Project Report

Group 14 (1C) Rotational Wheel Pendulum

by

Pieter van Rhee & Soham Prajapati

Student Name	Student Number
Pieter van Rhee	5407281
Soham Prajapati	5712297

TUDelft

Contents

1 First Principles Modelling

In this report, we will discuss the integration project for the master Systems & Control TU Delft. The objective is to build two different controllers to do both regulation and reference tracking in the unstable equilibrium point of a rotational pendulum.

1.1. Euler-Lagrange Modelling

In this section, the underlying physics of the rotational wheel pendulum are derived using Euler-Lagrange equations. The rotational wheel pendulum can be simplified as a double pendulum with a first arm as a compound pendulum and second arm as a point mass. The schematic of this simplified description is given below with all the necessary notations.

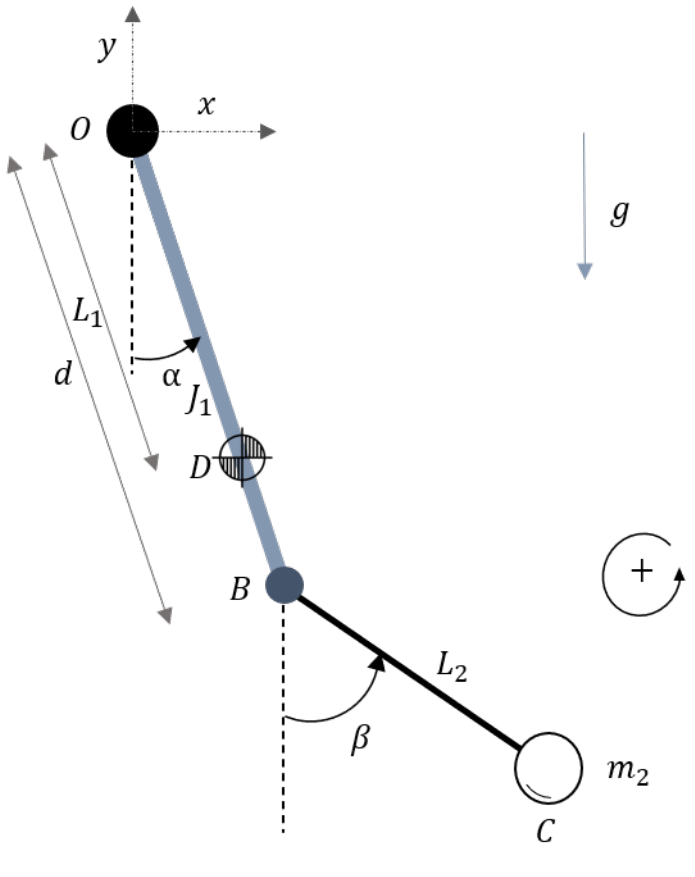

Figure 1.1: Schematic of a planar Double Pendulum

From above figure, α and β are the angular displacement of the first (OB) and second (BC) arms respectively with respect to the vertical axis. Anti-clockwise direction is considered positive. Points O and B are the axis of rotation of arms OB and BC respectively. The first arm OB is a compound pendulum of length d with the centre of gravity at distance L_1 at point D. The second arm has a point mass m_2 at distance L_2 from its axis of rotation B.

A theoretical model of the dynamics of the system is now derived below using α and β as the generalized coordinates in the Lagrange framework. The units of all the variables defined in this model are given in Table 1.1.

The initial modelling accounts to understand detailed underlying nonlinear dynamics of the double pendulum. In the later stages, the derived model will be simplified further for the sake of linear identification.

Symbol	α	β	d	L_1	L_2	g
Unit	rad	rad	m	m	m	m s^{-2}
Symbol	J_1	m_1	m_2	c_1	c_2	u
Unit	kgm^2	kg	kg	Nms	Nms	Nm

Table 1.1: Notations and their units

With O as the origin, the position of point mass m_2 can be given as $\mathbf{r} = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} dsin(\alpha) + L_2 sin(\beta) \\ -dcos(\alpha) - L_2 cos(\beta) \end{pmatrix}$, which follows that $\dot{\mathbf{r}} = \begin{pmatrix} dcos(\alpha)\dot{\alpha} + L_2 cos(\beta)\dot{\beta} \\ dsin(\alpha)\dot{\alpha} + L_2 sin(\beta)\dot{\beta} \end{pmatrix}$. Now, the kinetic energy of arms OB and BC can be formulated as $K_1 = \frac{1}{2} J_1 \dot{\alpha}^2$ and $K_2 = \frac{1}{2} m_2 \dot{\mathbf{r}}^T \dot{\mathbf{r}}$ where $\dot{\mathbf{r}}^T \dot{\mathbf{r}} = d^2\dot{\alpha}^2 + L_2^2\dot{\beta}^2 + 2dL_2\dot{\alpha}\dot{\beta}cos(\alpha - \beta)$.

Similarly, the potential energy of OB and BC can be obtained as $P_1 = -m_1 g L_1 cos(\alpha)$ and $P_2 = -m_2 g(dcos(\alpha) + L_2 cos(\beta))$.

The equations of motion can be derived using Lagrangian mechanics where they take the following form

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i \tag{1.1}$$

where, L is the Lagrangian, q_i is the ith generalized coordinate and Q_i is the generalized force acting on the ith coordinate system.

The Lagrangian above denoted by L can be defined as,

$$\begin{aligned} L &= K_1 + K_2 - P_1 - P_2 \\ &= \frac{1}{2} J_1 \dot{\alpha}^2 + \frac{1}{2} m_2 (\dot{\mathbf{r}}^T \dot{\mathbf{r}}) + m_1 g L_1 cos(\alpha) + m_2 g(dcos(\alpha) + L_2 cos(\beta)) \\ \implies \frac{d}{dt}\frac{\partial L}{\partial \dot{\alpha}} &= (J_1 + m_2 d^2)\ddot{\alpha} + m_2 d L_2 \ddot{\beta} cos(\alpha - \beta) - m_2 d L_2 \dot{\beta}(\dot{\alpha} - \dot{\beta}) sin(\alpha - \beta) \\ \frac{\partial L}{\partial \alpha} &= -m_2 d L_2 \dot{\alpha}\dot{\beta} sin(\alpha - \beta) - g sin(\alpha)(L_1 m_1 + d m_2) \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{\beta}} &= m_2 L_2^2 \ddot{\beta} + m_2 d L_2 \ddot{\alpha} cos(\alpha - \beta) - m_2 d L_2 \dot{\alpha}(\dot{\alpha} - \dot{\beta}) sin(\alpha - \beta) \\ \frac{\partial L}{\partial \beta} &= m_2 d L_2 \dot{\alpha}\dot{\beta} sin(\alpha - \beta) - g sin(\beta) L_2 m_2 \end{aligned}$$

Since the The generalized forces can further be written with $D = \frac{1}{2} c_1 \dot{\alpha}^2 + \frac{1}{2} c_2 \dot{\beta}^2$ accounting for damping due to coulomb friction in both the arms modelled by constants c_1 and c_2 for the first arm and second arm respectively.

$$\begin{aligned} Q_i &= F - \frac{\partial D}{\partial \dot{q}_i} \\ \implies Q_1 &= u - c_1 \dot{\alpha} \\ Q_2 &= -c_2 \dot{\beta} \end{aligned}$$

For the rotational pendulum setup, u can be modeled as the torque generated by the motor driving the first pendulum arm. The actual input is the voltage on the motor. One could take into account the transition from voltage to torque by modeling the motor. However, in this work it is assumed that the torque is proportional to the voltage.

Substituting above results give two differential equations for the nonlinear dynamics of both the arms.

$$\begin{aligned} (J_1 + m_2 d^2)\ddot{\alpha} + m_2 d L_2 \ddot{\beta} cos(\alpha - \beta) + m_2 d L_2 \dot{\beta}^2 sin(\alpha - \beta) + g sin(\alpha)(L_1 m_1 + d m_2) &= u - c_1 \dot{\alpha} \\ m_2 L_2^2 \ddot{\beta} + m_2 d L_2 \ddot{\alpha} cos(\alpha - \beta) - m_2 d L_2 \dot{\alpha}^2 sin(\alpha - \beta) + g sin(\beta) L_2 m_2 &= -c_2 \dot{\beta} \end{aligned} \tag{1.2}$$

Define $\mathbf{q} = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}$ and using equation 1.2,

$$\begin{aligned}&\begin{pmatrix}J_1 + m_2d^2 & m_2dL_2cos(\alpha-\beta)\\ m_2dL_2cos(\alpha-\beta) & m_2L_2^2\end{pmatrix}\begin{pmatrix}\ddot{\alpha}\\ \ddot{\beta}\end{pmatrix}\\ &+\begin{pmatrix}c_1 & m_2dL_2\dot{\beta}sin(\alpha-\beta)\\ -m_2dL_2\dot{\alpha}sin(\alpha-\beta) & c_2\end{pmatrix}\begin{pmatrix}\dot{\alpha}\\ \dot{\beta}\end{pmatrix}+\begin{pmatrix}(L_1m_1+dm_2)gsin(\alpha)\\ m_2gL_2sin(\beta)\end{pmatrix}=\begin{pmatrix}u\\ 0\end{pmatrix}\end{aligned} \tag{1.3}$$

Equation 1.3 can be equivalently represented as $\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{K}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \bar{\mathbf{B}}u$.

1.2. Linearization

Define the states of the system as $\mathbf{x} := [x_1, x_2, x_3, x_4]^T := [\alpha, \beta, \dot{\alpha}, \dot{\beta}]^T = [\mathbf{q}, \dot{\mathbf{q}}]^T$.

$$\begin{aligned}\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{K}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) &= \bar{\mathbf{B}}u\\ \implies \dot{\mathbf{x}} &= \begin{pmatrix}\dot{\mathbf{q}}\\ \mathbf{H}^{-1}(\bar{\mathbf{B}}u - \mathbf{K}\dot{\mathbf{q}} - \mathbf{G})\end{pmatrix}\\ &= \mathbf{f}(\mathbf{x}, u)\end{aligned} \tag{1.4}$$

Computing

$$\mathbf{H}(\mathbf{q})^{-1} = \frac{1}{\Delta}\begin{bmatrix}m_2L_2^2 & -m_2dL_2cos(\alpha-\beta)\\ -m_2dL_2cos(\alpha-\beta) & J_1 + m_2d^2\end{bmatrix}$$

where, $\Delta = J_1m_2L_2^2 + m_2^2L_2^2d^2(1 - cos^2(\alpha-\beta))$
The first order Taylor series approximation around a fixed point $(\mathbf{x}^*, u^*)$ is given by,

$$\begin{aligned}\mathbf{f}(\mathbf{x}, u) &\approx \mathbf{f}(\mathbf{x}^*, u^*) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x} - \mathbf{x}^*) + \frac{\partial \mathbf{f}}{\partial u}(u - u^*)\\ &= \mathbf{f}(\mathbf{x}^*, u^*) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\mathbf{x}_\Delta + \frac{\partial \mathbf{f}}{\partial u}u_\Delta\end{aligned} \tag{1.5}$$

At point $(\mathbf{x}^*, u^*) = (\mathbf{0}, 0)$, from equation 1.4 since $u = 0$ and $sin(q) = 0 \implies \mathbf{G} = \mathbf{0}$, $\mathbf{f}(\mathbf{x}^*, u^*) = \mathbf{0}$. Thus, $(\mathbf{0}, 0)$ is an equilibrium point of the system (pendulum hanging down). The linear state space equation at this equilibrium can be given as,

$$\begin{aligned}\mathbf{f}(\mathbf{x}, u) &\approx \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\mathbf{x}_{\boldsymbol{\Delta}} + \frac{\partial \mathbf{f}}{\partial u}u_\Delta\\ &= \mathbf{Ax} + \mathbf{Bu}\end{aligned} \tag{1.6}$$

where $\mathbf{A} \in \mathbb{R}^{4\times4}$ and $\mathbf{B} \in \mathbb{R}^{4\times1}$ given as,

$$\mathbf{A} = \begin{bmatrix}\mathbf{0}_{2\times2} & \mathbf{I}_{2\times2}\\ -\mathbf{H}^{-1}\frac{\partial \mathbf{G}}{\partial \mathbf{q}} & -\mathbf{H}^{-1}\mathbf{K}\end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix}\mathbf{0}_{2\times1}\\ \mathbf{H}^{-1}\bar{\mathbf{B}}\end{bmatrix} \tag{1.7}$$

Computing

$$-\mathbf{H}^{-1}(\mathbf{0})\frac{\partial \mathbf{G}(\mathbf{0})}{\partial \mathbf{q}} = -\frac{1}{J_1m_2L_2^2}\begin{bmatrix}(L_1m_1 + dm_2)gm_2L_2^2 & -m_2^2gdL_2^2\\ -m_2dL_2(L_1m_1 + dm_2)g & (J_1 + m_2d^2)(m_2gL_2)\end{bmatrix}$$

$$-\mathbf{H}^{-1}(\mathbf{0})\mathbf{K}(\mathbf{0},\mathbf{0}) = -\frac{1}{J_1m_2L_2^2}\begin{bmatrix}m_2c_1L_2^2 & -m_2c_2dL_2\\ -m_2dL_2c_1 & (J_1 + m_2d^2)c_2\end{bmatrix}$$

Thus, denoting the equilibrium point $(\mathbf{x}^*, u^*) = (\mathbf{0}, 0)$ as case 1 configuration,

$$\mathbf{A}_1 = \begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ -\frac{(L_1m_1+dm_2)g}{J_1} & \frac{m_2gd}{J_1} & -\frac{c_1}{J_1} & \frac{c_2d}{J_1L_2}\\ \frac{gd(L_1m_1+dm_2)}{J_1L_2} & -\frac{g(J_1+m_2d^2)}{J_1L_2} & \frac{c_1d}{J_1L_2} & -\frac{c_2(J_1+m_2d^2)}{J_1m_2L_2^2}\end{bmatrix} \qquad \mathbf{B}_1 = \begin{bmatrix}0\\ 0\\ \frac{1}{J_1}\\ -\frac{d}{J_1L_2}\end{bmatrix} \tag{1.8}$$

Let case 2 configuration for the pendulum be when the first arm is inverted but second arm is hanging down, that is, at point $(\mathbf{x}^*, u^*) = ([\pi, 0, 0, 0], 0)$ since $\mathbf{f}(\mathbf{x}^*, u^*) = \mathbf{0}$, from equation 1.4 and 1.5,

$$\mathbf{A}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{(L_1 m_1 + d m_2) g}{J_1} & -\frac{m_2 g d}{J_1} & -\frac{c_1}{J_1} & -\frac{c_2 d}{J_1 L_2} \\ \frac{g d (L_1 m_1 + d m_2)}{J_1 L_2} & -\frac{g (J_1 + m_2 d^2)}{J_1 L_2} & -\frac{c_1 d}{J_1 L_2} & -\frac{c_2 (J_1 + m_2 d^2)}{J_1 m_2 L_2^2} \end{bmatrix} \quad \mathbf{B}_2 = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{J_1} \\ \frac{d}{J_1 L_2} \end{bmatrix} \tag{1.9}$$

Let case 3 configuration for the pendulum be when the first arm is hanging down, but the second arm is inverted, that is, at point $(\mathbf{x}^*, u^*) = ([0, \pi, 0, 0], 0)$ since $\mathbf{f}(\mathbf{x}^*, u^*) = \mathbf{0}$, from equation 1.4 and 1.5,

$$\mathbf{A}_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{(L_1 m_1 + d m_2) g}{J_1} & \frac{m_2 g d}{J_1} & -\frac{c_1}{J_1} & -\frac{c_2 d}{J_1 L_2} \\ -\frac{g d (L_1 m_1 + d m_2)}{J_1 L_2} & \frac{g (J_1 + m_2 d^2)}{J_1 L_2} & -\frac{c_1 d}{J_1 L_2} & -\frac{c_2 (J_1 + m_2 d^2)}{J_1 m_2 L_2^2} \end{bmatrix} \quad \mathbf{B}_3 = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{J_1} \\ \frac{d}{J_1 L_2} \end{bmatrix} \tag{1.10}$$

Similarly, let case 4 configuration for the pendulum be when the first arm and second arm both are inverted, at point $(\mathbf{x}^*, u^*) = ([\pi, \pi, 0, 0], 0)$ since $\mathbf{f}(\mathbf{x}^*, u^*) = \mathbf{0}$, from equation 1.4 and 1.5,

$$\mathbf{A}_4 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{(L_1 m_1 + d m_2) g}{J_1} & -\frac{m_2 g d}{J_1} & -\frac{c_1}{J_1} & \frac{c_2 d}{J_1 L_2} \\ -\frac{g d (L_1 m_1 + d m_2)}{J_1 L_2} & \frac{g (J_1 + m_2 d^2)}{J_1 L_2} & \frac{c_1 d}{J_1 L_2} & -\frac{c_2 (J_1 + m_2 d^2)}{J_1 m_2 L_2^2} \end{bmatrix} \quad \mathbf{B}_4 = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{J_1} \\ -\frac{d}{J_1 L_2} \end{bmatrix} \tag{1.11}$$

For output to be first two-states: α, β, and no feedforward in the system, we define,

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{1.12}$$

$$\mathbf{D} = 0 \tag{1.13}$$

Thus, a linear time-invariant system in continuous time for ith equilibrium can be written as,

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}_i \mathbf{x} + \mathbf{B}_i u \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}u \end{aligned} \tag{1.14}$$

Equations 1.8, 1.9, 1.10 and 1.11 that the system and input matrices $\mathbf{A}_i$ and $\mathbf{B}_i$ have similar structure for all four equilibrium: stable and unstable. Only some entries have a sign change which is due to the change in direction of the restoring force for stable and unstable equilibrium. This will be essential for linear identification discussed in the next chapter.

2
Identification

The linear dynamics for all the equilibrium points discussed in the previous section have similar structures and represent the dynamics of the perturbed states around the equilibrium. To extract the dynamics in a statistically optimal way, system identification can be used on our system. In particular, since we know the system dynamics for all the equilibrium points, grey-box model based system identification will be employed. The below figure from [4] shows the expressive advantage of grey-box modelling as compared to white-box and black-box modelling such that it minimizes the total estimation error.

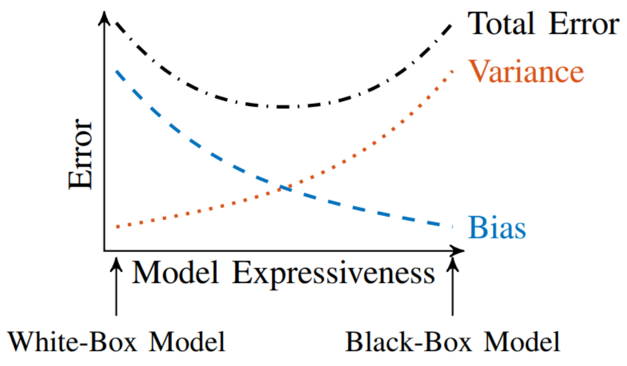

Figure 2.1: The bias-variance tradeoff as a function of model expressiveness

2.1. Calibration

For this work, the modeled angular displacement of the first arm and the second arm are denoted by α and β as discussed in the last chapter. Whereas, for pendulum setup, the sensor measurements can be denoted as θ_1 and θ_2 for the first and second arms respectively. As shown in figure 1.1, the theoretical model works with α and β angles relative to the vertical axis. However, the sensors give θ_1 relative to the vertical axis for the first arm (OB) and θ_2 relative to the first arm. Also the positive direction of angular displacement differs between the sensors and the model. To account for this $\alpha = -\theta_1$ and $\beta = \alpha - \theta_2$. These transformations are applied within Simulink.

Calibrating the setup at case 1, an offset was added to the measurements such that α and β are zero in the desired position. For a full rotation $max(\alpha) - min(\alpha) \approx 5$ the same thing holds for β. To work in radians the measurements are multiplied by $\approx \frac{2\pi}{5}$.

Some of the desired control operates close to $\alpha = \pi$ rad or $\beta = \pi$ rad. Calibrating around $\alpha = 0$ rad or $\beta = 0$ rad and adding π is not fully accurate and gives a slight offset. To account for this the calibration offset is changed depending on the operation point of the controller.

2.2. Drop test for second arm

Prediction error methods (PEM) will be used for the identification. To prepare for this good initial guesses are needed, to not end up in sub-optimal local minima. Using a simple drop test for the second arm of the pendulum, initial guesses for L_2 and c_2/m_2 are made. For the drop test the β angle is measured for an initial condition response starting from $\boldsymbol{x} = \begin{pmatrix} 0\ rad & \approx 0.5\ rad & 0\ rad/s & 0\ rad/s \end{pmatrix}^T$.

$$T \approx \frac{2\pi}{\sqrt{\frac{g}{L}}} \tag{2.1}$$

From equation 2.1, where $T \approx 0.595$ seconds (follows from figure 2.2) is the oscillation period in the drop test, it follows that $L_2 \approx \frac{g}{(\frac{2\pi}{T})^2} \approx 0.089$ m.

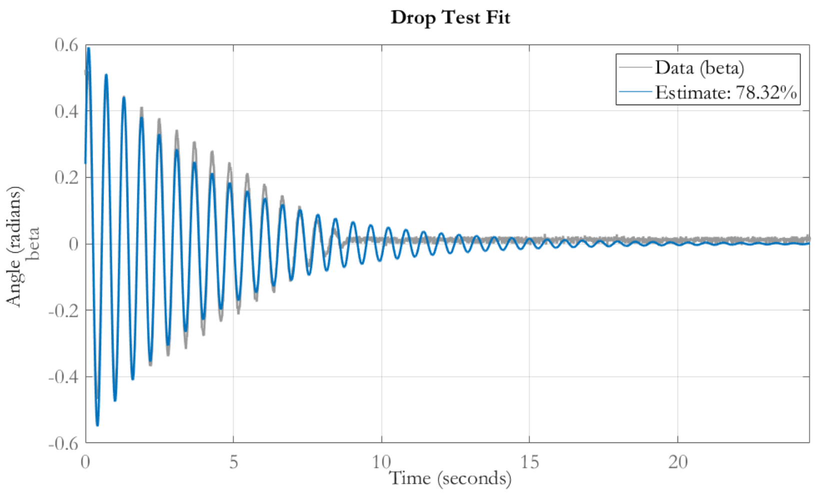

Figure 2.2: Drop Test Fit

The dynamics of the drop test can be modeled by a simplified state space model. With $\mathbf{x} = \begin{pmatrix}\beta \\ \dot{\beta}\end{pmatrix}$, $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ \frac{-g}{L_2} & \frac{c_2}{m_2}\frac{1}{L_2^2}\end{pmatrix}$, $\mathbf{B} = \begin{pmatrix}0 \\ 0\end{pmatrix}$, $\mathbf{C} = \begin{pmatrix}1 & 0\end{pmatrix}$ and $\mathbf{D} = \begin{pmatrix}0\end{pmatrix}$. In this model only $\frac{c_2}{m_2}$ is unknown. An initial response of the model was fitted to the drop test data for a range of $\frac{c_2}{m_2}$ values. From this an approximation of $\frac{c_2}{m_2} \approx 0.0038$ was derived which gave the best fit to the data, as shown in figure 2.2. The estimation accuracy is assessed using root mean square as the metric between the data and estimation. These values will be used as initial guesses in later identification steps.

2.3. Experiments

For this system, there are four equilibrium points. In order to extract the complete behavior of the system without any human interference, it is essential to identify the stable equilibrium. For unstable equilibrium, any small displacement will result in forces or torques that move the system away from the linearized environment, escalating the perturbed states. So, out of the two stable equilibrium points, we arbitrarily choose equation 1.8 to begin with data collection of inputs and outputs. It follows that the linearized system can be used for all equilibrium points by only changing some minus signs.

2.3.1. Experiment Design

Data was derived by giving cosine waves as an input voltage on the motor and measuring the angles α and β as the output. The initial position for these experiments was $\boldsymbol{x} = \begin{pmatrix}0\ rad & 0\ rad & 0\ rad/s & 0\ rad/s\end{pmatrix}^T$. It was chosen to use cosine waves instead of sine waves such that the pendulum would oscillate symmetrically about the vertical axis. Still, the pendulum showed some drift due to some motor dynamics (like back-emf) or static friction which wasn't modeled. This drift was a result of accumulated low-frequency components in the signal. To deal with this a high pass filter was used such that the model identification isn't based on this drift (2.3.2). Experiments were done for frequencies between 4 rad/s and 70 rad/s. It was tried to identify on frequencies that are close to what the system will be experiencing during its control runs. The lower range of these frequencies is similar to the frequencies that will be used for the reference signal when reference tracking. Also, these lower frequencies are close to the autonomous drop test frequency, so it represents what the controller needs to control for in the stable equilibrium. When stabilizing the unstable equilibrium points the controller will need to respond quickly so it should also know how the system reacts for high frequencies. That is why it was chosen to also identify on frequencies as high as 70 rad/s.

2.3.2. Data processing

A high pass filter was used to counteract drift. It was chosen to use a simple Simulink transfer function block to process the incoming data in real time. Offline filtering would cause less phase distortions, but this real-time filter is not complicated and was good enough for the job.

A first-order filter is used in the form of $\frac{s}{s+\tau}$. Where τ is tuned until the desired performance is visible. In this case $\tau = 1.2$, which corresponds with a cutoff frequency of $f_c = \frac{1}{\tau} \approx 0.833\ rad/s$. In the background, Simulink discretizes this function to become $\frac{z-1}{z-0.9881}$ to deal with discrete data. This filters away the unwanted low frequencies while keeping the actuated frequencies that are larger than $0.833\ rad/s$. In figures 2.3 and 2.4 the performance of the filter is visualized. It shows that the drift is counteracted as needed. Notice that this filter is only used for identification and not for control.

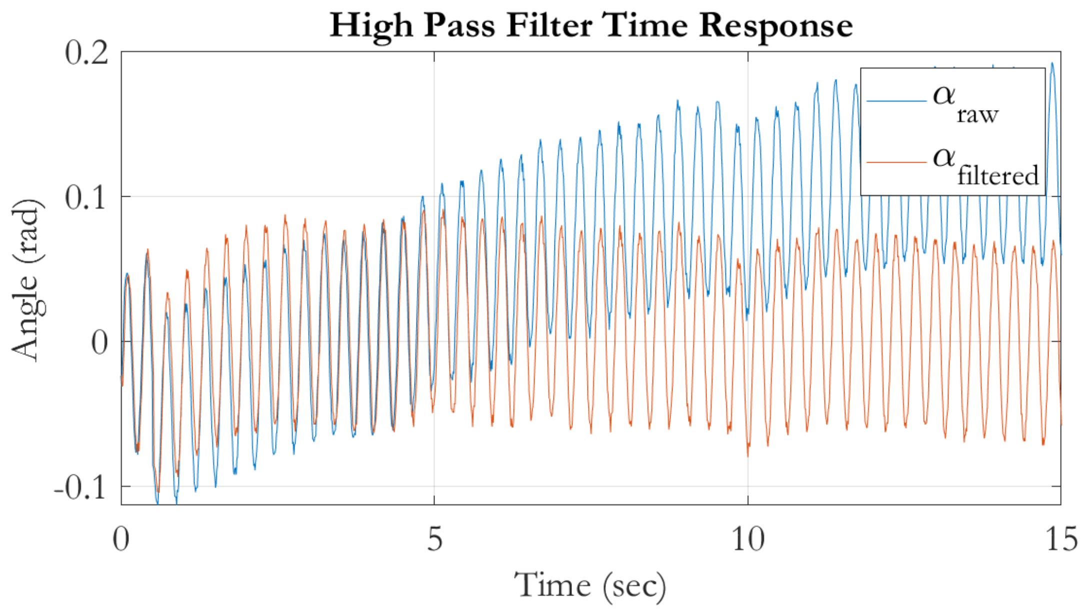

Figure 2.3: High Pass Filter Time Response

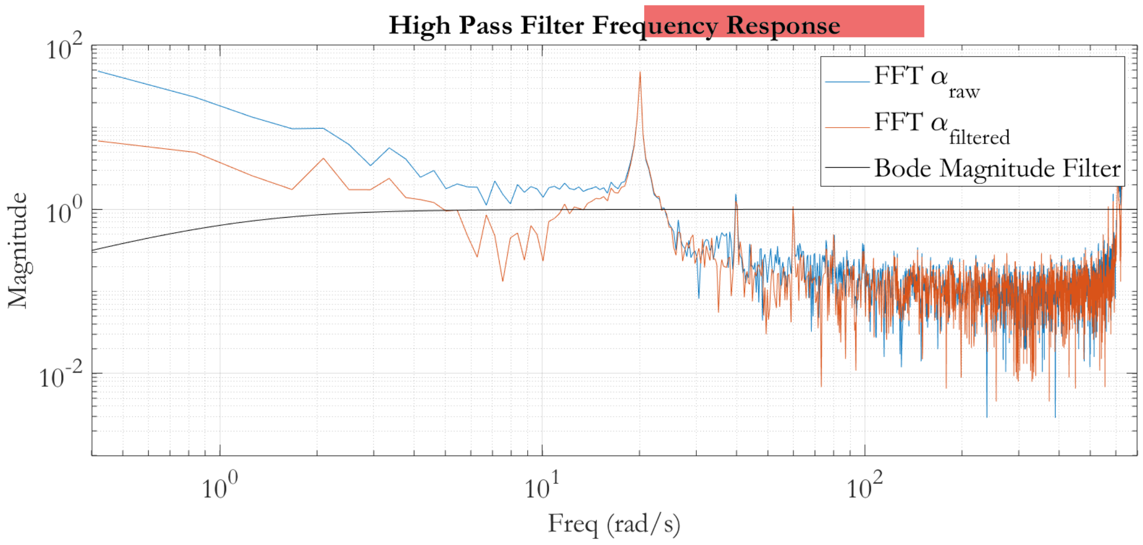

Figure 2.4: High Pass Filter Frequency Response

2.4. Prediction Error Methods

A PEM minimizes a cost function that uses a predictor. The predictor results in $\hat{y}_k$ which is based on u_k and y_k (see 2.2).

$$\hat{x}_{k+1} = \mathbf{A}\hat{x}_k + \mathbf{B}u_k + K(y_k - \mathbf{C}\hat{x}_k - \mathbf{C}u_k), \qquad \hat{y}_k = \mathbf{C}\hat{x}_k + \mathbf{D}u_k \tag{2.2}$$

In this predictor $\mathbf{A}(\theta)$, $\mathbf{B}(\theta)$, $\mathbf{C}(\theta)$, $\mathbf{D}(\theta)$, $\mathbf{K}(\theta)$ and $\hat{x}_0(\theta)$ are based on certain parameters θ. In the case of the pendulum, these parameters consist of certain lengths, masses, and damping coefficients. The goal is then to optimize θ such that $\hat{y}_k \approx y_k$. This is done with the optimization criterion $\hat{\theta}_N \in \mathbb{R}^p = argmin \frac{1}{N} \sum_{k=0}^{N-1} ||y_k - \hat{y}_k||_2^2$. In MATLAB, `idgrey()` was used to define the state space system in terms of the parameters. Next `greyest()` does the actual optimization. The search method was set to `'auto'`, this means that at each iteration several line search methods were used (Gaus-Newton, Levenberg-Maquardt, Adaptive Gaus-Newton, and Steepest Descent). The first direction resulting in a good reduction of the costs is used.

2.5. Identification for first arm

In this section, a prediction error method (PEM) is used to identify initial guesses for the parameters of the first arm of the pendulum (L_1, m_1, and c_1). This is still a preparation step for the final identification discussed in section 2.6. First very rough initial guesses are defined. $L_1 \approx 0.1\ m$ based ruler measurement. $m_1 \approx 0.3\ kg$ based on rough size and density guess of the arm ($length \times width \times depth \times density \approx 0.15 \times 0.03 \times 0.008 \times 7600 \approx 0.3\ kg$). Last $c_1 \approx 0.15\ Nms$ is based on an iterative trial and error process, based on several PEM's and the corresponding comparisons on the validation data.

For this identification step only the α measurements are used and the state space model is simplified. The `idgrey()` function is used to define our state space model. It uses a function which takes $\theta = [L_1\ m_1\ c_1]$ as inputs and has **A**, **B**, **C** and **D** as outputs. With $x = \begin{pmatrix} \alpha \\ \dot{\alpha} \end{pmatrix}$, $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ \frac{-g}{L_1} & \frac{c_1}{J_1} \end{pmatrix}$, $\mathbf{B} = \begin{pmatrix} 0 \\ \frac{1}{J_1} \end{pmatrix}$, $\mathbf{C} = \begin{pmatrix} 1 & 0 \end{pmatrix}$ and $\mathbf{D} = \begin{pmatrix} 0 \end{pmatrix}$. With $J_1 = m_1 L_1^2$. Next `greyest()` is used to apply the PEM (as explained in section 2.4).

The training was done on three experiments where the inputs were cosines with frequencies: 7 rad/s, 34 rad/s, and 70 rad/s.

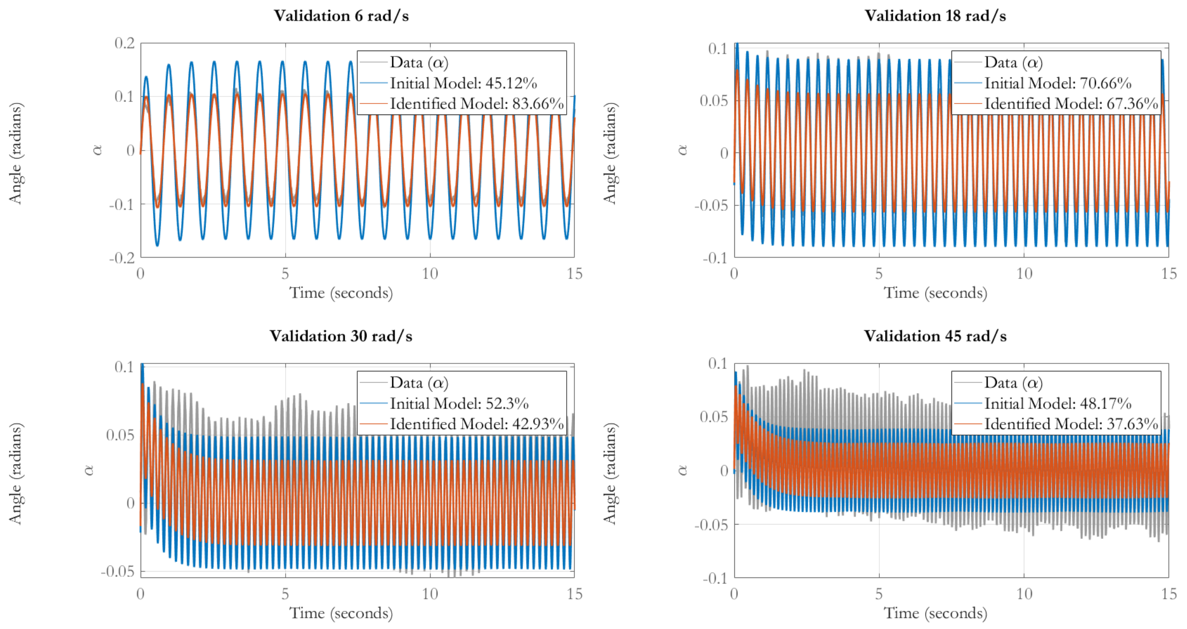

Figure 2.5: Validation for identification of the first pendulum arm

The validation is done with frequencies: 6 rad/s, 18 rad/s, 30 rad/s and 45 rad/s. Validation was done using `compare()`, this function plots simulated responses of specified models on top of input/output data. In the legend a percentage fit on the data is mentioned, this is the normalized root mean square (NRMSE). Figure 2.5 shows that the accuracy of the identification is high for lower frequencies and decreases for higher frequencies. The PEM identified the following values for the parameters: $L_1 = 0.1187\ m$, $m_1 = 0.2936\ kg$ and $c_1 = 0.2377\ Nms$. These are in a physically reasonable range and will be used as initial guesses in the final identification step to identify the coupling terms.

2.6. Identify coupling terms

To identify the coupling terms in $\mathbf{A}$ (see equation 1.8), and the final model in general, again a PEM is used (see section 2.4). The values for L_2 and $\frac{c_2}{m_2}$ identified in section 2.2 and the values for L_1, m_1 and c_1 identified in section 2.5 are used as the initial guesses for this last identification step. Only an initial guess for m_2 still needs to be defined. This is based on the rough shape and density guess of the second pendulum arm ($m_2 \approx \pi \times (radius_1^2 \times length + \frac{4}{3} radius_2^3) \times d \approx \pi(0.003^2 \times 0.1 + \frac{4}{3} 0.01^3) \times 7600 \approx 0.02$). This guess is also partly a result of some trial and error iterations of applying the PEM with slightly different initial guesses for the size and density for m_2.

For this identification step, both α and β measurements are trained on. Again `idgrey()` is used. This time it uses a function with $\theta = [L_1\ L_2\ m_1\ m_2\ c_1\ c_2]$ as inputs and the state space system matrices as given in equations 1.8 and 1.12 as outputs.

After this `greyest()` is used again to identify the parameters. Training was done on six experiments where the inputs were cosines with frequencies: 4 rad/s, 7 rad/s, 12 rad/s, 16 rad/s, 45 rad/s and 70 rad/s.

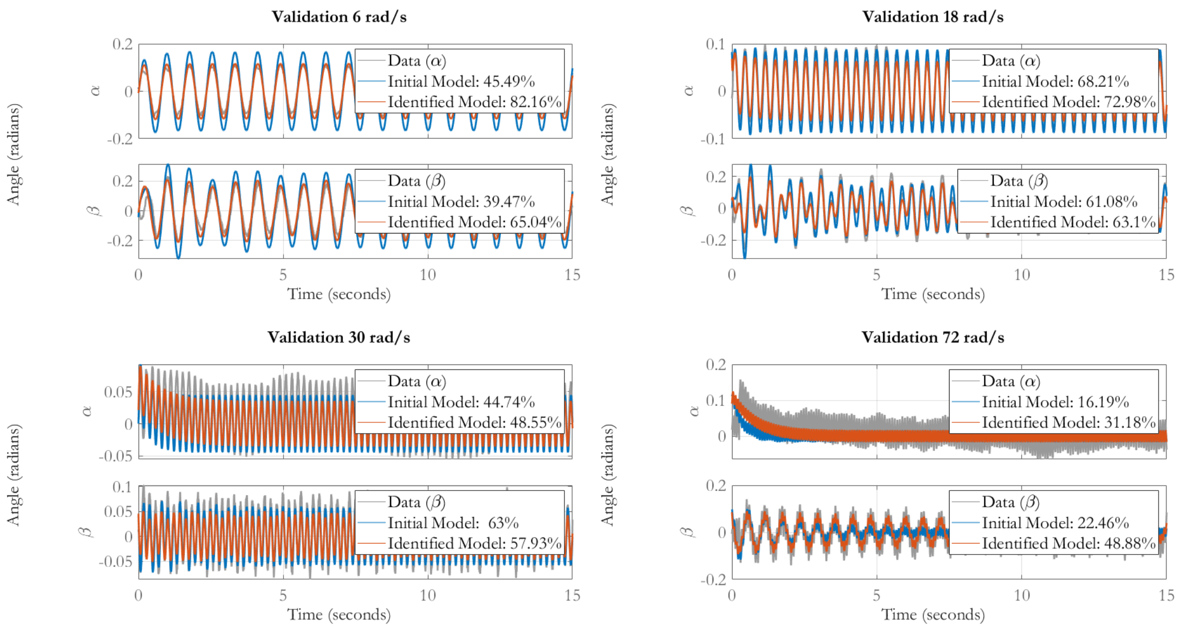

Figure 2.6: Validation for final identification

This time the validation is done with frequencies: 6 rad/s, 18 rad/s, 30 rad/s and 72 rad/s. Figure 2.6 shows that the accuracy on α is still high for low frequencies but low for high frequencies. The accuracy on β changes less for different actuation frequencies, and stays approximately between $50\% - 65\%$. This is not a very accurate fitting, but turned out good enough for the control. The PEM identified the following $\mathbf{A}_0$ and $\mathbf{B}_0$ matrix (see equation 2.3). These form the state space model in continuous time.

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -79.53 & 2.231 & -64.1 & 0.001807 \\ 87.29 & -110.1 & 70.36 & -0.08917 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 298.8 \\ -327.9 \end{bmatrix} \tag{2.3}$$

2.7. Discretized system

The above identified system is in continuous time. In order to implement them for the rotational pendulum setup which uses digital signals as input and measurements, a discretized system is required. The system is discretized using zero-order hold (ZOH) with a sampling time of $h = 0.01$ seconds.

$$\begin{cases} \mathbf{x}[k+1] & = \mathbf{\Pi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \\ \mathbf{y}[k] & = \mathbf{C}\mathbf{x}[k] + \mathbf{D}u[k] \end{cases}, \tag{2.4}$$

where, discrete state-space system matrix $\mathbf{\Pi} = e^{\mathbf{A}h}$ and input matrix $\mathbf{\Gamma} = \int_0^h e^{\mathbf{A}s} ds\mathbf{B}$.

In MATLAB, using `c2d()` function results in the below equation for system and input matrices. Note that $\mathbf{C}$ stays the same for discrete and continuous.

$$\mathbf{\Pi} = \begin{bmatrix} 0.9968 & 0.0001 & 0.0074 & 0.0000 \\ 0.0036 & 0.9945 & 0.0029 & 0.0100 \\ -0.5859 & 0.0164 & 0.5249 & 0.0001 \\ 0.6414 & -1.0923 & 0.5201 & 0.9936 \end{bmatrix} \qquad \mathbf{\Gamma} = \begin{bmatrix} 0.01217 \\ -0.01334 \\ 2.199 \\ -2.408 \end{bmatrix} \tag{2.5}$$

3
Estimator Design

The rotational wheel pendulum setup has only two measurable outputs α and β, whereas the system identified is 4th order. The controller designs discussed in the next sections require full-state feedback and therefore, it is essential to reconstruct the other two states $\dot{\alpha}$ and $\dot{\beta}$ from the input and measured outputs. In this section, this state reconstruction is achieved using a Luenberger observer.

3.1. Luenberger Observer

The linear system is observable if and only if its observability matrix $\mathbf{W}$ has full column rank. Since our identified system is a 4th order, the observability matrix for nth order system becomes

$$\mathbf{W} := \begin{bmatrix} \mathbf{C} \\ \mathbf{C\Pi} \\ \vdots \\ \mathbf{C\Pi}^{n-1} \end{bmatrix} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C\Pi} \\ \mathbf{C\Pi}^2 \\ \mathbf{C\Pi}^3 \end{bmatrix}.$$

Using MATLAB, $\texttt{rank(ctrb(}\mathbf{\Pi}^T\texttt{,}\mathbf{C}^T\texttt{)) = 4}$ which is full rank. Hence, $(\mathbf{\Pi}, \mathbf{C})$ is observable. Thus, it is possible to build an observer for the system. The approximated unmeasurable state in equation 2.4 can be derived using

$$\begin{aligned} \hat{\mathbf{x}}[k+1] &= \mathbf{\Pi}\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] \\ \hat{\mathbf{y}}[k] &= \mathbf{C}\hat{\mathbf{x}}[k] + \mathbf{D}u[k]. \end{aligned}$$

The system matrices $\mathbf{\Pi}, \mathbf{\Gamma}, \mathbf{C}, \mathbf{D}$ and input $u[k]$ are known, however, the initial conditions might differ. So, the idea behind using the Luenberger observer is to introduce feedback from the measured output $\mathbf{y}[k]$ to correct the difference between measured and estimated outputs $(\mathbf{y}[k] - \mathbf{y}[\hat{k}])$. This idea can be formulated as

$$\hat{\mathbf{x}}[k+1] = \mathbf{\Pi}\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] + L\left(\mathbf{y}[k] - \mathbf{C}\hat{\mathbf{x}}[k] - \mathbf{D}u[k]\right), \tag{3.1}$$

where $\mathbf{L}$ is the observer gain matrix. The above equation is the Luenberger observer. Using this observer, the error dynamics between the estimates and the states, i.e., $\mathbf{x}_e[k] = \hat{\mathbf{x}}[k] - \mathbf{x}[k]$ will be

$$\mathbf{x}_e[k+1] = (\mathbf{\Pi} - \mathbf{LC})\mathbf{x}_e[k]. \tag{3.2}$$

The $\mathbf{L}$ is determined using the observer poles such that the above error dynamics given by $(\mathbf{\Pi} - \mathbf{LC})$ is asymptotically stable: $\lim_{k\to\infty} \mathbf{x}_e[k] = 0$. Using thumb rules [1], it is convenient to keep observer poles two to ten times faster than control poles. So, taking into account the LQR controller poles discussed in the next sections which had the smallest pole at about 0.02 (for every equilibrium), the observer poles for this system are chosen about 10 times closer to the origin: [0.001, 0.002, 0.003, 0.004], also maintaining the algebraic multiplicity of one for each pole.

$$\mathbf{L} = \begin{bmatrix} 1.5148 & 0.0003 \\ 0.5231 & 1.9842 \\ 36.2474 & 0.0359 \\ 69.8398 & 97.4781 \end{bmatrix}$$

Using this gain, the estimated state trajectory is compared with the measurements. To obtain these results, manual disturbances were given to the setup with different initial conditions for the observer and the system to verify if the error dynamics are asymptotically stable.

From the above figure, it is illustrated in the first few timestamps how the estimated output overshoots from the measurements but later converges. However, there is a small problem encountered with this observer design. The

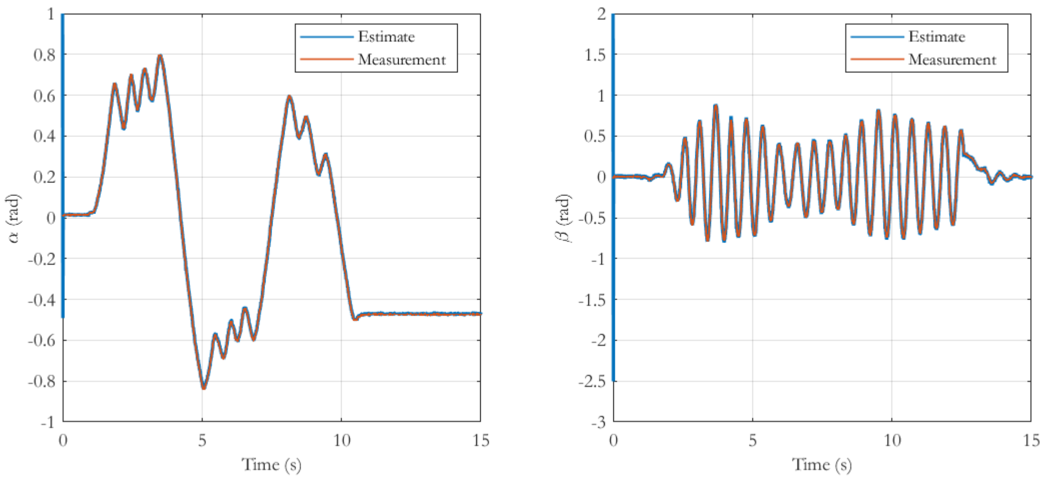

Figure 3.1: Luenberger observer estimates for α and β

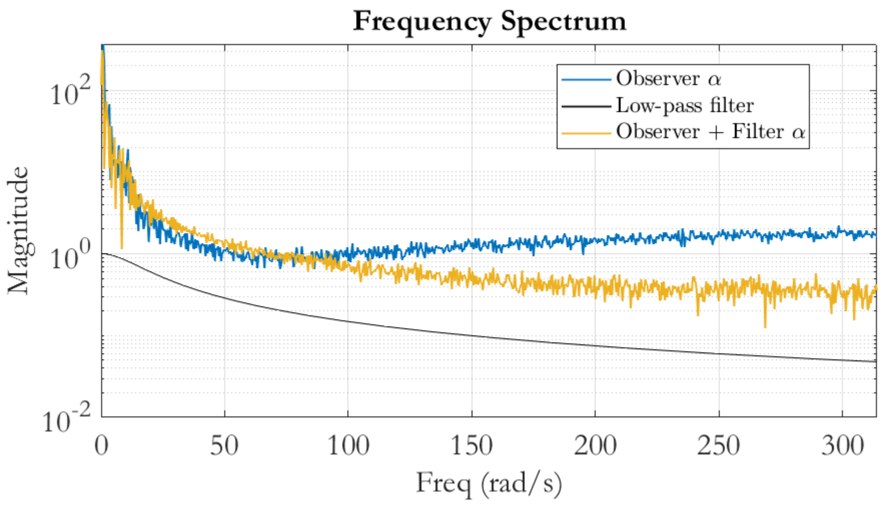

Figure 3.2: Frequency spectrum of observer estimate, low-pass filter and observer estimate with a filter

estimates are noisy. The frequency spectrum of the estimate is shown in figure 3.2 for α. This is not desirable for devising a controller because it can lead to a noisy control input which utilizes the estimated full-state feedback.

Attempts were made to change the poles of the observer, however, the noise persisted. As a consequence, a low-pass filter was implemented to the observer output with a cut-off frequency of 15 rad/s. The frequency spectrum of the filtered response is also shown in figure 3.2. Implementing this filtered observer estimate results in a smoother estimate as shown in figure 3.3, thus aiding a desirable control action. It is important to note that the low-pass filter induces delay in the system since it dampens the high frequency (fast) components of the signal. Therefore, tuning the cut-off frequency of the low-pass filter weighs the smoothness verses delay trade-off. In this work, the cut-off frequency was selected such that the added delay (of about 0.09 seconds) does not hamper the controller performance much but denoising the estimate.

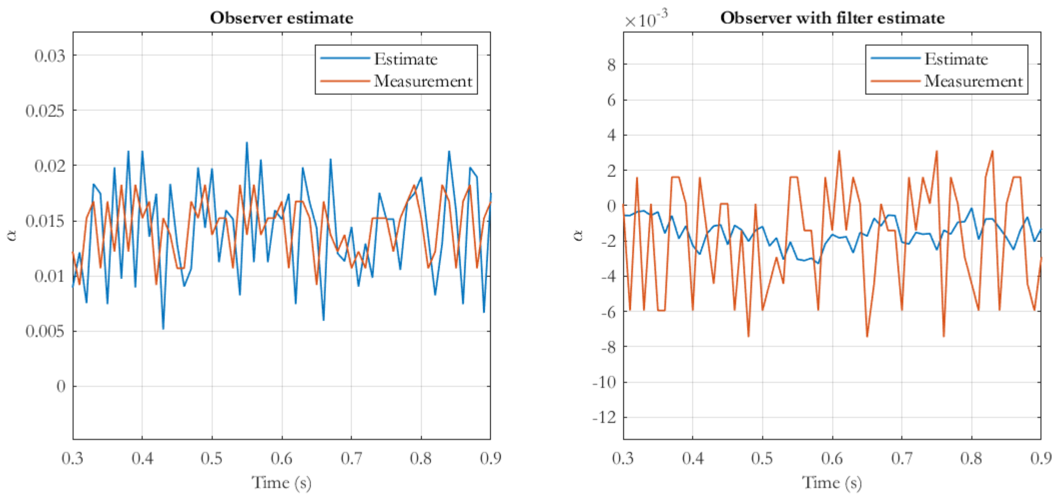

Figure 3.3: Comparison of observer estimate and observer with filter estimate

The full reconstruction of the states for the given observer with filter as the estimator is illustrated below in figure 3.4.

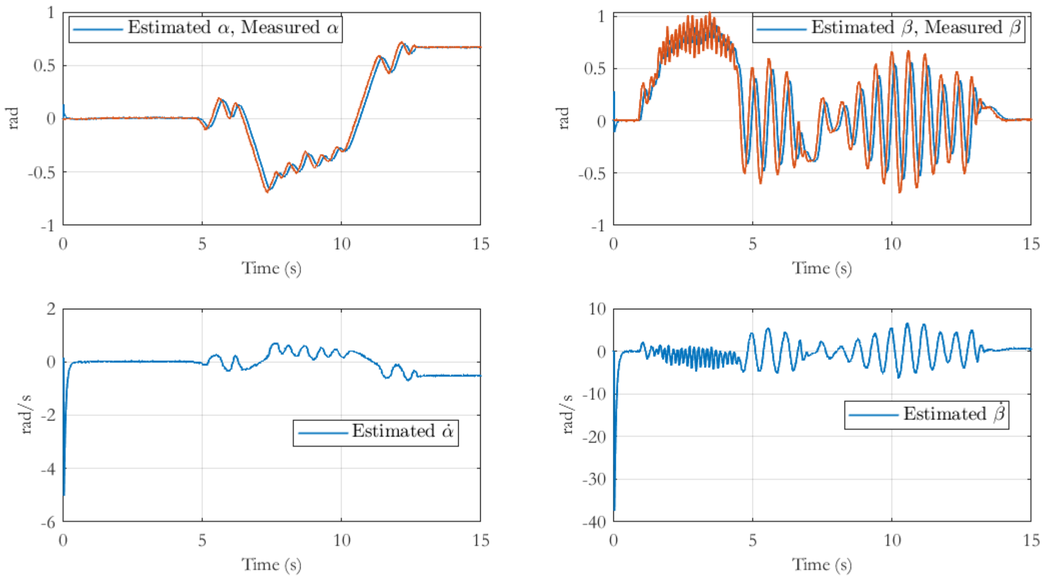

Figure 3.4: Full state reconstruction using Luenberger observer with a low-pass filter

This designed observer will be used to reconstruct all the states necessary for the upcoming discussion on controller design.

3.2. Kalman filter

The Kalman filter (KF) is a computation scheme to reconstruct the state of a given state-space model in a statistically optimal manner. Optimal in minimizing the variance of the state-reconstruction error conditioned on the acquired measurements. In the KF, the given system can be considered as,

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k, & w_k &\sim \mathcal{N}(0, Q) \\ y_{k+1} &= Cx_k + v_k, & v_k &\sim \mathcal{N}(0, R) \end{aligned} \tag{3.3}$$

where, Q and R are deterministic covariance matrices describing the process noise and measurement noise respectively.

Given the initial estimate $\hat{x}_0$, covariance P_1, A, B, C, Q and R at every time instance a filtered estimate $\hat{x}_{k|k}$ and uncertainty $P_{k|k}$ are computed by the KF. At each timestep, the KF consists of two update steps:

1. Measurement update: With Kalman gain $K_k = P_{k|k-1}C^T(CP_{k|k-1}C^T + R)^{-1}$,

$$\begin{aligned} \hat{x}_{k|k} &= \hat{x}_{k|k} + K_k(y_k - C\hat{x}_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - P_{k|k-1}C^T(CP_{k|k-1} + R)^{-1}CP_{k|k-1} \end{aligned} \tag{3.4}$$

2. Time update:

$$\begin{aligned} \hat{x}_{k+1|k} &= A\hat{x}_{k|k} + Bu_k \\ P_{k+1|k} &= AP_{k|k}A^T + Q \end{aligned} \tag{3.5}$$

3.2.1. Potential reasons for suboptimal performance of KF

For the proposed work, the implemented KF performed well on the stable equilibrium $\mathbf{x} = [0, 0, 0, 0]$ and $[\pi, 0, 0, 0]$, resulting a smooth estimate and without any delay which in this case is better than the observer with filter. However, it lead to divergence of the estimate trajectory for the other two unstable equilibrium. Upon careful observations, the following could be the possible reasons for this,

- Tuning the Q and R matrices: The R matrix was calculated using the correlation of the two output signals for non-zero inputs, so that it can capture the noise. The Q matrix was also tuned for a variety of cases. However, all the attempts were futile in stabilizing the unstable equilibrium. Possibly, more tuning needs to be performed to get good results.
- Violation of Gaussian Assumption: Kalman filter is optimal under the assumption that the noise are Gaussian and have a known covariance matrix. Violating this assumptions could degrade the performance of the Kalman filter. Whereas the Luenberger observer does not explicitly consider the noise characteristics.

For the above reasons, in the upcoming discussion Luenberger observer was used instead of the KF.

4
Controller Design

In this section, two different control methods were used for regulation and reference tracking of the pendulum. Historically [3], it is convenient to start with optimal control methods to get an intuitive perspective on the nominal performance of the system. This is because, if the nominal performance is empirically falsified, only then a more complex and computationally intensive method like H_∞ control, can be introduced for robustness requirements. Therefore, in this work, the linear quadratic regulator (LQR) and model predictive control (MPC) is implemented for the control tasks.

4.1. LQR Design

LQR works through the minimization of a cost function in which each state and each input is weighted. The cost function is shown in 4.1. These weights are determined by the designer, in this case they take the shape of equation 4.2 for 4th order system.

$$J = \sum_{k=1}^{\infty} (x[k]^T \mathbf{Q} x[k] + u[k]^T \mathbf{R} u[k]) \tag{4.1}$$

$$\mathbf{Q} = \begin{bmatrix} q_\alpha & 0 & 0 & 0 \\ 0 & q_\beta & 0 & 0 \\ 0 & 0 & q_{\dot\alpha} & 0 \\ 0 & 0 & 0 & q_{\dot\beta} \end{bmatrix} \qquad \mathbf{R} = \begin{bmatrix} r_u \end{bmatrix} \tag{4.2}$$

The minimization of cost function 4.1 is done by solving a discrete-time algebraic Riccati equation. To do this in MATLAB the `idare()` command was used. The final result of LQR is a controller gain K_{lqr}. This gain is used in a feedback loop such that $u = -K_{lqr}\hat{x}$. Figure 4.1 shows a simplified block diagram, to visualize the feedback loop, it is not the full Simulink model.

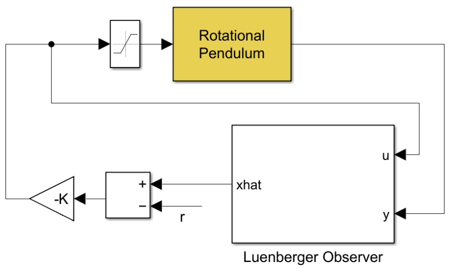

Figure 4.1: Simplified schematic of LQR combined with Luenberger Observer

The values for q_α, q_β, $q_{\dot\alpha}$, $q_{\dot\beta}$ and r_u in equation 4.2 needs to be tuned for the desired control behaviour. These values can differ between regulation and tracking or between control in different equilibrium points. In general q_β should be relatively bigger than the other values. This is because stabilizing an unstable equilibrium point in our system comes down to keeping $\beta \approx \pi$, so this objective should get the most weight. The weight on q_α is in general the second biggest because we also want α to return to its equilibrium position eventually. The exact weights for each control objective are tuned through an iterative trial and error process.

For regulation the reference signal $r[k] = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}^T$. For tracking a sine is generated for the reference on α ($r[k] = \begin{pmatrix} sin[k] & 0 & 0 & 0 \end{pmatrix}^T$). For regulation, the controller is pulling all states to zero. By subtracting r from $\hat{x}$, the controller pulls the states to r instead. This is because it sees the reference as a sort of error and it is now pulling $\hat{x} - r$ to zero instead.

4.2. MPC Design

One important drawback in the LQR design is that the control law did not take into account the physical limitations of the motor torque input for a safe operation. The input given to the system should be $u[k] \in [-1, 1]$. From figure 4.1, adding a saturation block is a non-optimal way. Therefore, a better way is to optimize the controller input is to handle the input constraint as well. MPC is a suitable choice for such task since it has the capability to handle the constraints and ensures an optimally safe system operation. Moreover, MPC is also capable to handle model uncertainty at every instant, which avoids causing error in forecasting. MPC is based on the receding horizon policy in which a model is exploited online (for every instant) to predict its likely future evolution and then a control sequence $(u[0], u[1], u[2], ...u[N])$ is computed for a finite horizon N, but only the first control input $u[0]$ from the sequence is used to take action.

In the given pendulum setup, the outputs α and β are only available from the measurement. Therefore, to devise an MPC controller using output feedback, the estimator designed in chapter 3 is used to reconstruct the states. The online MPC problem for N horizon can be formulated from [2] as,

$$\mathbb{P}_N(\hat{x}_0) : \begin{cases} \min\limits_{\mathbf{u_N}} & V_N(\hat{x}_0, \mathbf{u}_N) \\ \text{s.t.} & \mathbf{u_N} \in \mathcal{U}_N(x) \end{cases} \tag{4.3}$$

where $\hat{x}_0$ is the initial state estimate at every iteration given as a parameter to the online optimization. The input sequence $\mathbf{u}_N$ are the decision variables of the problem which satisfy $\mathcal{U}_N(x)$: the set of admissible control sequence for N steps. This set basically ensures that the input sequence lies in the input constraint set $\mathbb{U}$ and the states evolution resulting from those sequence also satisfy the state constraint set $\mathbb{X}$.

The objective function with stage cost $l(\cdot)$ and terminal cost $V_f(\cdot)$ can be defined as,

$$V_N(\hat{x}_0, \mathbf{u_N}) = \frac{1}{2}\sum_{k=0}^{N-1} l(\hat{x}[k], u[k]) + \frac{1}{2}V_f(\hat{x}[N]). \tag{4.4}$$

The stage cost $l(\cdot)$ penalizes the four states and the input. The terminal cost $V_f(\cdot)$ penalizes the states at the final instant of every horizon. For reference tracking, using output feedback, one is provided with $y_{ref}[k]$ as the reference signal. However, since the cost function takes into account the state estimates, these costs are

$$\begin{aligned} l(\hat{x}[k], u[k]) &= (\hat{x}[k] - x_{ref})^T Q(\hat{x}[k] - x_{ref}) \\ &\quad + (u[k] - u_{ref})^T R(u[k] - u_{ref}) \\ V_f(\hat{x}[N]) &= (\hat{x}[N] - x_{ref})^T P(\hat{x}[N] - x_{ref}). \end{aligned} \tag{4.5}$$

The $Q \in \mathbb{R}^{4\times4}$ and $R \in \mathbb{R}^{1\times1}$ are the weights to penalize the states and input respectively as discussed in the LQR design. The choice of the terminal cost utilizing the solution P of the discrete algebraic Riccati equation (DARE) will be discussed further in this section. Here, x_{ref} and u_{ref} are the target reference signal for states and inputs which are chosen apriori (of solving MPC) using an optimal target selection (OTS) problem [2] solved online (at each iteration). Further, the cost function 4.5 can be written as a quadratic programming problem. Consider the prediction matrices T and S for N horizon with $\mathbf{x}_{N+1} = (x(0), x(1), ..., x(N)) \in \mathbb{R}^{4(N+1)}$ and $\mathbf{u}_N = (u(0), u(1), ..., u(N-1)) \in \mathbb{R}^N$ which satisfy $\mathbf{x}_{N+1} = Tx_0 + S\mathbf{u}_N$. These prediction matrices can be given as,

$$T = \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \quad S = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ \vdots & \vdots & 0 & \dots & 0 \\ A^{N-1} & A^{N-2}B & A^{N-3}B & \dots & B \end{bmatrix}. \tag{4.6}$$

Using the prediction matrices, the cost function can be simplified to,

$$V_N(\mathbf{u}_N) = \frac{1}{2}\mathbf{u}_N^T H \mathbf{u}_N + h^T \mathbf{u}_N + c \tag{4.7}$$

$$\begin{aligned} H &= (S^T \bar{Q} S + \bar{R}) \\ h^T &= (x_0^T T^T \bar{Q} S - x_{ref}^T \bar{Q} S - u_{ref}^T \bar{R}) \\ c &= (x_0^T T^T \bar{Q} T x_0 - x_0^T T^T \bar{Q} x_{ref} - x_{ref}^T \bar{Q} T x_0 + x_{ref}^T \bar{Q} x_{ref} + u_{ref}^T \bar{R} u_{ref}) \end{aligned} \tag{4.8}$$

Thus, the objective function can be represented as a quadratic programming problem. Next, the OTS optimization problem is parameterized over y_{ref} for a given instant as defined.

$$(x_{ref}, u_{ref})(y_{ref}) \in \begin{cases} \text{argmin}_{x_r, u_r} J(x_r, u_r) \\ \text{s.t.} \begin{bmatrix} I - \Pi & -\Gamma_d \\ C & 0 \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} 0 \\ y_{ref} \end{bmatrix} \\ (x_r, u_r) \in \mathbb{X} \times \mathbb{U} \end{cases} \tag{4.9}$$

Note that for regulation task, this OTS problem can be solved offline and given into the MPC objective function in equation 4.8 directly.

So far, all the essential components of MPC control are defined. Now, they can be tailored for the pendulum setup. In this work, there is no state constraint set in the optimization problem. However, the asymptotic stability of the states are ensured using a soft-constraint in the form of terminal cost. The terminal cost in equation 4.5 is chosen to be the value function of infinite horizon LQ problem, utilizing the solution P of DARE. This is done for the following reasons,

- This choice is made so that the nominal stability can be inherited for some set of initial conditions to obtain the solution for infinite horizon (but now) constrainted optimal control problem.
- One more advantage is that if the control horizon N is large enough, $x(N)$ will be in the neighborhood of the origin (for $\hat{x}[k] - x_{ref}$) where the optimal control law using LQR can take over and stabilize the system while respecting the state and input constraints.
- The choice of P from DARE makes the terminal cost a candidate for control Lyapunov function which can be used to prove the asymptotic stability of the closed-loop dynamics.

The input signal to the pendulum setup is in the form of motor torque which has physical limitations. This input constraint set $\mathbb{U}$ can be given by the following inequality,

$$\begin{aligned} \begin{bmatrix} 1 \\ -1 \end{bmatrix} u[k] &\leq \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ \implies A_{in} u[k] &\leq b_{in} \end{aligned} \tag{4.10}$$

The stage cost and terminal cost are continuous and have value zero at the origin. Moreover, the input constraint set is closed and bounded ($\leq$ inequality and finite value). This observations allow for Proposition 2.4 to hold in [2] which verifies the existence of optimal solution for this problem. Therefore, from equation 4.8 and 4.10 the optimization problem is a quadratic programming problem with linear inequalities.

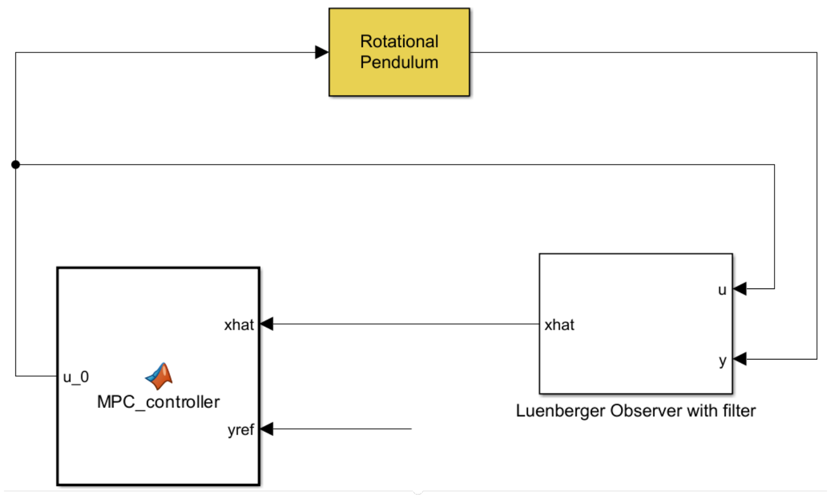

Figure 4.2: Simplified schematic of MPC framework for reference tracking

So, in MATLAB/Simulink a user-defined function block using `quadprog` function is implemented to solve $\mathbb{P}(x[k])$. The `active-set` algorithm is used by the `quadprog` to compute an optimal solution. In `active-set` at each iteration, solve the QP subproblem considering only the active constraints. This involves solving a smaller QP where the active constraints are treated as equalities. The same algorithm was also used for OTS problem.

The pseudo code for the MPC functional block is given below.

Algorithm 1 Output feedback MPC for Reference Tracking

```
Require: $\hat{x}[k], y_{ref}[k]$
1: $(x_{ref}[k], u_{ref}[k]) \leftarrow \text{OTS}(y_{ref}[k])$
2: $\mathbf{u}_N \leftarrow \mathbb{P}(\hat{x}[k])(x_{ref}[k], u_{ref}[k])$
3: $u_0 \leftarrow \mathbf{u}_N(0)$
```

5 Numerical results

The control is mainly focused around unstable equilibrium position $(\mathbf{x}^*, u^*) = ([0, \pi, 0, 0], 0)$ (see 1.10). In this chapter the results for the control methods around this equilibrium point are shown. Also control around the stable equilibrium point $(\mathbf{x}^*, u^*) = ([0, 0, 0, 0], 0)$ is mentioned, as this was used as an intermediate step towards controlling the unstable position. The plots will show the discrete derivative of the measurements α and β. These are only used to visualize the performance, and are not used by the observer or controller. Moreover, the below control methods were evaluated based on its ability to counteract any external disturbances and maintain the system at or return it to the equilibrium position. For this, two performance metrics were used: settling time and peak overshoot. To achieve an optimal nominal performance, the controller should be designed to minimize settling time while maintaining a low peak overshoot. Also, keeping in mind the typical oscillatory behavior of the observer estimates in the initial time-stamps, the controller operation was timed with a delay of 2 seconds until the estimation error dynamics converge. Therefore, in the numerical results there is no initial overshoot in the estimated state trajectory.

5.1. LQR regulation

In this and the following section, the LQ regulation and reference tracking is discussed. The above performance metrics will be used to evaluate the controller by giving an external (by human) disturbance to the second arm of the pendulum. Figures 5.1 and 5.2 show how these disturbances are rejected for two equilibrium points.

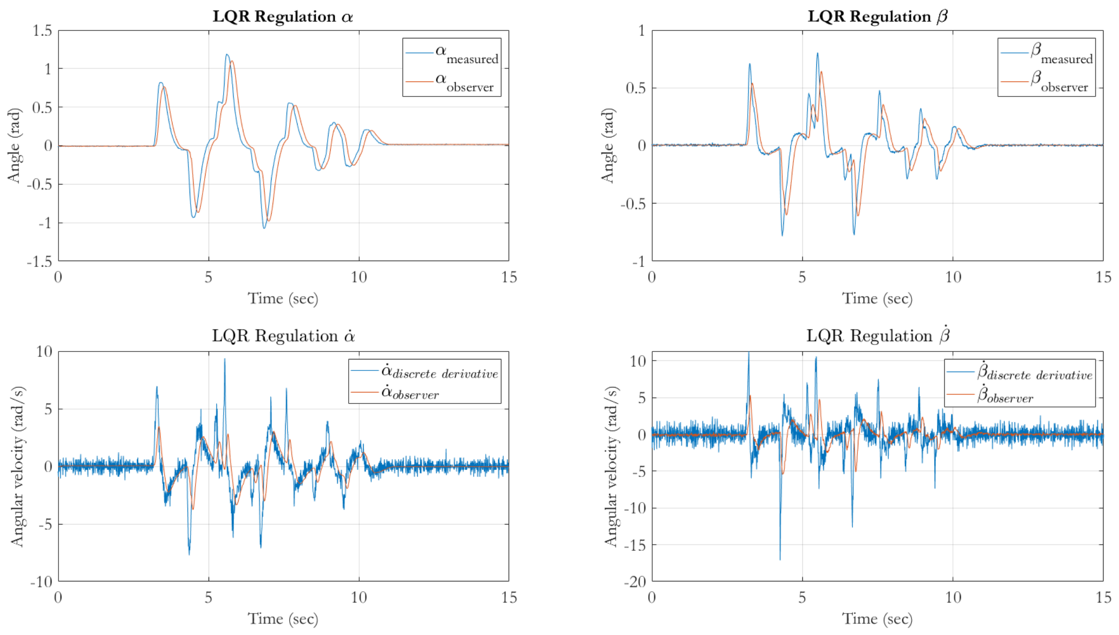

Figure 5.1: LQR Regulation $[0\ 0\ 0\ 0]^T$

For regulation in $\mathbf{x} = [0\ 0\ 0\ 0]^T$, LQR tuning ended up with: `Q = diag([80,100,2,2])` and `R=1`. This shows that stabilizing β is seen as most important and bringing α back to zero is also seen as quite important.

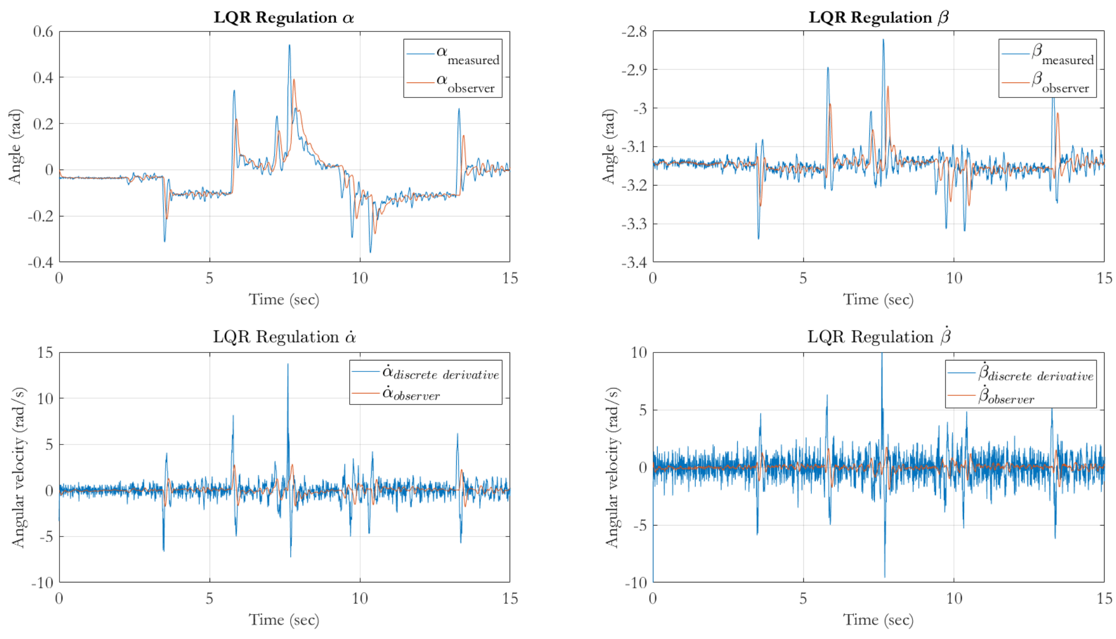

Figure 5.2: LQR Regulation $[0\ \pi\ 0\ 0]^T$

For regulation in $\mathbf{x} = [0\ \pi\ 0\ 0]^T$, LQR tuning ended up with: `Q = diag([80,100,3,7])` and `R=1`. Which is similar to the stable equilibrium only $\dot{\beta}$ is given slightly higher relative importance. This is done to react a bit more aggressively when β starts to destabilize.

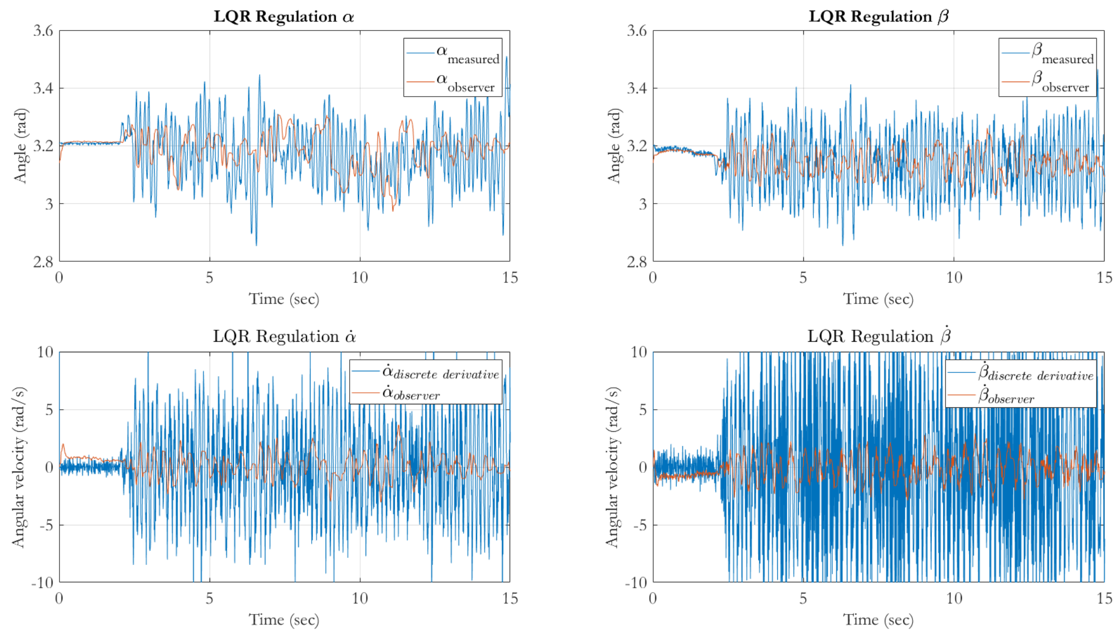

Figure 5.3: LQR Regulation $[\pi\ \pi\ 0\ 0]^T$

For regulation in $\mathbf{x} = [\pi\ \pi\ 0\ 0]^T$, LQR tuning ended up with: `Q = diag([30,100,20,1])` and `R=1`. Stabilizing β is the main objective so this state gets the most weight. We can see in 5.3 that the control action is quite noisy. Attempts were made to reduce this by increasing the weight on $\dot{\alpha}$ a bit, without too much success. We can see that the controller can stabilize the unstable equilibrium, but it creates quite some shacky movement.

5.2. LQR tracking

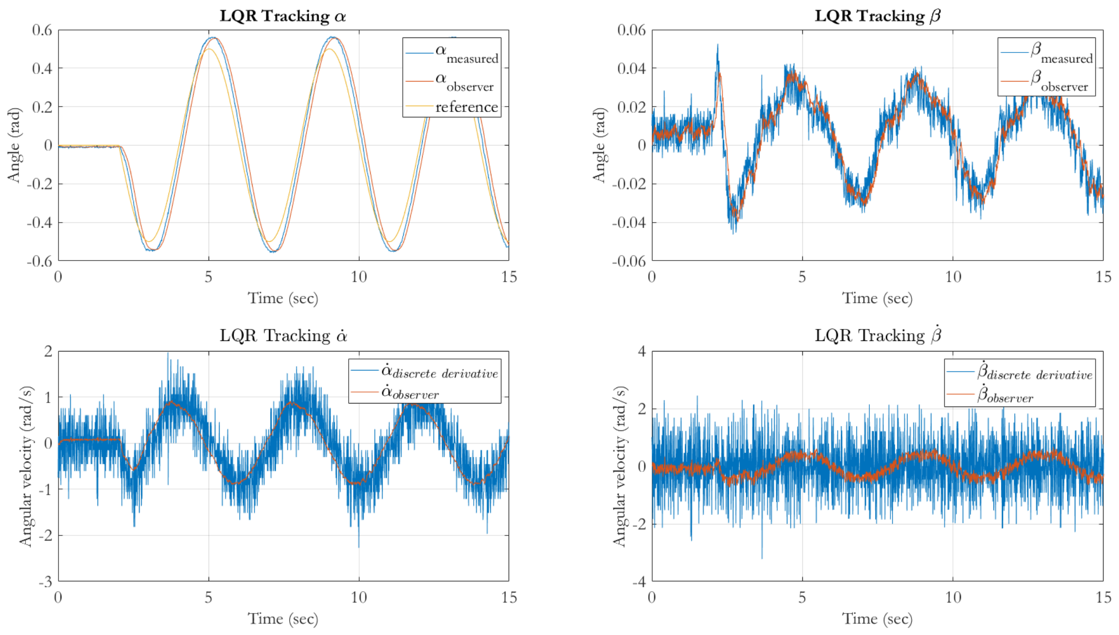

Figure 5.4: LQR Sine Tracking $[0\ 0\ 0\ 0]^T$

For tracking in $\mathbf{x} = [0\ 0\ 0\ 0]^T$, LQR tuning ended up with: `Q = diag([200,100,2,2])` and `R=1`. This time α was given more weight because tracking the reference is now seen as most important. This is possible without losing stability because for this equilibrium point β is already stable. Figure 5.4 shows that this causes β to also show some periodic behavior, but with quite a small amplitude. Which is accepted, to get better reference tracking on α.

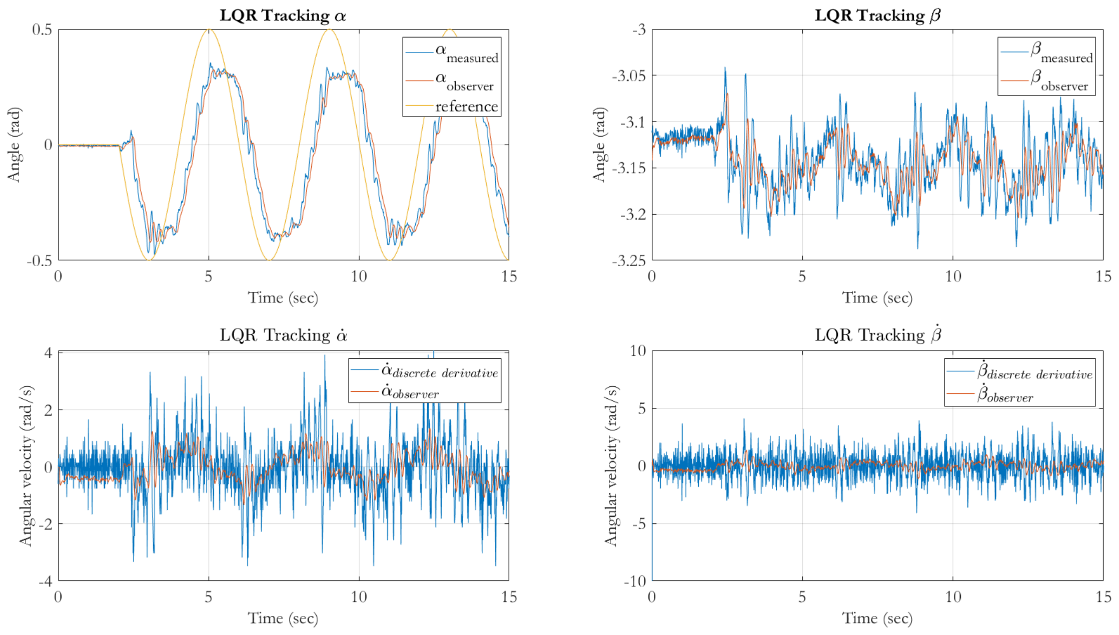

Figure 5.5: LQR Sine tracking $[0\ \pi\ 0\ 0]^T$

For tracking in $\mathbf{x} = [0\ \pi\ 0\ 0]^T$, LQR tuning ended up with: `Q = diag([140,100,2,2])` and `R=1`. Again α was given more than β because tracking is seen as the primary objective. In this equilibrium point, β is not stable without control, so there is more of a trade-off between tracking and stabilizing β. That is also why the weight on α is not as big as for LQR tracking in the stable equilibrium.

5.3. MPC regulation

The evaluation of MPC controller is also performed in a similar way as in the case of LQR with some human disturbances given to the system starting from the equilibrium position.

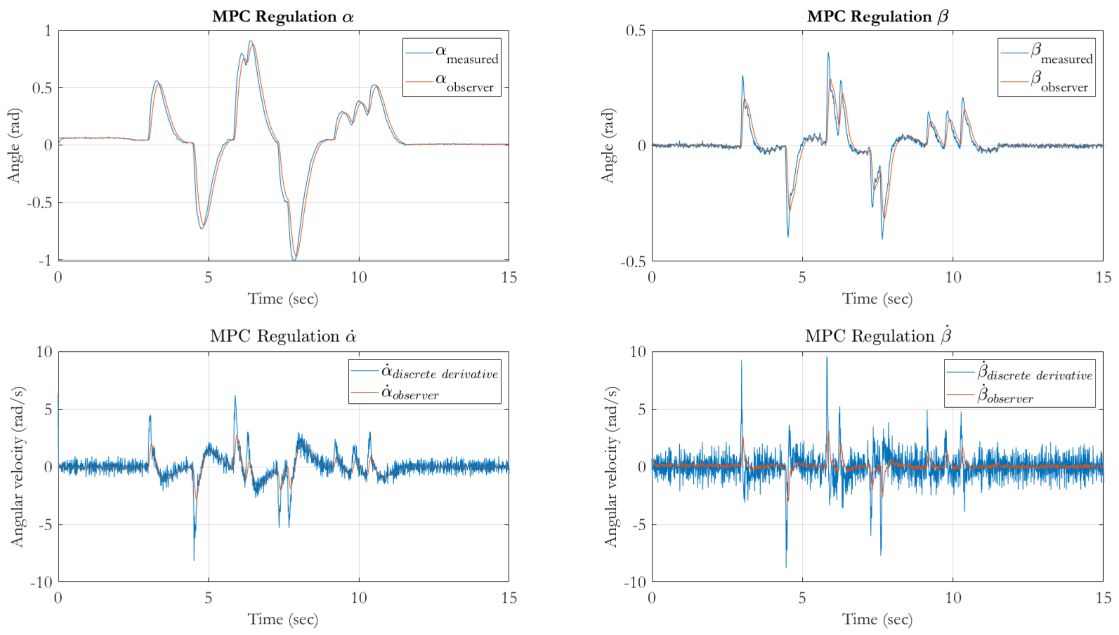

Figure 5.6: MPC Regulation $[0\ 0\ 0\ 0]^T$

For regulation in $\mathbf{x} = [0, 0, 0, 0]^T$, tuning the MPC ended up with: horizon of $N = 10$, `Q = diag([20,20,1,2])` and `R=1`. Penalizing the α and β equally and a higher penalty on $\dot{\beta}$ as compared to that of $\dot{\alpha}$ helped in obtaining an optimal nominal performance.

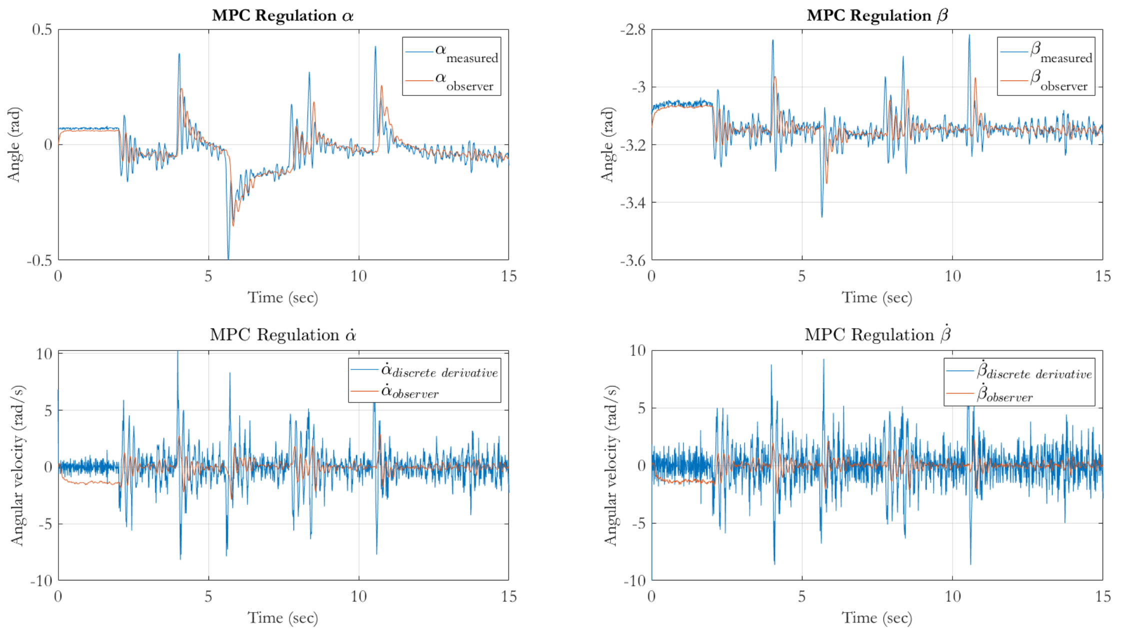

Figure 5.7: MPC Regulation $[0\ \pi\ 0\ 0]^T$

For regulation in $\mathbf{x} = [0, \pi, 0, 0]^T$, tuning the MPC ended up with: horizon of $N = 10$, `Q = diag([70,10,1,1])` and `R=1`.

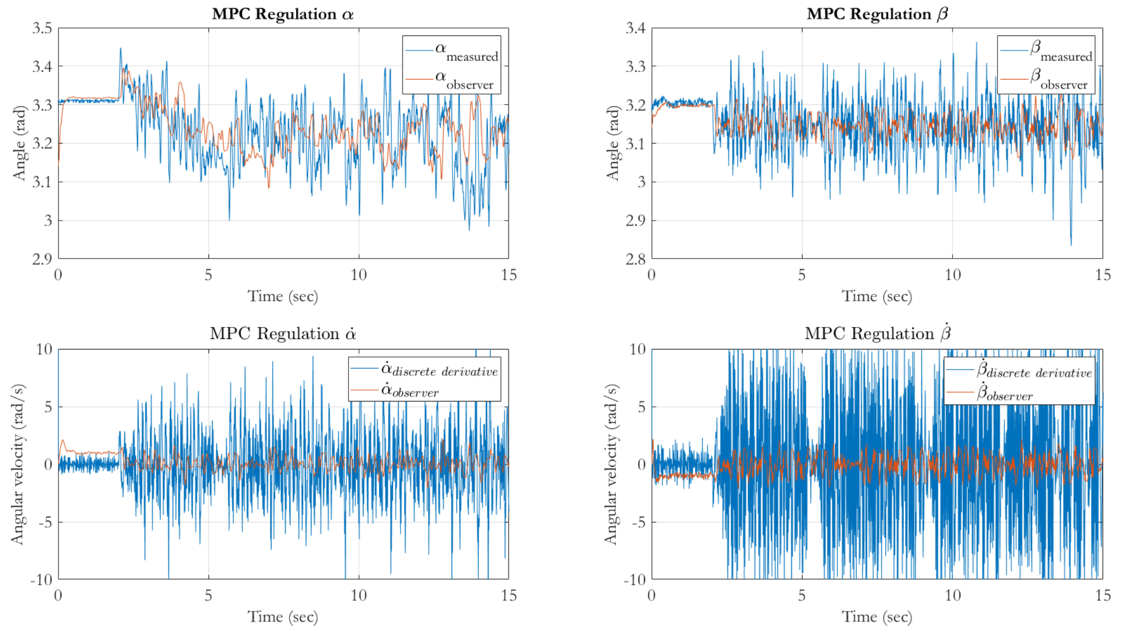

Figure 5.8: MPC Regulation $[\pi\ \pi\ 0\ 0]^T$

For regulation in $\mathbf{x} = [\pi, \pi, 0, 0]^T$, tuning the MPC ended up with: horizon of $N = 10$, `Q = diag([3,5,1,1])` and `R=1`. As seen in the LQR design, the control actions in figure 5.8 are also very high frequency, so attempts were made to penalize $\dot{\alpha}$, however, the performance got degraded further and so the aforementioned weights required no further tuning.

5.4. MPC tracking

For tracking using MPC controller, a discrete sine wave is applied as the reference signal. From figure 4.2 here, $y_{ref}[k] = [sin(\omega k) \quad 0]^T$.

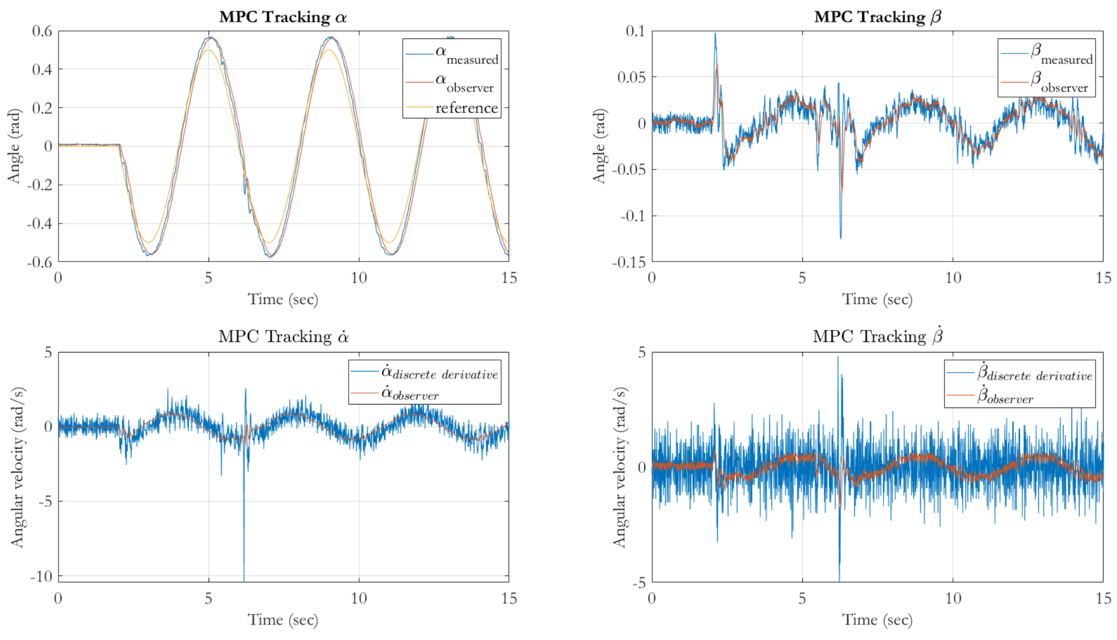

Figure 5.9: MPC Sine Tracking $[0\ 0\ 0\ 0]^T$

For tracking the y_{ref} at position $\mathbf{x} = [0\ 0\ 0\ 0]^T$: MPC with horizon $N = 10$ and `Q = diag([500 20 1 2])` and `R = 1` gives a good following. It should be noted that achieving optimal reference tracking for α necessitated the use of a significantly high gain. Despite this adjustment, the performance of β was not significantly affected.

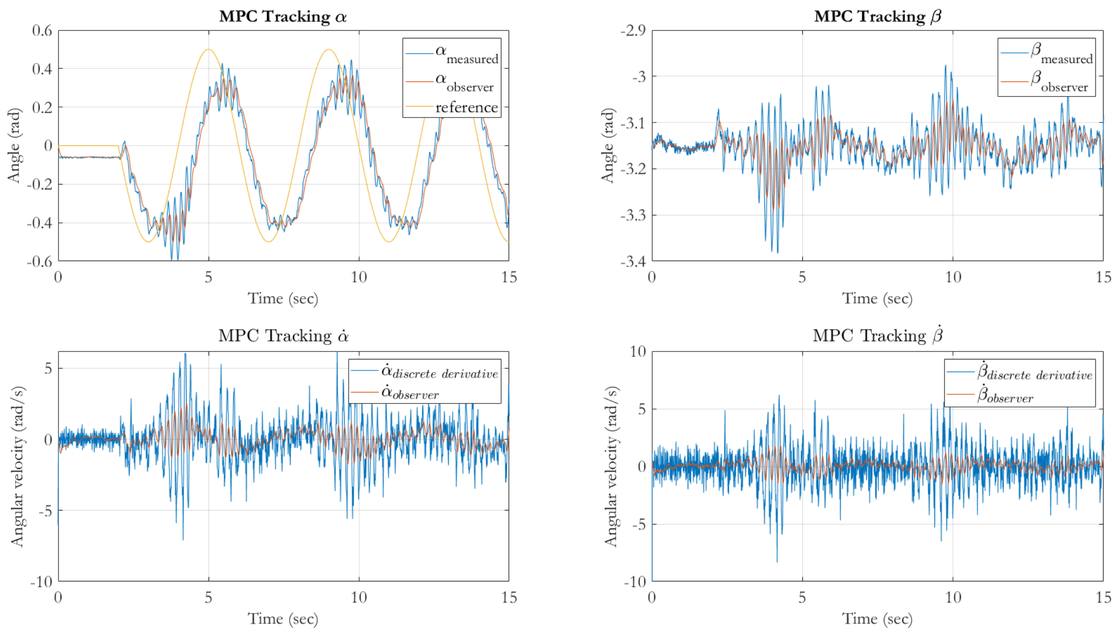

Figure 5.10: MPC Sine tracking $[0\ \pi\ 0\ 0]^T$

For tracking at position $\mathbf{x} = [0 \quad \pi \quad 0 \quad 0]^T$: tuning the MPC ended up with: horizon $N = 10$ and `Q = diag([70 10 1 1])` and `R = 1`.

5.5. Analysis of the MPC and LQR

- For both LQR and MPC for equilibrium position for figures 5.5 and 5.10, the controllers are quite worse in performance as compared to those of stable equilibrium. This could be because the system was identified at the stable equilibrium and was manipulated for unstable equilibrium using system's knowledge (its symmetrical configuration). However symmetry might not be fully valid, which could have degraded the model accuracy at the unstable equilibrium. Also, the state trajectory shows that there is a nonlinearity (as some delay) in the system for this equilibrium which is also not modelled properly.
- The worse response of unstable equilibrium can also be due to the fact that relatively more weights are needed to penalize changes in β. However, due to the coupling of the system, the problem could not be explicitly handled. There is a trade-off between tracking α and stabilizing β. Because in this equilibrium point, β isn't stable on its own, the trade-off needs to lean a bit more towards that goal compared to the stable case.
- For unstable equilibriums, it is seen that MPC performs worse than the LQR. This can be because of solver convergence and computational complexity. For low horizons, MPC might be giving a sub-optimal performance. Attempts were made to increase the horizon, however, this lead to computational delay leading to instability and further degraded performance.
- In all above cases, it is interesting to observe that the weights required for the nominal performance in MPC is not as high as that needed for LQR tasks. This shows the advantage of receding horizon policy: re-optimizing the system at every iteration for a finite horizon as compared to doing an offline optimization once and for all. The former requiring lower weights to maintain stability and performance since the weight tuning only affects finite horizon behavior.

6 Conclusion

The project achieved its control objectives using Linear Quadratic Regulator (LQR) and Model Predictive Control (MPC). While successful, there are areas for potential improvement in controlling the unstable equilibrium configurations. The analysis of MPC and LQR show some areas of such potential improvement. For future work, integrating Linear Quadratic Integral (LQI) control with a Kalman filter could enhance performance. Additionally, efforts should be made to increase robustness, possibly by addressing model inaccuracies and computational complexities observed in the current system. From the nonlinear dynamics, a possible direction is also in devising controller for swing-up of the pendulum setup.

References

[1] Karl Johan Astrom and Richard M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers.* USA: Princeton University Press, 2008. ISBN: 0691135762.

[2] J. Rawlings and D. Mayne. *Model Predictive Control: Theory and Design.* Nob Hill Publishing, 2008.

[3] Michael G Safonov. "Origins of robust control: Early history and future speculations". In: *Annual Reviews in Control* 36.2 (2012), pp. 173–181.

[4] Jayesh K Gupta et al. "A general framework for structured learning of mechanical systems". In: *arXiv preprint arXiv:1902.08705* (2019).