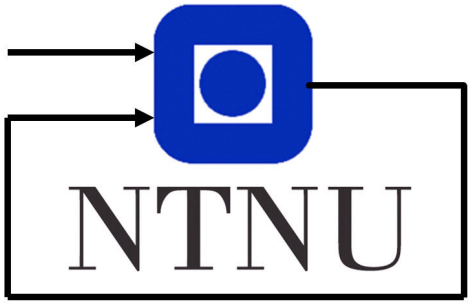
Dynamics and control of helicopter model

Group 95
Andreas Tufte 498284
Martin Kraft 507680

Printed November 18, 2020



Department of Engineering Cybernetics

Abstract

Control theory deals with control of dynamical systems in engineered processes and machines. The dynamics of helicopters are due non-linearities and complicated models. However, by linearization and applying linear system theory, it is possible to obtain suitable control.

This report covers practical experience in lab work on a helicopter model in the course TTK4115 Linear System Theory. The course is a part of late undergraduate or early graduate levels in becoming a control engineer. As a result, the report is targeted towards examiners; organized and discusses related control theory problems consecutively in the same manner as the lab manual.

Contents

1 Introduction

In this lab, the dynamics of a helicopter model is linearized to apply linear system theory. This enables the possibility to implement controllers from state feedback, feedforward and also integral controllers. First, we look at the mathematical description of the system, introduces constants and explain how we determined to find these. Secondly, we implement three types of controllers and tune these by methods such as pole placement and cost-specification. To the end, we look at two different types of state estimators; a Luenberger observer and the Kalman filter. We are using helicopter numbered 1 at the lab.

2 Theory and method

The helicopter model is presumed known, but an illustration is found in figure 1. By the holonomic constraints, the general coordinates describing the system (helicopter position) are angles for a travel λ, elevation e and pitch p. The dynamics of the system needs the derivatives also, correspondingly travel rate $\dot{\lambda}$, elevation rate $\dot{e}$ and pitch rate p. This information is summarized in a state-space

$$\mathbf{x} := \begin{bmatrix} p & \dot{p} & e & \dot{e} & \lambda & \dot{\lambda} \end{bmatrix}^\top . \tag{1}$$

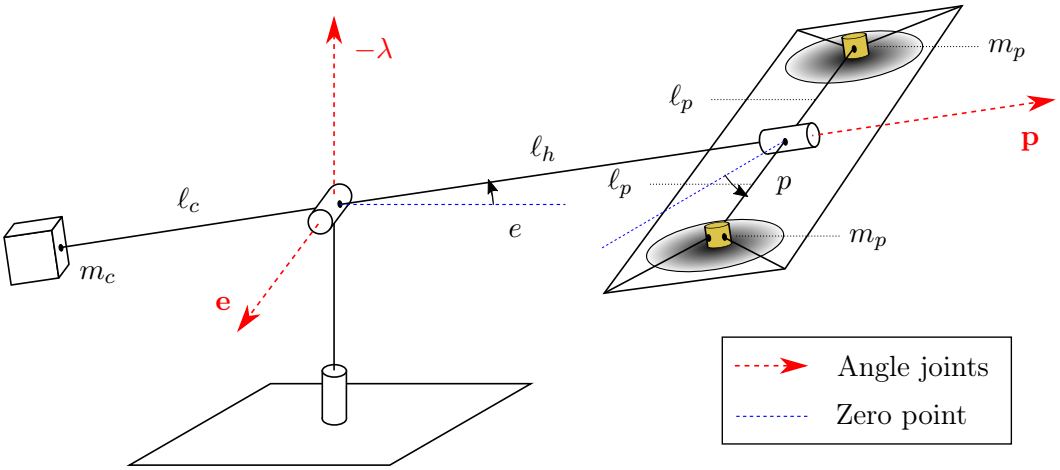


Figure 1: Helicopter model with angle directions as given in red. Gravitational forces are not shown. Masses and lengths are defined in the figure.

2.1 Theory

The propellers act with a force proportional to applied motor voltages. Denoting subscript f for front motor (helicopter points in positive λ) and b

for back motor, the propeller forces are $F_f = K_f V_f$ and $F_b = K_f V_b$. These acts orthogonal to pitch angle. In the following, we let the true input to our system by determined by the sum of the motor voltages V_s and their difference V_d such that

$$V_s := V_b + V_f, \qquad V_d := V_b - V_f. \tag{2}$$

From a linearized operating point, we work with an input vector as the deviation from the true input vector as

$$\mathbf{u} := \begin{bmatrix} \tilde{V}_s \\ \tilde{V}_d \end{bmatrix} = \begin{bmatrix} \tilde{V}_s - V_{s,0} \\ \tilde{V}_d - V_{d,0} \end{bmatrix}, \tag{3}$$

where tilde denotes deviation from linearized point with subscript 0.

Neglecting friction and centripetal forces, the dynamics is found by Newton's second law for rotation in each angle direction[1]. This yields

$$\begin{aligned} J_p \ddot{p} &= L_1 V_d, \\ J_e \ddot{e} &= L_2 \cos(e) + L_3 V_s \cos(p), \\ J_\lambda \ddot{\lambda} &= L_4 V_s \sin(p) \cos(e), \end{aligned} \tag{4}$$

with constants supplied in table 1.

Table 1: Constants in dynamics.

Constant	Parameters	Units
L_1	$\ell_p K_f$	$\mathrm{NV}^{-1}\mathrm{m}$
L_2	$\ell_c F_{g,c} - 2\ell_h F_{g,f}$	Nm
L_3	$\ell_h K_f$	$\mathrm{NV}^{-1}\mathrm{m}$
L_4	$\ell_h K_f$	$\mathrm{NV}^{-1}\mathrm{m}$

The dynamics of the system in (4) is non-linear. To apply linear system theory, the system is linearized around $p, e, \lambda = 0$. An equilibrium (rates zero) at this point can be achieved by a constant voltage sum $V_{s,0}$. Note that we do not need a constant voltage difference for equilibrium, such that $V_{d,0} = 0$. This is implied by the first equation in (4) which means that we can write $V_d \equiv \tilde{V}_d$.

The constant voltage sum is found from the second equation in (4) by setting $e = p = 0$. Combined, the voltages in equilibrium are

$$\mathbf{u}_0 = \begin{bmatrix} V_{s,0} \\ V_{d,0} \end{bmatrix} = \begin{bmatrix} -\frac{L_2}{L_3} \\ 0 \end{bmatrix}. \tag{5}$$

At the linearized point, the deviation or perturbed states in (1) equals the state space itself. If we let $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, then the first order taylor-expansion

[1] As the derivation was a part of the lab preparations, it is not included

yields the linearized model as

$$\dot{\mathbf{x}} = \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_0 \mathbf{x} + \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_0 \mathbf{u}, \tag{6}$$

where $(\,\cdot\,)|_0$ denotes evaluation of the jacobians in equilibrium. Applying (6) by the relations in (4), the full state equation $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ is

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \dot{e} \\ \ddot{e} \\ \dot{\lambda} \\ \ddot{\lambda} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ K_3 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \lambda \\ \dot{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix}, \tag{7}$$

where the constants K_1,K_2 and K_3 is supplied in table 2. Another representation of (7) of interest, is

$$\begin{aligned} \ddot{p} &= K_1 V_d, \\ \ddot{e} &= K_2 \tilde{V}_s, \\ \dddot{\lambda} &= K_3 p. \end{aligned} \tag{8}$$

Table 2: Constants in the state space formulation.

Constant	Parameters	Units
K_1	$\frac{\ell_p K_f}{2m_p \ell_p^2}$	$\mathrm{V}^{-1}\mathrm{s}^{-2}$
K_2	$\frac{\ell_h K_f}{m_c \ell_c^2 + 2m_p \ell_h^2}$	$\mathrm{V}^{-1}\mathrm{s}^{-2}$
K_3	$-\frac{\ell_c m_c g - 2\ell_h m_p g}{m_c \ell_c^2 + 2m_p(\ell_h^2 + \ell_p^2)}$	s^{-2}

By the linearized dynamics, according to (8), travel acceleration is proportional to pitch angle. However, this not true in general. If the rotors do not apply any forces (i.e. only gravitational forces apply), it is possible to swing the helicopter around λ-axis with centripetal forces keeping the helicopter in a somewhat stationary orbit with $\lambda \simeq$ const. This contradicts any acceleration of λ posed by any $p \neq 0$.

This sums up the theory needed to implement the controllers and look at the behavior of the system. Further theory is supplied when it is relevant to the discussion.

2.2 Method

In this lab, MATLAB® and Simulink® software was used to implement controllers and look at the measurements. We have also used block diagrams in our understanding and to develop the Simulink implementations. Saved files were shared using version control on Github.

3 Monovariable control

At first, we tried to control the helicopter by feedforward of the voltage difference V_d and the voltage sum V_s with a joystick. This was hard, and motivates the implementation of suitable controllers.

In order to control pitch and elevation rate, it is possible to introduce a monovariable controller for each. In this setup, the elevation rate is controlled by the voltage sum V_s and pitch is controlled by the voltage difference V_d. This is possible due to (p, V_d) and (e, V_s) being decoupled in (8).

3.1 Parameter identification

Before continuing with implementation of the controllers, we checked the encoder measurements and made sure that all encoders were correct at the linearized operating point. All encoder values are set to zero each time Simulink is connected to the helicopter. By starting the helicopter at rest on the table, all readings except the elevation is correct. This is underlying the assumption that the helicopter pitch angle starts at $p = 0$.

An offset for the elevation was found by scoping the elevation e, and lifting the helicopter to horizontal position. The elevation offset was found to be $e_{\text{off}} = 0.415$ rad $\simeq 24$ deg, and corrected.

The voltage sum $V_{s,0}$ was found by holding the helicopter in horizontal position and applying enough voltage to see that the helicopter could stand at $e = 0$ on its own. This constant was found by scoping the voltage sum as $V_{s,0} = 5.48$ V.

The final constant to be decided before any implementations, was the motor force constant K_f. By equation (5), $V_{s,0} = -\frac{L_2}{L_3}$ and values in table 1, the constant is

$$K_f = -\frac{L_2}{\ell_h V_{s,0}} = -\frac{(\ell_c F_{g,c} - \ell_h 2F_{g,f})}{\ell_h V_{s,0}} = 0.244 \text{ NV}^{-1}.$$

All information regarding the setup has been found at this point, and the experimental values are summarized in table 3. Note that we are not currently dealing with any uncertainties in our model or parameters. Small uncertainties in our constants do not affect the controllers as much.

Table 3: Experimental constant values on helicopter 1.

Constant	Value	Units
$-e_{\text{off}}$	0.415	rad
$V_{s,0}$	5.48	V
K_f	0.244	NV^{-1}

3.2 PD control pitch

A PID controller was given for the elevation rate, and we implemented the PD controller for pitch by

$$V_d = K_{pp}(p_c - p) - K_{pd}\dot{p}, \tag{9}$$

which, by $\ddot{p} = K_1 V_d$ in (8), equals to

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

In order to determine the constants K_{pp} and K_{pd}, pole placement was used. The poles of the transfer function from p_c to p, $G(s) = \frac{p}{p_c}(s)$, is found by the Laplace transform of the above. This yields

$$s^2 p(s) = K_1 K_{pp} p_c(s) - K_1 K_{pp} p(s) - s K_{pd} p(s),$$

which, on the form of $G(s)$, is

$$G(s) = \frac{p}{p_c}(s) = \frac{K_1 K_{pp}}{s^2 + K_1 K_{pd} s + K_1 K_{pp}}. \tag{10}$$

A general denominator for two poles is

$$(s - \lambda_1)(s - \lambda_2) = s^2 - (\lambda_1 + \lambda_2)s + \lambda_1 \lambda_2.$$

By directly comparing to the denominator of (10), the coefficients are

$$K_{pd} = \frac{-(\lambda_1 + \lambda_2)}{K_1}, \qquad K_{pp} = \frac{\lambda_1 \lambda_2}{K_1}. \tag{11}$$

The implementation of (9) in Simulink by pole placement in (11) is showned in figure 2 and figure 3.

λ_1 and λ_2 is chosen to get desired behavior of the pitch. The poles should be chosen to control the pitch rapidly without excessive oscillations.

In the beginning of the pole placements, we knew from previous courses in control theory that $\mathrm{Re}\{\lambda\} < 0$ should be chosen to make the impulse response stable and that the poles should appear in complex conjugate pairs if we choose imaginary parts. This is also due to the fact that the coefficients K_{pp} and K_{pd} should be real. Whenever $\lambda_2 = \lambda_1^*$, the sum and the multiplication in (11) yields real values.

As we controlled the pitch by a joystick, we are actually controlling a servomechanical system. This was new to us, and made the evaluation of the pole placement somewhat subjective and difficult. We thought we had found a nice set of poles with $\lambda_1 = -5 + 3i$ and $\lambda_2 = -5 - 3i$. This made the helicopter stable for a fast change in pitch reference. However, when further investigating the response through a step response as showned in figure 4, we ended up with the placements of $\lambda_1 = -5 + 1.5i$ and $\lambda_2 = -5 - 1.5i$. The

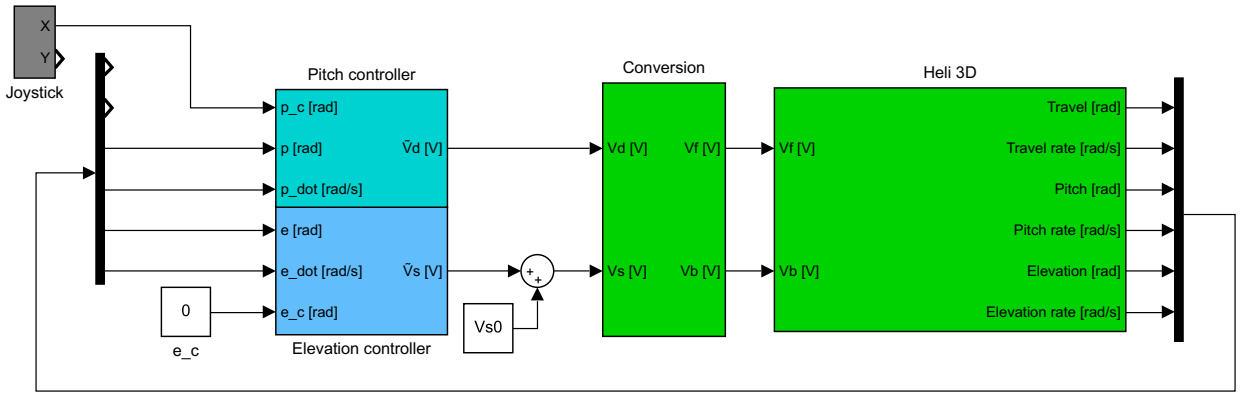


Figure 2: Simulink diagram for monovariable control in day 1.

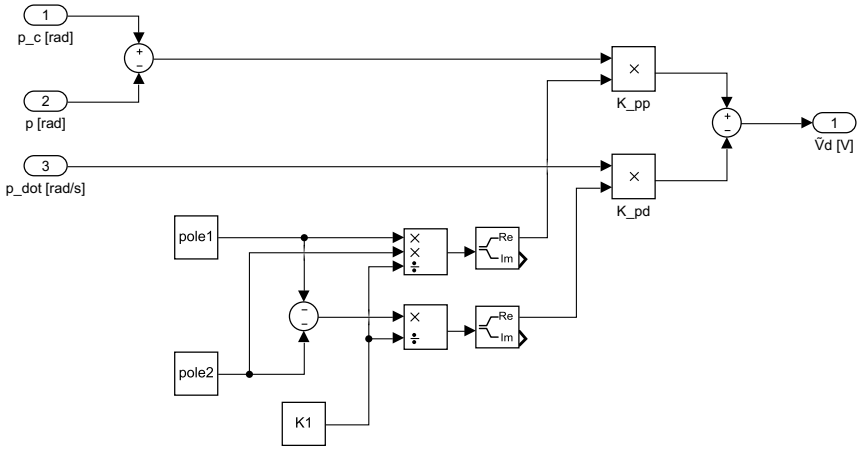


Figure 3: Pitch controller implementation by equation (9). The constant `K1` and poles `pole1`, `pole2` are fetched by Matlab script.

change of the imaginary parts was halved from first testing the helicopter servomechanically.

A general step response from one reference level to another, is proportional to the step difference and by $\sim e^{\lambda t}$. This is a decaying exponential on the form $\sim e^{-\alpha t}e^{\pm i\beta}$ when we write $\lambda_{1,2} = -\alpha \pm i\beta$. We see that choosing poles further into the left half plane, increases α. This makes the time constant of our system faster. Likewise for the imaginary part, reducing β makes the system oscillating less.

A result of reducing the imaginary part of our pole placement, is that we are keeping the fast response, and the oscillatory part should decrease. This is seen in figure 4.

Another comparison to our first choice is that the response of $-5 + 3i$ over shoots when first approaching the reference p_c. Interestingly, an over

shoot also occured when we tried to make the response faster as we switched to $-7 + 1.5i$. Comparing to our choice, it seemed like the response were as fast in the beginning, but differed the response with $-7 + 1.5i$ failed to slow down. When also trying $-3 + 1.5i$, we ended up with a slower response, but this had little oscillations.

The reason for the response having a pretty similar rate of change of pitch, might be due to saturations on the system inputs or physical inertia. If we are concerned with keeping a smooth helicopter, one might turn to the poles of $-3 + 1.5i$.

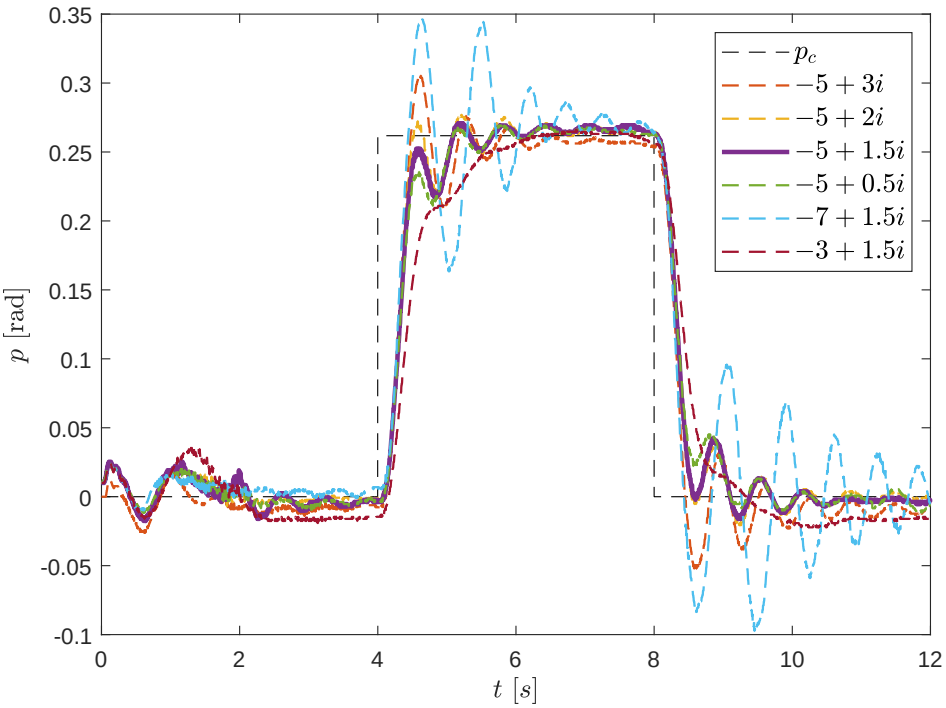


Figure 4: Step response by pole placement to pitch PD controller. Highlighted the selected pole.

4 Multivariable control

In this part, we want to control pitch and elevation rate as in the last section, but with a multivariable controller instead. This enables feedback of the states p, $\dot{p}$ and $\dot{e}$. Firstly, we write the state-space form $\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$ with state vector and input vector are

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} \qquad \text{and} \qquad \mathbf{u} = \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix}, \tag{12}$$

the expression is

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix}}_{\mathbf{B}} \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix}. \tag{13}$$

We now wish to implement a state-feedback controller with reference-feed-forward on the form

$$\mathbf{u} = \mathbf{Fr} - \mathbf{Kx}, \tag{14}$$

where

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \end{bmatrix}. \tag{15}$$

There are several methods to find a $\mathbf{K}$-matrix. In this section we will use pole placement and the LQR-method.

We want $\mathbf{y(t)} \rightarrow \mathbf{r}$ when t $\rightarrow \infty$ If we substitute (14) in for $\mathbf{u}$ in (**??**) and assume steady state ($\dot{\mathbf{x}} = \mathbf{0}$), we end up with

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{B}(\mathbf{Fr} - \mathbf{Kx}) = \mathbf{0}(\mathbf{A} - \mathbf{BK})\mathbf{x}_\infty = -\mathbf{BFry}_\infty = \mathbf{C}(\mathbf{A} - \mathbf{BK})^{-1}\mathbf{BFr} \tag{16}$$

$$\implies \mathbf{F} = \mathbf{C}[(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{B}]^{-1} \tag{17}$$

We may now derive the $\mathbf{F}$-matrix as a function of $\mathbf{K}$.

This gives the matrix $\mathbf{F}$ as

$$\mathbf{F} = [\mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{B}]^{-1} = \begin{bmatrix} k_{11} & k_{13} \\ k_{21} & k_{23} \end{bmatrix}, \tag{18}$$

4.1 Implementation

Figure 5 shows the block diagram of the system with state-feedback and reference-feed-forward. We implemented it in Simulink as shown in figure 6.

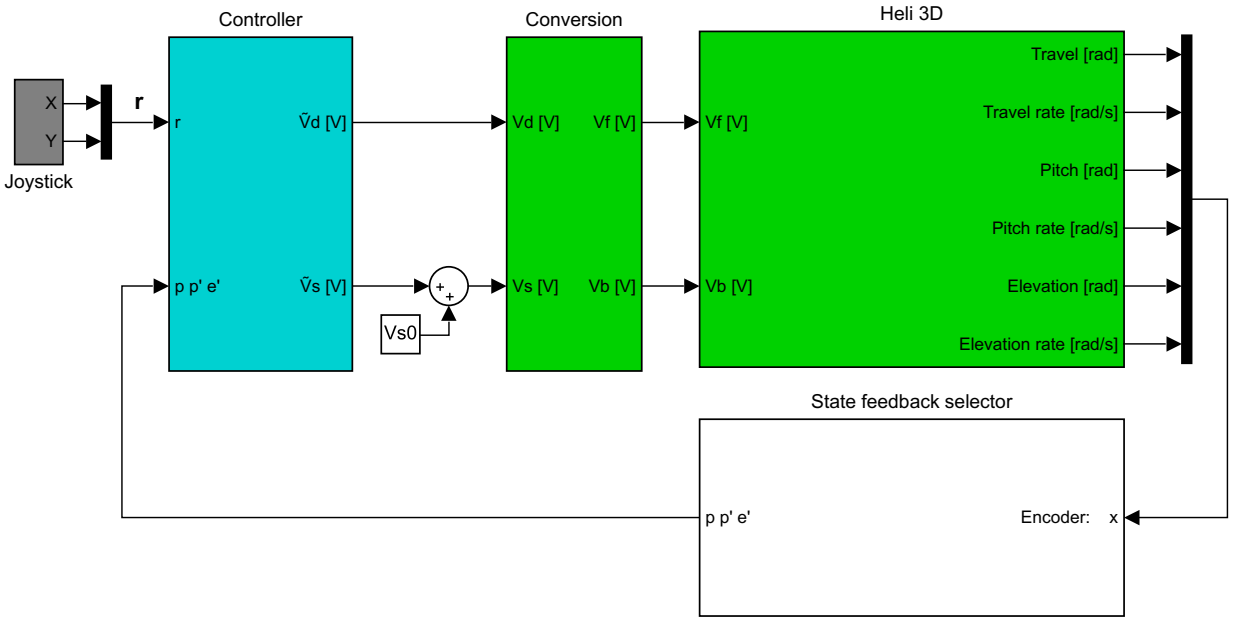


Figure 5: Lab 2 setup.

4.2 Pole placement

As shown in 17, the system with state- feedback and reference-feed-forward may be written as

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}(\mathbf{F}\mathbf{r} - \mathbf{K}\mathbf{x}) \tag{19}$$

we may modify this to look like

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{x} + \mathbf{B}\mathbf{F}\mathbf{r} \tag{20}$$

the feedback-dynamics are now dependent on the $(\mathbf{A} - \mathbf{B}\mathbf{K})$-matrix. With the MATLAB place command, we chose the poles of this matrix, and are given the resultant $\mathbf{K}$. $\mathbf{F}$ then follows as shown in 18

The poles were placed using the MATLAB place-command as shown below.

The initial poles were placed somewhat arbitrary, but modified according the the response based on servo input. Later on, a standardized testing program was implemented, simulating a step response. The performance of the system using a range of poles on this test-system is shown in figure 8.

Figure 8 shows various response in pitch of the system when applied a standardized testing program. We found the best set of poles to be $p = \begin{bmatrix} -3, & -3+3i, & -3-3i \end{bmatrix}^T$ becasue it follows the reference most closely of the poles considered. The red plot shows the response of the system when the complex conjugated poles are further into the left hand plane. This leads to a larger time constant, and a less favourable response.

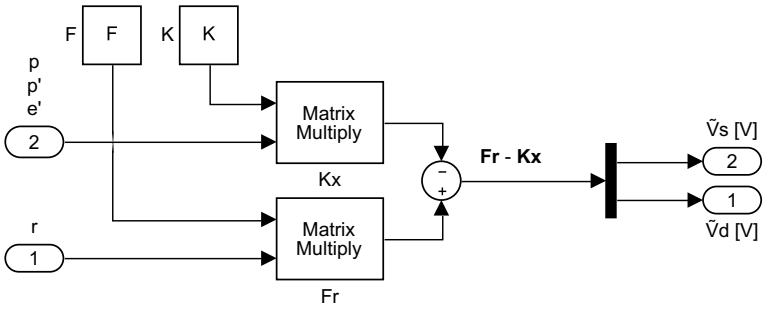


Figure 6: Simulink implementation of the controller without integral effect.

```
% System
A_c = [0   1   0;
       0   0   0;
       0   0   0];

B_c = [0   0;
       0   K1;
       K2  0];

% Poles
P = [-3; -3 - 3i; -3 + 3i];

% Controller gain matrix
K = place(A_c, B_c, P);
```

Figure 7: Pole placement of controller without integral effect. `A_c` and `B_c` corresponding to **A** and **B** of the system.

Moving the complex conjugated poles closer to the real axis, as shown in the yellow and purple plot. As complex conjugate poles may be written as $\lambda_{1,2} = [\alpha \pm \beta i]$, and the relative damping factor as $\zeta = \alpha/\sqrt{\alpha^2 + \beta^2}$, we see that a decrease in β leads to a smaller ζ and therefore less damping. This explains the overshot seen in by the yellow and purple plot.

The choice of poles gave the following **K** and **F**

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 16.2646 \\ 16.7752 & 5.5917 & 0 \end{bmatrix} \text{ and } \mathbf{F} = \begin{bmatrix} 0 & 16.2646 \\ 16.7752 & 0 \end{bmatrix}. \quad (21)$$

4.3 LQR

The MATLAB lqr(A,B,Q,R)-command yields the K-matrix minimizing the cost function given as

$$J = \int_0^\infty \mathbf{y}^T(t)\mathbf{Q}(t)\mathbf{y}(t) + \mathbf{u}^T(t)\mathbf{R}(t)\mathbf{u}(t)dt \quad (22)$$

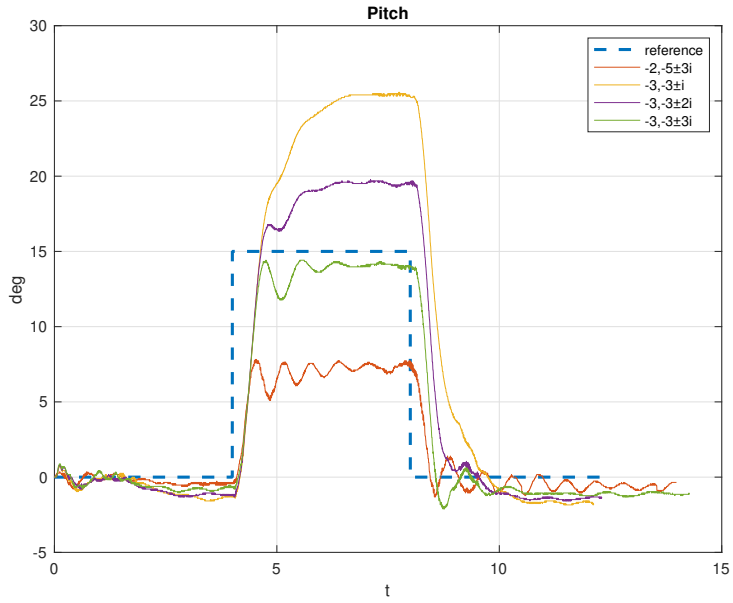


Figure 8: Pitch response using pole placement

Here **Q** and **R** are matrices determining a weight, or cost, for how much the system are willing th change the states and input respectively.

For simplicity, we determine **Q** and **R** to be diagonal-matrices. This means the elements along the diagonal directly represents the weighting given to the corresponding state or input from 12. This makes tuning easier.

We experimentally found a certain range of values giving a testable system. The values of the **R**-matrix had to be at least a order of magnitude less than the values in **Q**. Having them about the same means applying a lot of cost on input, which led to a slow system unable to reach reference. The values in **R** could however not become too small, because that would lead to to much input and a saturation on the physical motors.

Figure 9 shows the responses of some of the configurations giving a testable system witch regards to pitch. The red plot is unable to reach its reference because we are assigning to little of cost to deviate from the pitch reference. Increasing this led to a better response, as shown by the yellow and purple plot. A further increase led to the response closest to the reference, as shown by the purple plot. Although then we saw some oscillatory movement and decided not to increase it further.

The respons with regards to shows quite similar behavior between the two in figure 10. Although the response does overshot the reference, we decided that this was not as important as long as the helicopter manged to keep the desired elevation experimentally, which it did. We eventually

decided upon

$$\mathbf{Q} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 150 \end{bmatrix} \text{ and } \mathbf{R} = \begin{bmatrix} .08 & 0 \\ 0 & .1 \end{bmatrix} \tag{23}$$

these matrices yields $\mathbf{K}$ and $\mathbf{F}$ as

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 43.3013 \\ 31.6228 & 12.60720 & 0 \end{bmatrix} \text{ and } \mathbf{F} = \begin{bmatrix} 0 & 43.3013 \\ 31.6228 & 0 \end{bmatrix} \tag{24}$$

Interestingly, these are very dissimilar to $\mathbf{K}$ and $\mathbf{F}$ found by pole placement in figure 21.

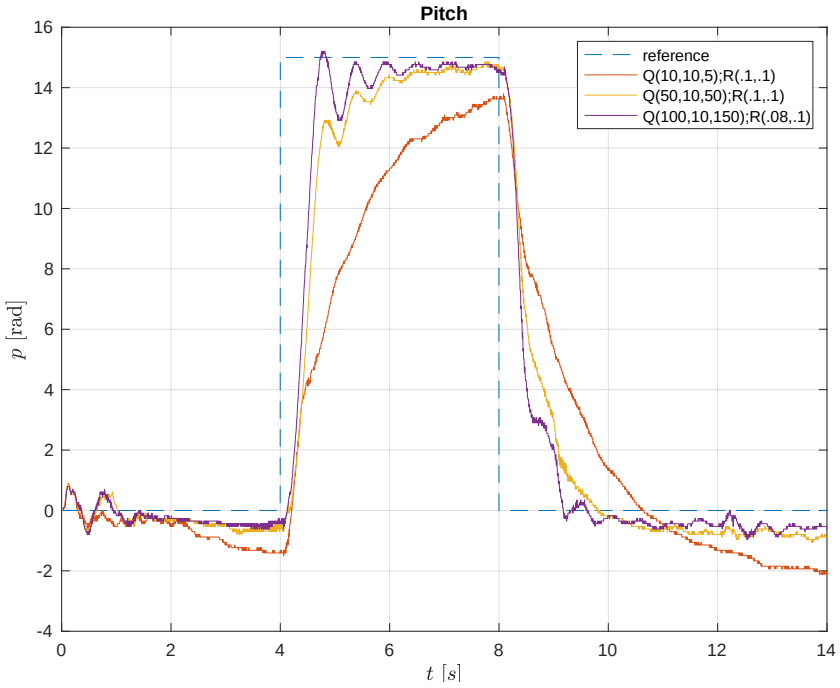


Figure 9: The responses in pitch from various LQR-configurations [NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

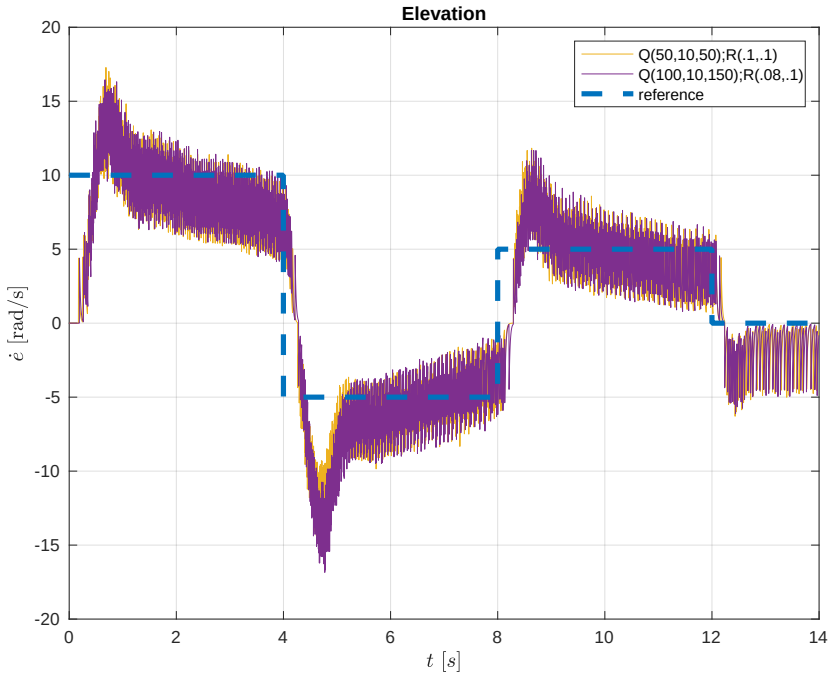


Figure 10: Elevation rate response using LQR. [NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

4.4 LQR with integral action

To augment the controller for integral effect, we introduce two new states

$$\dot{\gamma} = p - p_c \tag{25a}$$
$$\dot{\zeta} = \dot{e} - \dot{e_c} \tag{25b}$$

An integral controller is defined as

$$\mathbf{x}_a = \int_0^t \mathbf{C}\mathbf{x}(\tau) - \mathbf{r}(\tau) d\tau \tag{26}$$

yielding

$$\dot{\mathbf{x}}_a = \begin{bmatrix} \dot{\gamma} \\ \dot{\zeta} \end{bmatrix} = \begin{bmatrix} p - p_c \\ \dot{e} - \dot{e}_c \end{bmatrix} = \mathbf{C}\mathbf{x}_a + \mathbb{I}\mathbf{r} \tag{27}$$

If we include the augmented states in our state-space-system, giving the the augmented $\mathbf{A}$ and $\mathbf{B}$ new symbols, it will look like

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{x}}_a \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_a \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u} + \begin{bmatrix} \mathbf{0} \\ \mathbb{I} \end{bmatrix} \mathbf{r} \tag{28}$$

We set

$$\overline{\mathbf{A}} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \ , \overline{\mathbf{B}} = \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} \text{ and } \mathbf{G} = \begin{bmatrix} \mathbf{0} \\ \mathbb{I} \end{bmatrix} \tag{29}$$

so that the system ends as

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \\ \dot{\gamma} \\ \dot{\zeta} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}}_{\overline{\mathbf{A}}} \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \\ \gamma \\ \zeta \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}_{\overline{\mathbf{B}}} \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}}_{\mathbf{G}} \begin{bmatrix} p_c \\ \dot{e}_c \end{bmatrix} \tag{30}$$

The implementation of the integral effect is shown in figure 11. The further process for finding the feedback- and feed-forward gain are the same as in the LQR-task without integral effect. We must however, as we now have five states, modify the $\mathbf{Q}$ matrix to now include weighting for the augmented states.

The new $\mathbf{Q}$ matrix, which we mark as $\overline{\mathbf{Q}}$, is given as $5x5$-diagonal matrix.

As the nubmer of inputs remain unchanged, the $\mathbf{R}$-martix from the previous task is still valid.

We may then use the MATLAB lqr-command to find the optimal feedback gain as $\overline{K} = lqr(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{Q}}, \mathbf{R})$

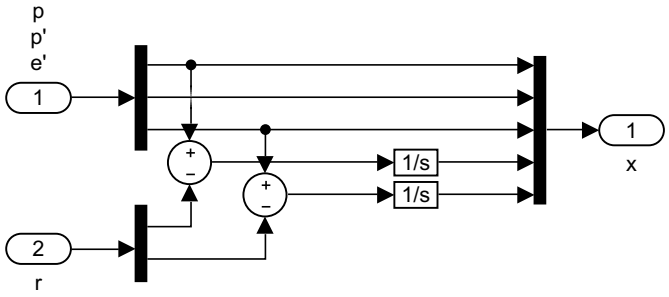


Figure 11: Simulink implementation of the controller with integral effect.

We initially tried the same values as for the regulator without integral effect 23, leaving the weighting for the new states to 1. This however, led to a quite large overshoot as shown by the red plot in 12. Furthermore, it was not able to reach, and hold requested elevation experimentally. The new state ζ is the integral $e - e_c$. Increasing the corresponding weight in $\overline{\mathbf{Q}}$ while decreasing the weight corresponding for $\dot{e}$, let the system reach reach requested elevation in a rapid manner. However, as the pitch and elevation are linked 4, the system became slower to respond to changes in pitch reference, as shown by the yellow plot in 12. We hence increased the weight on p to minimize the the deviation. Ended up with

$$\overline{\mathbf{Q}} = \begin{bmatrix} 150 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 50 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 50 \end{bmatrix} \text{ and } \mathbf{R} = \begin{bmatrix} .08 & 0 \\ 0 & .3 \end{bmatrix}, \tag{31}$$

because it, as shown by the purple plot in figure 12 had a relatively quick and stable response.

The resultant $\overline{\mathbf{K}}$ and $\overline{\mathbf{F}}$ then became:

$$\overline{\mathbf{K}} = \begin{bmatrix} 0 & 0 & 29.9345 & 0 & 25.0000 \\ 23.3511 & 8.7668 & 0 & 2.5820 & 0 \end{bmatrix} \tag{32}$$

$$\overline{\mathbf{F}} = \begin{bmatrix} 0 & 25.0000 \\ 23.3511 & 0 \end{bmatrix} \tag{33}$$

As we expect, the resultant values in the $\overline{\mathbf{F}}$ are less than in $\mathbf{F}$ because a lot of the job of the feedback is now performed by the integral.

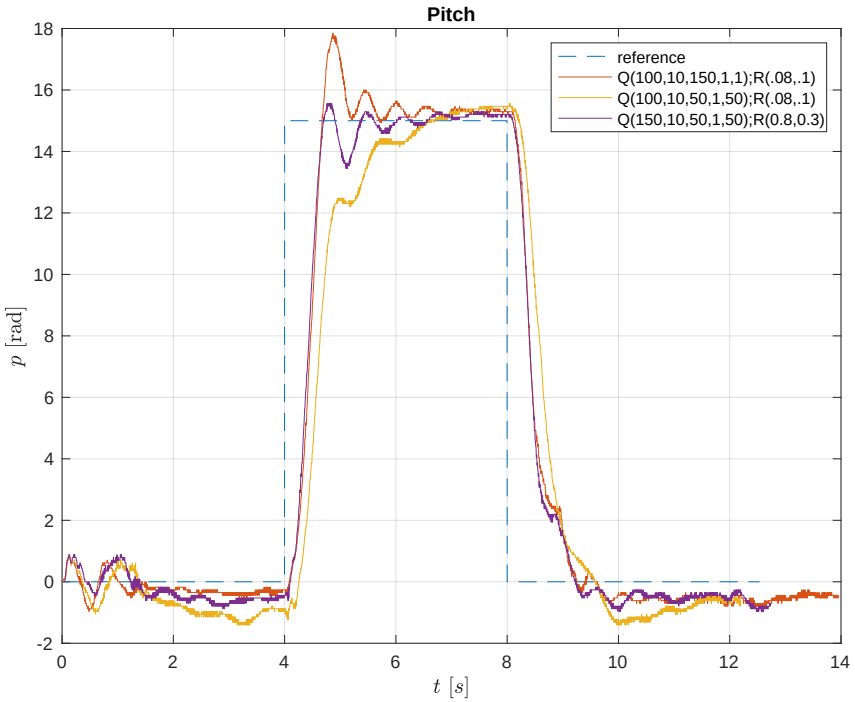


Figure 12: Pitch response LQR with integral effect.[NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

5 Luenberger observer

As real helicopters are not attached to the ground, the encoder values serves idealistic for control feedback, but not a realistic feedback. Instead, an inertial measurement unit (IMU) is mounted in the middle of the propellers. This unit gives measurements for acceleration vector and and gyroscope vector according to the reference system in figure 13. The measurements are

$$\mathbf{a} := \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}, \qquad \omega := \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}, \tag{34}$$

where $\mathbf{a}$ is the acceleration vector and the gyroscope vector ω have angle rates as $\dot{x}, \dot{y}, \dot{z}$ (not to be confused with linear derivative).

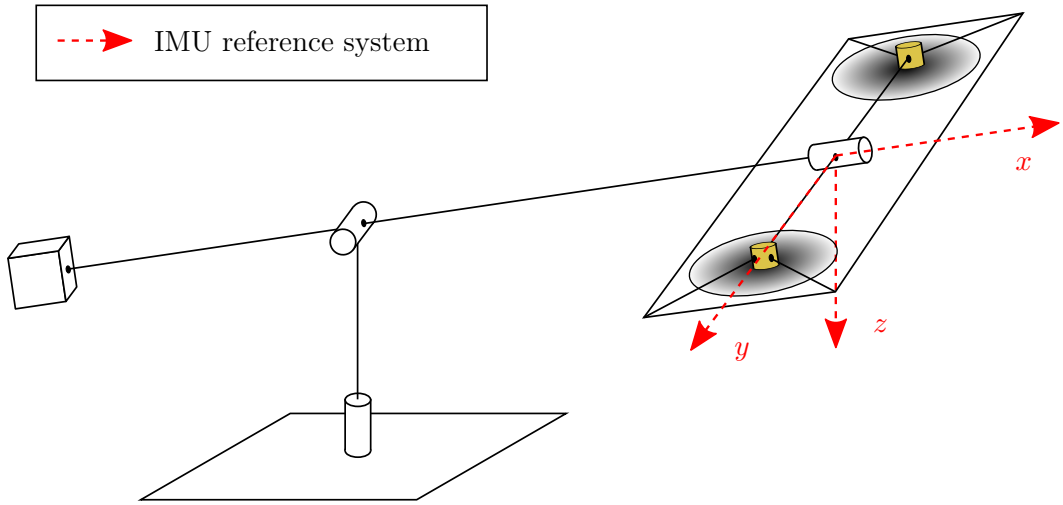


Figure 13: Helicopter IMU reference system.

We can see from figure 13 that the axis of the pair $\{x, p\}$ coincides, and the axis pairs $\{y, e\}$ and $\{z, \lambda\}$ are parallel axis when operating at the linearized point.

In the folowing part, we will look at the readings from the IMU, and explain why it is not possible to use the gyroscope rates directly for a measure for travel rate $\dot{\lambda}$, elevation rate $\dot{e}$ and pitch rate $\dot{p}$. Instead, we need to perform a transformation given by the pitch and elevation angle before the values can be used.

We should note at this point that the acceleration measurements of the gravitational field are used to calculate the pitch and elevation angle. Unfortunately, we are not able to measure the travel angle λ due to the gravitational field being parallel to λ-axis. This creates a symmetry in our setup, and marks the important notation that the angle λ is a state that is not measurable from the IMU. In order to measure this angle however, a real

helicopter could use GPS (e.g. flying in a direction would let you know which way the craft is heading) or use Earths magnetic field.

In this setup, we are not concerned about correcting the acceleration measurements by effects such as centripetal forces or other accelerations. Some discussion will follow.

At the end of the task, we will use the more realistic feedback (IMU measurements) for the control of the helicopter model, but the measurements contains noise. To smooth out the signal, we implement a Luenberger observer and tune this by pole placements. The Simulink implementation for this task is given by figure **??**. Here, we implemented a measurement conversion from the IMU. The Luenberger observer takes the input **u** and measurements **y** to estimate the states in our system. We also added an option to switch between encoder feedback and estimated states feedback.

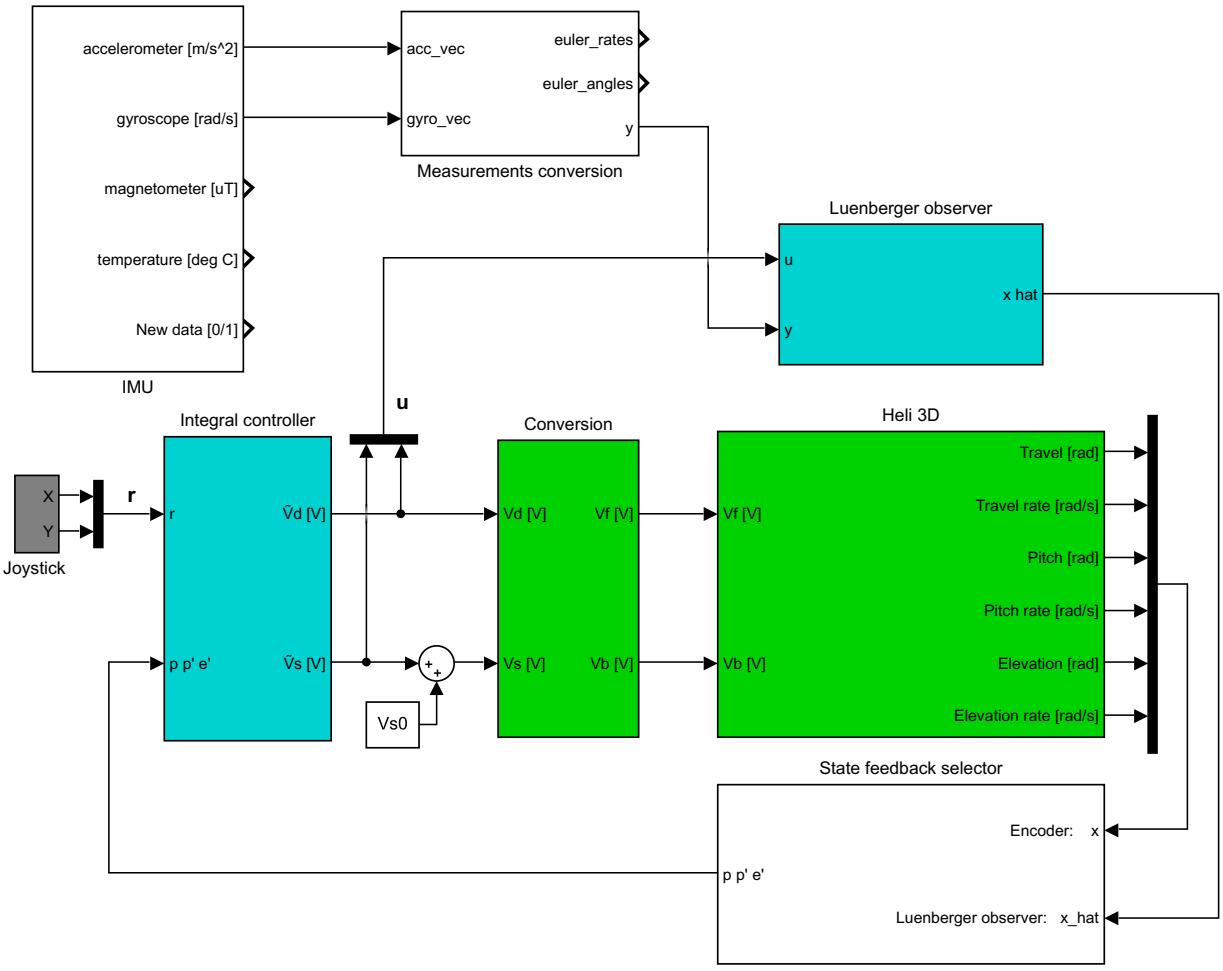


Figure 14: Setup for day 3.

5.1 IMU readings and transformation

The raw gyroscope measurements ω are compared to the encoder rates in figure 15. Here the helicopter is moved by hand at one axis at a time, starting from the linearized point. The gyroscope rates and encoder rates coincides. This is because the corresponding axis of the IMU reference system and the the helicopter model reference system is parallel, i.e. $x||p$, $y||e$ and $z||\lambda$. By holding two axis invariant, the resulting change in the last axis is the same for both reference systems.

When the pitch rate reached a level of about $\sim$ 4.4 rad/s in 15, the gyroscope measurement turned to a negative value at $\dot{x} = -4.363$ rad/s $=$ 250.0 deg/s. This can be seen at $t = 4.8$ s. We suppose this discrepancy is due to limitation or saturation on the rate measurements in the equipment. Such large rates do not happen when controlling the helicopter normally, such that this discrepancy is not something to worry about.

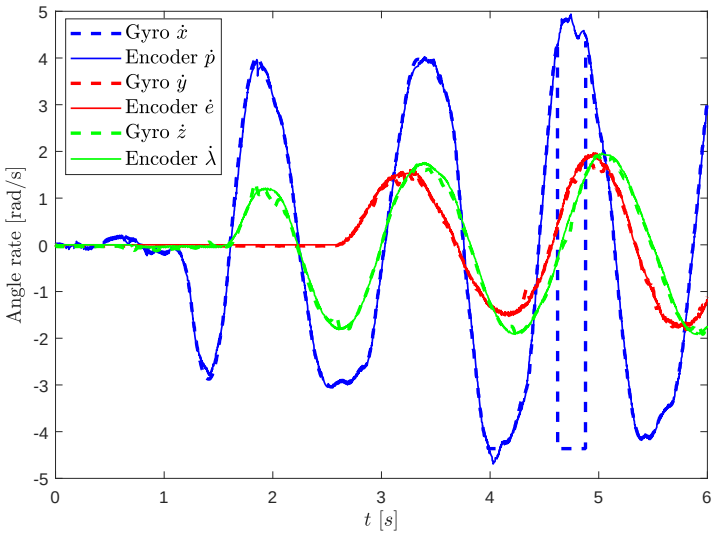


Figure 15: Raw gyroscope measurements compared to encoder rates.

It is interesting to see what happens when rotating the helicopter around multiple axis at the same time. Before continuing with the discussion, we show how the angles e and p is found from the acceleration vector. We state without deriving[2] that pitch is calculated as

$$p = \arctan\left(\frac{a_y}{a_z}\right), \tag{35}$$

[2]this is a part of the preparations

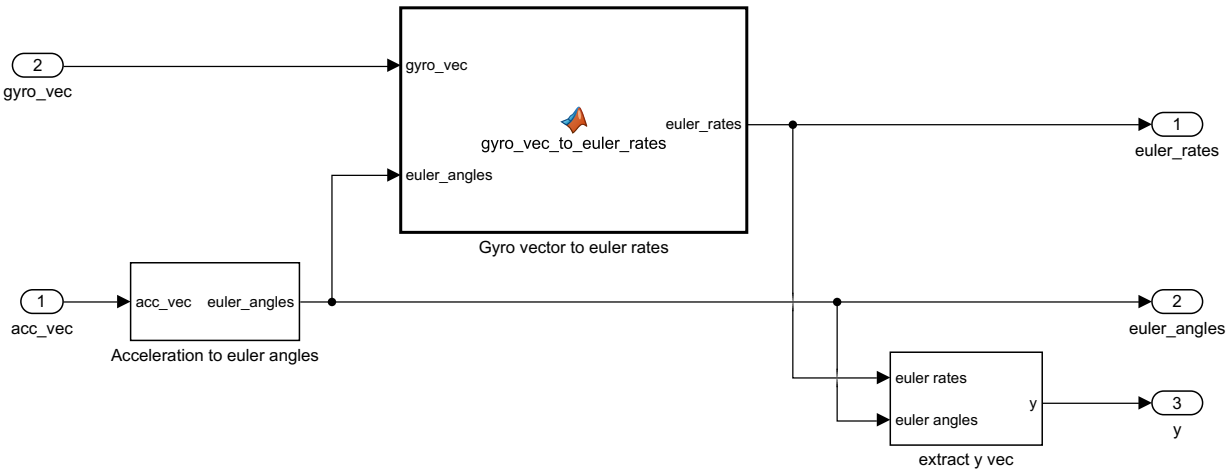


Figure 16: Simulink sub-block to convert the measurement from the IMU to the measurement vector $\mathbf{y}$ in figure 14.

and the elevation as

$$e = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right). \tag{36}$$

We also make the assumption that pitch and elevation angle cannot loop. Furthermore, a transformation for the gyroscope vector based on elevation and pitch was given as the Matlab function `gyro_vec_to_euler_rates`. The implementation of (35) and (36) with the given Matlab function is shown in figure 16 and 17.

Figure 18 shows how the raw gyroscope measurements and transformed measurements compare to the encoder rates by doing the same test as in figure 15, but adding a rotation in a second axis.

When moving the helicopter with an elevation and pitch movement in figure 18a, we noticed that the amplitude of the gyroscope rate $\dot{y}$ was lower than the encoder rate $\dot{e}$ when $\dot{p} \simeq 0$. One example occurs at $t = 3.5$ s. This shows that a tilt of the helicopter ($p \neq 0$) reduces the measurements of the elevation from $e = 0$. The transformed gyroscope measure was correct.

When moving the helicopter with an elevation and travel movement in 18b, the rates $\dot{\lambda}$ and $\dot{z}$ did not show any significant discrepancy from the encoder rates $\dot{e}$ and $\dot{\lambda}$. However, we noticed that in this situation, the transformed gyroscope rates gave worse measurements than the raw ones. This is due to acceleration of the helicopter by (e.g. movements up and down in elevation) affecting the measurement of the true gravitational field. This corrupts the calculation of the elevation angle and makes the transformation of raw gyroscope measurements by an "effective" gravitational field.

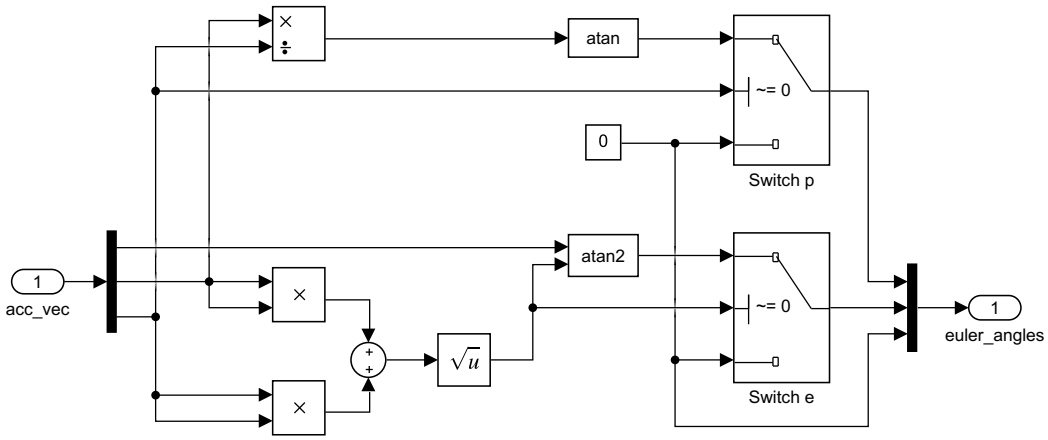


Figure 17: Simulink sub-block to convert acceleration vector to angles from figure 16. Note that travel λ is set to zero as it is not measurable.

When moving the helicopter with a pitch and travel movement in 18c, the only clear discrepancy was the raw gyroscope $\dot{z}$ around $\dot{x} \simeq \dot{z} \simeq 0$ as can be seen around $t = 2.6$ s. When $\dot{\lambda}$ and $\dot{p}$ changes sign, the raw gyroscope rate $\dot{x}$ showed a lower absolute value than the encoder rate $\dot{p}$. This also happened at around $t = 3.5$ s and $t = 4.2$ s. The corrected values matched the encoder rates in greater detail.

Some explanations of the discrepancies of the rates might be resolved by figure 19. This shows how the acceleration vector $\mathbf{a}$ is affected by similar movements as in figure 15, by restricting the movements to one axis at a time.

When moving the helicopter with travel movement in figure 19c, then a_y oscillated in accordance to the motion, but a_x oscillated with twice the frequency. As the helicopter is moving in i somewhat circular orbit between the endpoints, the centripetal force is acting towards the center. This causes a_x to be negative in between the endpoints when $a_y = 0$, e.g. at $t = 2.6$ s or $t = 5$ s. At the endpoints, the centripetal force $a_c \sim \dot{\lambda}^2 \ell_h$ goes to zero making $a_x \simeq 0$ when the magnitude of a_y is at max.

When moving the helicopter with pitch movement in figure 19a, then a_y oscillated in accordance with the motion, and a_z oscillated at twice the frequency. We state that $a_z = -g$ (helicopter counteracting gravitational acceleration) in equilibrium. When the helicopter is not tilted ($p = 0$), then $a_z \simeq -g$ at e.g. $t = 2.6$ s. When the helicopter is tilted in either way, then a_z has a smaller absolute value. This is due to the resulting normal force component in z-direction becomes smaller (it is at max when $p = 0$).

When moving the helicopter with elevation movement in figure 19b, both a_x and a_z oscillated in accordance to the motion. We see that a_z oscillates around $a_z = -g$ and a_x oscillates around $a_x = 0$. An explanation for a_z is

because of the needed acceleration at the endpoints to change direction of the movements in elevation. As for a_x, this is due to the normal force giving a positive component to a_x when $e > 0$ and negative component to a_x when $e < 0$.

In figure 20, we have plotted the transformed acceleration vector to pitch and elevation by similar movements as in figure 15, by only moving the helicopter at one axis at a time. We included λ to emphasize that this is set to zero. From the plot, movements in pitch did showed that by using (35), the measured pitch followed the encoder pitch. As for the elevation given by (36), we see that the elevation is off. When the helicopter is at top, it is accelerated downwards and in effect zeros out some of the gravitational measure and making a_y smaller. This makes the argument in (36) bigger, and as arctan is monotonic, elevation e is increased at the top. We also have a significant acceleration at the bottom which makes the calculated elevation a higher result than the encoder.

The elevation and pitch angle was correct when slowly moving the helicopter as shown for elevation in figure 21. The slow or rectilinear change of elevation approximates a non-accelerated reference, such that elevation and pitch angles are correct from acceleration measurements.

In summary of this subsection, we saw that the raw gyroscope rates closely resembled the encoder rates, but failed to give precise measurements. This happened for example when tilting the helicopter ($p \neq 0$) such that the measurements for elevation rate had lower amplitudes. In the case of moving the helicopter in travel, the measurements for pitch rate also had lower absolute values. The correction done by transforming the gyroscope values given by pitch and elevation, helped in most of the cases. One counter-example of this was with movements in both elevation and travel in figure 18b. In this case, the transformation made the raw measurement worse. This was likely caused by accelerations in the movement of the helicopter.

We also saw that the movements and acceleration in elevation angle had the most impact on calculating elevation and pitch angle to be used in the transformation. The tests deducted was rapid movement changes, and as a result, distorted the real measurements of the acceleration arrow. However, when the system behaved more slowly, then the calculated pitch and elevation angles were correct. This made the transformed gyroscope measurements closely resemble the encoder rates. An improvement on the system implemented in figure 16 would be to compensate for accelerations such as centripetal forces.

5.2 Observability

We concluded that the measure of the travel angle λ is not possible from the IMU. Therefore, it is not possible to use this as a measure for comparison in a Luenberger observer. We develop the following state-space equation

$$\begin{bmatrix} \dot{p} \\ \ddot{p} \\ \dot{e} \\ \ddot{e} \\ \ddot{\lambda} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ K_3 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \dot{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V}_s \\ V_d \end{bmatrix}, \tag{37}$$

following directly from equation (7) by removing λ. This is the linear system we are using in the Luenberger observer and we denote the state-space in (37) as $\dot{\mathbf{x}} = \mathbf{A}_o\mathbf{x} + \mathbf{B}_o\mathbf{u}$. Observability is a necessary criteria for the observer, and we will investigate which states are needed to be measured.

The observability matrix is defined as

$$\mathcal{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C}\mathbf{A}_o \\ \vdots \\ \mathbf{C}\mathbf{A}_o^{n-1} \end{bmatrix}, \tag{38}$$

with n being the number of states.

If we try to only measure the angle rates, then we have the $\mathbf{C}$ matrix in $\mathbf{y} = \mathbf{C}\mathbf{x}$ as

$$\mathbf{C}_o = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{39}$$

which makes the observability matrix in (38) as

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ K_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & K_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{40}$$

This matrix has $\text{rank}(\mathcal{O}) = 4 \neq n$, which makes the system not observable. From the lack of observability, this means that we cannot deduce all the states in (37) by only measure the rates of the angles. The explanation for this is that if we only know the rate of a quantity, an try to integrate to find the quantity, we are missing out an integration constant. However, by differentiating a quantity, it is always possible to deduce the quantity rate.

From this knowledge, it is possible to graphically show how it is possible to derive some states from others in figure 22. We have not seen this notation anywhere, so we call this an "observability graph". The observability graph have verticies that represents states, and directed edges with weights showing the relation from one state to another. The pitch angle p can be found by derivative of λ twice with respect to time multiplied by the constant K_3 by equation (4). From this graph, we see that in theory, the linear system in (37) is completely observable by only measuring e and $\dot{\lambda}$.

The observability matrix can verify that we indeed only need to measure p and $\dot{\lambda}$. The $\mathbf{C}$ matrix becomes

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{41}$$

and only the three first computations are needed for the observability matrix to obtain full rank as

$$\mathcal{O} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ K_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & K_3 & 0 & 0 & 0 \end{bmatrix}. \tag{42}$$

This shows that the system in (37) is observable since the observability matrix has full rank.

As a sidenote for observability, differentiating may cause amplifing of noise in our measurements. It is of this reason that we postulate that when designing an observer, one should aim for as many measured states as possible. Even though the system should theoretically be possible to control with an observer that only receives measure of e and $\dot{\lambda}$, the system would problaby behave poorly. Unfortunately, we did not test the Luenberger observer given in the next part by only receiving e and $\dot{\lambda}$. We think the system might behave poorly by two reasons. (1) The system is actually non-linear, and the estimation from an observer might not compensate the model as good outside the linear operating point. (2) As briefly described, derivations amplify noise. This might be a trouble for the controller if the estimated state is related by the derivative of another state, and used as feedback. As an example, $p = K_3\ddot{\lambda}$ making the integral effect $\int(p_c - p)\mathrm{d}t$ integrating an

error because of p is fluctuating (contains more noise). This would vary the input voltage sum V_s more than necessary.

5.3 State estimator

As the measurements from the IMU contains noise, we introduce the Luenberger observer that smooths the signal. This has the form

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}_o\hat{\mathbf{x}} + \mathbf{B}_o\mathbf{u} + \mathbf{L}(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}}), \qquad \hat{\mathbf{y}} = \mathbf{C}\hat{\mathbf{x}} \tag{43}$$

where hat denotes the estimate of the state. The error dynamics of (43) can be found by denoting $\mathbf{e} := \mathbf{x} - \hat{\mathbf{x}}$. When there is noise in the measurement, $\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{n}$, then (43) can be written as

$$\dot{\mathbf{e}} = (\mathbf{A} - \mathbf{L}\mathbf{C})\mathbf{e} - \mathbf{L}\mathbf{n}. \tag{44}$$

The poles of $(\mathbf{A} - \mathbf{L}\mathbf{C})$ determines the stability and dynamics of the error of the estimated states.

The implementation of the Luenberger observer is shown in figure 23, and below is the Matlab code used to generate the matrices $\mathbf{A}_0$, $\mathbf{B}_0$ and pole placement by `transpose(place(transpose(Ao), transpose(Co), Po))`. The `place`-command computes the gain matrix $\mathbf{L}$ from our choice of poles.

```
% Observer matrices
Ao = [0   1   0   0   0;
      0   0   0   0   0;
      0   0   0   1   0;
      0   0   0   0   0;
      K3  0   0   0   0];

Bo = [0   0;
      0   K1;
      0   0;
      K2  0;
      0   0];

Co = eye(5);

Po = 5*[-1; -4; -4+0.2i; -4-0.2i; -0.5];

% Luenberger gain matrix
L = transpose(place(transpose(Ao), transpose(Co), Po));
```

When placing the poles for the Luenberger observer, we should make the observer faster than the dynamics of the system with the controller $(\mathbf{A}-\mathbf{B}\mathbf{K})$. This means placing the poles to the left of all poles in the system with controller. We experienced with placing the poles in close proximity to the left of the poles of $(\mathbf{A}-\mathbf{B}\mathbf{K})$. This resulted in slow dynamics of the estimates, such that the amplitudes were lower.

By tuning, we ended up with the poles

$$5 \cdot \begin{bmatrix} -1 & -4 & -4+0.2i & -4-0.2i & -0.5 \end{bmatrix}, \tag{45}$$

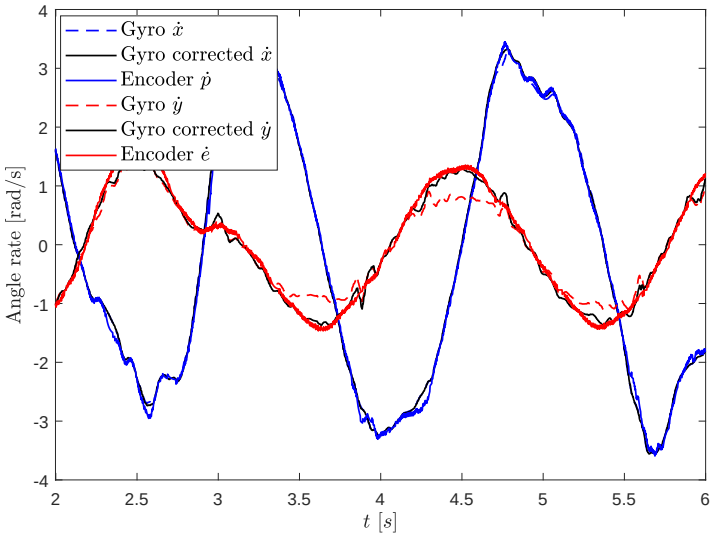
which resultet in the following gain matrix

$$\mathbf{L} = \begin{bmatrix} 20 & 0 & 0 & 0 & 0 \\ 1 & 20 & 0 & 0 & 0 \\ 0 & 0 & 5 & 1 & 0 \\ 0 & 0 & 0 & 20 & 0 \\ 0.9667 & 0 & 0 & 02.5 & \end{bmatrix}. \tag{46}$$

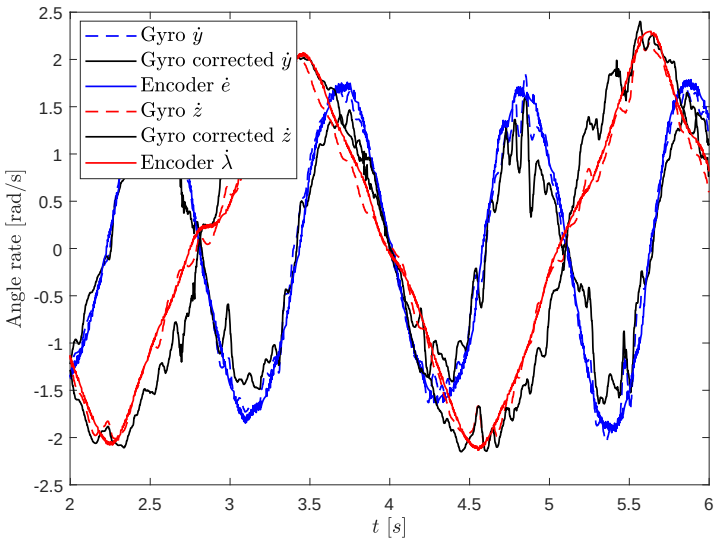
We note that lower values of $\mathbf{L}$ will result in slower estimators, as the correction term in (43) do not contribute as much. However, high values of $\mathbf{L}$ will result in more noise being amplified in the error dynamics as equation (44) will show. If the model contains a constant disturbance or inaccuracy, then higher $\mathbf{L}$ should be preferred. In our pole placement, we noted that the system did not require a very high gain matrix. This let us should some poles with a relative high real part.

Plot over the estimate for pitch and elevation is given in figure 24. Interestingly, it seems that the dynamics is slower for elevation than for pitch. This turned out to be a good thing as the measurement noise reached some high peaks for the elevation at $t = 1$ s and $t = 9$ s. If the dynamics of the luenberger observer was higher, then we would see a more significant increase at these time stamps.

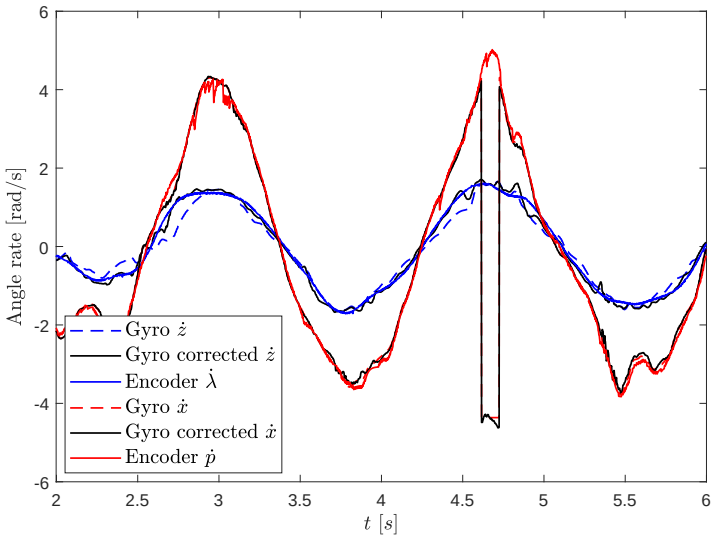
When we tried to tune the poles more into the left half plane, then we got a significant more noise as the observer more closely followed the noise in our measurements.



(a) Elevation and pitch movement.

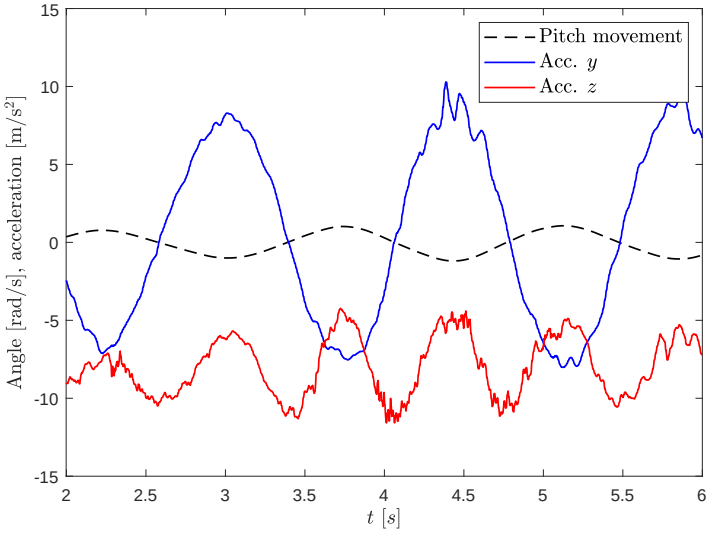


(b) Elevation and travel movement.

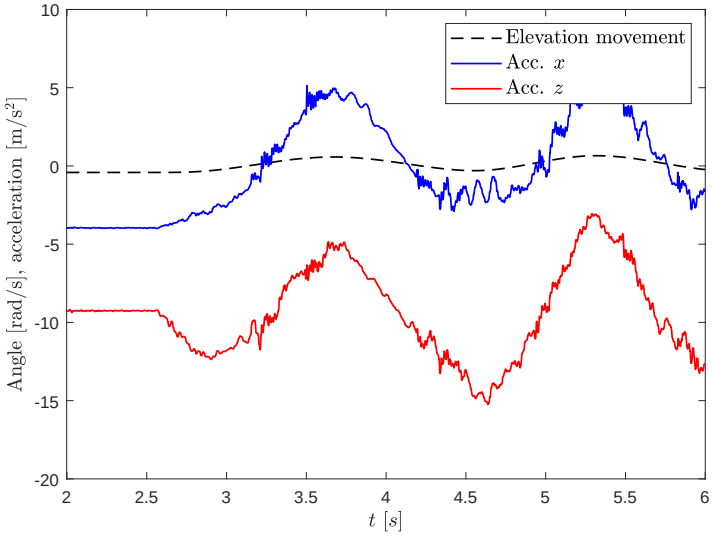


(c) Pitch and travel movement.

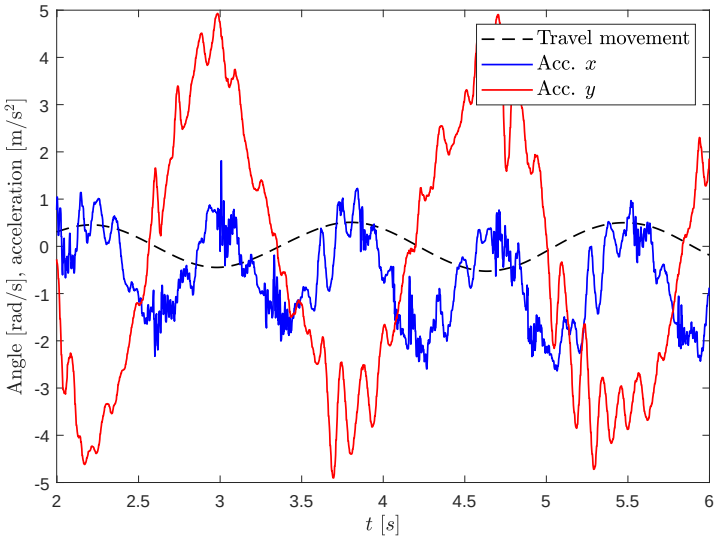
Figure 18: Raw and transformed gyroscope measurements compared to encoder rates by manually moving the helicopter model in two axis at the same time.



(a) Pitch movement.



(b) Elevation movement.



(c) Travel movement.

Figure 19: How acceleration components is affected by moving the helicopter at one axis at a time. The component of acceleration orthogonal to the movement is not shown.

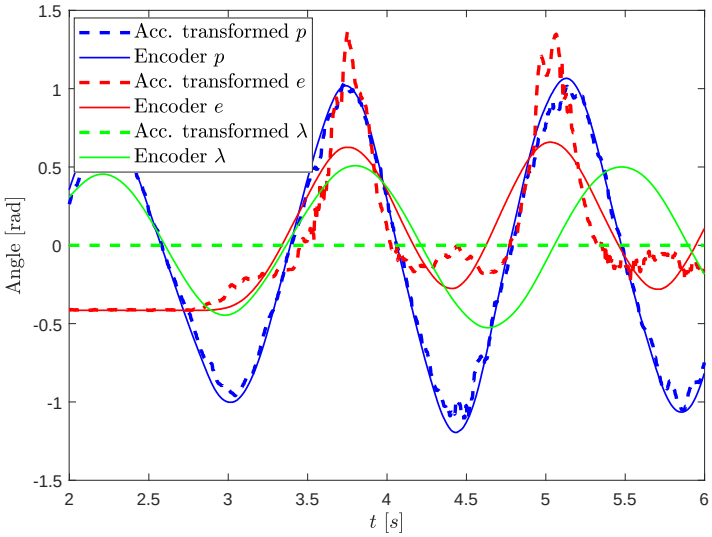


Figure 20: Transformation of acceleration measurement to elevation and pitch. Rapid acceleration in elevation angle give rise to a large deviation from the elevation as measured from the encoder.

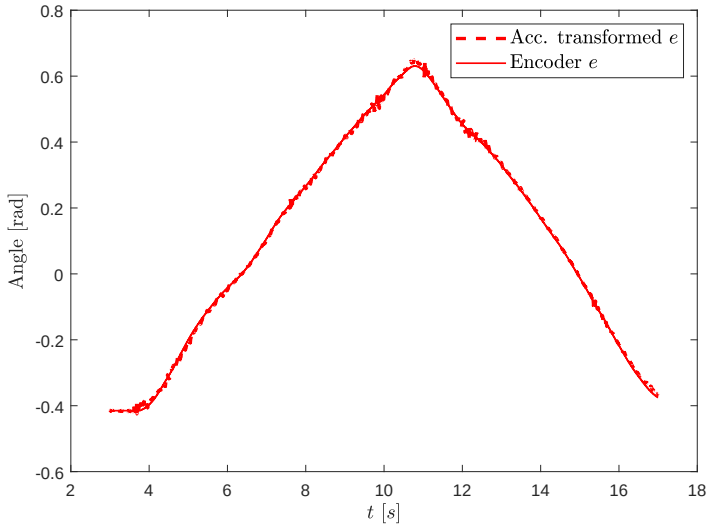


Figure 21: Slowly changing elevation yields correct elevation from the acceleration measurements as opposed to the accelerated reference system in figure 20.

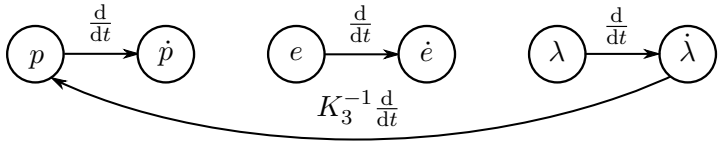


Figure 22: "Observability graph". Verticies show states, and edges their relation from one state to another. By counting the number of states needed to traverse all the other states, we find that only measuring e and λ is needed to make the system observable.

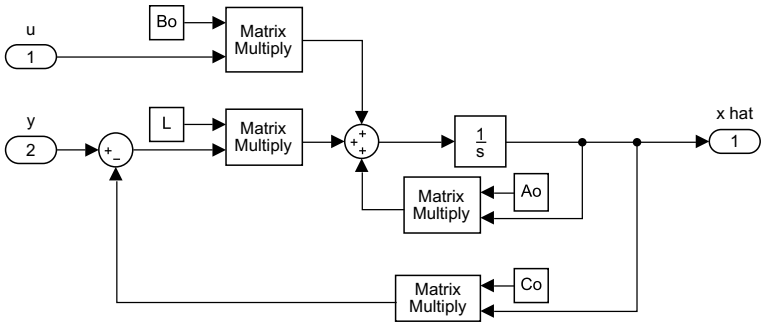
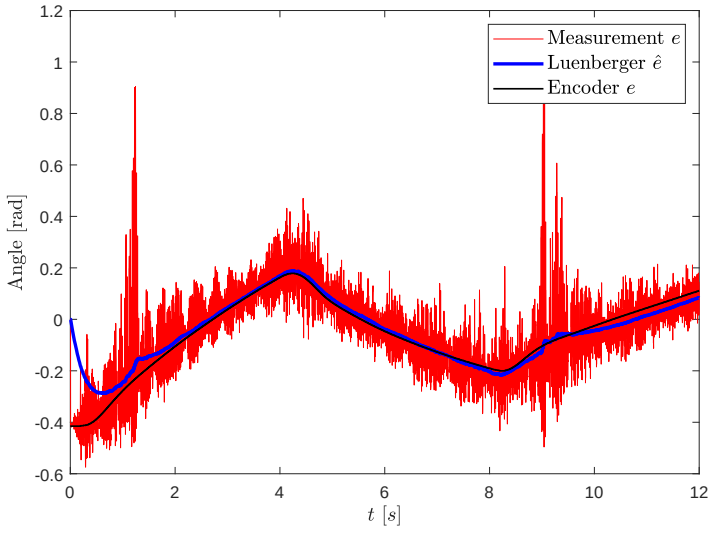
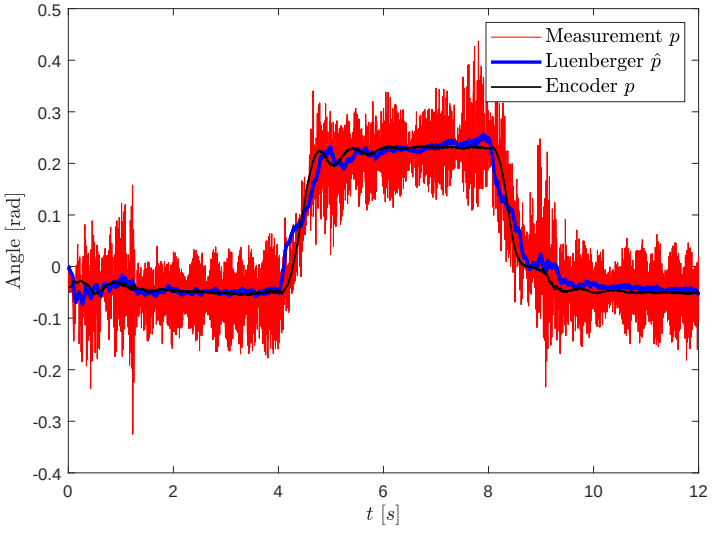


Figure 23: Simulink implementation of Luenberger observer by equation (43). The matrices `Ao`, `Bo`, `Co` and Luenberger gain `L` are retrieved by Matlab script.



(a) Elevation estimate.



(b) Pitch estimate.

Figure 24: Luenberger observer compared to the raw measurement and encoder values.

6 Kalman filter

Given a stochastic system as

$$\dot{\mathbb{x}}(t) = \mathbf{A}\mathbb{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}\mathbb{w}(t) \tag{47}$$
$$\mathbb{y} = \mathbf{C}\mathbb{x}(t) + \mathbb{v}(t)$$

Where $\mathbb{w}$ is the disturbance affecting the process, and $\mathbb{v}$ the measurment noice.

We define the error in our state estimation as

$$\hat{\mathbb{e}}(t) = \mathbb{x}(t) - \hat{\mathbb{x}}(t) \tag{48}$$

The covariance of the estimatation error is given as

$$\hat{\mathbf{P}}(t) = E[\mathbb{e}(t)\mathbb{e}^T(t)] \tag{49}$$

The elements along the diagonal of the $\hat{\mathbf{P}}$ signifies the uncertainty in the estimates. The Kalman filer aims to minimize this.

6.1 Noise estimate

The measurment covariance matrix is defiend as

$$\mathbf{R}_d = E[\mathbb{w}\mathbb{v}^T] \tag{50}$$

and gives an indication of the expected measurement noise.

We attempted to determine it experimentally by taking a time series while the helicopter was laying on the ground, and another when it flew still along the linearization point.

The result is given in equation 25. The variations were a lot larger than when laying on the ground, which is reasonable because of vibrations, sub-optimal controller and motor noise. The noise resembles bounded Gaussian noise because it is relatively unbiased. It could also be white considering its intensity remain somewhat unchanged.

We used the MATLAB Cov()-command to find the covariance of the noise as

$$\mathbf{R}_d = \begin{bmatrix} 0.001672 & 0.000587 & -0.000146 & -0.001441 & -0.0000083 \\ 0.000587 & 0.000579 & -0.001000 & -0.000034 & -0.000057 \\ -0.000146 & -0.001000 & 0.00376042 & -0.001196 & 0.000900 \\ -0.001441 & -0.000034 & -0.001196 & 0.002239 & 0.000017 \\ -0.0000083 & -0.000057 & 0.000900 & 0.000017 & 0.000904 \end{bmatrix} \tag{51}$$

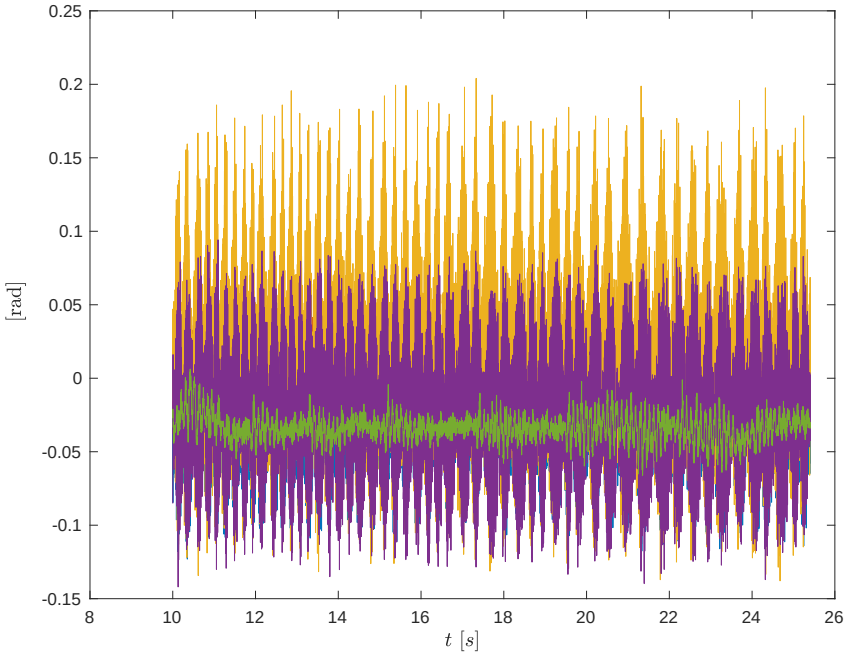


Figure 25: The measurments vector $\mathbf{y}$ when flying still at linearization point. The plot serves as an illustration for the approximation of white noise in the measurements. [NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

6.2 Discretization

When discretizing, we wish to turn the continuous model shown in equation 7 into a discrete model. This is done in MATLAB as

$$[\mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d] = ssdata(c2d(ss(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, timestep))) \tag{52}$$

the timestep is given in the assignment as 0.002.

This gave

$$\mathbf{A}_d = \begin{bmatrix} 0 & 0.0020 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.0020 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.0020 \\ 0.0019 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{53}$$

$$\mathbf{B}_d = \begin{bmatrix} 0 & 0 \\ 0 & 0.0021 \\ 0 & 0 \\ 0.0004 & 0.0020 \\ 0 & 1 \end{bmatrix} \tag{54}$$

$$\mathbf{C}_d = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{55}$$

$$\mathbf{D} = 0$$

The discrete system the looks like

$$\begin{aligned} \mathbb{x}[k+1] &= \mathbf{A}_d \mathbb{x}[k] + \mathbf{B}_d \mathbf{u}[k] + \overline{\mathbb{w}}[k] \\ \mathbb{y} &= \mathbf{C}_d \mathbb{x} + \overline{\mathbb{v}}[k] \end{aligned} \tag{56}$$

where $\overline{\mathbb{w}}$ and $\overline{\mathbb{v}}$ represents respectively the average disturbance and noise across a sampling interval.

6.3 Implementation

The setup for day 4 is shown in figure 26 where we introduced the subsystem Kalman filter. The following code snippets show the prediction part and correction as shown in figure 27.

```
function [x_bar, P_bar] = Prediction(x_hat, P_hat, u, A_d, Q_d, B_d)

    x_bar = A_d*x_hat + B_d*u;
    P_bar = A_d*P_hat*transpose(A_d) + Q_d;

end
```

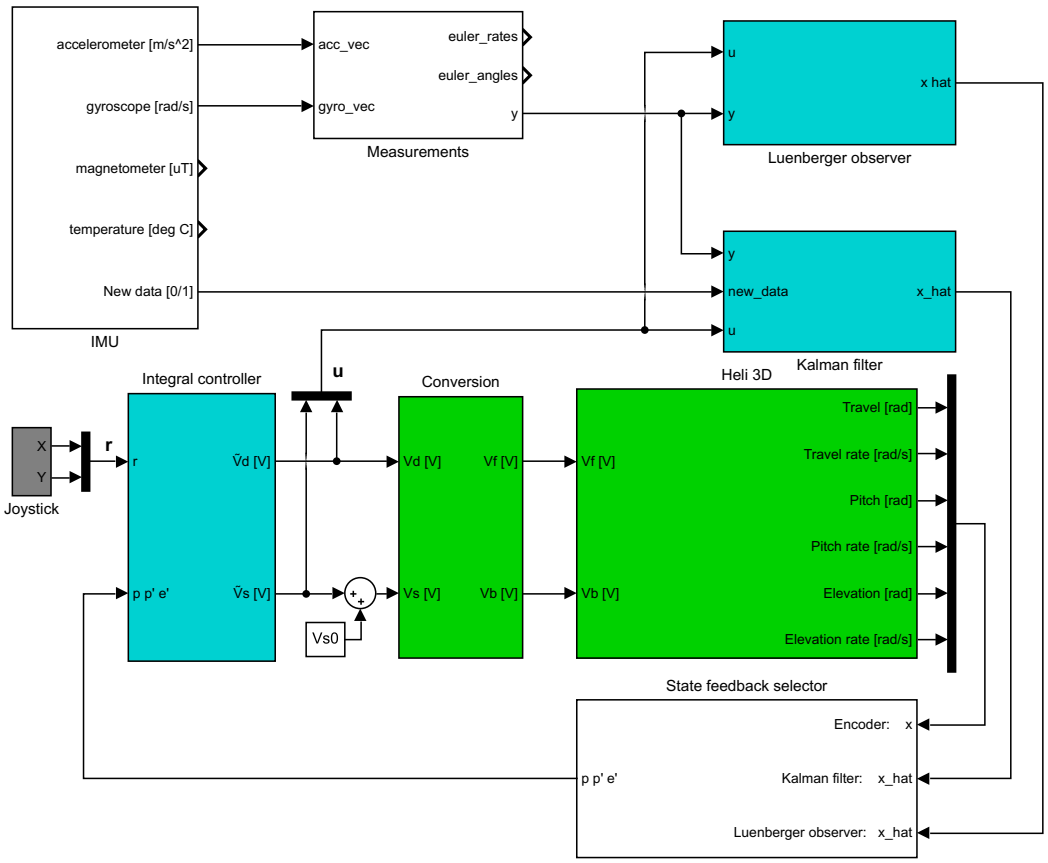


Figure 26: Lab setup day 4.

```
function [x_hat, P_hat] = correction(x_bar, P_bar, y, new_data, R_d, C_d)

    if (new_data)
        K = P_bar*transpose(C_d)*inv(C_d*P_bar*transpose(C_d) + R_d);
        x_hat = x_bar + K*(y - C_d*x_bar);
        P_hat = (eye(6) - K*C_d)*P_bar*transpose(eye(6) - K*C_d) + K*R_d*transpose(K);
    else
        x_hat = x_bar;
        P_hat = P_bar;
    end

end
```

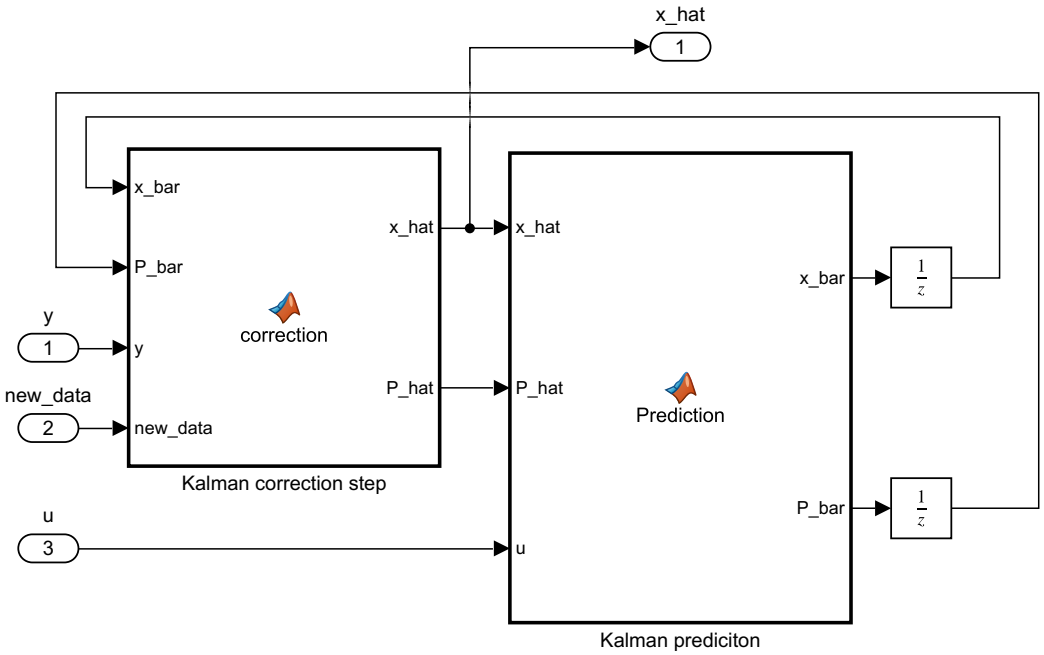


Figure 27: Kalman filter.

```
% Full state-space matrices
A = [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;
     K3  0   0   0   0   0];

B = [0   0;
     0   K1;
     0   0;
     K2  0;
     0   0;
     0   0];

C = [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   0   1];

D = 0;

% Discretization
timestep = 0.002;
[A_d, B_d, C_d, D_d] = ssdata(c2d(ss(A,B,C,D), timestep));

Q_d = 10e-6*diag([.05; 1; .1; .1; 1; 1;]);
```

6.4 Experimentation

As $\mathbf{Q}_d$ represents the uncertainty of our model, letting $\mathbf{Q}_d \to 0$ is equal to saying that our model is without uncertainty. It will then disregard any measurement, and only estimate according to the model. Conversely, letting $\mathbf{Q}_d \to \infty$ converts to completely disregarding the validity of the model. The estimation will then solely be based upon the measurements. Experimentally, neither led to a useful state estimator.

Considering the implementation of the Kalman filter, as shown in figure 26, based upon the equations of the equations in the preparation, we see that setting $new\_data = 0$ will circumvent the correction step. It is the same as saying $\hat{\mathbb{x}} = \bar{\mathbb{x}}$ and $\hat{\mathbf{P}} = \bar{\mathbf{P}}$. Since $\bar{\mathbb{x}}$ is fed into the correction-block through a unit-delay-block, the system then becomes

$$\hat{\mathbb{x}}[k+1] = \mathbf{A}\hat{\mathbb{x}}[k] + \mathbf{B}\mathbf{u}[k] \tag{57}$$

$$\hat{\mathbf{P}}[k+1] = (\mathbb{I} - \mathbf{K}\mathbf{C}_c)\hat{\mathbf{P}}[k](\mathbb{I} - \mathbf{K}\mathbf{C}_c)^T + \mathbf{K}\mathbf{R}_d\mathbf{K}^T \tag{58}$$

Considering, for simplicity, a autonomous SISO-system. Then these equations

$$\hat{x}[k+1] = a\hat{x}[k] \tag{59}$$

$$\hat{p}[k+1] = (1-kc)^2\hat{p}[k] \tag{60}$$

obviously becomes unstable if a or $(1-kc)^2$ is 1. The eigenvalues of $\mathbf{A}_d$ are all equal to 1. This means instability in a discrete system.

In figure 28 the estimated pitch is plotted up against the pitch measurement of the encoder. The exponential growth characteristic for an unstable system is noticeable, as well as the sudden correction when new data is introduced after about 32 seconds. This rapid correction happens because $\hat{\mathbf{P}}$ has been allowed to grow exponentially, leading to a large $\mathbf{K}$ once $new\_data = 1$. This again leads to a large correction of $\hat{\mathbf{x}}$.

Comparison Luenberger observer and Kalman filter

From a purely practical standpoint, the Luenberger observer and Kalman filter did not expose to much a difference in the results. As seen in figure 29, the estimates from the Kalman and the Luenberger observer are similar. However, we found the Luenberger observer to be a less intuitive with regards to tuning, than the Kalman filter. While the Kalman filter has a direct relationship between the values along the diagonal in $\mathbf{Q}_d$ and the corresponding elements in the state vector, he relationship between the poles of the $(\mathbf{A} - \mathbf{L}\mathbf{C})$-matrix in the Luenberger and the system states are less clear. From a more theoretical perspective, Luenbergers $\mathbf{L}$-matrix is fixed. This makes the operation deterministic, which is a general characteristic for observers. The Kalman Filter however continuously updates its Kalman gain $\mathbf{K}$ based upon the model uncertainty and measurement noise.[1]

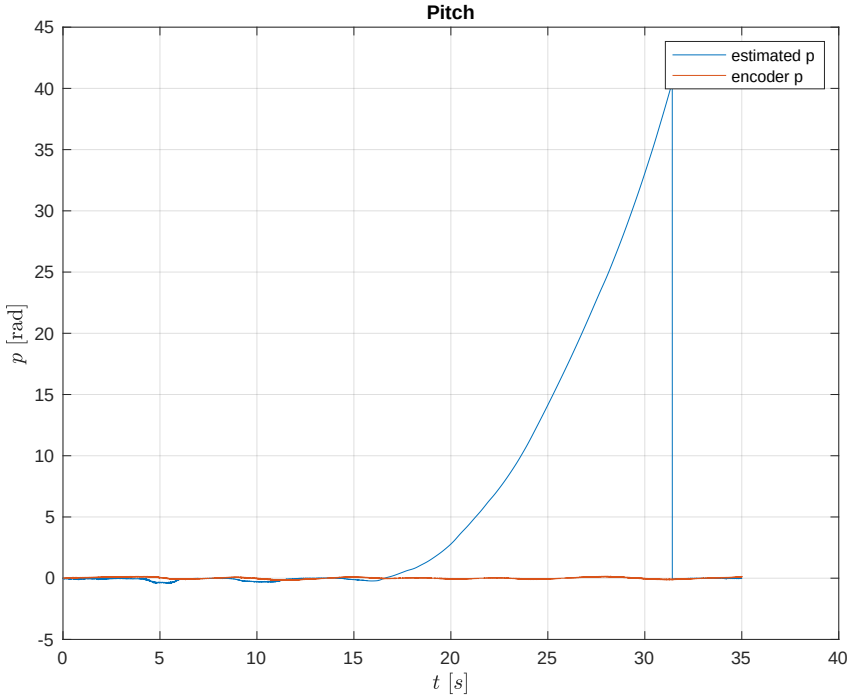
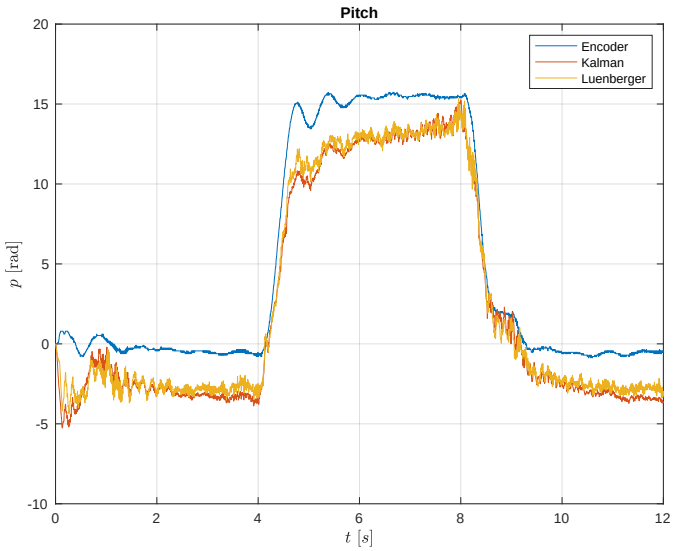


Figure 28: Estimated pitch versus measured when the correction step was [NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

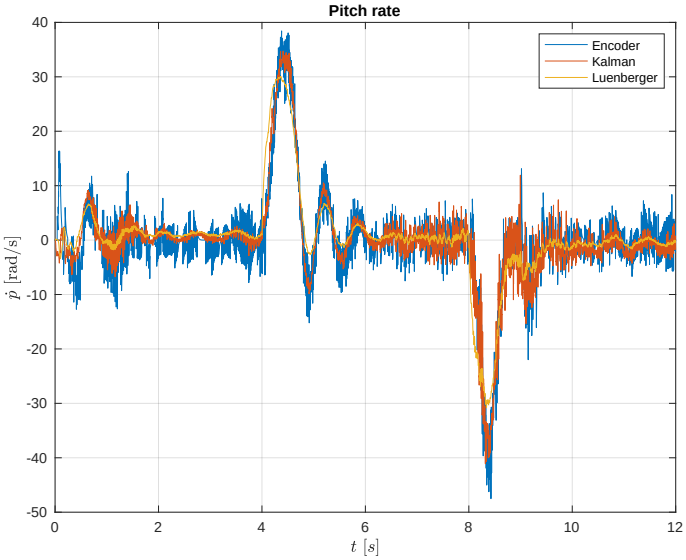
6.5 Tuning

As discussed in section 6.4, $\mathbf{Q}_d$ has to be small in order for the estimator to regard the model as useful. Since the system is using radians, $\mathbf{Q}_d$ obviously has to be a lot less than $\mathbb{I}$ because a uncertainty of ± 1 radian is unmanageable. We quickly found a workable order of magnitude of 10^{-6} and tuned thereafter.

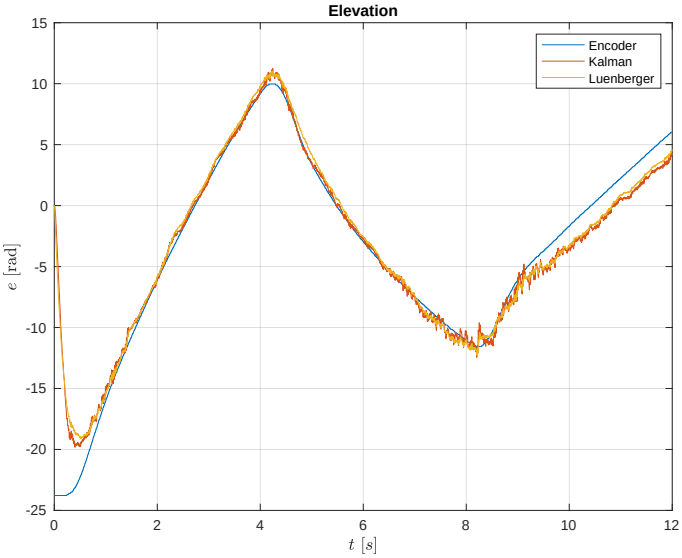
From the couple of more viable solutions, we chose the $\mathbf{Q}_d$ having $10^{-6}[.05, 1, .1, .1, 1, 1]$ along the diagonal. Other configurations had a larger value in $\mathbf{Q}_d$ corresponding to rates such as, elevation rate. This means it trust the measurements to an larger extent. The measurements are noisy, which leads to unwanted spikes, seen for instance by the yellow plot of figure 30.



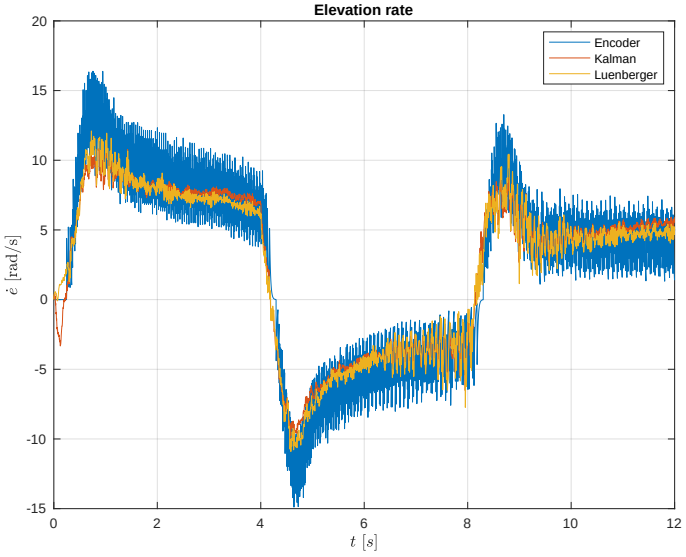
(a) Pitch



(b) Pitch rate



(c) Elevation



(d) Elevation rate

Figure 29: How the estimates from the Luenberger Observer and Kalman filter compare to the encoder measurments and each other [NOTE: these plot is plotted in degrees, not radians as stated along the y-axis]

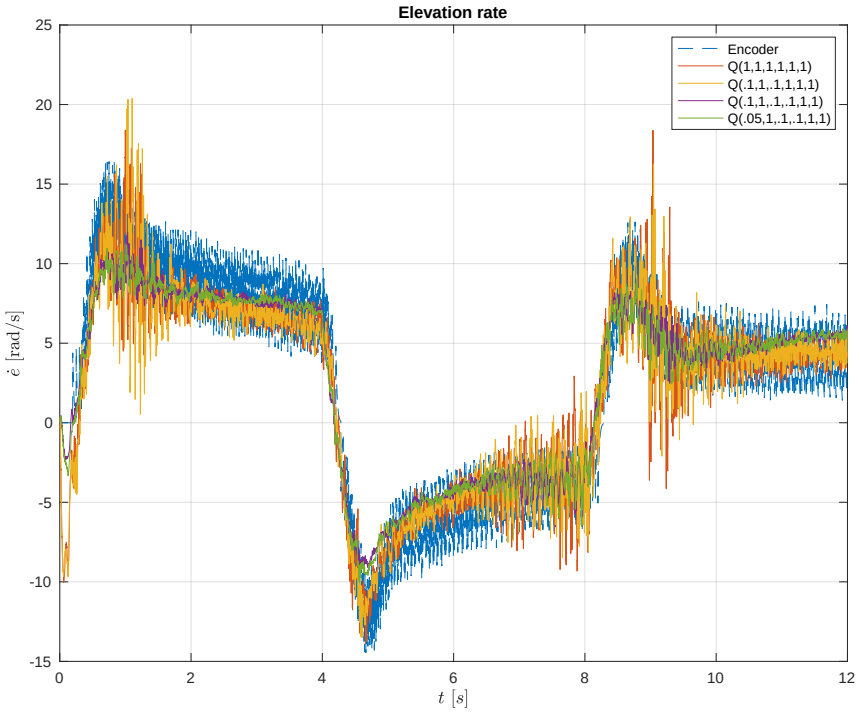


Figure 30: The encoder measurment of elevation rate up againts a series of measurments from various Kalman filters. [NOTE: this plot is plotted in degrees, not radians as stated along the y-axis]

7 Conclusion

In this lab, we have briefly tried to fly the helicopters via open-loop control in the beginning, and found that the closed-loop controllers gave a stable and fast control over pitch and elevation rate. It was not clear when going from pole placement to LQR and LQR with integral that we managed to tune in a better control over the helicopter. Likewise, the Kalman filter was not obviously better than the Luenberger observer, and may speak for the difficulty of tuning.

The linearization and assumptions the model is built upon proved valid to a certain extent through experimentation, as it could control the helicopter. Because linear systems provides us with so many useful tools, it is clearly a useful technique on relatively complex systems such as this one.

References

[1] Taha A. Wang J. Qi J. “Comparing Kalman Filters and Observers for Power System Dynamic State Estimation with Model Uncertainty and Malicious Cyber Attacks.” In: (2018).

References

[1] C.T. Chen, Linear System Theory and Design, 3rd. ed. (Oxford University Press, Oxford, 1999).

[2] Norwegian University of Science and Technology. (2020, October 6). TTK4115 - Linear System Theory. Retrieved from https://www.ntnu.edu/studies/courses/TTK4115

[3] Laboratory for Department of Engineering Cybernetics, NTNU, 2020.