TTK4115 - Linear Systems Theory

Helicopter 2 Lab Report

Group 54
Anders Tørresen (Student 129220)
Even Årekol (Student 484224)
Fredrik Siem Taklo (Student 484137)

Abstract

As part of the course TTK4115 "Linear System Theory" we have been experimenting with a model of a helicopter in a lab. This report contains a brief hypothesis for each assignment, as well as our findings after having performed them.

13/11/2024

Table of Contents

List of Figures

1 Monovariable control

The goal of this part is to linearize the system around a flat elevation and pitch angle, then implement a PD controller to make it stay in place. To do this, a model of the system is necessary. The lab preparation provides the following figures and equations ([2]):

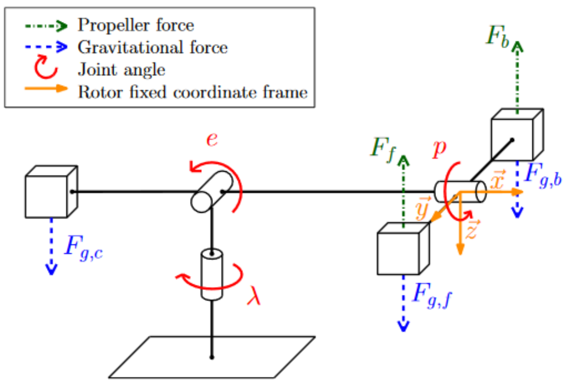

Figure 1: Forces and angles.

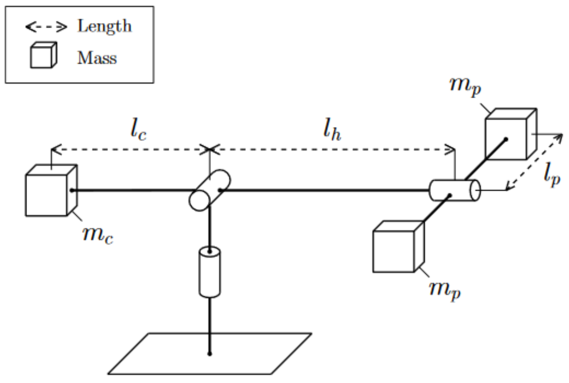

Figure 2: Masses and distances.

1.1 Theory

1.1.1 Equations of motion

The forces from the propellers are given by

$$F_f = K_f V_f \quad (1a)$$
$$F_b = K_f V_b \quad (1b)$$

where K_f is a motor force constant proportional to the voltage output to the propeller.

Newtons second law of rotation states that

$$\Sigma\tau = I\alpha \quad (2)$$

where the sum of the torques τ is equal to the moment of inertia I multiplied by the angular acceleration α. This formula can be used alongside the forces that cause an angular acceleration for the angles p, e and λ.

Equations of motion

$$J_p\ddot{p} = L_1 V_d \quad (3a)$$
$$J_e\ddot{e} = L_2 cos(e) + L_3 cos(p) \quad (3b)$$
$$J_\lambda\ddot{\lambda} = L_4 V_s cos(e) sin(p) \quad (3c)$$

V_s is the sum of of motor voltages, while V_d is the difference between them:

$$V_s = V_b + V_f \quad (4a)$$
$$V_d = V_b - V_f \quad (4b)$$

L_1, L_2, L_3, L_4 are constants that can be calculated by looking at the sum of forces acting on their respective axis

$$L_1 = K_f l_p \tag{5a}$$
$$L_2 = g(m_c l_c - 2m_p l_h) \tag{5b}$$
$$L_3 = K_f l_h \tag{5c}$$
$$L_4 = -K_f L_h \tag{5d}$$

1.1.2 Linearization

The system will be linearized around an equilibrium point of $e = 0°$ and $p = 0°$.

At this point it is assumed that the forces of inertia are constant and given by

$$J_p = 2m_p l_p^2 \tag{6a}$$
$$J_e = m_c l_c^2 + 2m_p l_h^2 \tag{6b}$$
$$J_\lambda = m_c l_c^2 + 2m_p(l_h^2 + l_p^2) \tag{6c}$$

A constant voltage V_{s0} is needed to keep the elevation axis still at $e = 0°$. The value of V_{s0} will be an offset to V_s such that the origo becomes the equilibrium point. The linearized sum of voltages can be expressed as $\tilde{V}_s = V_s - V_{s0}$

The equation for linearization of a function $f(x, y)$ at a point (a, b) is given by

$$f(x,y) \approx f(a,b) + \left.\frac{\partial f(x,y)}{\partial x}\right|_{(a,b)} (x-a) + \left.\frac{\partial f(x,y)}{\partial y}\right|_{(a,b)} (y-b) \tag{7}$$

By applying the linearization formula to the equations of motion, linearizing around the point $(0, 0)$, the linearized equations of motions become

Linearized equations of motion

$$\ddot{p} = K_1 V_d \tag{8a}$$
$$\ddot{e} = K_2 \tilde{V}_s \tag{8b}$$
$$\ddot{\lambda} = K_3 p \tag{8c}$$

The constants K_1, K_2 and K_3 are given by:

$$K_1 = \frac{L_1}{J_p} \tag{9a}$$
$$K_2 = \frac{L_3}{J_e} \tag{9b}$$
$$K_3 = \frac{L_4}{J_\lambda} \tag{9c}$$

1.1.3 PD Control

A PD controller is to be implemented to control the pitch angle p. The controller is given as

$$V_d = K_{pp}(p_c - p) - K_{pd}\dot{p} \tag{10}$$

where p_c is the desired reference for the pitch angle p given by the joystick.

Deriving the transfer function of a PD controller on the system is useful for analyzing the behavior of the system based on its poles and zeroes. Substituting 10 into the equation of motion for the pitch angle given in (8a) yields

$$\ddot{p} = K_1 K_{pp}(p_c - p) - K_1 K_{pd}\dot{p} \tag{11}$$

The transfer function is then given by the Laplace transform of (11):

$$G(s) = \frac{P(s)}{P_c(s)} = \frac{K_1 K_p p}{s^2 + K_1 K_p d s + K_1 K_p p} \tag{12}$$

1.1.4 Pole placement

By solving the characteristic equation of the linearized system's transfer function, it was found that the poles of the system (λ_1, λ_2) can be described as a function of K_{pp} and K_{pd}.

$$K_{pp} = \frac{\lambda_1 \lambda_2}{K_1} \tag{13}$$

$$K_{pd} = -\frac{\lambda_1 + \lambda_2}{K_1} \tag{14}$$

An important principle regarding pole placement is that a system is stable for poles in the left-half-plane and unstable for poles in the right-half-plane, and that the oscillations of the response are given by the magnitude of the imaginary part of the poles.

The following poles with their respective hypotheses were tested using the pitch controller:

1. **Critically damped** $(\lambda_1 = -10)$ $(\lambda_2 = -10)$
   When both poles are negative and equal to each other a fast response is expected, without overshoot or oscillations.
2. **Overdamped** $(\lambda_1 = -5)$ $(\lambda_2 = -10)$
   Two negative poles without imaginary parts should yield a slower response without overshooting.
3. **Underdamped** $(\lambda_1 = -6 + 3i)$ $(\lambda_2 = -6 - 3i)$
   Two negative poles with imaginary parts should result in decaying oscillations before stabilizing.
4. **Marginally stable** $(\lambda_1 = 0)$ $(\lambda_2 = -5)$
   With one pole being zero, it should theoretically be possible to keep the system in equilibrium as long as it is not perturbed. This is unrealistic with an imperfect model like this, so it is assumed to become unstable.
5. **Unstable** $(\lambda_1 = 5)$ $(\lambda_2 = 5)$
   Two possible poles means that the system is unstable. The response to the input will keep growing.
6. **Unstable with oscillation** $(\lambda_1 = 2 + 3i)$ $(\lambda_2 = 2 - 3i)$
   Adding imaginary numbers to unstable poles should give oscillations that grow over time. It may be hard to see this behavior clearly for the pitch controller, since it is likely to crash before then.

1.2 Results

1.2.1 Manual control

The Simulink model for manual control is shown in figure 3. The raw values from the joystick were not high enough to get the helicopter off the ground. A gain of around 5 for V_d and 7 for V_s gave a result that was manageable to control around the equilibrium point. Using the joystick, a suitable value for V_{s0} was found at around 6.8V, where both the pitch and elevation angle stayed relatively still, as shown in figure 4.

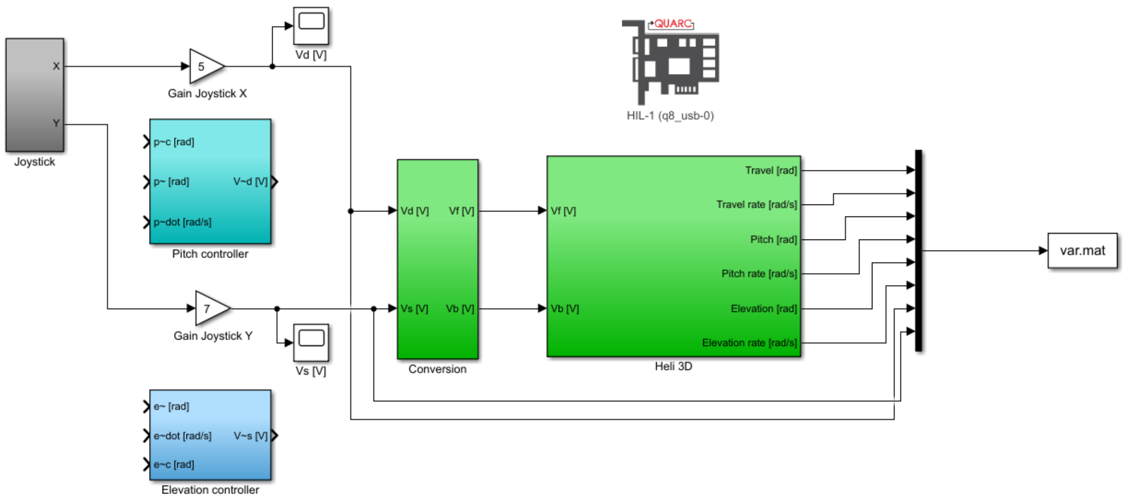

Figure 3: Simulink model during manual control

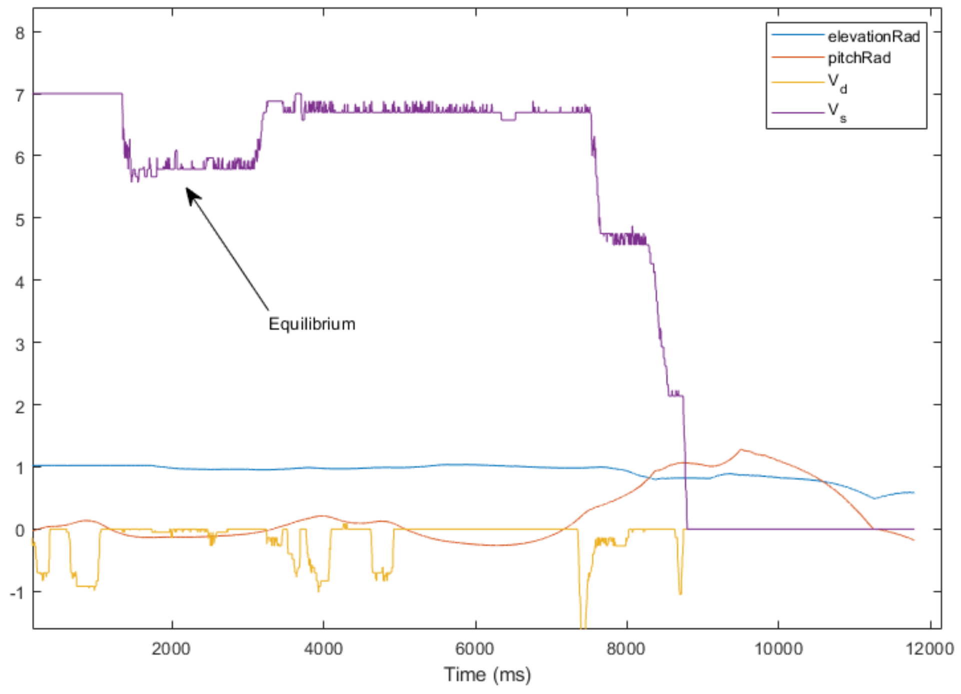

Figure 4: Angles and output voltages while finding V_{s0}

1.2.2 Parameter identification

By monitoring the encoder values, it was discovered that the elevation angle was -0.465 radians from the equilibrium point while resting on the ground. Using this as an offset shifts the origo to the linearized position.

Substituting V_{s0} into equation 8b, an expression for the motor force constant K_f is derived:

$$K_f = g\frac{2m_p - m_c}{V_{s0}} \tag{15}$$

1.2.3 PD control

The simulink model below was achieved by connecting the joystick axes to the pitch and elevation controllers, then modelling the pitch controller based on equation 11.

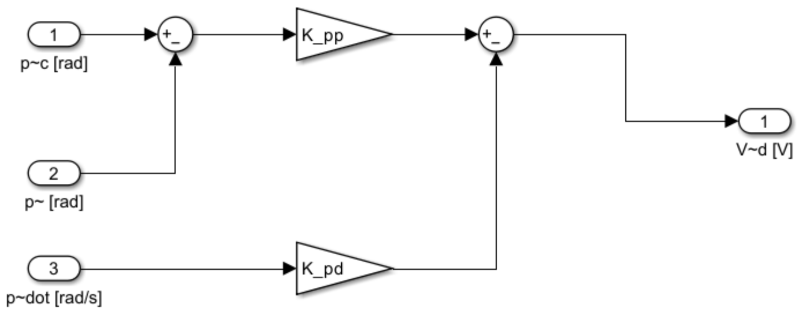

Figure 5: Simulink pitch controller given by equation 11

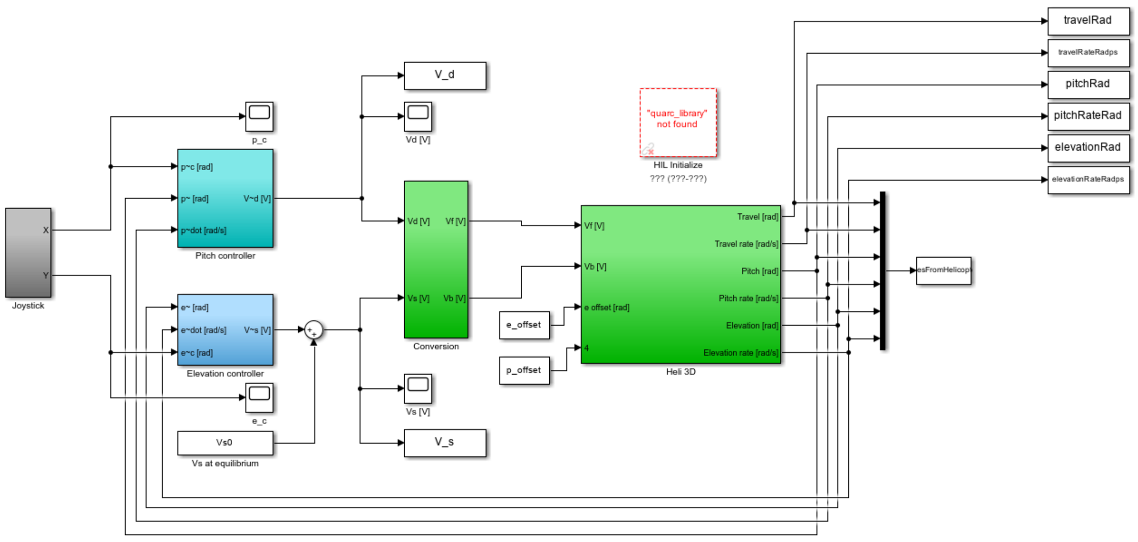

Figure 6: Simulink model with pitch and elevation controllers

1.2.4 Evaluation of pole placement

The system was tested by using the planned poles for the pitch controller from section 1.1.4. A slightly faster response was observed for the critically damped poles, but they all had varying degree of overshoot and oscillations. Some of this can likely be attributed to the imperfect model and the equilibrium points not being exactly zero degrees relative to gravitational pull. For the marginally stable and unstable poles, the pitch would quickly flip on its side and send the travel axis spinning.

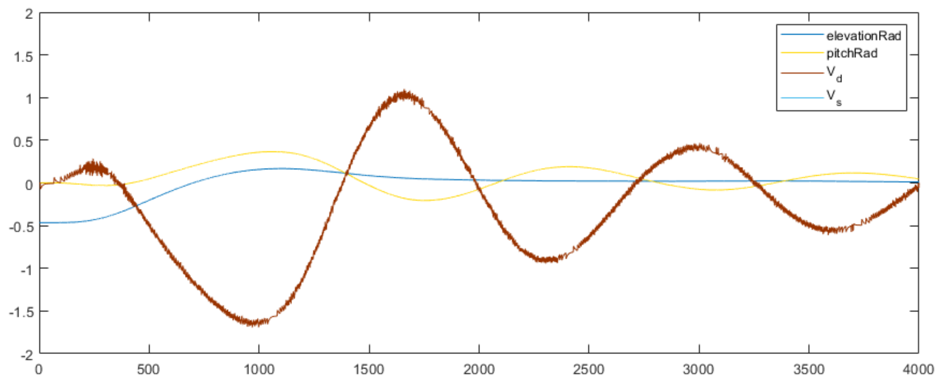

Figure 7: Critically damped response ($\lambda_1 = -10, \lambda_2 = -10$)

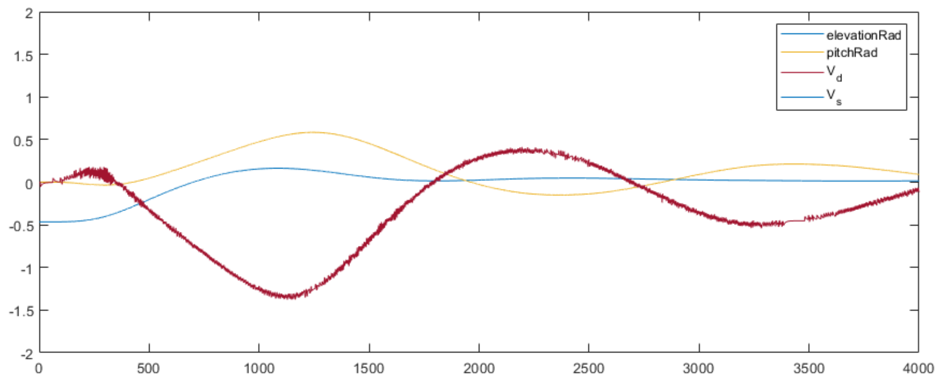

Figure 8: Overdamped response ($\lambda_1 = -5, \lambda_2 = -10$)

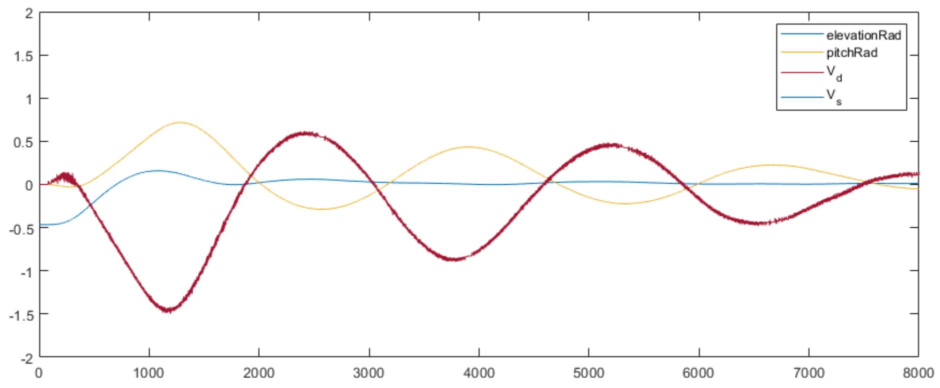

Figure 9: Underdamped response ($\lambda_1 = -6 + 3i, \lambda_2 = -6 + 3$)

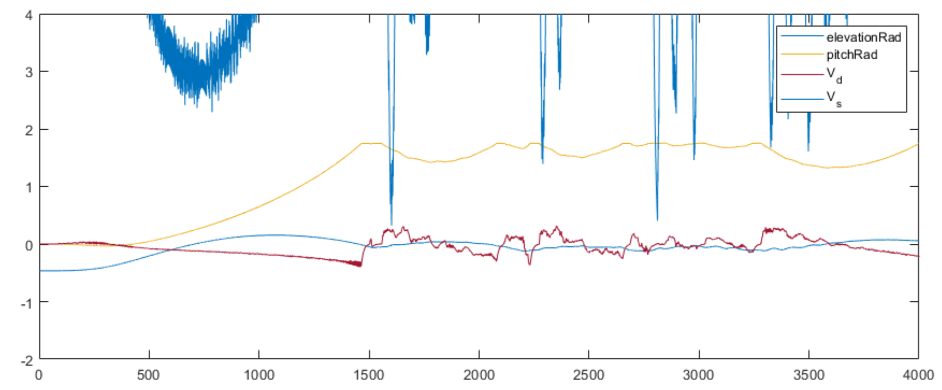

Figure 10: Marginally stable response ($\lambda_1 = 0, \lambda_2 = -5$)

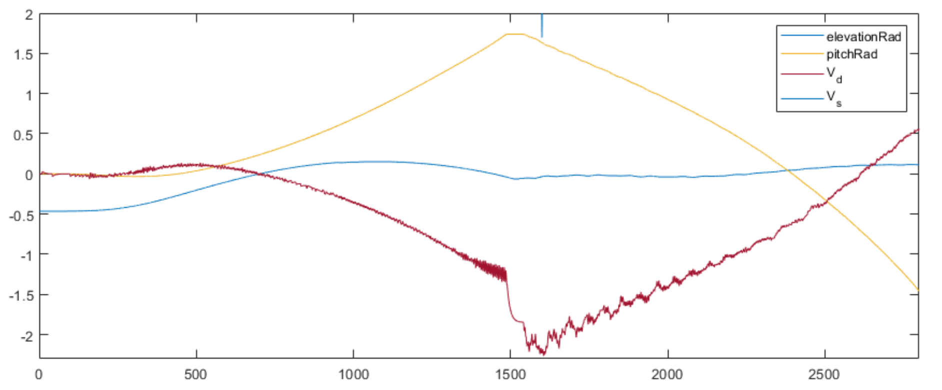

Figure 11: Unstable response ($\lambda_1 = 5, \lambda_2 = 5$)

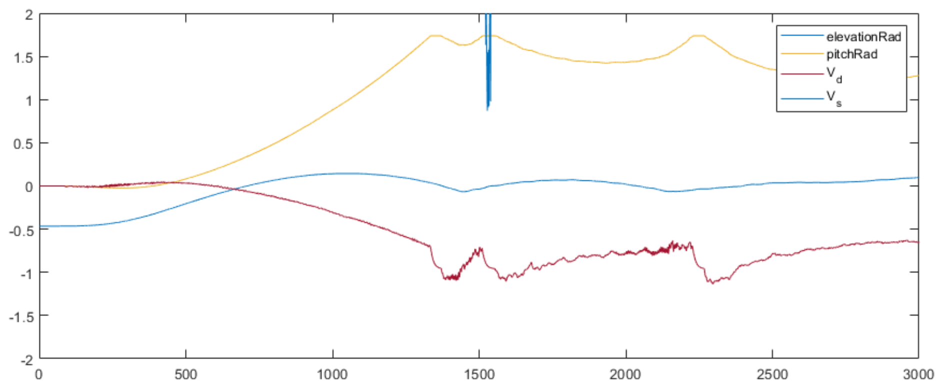

Figure 12: Unstable oscillating response ($\lambda_1 = 2 + 3i, \lambda_2 = 2 - 3i$)

2 Multivariable control

This part involves implementation of a state feedback controller to control the system multivariably. The feedback matrix is tuned through a Linear Quadratic Regulator (LQR), which uses a cost function to determine gains for the closed-loop controller. Weights are included based on which states that should receive priority, as well as how much the system is allowed to use its inputs (e.g. motor voltage). In addition, integral action was implemented to improve the performance of the feedback controlled system.

2.1 Theory

2.1.1 State space formulation

The linearized system of equations for pitch and elevation given in (8a-8b) are put in a state-space formulation on the form:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{16}$$

Where the state vector $\boldsymbol{x}$ and the input vector $\boldsymbol{u}$ are given by:

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} \quad \text{and} \quad \mathbf{u} = \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix}$$

The state-space formulation of the system then becomes:

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix} \tag{17}$$

2.1.2 Controllability

The controllability of the system may be examined by studying the rank of the controllability matrix for $n = 3$ dimensions.

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

The controllability matrix for the system is given by

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

The controllability matrix has full rank, and the system is therefore controllable.

2.1.3 Feedback and feedforward

The reference $\boldsymbol{r} = [p_c, \dot{e_c}]^T$ for the pitch angle p and elevation rate $\dot{e}$ is given by the joystick output. A state-feedback controller to follow this reference is to be implemented on the form

$$\boldsymbol{u} = \boldsymbol{F}\boldsymbol{r} - \boldsymbol{K}\boldsymbol{x} \tag{18}$$

By inserting the state-feedback controller with reference-feed forward $\boldsymbol{u}$ into the state-space equation $\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu}$, an expression for the matrix $\boldsymbol{F}$ as a function of the elements of matrix $\boldsymbol{K}$ may be derived.

$$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu} = \boldsymbol{Ax} + \boldsymbol{B}(\boldsymbol{Fr} - \boldsymbol{Kx}) \tag{19}$$

Where matrix $\boldsymbol{K}$ is on the form

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

When time goes to infinity, a theoretical assumption can be made that the states will go to their reference points and the system will stay at an equillibrium. Thus the matrix $\boldsymbol{F}$ can be found by analyzing the equation

$$(\boldsymbol{A} - \boldsymbol{BK})\boldsymbol{x_\infty} = -\boldsymbol{BFr} \tag{20}$$

The relation between $\boldsymbol{F}$ and elements of $\boldsymbol{K}$ is found to be

$$\boldsymbol{F} = \begin{bmatrix} 0 & k_{13} \\ k_{21} & 0 \end{bmatrix} \tag{21}$$

2.1.4 Linear Quadratic Regulator (LQR)

Through an LQR, the values of the $\boldsymbol{K}$ matrix will be chosen to minimize the cost function

$$J = \int_0^\infty \left(\mathbf{x}^\top(t)\boldsymbol{Q_{LQR}}\mathbf{x}(t) + \mathbf{u}^\top(t)\boldsymbol{R_{LQR}}\mathbf{u}(t)\right) dt \tag{22}$$

This function increases the further (and longer) the states and inputs stray from zero. If desired, different setpoints can be achieved by adding an offset to the state and input matrices. The matrix $\boldsymbol{Q_{LQR}}$ denotes the costs for state error and $\boldsymbol{R_{LQR}}$ denotes the costs for the input. The matrices penalize high state error and high inputs respectively. The following hypothesis is made on how different choices of $\boldsymbol{Q_{LQR}}$ and $\boldsymbol{R_{LQR}}$ affect the physical helicopter:

Hypothesis for choices of $\boldsymbol{Q_{LQR}}$ and $\boldsymbol{R_{LQR}}$

The general hypothesis is that increasing the weights in $\boldsymbol{Q_{LQR}}$ will result in a faster response as it penalizes high deviation from the setpoint. As such, higher values should be chosen for the states that need to be in equilibrium, or states that should converge faster.

Increasing the terms in $\boldsymbol{R_{LQR}}$ will result in less aggressive input voltages, as weighting $\tilde{V}_s$ or V_d should penalize high voltage, meaning energy consumption, to the propellers.

2.1.5 Integral action

An integral effect for the elevation rate and pitch angle is to be included in the controller. This is done by introducing two additional states γ and ζ, for which the differential equations are given by

$$\dot{\gamma} = p_c - p \tag{23}$$

$$\dot{\zeta} = \dot{e_c} - \dot{e} \tag{24}$$

The following augmented state vector will be used

$$\boldsymbol{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \\ \gamma \\ \zeta \end{bmatrix} \tag{25}$$

The system will be on the following form

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{G}\boldsymbol{r} \tag{26}$$

$$\boldsymbol{u} = \boldsymbol{F}\boldsymbol{r} - \boldsymbol{K}\boldsymbol{x} \tag{27}$$

The augmented $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{G}$ matrices are given by extending the state space with the new state vector:

$$\underbrace{\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \\ \dot{\gamma} \\ \dot{\zeta} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p \\ \dot{p} \\ \dot{e} \\ \gamma \\ \zeta \end{bmatrix}}_{\dot{\mathbf{x}}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix}}_{\dot{\mathbf{u}}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{G}} \underbrace{\begin{bmatrix} p_c \\ \dot{e_c} \end{bmatrix}}_{\dot{\mathbf{r}}} \tag{28}$$

Hypothesis on how the choices of the diagonal values in $\boldsymbol{Q_{LQR}}$ and $\boldsymbol{R_{LQR}}$.

Hypothesis for LQR with integral effect

Introducing an integral effect should reduce the steady-state error of the response. The gains for γ and ζ must however be balanced to avoid overshooting or slow responses. Too high feedback gain on γ or ζ will cause rapid changes and perhaps oscillatory behaviour or an unstable response. Additionally, the control effort penalty in $\boldsymbol{R_{LQR}}$ will determine how much effort is allowed to reduce these errors.

2.2 Results

2.2.1 LQR implementation and experiments

The controller was implemented by modeling the LQR controller (equation 19) as a subsystem in the simulink model.

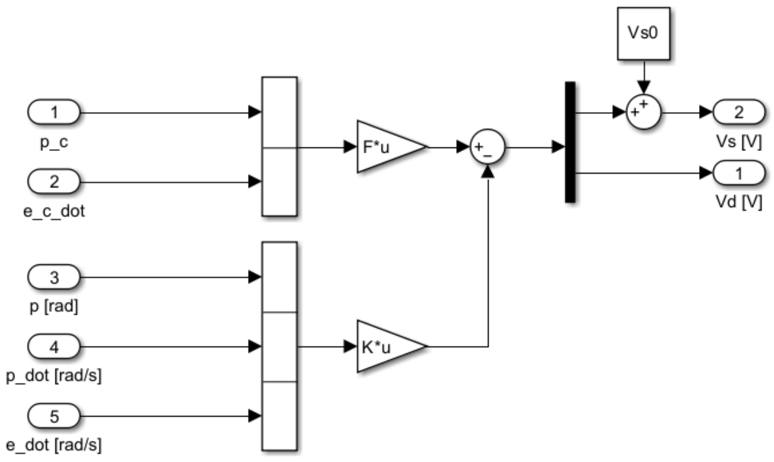

Figure 13: Simulink LQR subsystem

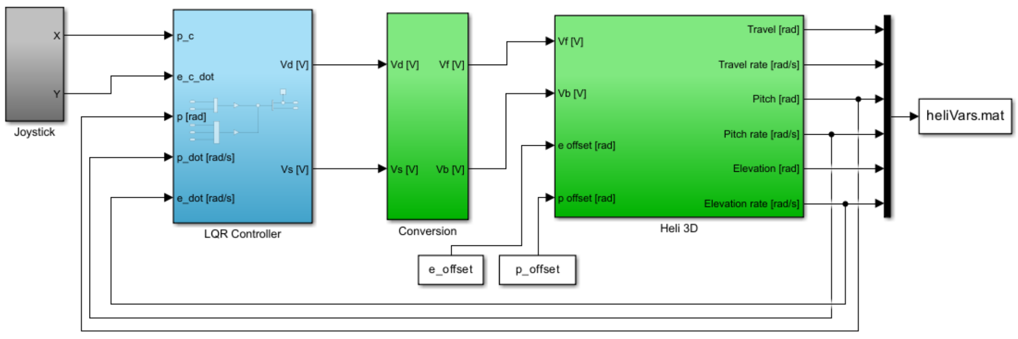

Figure 14: Updated simulink model for LQR control

Three sets of weights were tested for the original LQR controller. The testing strategy was based on quickly moving the joystick in both pitch and elevation axis to perturb the system, similar to an impulse response. The gain matrix **K** was calculated in MATLAB using the function *lqr(A,B,Q,R)*, and the values for Q and R were intialized using Bryson's rule [[3]]. The state and input weight matrices Q and R can be seen in the plots below, affecting the state vector **x** and input vector **u**:

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

2.2.1.1 Initial weights and step response

By weighting the pitch angle and the elevation angle's rate of change equally, overdamped behaviour was observed when starting from the ground. At around t=6000, a step input was applied to both states by tilting the joystick, to which the pitch angle responded with underdamped oscillations.

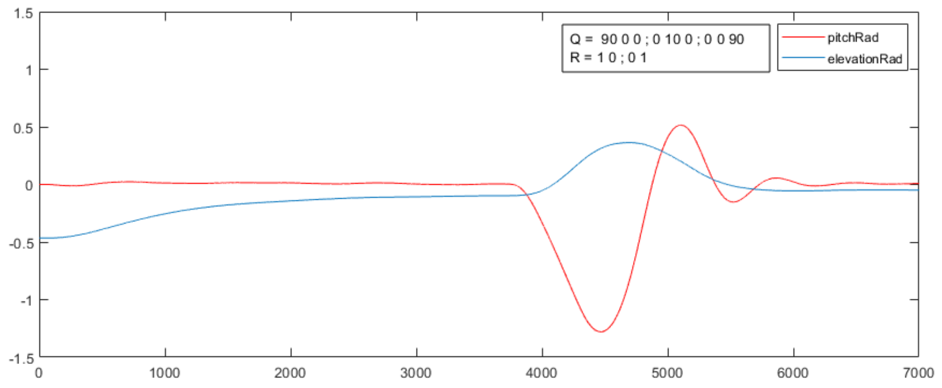

Figure 15: Plot of LQR performance and step response

2.2.1.2 Dampening step response by adjusting weights

To improve upon the pitch angle's oscillating response from the previous experiment, the pitch axis' rate of change was weighted higher. The result was a trade-off where the elevation angle deviated slightly more than before, but the pitch angle's response was much more damped. The angle deviation becomes less overall, is a result of the higher punishments of the deviation which results in a faster response, which verifies the hypothesis.

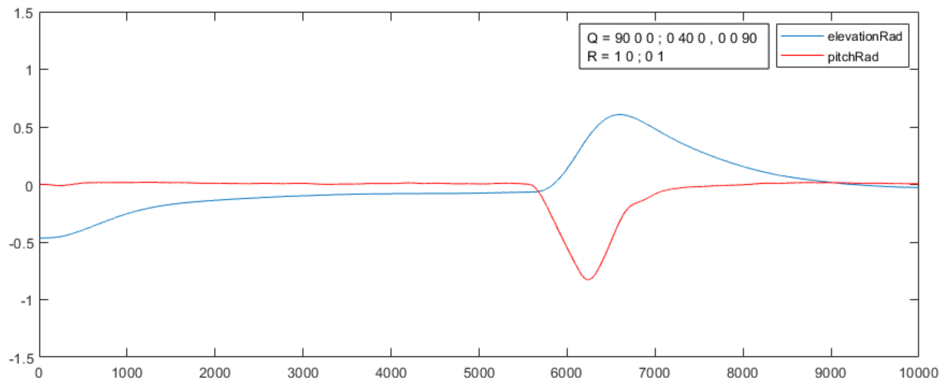

Figure 16: Plot of LQR performance and dampened step response

2.2.1.3 Effects of low weights on elevation

As a test, the weights were lowered for the elevation angle's rate of change. As expected, the elevation angle is now oscillating while the controller prioritizes keeping the pitch angle at its equilibrium.

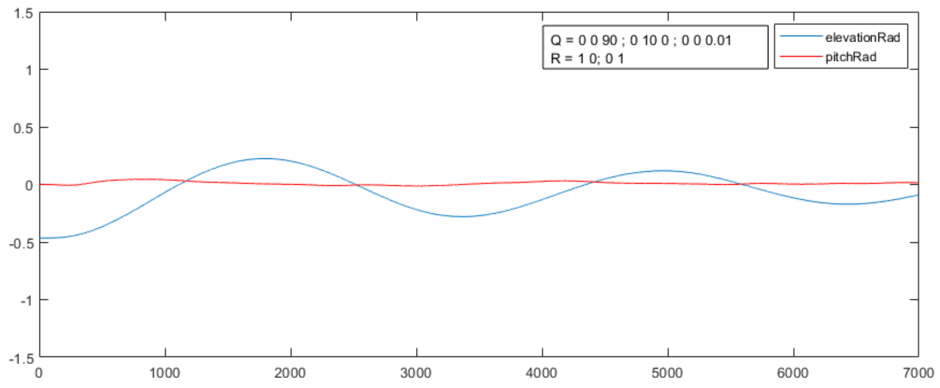

Figure 17: Plot of LQR performance and impulse response

2.2.2 Integral LQR implementation and experiments

As explained in section 2.1.5, integral action is added to the pitch angle and elevation rate by including two setpoint states, γ and ζ. The setpoints are controlled by the joystick's angles.

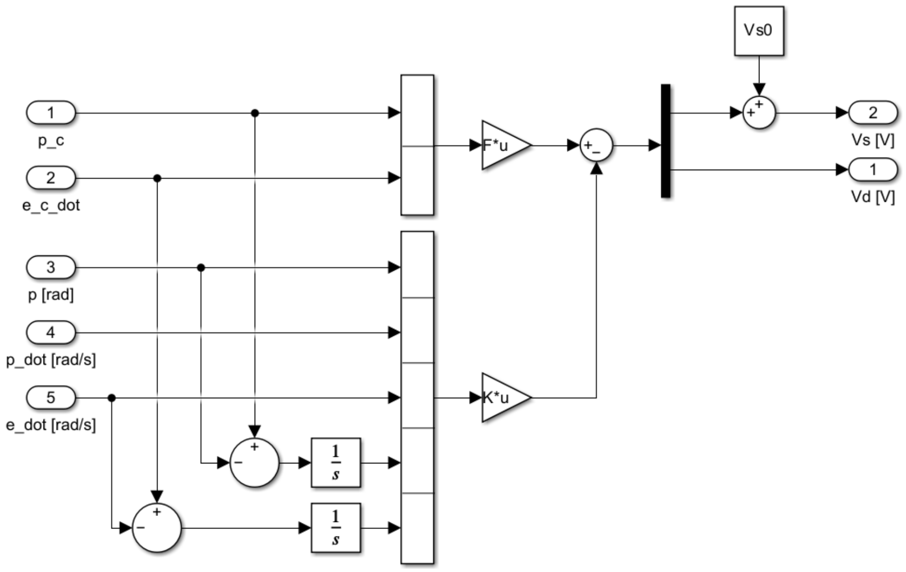

Figure 18: Updated simulink subsystem for integral action

2.2.2.1 Effect of new states

Elevation rate has been included in the plot to make it visible how it is affected by changing ζ with the joystick. After the system has lifted off the ground, the joystick was perturbed upwards and sideways simultaneously before letting it go. Since the LQR penalizes the rate of change in elevation, releasing the joystick makes it stay in place.

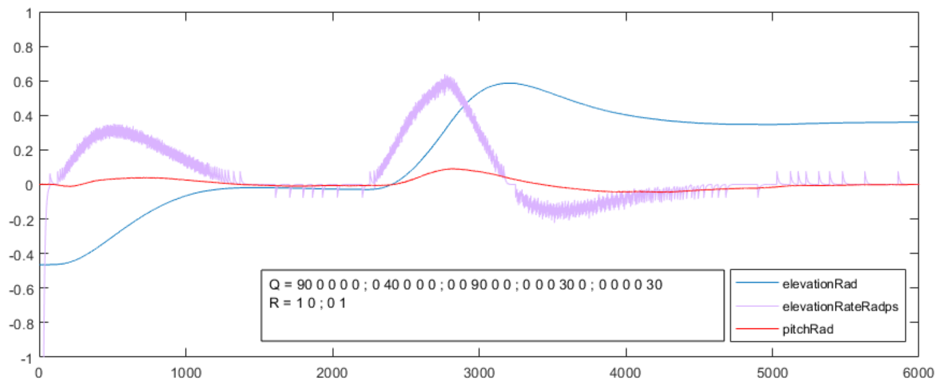

Figure 19: Plot of LQR performance with new states

2.2.2.2 Visible integral effect

To prove the point from the previous experiment: by running the same experiment with lowered weights for ζ, the controller steers the elevation angle back down to its equilibrium.

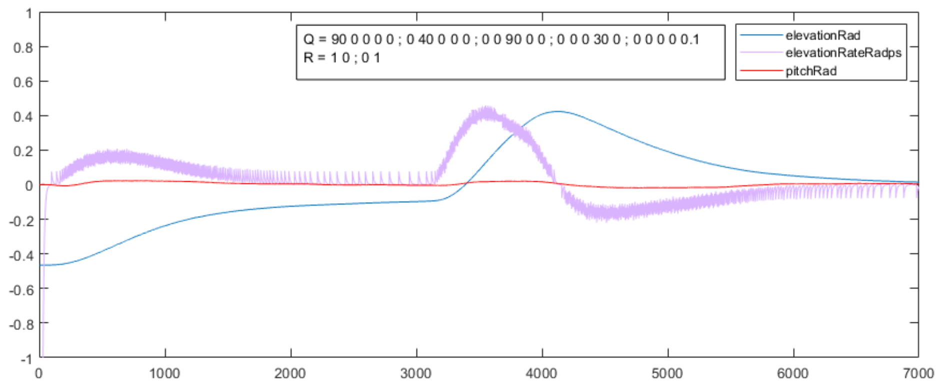

Figure 20: Plot of LQR performance with visible integral effect

2.2.2.3 Tuning for fast response

As a last experiment, the weights for the input voltages were reduced to allow for more effort from the propellers. An impressive liftoff was achieved without any overshoot. Thereafter, the elevation angle was shifted around using the joystick.

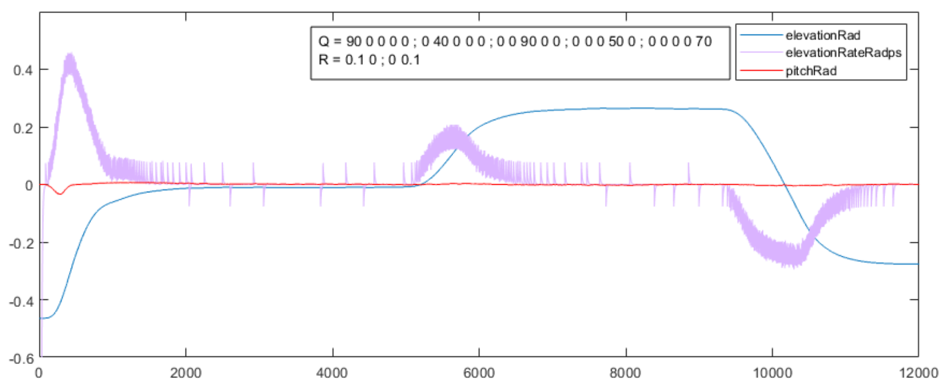

Figure 21: Plot of LQR performance with integral action, fast response

3 Luenberger observer

3.1 Theory

As the desire is to eventually have a free-flying helicopter, the encoder measurements will be replaced by an Inertial Measurement Unit (IMU). A problem with this is that the measurements are noisy, which can be solved by using a Luenberger observer for an observable LTI system. It will estimate the state values using both the model and measurements. For noisy signals, a well-tuned observer will give a more accurate estimate of the states than direct measurements.

3.1.1 Extended state-space formulation

Using the linearized equations of motion for the system given in (8), the extended state-space model can be formulated as shown in (35a)-(35b).

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \dot{e} \\ \ddot{e} \\ \ddot{\lambda} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ K_3 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \dot{\lambda} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \end{bmatrix}}_{\mathbf{B}} \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix} \tag{29a}$$

$$\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \dot{\lambda} \end{bmatrix} \tag{29b}$$

3.1.2 Observability

The minimum set of states required to be measured can be found by looking at the linearized equations of motions (8). Since p can be derived from the term for $\ddot{\lambda}$, only e and $\dot{\lambda}$ need to be measured to determine all states of the system. The reason why $\ddot{\lambda}$ must be measured and not p, is that going the other way around will require integration, which will yield unknown constants, therefore only differentiation will preserve all information.

In practice, however, the IMU will have highly frequent noise, making differation inaccurate. Therefore measuring all states is ideal to get the most accurate estimations.

3.1.3 Angle measurement

Since it was proved that it is not sufficient to simply use rate measurements given by the gyroscope, some angle measurements are required. The angle e and p can be found using the relation between forces of gravity and acceleration, and merging them by trigonometric relations as shown in (30) and (31).

$$e = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right) \tag{30}$$

$$p = \arctan\left(\frac{a_y}{a_z}\right) \tag{31}$$

3.1.4 State estimator

The states will be estimated by the linear observer shown in (32), where the observer gain L can be determined by pole placement.

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x}) \tag{32}$$

Hypothesis

Since the estimator combines the values of the measured states with the dynamics from the model, some noise is expected to disappear from the estimated states compared to their measurements.

As a rule of thumb, the poles of the observer should be several times faster (i.e. their real negative values should be 2-20 times greater) than in the original system to be able to keep up with its dynamics. However, if they are too fast the estimated states will inherit or even amplify noisy behavior from the original states. If the measurements are very noisy, fast poles with high oscillations are not desired, and poles with moderate real and imaginary magnitude relatively close to the system dynamics would make the noise less prevalent in the estimations.

Spreading the poles as a half circle in the left half plane with equal mutual distance should yield preferable behavior close to that of an underdamped system. Increasing the radius of the circle thus increasing the real part of the poles, should make the estimator faster. Increasing the angle between the poles, should lead to more overshoot [1].

The theoretical relationship between observability and pole placement is that observability ensures that the estimator poles can be freely placed to achieve a desired estimation performance. Lack of observability restricts this freedom, making it impossible to achieve fast or accurate state estimation for certain states.

3.2 Results

3.2.1 IMU characteristics

When moving the axes around manually, it was observed that the IMU's gyroscope measurements of were significantly more noisy than those of the encoders. Still, the gyroscope tracked the angular velocities reasonably well until the pitch angle was rotated by 90°. Because the gyroscope is mounted on the pitch axis, this caused the reference of the z and y-axes to get flipped such that the wrong axes become read for the elevation and travel rates. As such, the gyroscope measurements are not suitable to be used directly as state feedback for the system.

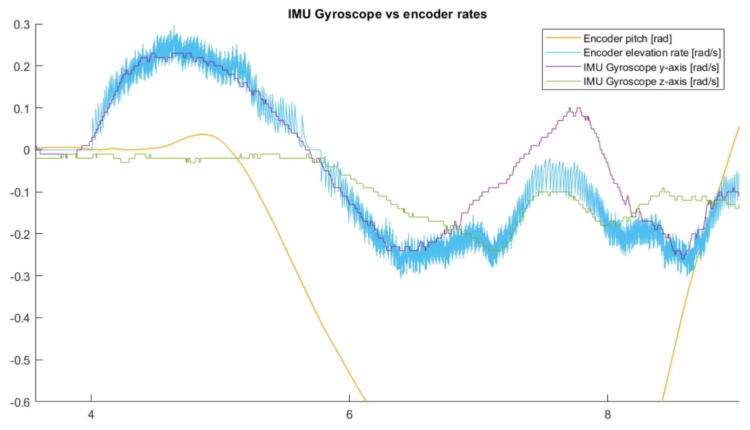

Figure 22: Plot showing the flipping of IMU gyroscope axes when pitch is rotated 90°

3.2.2 Transformations

To get correct measurements from the IMU, a function was provided for us that corrected the rates based on the measured angles. A subsystem was made in Simulink for converting the IMU's accelerometer data into pitch and elevation angles, according to equations 30 and 31. Functions were added to ensure that the states only get updated when a new measurement is taken by the IMU, and to avoid dividing by zero.

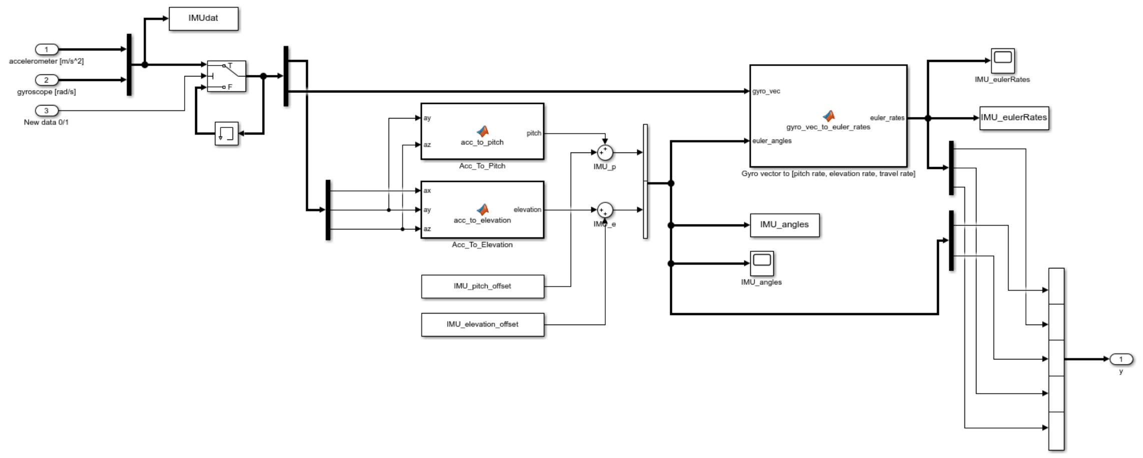

Figure 23: IMU Simulink subsystem

After implementing this subsystem the measured states from the IMU corresponded well with those of the encoders, albeit still more noisy. Note that the noise is not clearly visible in figure 24 since the motors are not running and thus not causing any vibrations. A static offset of around 0.5 radians was observed and compensated for in the elevation angle at this point.

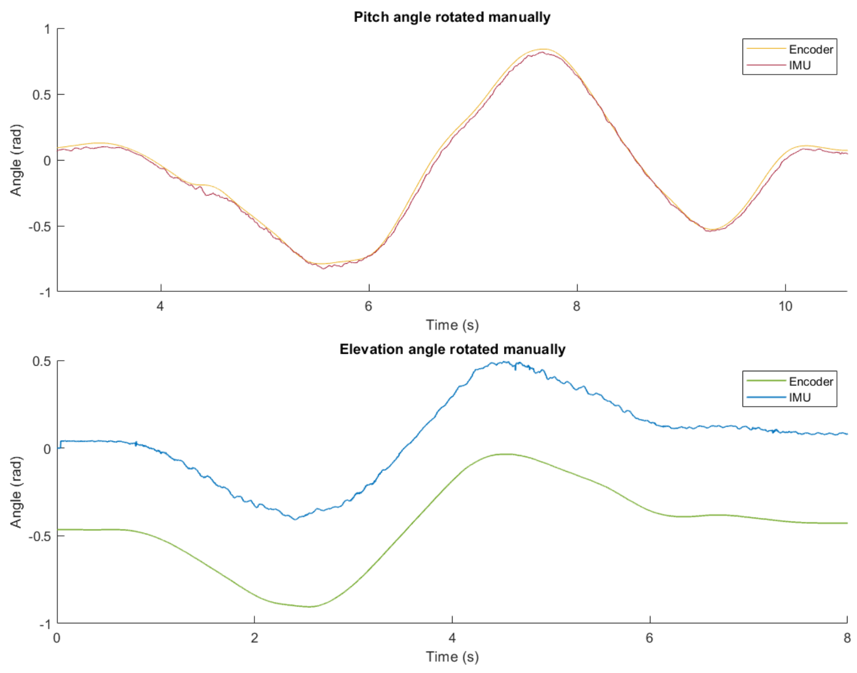

Figure 24: Comparison of calculated angles from IMU vs encoders

3.2.3 State estimator

The observer was implemented as a subsystem in Simulink, based on equation 32.

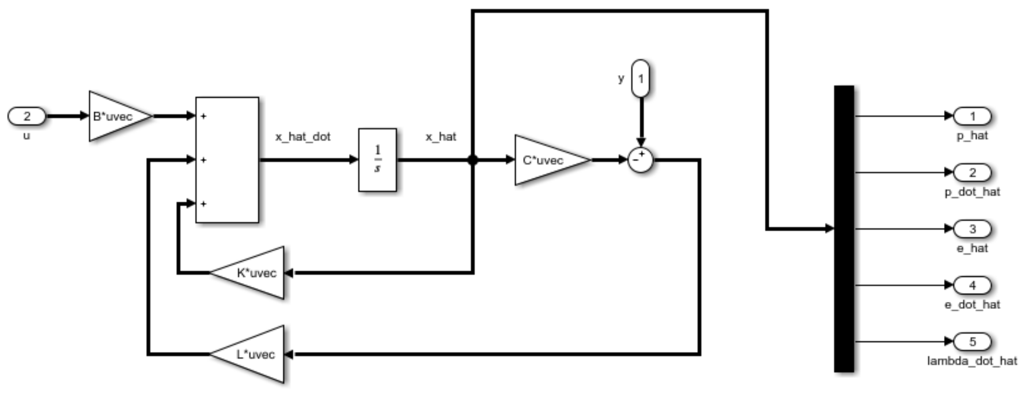

Figure 25: Luenberg observer in simulink

When placing the poles, the principle from the hypothesis was used. The following code was used to spread them out over a half-circle on the left-half plane, starting at 45° with a radius equal to 5 times the magnitude of the fastest pole from the original system (the LQR):

```
p_multiplier = 5;
p_radius = min(p_lqr) * p_multiplier
phi = pi/4; % Spread around +/- pi/4 radians
spread = -phi:(phi/(2)):phi;
p_est = p_radius*exp(1i*spread)
```

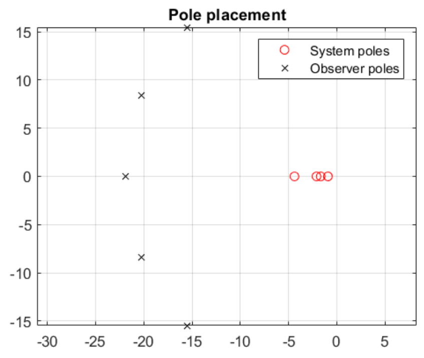

Figure 26: Luenberger observer poles

Attempts were made at placing the poles anywhere from 2-15 times that of the original system, while the angle of the half circle was moved between 0° to 60°. In the end, using a multiplier of 5 and an angle of 45° yielded the most decent compromise in accuracy between pitch and elevation. The noise from the IMU was mostly eliminated and the states tracked the general motion of the system, although with an overshoot significantly greater than shown by the encoders. The underdamped behavior predicted in the hypothesis was thus not achieved. Since the observer states are based on noisy differentiated signals, it is to be expected that the end result will differ to some degree from reality.

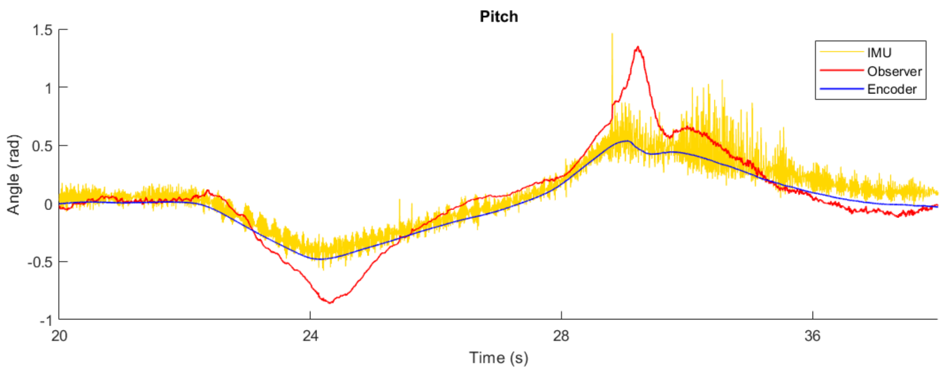

Figure 27: Comparison of observer vs encoder pitch

4 Kalman filter

4.1 Theory

In contrast to the Luenberger observer, the Kalman filter takes into account measurement noise when weighing the effect of the system model and measurements when estimating the states. Based on the assumption of the process noise, the Kalman filter will compute an optimal feedback gain for the estimator, to minimize the variance of the estimation error.

The equations for the Kalman filter are given a priori as (33) and a posteriori (34).

$$L[k] = \bar{P}[k]C_d^T(C_d\bar{P}[k]C_d^T + R_d)^{-1} \tag{33a}$$
$$\hat{x}[k] = \bar{x}[k] + L[k](y[k] - C_d\bar{x}[k]) \tag{33b}$$
$$\hat{P}[k] = (I - L[k]C_d)\bar{P}[k](I - L[k]C_d)^T + L[k]R_dL^T[k] \tag{33c}$$

$$\hat{x}[k+1] = A_d\hat{x}[k] + B_d u[k] \tag{34a}$$
$$\bar{P}[k+1] = A_d\hat{P}[k]A_d^T + Q_d \tag{34b}$$

Where R_d is the covariance matrix, and Q_d are the weights between measurement and system model trust.

4.1.1 Discretization

To use the discrete Kalman filter, the model needs to be discretized. When discretizing a model, sampling is used in the calculation. The sampling time used when running the estimator needs to be the same as the model is based on. To discretize the model, the function "c2d()" (Continuous to discreet) in Matlab is used. The continuous model from part 3 is extended to include λ.

$$\begin{bmatrix} \dot{p}[k+1] \\ \ddot{p}[k+1] \\ \dot{e}[k+1] \\ \ddot{e}[k+1] \\ \dot{\lambda}[k+1] \\ \ddot{\lambda}[k+1] \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0.002 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.002 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1.77\cdot 10^{-6} & 1.77\cdot 10^{-9} & 0 & 0 & 1 & 0.002 \\ 1.77\cdot 10^{-3} & 1.77\cdot 10^{-6} & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{A_d}} \begin{bmatrix} p[k] \\ \dot{p}[k] \\ \dot{e}[k] \\ \ddot{e}[k] \\ \lambda[k] \\ \dot{\lambda}[k] \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 1.99\cdot 10^{-6} \\ 0 & 1.99\cdot 10^{-3} \\ 2.11\cdot 10^{-6} & 0 \\ 2.11\cdot 10^{-3} & 0 \\ 0 & 5.86\cdot 10-13 \\ 0 & 1.17\cdot 10^{-9} \end{bmatrix}}_{\mathbf{B_d}} \begin{bmatrix} \tilde{V_s}[k] \\ V_d[k] \end{bmatrix} + \mathbf{w_d}[k] \tag{35a}$$

$$\mathbf{y}[k] = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{C_d}} \begin{bmatrix} p[k] \\ \dot{p}[k] \\ e[k] \\ \dot{e}[k] \\ \lambda[k] \\ \dot{\lambda}[k] \end{bmatrix} + \mathbf{v_d}[k] \tag{35b}$$

$$\mathbf{w_d} \sim \mathcal{N}(0, Q_d), \mathbf{v_d} \sim \mathcal{N}(0, R_d) \tag{35c}$$

4.1.2 Experimentation

Hypothesis for tuning of Q_d

By changing values for Q_d the Kalman filter can be tuned to fit the process. With high values for Q_d, the Kalman filter trusts the measurements more than the model. This may result in a noisy estimate as the noise from the measurement is included in the estimate. On the other hand, with small values for Q_d, the model is trusted more than the measurements. To small values, for Q_d will result in drift of the estimates as the estimates are not corrected sufficiently by the measured values. With balanced values for Q_d, notice is suppressed sufficiently and the estimated values are close to the actual values of the states.

Since the derivatives of the angles are more noisy, it makes sense to trust the measurements of the angles more than the angle rates.

4.2 Results

4.2.1 Noise estimate

When implementing the Kalman filter, an estimate of the measurement's covariance needs to be calculated. To obtain this, the states are sampled both when the helicopter is lying still on the floor and when it's flying steady at equilibrium.

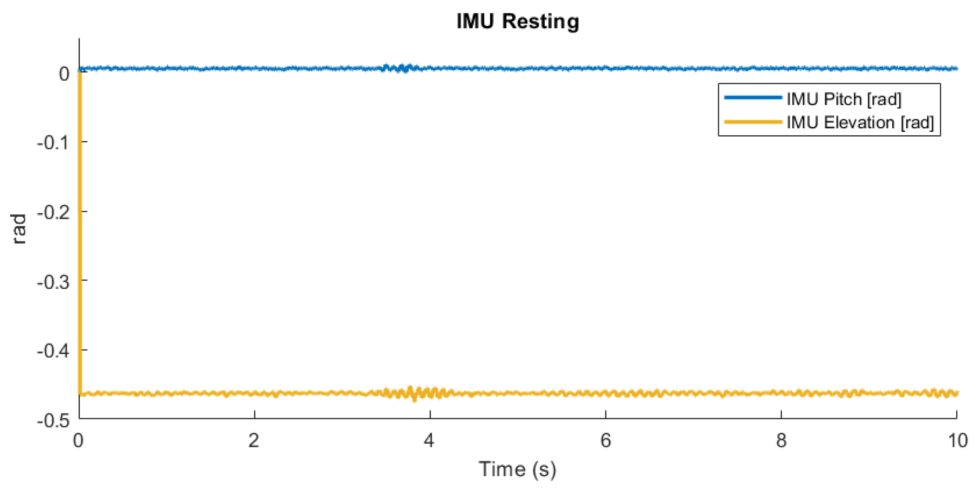

Figure 28: Pitch and elevation from IMU while helicopter lays still

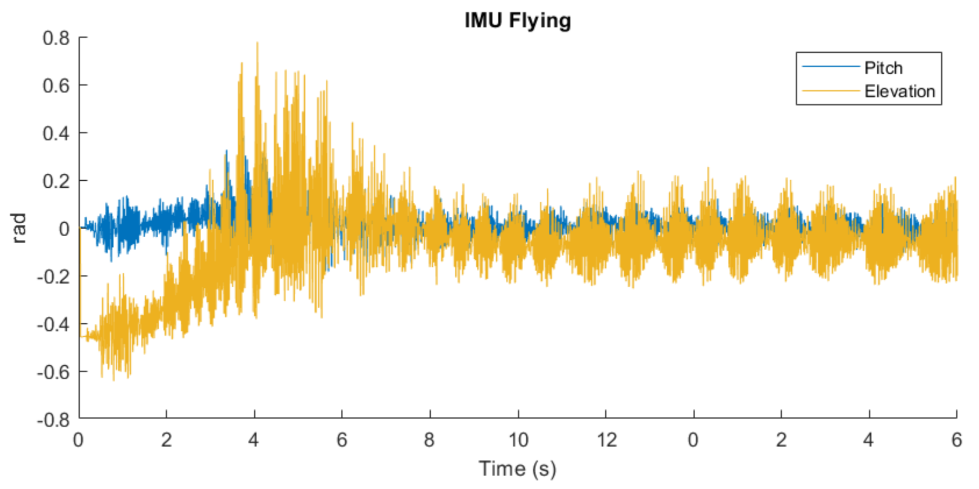

Figure 29: Pitch and elevation from IMU while helicopter flying

The covariance when the helicopter is flying is greater compared to when the helicopter is lying still. This is due to the measurements taken when the helicopter is lying still containing less measurement noise $\mathbf{v_d}$ as there is no elector magnetic inference (EMI) from the motors. When sampling during flight, noise from the motors is included. System noise such as vibration will also be included. However, measurements from when the helicopter was flying were used to calculate the covariance matrix $\mathbf{R_d}$ as it better represents the conditions the system will operate under. The covariance matrix $\mathbf{R_d}$ is calculated with the function "cov()".

$$\mathbf{R_d} = \begin{bmatrix} 3.2\cdot10^{-3} & 5.6\cdot10^{-4} & 5.6\cdot10^{-5} & -1.1\cdot10^{-3} & -9.8\cdot10^{-3} & -1.5\cdot10^{-3} \\ 5.6\cdot10^{-4} & 1.8\cdot10^{-3} & 2.4\cdot10^{-4} & -4.4\cdot10^{-4} & -5.1\cdot10^{-3} & -1.3\cdot10^{-3} \\ 5.6\cdot10^{-5} & 2.4\cdot10^{-4} & 3.0\cdot10^{-2} & -8.1\cdot10^{-3} & 2.7\cdot10^{-2} & 5.4\cdot10^{-3} \\ -1.1\cdot10^{-3} & -4.4\cdot10^{-4} & -8.1\cdot10^{-3} & 1.7\cdot10^{-2} & -3.0\cdot10^{-2} & -4.8\cdot10^{-3} \\ -9.8\cdot10^{-3} & -5.1\cdot10^{-3} & 2.7\cdot10^{-2} & -3.0\cdot10^{-2} & 7.1\cdot10^{-1} & 9.3\cdot10^{-2} \\ -1.5\cdot10^{-3} & -1.3\cdot10^{-3} & 5.4\cdot10^{-3} & -4.8\cdot10^{-3} & 9.3\cdot10^{-2} & 1.4\cdot10^{-2} \end{bmatrix} \tag{36}$$

4.2.2 Implementation

When implementing the Kalman filter two Matlab functions were made, One for calculating x-priori (priori) and one for calculating x-posteriori (posteriori). The function for x-prioroi is an implementation of equations 33 and was made to update every cycle of the Simulink. The x-posteriori function is implemented to update x-posteriori with equations 34 when measurements are received from the IMU. In between, the x-posterior is set to x-priori.

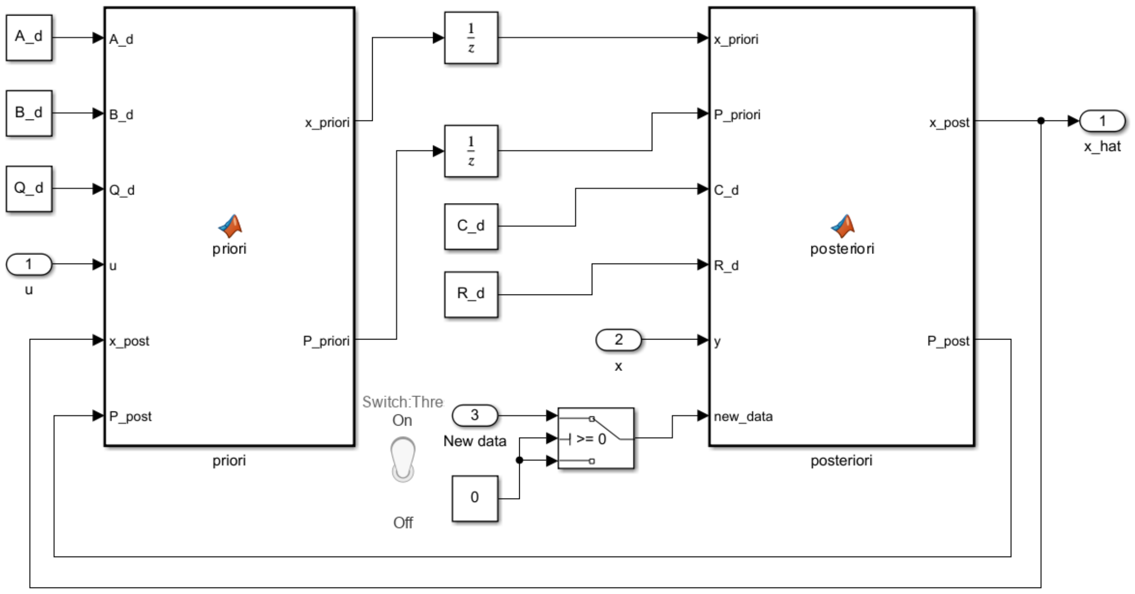

Figure 30: Simulink model of Kalman filter

4.2.3 Experimentation

$$Q_d = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 \end{bmatrix} Q_f \tag{37}$$

For experimentation and testing of the Kalman filter, the same $\mathbf{Q_d}$ matrix was used and only the factor Q_f was changed. Higher weights were chosen for measured angles than angular velocities because those measurements were less noisy. With low values of Q_f, the estimate was smooth but was not correct compared with the measured value. By experimenting with different values for Q_f, a balanced Kalman that both reduced the noise and estimated the states correctly was found.

If the update of new data is disabled, the Kalman filter only uses the priori calculation to estimate the states without being corrected by the measurements. This can be seen in figure 33.

The estimates were not able to give a good estimate and the states were drifting compared to the measured value. The curves are completely smooth as the model has no measurement noise or system noise.

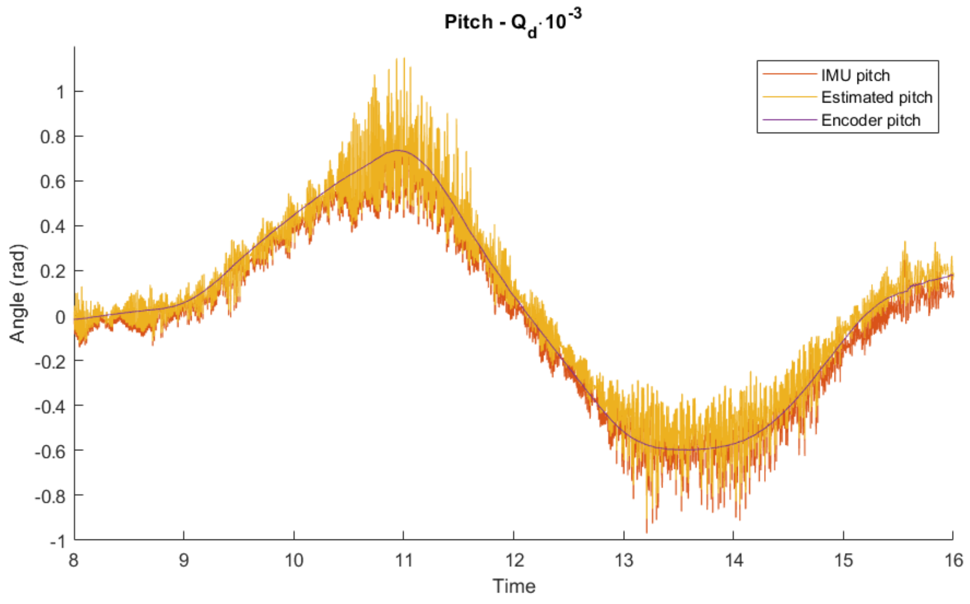

Figure 31: Kalman filter with $Q_f = 10^{-3}$

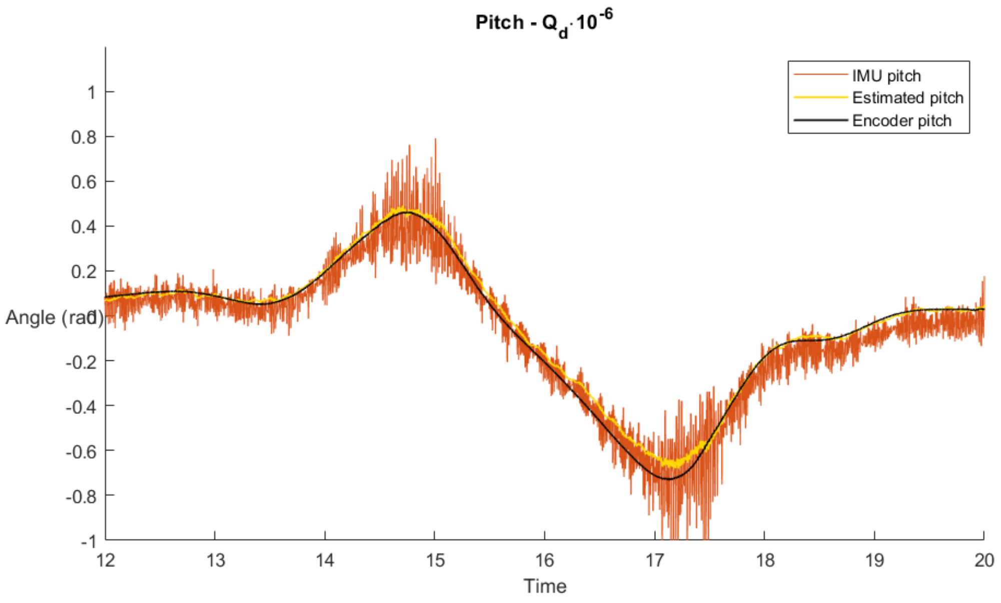

Figure 32: Kalman filter with $Q_f = 10^{-6}$

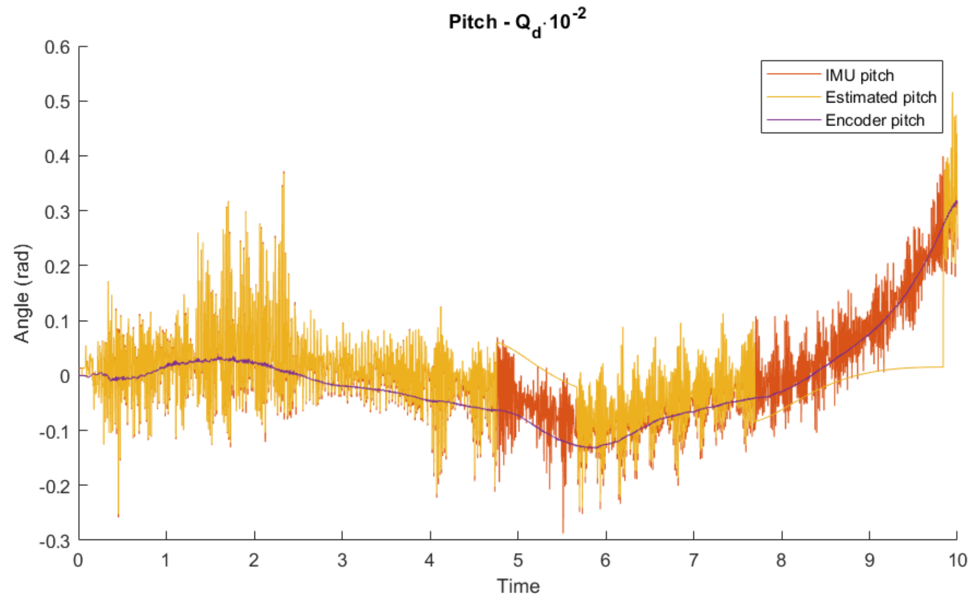

Figure 33: Kalman filter with $Q_f = 10^{-2}$ and IMU sampling toggled off/on

4.3 Comparison of Kalman filter and Luenberger observer

The Luenberger observer and the Kalman filter are both state estimation methods. The Luenberger observer depends on pole placement for the estimator, while the Kalman filter depends on pole placement on the error covariance matrix. The main difference is that the Kalman filter accounts for process and measurement noise. It is optimal in the sense of minimizing the mean square estimation error, assuming the noise is Gaussian. The Luenberger observer on the other hand, assumes that the model dynamics and measurements are accurate and only attempts to correct for small deviations due to initial estimation errors.

Gain L is typically set manually or using pole placement methods to achieve desired convergence rates, often without considering the statistical properties of noise.

The Kalmain Gain K on the other hand is computed through a recursive process which balances estimation quality and noise robustness at each time step. This gain adapts based on noise levels and system dynamics, making the Kalman filter optimal for minimizing estimation error in the presence of noise.

As seen in this report, the Kalman filter gives significantly better results as it accounts for measurement noise when picking the right Q_d values. The Luenberger observer yields less accurate estimations as the observer states are based on noisy differentiated signals, without doing stochastic computations to account for this noise.

References

[1] Morten Omholt Alver. *Lecture 6 State estimation*. NTNU Department of engineering cybernetics, 2024.

[2] NTNU Department of Engineering Cybernetics. 'Helicopter lab preparation'. In: *TTK4115 - Linear System Theory* 5.5 (2024).

[3] Mathworks. *Linear-Quadratic Regulator (LQR) design*. URL: https://se.mathworks.com/help/control/ref/lti.lqr.html (visited on 15th Sept. 2020).