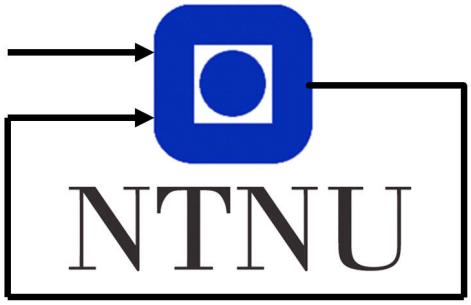
TTK4115 Helicopter labreport

Group 68
Sondre Bergum
Martin Madsen

October 24, 2017



Department of Engineering Cybernetics

Abstract

This report documents the work done in the course Linear System Theory at NTNU in the fall of 2017. We develop a mathematical model for our helicopter system, linearize it, and apply mono-variable control to it. However, the bulk of the report covers multi-variable control and state-estimation since that is the main focus of this course. The goal of the different control-methods are to provide as fast and accurate control over the helicopter as possible, as well as being a robust controller.

Contents

1 Motivation

The aim of this report is to investigate the dynamics of motion of a helicopter, then develop suitable models to control it, and furthermore, refine and implement them in MATLAB to get the best control that we can manage. The theoretical background for this analysis is the curriculum of the course "Linear system theory", TTK4115. The lab took place over 8 weeks in the fall of 2017 at NTNU.

The report, and assignments, are structured into multiple sections. Section 2 develops the dynamics of the helicopter by examining the torque at the joint angles, p, e, λ, and from that constructing differential equations relating the angles to the voltage applied to the helicopter rotors. From there we linearize our model and do a similarity transform to get a system of equations that we can apply our linear system theory too. In Section 3 we apply mono-variable control to control the pitch and travel of our helicopter. In Section 4 we examine the multi-variable system and apply optimal control. The last part of the assignment, Section 5, we introduce an observer to our system to suppress noise and perform state estimation. The last section, Section 6, is a final evaluation of our methods and results.

1.1 Lab setup and model of helicopter

A force diagram illustrating the forces acting on our helicopter is shown in Figure 2, which is available here [2]. The helicopter has two rotors providing thrust, that rotate around the pitch-axis, p. The weight of these rotors are then counteracted by a counterweight and can rotate around the elevation-axis, e. And this system can then rotate around the travel-axis, λ.

The helicopter is connected to a workstation PC so that it can be software controlled using MATLAB and Simulink. The joint angles are measured using a radial encoder at each joint, and the measurements are sent over to the workstation. A joystick is also connected to the workstation, and is used to set the reference points for our various controllers throughout this assignment.

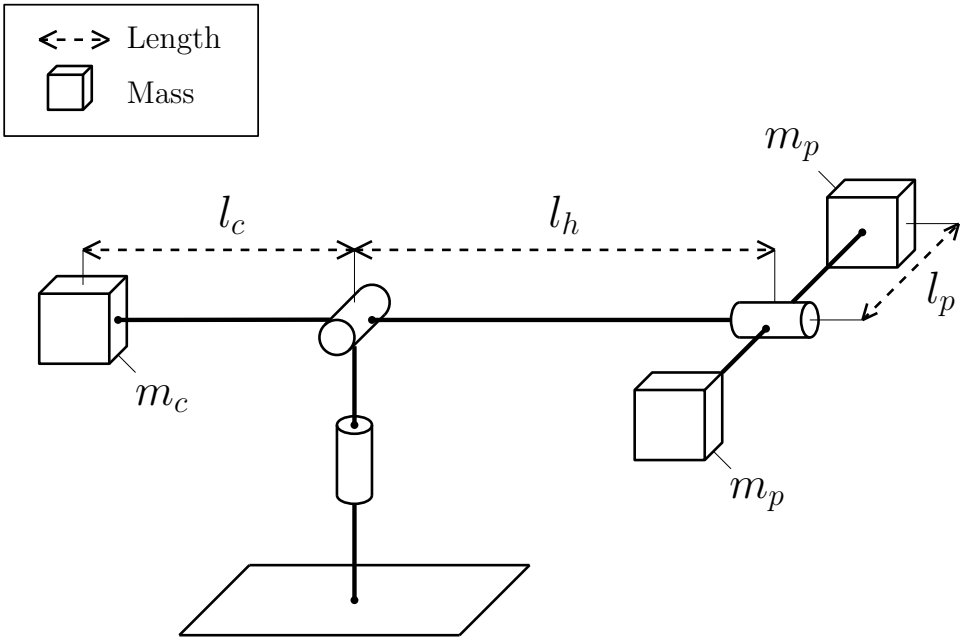


Figure 1: Masses and dimensions of our helicopter.

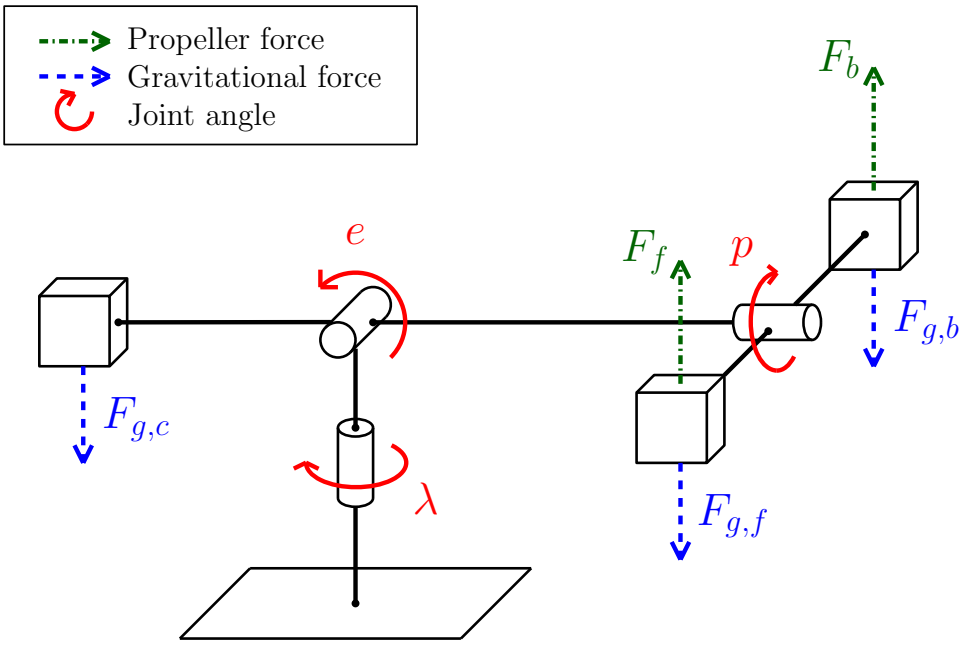


Figure 2: Force diagram representation of our helicopter.

2 Mathematical Modeling

In this section we will derive a mathematical model of our system which will serve as the foundation for the rest of the assignment.

The propeller forces for the front and back propellers are assumed to be in linear relation with the voltage inputs and are given by eq. (1):

$$F_f = K_f V_f \tag{1a}$$
$$F_b = K_f V_b \tag{1b}$$

where K_f is the motor constant and V_f and V_b are the input voltages for the front and back propellers, respectively.

2.1 System dynamics

All movement in the system will be part of a rotational displacement, so to analyze and derive a mathematical model we will be using the rotational version of Newton's 2. law shown in eq. (2):

$$I\alpha = \sum \tau \tag{2}$$

From this we analyze each joint's movement to derive equations of motion for pitch, elevation and travel, (p, e, λ).

Pitch:

$$\begin{aligned} J_p\ddot{p} &= \sum \tau_p && \text{(3a)}\\ &= F_f l_p - F_{g,f,tan} l_p - (F_b l_p - F_{g,b,tan} l_p) && \text{(3b)}\\ &= l_p(K_f V_f - m_p g\cos(p) - K_f V_b + m_p g\cos(p)) && \text{(3c)}\\ &= K_f l_p (V_f - V_b) && \text{(3d)} \end{aligned}$$

Elevation:

$$\begin{aligned} J_e\ddot{e} &= \sum \tau_e && \text{(4a)}\\ &= F_{g,c,tan} l_c + F_p l_h - F_{g,p,tan} l_h && \text{(4b)}\\ &= F_{g,c}\cos(e) l_c + (F_f + F_b)\cos(p) l_h - F_{g,p}\cos(e) l_h && \text{(4c)}\\ &= g(m_c l_c - 2m_p l_h)\cos(e) + K_f l_h V_s \cos(p) && \text{(4d)} \end{aligned}$$

Travel:

$$J_\lambda \ddot{\lambda} = \sum \tau_\lambda \tag{5a}$$
$$= -F_{p,tan} l_{h,\lambda} \tag{5b}$$
$$= -(F_f + F_b)\sin(p) l_h \cos(e) \tag{5c}$$
$$= -K_f l_h V_s \cos(e)\sin(p) \tag{5d}$$

From eq. (3)-(5) we arrive at the model:

$$J_p \ddot{p} = L_1 V_d \tag{6a}$$
$$J_e \ddot{e} = L_2 \cos(e) + L_3 V_s \cos(p) \tag{6b}$$
$$J_\lambda \ddot{\lambda} = L_4 V_s \cos(e)\sin(p) \tag{6c}$$

where V_s and V_d are given by eq. (7a) and eq. (7b) respectively and make out the input of the system:

$$V_s = V_f + V_b \tag{7a}$$
$$V_d = V_f - V_b \tag{7b}$$

and the constants L_i, $i = 1, 2, 3, 4$ are given by:

$$L_1 = K_f l_p \tag{8a}$$
$$L_2 = g(m_c l_c - 2 m_p l_h) \tag{8b}$$
$$L_3 = K_f l_h \tag{8c}$$
$$L_4 = -K_f l_h \tag{8d}$$

2.2 Linearization

Because the helicopter is a non-linear system we need to linearize the model first in order to design a linear controller. We linearize the equations of motion given by eq. (6) around the point of equilibrium:

$$\begin{bmatrix} p \\ e \\ \lambda \end{bmatrix} = \begin{bmatrix} p^* \\ e^* \\ \lambda^* \end{bmatrix} \quad \text{where} \quad \begin{bmatrix} p^* \\ e^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} \dot{p^*} \\ \dot{e^*} \\ \dot{\lambda^*} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{9}$$

for all time (this also implies $\ddot{p^*} = \ddot{e^*} = \ddot{\lambda^*} = 0$). To do this we need to find the values for eq. (10) such that the input $(V_s^*, V_d^*)^T$ keeps the helicopter in the equilibrium state.

$$\begin{bmatrix} V_s \\ V_d \end{bmatrix} = \begin{bmatrix} V_s^* \\ V_d^* \end{bmatrix} \tag{10}$$

Then we want to make a coordinate transformation on the system so the input $\mathbf{u} = (0, 0)^T$ when the system is stationary in the equilibrium point, keeps it in the equilibrium point. The transformation is the following:

$$\begin{bmatrix} \tilde{p} \\ \tilde{e} \\ \tilde{\lambda} \end{bmatrix} = \begin{bmatrix} p \\ e \\ \lambda \end{bmatrix} - \begin{bmatrix} p^* \\ e^* \\ \lambda^* \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \tilde{V_s} \\ \tilde{V_d} \end{bmatrix} = \begin{bmatrix} V_s \\ V_d \end{bmatrix} - \begin{bmatrix} V_s^* \\ V_d^* \end{bmatrix} \tag{11}$$

From Equation (6a):

$$\ddot{p} = \frac{1}{J_p} L_1 V_d \quad |_{\ddot{p^*}, V_d^*} \tag{12a}$$

$$0 = \frac{1}{J_p} L_1 V_d^* \implies \underline{V_d^* = 0} \tag{12b}$$

from Equation (6b):

$$\ddot{e} = \frac{1}{J_e}(L_2 \cos(e) + L_3 V_s \cos(p)) \quad |_{p^*, e^*, \ddot{e^*}, V_s^*} \tag{13a}$$

$$0 = \frac{1}{J_e}(L_2 + L_3 V_s^*) \tag{13b}$$

$$V_s^* = -\frac{L_2}{L_3} = \underline{g \frac{2m_p l_h - m_c l_c}{K_f l_h}} \tag{13c}$$

From eq. (9)-(13) follows:

$$\begin{bmatrix} \tilde{p} \\ \tilde{e} \\ \tilde{\lambda} \end{bmatrix} = \begin{bmatrix} p \\ e \\ \lambda \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \tilde{V_s} \\ \tilde{V_d} \end{bmatrix} = \begin{bmatrix} V_s - g\frac{2m_p l_h - m_c l_c}{K_f l_h} \\ V_d \end{bmatrix} \tag{14}$$

The inertia corresponding to the different joints are given as follows:

$$J_p = 2m_p l_p^2 \tag{15a}$$

$$J_e = m_c l_c^2 + 2m_p l_h^2 \tag{15b}$$

$$J_\lambda = m_c l_c^2 + 2m_p (l_h^2 + l_p^2) \tag{15c}$$

We then need to linearize the coordinate-transformed system to reach the desired result, and we start by substituting eq. (14) into eq. (6):

$$\ddot{\tilde{p}} = \frac{1}{J_p} L_1 \tilde{V_d} \tag{16a}$$

$$\ddot{\tilde{e}} = \frac{1}{J_e} L_2 \cos(\tilde{e}) + L_3 (\tilde{V_s} + V_s^*) \cos(\tilde{p}) \tag{16b}$$

$$\ddot{\tilde{\lambda}} = \frac{1}{J_\lambda} L_4 (\tilde{V_s} + V_s^*) \cos(\tilde{e}) \sin(\tilde{p}) \tag{16c}$$

These are second order differential equations, and to proceed with the linearization we need to change them into linked first order differential equations. This gives us a system on the form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{h}(\mathbf{x}, \mathbf{u}) \\ \mathbf{y} &= \mathbf{f}(\mathbf{x}, \mathbf{u}) \end{aligned} \tag{17}$$

We then use the functions $\mathbf{h}(\mathbf{x}, \mathbf{u})$ and $\mathbf{f}(\mathbf{x}, \mathbf{u})$ to change the system to the form of a state space model:

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

$$\begin{aligned} \text{where:} \quad \mathbf{A} = \frac{\delta \mathbf{h}(\mathbf{x}, \mathbf{u})}{\delta \mathbf{x}}|_{\mathbf{x}^*, \mathbf{u}^*}, \quad \mathbf{B} = \frac{\delta \mathbf{h}(\mathbf{x}, \mathbf{u})}{\delta \mathbf{u}}|_{\mathbf{x}^*, \mathbf{u}^*}, \\ \mathbf{C} = \frac{\delta \mathbf{f}(\mathbf{x}, \mathbf{u})}{\delta \mathbf{x}}|_{\mathbf{x}^*, \mathbf{u}^*}, \quad \mathbf{D} = \frac{\delta \mathbf{f}(\mathbf{x}, \mathbf{u})}{\delta \mathbf{u}}|_{\mathbf{x}^*, \mathbf{u}^*}, \end{aligned} \tag{18}$$

We are only interested in $\mathbf{A}$ and $\mathbf{B}$, because $\mathbf{y}$ in our system is already linear. Thus we get:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ K_3 & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 & K_1 \\ K_2 & 0 \\ 0 & 0 \end{bmatrix} \tag{19}$$

which makes our linearized system:

$$\begin{bmatrix} \ddot{\tilde{p}} \\ \ddot{\tilde{e}} \\ \ddot{\tilde{\lambda}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ K_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{p} \\ \tilde{e} \\ \tilde{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & K_1 \\ K_2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V}_s \\ \tilde{V}_d \end{bmatrix} = \begin{bmatrix} K_1 \tilde{V}_d \\ K_2 \tilde{V}_s \\ K_3 \tilde{p} \end{bmatrix} \tag{20}$$

or an alternative form to express it:

$$\ddot{\tilde{p}} = K_1 \tilde{V}_d \tag{21a}$$

$$\ddot{\tilde{e}} = K_2 \tilde{V}_s \tag{21b}$$

$$\ddot{\tilde{\lambda}} = K_3 \tilde{p} \tag{21c}$$

where the constants K_i, $i = 1, 2, 3$ are given by:

$$K_1 = \frac{L_1}{J_p} = \frac{K_f}{2 m_p l_p} \tag{22a}$$

$$K_2 = \frac{L_3}{J_e} = \frac{K_f l_h}{m_c l_c^2 + 2 m_p l_h^2} \tag{22b}$$

$$K_3 = \frac{L_4}{J_\lambda} V_s^* = g \frac{m_c l_c - 2 m_p l_h}{m_c l_c^2 + 2 m_p (l_h^2 + l_p^2)} \tag{22c}$$

2.3 Initial Flight of helicopter without controller

To grasp how it is to fly a helicopter without any controllers we connected the x-axis of our joystick to V_d and the y-axis to V_s. A gain of 1 for V_d and 10 for V_s was applied to convert the raw input from the joystick to a suitable voltage range for the helicopter. It was nearly impossible to fly the helicopter in any predictable manner without any controllers. This is because we go with linear input in a non-linear system.

2.4 Implementing offsets

To make sure the helicopters encoders were giving the output 0 for all 3 joint angles at the equilibrium point which we linearized about, we added offsets where necessary. For elevation we added a $-30°$ offset, while for pitch and travel we left the offset at 0. For pitch this would be fine as long as the helicopter was connected to the computer while resting on the table, while the initial state of travel is irrelevant.

We then calculated the motor constant, K_f, from measuring what input voltage for V_s (equal to the value of V_s^*) would keep the helicopter at a constant elevation in the equilibrium point. The input value we found to be the best fit was $V_s^* = 6.95$ V, thus from eq. (13c):

$$V_s^* = g\frac{2m_p l_h - m_c l_c}{K_f l_h} \tag{23a}$$

$$\implies \quad K_f = g\frac{2m_p l_h - m_c l_c}{V_s^* l_h} = 0.1437 \tag{23b}$$

This value for the motor constant will be used throughout the rest of the assignment.

2.5 Model limitations

Given the nature of linearization, we have a simplified, but imperfect model. The Discrepancies between our model and the helicopters behaviour are going to be small around the equilibrium we linearized about, but grow as the states move away from that point. This modelling error will have an effect on all other parts of this assignment, but in most cases the effect will be deemed acceptably small.

3 Mono-variable control

3.1 Mono-variable controllers for pitch angle and travel rate

In this section we connect the helicopter to an elevation controller in simulink that was handed out as part of the assignment. We then design our own controllers for pitch angle, $\tilde{p}$, and travel rate, $\dot{\tilde{\lambda}}$. The joystick's x-axis is then used to set the reference value for the travel rate.

3.1.1 PD-controller for pitch angle

The PD-controller, proportional- and derivative-controller, to control the pitch angle, p, is given by eq. (24):

$$\tilde{V_d} = K_{pp}(\tilde{p_c} - \tilde{p}) - K_{pd}\dot{\tilde{p}} \tag{24}$$

where $K_{pp}, K_{pd} > 0$ and $\tilde{p_c}$ is the desired reference for the pitch angle, p. Equation (24) is then substituted into eq. (21a) to get eq. (25):

$$\ddot{p} = K_1((K_{pp}(\tilde{p_c} - \tilde{p}) - K_{pd}\dot{\tilde{p}}) \tag{25}$$

We then take the Laplace transform of eq. (25) and find the transfer function from $\tilde{p}$ to $\tilde{p_c}$ and examine it to find reasonable values for K_pp and K_pd:

$$\frac{\tilde{p}(s)}{\tilde{p_c}(s)} = \frac{K_1K_{pp}}{s^2 + K_1K_{pd}s + K_1K_{pp}} \tag{26}$$

By using the principles developed for pole placement of transfer functions from Control Theory[3] we choose K_{pp} and K_{pd} so that we get a gain of one, $K_1K_{pp} = 1$, and poles in the right half-plane at $\lambda_1, \lambda_2 = -1$. This gives us a critical dampening ratio, ζ of 1, and the following theoretical values:

$$K_{pp} = \frac{1}{K_1} \qquad\qquad K_{pd} = \frac{2}{K_1} \tag{27}$$

3.1.2 P-controller for travel rate

We wish to control the travel rate of our helicopter, $\dot{\tilde{\lambda}}$, using a simple proportional controller given by eq. (28):

$$\tilde{p_c} = K_{rp}(\dot{\tilde{\lambda}_c} - \dot{\tilde{\lambda}}) \tag{28}$$

Equation (28) is then inserted into eq. (21c) to give eq. (29):

$$\ddot{\dot{\lambda}} = K_3 K_{rp}(\dot{\dot{\lambda}}_c - \dot{\dot{\lambda}}) \tag{29}$$

We, once again, take the Laplace transform of eq. (29) and rearrange to find the transfer function from $\dot{\dot{\lambda}}$ to $\dot{\dot{\lambda}}_c$. The resulting transfer function is given in eq. (30):

$$\begin{aligned} \frac{\dot{\dot{\lambda}}}{\dot{\dot{\lambda}}_c} &= \frac{K_3 K_{rp}}{s + K_3 K_{rp}} \\ &= \frac{\rho}{s+\rho}, \qquad \rho = K_3 K_{rp} \end{aligned} \tag{30}$$

Once more we use principles developed in Control Theory[3] and place the pole in the right half-plane, at $\lambda = -1$, giving us the following value for K_{rp}:

$$K_{rp} = \frac{1}{K_3} \tag{31}$$

3.2 Implementation of mono-variable controllers in Simulink

First, the pitch controller, eq. (24) was implemented in simulink, and the x-axis of the joystick used to set the reference for the pitch angle, p_c. The simulink diagram can be found in the appendix, fig. 11. The joystick output was also scaled by a suitable gain to get a reasonable range for our reference.

After that, we connect the travel rate controller, eq. (28), that gives us a desired value for the pitch angle reference, $\tilde{p_c}$, as the input to our pitch controller . The joystick now sets the reference for the travel rate, $\dot{\dot{\lambda}}_c$. The simulink diagrams for this controller is found in the appendix at fig. 12 and fig. 13.

3.3 Tuning of controllers

When trying out our controller based on the theoretical model we noticed a sluggish response, implying that our controlled, physical system was over-damped. This could be due to modeling errors and reducing the controller gains could improve this. The new final values for our controller gains we settled on is shown in eq. (32).

$$K_{pp} = \frac{1}{K_1} = 1.7534 \quad K_{pd} = \frac{0.9}{K_1} = 1.5781 \quad K_{rp} = \frac{0.5}{K_3} = -0.8173 \tag{32}$$

Reducing the values of K_{pd} and K_{rp} from the theoretical optimal values, like we do, pushes the eigenvalues of the system closer to the origo in the left half-plane.

3.4 Evaluation of controller design

The poles of our two transfer functions representing our controllers are chosen according to the idealized, linearized dynamics of our system. This means that even though our pole-placement is theoretically an optimal choice, the controllers will not behave as predicted. This is due to deviations from our theoretical model. Errors include modeling error such as inaccurate measurements of masses and lengths of the helicopter. The controllers also operate under the assumption that the dynamics of the system are linear, which they are not. This means that our controllers only function as expected in the linear region around the point of linearization.

Since the dynamics of the physical system are different than what our controllers are based on, one can expect better controller response with further manual tuning. However, we deemed our controllers fast and accurate enough around the linear region to move on to further assignments.

4 Multi-variable control

4.1 problem

This section constructs a state-space model for the dynamics of the helicopter. Then considers the controllability of the system before we develop a linear quadratic controller for the system using principles from optimal control[1].

4.1.1 State-space formulation of helicopter

The system of equations for pitch and elevation in eq. (21) can be written in the form of eq. (33), where $\mathbf{A}$ and $\mathbf{B}$ are matrices, and the state vector $\mathbf{x}$ and $\mathbf{u}$ are given by eq. (52):

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \quad \mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \tag{33}$$

$$\mathbf{x} = \begin{bmatrix} \tilde{p} \\ \dot{\tilde{p}} \\ \dot{\tilde{e}} \end{bmatrix} \quad \text{and} \quad \mathbf{u} = \begin{bmatrix} \tilde{V}_s \\ \tilde{V}_d \end{bmatrix} \tag{34}$$

$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are then given by eq. (35):

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{D} = 0 \tag{35}$$

We investigate the controllability of our system by checking the controllability matrix of our system[1] and examining if it has full row-rank:

$$\mathbb{C} = [\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A^2B}] = \begin{bmatrix} 0 & 0 & 0 & K_1 & 0 & 0 \\ 0 & K_1 & 0 & 0 & 0 & 0 \\ K_2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \Rightarrow \quad rank[\mathbb{C}] = 3 \tag{36}$$

Equation (36) shows that our system is in fact controllable and we can continue our quest for optimal control of the system.

4.1.2 Optimal control of helicopter

We aim to track a reference, $\mathbf{r}$, for the pitch angle, $\tilde{p}$, and elevation rate, $\dot{\tilde{e}}$ using a controller of the form given in eq. (37). The controller consists of two parts, first a reference feed-forward ($\mathbf{Pr}$), then a state feedback ($-\mathbf{Kx}$), this is called a linear quadratic regulator, LQR[1]. $\mathbf{K}$ is found by optimizing the cost function detailed in eq. (38).

$$\mathbf{u} = \mathbf{Pr} - \mathbf{Kx} \tag{37}$$

$$J = \int_0^\infty (\mathbf{x^T}(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u^T}(t)\mathbf{R}\mathbf{u}(t))dt \tag{38}$$

The weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ are diagonal matrices that determine the relative input- and output energy of the system. A further explanation can be found in [1]. To extract $\mathbf{K}$ from eq. (38), we use the MATLAB command `K=lqr(A,B,Q,R)`.

The matrix $\mathbf{P}$ is chosen so that eq. (39) holds for fixed values of $\dot{\tilde{p}}_c$ and $\dot{\tilde{e}}_c$, and it can be shown that $\mathbf{P}$ can be found by solving the expression in eq. (40)[1].

$$\lim_{t\to\infty} \tilde{p}(t) = \tilde{p}_c \quad \text{and} \quad \lim_{t\to\infty} \dot{\tilde{e}}(t) \quad = \dot{\tilde{e}}_c \tag{39}$$

$$\mathbf{P} = [\mathbf{C}(\mathbf{BK} - \mathbf{A})^{-\mathbf{1}}\mathbf{B}]^{-1} \tag{40}$$

The closed loop of our system, eq. (35), can be expressed as in eq. (41):

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{BK})\mathbf{x} + \mathbf{BPr} \tag{41}$$

4.1.3 Adding integral effect to system

The system is then modified to include an integral effect for the elevation rate, $\dot{\tilde{e}}$, and pitch angle, $\tilde{p}$. This results in two additional states, γ and ζ, that can be expressed as eq. (42). The new state- and input-vector are now given by eq. (43)

$$\begin{aligned} \dot{\gamma} &= \tilde{p} - \tilde{p}_c \\ \dot{\zeta} &= \dot{\tilde{e}} - \dot{\tilde{e}}_c \end{aligned} \tag{42}$$

$$\mathbf{x}_a = \begin{bmatrix} \tilde{p} \\ \dot{\tilde{p}} \\ \dot{\tilde{e}} \\ \gamma \\ \zeta \end{bmatrix} \quad \text{and} \quad \mathbf{u} \quad = \begin{bmatrix} \tilde{V}_s \\ \tilde{V}_d \end{bmatrix} \tag{43}$$

The new, augmented system have more states than our original system and its state-space equation has changed. The new states are denoted $\mathbf{x_a}$ and the system is described by $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$. Equation (44) represents the new system.

$$\dot{\mathbf{x}}_a = \begin{bmatrix} \dot{\tilde{p}} \\ \ddot{\tilde{p}} \\ \dot{\tilde{e}} \\ \dot{\gamma} \\ \dot{\zeta} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \mathbf{x_a} + \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u} - \begin{bmatrix} \mathbf{0} \\ \mathbb{I} \end{bmatrix} \mathbf{r} = \bar{\mathbf{A}}\mathbf{x_a} + \bar{\mathbf{B}}\mathbf{u} - \begin{bmatrix} \mathbf{0} \\ \mathbb{I} \end{bmatrix} \mathbf{r} \tag{44a}$$

$$\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x_a} = \bar{\mathbf{C}}\mathbf{x_a} \tag{44b}$$

To this augmented system in eq. (44) we wish once more to apply a controller of the form of eq. (45). The closed loop of our augmented system is given by eq. (46).

Notice that $\bar{\mathbf{P}}$ in eq. (46) consists of the same $\mathbf{P}$ as in eq. (41) adjoined with the identity matrix. This is because the reference feed-forward of our controller, eq. (45), no longer serves the purpose of removing the error when time approaches infinity, as it was originally designed for in eq. (40). In fact, trying to calculate a new $\bar{\mathbf{P}}$ for the augmented system using eq. (40) will lead to the null-matrix as the purpose of the integral effect is to remove stationary errors. The $\mathbf{P}$-matrix from the unaugmented system is included in our controller to provide faster and more accurate tracking.

$$\mathbf{u} = -\bar{\mathbf{K}}\mathbf{x_a} + \bar{\mathbf{P}}\mathbf{r} \tag{45}$$

$$\dot{\mathbf{x_a}} = (\bar{\mathbf{A}} - \bar{\mathbf{B}}\bar{\mathbf{K}})\mathbf{x} + \bar{\mathbf{B}}\bar{\mathbf{P}}\mathbf{r} \quad \bar{\mathbf{P}} = \begin{bmatrix} \mathbf{P} \\ -\mathbb{I} \end{bmatrix} \tag{46}$$

4.2 MATLAB implementaion of LQR and augmented system

As previously explained, the matrix $\mathbf{K}$ and $\bar{\mathbf{K}}$ are found using the MATLAB-command `K=lqr(A,B,Q,R)`. For this to work our systems, eq. (35) and eq. (44), needs to be implemented in code. The code needed to implement this can be found in fig. 6 and fig. 7. The simulink diagrams found in fig. 14 - 16.

4.3 Evaluation of LQR controllers

4.3.1 Tuned values for LQR without integral effect

After a couple hours of tuning these were the values we settled on for the weighting matrices, $\mathbf{Q}$ and $\mathbf{R}$, stated in eq. (47). Each entry on the main diagonal of $\mathbf{Q}$ corresponds to the relative weight of output energy of each state, while each entry of the main diagonal of $\mathbf{R}$ corresponds to the weight of input energy for each input, $\tilde{V_s}$ and $\tilde{V_d}$. Notice that elevation rate has a

much higher weight than pitch and pitch rate, and pitch rate has the lowest weight of all. This weighting schemes represents the importance we put on each state, with the highest weight being the one we wish to have the most accurate response. The same argument can be made for $\mathbf{R}$ where $\tilde{V}_d$ is the most important input.

$$\mathbf{Q} = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 200 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} 5 & 0 \\ 0 & 10 \end{bmatrix} \tag{47}$$

Using the MATLAB command detailed earlier gives us the following $\mathbf{K}$ and $\mathbf{P}$ that are used in our controller:

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 6.3246 \\ 2.2361 & 2.9735 & 0 \end{bmatrix}, \mathbf{P} = \begin{bmatrix} 0 & 6.3246 \\ 2.2361 & 0 \end{bmatrix} \tag{48}$$

4.3.2 Tuned values for LQR with integral effect

The LQR with integral effect follows the same tuning scheme as the LQR without integral effect. Here the weight of the pitch is increased slightly to give it a better response. Also two more states are included in $\mathbf{Q}$, γ and ζ, corresponding to the integral of pitch and elevation respectively. Final values for the augmented weighting matrices is given by eq. (49):

$$\bar{\mathbf{Q}} = \begin{bmatrix} 60 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 200 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 30 \end{bmatrix}, \bar{\mathbf{R}} = \begin{bmatrix} 5 & 0 \\ 0 & 10 \end{bmatrix} \tag{49}$$

Giving the following augmented $\bar{\mathbf{K}}$ and $\bar{\mathbf{P}}$:

$$\bar{\mathbf{K}} = \begin{bmatrix} 0 & 0 & 9.6633 & 0 & 2.4495 \\ 3.6668 & 3.7228 & 0 & 1 & 0 \end{bmatrix}, \bar{\mathbf{P}} = \begin{bmatrix} 0 & 6.3246 \\ 2.2361 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \tag{50}$$

Notice that $\bar{\mathbf{P}}$ is determined by eq. (46) and not by the expression in eq. (40).

4.3.3 Comparison of optimal control with and without integral effect

The response of elevation without and with integral effect can be found in fig. 19 and fig. 21 in appendix D, respectively. The pitch response without and with integral effect can be found in fig. 20 and fig. 22 in appendix D, respectively. When comparing responses we used step functions as the reference input so the situations would be as equal as possible.

We see quite a big difference regarding the elevation response as when the elevation rate reference is set back to 0 after being raised, the elevation is actually kept constant shortly after, with integral effect. Without the integral effect the controller is unable to keep the travel rate at 0 until the elevation reenters the equilibrium point. This is due to the integral effect cumulatively integrating the difference. For the pitch we see very little difference with and without the integral effect.

5 State estimation

In this part of the assignment we will only use the measurements of $\tilde{p}, \tilde{e}, \tilde{\lambda}$, and estimate the corresponding angular velocity with an observer rather than using numerical derivation. So we're going to derive a state-space model of the system in order to implement this observer.

5.1 State-space formulation

From eq. (20) we're deriving a state-space model on the form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} \end{aligned} \tag{51}$$

where $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are matrices and the state vector, the input vector and the output vectors are given by eq. (52) and the system matrices by eq. (53):

$$\mathbf{x} = \begin{bmatrix} \tilde{p} \\ \dot{\tilde{p}} \\ \tilde{e} \\ \dot{\tilde{e}} \\ \tilde{\lambda} \\ \dot{\tilde{\lambda}} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \tilde{V}_s \\ \tilde{V}_d \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} \tilde{p} \\ \tilde{e} \\ \tilde{\lambda} \end{bmatrix} \tag{52}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ K_3 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{53}$$

5.2 Observability and observer

To create an observer for our system we first need to determine if our system is observable, an observability test using the principle of duality is described in [1]. The principle of duality states that the system is observable if the controllability-matrix of the transposed system has full rank. Since the dimensions of our observer is so large we instead opt to use the MATLAB command `obsv(A,C)` nested inside `rank(MATRIX)` to find the rank of our observability matrix, $\mathbb{O}$, instead of computing it by hand.

The conclusion is that the rank of $\mathbb{O}$ match the dimensions of A, and hence our system is controllable.

We then make the observer on the form of eq. (54). Here $\mathbf{L}$ is the Luenberger obeserver gain[1]. Tuning our observer is a matter of placing poles of $\mathbf{L}$ in eq. (54). The easiest way to do this is to use the MATLAB command `place(A',C',pol)`, where `pol` is a vector consisting of the desired poles. Note that the $\mathbf{A}$ and $\mathbf{C}$ matrices must be transposed in order for the MATLAB command to work, luckily for us the poles of a system is unchanged when the system is transposed[3].

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{L}(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}}) \tag{54}$$

5.3 Reducing measurements of our observer

We wish to see if we can reduce the dimensions of the measurement vector, $\mathbf{y}$, by omitting either $\tilde{p}$ or $\tilde{\lambda}$. We use our same procedure for determining the observability of our system, first without $\tilde{p}$, then without $\tilde{\lambda}$, in $\mathbf{y}$. This change will cause the $\mathbf{C}$-matrix of our system to change and hence the observability matrix will change. The new $\mathbf{C}$-matrix corresponding to the omission of $\tilde{p}$ from the measurements can be seen in eq. (55a), while the $\mathbf{C}$-matrix corresponding to omission of $\tilde{\lambda}$ is shown eq. (55b). Equation (56a) shows that removing $\tilde{p}$ does not make our system unobservable, while removing $\tilde{\lambda}$ will result in an unobservable system as seen in eq. (56b).

$$\mathbf{C}_{without\tilde{p}} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{55a}$$

$$\mathbf{C}_{without\tilde{\lambda}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{55b}$$

$$rank(\mathbb{O}_{without\tilde{p}}) = 6 = \dim(A) \tag{56a}$$

$$rank(\mathbb{O}_{without\tilde{\lambda}}) = 4 \neq \dim(A) \tag{56b}$$

Since we now know that the removal of $\tilde{\lambda}$ makes our system unobservable we instead move onward in constructing our minimal observer by removing $\tilde{p}$ from the measurement vector, $\mathbf{y}$. The new measurement vector is given by eq. (57). The same procedure for placing poles of our first observer, described in the prevous section, can be applied to place poles of our minimal observer as they have the same observer-form, eq. (54).

$$\mathbf{y} = \begin{bmatrix} \tilde{e} \\ \tilde{\lambda} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x} \tag{57}$$

5.4 Pole-placement of observers

Placing the poles for the observer taking all three states was quite a trivial task. As mentioned in the previous section we used MATLAB commands to construct the **L** with desired poles. The code can be found in fig. 8.

Placing the poles for the minimal-state observer however was not so trivial. We couldn't manage to keep the pitch stable no matter where we placed the poles. Our best response was obtained by using the values in fig. 9. Why we had these difficulties are further discussed in the conclusion.

5.5 Comparison and evaluation of observers

When working with a state estimator it's interesting to see how close to the measured state our estimators get. The response of measured and estimated states using the LQR-regulator without and with integral effect can be found in fig. 23 and fig. 24 in appendix D, respectively. We chose to look at the states pitch, elevation rate and travel.

From the plots we can see that the estimation is very good for pitch and travel, and a bit more off point with elevation rate. The difference in tracking between the controllers with and without integral effect are also unnoticeable.

When we look at the minimal state estimator in fig. 25 however, the responses are very different. There is a phase lag on the estimate for pitch and elevation, but the estimated travel is fairly good. The plot is incomplete because we had to shut down the helicopter early. This was to not damage it because of the oscillating pitch. The abrupt turn in the measured pitch is due to the helicopter head reaching it's maximum pitch and rebounding the other way.

6 Conclusion

6.1 Mono- versus Multi-variable control

Mono-variable control gave us some degree of control over the helicopter, however, there were still some difficulties. The controller was not very robust and it was quite easy to provide a series of input that made the system unstable. One possible cause of this is our linearized system dynamics. The linearization makes it possible to control the helicopter, but only in the linear region surrounding the point of linearization, meaning that if we manage to get our helicopter outside the linear region it will become unstable.

Another limitation of mono-variable control is that it is assumed that the states are independent of each other, for example, changing elevation should not affect the pitch and vice versa. This is not true as changing one of the inputs, $\tilde{V}_d$ and $\tilde{V}_s$, will affect the other. An example of this can be seen if one change the pitch direction fast enough as this causes the elevation of the helicopter to drop.

The multi-variable controller was much more robust and accurate. After the final tuning it provided better control than the mono-variable alternative as well as removing stationary deviations. It is almost impossible to make the physical system unstable as long as the inputs are reasonable. The LQR with integral effect provides much better tracking, is faster, and more robust than the mono-variable controller.

Another benefit of the LQR is that our system is optimally controlled, and we can reap all the benefits that this implies. One point of exceptional importance is that this gives us much greater confidence as we move on to constructing our observer, as the LQR is an ideal controller since it gives a really low phase-margin[1].

6.2 Observers and difficulties in their implementation

Constructing and tuning our full-state observer took some effort, but in the end it was certainly possible. From the plots we constructed we see that it provides good tracking for both the normal LQR and the one with integral effect when compared to the actual measurement from the radial encoders.

However, when comparing our system with and without observer qualitatively, it is difficult to say if the behaviour of the helicopter has improved. The only thing we can say with any degree of confidence is that the system with full-state observer has comparable performance to the one without observer.

A huge assumption made when constructing our minimal state-observer is the separation principle[1]. We assumed that the error dynamics could be determined without impacting the system dynamics. No matter how hard we tried to tune our estimator we never managed to get a stable physical system. This could be because we simply do not have the required knowledge

to find the right **L**-gain. However, a more likely explanation is that the separation principle is not valid in this system. That together with too much simplification of our system model and suboptimal tuning of controllers.

6.3 Possible improvements

Firstly our model can be improved; We can measure the physical constants given to us in table 1 and we can construct less simplified equations of motion. This would give us a more correct starting point when we construct our controller.

If we had more time to tune both LQR controllers we would have more robust and accurate control of our helicopter. It could also enable us to get better results on the minimal state observer.

Utilizing a better algorithm for placing poles of our minimal state observer than simply placing them in a fan in the left half-plane could give us a more stable minimal state observer.

A Numerical values

Table 1: Numerical values for helicopter system

Symbol	Parameter	Value	Unit
g	Gravitational constant	9.81	m/s^2
l_c	Distance from elevation axis to counterweight	0.46	m
l_h	Distance from elevation axis to helicopter body	0.66	m
l_p	Distance from pitch axis to motor	0.175	m
m_p	Motor mass	0.72	kg
m_c	Counterweight mass	1.92	kg

B MATLAB code

Figure 3: Initialization of physical constants used by all proceeding code files.

```
%%%%%%%%%%% Calibration of the encoder and the hardware
%%%%%%%%%%% for the specific helicopter
Joystick_gain_x = 1;
Joystick_gain_y = -1;


%%%%%%%%%%% Physical constants
g = 9.81; % gravitational constant [m/s^2]
l_c = 0.46; % distance elevation axis to counterweight [m]
l_h = 0.66; % distance elevation axis to helicopter head [m]
l_p = 0.175; % distance pitch axis to motor [m]
m_c = 1.92; % Counterweight mass [kg]
m_p = 0.72; % Motor mass [kg]

v_s_star = 6.95; %Equilibrium voltage for elevation
k_f = g*(2*m_p*l_h-m_c*l_c)/(v_s_star*l_h); % Motorforce constant

k_1 = k_f/(2*m_p*l_p);
k_2 = (k_f*l_h)/(m_c*(l_c^2) + 2*m_p*(l_h^2));
k_3 = g*(m_c*l_c-2*m_p*l_h)/(m_c*(l_c^2)+2*m_p*((l_h^2)+(l_p^2)));
```

Figure 4: Pitch controller gains.

```
%%%%%%%%%%% Pitch controller
k_pp = 1/k_1; %exact value for k_pp, used in pitch regulator
k_pd = 0.9/k_1; %Best response we found
```

Figure 5: Travel rate gain.

```
%%%%%%%%%%% Travel rate controller
k_rp = 0.5/k_3;
```

Figure 6: Initialization of multivariable system and LQR regulator.

```
%%%%%%%%%%% Multivarible system
A = [0 1 0; 0 0 0; 0 0 0];
B = [0 0; 0 k_1; k_2 0];
C = [1 0 0; 0 0 1];

Q = diag([50 10 200]);
R = diag([5 10]);

K = lqr(A, B, Q, R);
P = inv(C*inv(B*K-A)*B);
```

Figure 7: Adding integral effect to the multivariable system and applying LQR.

```
%%%%%% Integral effect
A_aug = [A, zeros(3,2); C, zeros(2,2)];
B_aug = [B; zeros(2,2)];

Q_aug = diag([60 10 200 10 30]);

K_aug = lqr(A_aug, B_aug, Q_aug, R);
```

Figure 8: State-space model of the observer and finding **L** by placing poles.

```
%%%%%% Observer

A_obs = [0 1 0 0 0 0;
         0 0 0 0 0 0;
         0 0 0 1 0 0;
         0 0 0 0 0 0;
         0 0 0 0 0 1;
       k_3 0 0 0 0 0];

B_obs = [0 0;
         0 k_1;
         0 0;
       k_2 0;
         0 0;
         0 0];

C_obs = [1 0 0 0 0 0;
         0 0 1 0 0 0;
         0 0 0 0 1 0];

Ob = rank(obsv(A_obs, C_obs));

%Want observer poles x4-10 faster than system
%Slowest ple in A_aug_cl: -0.4434
%-> slowest pole in observer (-1.7)-(-4.5)


Pol = [-4 -5 -6 -7 -8 -9];
L = place(A_obs', C_obs', Pol')';
```

Figure 9: Minimal observer, observability checks for all measurement vectors, and finding **L** by placing poles.

```
%%%%%% Observer

A_obs = [0 1 0 0 0 0;
         0 0 0 0 0 0;
         0 0 0 1 0 0;
         0 0 0 0 0 0;
         0 0 0 0 0 1;
       k_3 0 0 0 0 0];

B_obs = [0 0;
         0 k_1;
         0 0;
       k_2 0;
         0 0;
         0 0];

C_obs = [1 0 0 0 0 0;
         0 0 1 0 0 0;
         0 0 0 0 1 0];

C_obs_not_p = [0 0 1 0 0 0;
               0 0 0 0 1 0];

C_obs_not_lambda = [1 0 0 0 0 0;
                    0 0 1 0 0 0];

Ob = rank(obsv(A_obs, C_obs));
Ob_not_p = rank(obsv(A_obs, C_obs_not_p));
Ob_not_lambda = rank(obsv(A_obs, C_obs_not_lambda));

%Want observer poles x4-10 faster than system
%Slowest pole in A_aug_cl: -0.4434
%-> slowest pole in observer (-1.7)-(-4.5)
convert = pi/180;
theta = 20*[1 -1 2 -2 3 -3]*convert;
Pol = -2*[exp(1i*theta(1)) exp(1i*theta(2)) exp(1i*theta(3))
          exp(1i*theta(4)) exp(1i*theta(5)) exp(1i*theta(6))];
L = place(A_obs', C_obs_not_p', Pol')';

A_LC = A_obs-L*C_obs_not_p;
poles_A_LC = eig(A_LC);
```

C Simulink diagrams

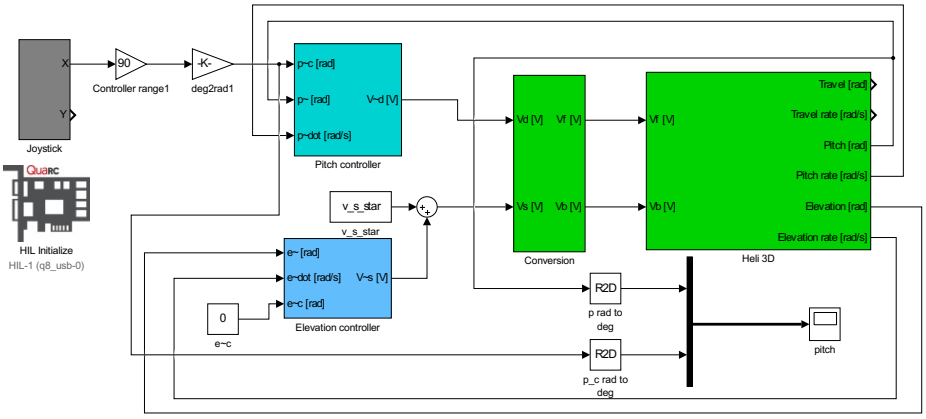


Figure 10: Simulink implementation of system with pitch controller.

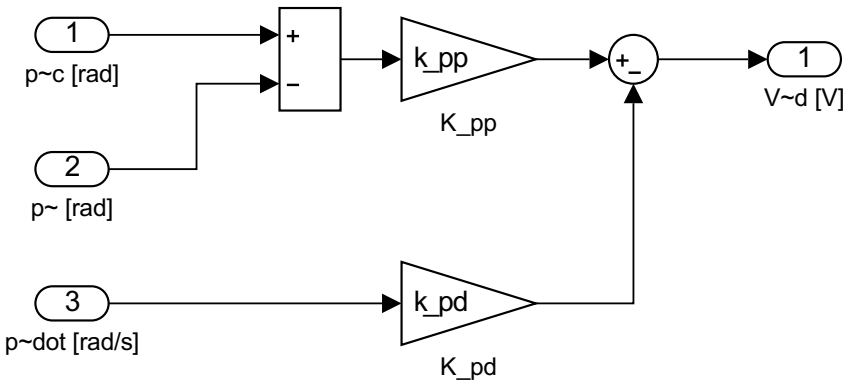


Figure 11: Simulink implementation of pitch controller.

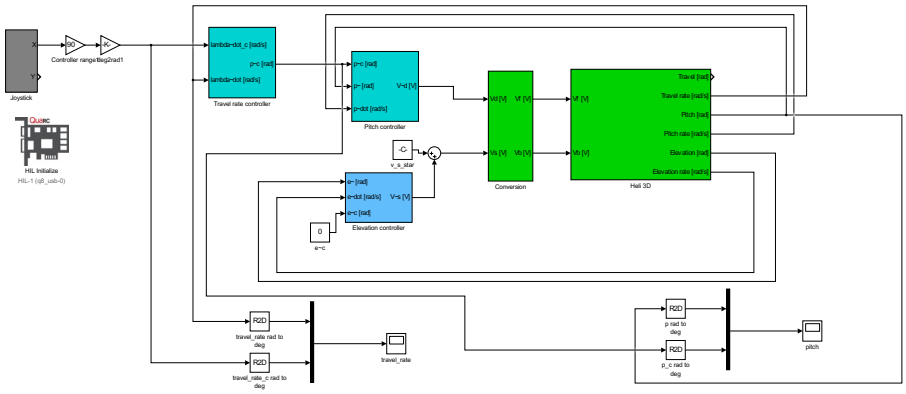


Figure 12: Simulink implementation of system with travel rate controller and pitch controller.

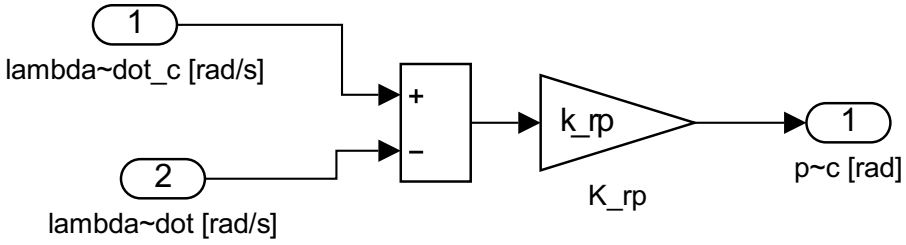


Figure 13: Simulink implementation of travel rate controller

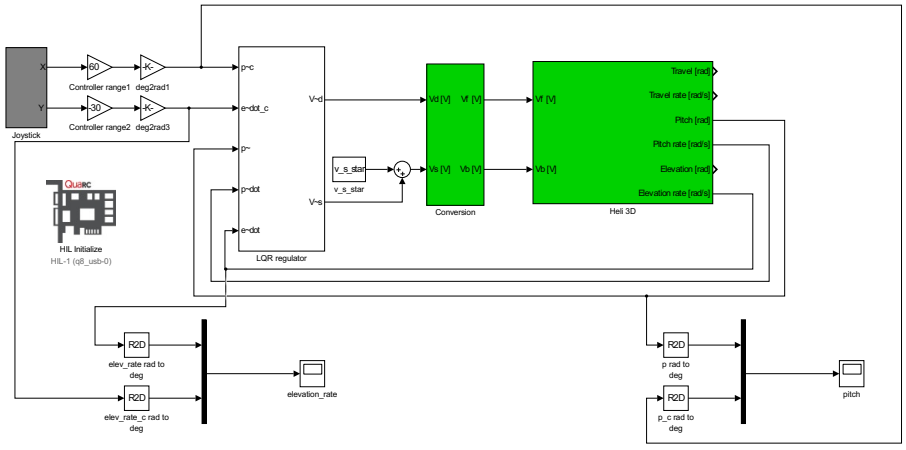


Figure 14: Simulink implementation of system with LQR controller.

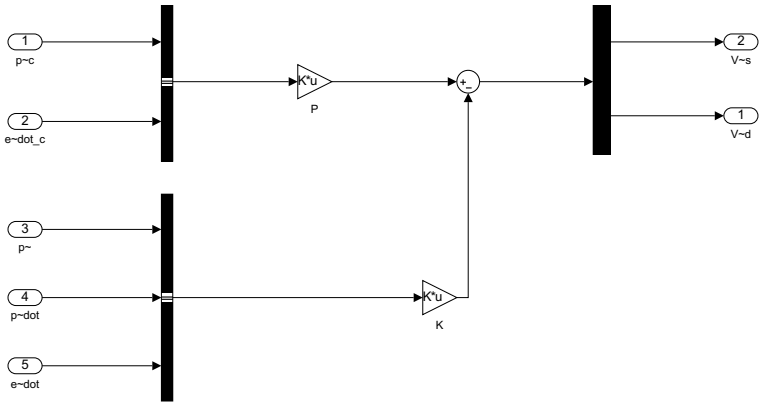


Figure 15: Simulink implementation of LQR controller.

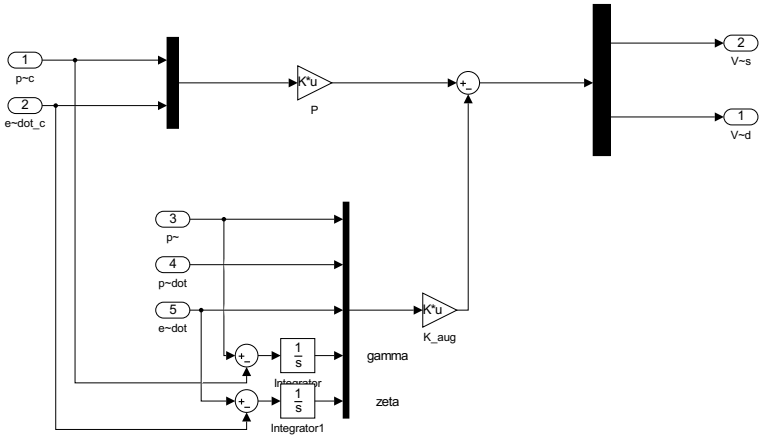


Figure 16: Simulink implementation of LQR controller with added integral effect.

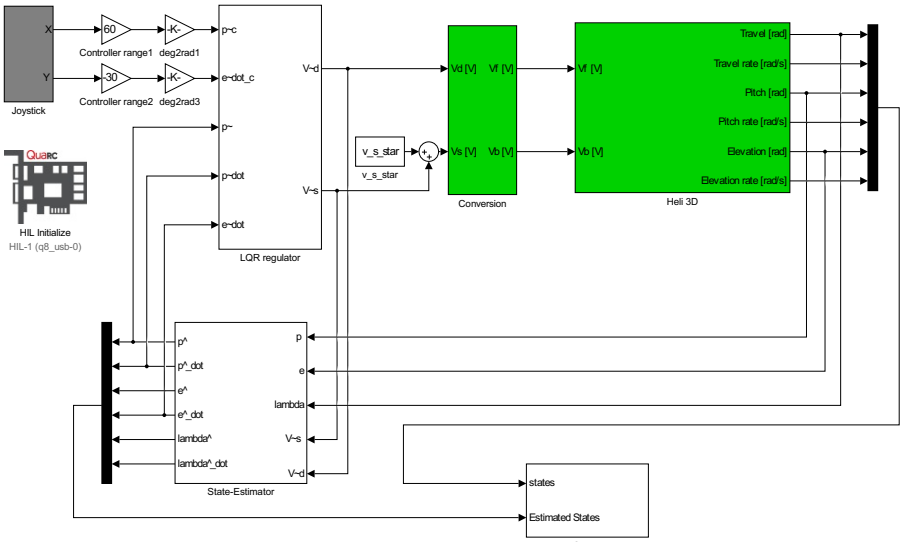


Figure 17: Simulink implementation of system with observer.

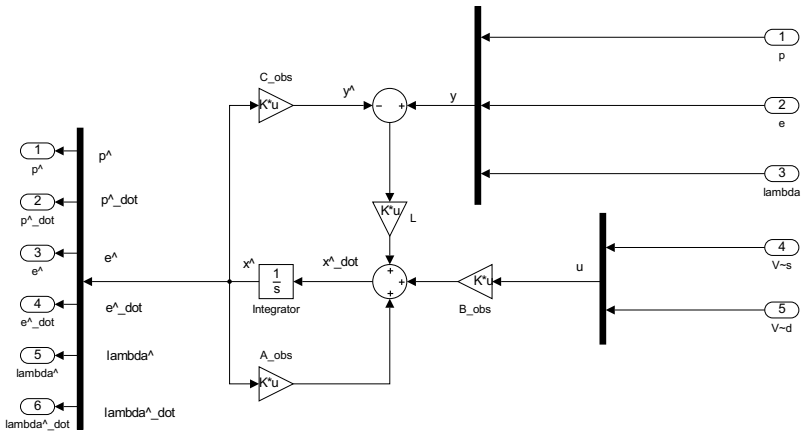


Figure 18: Simulink implementation of observer.

D Plots

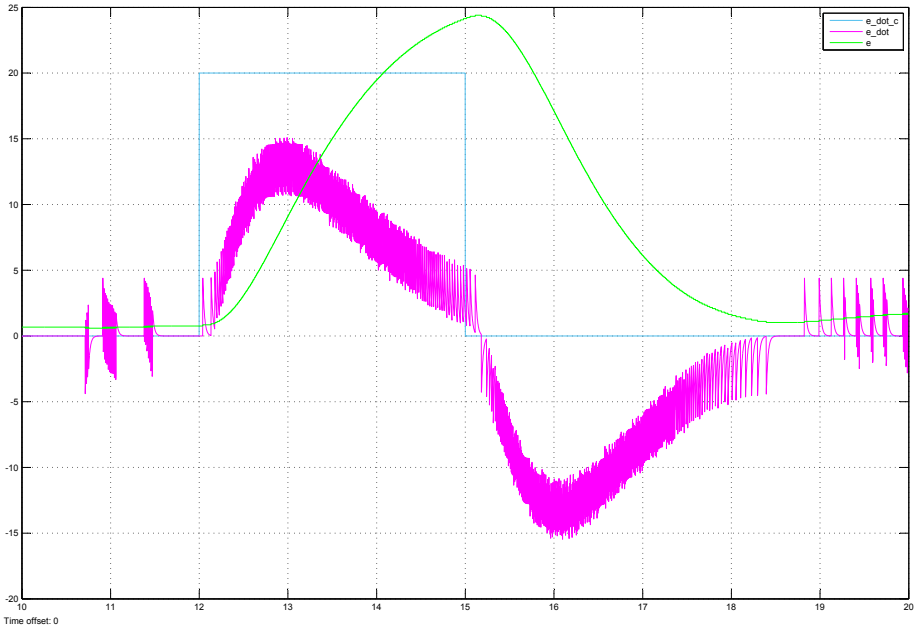


Figure 19: Elevation response for LQR without integral effect.

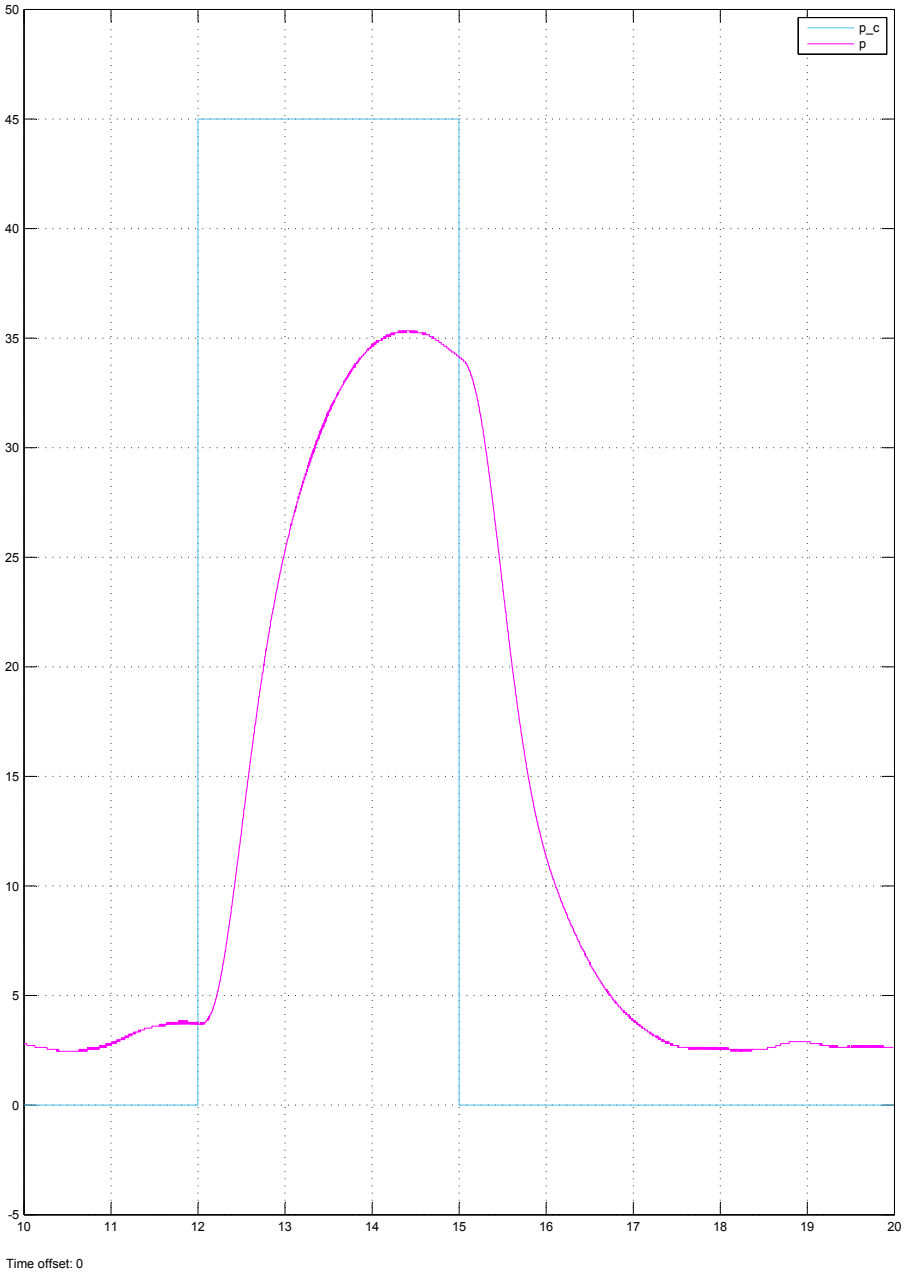


Figure 20: Pitch response for LQR without integral effect.

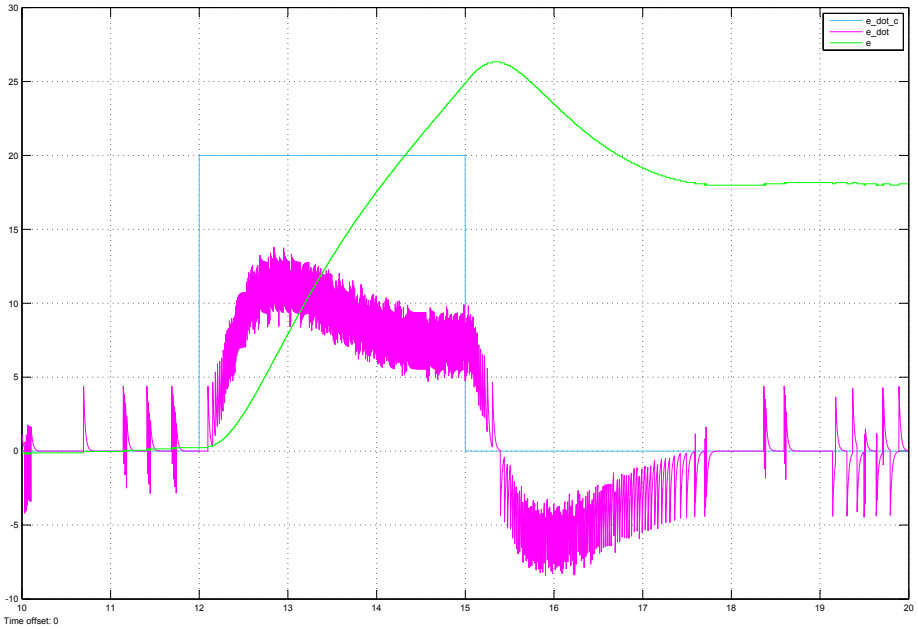


Figure 21: Elevation response for LQR with integral effect.

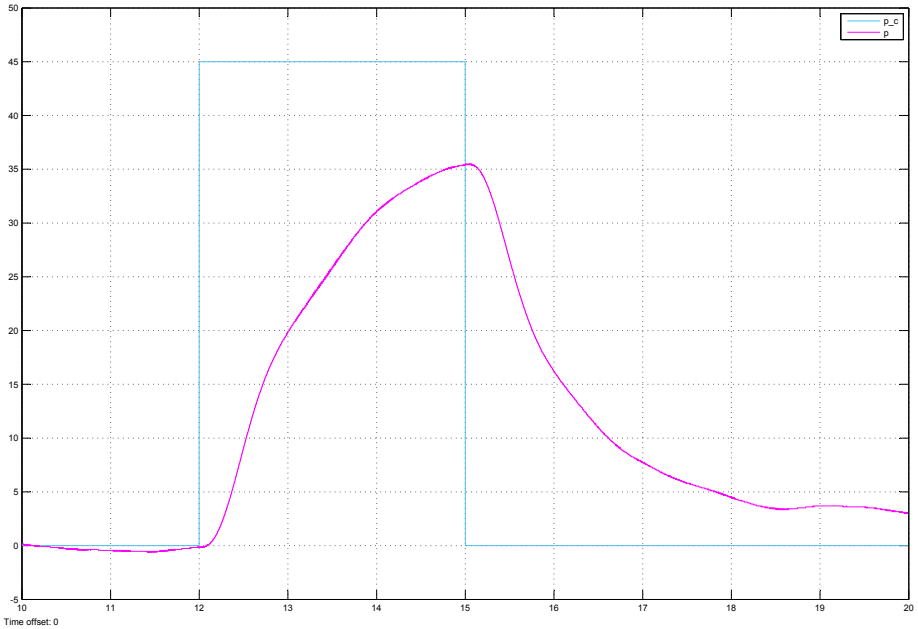


Figure 22: Pitch response for LQR with integral effect.

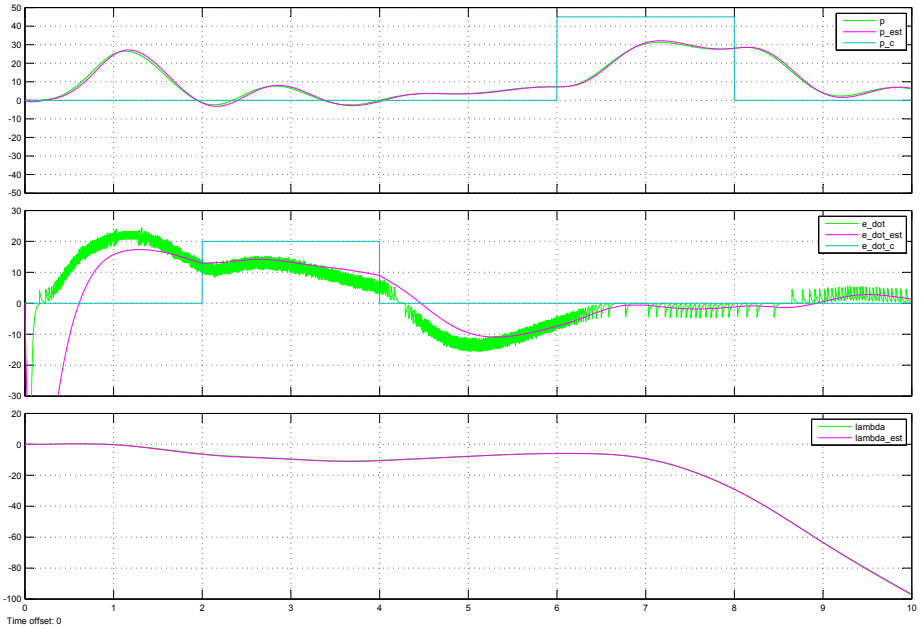


Figure 23: Full state observer with LQR controller without integral effect.

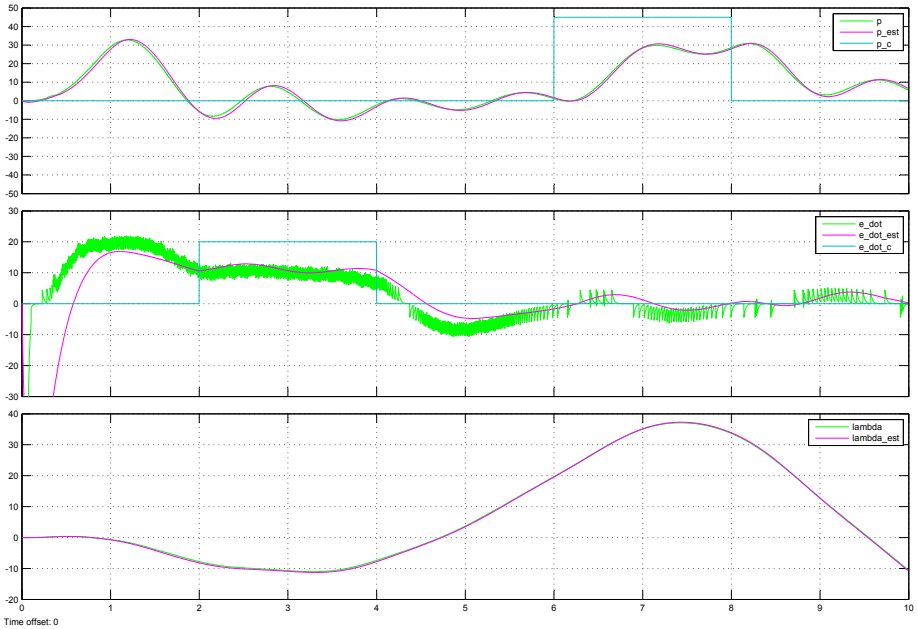


Figure 24: Full state observer with LQR controller with integral effect.

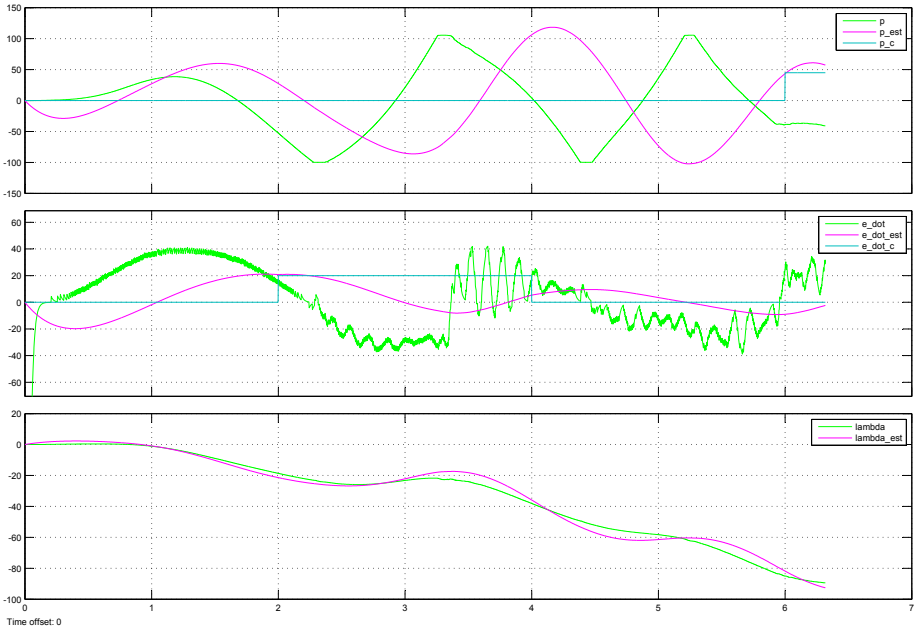


Figure 25: Minimal state observer with LQR controller with integral effect.

References

[1] Chi-Tsong Chen. *Linear System Theory and Design*. Oxford University Press, Incorporated, 2014.

[2] ntnu itk. *Labreport Tips and Tricks*. URL: `https://github.com/ntnu-itk/labreport`.

[3] B.A. Foss J.G. Balchen T. Andresen. *Reguleringsteknikk*. Institutt for teknisk kybernetikk, NTNU, 2016.