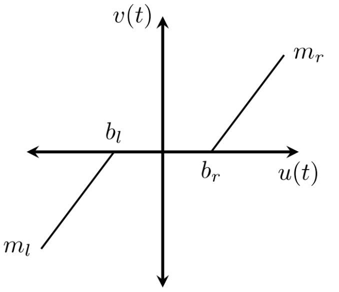
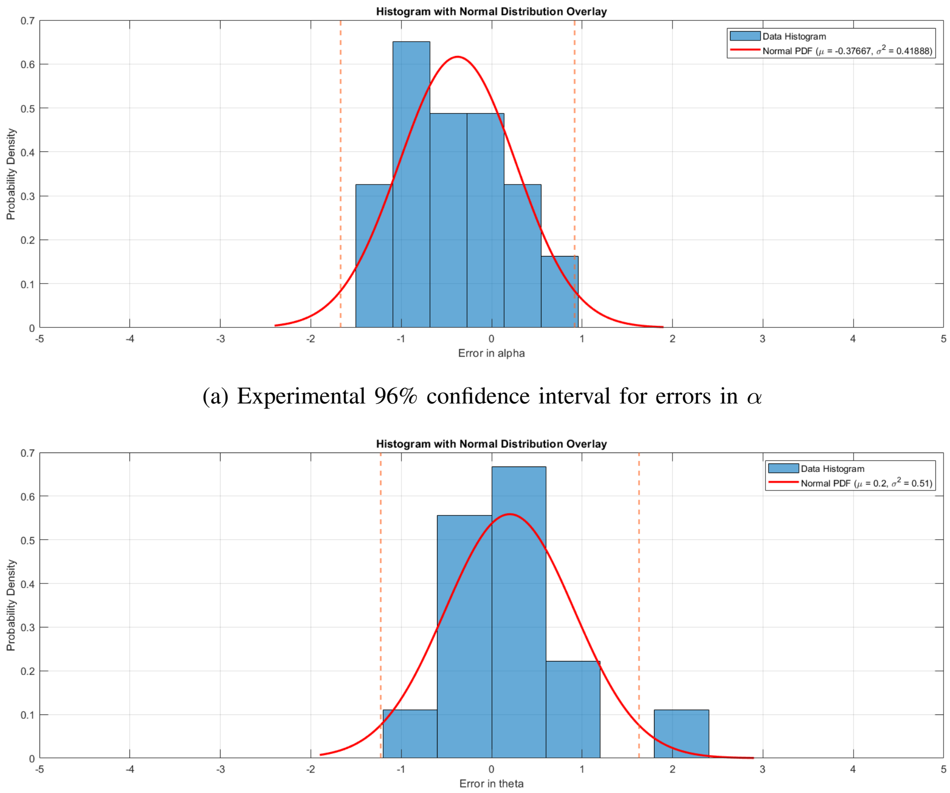


Fig. 4: Smooth dead-zone model [2]

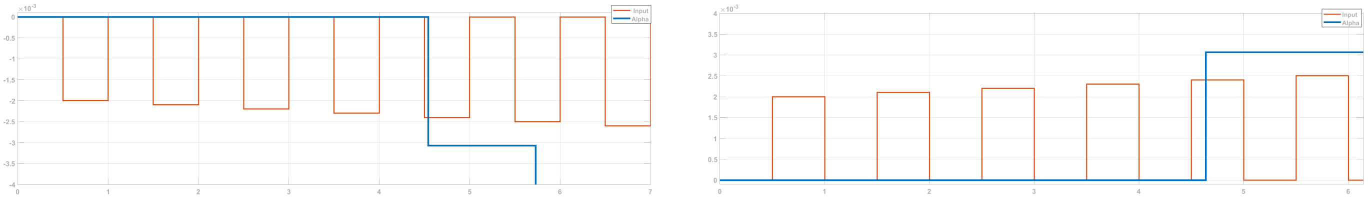
The dead-zone compensator will be a simple amplifier that adds the offset of $b_r$ when the input is larger than zero, and subtracts the offset $b_l$ if the input in less then zero. The values for $b_l$ and $b_r$ have been determined experimentally by feeding the system a square wave and gradually increasing the amplitude to determine the input magnitude for which the dead-zone is overcome, as can be seen in figure 5. The eventual dead-zone breaking points were estimated to be $b_l = b_r = 0.0024$.



(a) Experimental 96% confidence interval for errors in $\alpha$

(b) Experimental 96% confidence interval for errors in $\theta$

Fig. 3: Statistical characteristics of experimentally determined sensor errors



(a) Calibration of breaking point $b_l$

(b) Calibration of breaking point $b_r$

Fig. 5: Calibration of breaking points $b_l$ and $b_r$

## IV. Identification

Before viable controllers can be synthesized to control the Furuta pendulum, the dynamical system needs to be identified first. Models for dynamical systems can be categorized as white-box models, grey-box models and black-box models. For a white-box model, the underlying principles are completely known and the necessary parameters are available. For a black-box model, there is no prior knowledge of the system or model structure. Both the model structure and parameters are completely inferred from data. Grey-box models lie somewhere in between. They combine a first principle model with data-based parameter estimation techniques to recover a viable model. For the identification and control of this Furuta pendulum, a grey-box model approach is taken. Specifically, the first principle model is used to determine model structure and to leverage this structure to be able to identify the system near the unstable equilibrium using data obtained from excitations near the stable equilibrium. First, the experiment set-up, used input sequences and pre-processing methods are elaborated upon. Then, the used identification method is motivated and explained. Finally the results are presented and verified.

### A. *Experiment set-up*

To be able to perform identification, experiments need to be run to retrieve data suitable for identification methods. Designing experiments, fitting a model and validation are all part of an iterative process which can be seen in figure 6 from Verhaegen (2007) [3].

This iterative approach was adopted in the project. A large number of experiments were conducted, implementing different strategies for input signals. To get a persistently exciting signals, and to capture the full system dynamics, a sufficiently large spectral range of the input signal is desired. Going on with this reasoning, initially a sweep or chirp signal was used to excite the system. However, nonlinearities due to the ”sticky” behaviour of the pendulum for small inputs [1], resulted in measurements that yielded unfit data to fit the linearized dynamics. The sticking behaviour however was observed to subside for higher frequencies. This motivated the use of a pseudo-random binary sequence as an input for the system. This signal was defined as follows:

let $v_k$ be a zero mean Gaussian random sequence with unit variance. Then

$$w_k = \begin{cases} -\delta, \text{ if } v_k < 0 \\ \delta, \text{ if } v_k \geq 0 \end{cases}$$

Here $\delta$ defines the PRBS gain. PRBS signals posses a wide frequency band (as can be seen in figure 7), while remaining bounded in the amplitude of the actual signal [4]. An added benefit of using this signal is that while the angle $\theta$ will stay close to zero, a wide range of values for $\alpha$ will be contained in the experiment data. This is desirable to accurately capture the effect of the cable on the rotary arm’s angular acceleration.

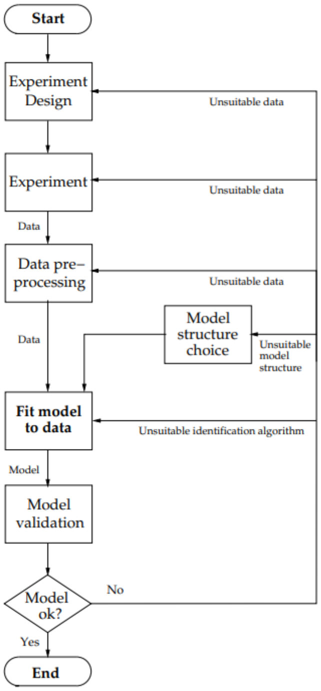


Fig. 6: The Identification cycle [3]

### B. *Pre-processing of data*

Because there was very little noise observed in the output signal, the data has not been filtered in pre-processing. Depending on the experiment, the bias has in some cases been removed from the measurements and inputs. This was mainly done to remove nonlinearity induced offsets caused by the encoder cable and the ’sticking’ behaviour.

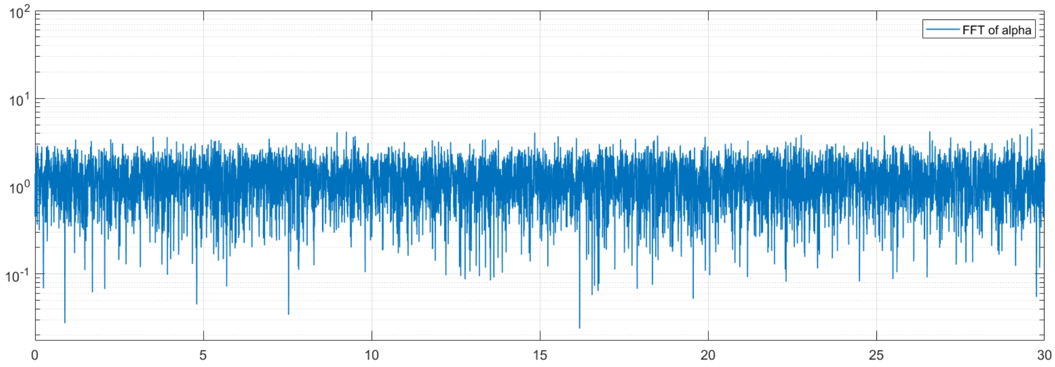


Fig. 7: PRBS frequency content

### C. *Identification method*

This section will first give a primer on both general prediction error methods, output-error methods and subspace identification methods for completeness. Then, a method is proposed and implemented that combines these methods together with the structure of this system to achieve identification without the need of any initial parameter guesses (this method is generalized in Appendix A in XII-A).

*1) **Prediction Error Method identification**:* Prediction error methods are parameter estimation methods that try to estimate the system matrices A, B, C, D and K of the following system:

$$\begin{cases} \hat{x}_{k+1} = A\hat{x}_k + Bu_k + Ke_k \\ \hat{y}_k = C\hat{x}_k + Du_k + e_k \end{cases}$$

[1] the sticking is likely a remnant of the Coulomb friction, which cannot be compensated perfectly

Where $e_k$ is assumed to be Gaussian white noise. The parameter optimization is done by solving the following optimization problem of minimizing the one-step-ahead prediction error.

$$\hat{\theta}_N = \arg\min_{\theta} \frac{1}{N} \sum_{k=0}^{N-1} ||y_k - \hat{y}(k|k-1, \theta)||_2^2$$

The matrix K in the state space model is computed as a Kalman filter gain; this will not only minimize the errors but also create a minimum-variance estimate of the parameters. For many problems, this will result in better estimates than for the case where K is taken to be 1. Such prediction error methods are called output-error methods. The exception to this is the case where there is almost no measurement noise, as is the case for this system. Then, omitting the the objective of creating a minimum variance estimate simplifies the optimization and could potentially yield a better estimation. Following this reasoning, an output-error method was adopted in the final identification method. Some experimental proof that it is indeed the case that the output-error method yields better results for this specific case is given at the end of this section.

Although some modifications of the output error and prediction error methods like the output normal form [3] aid in solving this optimization problem, the problem generally remains to be a very nonlinear and nonconvex optimization problem. This means that to end near the global minimum and get a good result on the identification, a good initial guess is needed. As stated in the objectives of this project in section I, such an initial guess will be assumed to not be available.

*2) **Subspace identification**:* Another class of identification methods are the subspace identification methods. Subspace methods try to construct a realization by retrieving certain subspaces related to the dynamical system from the block Hankel matrices of input and output data [3]. Subspace methods solve the identification problem by solving a least-squares problem instead of a nonlinear optimization problem. Because of this, subspace identification methods do not need a good initial guess.

Subspace identification does have the drawback of estimating a realization only up to a similarity transformation. This is a problem as the identification of the uncontrolled system can only be done around the stable equilibrium point. To be able to convert the obtained estimated model from this stable equilibrium to the unstable equilibrium, the specific form given in equation 1 is needed.

*3) **System structure**:* For the final identified model, we intended to utilize the structure of the obtained realization to create a realization around the unstable equilibrium (by switching the signs of certain matrix entries, see chapter II). To still be able to utilize subspace identification methods the structure of the A and C matrix can be used.

First, if we choose the state of the system to be $x = [\alpha\ \theta\ \dot{\alpha}\ \dot{\theta}]^T$ and the outputs to be $y = [\alpha\ \theta]^T$. We know that a realization will exist with the following form:

$$\begin{cases} \begin{bmatrix} \dot{\alpha} \\ \dot{\theta} \\ \ddot{\alpha} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} \alpha \\ \theta \\ \dot{\alpha} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ B_3 \\ B_4 \end{bmatrix} u \\ \begin{bmatrix} \alpha \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \theta \\ \dot{\alpha} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \end{cases}$$

Notice that the zero entries in the matrix B are implied by the state vector and the first two rows of A. Next, the fact that the subspace method estimates a realization of a system up to a similarity transform can by used in combination with the structure of this system to create such a similariy transformation matrix T.

For a observability matrix based on the structured state space from above $\mathcal{O}_s$ and the observability matrix of the via subspace estimated realization $\mathcal{O}$ we have

$$\mathcal{O}_s = \mathcal{O}T^{-1}$$
$$\begin{bmatrix} C_s \\ C_s A_s \\ \vdots \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \end{bmatrix} T^{-1}$$

Now using the structure of the system

$$C_s A_s = \left[\begin{array}{c|c} 0 & I_2 \end{array}\right] \Leftarrow \begin{cases} C_s = \left[\begin{array}{c|c} I_2 & 0 \end{array}\right] \\ \\ A_s = \left[\begin{array}{c|c} 0 & I_2 \\ \hline \Phi_1 & \Phi_2 \end{array}\right] \end{cases}$$

Reveals the following structure in the observability matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \end{bmatrix} T^{-1}$$

This holds for

$$T = \begin{bmatrix} C \\ CA \end{bmatrix}, \quad \text{if } \begin{bmatrix} C \\ CA \end{bmatrix} \text{ is full rank}$$

A general proof for this rank condition is not given here, but for the obtained matrices $(C_{sub}, A_{sub})$ given in table III this condition can be easily verified to hold.

While for a perfect realization estimation this transformation might result in the desired system, this will not be the case in reality. The obtained transformed system is close to the desired structure, but some refinements need to be done. This was done by first enforcing the structure by setting $B_1, B_2, A_{41}$ to be zero, and then using these structured matrices as an initial guess for PEM identification (in this case output-error identification specifically). This procedure is generalized in section XII-A for a class of systems.

### D. *Selection of subspace identification algorithm*

For ill-conditioned / unstable systems, the MOESP algorithm seems to perform better than N4SID [5]. A drawback of the MOESP algortihm is that it assumes Gaussian white noise, an assumption that might not be reasonable for any physical system. A solution to this is the PO-MOESP algorithm, that uses past inputs and past outputs in instrumental variables to remove colored noise induced bias from the estimation.

### E. *Results*

The resulting state spaces of each step can be seen in the table below,

<table>
<tr><th>Estimation step</th><th>Realization</th></tr>
<tr><td>$\left[\begin{array}{c|c} A_{sub} & B_{sub} \\ \hline C_{sub} & D_{sub} \end{array}\right]$</td><td>$\left[\begin{array}{cccc|c} -0.7893 & 2.369 & 1.949 & 6.896 & -851.6 \\ -0.6989 & -0.8928 & -12.75 & 10.24 & -360.6 \\ -0.04904 & 10.77 & -0.2488 & -7.021 & 156 \\ -1.02 & -7.011 & 4.837 & -4.11 & -517.9 \\ \hline -0.1618 & 0.05943 & 0.1521 & 0.3574 & 0 \\ -0.007331 & 0.1786 & 0.05726 & -0.1446 & 0 \end{array}\right]$</td></tr>
<tr><td>$\left[\begin{array}{c|c} A_t & B_t \\ \hline C_t & D_t \end{array}\right]$</td><td>$\left[\begin{array}{cccc|c} 0 & 0 & 1 & 0 & -45.05 \\ 0 & 0 & 0 & 1 & 25.65 \\ -20.74 & 124 & -6.208 & -0.3546 & 2386 \\ 20.53 & -239.5 & 4.677 & 0.1672 & -2035 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{array}\right]$</td></tr>
<tr><td>$\left[\begin{array}{c|c} A_s & B_s \\ \hline C_s & D_s \end{array}\right]$</td><td>$\left[\begin{array}{cccc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -8.824 & 117.6 & -4.878 & 1.696 & 2017 \\ 0 & -244.2 & 2.743 & -0.9829 & -1938 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{array}\right]$</td></tr>
</table>

TABLE III: State space estimations for the subspace step, transformation step and the final PEM derived structured step

Notice that we used our first principle model [2] to enforce the entry A(4,1) to be zero, conforming to the model structure.

### F. *Discretization*

While the fitted model is for a continuous time system, measurements and inputs will be discrete and have a sampling time of $T_s = 0.01$s. Before control design can begin, the system is therefore first discretized using the zero order hold method. Motivation for this is that this most accurately models what is happening in the actual system. No difference in identification results were observed on the discretized system.

[2] Note that a first principle model is not even neccesary here, only the physical insight that the set of al equilibrum points can be denoted as $\{x|\theta = \dot{\theta} = \dot{\alpha} = 0, |\alpha| < 90°\}$ is enough, as this directly implies no coupling between $\alpha$ and $\theta$ is desirable

### G. *Validation*

To validate the found model, validation sets were created by running independent experiments. By looking at the frequency response of the system for a PRBS input in figure 8, it is evident that most of the system dynamics lie within the [0, 20] rad/s frequency band with the 0 dB crossover frequency around 12 rad/s. As fast dynamics are desired in the controlled system, we are mainly concerned with the medium to high frequencies within this band. To test if the model is able to capture dynamics in this frequency domain, a square wave at 8 rad/s, a square wave at 4 rad/s and a colored PRBS were chosen as validation input sequences. The colored PRBS is created by first filtering the zero mean Gaussian noise sequence $v_k$ with a low pass filer that has a cut-off frequency at 20 rad/s.

First looking at the validation data for the 8 rad/s square wave in figure 9, the model is seen to perform reasonably well, although not perfect. In figure 9b both the autocorrelation (left) of the residuals and the cross-correlation (right) between the residuals and the inputs are given. The blue band denotes the 99% confidence interval. Although the cross-correlation largely falls within this range, the autocorrelation of the residuals does not. For both the residuals of $\alpha$ (top) and the residuals of $\theta$ (bottom), there is a periodic trend. For, PEM identification, this would be a problem as these methods incorporate a noise model. If a PEM identified model exhibits some structure in the autocorrelation of the residuals, this means that the model is insufficient. For output-error methods however, this does not directly imply an insufficient model as the the output-error methods do not include a noise model. The structure in the autocorrelation merely reflects the presence of unmodeled nonlinear dynamics and friction dynamics. The claim that the auto-correlation of the residuals is not an indicator for performance when using the output-error method is addressed at the end of this section.

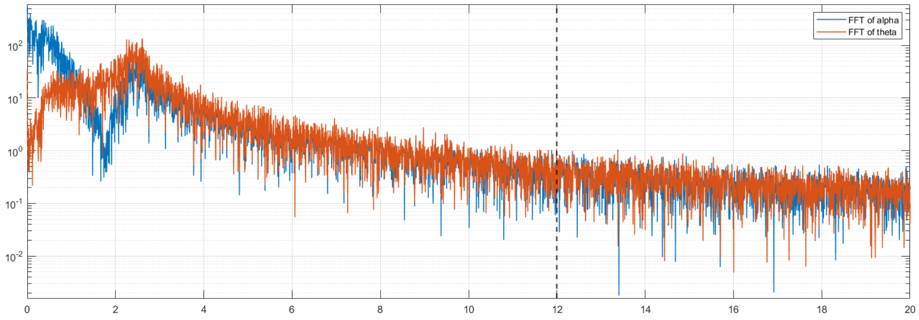
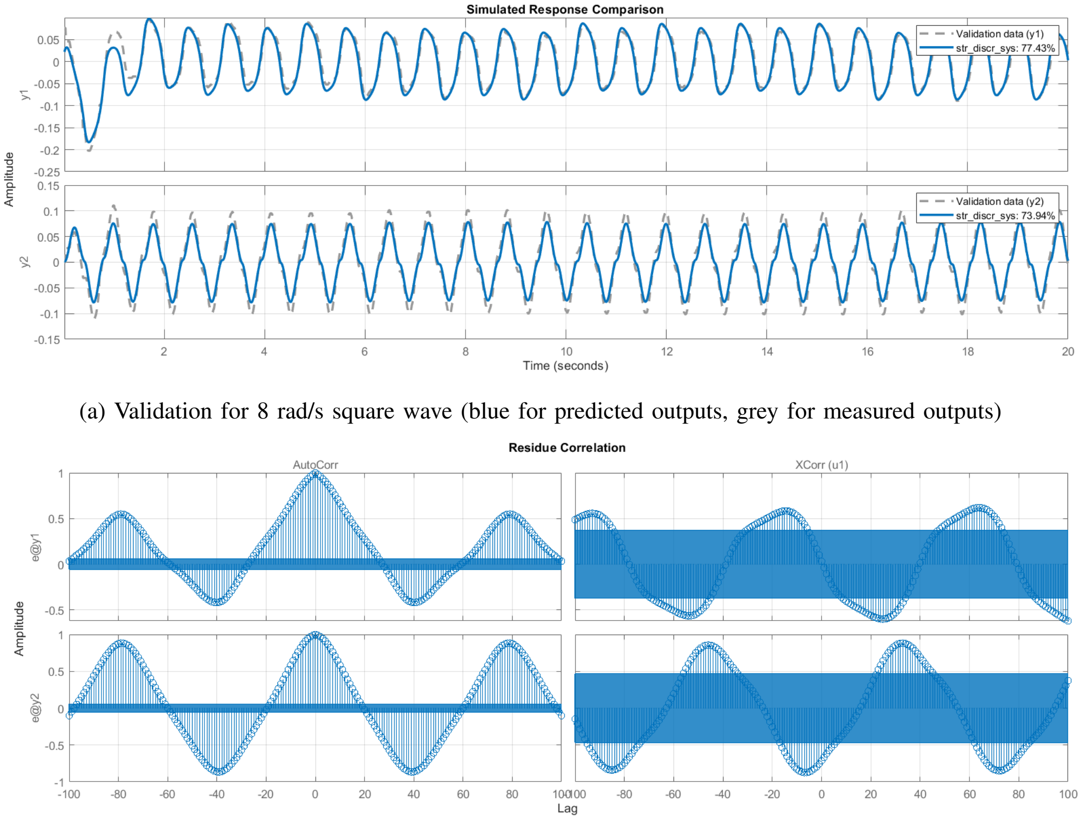


Fig. 8: Frequency response to a PRBS input

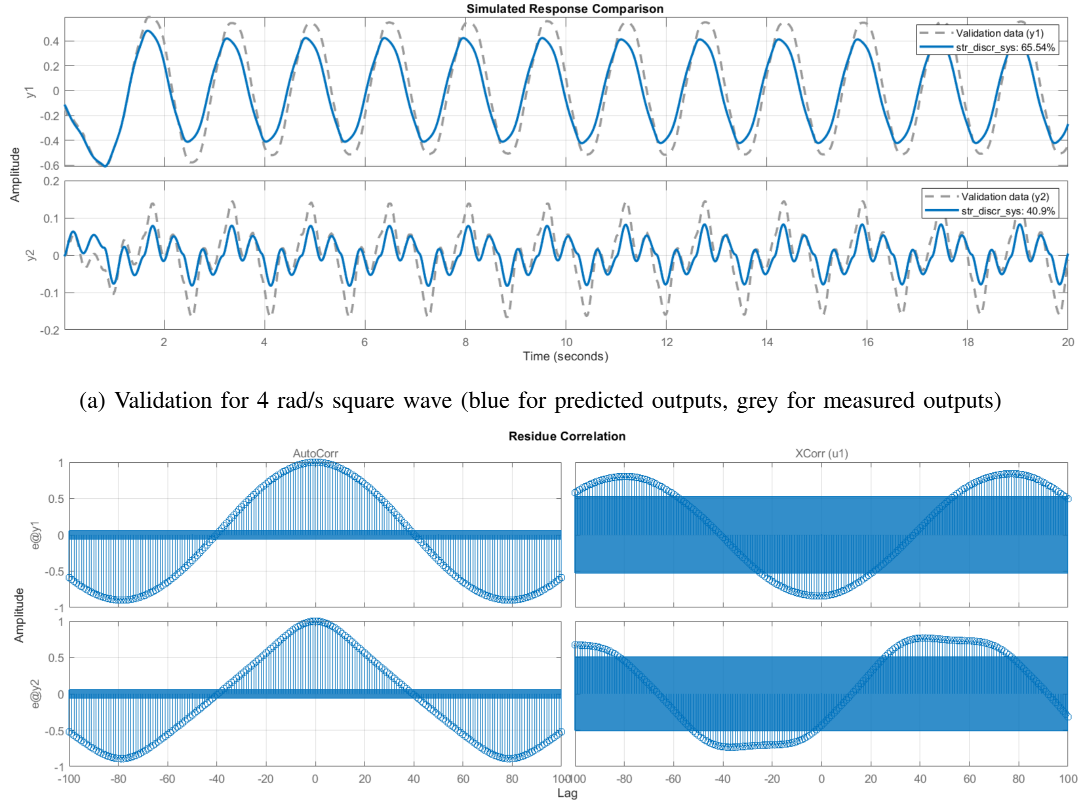
When the input signal is of a lower frequency, for example 4 rad/s, the performance of the model declines 10. Performance decreases even further when the system is tested on a square wave input with frequency lower than 4 rad/s. This indicates a limitation of the model; low frequency inputs cause significant prediction errors. A possible explanation for this is that there is hysteresis (the 'sticking that has been mentioned before) that is more significant for lower input frequencies. The validation results for 4 rad/s indeed show some delay in the response of alpha 10a.



(a) Validation for 8 rad/s square wave (blue for predicted outputs, grey for measured outputs)

(b) Residuals for 8 rad/s square wave

Fig. 9: Validation and residuals for 8 rad/s square wave

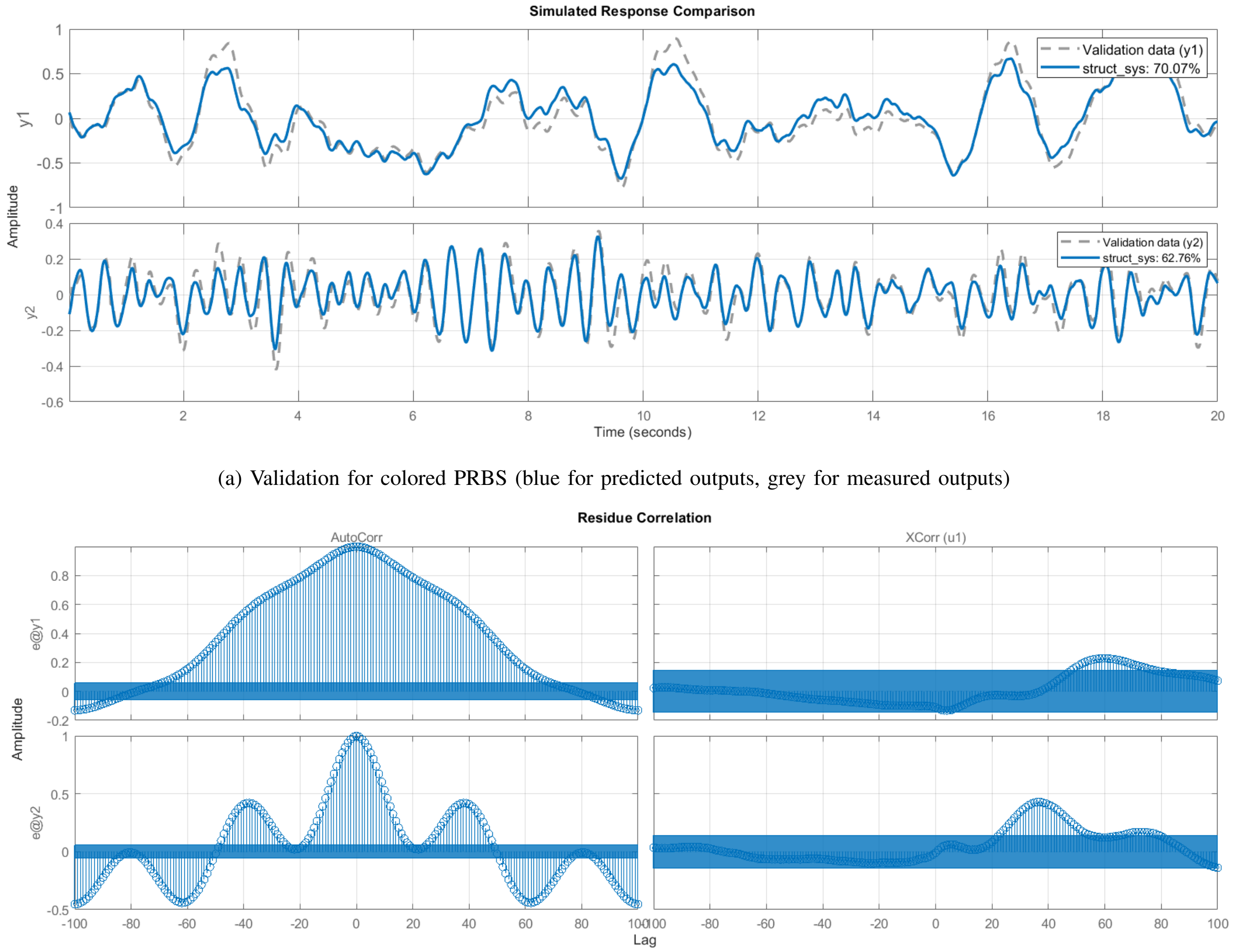


(a) Validation for 4 rad/s square wave (blue for predicted outputs, grey for measured outputs)

(b) Residuals for 4 rad/s square wave

Fig. 10: Validation and residuals for 4 rad/s square wave

Lastly the validation results for the colored PRBS are given in 11. Again, the cross-correlation between the input and the outputs is within the 99% confidence inteval, the auto-correlations of the residuals imply there are some unmodelled dynamics, but the overal response in figure 11a again looks reasonably well.



(a) Validation for colored PRBS (blue for predicted outputs, grey for measured outputs)

(b) Residuals for colored PRBS

Fig. 11: Validation and residuals for colored PRBS

***Conclusion on model validity:*** The model has been seen to perform reasonably well in the higher frequencies within the system's frequency band. The model performance however degrades when the system is excited by lower frequencies. As the aim is to design a controller that balances the pendulum, something that will require fast controller dynamics and therefore high frequency control inputs, we expect the model to still be able to perform the control task well. This assumption is proven by the fact that both the LQG and MPC controllers were able to complete the given control tasks well (see section VI and VIII).

***Comparison general prediction error method and output-error method:*** Finally a comparison between the output-error method and the prediction-error method with noise model is done. The results in figure 12 support the claim that in the case of this system, identification is actually improved by omitting a noise model. At the beginning of this section the claim was made that the structure in the auto-correlation of the residuals was not indicative of performance when using the output-error method. Indeed, when utilizing PEM the residual auto-correlation on validation is largely removed, while the actual performance is worse than for the output-error method 13.

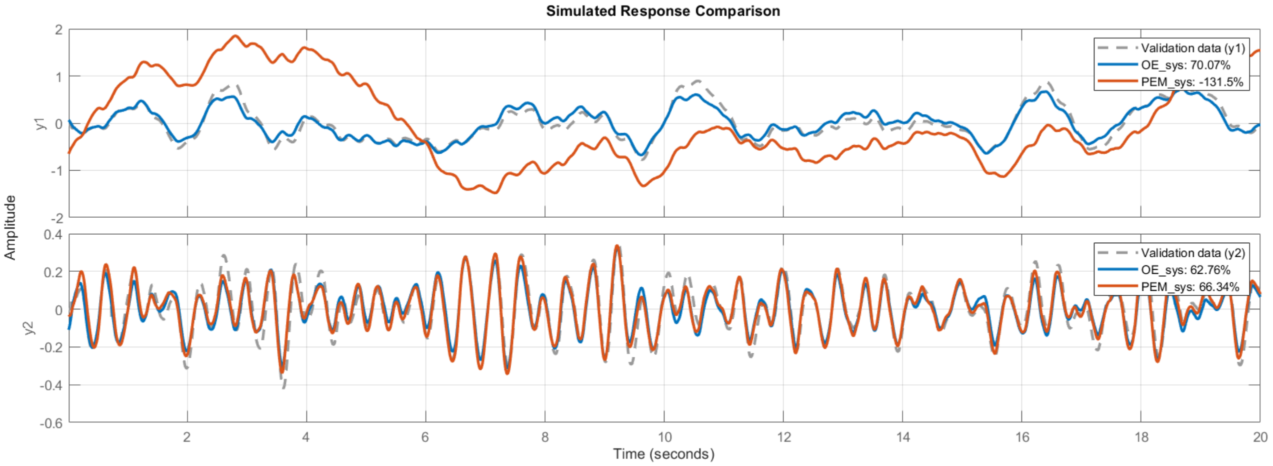


Fig. 12: Comparison of output-error (blue) and prediction error method (orange) on colored PRBS validation set (grey)

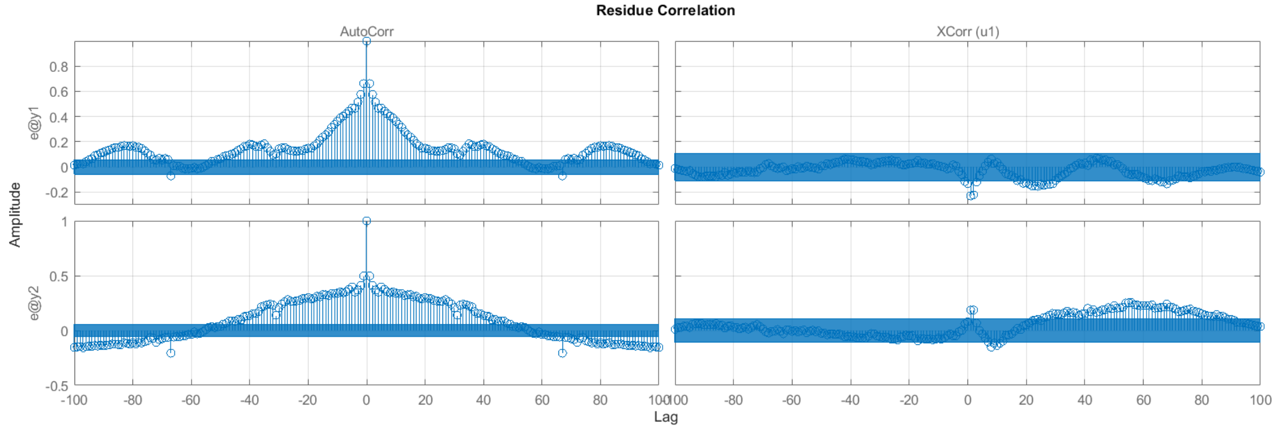


Fig. 13: Residuals of PEM results of colored PRBS validation

### *Effect of transformation and structure enforcement via output-error method on system*

The effect of imposing the structure and using the output-error method to refine the estimate can be examined by looking at the poles of the system identified via subspace compared to the poles of the final identified systems. These poles are near identical as can be seen in figure 16 (note the scale of the plot). These poles are of the linearized system around the stable equilibrium.

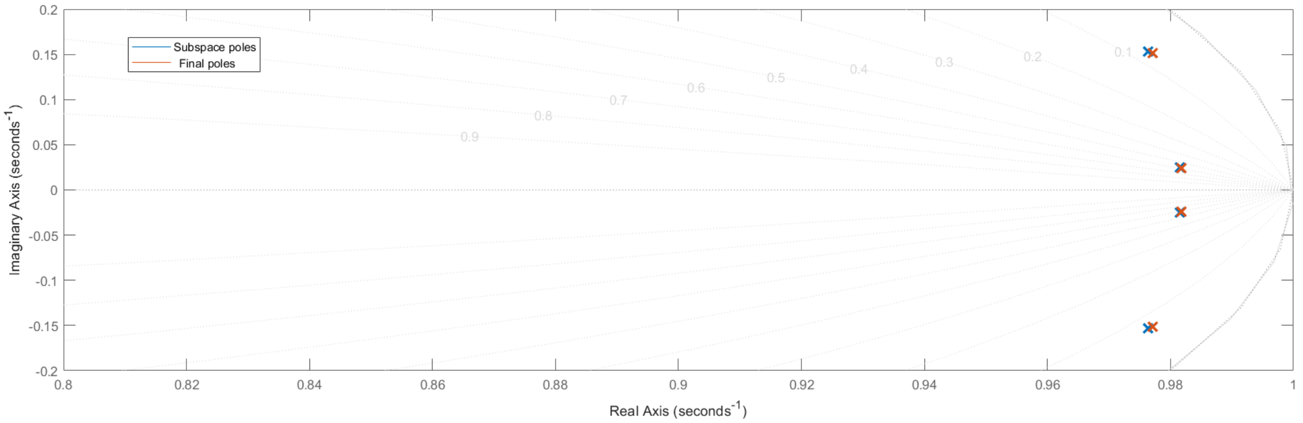


Fig. 14: Poles of the subspace identified system and the final identified system

## V. State Estimation

Before LQR and MPC controllers can be designed, the problem of state estimation needs to be solved first. For this, an observer will be constructed that computes a state estimation based on output and input measurements. There are a number of ways to do this. For example, one could employ a nonlinear observer, such as a high gain observer [6], or opt for a linear observer such as a Luenberger observer or a Kálmán observer. As we have assumed the linearized model to sufficiently capture model dynamics near the unstable equilibrium, it does not make sense to use a nonlinear observer. Therefore, a linear observer is chosen. For a linear observer, the existence of an observer comes down to the detectability of the system [7]. By the structure of the observability matrix $\mathcal{O}_s$ (the first n rows form an identity matrix), we can directly conclude the observability and therefore detectability of the linearized continuous time system. For the discretized system, the observability matrix is given as:

$$\mathcal{O}_s = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0.9996 & 0.0058 & 0.0098 & -0.0000 \\ 0.0000 & 1.0121 & -0.0001 & 0.0100 \\ 0.9983 & 0.0226 & 0.0190 & -0.0001 \\ 0.0000 & 1.0485 & -0.0005 & 0.0201 \\ 0.9962 & 0.0501 & 0.0279 & 0.0000 \\ 0.0001 & 1.1094 & -0.0012 & 0.0307 \end{bmatrix}$$

Which can also be verified, using for example `rank` in `Matlab`, to be full rank.

A linear observer then can be found that estimates the states by minimizing the error via output-error feedback:

$$\begin{cases} \hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - \hat{y}_k) \\ \hat{y}_k = C\hat{x}_k \end{cases}$$

The eventual choice for the linear observer in this project is the Kálmán observer. The reason for choosing the Kálmán observer and not a Luenberger observer is that the Kálmán observer creates an unbiased estimate that minimizes the variance of the error. This is useful in this case not because of measurement noise, but rather because of the process noise. Let the model below denote the obtained state space description corrupted with some process noise $w_k \in \mathbb{R}^n$ and some measurement noise $v_k \in \mathbb{R}^l$.

$$\begin{cases} x_{k+1} = Ax_k + Bu_k + w_k \\ y_k = Cx_k + v_k \end{cases}$$

This description is a more accurate representation of the real system. Here $w_k$ can be thought of as the error between the obtained model and the real system. For a near perfect model, $w_k$ can be seen as some gaussian noise. For the obtained model, which is not near perfect, $w_k$ cannot be seen as gaussian noise. However, as argued before, the model performs well for higher frequencies (which is the domain of interest). The hypothesis therefore is that for the operating range of the controlled system, the assumption that $v_k$ and $w_k$ can be estimated as white noise sequences is valid. The proof that the Kálmán observer works in the physical system can be seen in figure 15.

For LTI systems with $(A, C)$ observable and some weighing matrices $Q, R \succ 0$ for which $(A, Q^{1/2})$ reachable, the steady state solution to the Kalman filtering problem can be given as follows [8].

$$L = (S + APC^\top)(CPC^\top) + R)^{-1}$$

Where

$$P = \lim_{k \to k} P(k|k-1) \succ 0, \quad \text{for any initial } P(0|-1) \succ 0$$

$$P = APA^\top + Q - (A + APC^\top)(CPC^\top + R)^{-1}(S + APC^\top)^\top$$

$$\begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \Delta(k-j) = \mathbb{E} \begin{bmatrix} w_k \\ v_k \end{bmatrix} \begin{bmatrix} w(j)^\top & v(j)^\top \end{bmatrix}$$

The covariance matrices Q and R are actually not known, but are rather taken to be weighting matrices that should be tuned, similar to how the weighting matrices of an LQR are to be tuned (as both are solved using the discrete algebraic Riccati equation the duality is not unexpected).

| Weighting matrix | value |
|---|---|
| $Q$ | $\text{diag}(10^{-2}, 10^{-3}, 10^{-2}, 10^{-2})$ |
| $R$ | $10^{-9}$ |

TABLE IV: Weighting matrices for Kálmán observer

### *Implementation*

The Kálmán observer is implemented using the `Kalman` block in `Simulink`. To tune the weighting matrices, a large variance in process noise was taken to account for inaccuracies in the model and a low variance in measurement noise was taken as the setup exhibited almost no measurement noise. The final matrices are as follows given in table IV.

### *Results*

The resulting estimation error of the Kálmán filter is given in figure 15, where the estimation error is given by $\tilde{y}_k = y_k - \hat{y}_k$. The results are obtained from the LQR controlled system which is initiated at the unstable equilibrium, brought up to the unstable equilibrium and activated at around 3.8 seconds. The error dynamics are asymptotically stable. Furthermore, the estimation errors are negligible over the the whole trajectory. These two features indicate that the observer is performing well.

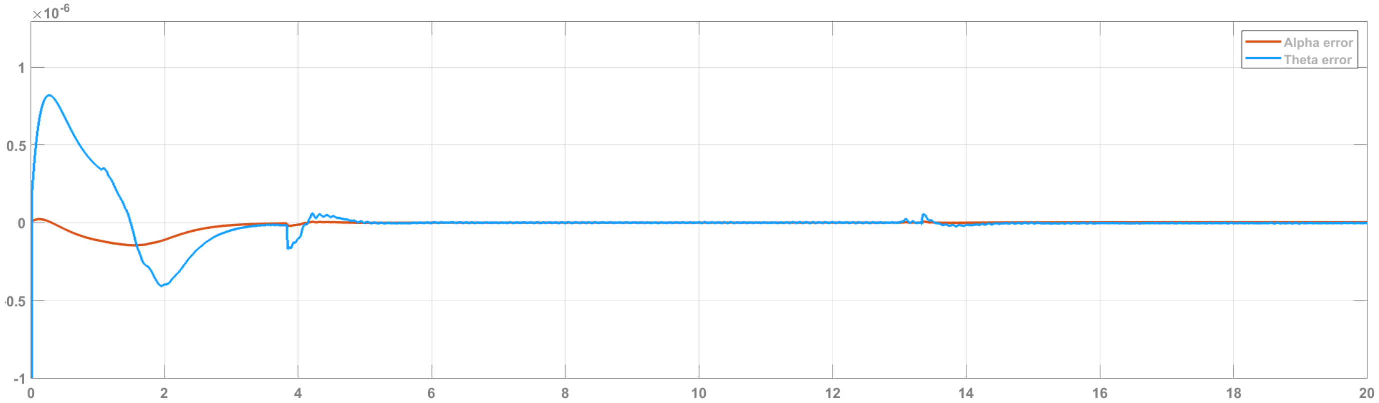


Fig. 15: Estimation errors of alpha and theta (rad) over time (s)

# VI. LQR Control

The Linear Quadratic Regulator is a classical optimal control method widely used for the design of state-feedback controllers in linear time-invariant systems. Its core objective is to minimize a quadratic cost function that balances state deviation and control effort under the constraint of closed-loop stability.

The first step in the synthesis of a controller for a linear system is to check the stabilizability of a system [7]. For the obtained model, the controllability matrix is

$$\mathcal{C} = \begin{bmatrix} B & BA & BA^2 & BA^3 \end{bmatrix} = \begin{bmatrix} 0.0990 & 0.2901 & 0.4731 & 0.6502 \\ 0.0959 & 0.2852 & 0.4744 & 0.6685 \\ 19.6110 & 18.6645 & 17.9637 & 17.4949 \\ 19.0921 & 18.8475 & 19.0865 & 19.8039 \end{bmatrix}$$

This matrix again can be veryfied with the use of for example `rank` in `Matlab` to be full rank, system is thus fully controllable.

For a discrete-time linear system:

$$x_{k+1} = Ax_k + Bu_k$$

the LQR aims to find the state-feedback control law:

$$u_k = -Kx_k$$

that minimizes the cost function:

$$J = \sum_{k=0}^{\infty} \left( x_k^\top Q x_k + u_k^\top R u_k \right)$$

where $Q$ is used to decide how important each state is in the cost function, and $R$ is used to decide how much we want to limit the magnitude of the control input. A higher value in $Q$ means that the controller will penalize that state more and try to keep it smaller, a higher value in $R$ means that the controller will rely less on the input and try to use smaller control actions. The optimal feedback gain $K$ can be obtained by solving the Discrete Algebraic Riccati Equation:

$$P = A^\top PA - A^\top PB(B^\top PB + R)^{-1} B^\top PA + Q$$

Then, the feedback gain is computed as:

$$K = (B^\top PB + R)^{-1} B^\top PA$$

as the system is fully controllable. The Riccati equation has a unique positive definite solution, and the optimal feedback gain $K$ exists and can be computed accordingly. This result guarantees the control law $u = -Kx$ is capable of stabilizing the system and shaping the closed-loop dynamics with the selected weighting matrices $Q$ and $R$.

The corresponding feedback gain $K$ leads to a closed-loop matrix:

$$A_{\text{cl}} = A - BK$$

where all eigenvalues lie inside the unit circle as can be seen in figure 16. So the closed-loop system is stable and the trajectories converge to the origin.

This can also be verified by using a Lyapunov function $V(x) = x^\top Px$. For each time step, the difference is:

$$V(x_{k+1}) - V(x_k) = -x_k^\top (Q + K^\top RK) x_k$$

which is negative for any nonzero $x_k$. This implies that the energy in the system decreases over time.

## A. Computation of Q and R

The state weighting matrix $Q$ and the control weighting matrix $R$ are tuning parameters in the LQR controller that reflect the relative importance assigned to state deviations and control effort. The only objective for the control efforts is that the magnitude of the control input should not exceed one, as this will trigger the input saturation which compromises stability and performance. As the control inputs stayed well behaved and below 1 during tuning, the control weight $R$ could actually be chosen to be quite small

$$R = 10^{-2}$$

Initially, the state weighting matrix $Q$ was selected using the standard structure:

$$Q = C^\top C$$

This structure takes the measured outputs as the state cost to be minimized. However, this form treats all observable states equally and does not consider the differences in their dynamic significance and magnitude scales. With this $Q$, the Furuta pendulum could not be stabilized.

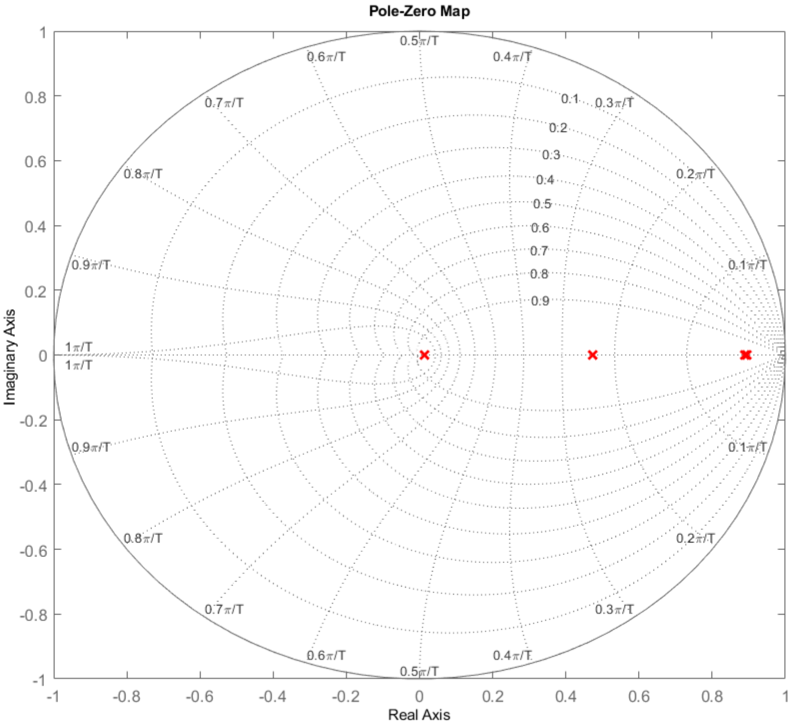


Fig. 16: Poles of the LQR controlled system

A crucial aspect in tuning the LQR weighting matrices was the state state error in $\theta$. In 19, this steady state error is seen to persist even in the finalized LQR. As mentioned at the start of this section, the LQR stabilizes the system in a fundamental way using the Riccati equation to compute a stabilizing solution which satisfies the Lyapunov decrease condition. Therefore the steady-state error in $\theta$ cannot be removed by changing the weights in $Q$, the controller *thinks* the offset state is the equilibrium. This problem is therefore due to model inaccuracy.

To deal with this, the following strategy was employed. For one, the third and fourth state that contain information about the angular velocities are incorporated in the cost function. By making $Q$ diagonal and ensuring the contribution of the term $q_3x_{3,k}^2 + q_4x_{4,k}^2$ is large, the controller prioritized zero angular velocities over a potential steady state error in $\theta$. The only way to make the angular velocities go to zero is to stabilize around the physical equilibrium and therefore the LQR stabilizes the system even though, according to the model, there is a steady state error in $\theta$. Because the states $x_3$ and $x_4$ have a large amplitude compared to $x_1$ and $x_2$, as can be seen in figure 17, the weights $q_3$ and $q_4$ did not need to be large.

By choosing $q_1$ large to achieve fast tracking we arrive at the final state weighting matrix:

$$Q = \begin{bmatrix} 10^1 & 0 & 0 & 0 \\ 0 & 10^{-1} & 0 & 0 \\ 0 & 0 & 10^{-3} & 0 \\ 0 & 0 & 0 & 10^{-3} \end{bmatrix}$$

### B. *Reference tracking*

To achieve steady-state tracking of a reference, a feedforward term $G$ is added to the control law:

$$u_k = -Kx_k + Gr$$

where $K$ is the LQR feedback gain, $r$ is the constant reference, and $G$ is the feedforward gain used to shift the steady-state value of the system toward the reference.

In this Furuta pendulum system, not all state variables are of interest. We only focus on tracking the rotary arm angle $\alpha$, keeping the pendulum arm angle $\theta$ at zero. These variables are part of the system state $x$ and are measured and used as system outputs. For this reason, it is more appropriate to consider output tracking rather than full state tracking when designing the feedforward gain.

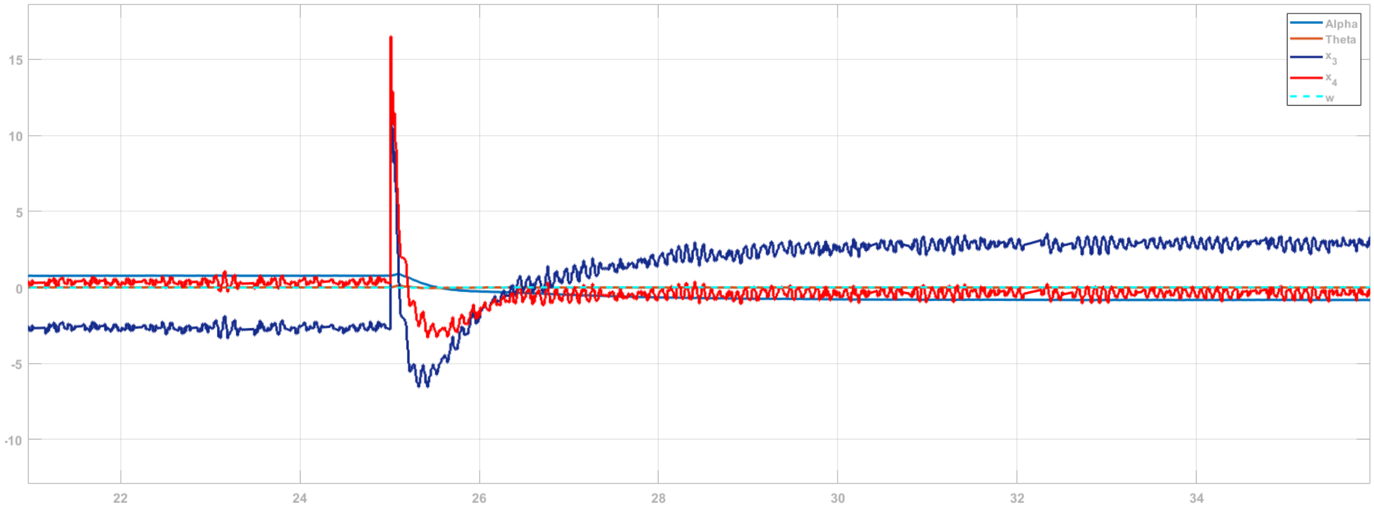


Fig. 17: The states $x_3$ and $x_4$ are large compared to states $x_1$ and $x_2$.

The output is defined as

$$y = Cx$$

where output matrix $C$ selects the two angle states from the full state vector to ensure the output $y$ reaches the reference $r$ in steady state.

At steady state, the discrete-time system satisfies:

$$x = x^+ = Ax + Bu$$

Solving for the steady-state state vector:

$$x = (I - A)^{-1}Bu$$

The steady-state output is then given by:

$$y = Cx = C(I - A)^{-1}Bu$$

The open-loop DC gain of the system is:

$$G_{\text{DC}}^{\text{open}} = C(I - A)^{-1}B$$

Substituting the control law into the system dynamics, the closed-loop DC gain is

$$G_{\text{DC}}^{\text{closed}} = C(I - A + BK)^{-1}B$$

In MATLAB, the value of $G$ can be computed using the `dcgain` function from the Control System Toolbox.

So the final value of feedward gain G is 0.242.

### C. *Simulation results*

In figure 19 the obtained LQR has been simulated to track a reference of 1. Although the rotary arm tracks the reference very well, the pendulum angle seems to have a steady state error.

### D. *Disturbance rejection*

To increase the robustness against disturbances of the controller, a disturbance rejector and estimator were added.

First, the state space can be expanded to include an estimated disturbance $\hat{w}$

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{w}_{k+1} \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & \Phi_w \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hat{w}_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_k + \begin{bmatrix} L \\ L_w \end{bmatrix} \varepsilon_k$$

To determine an optimal $L_w$, the augmented system can be used for the Kálmán filtering problem. Assuming a constant disturbance offers a particularly simple approach at rejecting disturbances. Under this assumption, $\Phi_w = 1$. Incorporating the estimated disturbance into the control law with $u_k = -K\hat{x}_k - K_w\hat{w}_k$, where K is derived be the discrete algebraic Riccati equation, will give a perfect disturbance cancellation if $K_w = 1$. By assuming the variance on the disturbance $w$ to be low

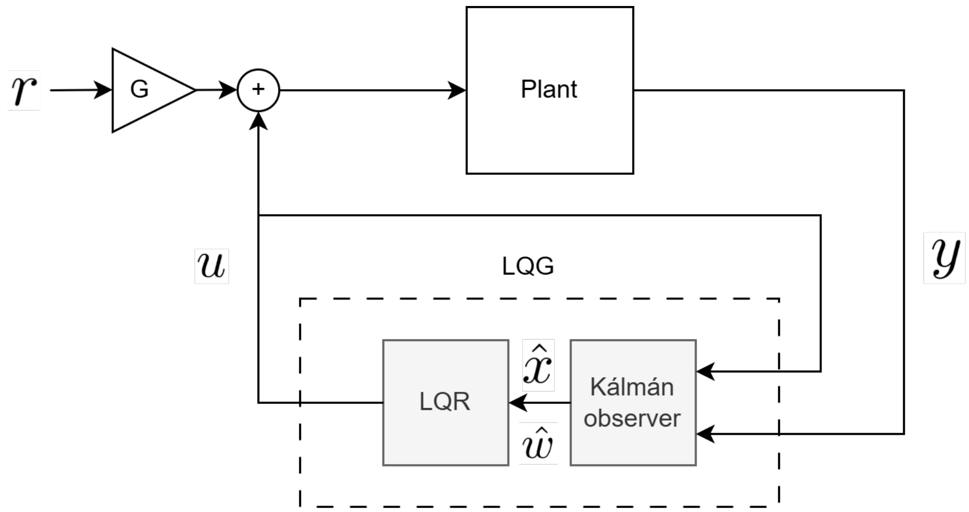


Fig. 18: Implemented LQG controller

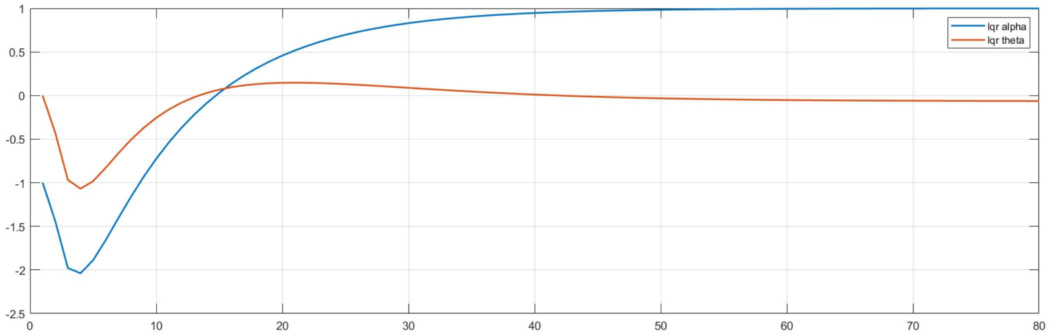


Fig. 19: Simulation of LQR over 0.8 seconds

(for example, in this project the weight $Q_w$ was chosen to be $10^{-7}$), the value of the estimate can quickly be adjusted by the observer. The obtained disturbance rejector functions as an integrator derived by output error.

As the augmented system is estimated, the augmented system needs to be detectable. The observability matrix for the augmented system is

$$\mathcal{O}_s^{aug} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 & 0 \\ 0.9996 & 0.0058 & 0.0098 & -0.0000 & 0.0990 \\ 0.0000 & 1.0121 & -0.0001 & 0.0100 & 0.0959 \\ 0.9983 & 0.0226 & 0.0190 & -0.0001 & 0.3891 \\ 0.0000 & 1.0485 & -0.0005 & 0.0201 & 0.3810 \\ 0.9962 & 0.0501 & 0.0279 & 0.0000 & 0.8621 \\ 0.0001 & 1.1094 & -0.0012 & 0.0307 & 0.8555 \\ 0.9934 & 0.0882 & 0.0363 & 0.0003 & 1.5123 \\ 0.0002 & 1.1958 & -0.0021 & 0.0418 & 1.5239 \end{bmatrix}$$

Which also can be verified to be full rank. This means the augmented system is again observable and therefore detectable

### *E. Results*

To show the stability, reference tracking and disturbance rejection capabilities of the LQR a demonstration was done. In figure 20, the results of this demonstration can be seen. The demonstration consisted of the pendulum tracking a square wave with amplitude $\pi/4$. Between 35 seconds and 70 seconds a series of physical disturbances were given in the form of ’flicking’ the end of the pendulum and arm. The demonstration shows that the controller both achieves tracking and can reject disturbances.

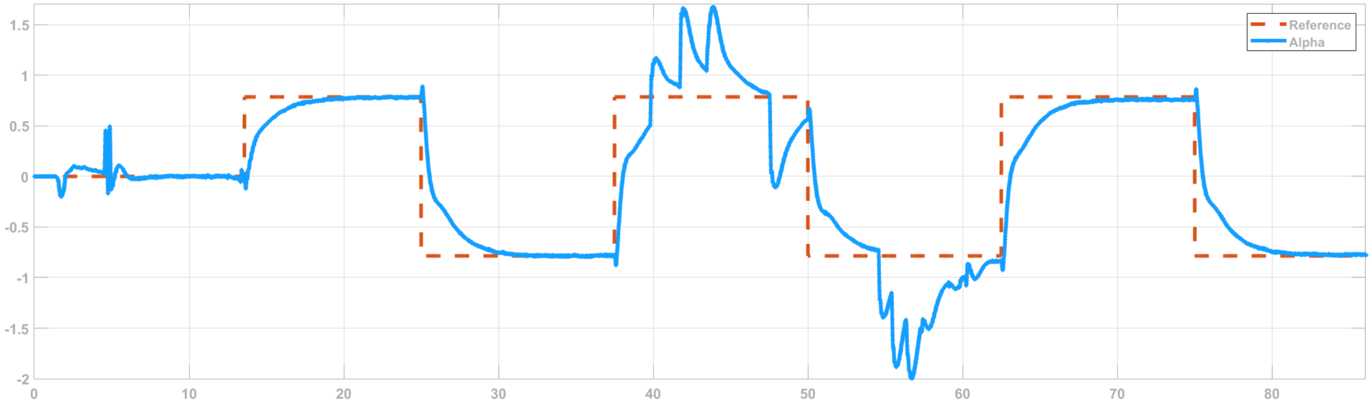


Fig. 20: Demonstration of LQR tracking and disturbance rejection

## VII. Swing-up control

In this section, the swing-up controller is designed based on nominal system parameters provided in the assignment. The reason that the additional objective of not using these parameters are relaxed for this additional controller is that to create a viable swing-up hybrid controller, an accurate estimate for these parameters is not crucial. The swing-up controller only functions as a way to bring the pendulum sufficiently close to the unstable equilibrium so that the hybrid controller can switch to the accurate linear controllers, and therefore the swing-up controller does not need to be very accurate.

The goal of the swing-up controller is to bring the pendulum up from the stable equilibrium state to the unstable equilibrium state. The LQR synthesizes until now and the upcoming MPC are linear controllers, that is, they work by using linearized model near the unstable equilibrium and therefore are only valid locally. It is not possible to drive the system directly from an arbitrary initial condition to the desired equilibrium using linear feedback control. To reach the target state, we use an energy regulation method to provide enough input to the system to achieve the swing-up.

The nonlinear swing-up controller is based on the total mechanical energy conservation of the pendulum, rather than the complete system [9]. The controller regulates the control input such that the total energy of the pendulum gradually approaches the target energy corresponding to the upright configuration. Define the potential energy for downright position to be zero.

The potential energy is given by:

$$E_p = \frac{1}{2} m_p g L_p (1 - \cos\theta)$$

The kinetic energy is given by:

$$E_k = \frac{1}{2} J_p \dot{\theta}^2$$

The total mechanical energy is then:

$$E = E_k + E_p = \frac{1}{2} J_p \dot{\theta}^2 + \frac{1}{2} m_p g L_p (1 - \cos\theta)$$

For the upright position, $\theta = \pi$ and angular velocity $\dot{\theta} = 0$, the total mechanical energy is $E_r = m_p g L_p$.

Compute the time derivative of total mechanical energy $E$:

$$\dot{E} = \dot{\theta} \left( J_p \ddot{\theta} + \frac{1}{2} m_p g L_p \sin\theta \right)$$

In a classical inverted pendulum model with a fixed pivot, the dynamics of the pendulum arm are described by:

$$J_p \ddot{\theta} + \frac{1}{2} m_p g L_p \sin\theta = 0$$

While, in the Furuta pendulum system, the pendulum is mounted on the end of a horizontally rotating arm. The pivot point accelerates linearly with acceleration $u$. So an additional inertial torque is introduced on the pendulum. To account for this effect, the dynamics are modified as:

$$J_p \ddot{\theta} + \frac{1}{2} m_p g L_p \sin\theta = \left( \frac{1}{2} m_p L_p \cos\theta \right) u$$

The additional term on the right-hand side represents the torque induced by the base acceleration acting on the pendulum's center of mass.

$u$ is the linear acceleration at the pendulum connection driven by rotary arm. It is controlled by the DC motor input and can be regulated by the input voltage.

Then we get:

$$\dot{E} = \frac{1}{2} m_p L_p \dot{\theta} \cos(\theta) u$$

Thus the energy change is proportional to the control input $u$, and depends on the angular velocity and its orientation. So we design the swing-up controller as:

$$u = k_e (E - E_r) \mathrm{sgn}(\dot{\theta} \cos\theta)$$

where $k_e > 0$ is a tunable gain, $E_r$ is the total mechanical energy for upright position. This controller ensures energy increase when the pendulum swings toward the upright direction and reduces it otherwise.

To reduce chattering, we use a saturated control law:

$$u = \mathrm{sat}_{u_{\max}} \Big( k_e (E - E_r) \mathrm{sgn}(\dot{\theta} \cos\theta) \Big) \tag{2}$$

where $\mathrm{sat}_{u_{\max}}()$ is the saturation function limiting the control signal to within $[-u_{\max}, u_{\max}]$.

To analyze stability, define a Lyapunov function as:

$$V = (E - E_r)^2$$

and its time derivative is

$$\frac{dV}{dt} = -m_p L_p k_e \Big( (E - E_r) \dot{\theta} \cos(\theta) \Big)^2$$

When $\dot{\theta} \cos\theta \neq 0$, the derivative of Lyapunov function is negative definite. So the energy will converge to the desired value $E_r$.

From DC motor model, the voltage can be approximated as $V_m \approx R_m \frac{\tau_m}{k_m}$. So the torque is $\tau_m = \frac{k_m}{R_m} V_m$.

As the motor torque is applied to an module attachment hub with radius $r_h$. The resulting linear force is:

$$F = \frac{\tau_m}{r_h} = \frac{k_m V_m}{R_m r_h}$$

So the final control law is expressed as:

$$V_m = \frac{R_m r_h m_r}{k_m} u = \frac{R_m \, r_h \, m_r}{k_m} \cdot \mathrm{sat}_{(a_{\max}, -a_{\max})} \Big( k_e (E_r - E) \, \mathrm{sgn}(\dot{\theta} \cos\theta) \Big)$$

Based on empirical testing and repeated manual adjustments, the gain is selected as $k_e = 0.15$. This value provided a satisfactory trade-off between convergence speed and overshoot during the swing-up phase.

Define the threshold angle $\varepsilon$ as $5^\circ$. Within this range, the controller transitions from the energy-based swing-up mode to a balance-up controller (LQR or MPC), which is responsible for stabilizing the pendulum around the upright equilibrium point and tracking reference signals.

This switching mechanism introduces discrete mode transitions into the control system and builds a hybrid control system. The control logic switches discretely between different controller modes based on a state-dependent condition.

The switching logic is defined as:

$$u_k = \begin{cases} u_{\text{balance},k} & \text{if } |\theta_k - \pi| < \varepsilon \\ u_{\text{swing-up},k} & \text{otherwise} \end{cases}$$

This control structure ensures the system first gains enough energy to reach the upright region and then locally stabilize around the equilibrium point by transitioning directly into a linear stabilizing strategy without any delay.

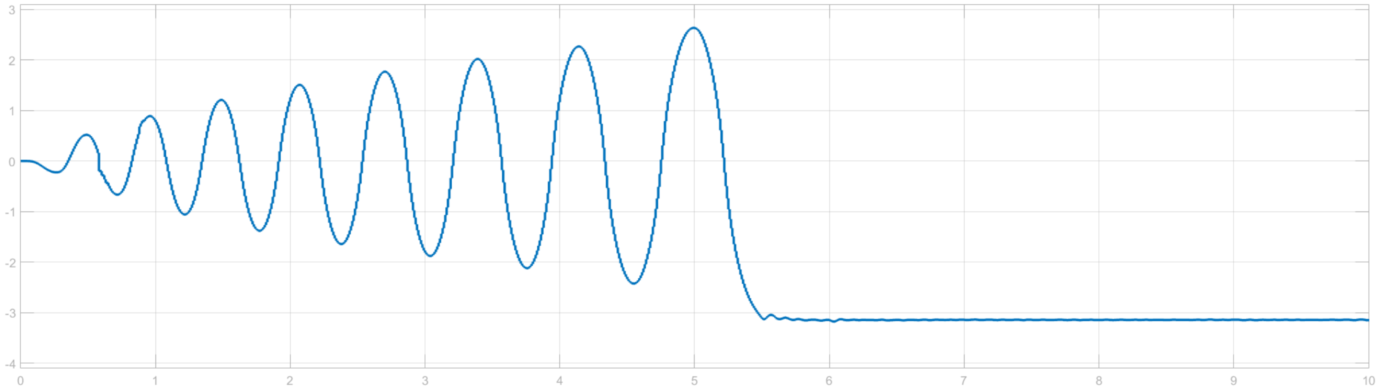


Fig. 21: Swing-up outputs for theta over time (s)

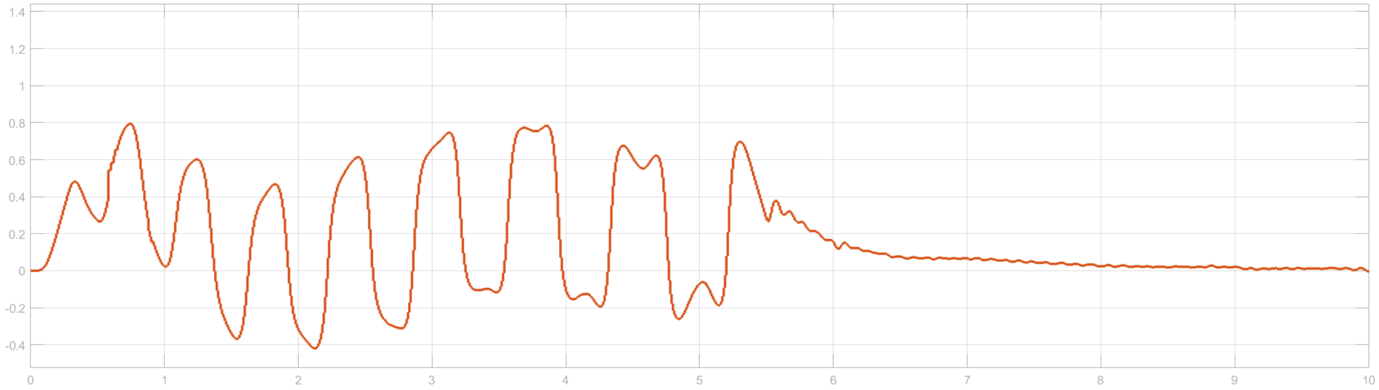


Fig. 22: Swing-up outputs for alpha over time (s)

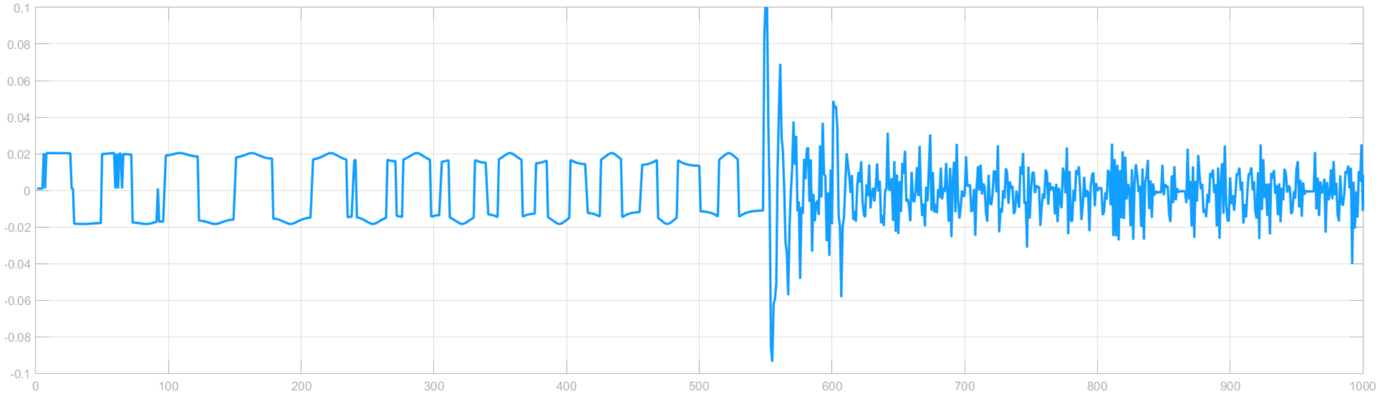


Fig. 23: Swing-up control inputs over time (ms)

## VIII. Model Predictive Control

A drawback of classical control methods like LQR are that input-, state- and output constraints cannot be directly enforced upon the system. For the case of this Furuta pendulum, the following constraints can be formulated:

1) $|u(t)| \leq 1 \quad \forall t \in \mathbb{R}$ , added as the actual voltage input is clipped as magnitude 1.
2) $|\alpha(t)| \leq 90^\circ \quad \forall t \in \mathbb{R}$, added to enforce some operating range.

The existence of these constraints motivates the implementation of an optimization based control method. The method implemented in this project is Model Predictive Controller.

### *Problem formulation*

Using the same model as used before, the linear Model Predictive Control problem consists of the following elements [10]:

1) A linearized model around an equilibrium point $x^+ = Ax + Bu$.
2) A contrained input-state set, or polytope, $\mathbb{Z} = \{(x,u)|x \in \mathbb{X}, u \in \mathbb{U}\}$, where $\mathbb{X}$ is the state constrained set and $\mathbb{U}$ is the input constrained set.
3) A control invariant terminal set $\mathbb{X}_f$
4) A prediction horizon $N$
5) A set of weight matrices $Q, R, P$
6) A cost function $J(x) = \sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k^T) + x_N^T P x_N$

The Model Predictive Controller solves an optimization algorithm at every time step. The optimization consists of minimizing the cost function $J(x)$ such that the states and input remain inside of the predefined constraint sets $\mathbb{X}$ and $\mathbb{U}$, and the final state $x_N$ will be in a terminal constraint set $\mathbb{X}_f$. Every element of the MPC will now be defined.

### A. *Linearized model*

As the name suggests, the Model Predictive Controller uses an internal model to predict future states for a given input. The standard MPC uses a linear model. For the Furuta pendulum, the same linearized model will be used as before to serve as the internal model.

### B. *State-input constraint set*

As the input $u$ has been bounded by [-1,1] by the system as a safety measure, the input magnitude should stay below a value of 1. Also, a desirable operating range of the rotary arm could be $[-\pi/2, \pi/2]$. Using this description the constraint sets can be defined as follows:

- $\mathbb{X} = \{x|\ |x_{1,k}| \leq \pi/2\}$
- $\mathbb{U} = \{u|\ |u| \leq 1\}$
- $\mathbb{Z} = \mathbb{X} \cap \mathbb{U} = \{(x,u)|\ |u| \leq 1,\ |x_{1,k}| \leq \pi/2\}$

### C. *Stability and the terminal set*

To ensure stability of an LQR controlled system, one only needs to verify the existence of a stabilizing solution $P$ to the discrete algebraic Riccati equation and use the feedback gain computed as $K = (B^\top PB + R)^{-1}B^\top PA$. Because the controls of a MPC can be constrained, LQR feedback does not imply control invariance. MPC solves this by implementing a horizon and employing dynamic programming to find a stabilizing sequence of inputs [10]. To satisfy the mathematical induction that is used to prove stability, it is essential to have a terminal state $x_N$ which is verifiably in a control invariant set.

This can be done by setting the terminal set as a invariant contraint admissable set for $x^+ = A_k x$, under the assumption that the terminal set is small enough to not activate constraints. This assumption will not be proven here (see [10] for details), but figure 24 implies that this is a reasonable assumption for most state trajectories. To compute the terminal set the `MPT3` toolbox is used. With `LQRset` the terminal set is computed by calculating the largest invariant set for the system $x^+ = (A - BK)x_k$, where $K$ is a LQR derived stabilizing feedback gain and $(x_k, u_k) \in \mathbb{Z}\ \ \forall\ x_k, u_k$. This terminal set is then a polytope of the form $Hx \leq \gamma$. A slice of the terminal set can be seen in figure 24 for $\dot{\alpha}, \dot{\theta} = 0$.

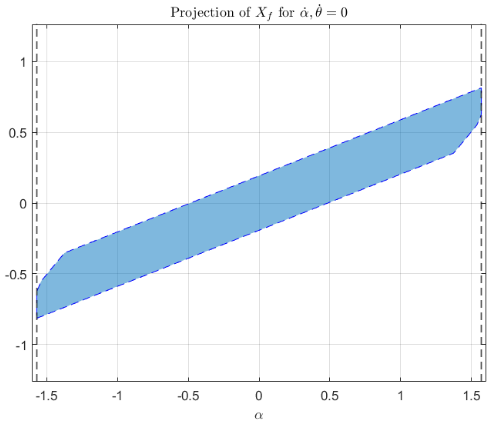


Fig. 24: Slice of the terminal set $\mathbb{X}_f$.

### D. *Terminal weighting matrix P*

As mentioned before, we want the terminal set to be a control invariant set for closed loop LQR control. Such a control would lead to the following Lyapunov equation:

$$\exists P \succ 0, \text{ such that } A_k^\top P A_k - P = -Q, \text{ where } Q \succ 0$$

Setting the terminal stage cost then to $V_f(x) = x^\top P x$ leads to Lyapunov decrease condition.

$$V_f(A_k x) = (A_k x)^\top P (A_k x) = x^\top P x - x^\top Q x < V_f(x)$$

Thus setting the terminal penalty weight to be the soluction to the discrete algebraic Riccati ensures the terminal set to be a invariant constraint admissable set for $x^+ = A_k x$ under the assumption that state and input constraints will not be active in the final stage.

### *E. Prediction horizon*

A key feature of the MPC is the prediction horizon $N$. As the terminal state is constrained, a sufficiently large horizon is needed to make sure the optimization is feasible for the desired operating range. On the other hand, a large prediction horizon leads to a higher dimensionality of the optimization problem, which leads to longer computation times. Therefore a prediction horizon needs to be found that balances these two requirements. After some experimentation, the largest prediction horizon that did not result in computational problems was a horizon of 15-20 steps.

### *F. Implementation & run time optimization*

To implement the MPC succesfully, it has to be able to compute the optimal control input within the sampling time $T_s = 0.01$ s. To achieve this, one could for example choose to use the `Model Predictive Control` toolbox `Simulink` blocks, which have the major drawback of not being able to enforce a terminal constraint set or a terminal penalty, or one could for example utilize the `mpt3` toolbox directly. For the second option, the speed constraint set by the sampling rate could not initially be fulfilled. One possibility to decrease the running time is to construct an explicit version of the MPC by calculating a piecewise affine control policy for the feasible set $\mathcal{X}$ off-line. For this system however, which is of significant dimensionality, implementing an explicit MPC would put serious constraints on the prediction horizon. The off-line computation was infeasible in terms of computation time when the prediction horizon exceeded 5 prediction steps. A solution was found by circumventing some of the `mpt3` overhead induced limitations on run time by converting the MPC to a `YALMIP` optimization problem and running the optimization algorithm directly. The run time for calculating and returning the optimal control input in simulation now stays below 0.007 s for a prediction horizon of 40 steps, decreasing the run time by more then a factor of 2.

### *G. Disturbance rejection*

To add disturbance estimating and rejection capabilities, again the state space is augmented as in section VI. The augmented state space is then used to achieve estimation of both the states and the disturbance. Then, only the states are fed into the MPC and a feedback gain of $K_w = 1$ was applied to the estimated disturbance in an outer-loop. In classical MPC, the disturbance rejection and tracking problem is normally solved by employing an Optimal Target Selection optimization that computes be ideal reference states and inputs online. Though this optimization scheme is light (it consists of minimizing a quadratic cost such that the reference states and reference inputs result in steady state under the estimated disturbances), it will add computation time as the optimization needs to be run for every timestep. This off course also introduces additional overhead which has an effect on run time as well. Because speed is the limiting factor of the MPC for this system, the Optimal Target Selection was replaced be the aforementioned much faster outer loop disturbance rejection feedback $K_w$.

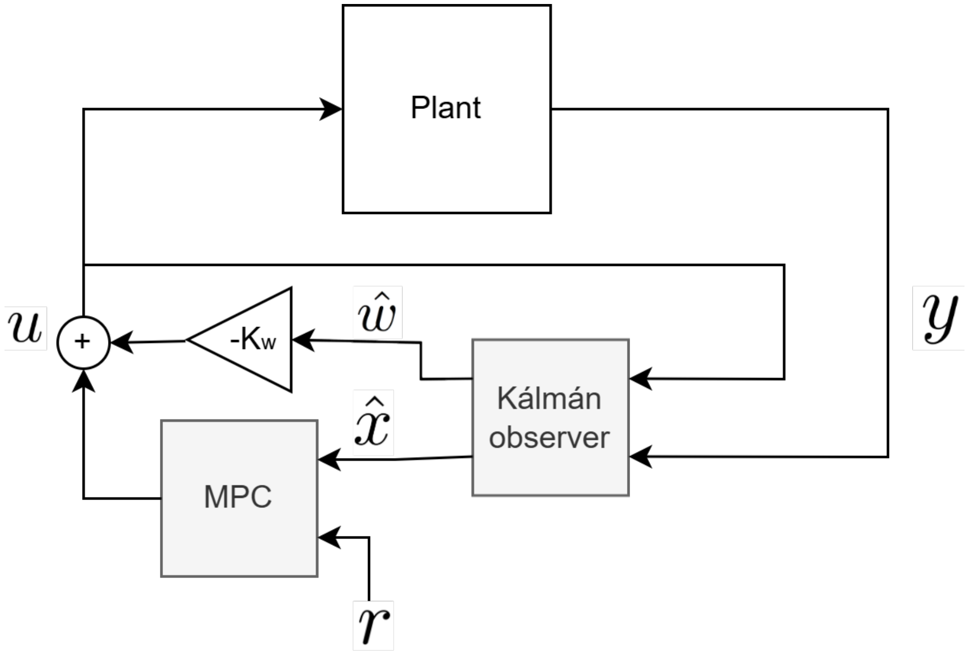


Fig. 25: Implemented Model Predictive Controller

### H. *Simulation*

To verify the run time optimized MPC beforehand, simulations were done using the linearized system dynamics. In figure 26 a both MPC and LQR state evolutions are given for an initial state $[-1, 0, 0, 0]^\top$ and a reference of 1. The MPC performs as well as the LQR, perhaps even better, while enforcing the given constraints on $\alpha$ and $u$. The slight steady state error in $\theta$ is still observed for the MPC, which is expected as it is inherent to the linearized model of the system. In figure 27 the corresponding control inputs are given for both the LQR and MPC and the MPC is seen to satisfy the input constraint $|u_k| \leq k$.

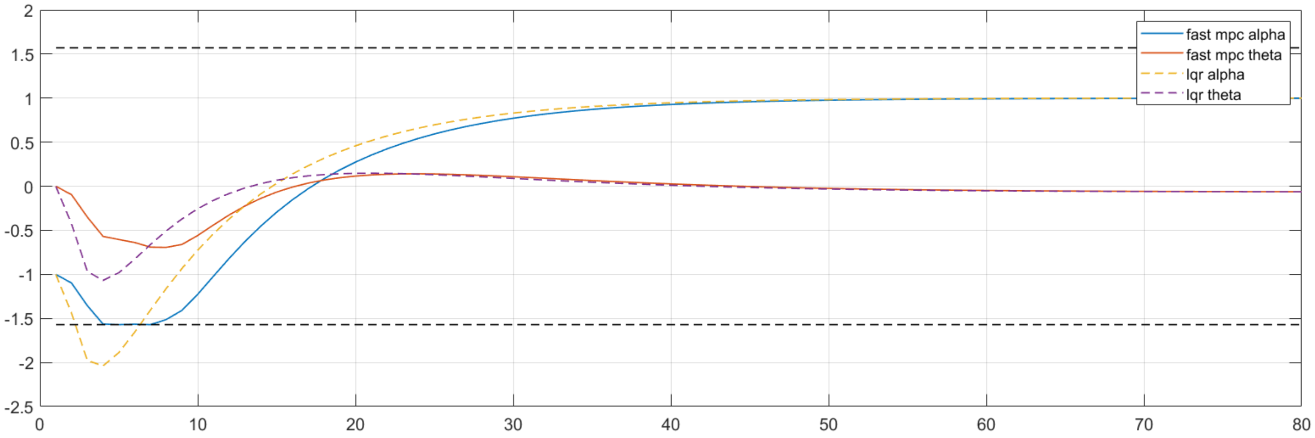


Fig. 26: Comparison LQR and MPC for prediction horizon N=20

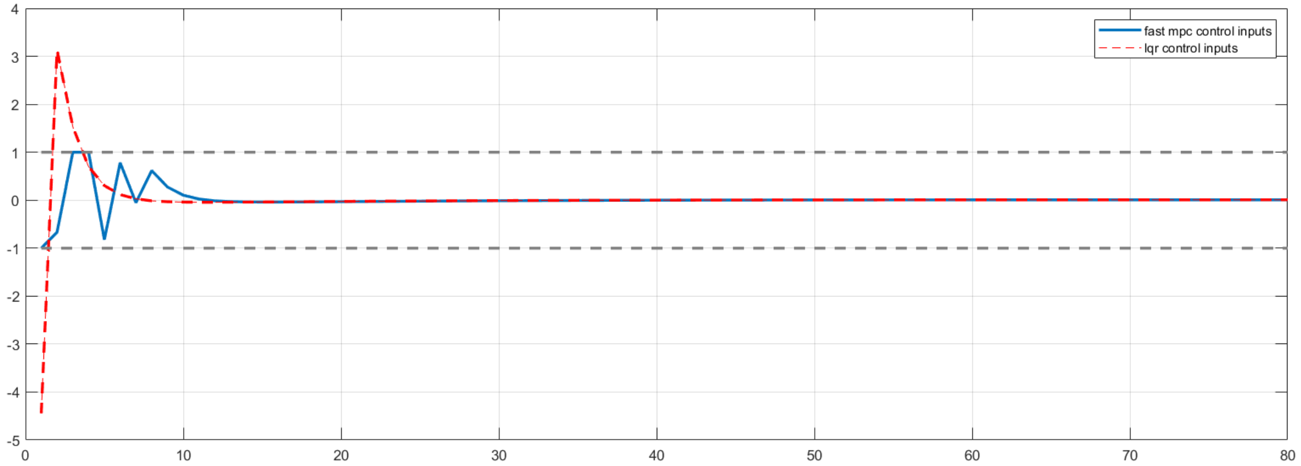


Fig. 27: Control inputs for LQR and MPC

### I. *Controller Results*

To show the stability, reference tracking and disturbance rejection capabilities of the controller, the following demonstration was done. While the controller is trying to follow a square wave with magnitude $\pi/4$, again the pendulum is perturbed by 'flicking' the end of the rotary arm and pendulum. The results of such a demonstration can be seen in figure 28. Here between 35 and 70 seconds, a number of disturbances are applied to the system, trying to push the system outside of the constraint set. The system can be seen to move in a more erratic way. This is expected, as more states in the horizon will be constrained when the system in pushed to the bounds of $\alpha$, leading to a differing behaviour from LQR optimal control.

To test if the MPC is actually enforcing the constraints to the capabilities of the physical controlled system, an experiment was conducted where the reference was a ramp signal that would eventually exceed the state constraint on $\alpha$. In figure 29, the response to such a ramp reference can be seen. The moment the pendulum reaches the state constraint, a conflicting behaviour can be seen as the controller tries to both enforce the constraint and track the (infeasible) reference. In figure 30,

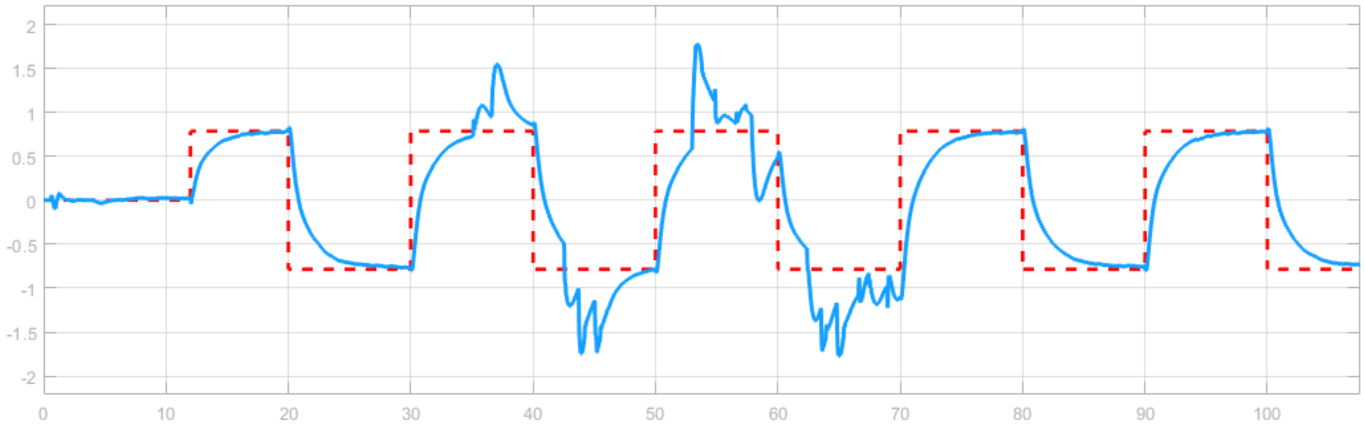


Fig. 28: Real life MPC demonstration results

the corresponding inputs are given. Input constraints can be seen to be enforced more rigorously as the controller has direct control over this.

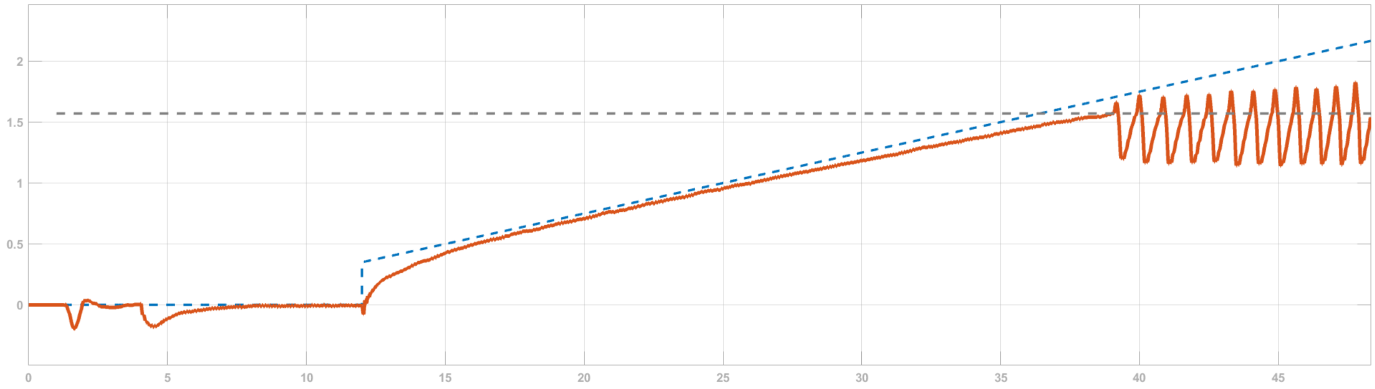


Fig. 29: Activation of the state constraint

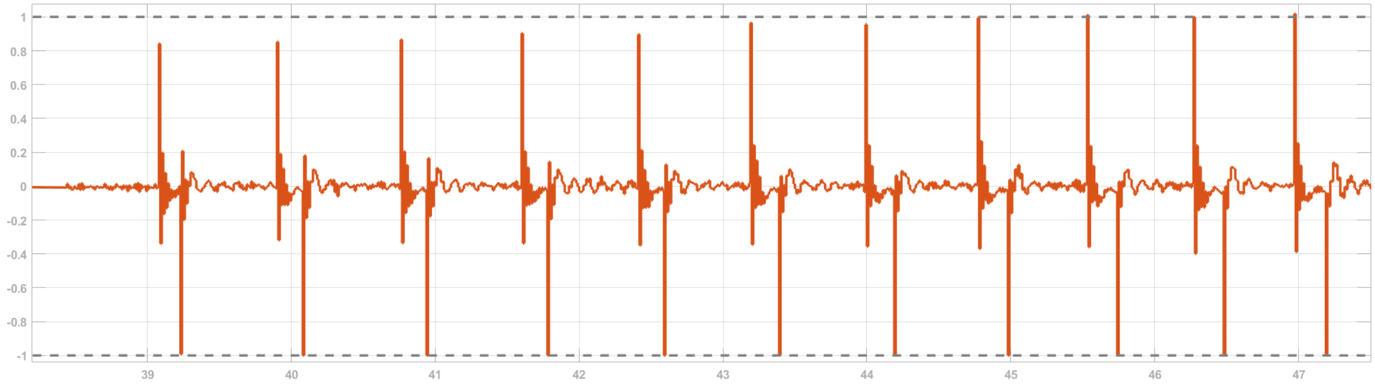


Fig. 30: Activation of the input constraint

# IX. Comparison

In this section, a comparison is made between the created LQR and MPC controllers. First, reference tracking capabilities and controller speed is discussed, then, disturbance rejection capabilities are discussed.

## A. Reference tracking & speed

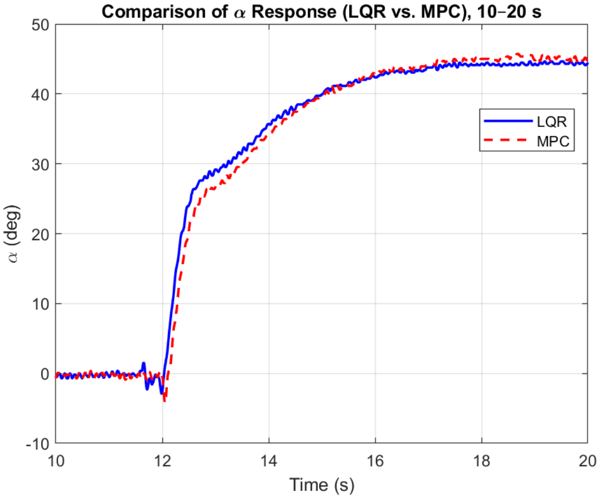
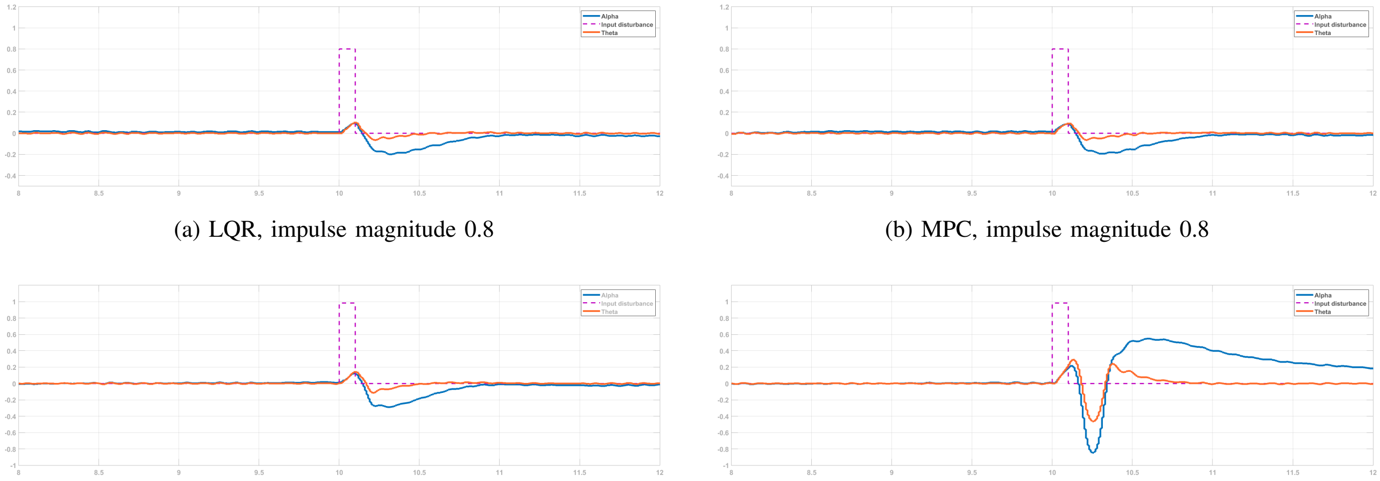


Fig. 31: Step response of LQR and MPC

In terms of reference tracking and step response speed, both LQR and MPC perform very similarly. As shown in Fig. 31, the step responses of the two controllers are nearly identical in rise time (approximately 6 seconds) and steady-state accuracy. The similarity between the LQR and MPC makes sense, as the MPC could be seen as an extension of LQR which has the ability to enforce constraints. The step response, which is to a reference of $\pi/4$, remains well inside of the state constraint set. Also for such a step input the control inputs remained well inside of the input constraint for both the LQR and MPC. As the constraints are not activated, we expect the LQR and MPC to respond near identically.

## B. Disturbance rejection

In terms of reference tracking and speed, the controllers are very comparable. It seems like the MPC performs like an LQR while also being able to enforce contstraints. However, there does seem to be a trade-off when looking at the disturbance rejection. In figure 32 the response of both the LQR (left) and MPC (right) to an input disturbance in the form of a impulse can be seen. For moderate input disturbances, the LQR and MPC again are very similar. For input disturbances close to 1, which is near the constraint $|u| < 1$, the performance of the MPC degrades while the LQR performs almost the same. Note that the disturbance rejecting capabilities for the MPC are implemented in a separate loop due to computation time constraints. The MPC therefore will not optimally account for disturbance over the complete prediction horizon. It might therefore *slightly* (as the estimated disturbance remains low compared to the states) over- or under-estimate the actual input supplied to the system. For physical disturbances in the demonstrations this did not cause any problems and was negligible. For very large input disturbances, this phenomenon does decrease performance.



(a) LQR, impulse magnitude 0.8

(b) MPC, impulse magnitude 0.8

(c) LQR, impulse magnitude 0.984

(d) MPC, impulse magnitude 0.984

Fig. 32: Disturbance rejection comparison of LQR and MPC controllers under different impulse magnitudes.

## X. Discussion

Though the use of the proposed identification method yielded sufficient results, the subspace obtained state space representation first had to be transformed and then refined via output-error method. The use of the output normal form could avoid this and enforce the desired model structure early in the identification process. Alternatively, tools like `System Identification Toolbox` in `Matlab` offer grey-box identification methods that can enforce relations between variables. This could help in estimating a more accurate model, but will need to estimate more parameters. To conclude discussion on the model, nonlinearities caused by friction and hysteresis remained a limiting factor in model performance and therefore control performance. The use of nonlinear friction models via for example (Wiener-)Hammerstein models could improve model accuracy [11]. Other significant nonlinearities in the dynamics of the system such as the Coriolis terms limited the operating range and performance of the linear controllers. A way of dealing with this could be to extend the MPC to a nonlinear controller such as a Differential Dynamic Programming controller (like the iLQR) [10], or to use partial feedback linearization [12][6]. Another significant limitation of the controllers was the need for recalibration of the feedforward gain every few days. Robust control methods could be employed to combat this and to ensure robust performance.

## XI. Conclusion

In this report, a Furuta pendulum was successfully identified, using only a first principle model and no prior information on system parameters. Subsequently, a LQR, MPC and Swing-up LQR/MPC were synthesizes to achieve the control objectives defined in section I. The identification method that is summarized in Appendix A has been showed to sufficiently identify the linearized model for control objectives. Both LQR and MPC achieved good reference tracking and disturbance rejection. The LQR and MPC function nearly identical except when states and inputs come close to the boundaries of the state and input constraint set. Then the MPC will compute an optimal control input that enforces these constraints, while the LQR cannot enforce any of these constraints. For the control objectives of this project, which did not specify any constraints, the LQR could be argued to outperform the MPC as it has superior disturbance rejection for larger input disturbances. For many applications of control outside of a lab though, there will however be state and input constraints. MPC is shown to be a great way to implement the LQR control design paradigm while accounting for such constraints.

## References

[1] B. S. Cazzolato and Z. Prime, “On the dynamics of the furuta pendulum,” *Journal of Control Science and Engineering*, vol. 2011, no. 1, p. 528341, 2011.

[2] C. Pérez-Gómez, J. Liceaga-Castro, and I. Siller-Alcalá, “Hard dead zone and friction modeling and identification of a permanent magnet dc motor non-linear model,” *WSEAS Trans. Syst. Control*, vol. 15, pp. 527–536, 2020.

[3] M. Verhaegen and V. Verdult, *Filtering and system identification: a least squares approach*. Cambridge university press, 2007.

[4] T. Roinila, H. Abdollahi, and E. Santi, “Frequency-domain identification based on pseudorandom sequences in analysis and control of dc power distribution systems: A review,” *IEEE Transactions on Power Electronics*, vol. 36, no. 4, pp. 3744–3756, 2020.

[5] S. Hachicha, M. Kharrat, and A. Chaari, “N4sid and moesp algorithms to highlight the ill-conditioning into subspace identification,” *International Journal of Automation and Computing*, vol. 11, no. 1, pp. 30–38, 2014.

[6] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.

[7] K. J. Åström and R. Murray, *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2021.

[8] B. D. Anderson and J. B. Moore, *Optimal filtering*. Courier Corporation, 2005.

[9] K. J. Åström and K. Furuta, “Swinging up a pendulum by energy control,” *Automatica*, vol. 36, no. 2, pp. 287–295, 2000.

[10] J. B. Rawlings, D. Q. Mayne, M. Diehl *et al.*, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.

[11] A. Brouri, "Identification of nonlinear systems structured by wiener- hammerstein model," *International Journal of Electrical and Computer Engineering*, vol. 6, pp. 167–176, 02 2016.

[12] K. Pathak, J. Franch, and S. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 505–513, 2005.

## XII. Appendix

### A. Generalized algorithm for identification of Lagrangian systems

As state spaces in which the state is defined as the generalized coordinates and the first derivatives of those generalized coordinates are often derived from the Euler-Lagrange method, such state spaces are here denoted as Lagrangian systems. The findings from the identification of the Furuta pendulum have been generalized to a broader class of systems, given in algorithm XII-A.

**Algorithm 1** Subspace-PEM identification for (Lagrangian) systems without initial guess

```
Given: A (Lagrangian) system where $x \in \mathbb{R}^{2n}$ is given as $[q_1, q_2, ..., q_n, \dot{q}_1, \dot{q}_2, ..., \dot{q}_n]^T$, $y \in \mathbb{R}^n$ is independent of the input $u \in \mathbb{R}^m$ and given as $[q_1, q_2, ..., q_n]^T$.
Required: $\begin{bmatrix} C_0 \\ C_0 A_0 \end{bmatrix}$ full rank, needs to be verified a posteriori.
1: Run an experiment around a stable equilibrium point
2: Run a PO-MOESP algorithm using the experiment data to obtain an initial realization $\left[\begin{array}{c|c} A_0 & B_0 \\ \hline C_0 & D_0 \end{array}\right]$
3: Compute the state transformation $T = \begin{bmatrix} C_0 \\ C_0 A_0 \end{bmatrix}$ that puts the initial realization into the structured form where,
       $C_s = \begin{bmatrix} I_n & 0 \end{bmatrix}, \quad A_s = \left[\begin{array}{c|c} 0 & I_n \\ \hline \Phi_1 & \Phi_2 \end{array}\right], \quad \text{with } \Phi_1, \Phi_2 \in \mathbb{R}^n$
4: Tune entries if necessary based on physical insights
5: Run a PEM algorithm that enforces the structure $B = \left[\begin{array}{c} 0 \\ \hline \Phi_3 \end{array}\right]$ where $\Phi_3 \in \mathbb{R}^{n \times m}$
```