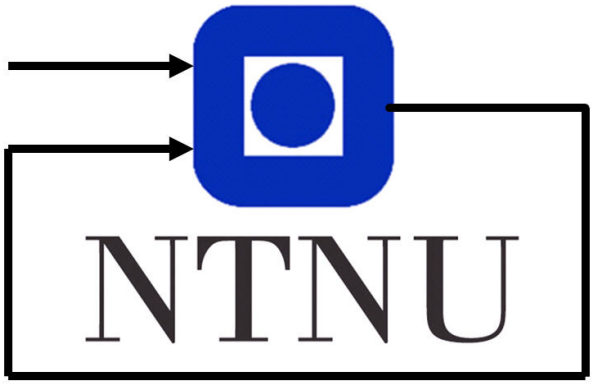
Helicopter Lab Report - TTK4115

Group 73
Daniel Azagra Næss
Lars Lien

November 18, 2025



Department of Engineering Cybernetics

Abstract

This report highlights the work conducted at the helicopter lab in the course TTK4115.

The helicopter was controlled using two different types of controllers: a monovariable PD controller, which controlled the pitch, and an LQR controller. We tested different poles and weighting matrices for the PD controller and the LQR, respectively.

Next, we extended the control system by adding state observers. First, a Luenberger Observer was implemented and tested with different observer poles. Afterwards, the observer was extended to a Kalman Filter. Adding the Kalman Filter resulted in an LQG controller.

The system performed well with all the different controllers. However, using only the PD controller did not allow control of any variables other than pitch. The other controllers enabled control of both pitch and elevation.

Contents

1 Introduction

This report highlights the work done at the helicopter lab in the course TTK4115-Linear systems theory at NTNU. In the lab-project we were assigned 4 parts, where we would use 2 different ways of controlling the helicopter, as well as introducing two different observers in the last two parts:

1. Monovariable control
2. LQR control
3. Luenberger observer
4. Kalman filter

This is also how the report is organized, as it is divided in 4 parts.

The helicopter has two propellers on one end and one counterweight on the other end, as shown in figure 1. The helicopter is modeled as three point masses: one on the side of the counterweight and two to represent the weight of the propellers. The forces and joints on the propellers can be seen in figure 2, where λ is the base, or travel angle; e is the angle affecting elevation; and p is the pitch angle.

The forces, also shown in figure 2, are F_f and F_b for the front and back propeller forces, $F_{g,c}$ for the gravitational force from the counterweight, and $F_{g,f}$ and $F_{g,b}$ for the gravitational forces from the propeller masses. The helicopter arms, l_c, l_h, and l_p, are shown in figure 3.

Figure 1: The helicopter in the lab.

2 Part I - Monovariable control

The helicopter lab consists in analyzing different control methods for the helicopter shown in figure 1.

2.1 Equations of motion

The first part of the lab consists of finding the equations of motion for the helicopter.

If we assume a constant relationship between the propellar force and voltage, we get

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

Where K_f is a motor constant and V_f and V_b are the motor voltages for the front and back propellar. In the rest of the report we will use the sum of the voltages $V_s = V_f + V_b$ and the difference between the voltages $V_d = V_f - V_b$ to control the helicopter.

From figure 2 and figure 3 you can show that the equations of motion are

$$J_p \ddot{p} = L_1 V_d \tag{2a}$$
$$J_e \ddot{e} = L_2 \cos(e) + L_3 V_s \cos(p) \tag{2b}$$
$$J_\lambda \ddot{\lambda} = L_4 V_s \cos(e) \sin(p) \tag{2c}$$

where L_1, L_2, L_3 and L_4 are constants, and the moments of inertia around the pitch, travel and elevation are J_p, J_λ and J_e.

If we linearize the system we get the linearized equations

$$\ddot{p} = K_1 V_d \tag{3a}$$
$$\ddot{e} = K_2 \tilde{V_s} \tag{3b}$$
$$\ddot{\lambda} = K_3 p \tag{3c}$$

where $K_1 = 0.5505,\ K_2 = 0.0885,\ K_3 = 0.6117$ are constants and $\tilde{V_s} = V_s - V_{s,0}$ where $V_{s,0}$ is such that $\tilde{V_s} = 0$ gives an equilibrium point at $e = 0$. In the lab we experimentally determined $V_{s,0} = 7.2$.

2.2 PD Control

For the first part of the lab a PD controller was implemented to control the pitch angle p. The equation for the controller is

$$V_d = K_{pp}(P_c - p) - K_{pd}\dot{p} \tag{4}$$

where p_c is the reference for the pitch angle given to the controller.

If you insert this into equation 3a You get

$$\ddot{p} = K_1 K_{pp} p_c - K_1 K_{pp} p - K_1 K_{pd} \dot{p} \tag{5}$$

If you take the laplace transform and do some algebra you can find the transfer function

$$G(s) = \frac{p(s)}{p_c(s)} = \frac{K_1 K_{pp}}{s^2 + K_1 K_{pd} s + K_1 K_{pp}} \tag{6}$$

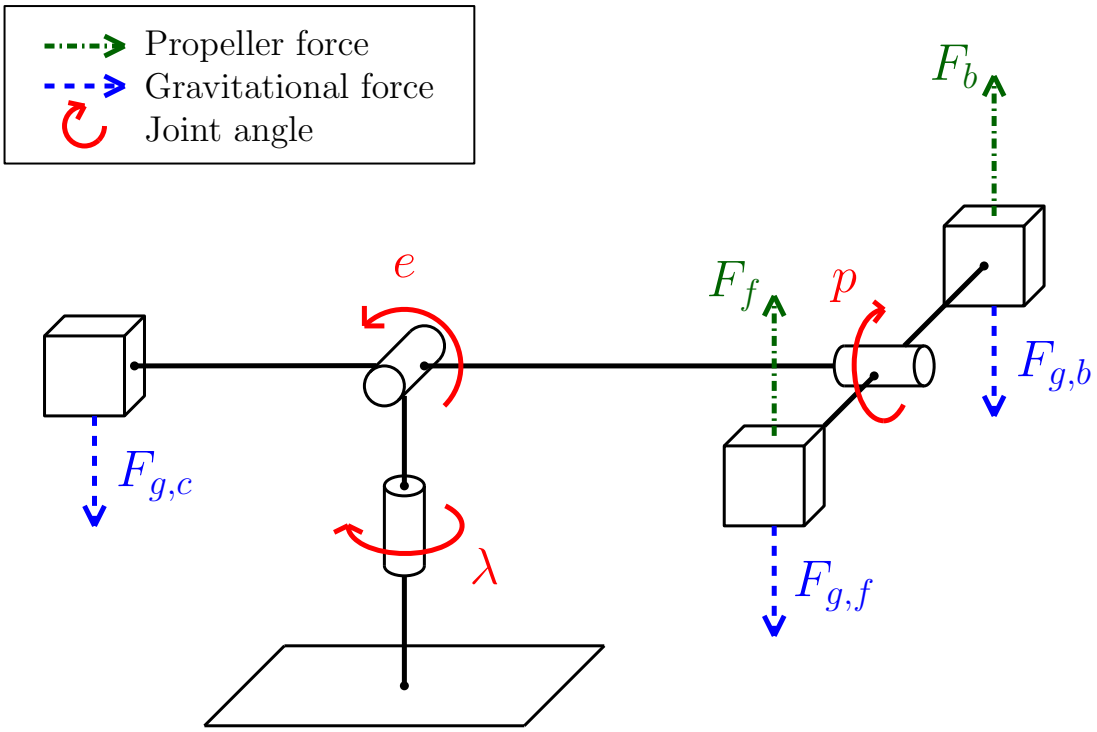


Figure 2: The forces and joints on the helicopter. Drawing taken from the Lab Preparation document [1]

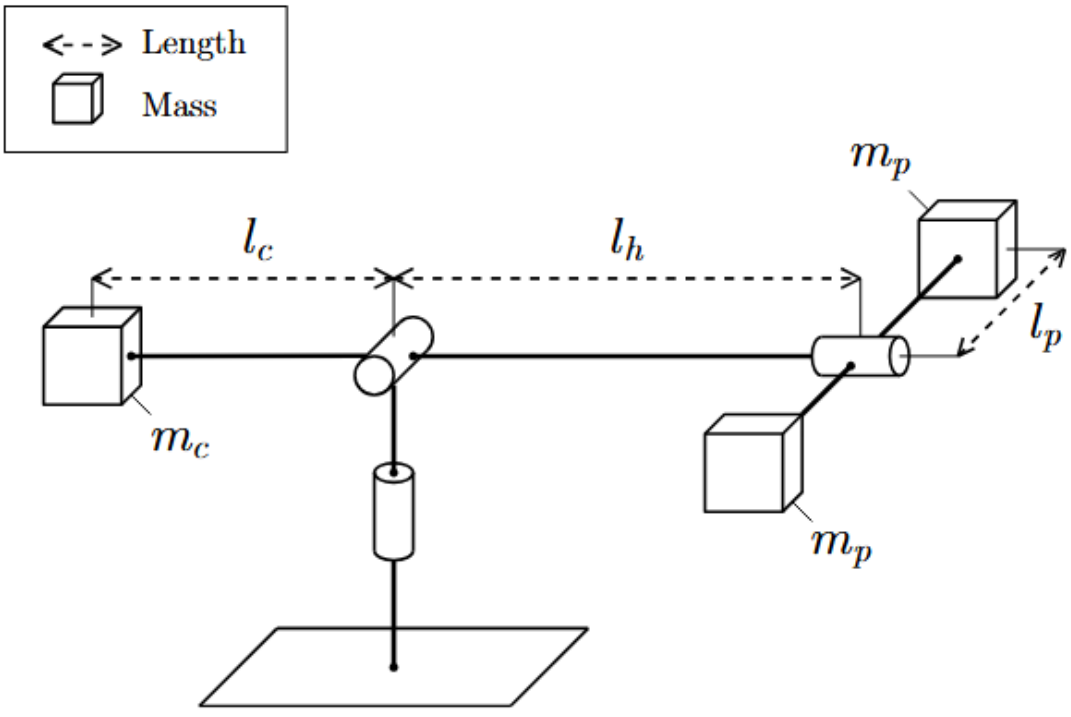


Figure 3: The masses and lengths of the helicopter. Drawing taken from the Lab Preparation document [1]

2.3 Pole placement

The poles λ_i of a transfer function $G(s)$ are the values for s that make the denominator $D_{tf}(s) = 0$. It is possible to design a controller just by choosing different values for the poles. The transfer

function given in equation 6 has two poles, λ_1 and λ_2.

It is possible to find the controller gains for the PD-controller using equation 7

$$K_{pp} = \frac{\lambda_1 \cdot \lambda_2}{K_1} \tag{7a}$$

$$K_{pd} = \frac{\lambda_1 + \lambda_2}{K_1} \tag{7b}$$

In theory, if the poles of the closed loop system are all in the left half plane (LHP) the system should be stable, and if there are one or more poles in the right half plane (RHP) the system should become unstable.

The system should also have a faster response the closer the poles are to being unstable, and a slower response the more negative their real parts are.

There is one caveat with this. Placing the poles too far to the left in the s-plane can lead to instability due to different factors like actuator saturation. And for non-linear systems instability due to saturation gets amplified, which is a problem for our helicopter, as it is a non linear system. Placing the poles too far too the right can also lead to instability due to model noise or modeling errors.

2.4 Test plan

For our testing in the lab there are some main pole configurations that could be interesting.

Firstly we would like to test how close to the RHP the poles can come before the closed-loop system becomes unstable.

Secondly we would like to test how negative the poles can be before the system becomes unstable due to saturation.

And thirdly we would like to see what poles give oscillations, and if we could get any poles to give us a critically damped system.

In theory, any complex conjugate poles should give oscillations.

Taking this into consideration, we decided to do the following tests:

Test nr.	Poles $[\lambda_1, \lambda_2]$	Hypothesis
1	-30, -25	Unstable due to saturation
2	-2, -2	Stable, fast response
3	-2, -5	Stable, should be faster than test 2

2.5 Experimentation

In all the experiments we tested the system with a step response. That was done by making $p_c \approx 0.5$ when $t = 5s$ We also wanted to test a critically damped system, but the plotting data was overwritten by another test, so the data was lost.

2.5.1 Test 1

When doig the experimentation we started with eigenvalues $\lambda_1 = -30$ and $\lambda_2 = -25$. We didn't plot anything as the helicopter crashed instantly. We instead started turning the eigenvalues more towards the RHP and found that the system started becoming stable when $\lambda_1 = \lambda_2 \approx -10$. The reason it was unstable with the first eigenvalues was probably due to saturation in the actuator.

2.5.2 Test 2

In test 2 we got some unexpected results as the system was unstable. As seen in figure 4 the helicopter crashed even though the poles were in the LHP. We we're unsure why this was the case. Maybe it was due to model noise, so the stability margins were not big enough even though the poles were in the LHP.

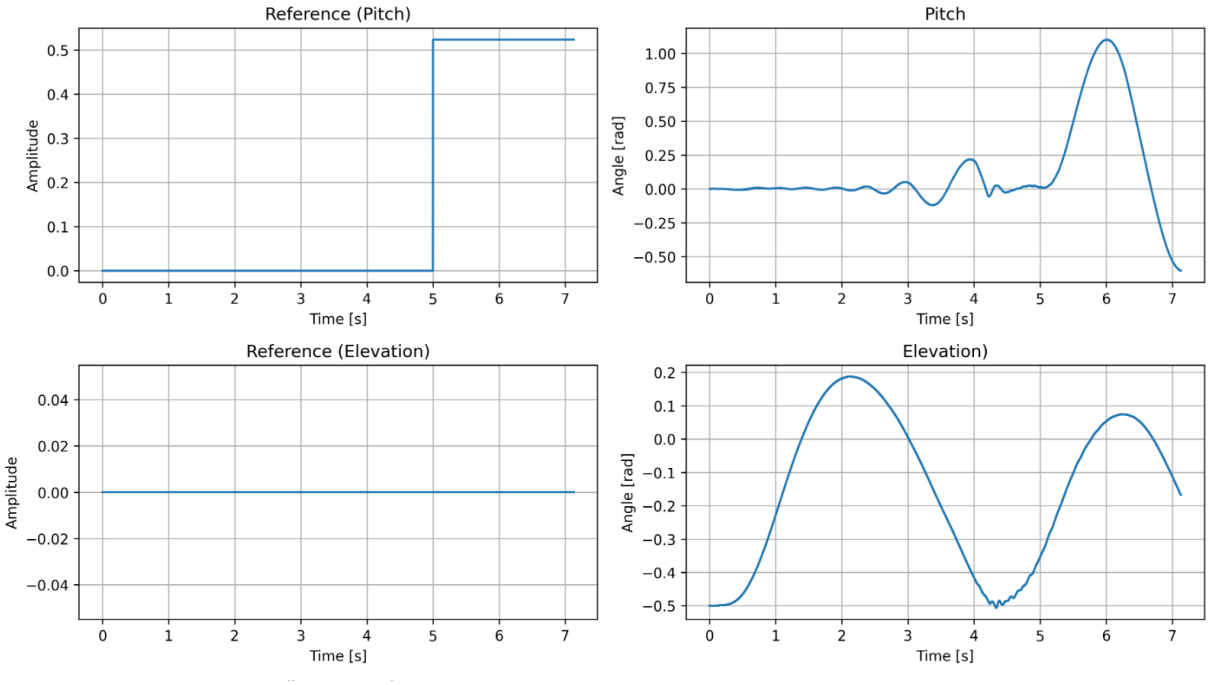


Figure 4: Plots from test 2

2.5.3 Test 3

In this test we tried some other poles, and got a pretty good step response. As seen in figure 5 the system managed to track the reference very well, even though there is some steady-state error.

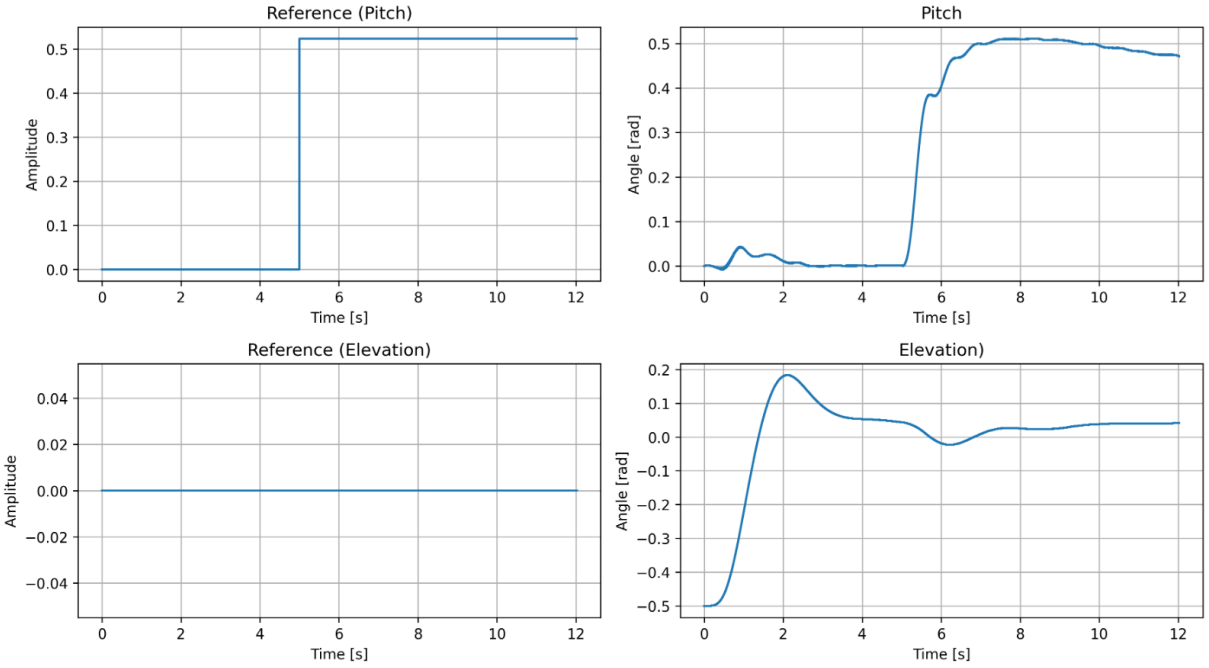


Figure 5: Plots from test 3

2.5.4 Conclusion

The control system behaved well when we found some poles that we're fitting and had enough margin. If the poles got smaller than $\lambda_i \approx -10$ the system would start becoming unstable due to saturation. And if they got bigger than $\lambda_1 = \lambda_2 \approx 2$ the system would become unstable.

3 Part II - Multivariable control

In the previous part we only controlled a single variable. To be able to control many variables at the same time we will in this part use state space methods and a Linear Quadratic Regulator.

3.1 State Space Formulation

The system equations for the pitch, pitchrate and elevationrate is given in equations 3. From these the state space formulation was determined to be:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix} \boldsymbol{u}.$$

where x and u are given by:

$$\boldsymbol{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix}, \qquad \boldsymbol{u} = \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix},$$

3.2 Controllability

For the LTI system

$$\dot{x} = Ax + Bu,$$

the controllability matrix is

$$\mathcal{C} = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}.$$

Using our state space fromulation we computed

$$AB = \begin{bmatrix} 0 & K_1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad A^2 = 0 \quad \Rightarrow \quad A^2B = 0.$$

Hence the controllability matrix reduces to

$$\mathcal{C} = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & K_1 \\ 0 & K_1 & 0 & 0 \\ K_2 & 0 & 0 & 0 \end{bmatrix},$$

$\text{rank}(\mathcal{C}) = 3 = n$, assuming $K_1 \neq 0$, and $K_2 \neq 0$ The constants are not zero as shown in section 2.1, which means that the system is controllable.

3.3 Feedback and Feedforward

In our state-feedback controller we want to employ a reference feedforward. The controller term then becomes

$$\boldsymbol{u} = \boldsymbol{F}\boldsymbol{r} - \boldsymbol{K}\boldsymbol{x} \tag{8}$$

Where $\boldsymbol{F}$ is the feedforward matrix, and

$$\boldsymbol{K} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \end{bmatrix} \tag{9}$$

is the controller gain.

In order for the system to track the reference $\boldsymbol{r}$. You can show that the feedforward matrix has to be:

$$\boldsymbol{F} = \begin{bmatrix} k_{11} & k_{13} \\ k_{21} & k_{23} \end{bmatrix}.$$

The $\boldsymbol{F}$ matrix was reassigned for each test based on the values for $\boldsymbol{K}$ by the LQR minimization.

3.4 LQR Control

LQR control consists in finding the optimal $\boldsymbol{K}$ matrix by minimizing the cost function

$$J = \int_0^\infty \left(x^T Q x + u^T R u \right) dt \tag{10}$$

Where Q and R are weighting matrices.

We used the Matlab function

```
K = lqr(A, B, Q_lqr, R_lqr);
```

to find our optimal controller gains.

3.5 Integral Effect

To include integral action for the elevation rate and pitch angle, two integrator states γ and ζ are introduced. Their dynamics are given by

$$\dot{\gamma} = p_c - p, \qquad \dot{\zeta} = \dot{e}_c - \dot{e}.$$

This leads to the extended state–space model

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{G}\boldsymbol{r}, \qquad \boldsymbol{u} = \boldsymbol{F}\boldsymbol{r} - \boldsymbol{K}\boldsymbol{x}.$$

The augmented state vector and the new $\boldsymbol{A}$ and $\boldsymbol{G}$ matricies become

$$\boldsymbol{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \\ \gamma \\ \zeta \end{bmatrix}, \qquad \boldsymbol{G} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}.$$

The introduction of the integrator states reduces steady-state error in pitch and elevation rate. When tuning the controller, increasing the diagonal weights in Q_{LQR} associated with γ and ζ makes the controller correct these errors more aggressively, while increasing the values in R_{LQR} penalizes control effort and slows the response. We found that the inclusion of the integral action did not change the F matrix.

3.6 LQR Tests

Test nr.	Q	R
1	$\text{diag}(10, 10, 100)$	I
2	$\text{diag}(100, 100, 100)$	$0.04 \cdot I$

The tests were preformed by doing a step test for the pitch at 6 seconds with different values. The resulting pitch is shown in Figure 6.

We performed two tests for the LQR system. The values for the Q and R matrices are shown in Table 3.6. Our hypothesis was that the first test would give a slow response, while the second test would give a faster response.

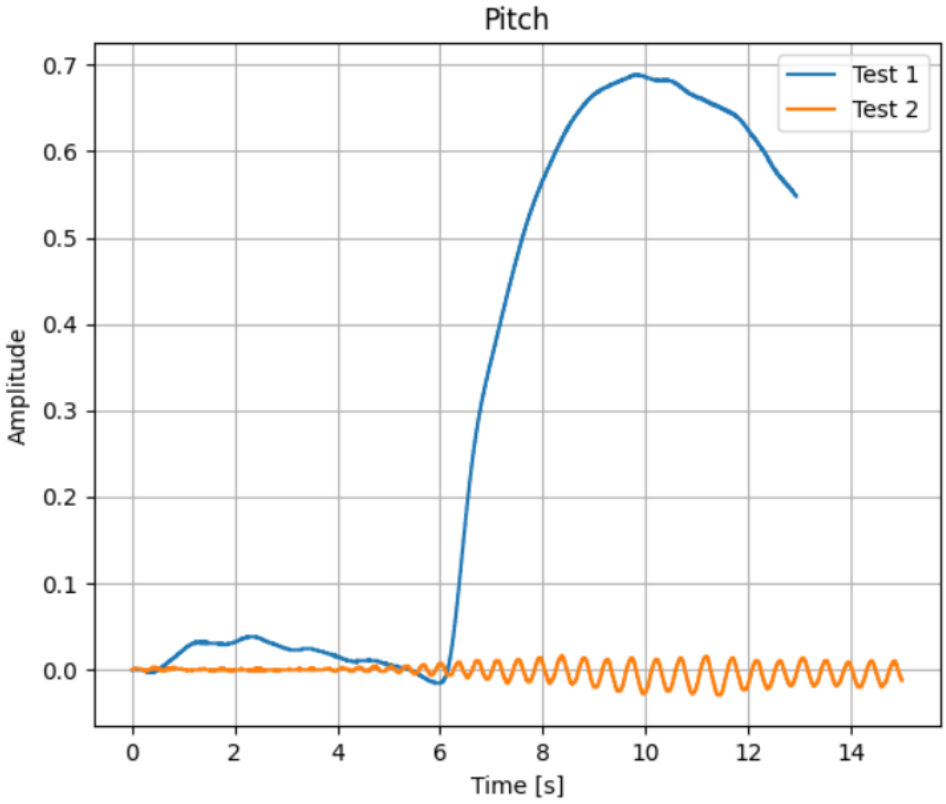


Figure 6: Shows the Test 1 and the ossilating Test 2

3.7 LQR Test Results

For the first test, we used a step block to set the reference pitch at 6 seconds. The response is shown as the blue line in Figure 6. For the second test, we expected to see a faster response, but we were surprised to observe oscillations in the pitch. Our theory is that the regulator became too fast due to high Q values and low R, causing the pitch to oscillate. For the second test, we only recorded the oscillations, and we disabled the step refrence. In hindsight, we should have performed more tests so that we could have compared the first test with additional ones, but unfortunately, this was not done.

3.8 Integral Tests and Results

The tests were performed by applying a step input to the pitch at 6 seconds using different values. The resulting pitch is shown in Figure 6. The tests were quite boring. We did not see this when i were at the lab, but when we ploted the reponses they looked very simmilar. We had hoped to see differnet reponses, but we must conclude that different values for the integral action did not have any significant effect.

Test nr.	Q_d	Hypothesis
1	$10 \cdot I$	Small Q values should give slow response
2	$\text{diag}(10, 10, 10, 100, 100)$	Larger Q values should give faster response
3	$\text{diag}(10, 10, 100, 10, 10$	The same as test 2 but for different states

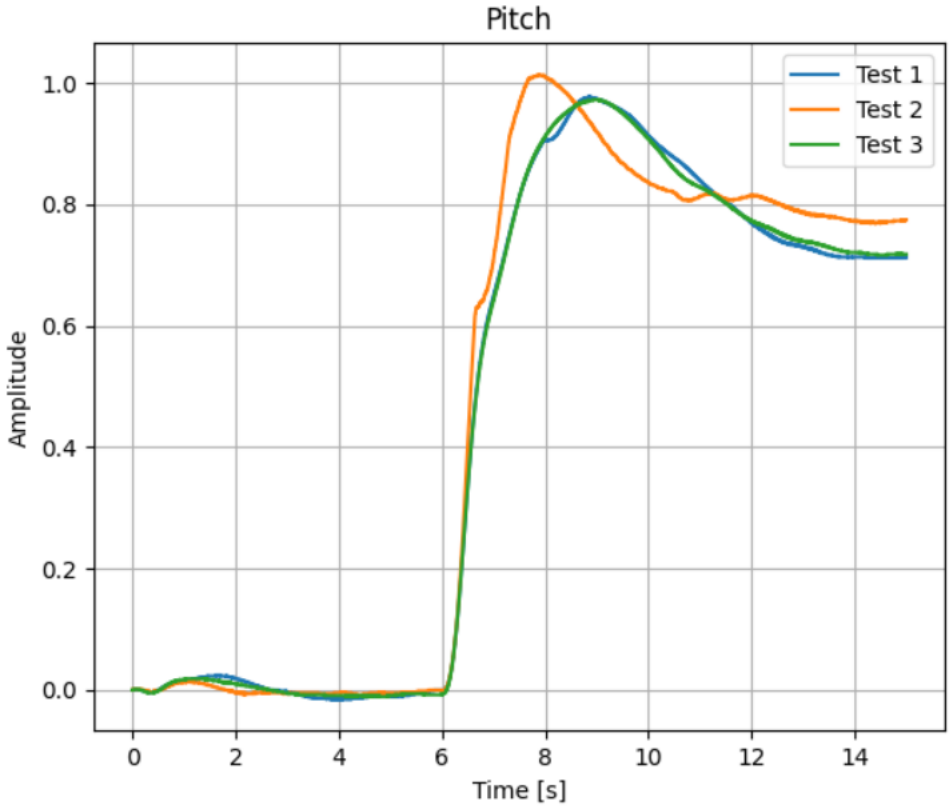


Figure 7: Shows the tree integral tests

4 Part III - Luenberger observer

In this task our goal was to use an IMU to measure the movement and find the state of the helicopter. For this task we are using a Luenberger observer to estimate the states based on the system model and measurements. In this sectiton we start by going through the different parts of the observer we used, and then we tested it with different tuning to see the differences.

4.1 State Formulation and Observability

For this task we have extended the state space formulation from the LQR's 3 states to also include the elevation e and the travel rate $\dot{\lambda}$. The equations for the travel is given by 3c and the new system state $\boldsymbol{x}$ and the matrecies $\mathbf{A}$ and $\mathbf{B}$ are thus given by:

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \dot{\lambda} \end{bmatrix}, \quad u = \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ K_3 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \end{bmatrix}$$

The observability matrix for a state space system is given by

$$\mathcal{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \vdots \\ \mathbf{CA}^{n-1} \end{bmatrix}$$

Our system is observable if $\text{rank}(\mathcal{O}) = n = 5$ To find the minimal number of states we need observe the system we tried different $\mathbf{C}$ matrecies and then used Matlabs `obsv(A,C)` to check the rank. The minimal observable $\mathbf{C}$ was found to be

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

With observability matrix

$$\mathcal{O} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ K_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & K_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{11}$$

With $\text{rank}(\mathcal{O}) = 5 = n$

Since we want to estimate all the states in our observer and we had an IMU measuring all states we instead used $\mathbf{C} = \mathbf{I}$ for all our tests.

4.2 Angle Measurements

As we saw in the previous part we need some angle measurements to be able to observe the system. Since the IMU measure the force counteracting gravity we can find equations for the pitch p and the elevation e. Given the accelleration from the IMU $[a_x, a_y, a_z]$ the angles are given by

$$p = \arctan\left(\frac{a_y}{a_z}\right), \tag{32a}$$

$$e = \arctan\left(\frac{q}{\sqrt{a_y^2 + a_z^2}}\right)$$

4.3 Observer Arcitecture

Our observer is on the form

$$\dot{\hat{x}} = \hat{\mathbf{A}}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} + \mathbf{L}\left(\mathbf{y} - \hat{\mathbf{C}}\mathbf{x}\right) \tag{12}$$

where the $\mathbf{L}$ matrix was found by placing poles we determined in our test plan. To place the poles we used the Matlab command `L = place(A', C', p).'` where p is the vector of poles.

4.4 Test plan

To test our system we tried two sets of poles.

4.4.1 Slow Poles

The first test we tried poles close to the imaginary axis. The poles were $\mathbf{p} = \begin{bmatrix}-2, -3, -2, -1.5, -3\end{bmatrix}$. Since the Our hypotesis was that this would lead to a observer that will track the states more slowly. We also predict that this observer would be quite resiliant to noise on the measurements.

4.4.2 Fast Poles

For the second test we tried to make the poles larger to see a different effect. The poles this time were $\mathbf{p} = [13, 9, 8, 10, 9]$. In this case we would expect the observer to track the states faster because the error dynamics are faster. This should also make the noise resilliance a bit worse for the system.

4.5 Test Results

For this test we compare the elevation estimated by the observer to the elevation measured by the encoders. The first test, using slow poles, is shown in Figure 8, while the test with fast poles is shown in Figure 9.

4.5.1 Slow Poles

As predicted, the response of the filter was slow. This is clearly visible in the elevation data. The filter was initialized to zero during startup, but the helicopter starts at approximately -0.5. When the startup occurred, the estimate required so much time to reach the actual (encoder) value that oscillations were induced during flight. These oscillations did not fully die out within the first 10 seconds.

On the other hand, the noise in the estimate was minimal. Comparing the signals in Figure 8 to those in Figure 9, we can clearly observe less noise in the slow-pole case.

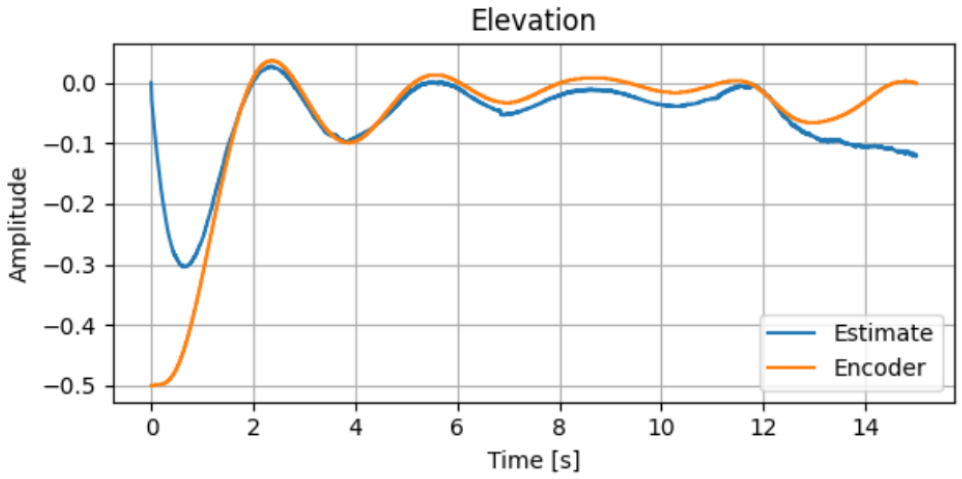


Figure 8: The Luenberger with slow response

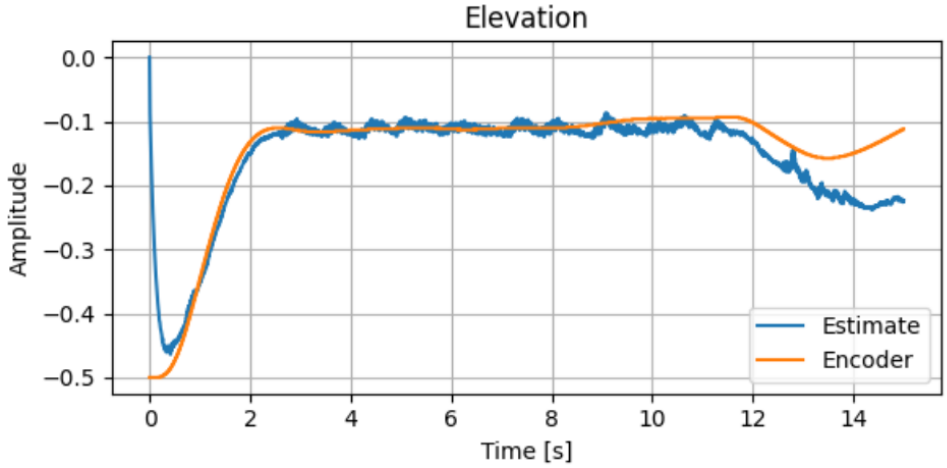


Figure 9: The Luenberger with fast response

4.5.2 Fast Poles

The second test, shown in Figure 9, exhibits a much faster response. The oscillations were eliminated, and the system quickly converged to the reference. However, we observed significantly more noise in the estimate compared to the signal in Figure 8.

5 Part IV - Kalman filter

5.1 Motivation

The Kalman filter is based on the Luenberger observer from equation 12 The difference is that where the the estimator feedback gain L for the luenberger observer is determined manually or through pole-placement, the Kalman filter estimates L based on our estimates for process noise and measurement uncertainty. Using this you estimate an optimal model uncertainty P_k for every timestep. That means that the estimator gain is changing through time. Therefore we denote it as $L[k]$, where k is the given time-step.

5.2 Discretization

As the kalman filter used in this lab is the discrete-time Kalman Filter, the system from has to be discretized. In Part IV we aim to estimate all the states. This means that our state vector will be

$$\boldsymbol{x} = \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \lambda \\ \dot{\lambda} \end{bmatrix} \tag{13}$$

And the system matrices in continuous time will be

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ K_3 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{14}$$

$\boldsymbol{C} = \boldsymbol{I}$ and $\boldsymbol{D} = \boldsymbol{0}$ using the `c2d` function in latex with a timestep of 0.002 we got the system matrices

$$\boldsymbol{A_d} = \begin{bmatrix} 1.0000 & 0.0020 & 0 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.0020 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0.0000 & 0.0000 & 0 & 0 & 1.0000 & 0.0020 \\ 0.0012 & 0.0000 & 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{15}$$

and

$$\boldsymbol{B_d} = \begin{bmatrix} 0 & 0.0000 \\ 0 & 0.0011 \\ 0.0000 & 0 \\ 0.0002 & 0 \\ 0 & 0.0000 \\ 0 & 0.0000 \end{bmatrix} \tag{16}$$

the $\boldsymbol{B}$ and $\boldsymbol{D}$ matricies stay the same.

5.3 Prediction and correction

The Kalman filter consists of two main steps. Firstly the system predicts the states using the system model, also called the a-priori estimate. Secondly, when you get a measurement, the system corrects the a-priori estimate with the measurement, also called the a-posteriori estimate. The algorithm for the steps is as follows:

A priori:

$$\bar{x}[k+1] = A_d\hat{x}[k] + B_d u[k] \tag{2.21d}$$

$$\bar{P}[k+1] = A_d\hat{P}[k]A_d^\top + Q_d \tag{2.21e}$$

A posteriori:

$$L[k] = \bar{P}[k]C_d^\top \left(C_d\bar{P}[k]C_d^\top + R_d\right)^{-1} \tag{2.21a}$$

$$\hat{x}[k] = \bar{x}[k] + L[k]\left(y[k] - C_d\bar{x}[k]\right) \tag{2.21b}$$

$$\hat{P}[k] = \left(I - L[k]C_d\right)\bar{P}[k]\left(I - L[k]C_d\right)^\top + L[k]R_dL^\top[k] \tag{2.21c}$$

Where L is the Kalman gain, P is the measurement covariance, R_d the measurement covariance matrix, Q_d the model covariance matrix. The $[k]$ denotes the value at the given time-step k. The bar on top denotes the a-priori estimate. And the hat denotes the a-posteriori estimates.

5.4 Test plan

For the test plan we decided to primarily test 3 different things. $Q_d \to \infty$ and $Q_d \to 0$. As well as trying to get a good response. For all the tests we used a value for R_d that we got by measuring the noise of the sensors while the helicopter was standing still at the linearization point:

$$R_d = \begin{bmatrix} 0.0067 & 0.0009 & 0.0073 & 0.0012 & 0.0013 & -0.0002 \\ 0.0009 & 0.0056 & 0.0014 & 0.0018 & -0.0011 & 0.0004 \\ 0.0073 & 0.0014 & 0.0313 & -0.0066 & -0.0004 & -0.0003 \\ 0.0012 & 0.0018 & -0.0066 & 0.0112 & 0 & -0.0010 \\ 0.0013 & -0.0011 & -0.0004 & 0 & 0.0115 & 0.0021 \\ -0.0002 & 0.0004 & -0.0003 & -0.0010 & 0.0021 & 0.0023 \end{bmatrix} \tag{17}$$

Our tests were:

Test nr.	Q_d	Hypothesis
1	$100000 \cdot \mathbf{I}$	The observer might become unstable, as it will only rely on noisy measurements, and will barely use the model to predict the states
2	$0.000001 \cdot I$	Unstable, will barely use the measurements to correct the a-priori estimate
3	$0.005I$	Stable

5.5 Results

5.5.1 Test 1

In this test we didn't give any reference other than 0. In the first test our hypthesis was that the system would become unstable and not track the reference that well, as the measurements

would be too noisy to give a good estimate, but it seems like the state estimator was just fine. That might be because we didn't give the system a reference change. You can see from figure 10 that there was some noise. At $t \approx 5$ the IMU signal cut out for unknown reasons, and you can see from figure 11 that the covariance increases drastically when that happens. At the same time you can see in figure 12 that the estimate starts slightly drifting away from the encoder measurement.

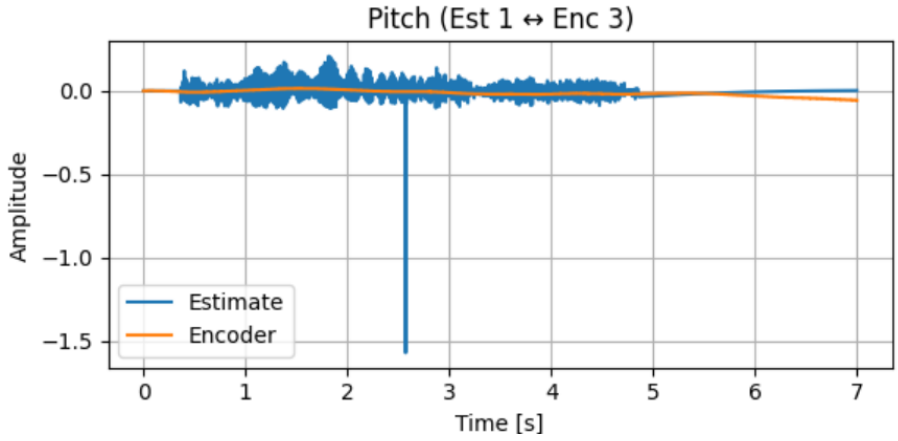


Figure 10: Estimate vs real value for pitch for test 1

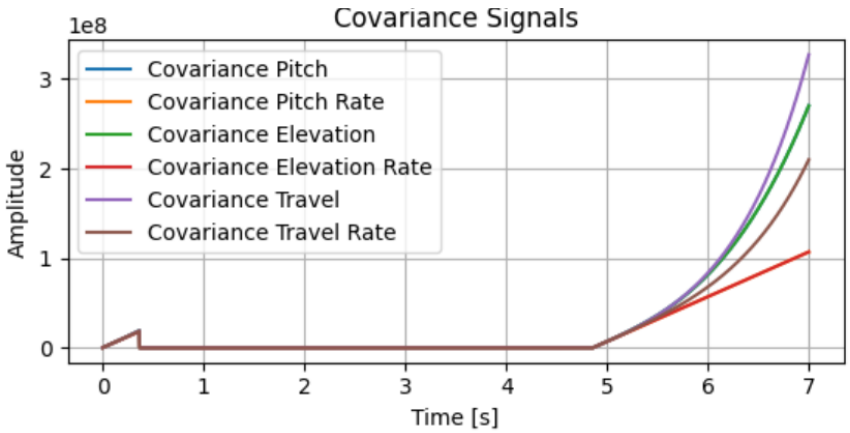


Figure 11: Covariance for test 1. At $t \approx 5$ the IMU dies, so the covariance skyrockets.

5.5.2 Test 2

In this test we had a step signal for the reference at $t = 4$. So what is plotted in 13 is the step response for the pitch. In this test we decided to see what happened when $Q_d \to 0$. What happened was that the estimates had no noise. This is due to the filter trusting the model a lot more than the measurements, just using the measurements for small corrections. At $t \approx 4$ the IMU signals died, and you can see the estimates quickly diverging from the real values.

5.5.3 Test 3

In test 3 we wanted to see if we could find a good middleground for our Q_d matrix. We wanted to make a stable system that would track the reference. At $t = 7$ the reference changes to 0.6 and in figure 15 we can see that the system manages to track the reference pretty well. In figure 16 you can see there is a big spike inthe covariance at $t = 0$. The reason for this is that we

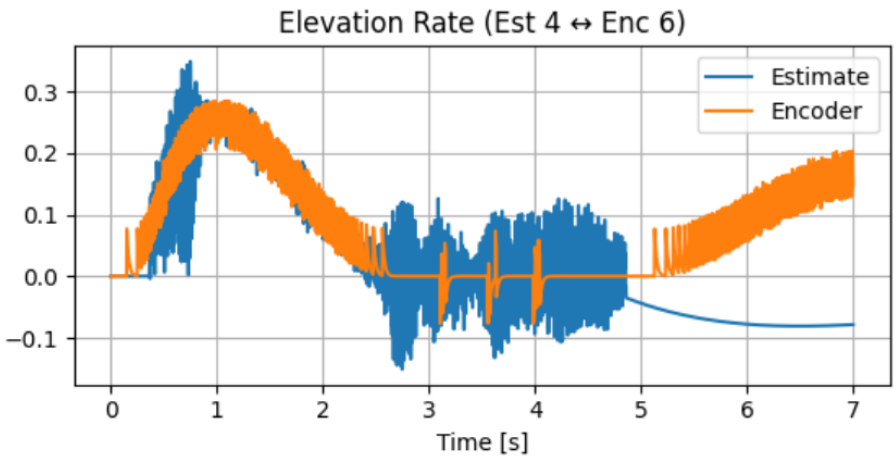


Figure 12: Estimate vs real value for elevation rate for test 1, the quick changes are from the helicopter control input spiking at different times.

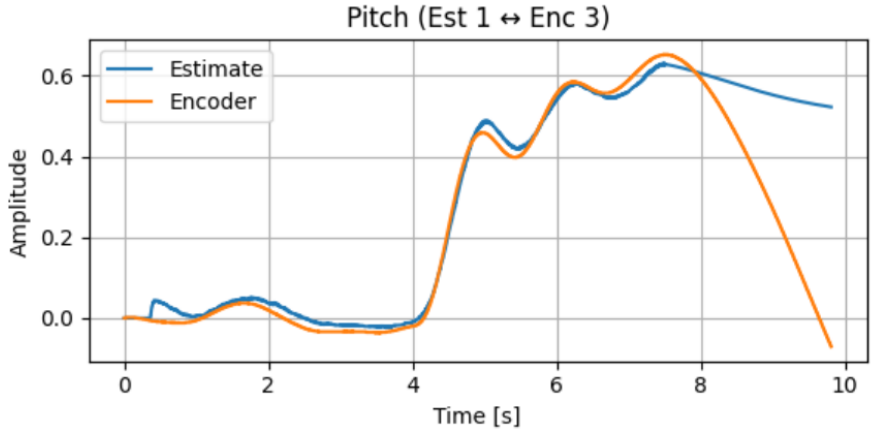


Figure 13: Pitch from test 2

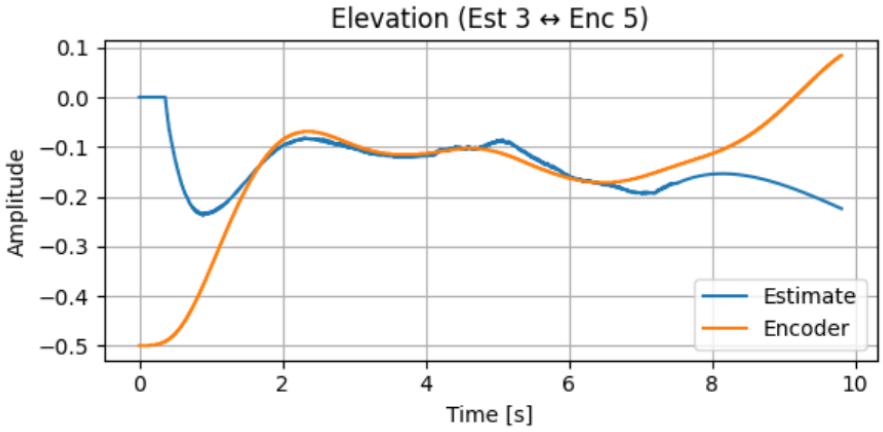


Figure 14: Elevation from test 2

initialized all the states at 0, when they aren't 0 at the start. This was fine, because the system managed to correct this anyway.

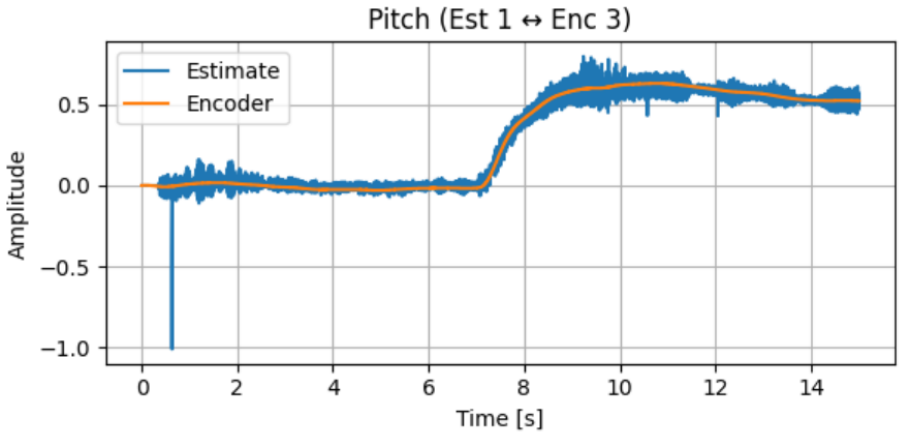


Figure 15: Pitch for test 3

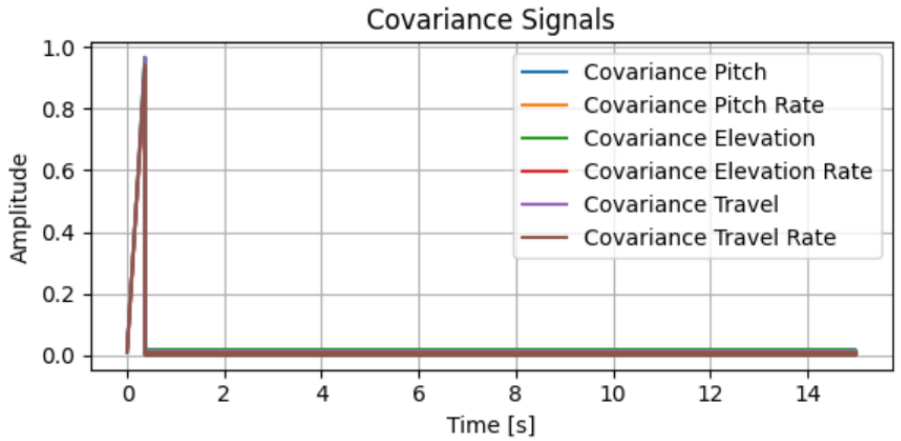


Figure 16: Covariance for test 3

5.5.4 Test Conclusion

The kalman filter, when used as a state estimator can give many different behaviours depending on what you set the covariance matrices and how you initialize the states. In our tests we showed that the kalman filter worked pretty well no matter what we set or Q_d matrix to. Probably because the model is pretty accurate. We found that a solid middle-ground between removing noise and tracking the reference was a covariance matrix $Q_d = 0.005 \cdot I$

References

[1] Emil Johnsen et al. *Helicopter Lab Preparation*. 2025.