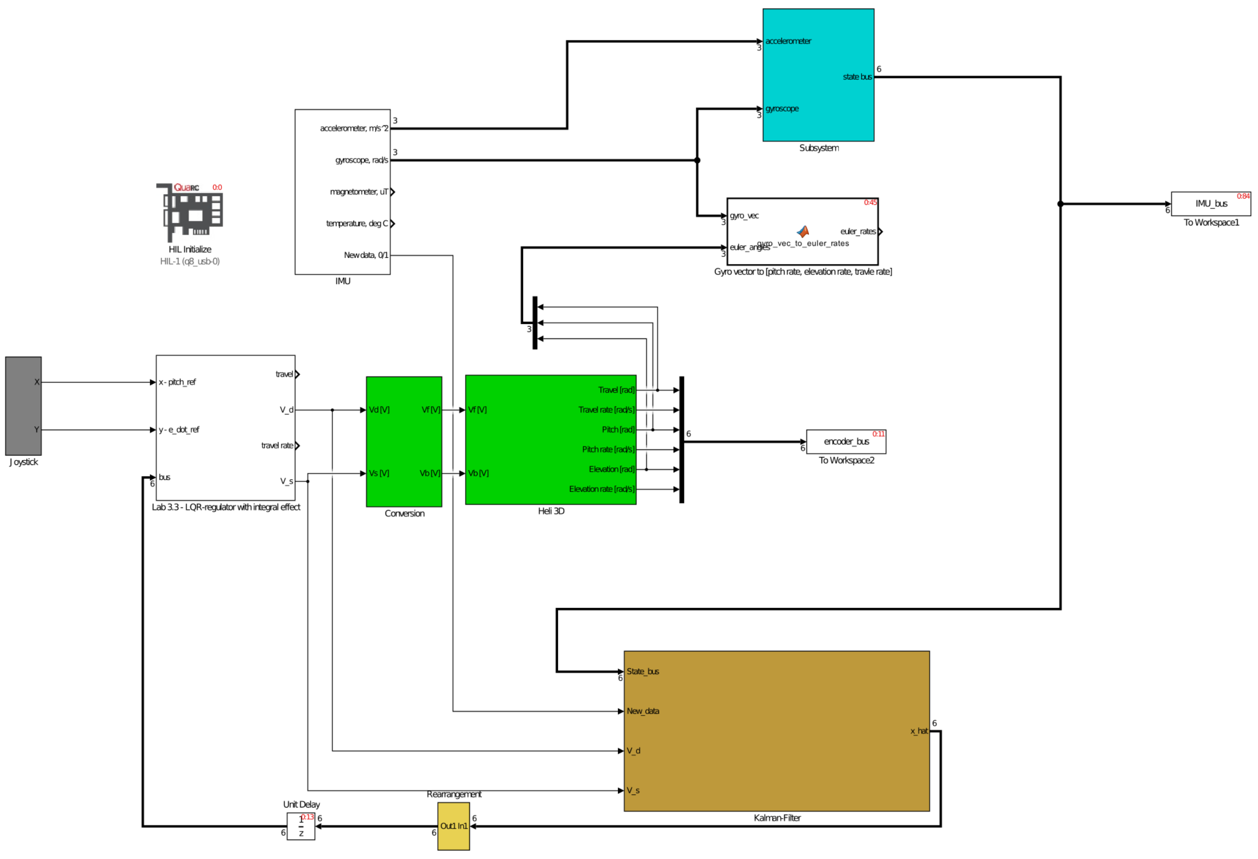
# Helicopter lab

Magnus Dyre Moe, Erlend Blomseth, Eivind Heldal Stray

Fall 2019



## 1 Part 1 - Modeling, validation and manual control

The helicopter is described by Figure 1 and has three degrees of freedom. Thrust from the propellers give us lift, $e$, different thrust on each propeller gives us roll, $p$, here referred to as pitch, and finally travel, $\lambda$, around the center.

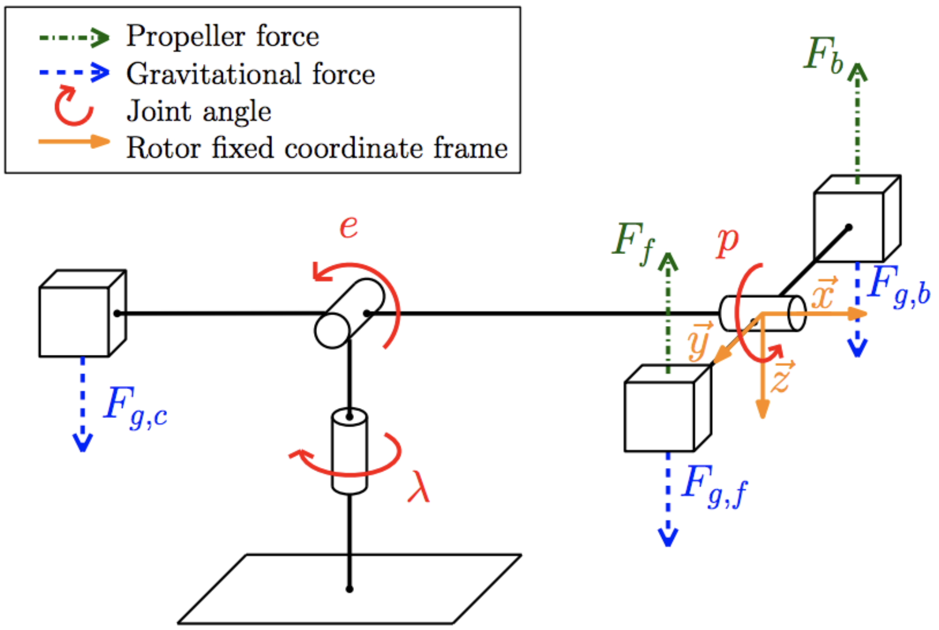


Figure 1: The helicopter

Table 1: Constant helicopter values.

| Symbol | Parameter | Value | Unit |
|---|---|---|---|
| g | Gravitational acceleration | 9.81 | $\frac{m}{s^2}$ |
| l$_p$ | Distance from pitch axis to motor | 0.175 | $m$ |
| l$_c$ | Distance from elevation axis til counterwigth | 0.46 | $m$ |
| l$_h$ | Distance from elevation axis to helicopter load | 0.66 | $m$ |
| J$_p$ | Moment of inertia for pitch | 0.044 | kg $m^2$ |
| J$_e$ | Moment of inertia for elevation | 1.03 | kg $m^2$ |
| J$_\lambda$ | Moment of inertia for travel | 1.08 | kg $m^2$ |
| m$_p$ | Motor mass | 0.72 | $kg$ |
| m$_c$ | Counterweight mass | 1.92 | $kg$ |

The values contained in Table 1 will be used throughout this lab.

We are given equations for moments of inertia

$$J_p = 2m_p l_p^2 \tag{1}$$
$$J_e = m_c l_c^2 + 2m_p l_h^2 \tag{2}$$
$$J_\lambda = m_c l_c^2 + 2m_p(l_h^2 + l_p^2), \tag{3}$$

where $J_p$, $J_e$ and $J_\lambda$ are the moments of inertia for the degrees of freedom described above.

The propeller forces $F_f$ (front propeller) and $F_b$ (back propeller) are given by a motor force constant $K_f$ and applied voltage $V_f$ or $V_b$. This gives us the following equations

$$F_f = K_f V_f \tag{4}$$

$$F_b = K_f V_b. \tag{5}$$

The sum of forces $F_f + F_b$ developed by the two propellers determines the lift of the helicopter and the difference between forces $F_b - F_f$ is proportional to the torque around the pitch axis. We also define the sum of voltage and the difference in voltage as

$$V_s = V_b + V_f \tag{6}$$

$$V_d = V_b - V_f. \tag{7}$$

## 1.1 Equations of motion

We assume $cos(x) \approx 1$ and $sin(x) \approx x$ for small angles. This assumption makes it possible for us to make a linear model of the system, which is key in order to control it for later exercises.

In order to model our system we use Newton's law for rotation

$$J\ddot{\theta} = \sum \tau. \tag{8}$$

The pitch is controlled by the difference between the forces created by the propellers,

$$\begin{aligned} J_p\ddot{p} =& l_p(F_b - F_f) \\ =& l_p K_f (V_b - V_f) \\ =& l_p K_f V_d \\ =& L_1 V_d, \end{aligned} \tag{9}$$

where $L_1 = l_p K_f$. $K_f$ is the motor constant and $l_p$ is the distance from pitch axis to the propellers.

The elevation of the helicopter is caused by the forces counteracting gravity,

$$\begin{aligned} J_e\ddot{e} =& l_h \cos(p)(F_f + F_b) - l_h \cos(e)(F_{g,f} + F_{g,b}) + l_c F_{g,c} \cos(e) \\ =& L_2(V_s - V_{s,0}), \end{aligned} \tag{10}$$

where

$$L_2 \approx l_h K_f \tag{11}$$

$$V_{s,0} \approx \frac{l_h(F_{g,f} + F_{g,b}) + l_c F_{g;c}}{L_2}. \tag{12}$$

$l_h$ is the distance from the elevation axis to the helicopter load and $l_c$ is the distance from the elevation axis to the counterweight. $F_{g,f} = F_{g,b} = m_c g$ where $m_c$ is the propeller mass and $g$ is gravity.

The travel angle is determined by the sum of force of the propellers times the displacement in pitch

$$\begin{aligned} J_\lambda \ddot{\lambda} &= l_h(F_f + F_b)\sin(p) \\ &\approx l_h K_f V_s p \\ &= L_3 V_s p, \end{aligned} \tag{13}$$

where $L_3 = l_h K_f$. $K_f$ is the motor force constant and $l_h$ is the distance from the elevation axis to the helicopter load.

Furthermore we define the following:

$$\ddot{p} = K_1 V_d \tag{14}$$
$$\ddot{e} = K_2 \tilde{V_s} \tag{15}$$
$$\ddot{\lambda} = K_3 p, \tag{16}$$

where

$$K_1 = \frac{L_1}{J_p} \tag{17}$$
$$K_2 = \frac{L_2}{J_e} \tag{18}$$
$$K_3 = \frac{L_3 V_s}{J_\lambda} \tag{19}$$
$$\tilde{V_s} = V_s - V_{s,0}. \tag{20}$$

## 1.2 Parameter identification

From the previous exercise we have managed to find a formula for $V_{s,0}$ which contains $L_2$ and therefore $K_f$. When flying the helicopter at equilibrium we found $V_{s,0}$ to be 8V. This gave us $K_f = 3.4$. These values for $V_{s,0}$ and $K_f$ will be used throughout the remainder of the lab.

## 1.3 Verification and manual control

By flying the helicopter for roughly 12 seconds, we plot and compare the values from the encoder against the modelled values calculating pitch, elevation and travel from the input voltage. It is clear from the plots in Figure 2, 3 and 4 when the helicopter is near the linearization point and when it is not.

When the helicopter does not fly around the linearization point, meaning with the helicopter flying only in a proximity close from center, the model is no longer correct. A key assumption for our model is that the angles are small, since we treat $cos(x) \approx 1$ and $sin(x) \approx x$ for small values of $x$. This means that once the helicopter moves away from equilibrium we have an error. Because we are measuring the angular acceleration we need to integrate twice in order to get angular position. Since travel is dependant on pitch we integrate four times in order to obtain angular position for travel. The result of this is that our error gets integrated and therefore increasing with time. We can clearly see the result of this in Figure 2, 3 and 4.

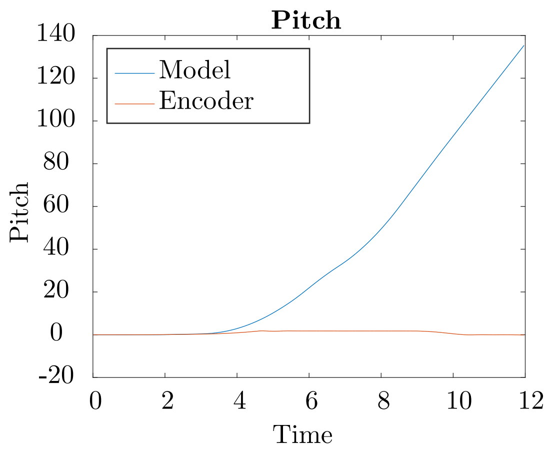


Figure 2: Pitch

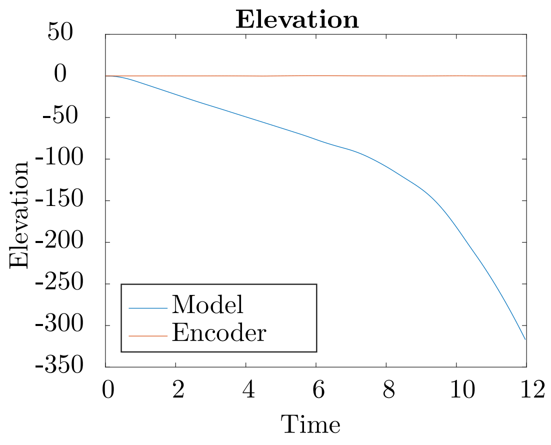


Figure 3: Elevation

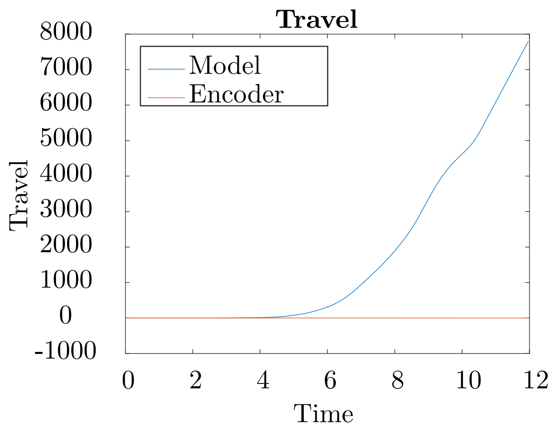


Figure 4: Travel

## 2 Part 2 - Mono-variable control

### 2.1 PD controller

We are given the PD-controller $V_d = K_{pp}(p_c - p) - K_{pd}\dot{p}$ where $K_{pp}, K_{pd} > 0$ and $p_c$ is the desired reference for the pitch angle $p$. From (14) we have $\ddot{p} = K_1 V_d$. Inserting $V_d$ into the equation and applying the Laplace transform we get

$$\frac{1}{K_1} p s^2 = K_{pp}(p_c - p) - K_{pd} p s, \tag{21}$$

resulting in the transfer function

$$G(s) = \frac{p}{p_c}(s) = \frac{K_1 K_{pp}}{s^2 + sK_1 K_{pd} + K_1 K_{pp}}. \tag{22}$$

The transfer function is realized in MATLAB with the `tf()` function:

```
numerator = K_1 * K_pp
denominator = [1, K_1*K_pd, K_1*K_pp]
sys = tf(numerator, denominator)
```

Because $K_{pp}, K_{pd} > 0$ and $K_1 > 0$ the system will always be stable, in theory. The Routh-Hurwitz test for stability, which says that the system is stable while we have a denominator on the form $a_2 s^2 + a_1 s + a_0$ if $a_2, a_1, a_0 > 0$, backs our claim that the transfer function will always be stable. However, this is in theory. We will examine our transfer function in the following exercises and see if it is indeed stable for all $K_{pp}, K_{pd} > 0$.

### 2.2 Pole placement

With the transfer function from the previous exercise we find the poles

$$\lambda_{1,2} = \frac{-K_1 K_{pd} \pm \sqrt{(K_1 K_{pd})^2 - 4K_1 K_{pp}}}{2}. \tag{23}$$

From here we were able to design our system using pole placement. We tried the poles in Table 2.

Table 2: Pole placement

| Trial | $\lambda_1$ | $\lambda_2$ | Theoretical stability | Observational stability |
|---|---:|---:|---|---|
| 1 | $-1$ | $-1$ | Critically damped | Unstable |
| 2 | $-2$ | $-2$ | Critically damped | Unstable |
| 3 | $-3$ | $-3$ | Critically damped | Underdamped |
| 4 | $-4$ | $-4$ | Critically damped | Underdamped |
| 5 | $-20$ | $-20$ | Critically damped | Unstable |
| 6 | $-5.2i$ | $5.2i$ | Undamped | Undamped |
| 7 | $-0.7 - 7.3i$ | $-0.7 + 7.3i$ | Underdamped | Undamped |
| 8 | $-6.8 - 2.9i$ | $-6.8 + 2.9i$ | Underdamped | Underdamped $\rightarrow$Critically damped |
| 9 | $-17.1$ | $-3.2$ | Overdamped | Overdamped |

To start off we decided to test the transfer function on the helicopter with a double pole at $-1$. In theory this should give us a critically damped response. However, the response started oscillating and quickly became unstable. Therefore we tried with double poles at $-2$ but the system was still unstable.

Therefore we tried a double pole at $-3$. This should in theory give a critically damped response faster than the previous double pole at $-2$. This time the system was stable, however, not critically damped.

Nevertheless, the response was quick. Then for double poles at $-4$ the system was still stable and faster than with poles at $-3$ which is to be expected.

We find these results to be peculiar. It is clear to us that our system goes from unstable to stable somewhere in between poles at $-2$ and $-3$. We assume this is a consequence of our model not being excactly like the real world. Some constants used to derive the transfer funciton could be different from reality, causing the system to behave a bit differently from what we would expect. Furthermore, the actuators have limitations that can cause different behaviour from what the theoretical model would give. We find our results to be strange.

Because of the previous results we were curious to see what would happen with poles at $-20$. The response was incredibly fast, maybe too fast, because it became unstable. We assume this is due to saturation.

We decided to check how the system would react with poles on the imaginary axis. Therefore we tried $K_{pd} = 0$ and $K_{pp} = 2$. This gave us poles at $\pm \frac{2}{K_1} i \approx \pm 5.2i$. The system reacted exactly how we thought it would, undamped as a harmonic oscillator should.

Our next test was to see how the system would react with poles wide on the imaginary axis while still being in the left half plane. We decided to check with poles at $-0.7 \pm 7.3i$. This should make for an underdamped response. However, the response was similar to our previous test with poles on the imaginary axis. We believe this comes down to physical limitations or the ratio between damping and resonance frequency.

For our next test we wish to examine the response of complex poles further left in the left half plane. We therefore set $K_{pd} = 1$ and $K_{pp} = 4$, resulting in poles at $-6.8 \pm 2.9i$. As theory would suggest this would giva an underdamped response, however the response was as close to critically stable as we have reached. The response was slightly different when trying with these poles. For our last couple of tries the system was critically damped, hence the arrow pointing from underdamped to critically damped.

For our last test we want to examine poles on the real axis, resulting in an overdamped response. $K_{pd} = 1.5$ and $K_{pp} = 4$ resulted in poles at $-17.1$ and $-3.2$. This resulted in a slower response, however it was smooth and steady.

## 2.3 Harmonic oscillator

We can view our transfer function as an harmonic oscillator on the following form

$$G(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 + \omega_0^2}. \tag{24}$$

By looking at our transfer function we can define the following

$$\begin{aligned} \omega_0^2 &= K_1 K_{pp}, & (25)\\ 2\zeta\omega_0 &= K_{pd} K_1, & (26)\\ \zeta &= \frac{K_1 K_{pd}}{2\sqrt{K_1 K_{pp}}}. & (27) \end{aligned}$$

The damping ratio, $\zeta$, is critical to analyze in order to have a system behaving in a desirable way. From theory we know that $\zeta = 0$ gives an undamped response, $\zeta < 1$ gives an underdamped/oscillating response, $\zeta = 1$ gives a critically damped response and $\zeta > 1$ gives an overdamped response. Resonance frequency describes the speed of our system. The greater the resonance frequency the faster the system reacts to change. When this information is clear we can start to examine our transfer function by altering $\omega_0$ and $\zeta$.

We examine the same poles as we tested in the previous exercise in regard to $\omega_0$ and $\zeta$. Table 3 contains the calculated $\omega_0$ and $\zeta$ for all the poles, with the trial number corresponding to the trial number in Table 2.

Table 3: Harmonic oscillator

| Trial | $\zeta$ | $\omega_0$ | Theoretical stability | Observational stability |
|---|---|---|---|---|
| 1 | 1 | 1 | Critically damped | Unstable |
| 2 | 1 | 2 | Critically damped | Unstable |
| 3 | 1 | 3 | Critically damped | Underdamped |
| 4 | 1 | 4 | Critically damped | Underdamped |
| 5 | 1 | 20 | Critically damped | Unstable |
| 6 | 0 | 5.2 | Undamped | Undamped |
| 7 | 0.10 | 7.33 | Underdamped | Undamped |
| 8 | 0.92 | 7.39 | Underdamped | Underdamped ->Critically damped |
| 9 | 1.38 | 7.39 | Overdamped | Overdamped |

In the first five tests we see that $\omega_0$ increases from test to test, while $\zeta = 1$. This should in theory make for a faster system. In the first two test with $\omega_0 = 1$ and $\omega_0 = 2$ the system was unstable, which we find to be strange. In section 2.2 we discussed the possible limitations our helicopter faces. However it could also come down to the resonance frequency. When we start the helicopter laying still on the table it could be that the system is to slow for it to become stable. Nevertheless we still discovered that the system became quicker by increasing $\omega_0$. For $\omega_0 = 20$ it was too quick, though, as the system reached its physical limitations, which we assume to be saturation. This shows us that $\omega_0$ determines how fast the system reacts.

When flying the helicopter with $\zeta = 0$ the system was undamped, to no surprise. However it also behaved undamped with imaginary poles in the left half plane with $\zeta = 0.10$. We assume this comes down to the ratio between resonance frequency and damping. If so it could be such that the damping in the system plays no actual effect and therefore the system behaves like a harmonic oscillator.

When trying with a damping factor close to one and a high resonance frequency the system was as close to critical stability as we reached. The system was quick and stable, to no surprise. There were only slight differences when launching the helicopter with $\zeta = 0.92$ and $\omega_0 = 7.39$. This makes it difficult for us to decide whether it is underdamped or critically damped. In our last try it was critically damped, hence the arrow pointing from underdamped to critically damped in Table 3.

For $\zeta = 1.38$ the system behaved exactly as to be expected, slow and steady.

For section 2.2 and 2.3, we have seen that the system does not behave quite as we would have expected. A final observation is that when we start the helicopter, we do so from a position where the helicopter is laying on the table. This is quite far from the linarization point, and this could contribute to the unstable behaviour seen with poles closer to zero than what we found to be stable.

## 3 Part 3 - Multi-variable control

### 3.1 State space formulation

We wish to look at our system with multi-variable control on state-space form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{28}$$

where $\mathbf{x}$, $\mathbf{u}$ are defined as

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} \quad \text{and} \quad \mathbf{u} = \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix}, \tag{29}$$

which gives the following state-space model

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix}, \tag{30}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \end{bmatrix}. \tag{31}$$

When we examine the controllability of the system we examine

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \mathbf{A}^2\mathbf{B} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & K_1 & 0 & 0 \\ 0 & K_1 & 0 & 0 & 0 & 0 \\ K_2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{32}$$

Since $rank(\boldsymbol{\mathcal{C}}) = 3$ we conclude that the system is controllable.

### 3.2 Linear Quadratic Regulators

Our goal is to control the helicopter with the joystick. Therefore we wish to track the reference $\mathbf{r} = \begin{bmatrix} p_c \\ \dot{e}_c \end{bmatrix}$. We then implement a linear state-feedback controller with reference-feed-forward

$$\mathbf{u} = \mathbf{F}\mathbf{r} - \mathbf{K}\mathbf{x} \tag{33}$$

The matrix $\mathbf{K}$ is determined by the linear quadratic regulator algorithm, which means that the control input $\mathbf{u} = -\mathbf{K}\mathbf{x}$ optimizes the following cost function:

$$J = \int_0^\infty \mathbf{x}^\mathbf{T}(t)\mathbf{Q}_{LQR}\mathbf{x}(t) + \mathbf{u}^\mathbf{T}(t)\mathbf{R}_{LQR}\mathbf{u}(t)dt. \tag{34}$$

By choosing $\mathbf{Q}_{LQR}$ and $\mathbf{R}_{LQR}$ we can choose the cost of each element in the states, controlling how we wish for our helicopter to behave.

When choosing $\mathbf{F}$ we examine the following equation:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ &= \mathbf{A}\mathbf{x} + \mathbf{B}(\mathbf{F}\mathbf{r} - \mathbf{K}\mathbf{x}) \\ &= (\mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{x} + \mathbf{B}\mathbf{F}\mathbf{r}. \end{aligned} \tag{35}$$

Our goal by using a reference-feed-forward is to make our helicopter follow our reference. When doing so $\mathbf{y}$ approaches the reference values in $\mathbf{r}$. When we are at our reference we are at equilibrium such that $\dot{\mathbf{x}} = \mathbf{0}$. When also considering that $\mathbf{x} = \mathbf{C^{-1}y}$ we solve the following equation:

$$\begin{aligned}(\mathbf{BK} - \mathbf{A})\mathbf{C^{-1}y} &= \mathbf{BFr} \\ \mathbf{y} &= \mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{BFr} \\ \mathbf{y} &= \mathbf{Ir},\end{aligned} \tag{36}$$

resulting in

$$\mathbf{F} = (\mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{B})^{-1}. \tag{37}$$

When implementing the LQR, we have to give the system appropriate weights in the $\mathbf{Q}_{LQR}$ and $\mathbf{R}_{LQR}$ matrices. By increasing or decreasing the different inputs in our matrices we give the different components different costs. Tuning the regulator, we found that increasing the cost of $\dot{e}$ allowed critical damping, yet quick response. Furthermore, tuning $\dot{p}$ and $p$ proved easiest by simply testing a set of different values until it was as critically damped and as quick as possible. We found that the following matrices works well for us:

$$\mathbf{Q}_{LQR} = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 30 \end{bmatrix} \quad \text{and} \quad \mathbf{R}_{LQR} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{38}$$

In MATLAB, the LQR was implemented with the `lqr()` function and matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{R}$ corresponding to the matrices in equations (31) and (38):

```
K = lqr(A,B,Q,R)
```

## 3.3 Integral action

For this part we have added two new differentiated states:

$$\dot{\gamma} = p - p_c \quad \text{and} \quad \dot{\zeta} = \dot{e} - \dot{e_c}. \tag{39}$$

This means we now have five different states, giving us the following state-space

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} + \mathbf{R_{ref}r} \tag{40}$$

The state-space can be described in the following way:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{p} \\ \ddot{p} \\ \ddot{e} \\ \dot{\gamma} \\ \dot{\zeta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ \dot{e} \\ \gamma \\ \zeta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} p_c \\ \dot{e_c} \end{bmatrix}. \tag{41}$$

Solving the equation in the same way as in the previous exercise we obtain the following equation

$$\mathbf{y} = \mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{BFr} + \mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{R_{ref}r} \tag{42}$$

where $\mathbf{C}$ is a $2 \times 5$ matrix because we wish to track the reference. Using MATLAB to evaluate the equation, we get

$$\mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{BF} = \mathbf{0} \quad \text{and} \quad \mathbf{C}(\mathbf{BK} - \mathbf{A})^{-1}\mathbf{R}_{ref} = \mathbf{I}, \tag{43}$$

where $\mathbf{0}$ and $\mathbf{I}$ are $2 \times 2$ matrices. Knowing this we see that $\mathbf{F}$ has infinite solutions since we can assign $\mathbf{F}$ any value without affecting the system.

After implementing integral effect we noticed that the helicopter followed the reference we sat by using the joystick. This result is not surprising since we track our reference and adapt our system to approach it. With the introduction of integral effect the system eliminates standard deviation which makes for an accurate form of control. We ended up with the following matrices for $\mathbf{Q}_{LQR}$ and $\mathbf{R}_{LQR}$:

$$\mathbf{Q}_{LQR} = \begin{bmatrix} 30 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{R}_{LQR} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{44}$$

For an energy efficient system limiting the system input, $\mathbf{R}_{LQR}$, by upping the cost, would be a wise choice. For our purpose we wish to have a quick responding system, hence no cost on the input.

## 4 Part 4 - Inertial Measurement Unit (IMU)

### 4.1 IMU characteristics

Comparing the gyro to encoder readings, we discovered a few interesting facts. With the helicopter in level flight, readings from each direction (pitch, elevation and travel) were nicely aligned. However, turning the helicopter 90 degrees, the encoder reads an upwards motion as an increase in elevation while the IMU displayed travel and vice versa for encoder travel and IMU elevation. This is displayed in Figure 6 and 8 where we can see the IMU elevation and encoder travel are aligned between 6 and 10 seconds. This is a direct result of the fact that the IMU sensors are placed on the beam leading to the rotors, and not at individual and suitable places.

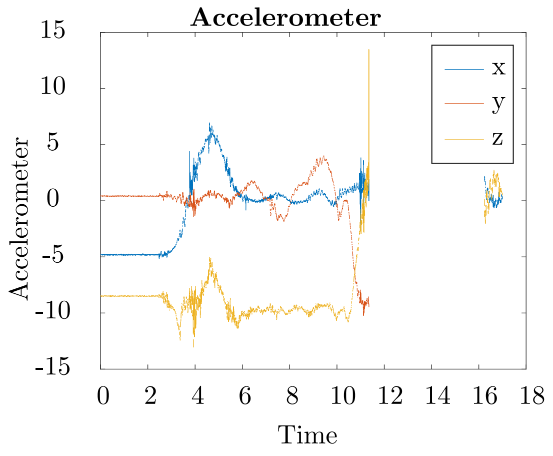


Figure 5: Accelerometer

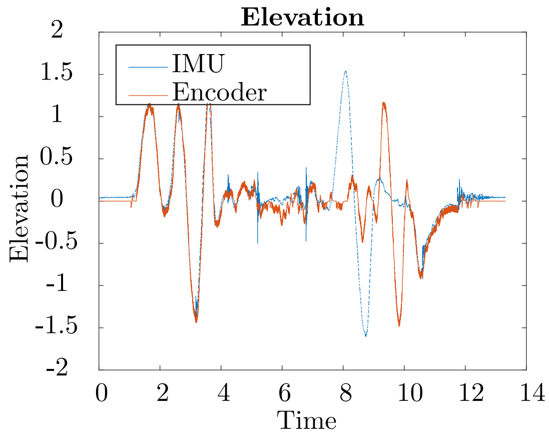


Figure 6: Elevation

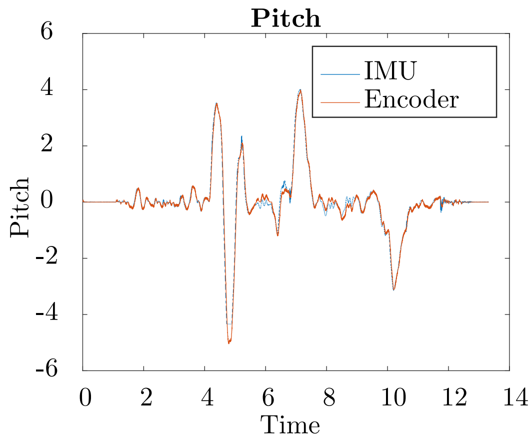


Figure 7: Pitch

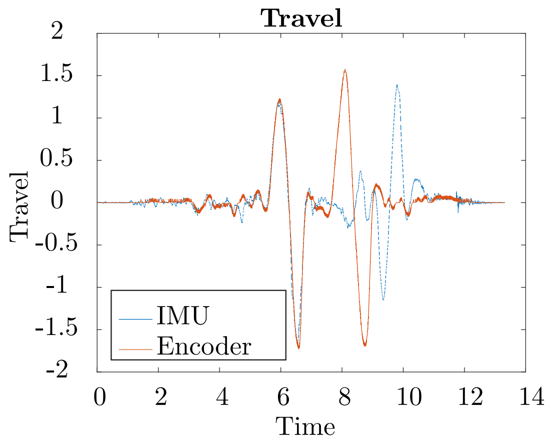


Figure 8: Travel

### 4.2 Gyroscope transform

We were given a Simulink block in the "IMU-slx" file to counteract the different measurements when the helicopter is rotated around the pitch angle. However we did not find any use in this block as our system worked nicely without using it. Actually it behaved weird when we used the Euler block and the measurements were way worse. Therefore we decided not to use this block for later assignments. We have, however, demonstrated the Kalman Filter both with and without the use of the Euler block to demonstrate why it was better for us to avoid the Euler block.

## 4.3 Observability

In this assignment we wish to measure velocity of pitch, elevation and travel. We are given the states

$$\mathbf{y}_{gyro} = \begin{bmatrix} \dot{p} \\ \dot{e} \\ \dot{\lambda} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \lambda \\ \dot{\lambda} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix}, \tag{45}$$

with information from previous exercises we can develop a state-space:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{p} \\ \ddot{p} \\ \dot{e} \\ \ddot{e} \\ \dot{\lambda} \\ \ddot{\lambda} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ K_3 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \lambda \\ \dot{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_1 \\ 0 & 0 \\ K_2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V_s} \\ V_d \end{bmatrix} \tag{46}$$

and

$$\mathbf{y}_{gyro} = \begin{bmatrix} \dot{p} \\ \dot{e} \\ \dot{\lambda} \end{bmatrix} = \mathbf{C}_{gyro}\mathbf{x} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ \dot{p} \\ e \\ \dot{e} \\ \lambda \\ \dot{\lambda} \end{bmatrix} \tag{47}$$

On the given form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y}_{gyro} &= \mathbf{C}_{gyro}\mathbf{x} \end{aligned} \tag{48}$$

When checking if the system is observable we see if the observability matrix is of full rank. Full rank here is the dimension of our state space, 6. The observability matrix is

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \mathbf{CA}^3 \\ \mathbf{CA}^4 \\ \mathbf{CA}^5 \end{bmatrix} \tag{49}$$

Using the rank and observability function in MATLAB gives us

$$rank(obsv(\mathbf{A}, \mathbf{C}_{gyro})) = 4. \tag{50}$$

Since we do not have full rank, the system is not observable, which is not a strange result when we are measuring velocity. When we only retrieve information about the x-, y- and z-speed, and do not know the initial position of the system in either direction, there is no way of knowing the current position of the system. Therefore we can conclude that the subset of $\mathbf{x}$ containing velocity is unobservable while the subset of $\mathbf{x}$ containing position is observable.

If we measure the subset of $\mathbf{x}$ containing position instead of velocity we get an observability matrix of full rank. By knowing the position at any given time, it is possible to find the velocity in any direction through differentiation.

When we measure velocity the system is controllable. If we were to measure position it would be observable, however, not controllable. When flying our helicopter it is obviously more important that we have a controllable system rather than an observable one.

## 4.4 Accelerometer

Every $a_i$, $i = x, y, z$, take advantage of a measure of the gravity pull. $a_i$ is the pull in i-direction. $a_x$ is the gravitational pull in the direction of the bar going through the elevation joint. $a_y$ is the pull along the bar between the propellers. $a_z$ is the vertical pull. $a_x$, $a_y$ and $a_z$ are also specified in Figure 1.

Considering this we can tell that $a_x$ has nothing to do with the pitch angle. When the helicopter is at equilibrium the pitch angle is zero. The pitch must therefore be the dislocation from the horizontal plane such that when the gravitational pull and the bar is perpendicular the pitch is zero. Therefore the pitch angle must be displacement in horizontal position divided by displacement in height giving us

$$p = arctan\left(\frac{a_y}{a_z}\right) \tag{51}$$

Elevation on the other hand is dependant on $a_x$. When the helicopter is at equilibrium the elevation angle is zero. Hence, we can tell that elevation is dependant on the position of $a_x$ in regard to where on the circle the helicopter is located. The circle here being the possible positions for the helicopter to be while rotating around its travel axis. The position on the edge of the circle is dependant on both $a_y$ and $a_z$. Using trigonometry and Pythagoras theorem the displacement on the circle edge is the hypotenuse of $a_y$ and $a_z$. Considering this and the fact that elevation is zero when at equilibrium we arrive at

$$e = arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right) \tag{52}$$

From a mathematical perspective and analyzing the helicopter we find the following

$$\begin{aligned} a_x &= sin(e) \\ a_y &= cos(e)sin(p) \\ a_z &= cos(e)cos(p) \end{aligned} \tag{53}$$

Using this we can arrive at the exact same equations for pitch and elevation

$$\begin{aligned} \frac{a_y}{a_z} &= \frac{cos(e)sin(p)}{cos(e)cos(p)} \\ &= tan(p) \\ &\implies p = arctan\left(\frac{a_y}{a_z}\right) \end{aligned} \tag{54}$$

$$
\begin{aligned}
\frac{a_x}{\sqrt{(a_y^2 + a_z^2)}} &= \frac{sin(e)}{\sqrt{(cos(e)sin(p))^2 + (cos(e)cos(p))^2}} \\
&= \frac{sin(e)}{cos(e)\sqrt{sin^2(p) + cos^2(p)}} \\
&= \frac{sin(e)}{cos(e)} \\
&= tan(e) \\
&\implies e = arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right)
\end{aligned}
\tag{55}
$$

As clearly displayed here we can arrive at the exact same equations using intuition and mathematical deduction.

## 4.5 Observability part II

For this exercise we expand our measurements to include pitch and elevation as well, giving us the following measurements

$$
\mathbf{y}_{IMU} = \begin{bmatrix} \dot{p} \\ \dot{e} \\ \dot{\lambda} \\ p \\ e \end{bmatrix} \tag{56}
$$

Giving us the following **C**-matrix

$$
\mathbf{C}_{IMU} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{57}
$$

Using the rank and observability function in MATLAB we get

$$
rank(obsv(\mathbf{A}, \mathbf{C}_{IMU})) = 5 \tag{58}
$$

which is not observable. It is clear that our measured states are observable. By reasoning, we can conclude that travel is unobservable, as we do not know the starting point at which the helicopter begins its journey around center.

## 4.6 Noise

In the plots we can see that all encoder values are zero except for pitch. This is because an offset we inserted into the system in order to make the helicopter level when flying to decrease travel.

Figure 9, 10, 11, 12 and 13 display noise from the IMU while the helicopter is on the table.
Figure 14, 15, 16, 17 and 18 display noise from the IMU when the helicopter is flying while not using the Euler block.
Figure 19, 20, 21, 22 and 23 display noise from the IMU while the helicopter is flying while using the Euler block.

From the plots we can clearly see noise. There is also a significant difference while using the Euler block and while not using it. The Euler block was supposed to eliminate the difference between the IMU and encoder, however we observe from the plots that the Euler block has in fact done the opposite.

Judging from the noise it looks to be biased, meaning it has an expected value different from zero. The signal noise does seem to be random, meaning we can not determine how it will look a second or ten from now. En example of the contrary is a sinus wave which we know the form of one second from now.

White noise has the properties of being unbiased and random. Our noise is very much random but it looks to be biased. Therefore we do not classify the noise as white.

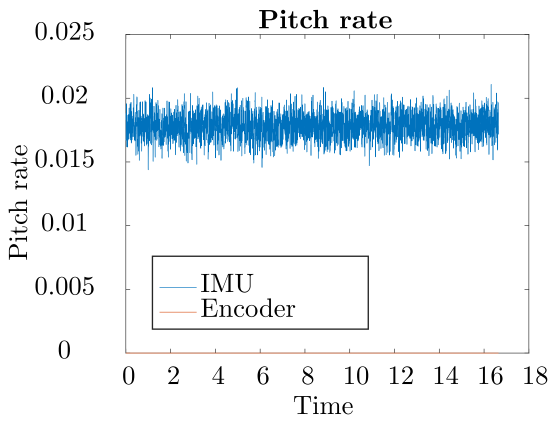


Figure 9: IMU pitch noise while the helicopter is lying still on the table.

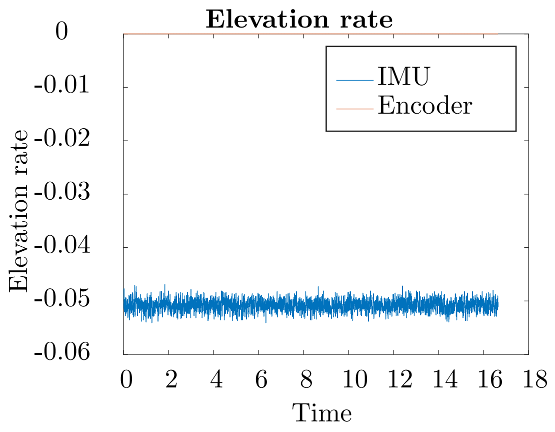


Figure 10: IMU elevation noise while the helicopter is lying still on the table.

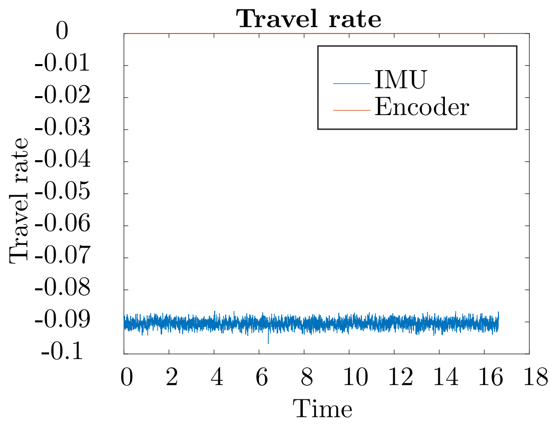


Figure 11: IMU travel noise while the helicopter is lying still on the table.

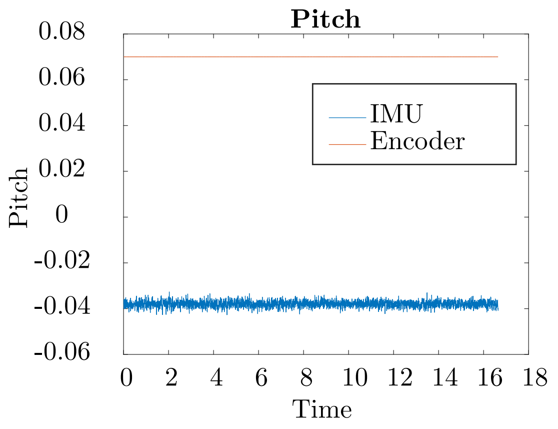


Figure 12: Accelerometer pitch noise while the helicopter is lying still on the table.

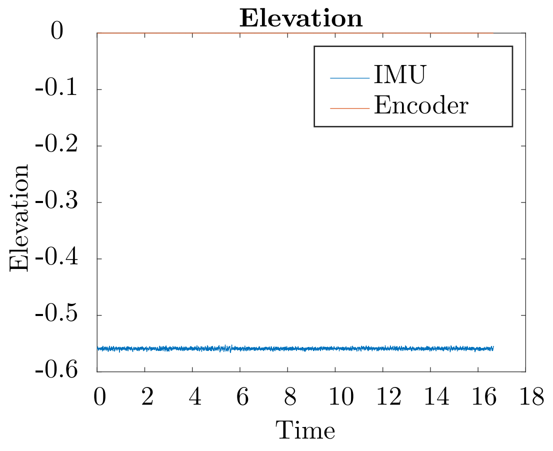


Figure 13: Accelerometer elevation noise while the helicopter is lying still on the table.

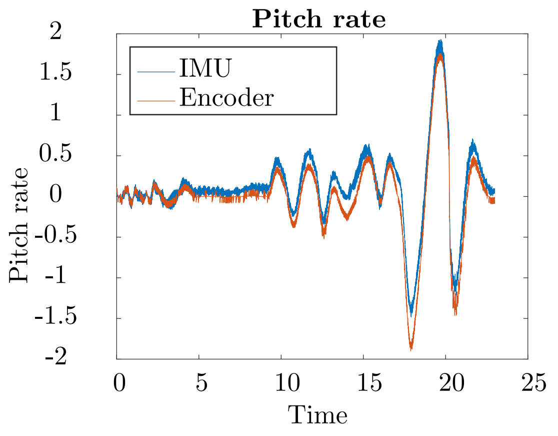


Figure 14: IMU pitch noise while the helicopter is flying without the Euler block.

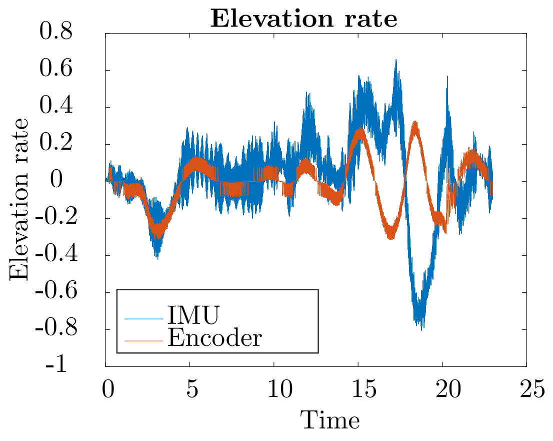


Figure 15: IMU elevation noise while the helicopter is flying without the Euler block.

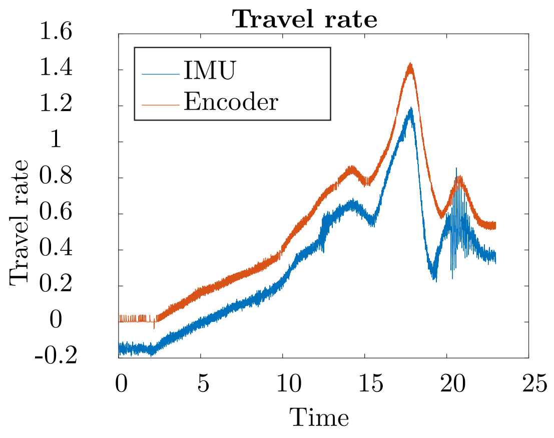


Figure 16: IMU travel noise while the helicopter is flying without the Euler block.

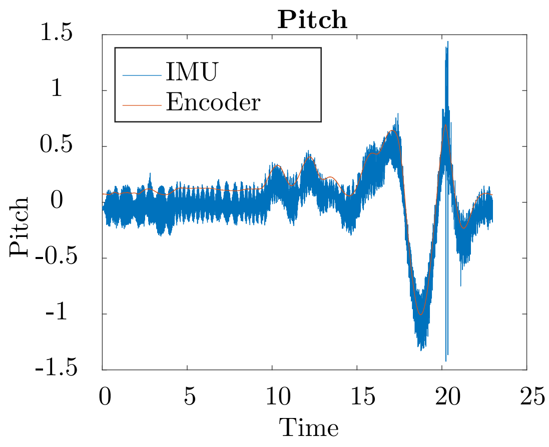


Figure 17: Accelerometer pitch noise while the helicopter is flying without the Euler block.

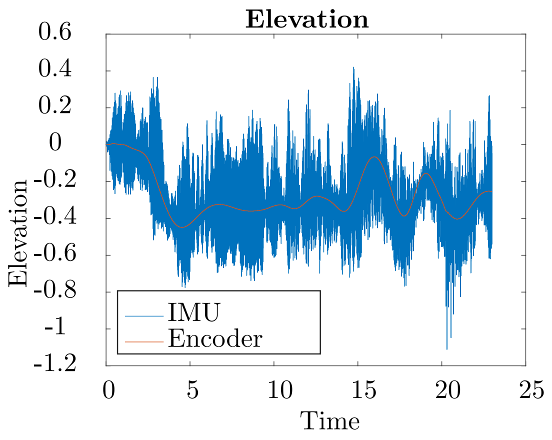


Figure 18: Accelerometer elevation noise while the helicopter is flying without the Euler block.

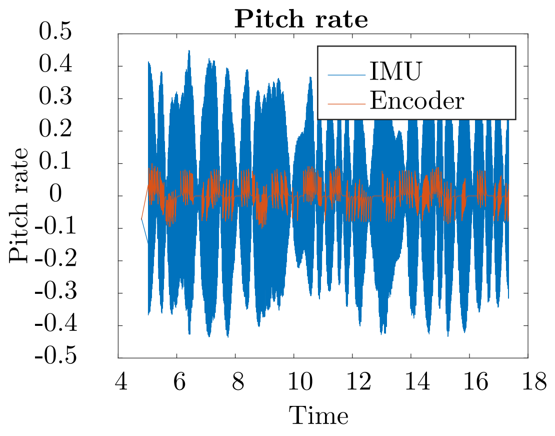


Figure 19: IMU pitch noise while the helicopter is flying with Euler rates.

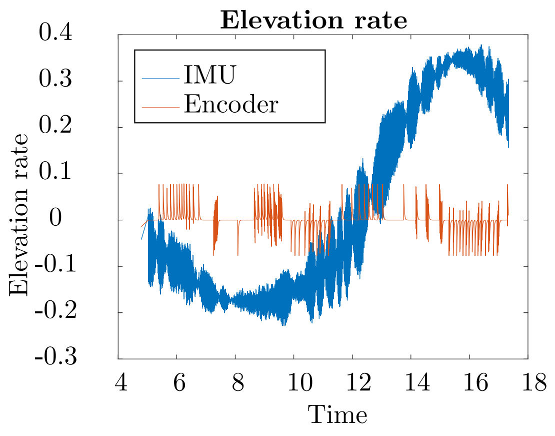


Figure 20: IMU elevation noise while the helicopter is flying with Euler rates.

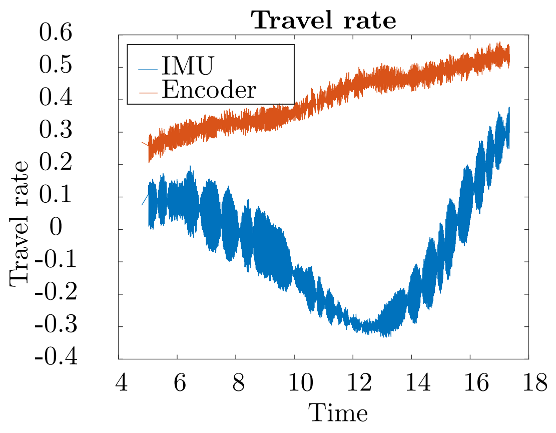


Figure 21: IMU travel noise while the helicopter is flying with Euler rates.

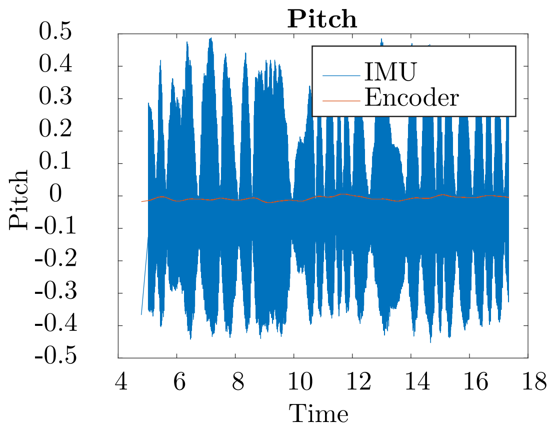


Figure 22: Accelerometer pitch noise while the helicopter is flying with Euler rates.

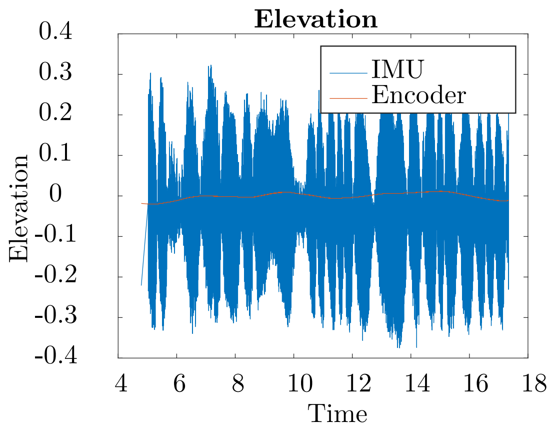


Figure 23: Accelerometer elevation noise while the helicopter is flying with Euler rates.

Finding the covariance matrix when the helicopter was laying on the table gave us

$$cov_{still}(IMU) = \begin{bmatrix} \dot{p}\dot{p} & \dot{p}\dot{e} & \dot{p}\dot{\lambda} & \dot{p}p & \dot{p}e \\ \dot{e}\dot{p} & \dot{e}\dot{e} & \dot{e}\dot{\lambda} & \dot{e}p & \dot{e}e \\ \dot{\lambda}\dot{p} & \dot{\lambda}\dot{e} & \dot{\lambda}\dot{\lambda} & \dot{\lambda}p & \dot{\lambda}e \\ p\dot{p} & p\dot{e} & p\dot{\lambda} & pp & pe \\ e\dot{p} & e\dot{e} & e\dot{\lambda} & ep & ee \end{bmatrix} = 10^{-5} \begin{bmatrix} 0.0865 & 0.0011 & 0.0008 & -0.0011 & 0.0028 \\ 0.0011 & 0.1032 & -0.0014 & -0.0008 & -0.0037 \\ 0.0008 & -0.0014 & 0.1487 & -0.0016 & 0.0017 \\ -0.0011 & -0.0008 & -0.0016 & 0.1527 & 0.0067 \\ 0.0028 & -0.0037 & 0.0017 & 0.0067 & 0.2969 \end{bmatrix} \tag{59}$$

In order to find the covariance matrix we need for later exercises we fly the helicopter at equilibrium and gather data from the IMU. We find the following covariance matrix

$$cov_{flying}(IMU) = \begin{bmatrix} \dot{p}\dot{p} & \dot{p}\dot{e} & \dot{p}\dot{\lambda} & \dot{p}p & \dot{p}e \\ \dot{e}\dot{p} & \dot{e}\dot{e} & \dot{e}\dot{\lambda} & \dot{e}p & \dot{e}e \\ \dot{\lambda}\dot{p} & \dot{\lambda}\dot{e} & \dot{\lambda}\dot{\lambda} & \dot{\lambda}p & \dot{\lambda}e \\ p\dot{p} & p\dot{e} & p\dot{\lambda} & pp & pe \\ e\dot{p} & e\dot{e} & e\dot{\lambda} & ep & ee \end{bmatrix} = \begin{bmatrix} 0.1034 & 0.0583 & 0.0373 & -0.0107 & -0.0010 \\ 0.0583 & 0.0628 & 0.0181 & -0.0136 & -0.0002 \\ 0.0373 & 0.0181 & 0.0210 & -0.0035 & -0.0009 \\ -0.0107 & -0.0136 & -0.0035 & 0.0033 & 0 \\ -0.0010 & -0.0002 & -0.0009 & 0 & 0.0007 \end{bmatrix} \tag{60}$$

It is no surprise that the covariance matrix while flying is of higher values than when the helicopter was laying flat on the table. As we can see from the graphs the values of noise is higher when flying, resulting in a higher covariance matrix. While flying, a slight change in position will result in greater noise from the IMU.

In later exercises $cov_{flying}(IMU)$ will be the covariance matrix for noise in our measurements defined as $\mathbf{R}_d$.

In order to obtain the $\mathbf{R}_d$ matrix in MATLAB, we first removed all the `Nan` fields from the IMU data, and then find the covariance of the IMU data with `cov(IMU)`:

```
IMU = [];
for i=1:length(IMU_bus.signals.values)
    if (~isnan(IMU_bus.signals.values(i)))
        IMU = [IMU; IMU_bus.signals.values(i, :)];
    end
end
R_d = cov(IMU);
```

Flying the helicopter with direct measurements we noticed that the helicopter did not reach level flight. We assume this to be because of the noise on from the IMU. Furthermore the system was stable, however it did not respond well to input from the joystick. Again, we assume this to be because of the noise from the IMU.

## 5 Prediction step

We are given following equations

$$\mathbf{x}[k+1] = \mathbf{A}_d\mathbf{x}[k] + \mathbf{B}_d\mathbf{u}[k] + \omega_d[k] \tag{61}$$
$$\mathbf{y}[k] = \mathbf{C}_d\mathbf{x}[k] + \mathbf{v}_d[k] \tag{62}$$
$$\omega_d \sim \mathcal{N}(0,\, \mathbf{Q}_d), \quad \mathbf{v}_d \sim \mathcal{N}(0,\, \mathbf{R}_d) \tag{63}$$

### 5.1 Discretization

In order to obtain the discretized system we used the MATLAB function `c2d()` with arguments **A**, **B**, **C** and sampling time $T = 0.002$s. This gives a discrete state-space model

```
sys = ss(A, B, C, 0);
sysd = c2d(sys, 0.002);
A_d = sysd.a;
B_d = sysd.b;
C_d = sysd.c;
```

### 5.2 State prediction

We wish to implement our Simulink model onto the discrete domain. Therefore we implement a state prediction using the following equation

$$\bar{\mathbf{x}}[k+1] = \mathbf{A}_d\bar{\mathbf{x}}[k] + \mathbf{B}_d\mathbf{u}[k] \tag{64}$$

After implementing the discrete function in Simulink we compared our discrete implementation with the encoder values. The graphs in Figure 24, 25 and 26 are comparisons between the encoder values and our estimated values.

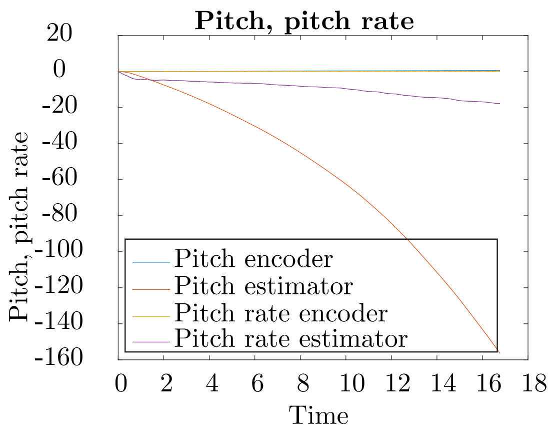


Figure 24: Pitch.

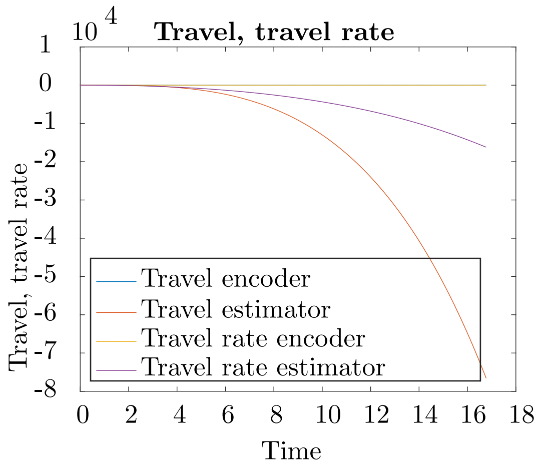


Figure 25: Travel.

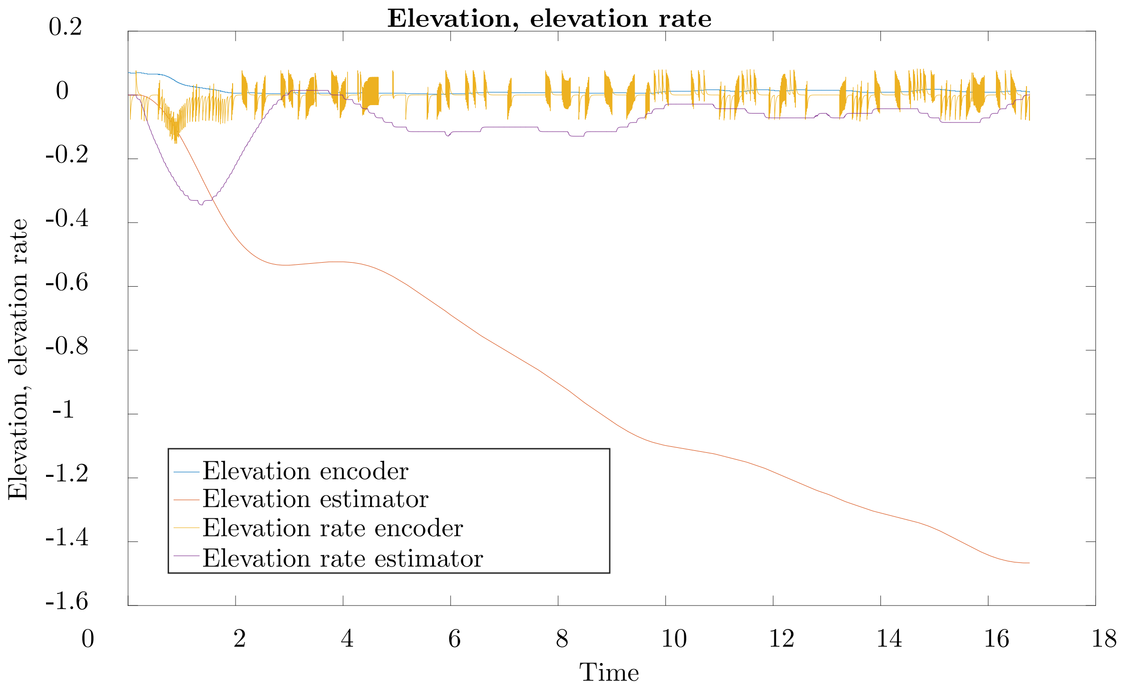


Figure 26: Elevation.

Our current prediction step is based on our previous prediction step. While the helicopter is at equilibrium this is great but once we move slightly away from equilibrium we have a state different from zero. Once the state is different from zero our problems start to occur. Because our estimator uses the last prediction step our states start to develop exponentially. The result of this can be seen in pitch and travel.

When flying the helicopter at equilibrium the helicopter was behaving nicely for some time. After a while it started to oscillate and then eventually unstable. If we do a deep dive into our open-loop feedback we can understand why. When using the open-loop estimator our next state is the result of the previous state. If our estimated state is slightly different from the actual state we have an error. Because our next state is based on our previous this error gets bigger. When the error gets bigger the system produces an input which brings our helicopter to the estimated state, which is different from the actual state. This brings the helicopter away from equilibrium. When this happens the estimator tries to compensate by producing an input to stabilize the system. However we get an error once again and the system has entered a vicious cycle. The system then becomes unstable.

Next was to fly the helicopter, starting at the table, which was unstable right away. This makes sense as our discrete model is only stable around equilibrium.

## 5.3 Prediction error covariance

Error in the open loop-estimation is defined as

$$\bar{\varepsilon} = \mathbf{x}[k] - \bar{\mathbf{x}}[k] \tag{65}$$

Where $\mathbf{x}[k]$ is the state and $\bar{\mathbf{x}}[k]$ is the estimated state.

We wish to find the covariance of $\bar{\varepsilon}[k+1]$ denoted as $\bar{\mathbf{P}}[k+1]$ given the previous time-step, $\bar{\mathbf{P}}[k]$. The covariance is defined as

$$\bar{\mathbf{P}}[k+1] = E[\bar{\varepsilon}[k+1]\bar{\varepsilon}^{\mathbf{T}}[k+1]] \tag{66}$$

First we need to find an expressions for $\bar{\varepsilon}[k+1]$. Using the formula for error in the open-loop-estimation we get

$$\begin{aligned}\bar{\varepsilon}[k+1] &= \mathbf{x}[k+1] - \bar{\mathbf{x}}[k+1] \\ &= \mathbf{A}_d\mathbf{x}[k] + \mathbf{B}_d\mathbf{u}[k] + \omega_d[k] - \mathbf{A}_d\bar{\mathbf{x}}[k] - \mathbf{B}_d\mathbf{u}[k] \\ &= \mathbf{A}_d\mathbf{x}[k] - \mathbf{A}_d\bar{\mathbf{x}}[k] + \omega_d[k] \\ &= \mathbf{A}_d\bar{\varepsilon}[k] + \omega_d[k]\end{aligned} \tag{67}$$

Such that

$$\bar{\varepsilon}[k+1] \sim \mathcal{N}(\mathbf{A}_d\bar{\varepsilon}[k],\, \mathbf{Q}_d) \tag{68}$$

Using this expression for $\bar{\varepsilon}[k+1]$ and $E[X^2] = E[X]^2 + Var[X]$ we can deduce the following equation

$$\begin{aligned}\bar{\mathbf{P}}[k+1] &= E[\bar{\varepsilon}[k+1]\bar{\varepsilon}^{\mathbf{T}}[k+1]] \\ &= E[\bar{\varepsilon}[k+1]]E[\bar{\varepsilon}[k+1]]^{\mathbf{T}} + Var[\bar{\varepsilon}[k+1]] \\ &= \mathbf{A}_d\bar{\varepsilon}[k](\mathbf{A}_d\bar{\varepsilon}[k])^{\mathbf{T}} + \mathbf{Q}_d \\ &= \mathbf{A}_d\bar{\varepsilon}[k]\bar{\varepsilon}^{\mathbf{T}}[k]\mathbf{A}_d^{\mathbf{T}} + \mathbf{Q}_d \\ &= \mathbf{A}_d\bar{\mathbf{P}}[k]\mathbf{A}_d^{\mathbf{T}} + \mathbf{Q}_d\end{aligned} \tag{69}$$

When choosing our $\mathbf{Q}_d$ in this assignment, we essentially choose how much we want the variance of $\bar{\mathbf{P}}$ to increase over time. When tuning the Kalman-filter, assigning a larger $\mathbf{Q}_d$ corresponds to trusting the measurements and therefore trusting our model less. We will choose a relationship of weighting which output to trust - the sensor or the model. Running a model over time with a variance for each sample will in the long run result in a model we cannot trust.

# 6 Correction step

## 6.1 Correction

We wish to implement an estimate, $\hat{\mathbf{x}}$, for our model. The estimate is given by

$$\begin{aligned}\hat{\mathbf{x}}[k] &= \bar{\mathbf{x}}[k] + \mathbf{K}[k](\mathbf{y}[k] - \bar{\mathbf{y}}[k]) \\ \bar{\mathbf{y}}[k] &= \mathbf{C}_d\bar{\mathbf{x}}[k]\end{aligned} \tag{70}$$

where $\mathbf{y}[k] - \bar{\mathbf{y}}[k]$ is the disagreement between prediction and measurement and $\mathbf{K}[k]$ is a weighting matrix. Combining the two equations gives us

$$\hat{\mathbf{x}}[k] = (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\mathbf{x}} + \mathbf{K}[k]\mathbf{y}[k] \tag{71}$$

When there are noe new measurements we use

$$\hat{\mathbf{x}}[k] = \bar{\mathbf{x}}[k] \tag{72}$$

We also want to use our estimated state as a feedback to our prediction giving us

$$\bar{\mathbf{x}}[k+1] = \mathbf{A}_d\hat{\mathbf{x}}[k] + \mathbf{B}_d\mathbf{u}[k] \tag{73}$$

We implement these equations in simulink on our way to making a Kalman filter.

## 6.2 Correction covariance

As in the previous section we wish to compute the covariance of the error in our estimate. The error is

$$\hat{\varepsilon}[k] = \mathbf{x}[k] - \hat{\mathbf{x}}[k] \tag{74}$$

and the covariance can be written as

$$\hat{\mathbf{P}} = E[\hat{\varepsilon}[k]\hat{\varepsilon}^{\mathbf{T}}[k]] \tag{75}$$

First we need to deduce an equation for the error

$$\begin{aligned}\hat{\varepsilon}[k] &= \mathbf{x}[k] - \hat{\mathbf{x}}[k] \\ &= \mathbf{x}[k] - (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\mathbf{x}}[k] - \mathbf{K}[k]\mathbf{y}[k] \\ &= \mathbf{x}[k] - \bar{\mathbf{x}}[k] + \mathbf{K}[k]\mathbf{C}_d\bar{\mathbf{x}}[k] - \mathbf{K}[k]\mathbf{C}_d\mathbf{x}[k] - \mathbf{K}[k]\mathbf{v}_d[k] \\ &= (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon} - \mathbf{K}[k]\mathbf{v}_d[k]\end{aligned} \tag{76}$$

Finding the expected value of the error gives us

$$\begin{aligned}E[\hat{\varepsilon}[k]] &= (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon}[k] - \mathbf{K}[k]E[\mathbf{v}_d[k]] \\ &= (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon}[k]\end{aligned} \tag{77}$$

And variance

$$\begin{aligned}Var[\hat{\varepsilon}[k]] &= Var[\mathbf{K}[k]\mathbf{v}_d[k]] \\ &= \mathbf{K}[k]Var[\mathbf{v}_d[k]]\mathbf{K}^{\mathbf{T}}[k] \\ &= \mathbf{K}[k]\mathbf{R}_d\mathbf{K}^{\mathbf{T}}[k]\end{aligned} \tag{78}$$

Finally giving us

$$\hat{\varepsilon}[k] \sim \mathcal{N}((\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon}[k],\ \mathbf{K}[k]\mathbf{R}_d\mathbf{K}^{\mathbf{T}}[k]) \tag{79}$$

Combining these equations with previous known equation we can deduce a formula for $\hat{\mathbf{P}}[k]$

$$\begin{aligned}\hat{\mathbf{P}}[k] &= E[\hat{\varepsilon}[k]\hat{\varepsilon}^{\mathbf{T}}[k]] \\ &= E[\hat{\varepsilon}[k]]E[\hat{\varepsilon}[k]]^{\mathbf{T}} + Var[\hat{\varepsilon}[k]] \\ &= (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon}[k]((\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\varepsilon}[k])^{\mathbf{T}} + \mathbf{K}[k]\mathbf{R}_d\mathbf{K}^{\mathbf{T}}[k] \\ &= (\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)\bar{\mathbf{P}}[k](\mathbf{I} - \mathbf{K}[k]\mathbf{C}_d)^{\mathbf{T}} + \mathbf{K}[k]\mathbf{R}_d\mathbf{K}^{\mathbf{T}}[k]\end{aligned} \tag{80}$$

When there are no new measurements available we use

$$\hat{\mathbf{P}}[k] = \bar{\mathbf{P}}[k] \tag{81}$$

We also want to use our estimated covariance as a feedback to our prediction giving us

$$\bar{\mathbf{P}}[k+1] = \mathbf{A}_d\hat{\mathbf{P}}[k]\mathbf{A}_d^{\mathbf{T}} + \mathbf{Q}_d \tag{82}$$

## 6.3 Weighting matrix K

We want to design our $\mathbf{K}[k]$ such that $\mathbf{K}[k]$ minimizes the error covariance. The Kalman filter only considers the variances and weight them equally. This gives us the following equation

$$var(\hat{\varepsilon}_p[k]) + var(\hat{\varepsilon}_{\dot{p}}[k]) + var(\hat{\varepsilon}_e[k]) + var(\hat{\varepsilon}_{\dot{e}}[k]) + var(\hat{\varepsilon}_\lambda[k]) + var(\hat{\varepsilon}_{\dot{\lambda}}[k]) + = tr(\hat{\mathbf{P}}[k]) \tag{83}$$

Also, because $\bar{\mathbf{P}}[k]$ is the estimated covariance matrix we know

$$\bar{\mathbf{P}}[k] = \bar{\mathbf{P}}^{\mathbf{T}}[k] \tag{84}$$

We are given following properties of trace

$$\frac{\partial tr(\mathbf{AB})}{\partial \mathbf{A}} = \mathbf{B}^{\mathbf{T}} \quad , \quad \frac{\partial tr(\mathbf{ACA}^{\mathbf{T}})}{\partial \mathbf{A}} = 2\mathbf{AC} \tag{85}$$

$$tr(\mathbf{D}) = tr(\mathbf{D}^{\mathbf{T}}) \quad , \quad tr(\mathbf{A} + \mathbf{B}) = tr(\mathbf{A}) + tr(\mathbf{B}) \tag{86}$$

To minimize $\mathbf{K}[k]$ we use the formula

$$\frac{\partial tr(\hat{\mathbf{P}}[k])}{\partial \mathbf{K}[k]} = \mathbf{0} \tag{87}$$

Using the properties of trace and other formulas from this section we arrive at the following. For simplicity we will exclude [k] while computing the equation.

$$
\begin{aligned}
\frac{\partial tr(\hat{\mathbf{P}})}{\partial \mathbf{K}} &= \frac{\partial tr((\mathbf{I}-\mathbf{K}\mathbf{C}_d)\bar{\mathbf{P}}(\mathbf{I}-\mathbf{K}\mathbf{C}_d)^{\mathbf{T}} + \mathbf{K}\mathbf{R}_d\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} \\
&= \frac{\partial tr((\mathbf{I}-\mathbf{K}\mathbf{C}_d)\bar{\mathbf{P}}(\mathbf{I}-\mathbf{K}\mathbf{C}_d)^{\mathbf{T}})}{\partial \mathbf{K}} + \frac{\partial tr(\mathbf{K}\mathbf{R}_d\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} \\
&= \frac{\partial tr((\bar{\mathbf{P}}-\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}})(\mathbf{I}-\mathbf{K}\mathbf{C}_d)^{\mathbf{T}})}{\partial \mathbf{K}} + 2\mathbf{K}\mathbf{R}_d \\
&= \frac{\partial tr((\bar{\mathbf{P}}-\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}})(\mathbf{I}-\mathbf{C}_d^{\mathbf{T}}\mathbf{K}^{\mathbf{T}}))}{\partial \mathbf{K}} + 2\mathbf{K}\mathbf{R}_d \\
&= \frac{\partial tr(\bar{\mathbf{P}}-\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}}\mathbf{K}^{\mathbf{T}}-\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}+\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d\mathbf{T}\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} + 2\mathbf{K}\mathbf{R}_d \\
&= \frac{\partial tr(\bar{\mathbf{P}}}{\partial \mathbf{K}} - \frac{\partial tr(\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}}\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} - \frac{\partial tr(\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}})}{\partial \mathbf{K}} + \frac{\partial tr(\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}}\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} + 2\mathbf{K}\mathbf{R}_d \\
&= 0 - \frac{\partial tr(\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}}\mathbf{K}^{\mathbf{T}})}{\partial \mathbf{K}} - (\mathbf{C}_d\bar{\mathbf{P}})^{\mathbf{T}} + 2\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + 2\mathbf{K}\mathbf{R}_d \\
&= -\frac{\partial tr(\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}^{\mathbf{T}})}{\partial \mathbf{K}} - \bar{\mathbf{P}}^{\mathbf{T}}\mathbf{C}_d^{\mathbf{T}} + 2\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + 2\mathbf{K}\mathbf{R}_d \\
&= -(\mathbf{C}_d\bar{\mathbf{P}}^{\mathbf{T}})^{\mathbf{T}} - \bar{\mathbf{P}}^{\mathbf{T}}\mathbf{C}_d^{\mathbf{T}} + 2\mathbf{K}\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + 2\mathbf{K}\mathbf{C}_d \\
&= 2(-\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{K}(\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d))
\end{aligned}
\tag{88}
$$

Giving us

$$
\begin{aligned}
\mathbf{0} &= 2(-\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{K}(\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d)) \\
\mathbf{K}((\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d) &= \bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} \\
\mathbf{K} &= \bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}}(\mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d)^{-1}
\end{aligned}
\tag{89}
$$

Finally inserting [k] we get our final equation

$$
\mathbf{K}[k] = \bar{\mathbf{P}}[k]\mathbf{C}_d^{\mathbf{T}}(\mathbf{C}_d\bar{\mathbf{P}}[k]\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d)^{-1} \tag{90}
$$

In order to determine that this is in fact is the minimum we need to differentiate it one more time and show that the result is positive. If so we can conclude that we have found the minimum for $\mathbf{K}[k]$

$$
\frac{\partial^2 tr(\hat{\mathbf{P}})}{\partial \mathbf{K}^2} = \mathbf{C}_d\bar{\mathbf{P}}\mathbf{C}_d^{\mathbf{T}} + \mathbf{R}_d > 0 \tag{91}
$$

Because of the positive result we can conclude that we have found the minimum of $\mathbf{K}[k]$.

## 6.4 Tuning

While tuning the Kalman filter we noticed that using the Kalman filter is a trade off balance between quickness and noise reduction. We noticed that prioritizing less noise and more reliability in our system made for a slower response in change. Therefore finding the optimal solutions such that we can trust the IMU and still have a reactive helicopter is the goal. We also noticed it was possible to crash the helicopter using violent inputs from the joystick. After a lot of tuning i both $LQR$ and the disturbance matrix $\mathbf{Q}_d$ we ended up with matrices which gave us a balance we were satisfied with.

$$\mathbf{Q}_d = \begin{bmatrix} 0.001 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.05 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad , \quad \mathbf{Q}_{LQR} = \begin{bmatrix} 35 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 30 \end{bmatrix} \quad , \quad \mathbf{R}_{LQR} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \tag{92}$$

## 6.5 The Kalman filter

Flying with an open-loop observer was worst form of control. When flying with the open-loop estimator the helicopter is stable around equilibrium and unstable elsewhere. After starting the helicopter at equilibrium we found that it became unstable after some time. This was due to an error between our actual state and our estimated state.

When flying with the measurements directly the helicopter behaved better, but not great. The helicopter was able to stay stable but reacted poorly to input from the joystick. It was also never close to level flight. Because of stability it was way better than using the open-loop estimator.

Using the Kalman filter as a feedback to the helicopter is even better, however, not perfect. The Kalman filter allowed for flying while not at equilibrium. Although it was stable the response was slower as it needed more time to reach referenced values. We also found there was quite a lot of noise from the IMU which was tricky to account for when tuning the filter. Nevertheless it was still better than flying the helicopter with an open-loop estimator or with direct measurements.

Flying the helicopter using the encoder was clearly the best. It was able to operate far from equilibrium which was similar to the Kalman filter, but the encoder was incredibly accurate also with little to no noise. Therefore it was possible to make it quicker and still behave nicely.

To conclude, the encoder was the best form of control, however the Kalman filter worked great as an alternative. When we are not able to use encoders the Kalman filter could definitely be a suitable option.

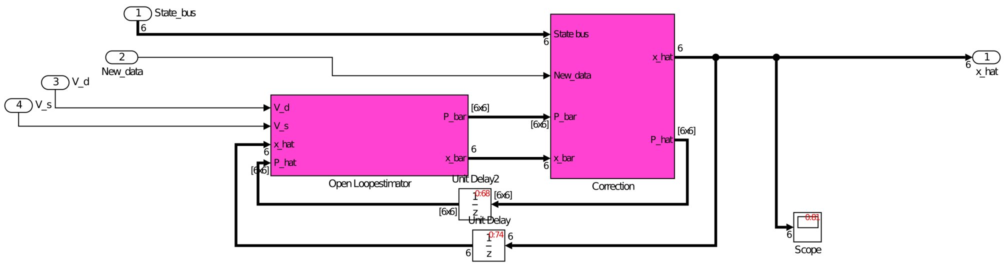


Figure 27: Overview of the Kalman filter.

## 6.6 Kalman Filtering results with and without the Euler block

Figure 28, 29, 30, 31 and 32 display the Kalman filter against encoder output with the Euler block. Figure 33, 34, 35, 36 and 37 display the Kalman filter against encoder output without the Euler block.

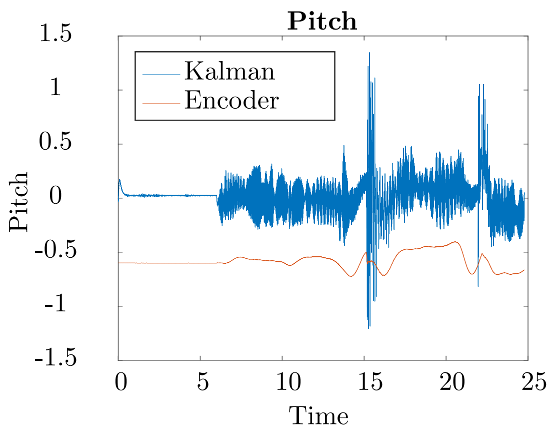


Figure 28: Pitch reading from the Kalman filter with the Euler block.

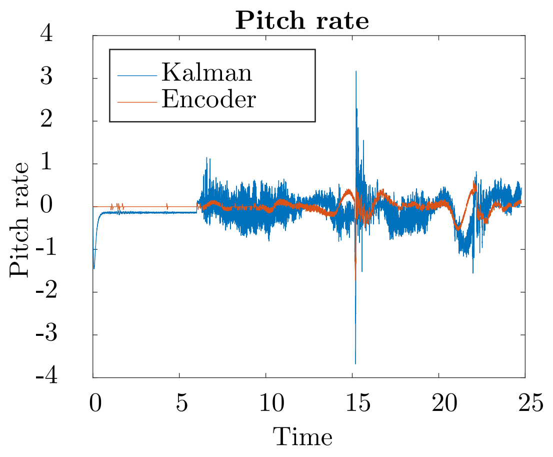


Figure 29: Pitch rate reading from the Kalman filter with the Euler block.

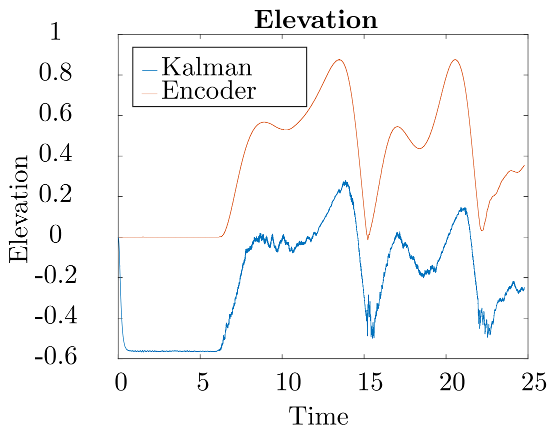


Figure 30: Elevation reading from the Kalman filter with the Euler block.

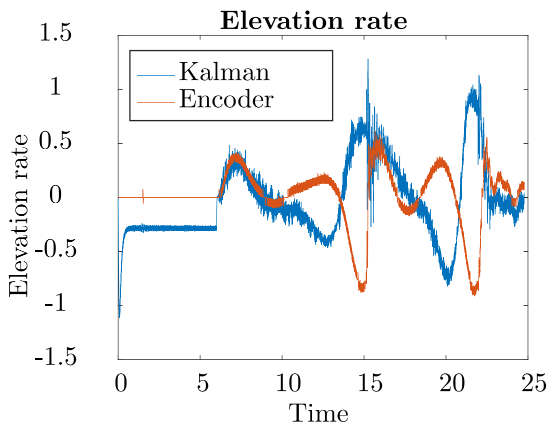


Figure 31: Elevation rate reading from the Kalman filter with the Euler block.

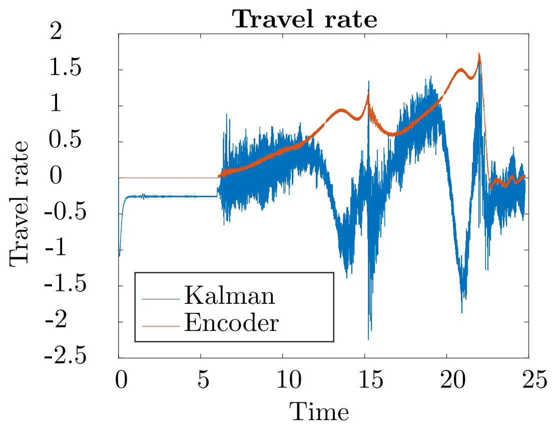
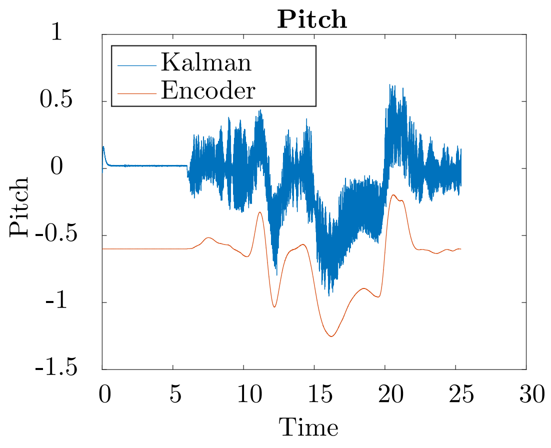


Figure 32: Travel rate reading from the Kalman filter with the Euler block.



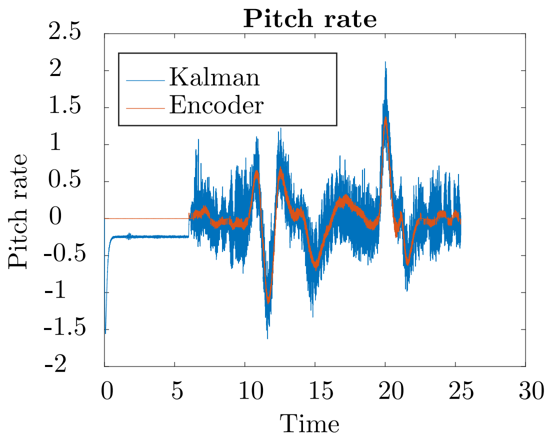


Figure 33: Pitch reading from the Kalman filter without the Euler block.

Figure 34: Pitch rate reading from the Kalman filter without the Euler block.

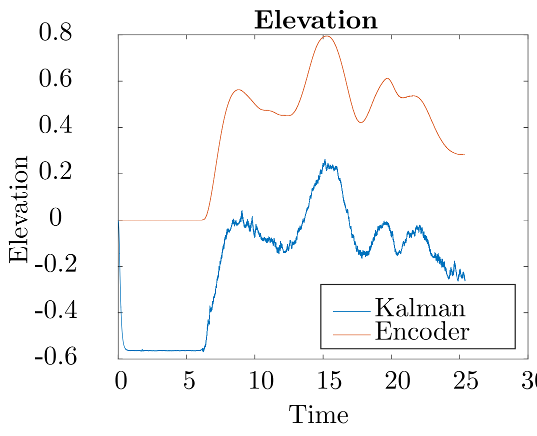


Figure 35: Elevation reading from the Kalman filter without the Euler block.

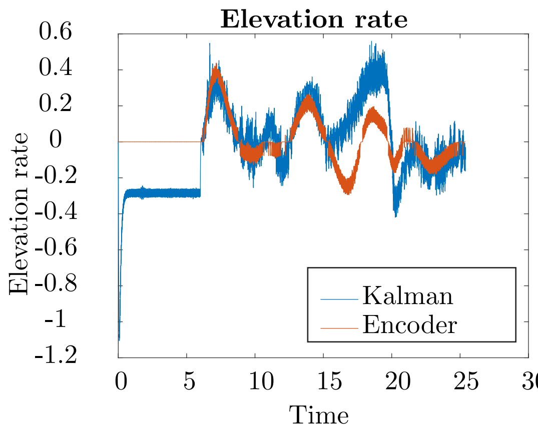


Figure 36: Elevation rate readinng from the Kalman filter without the Euler block.

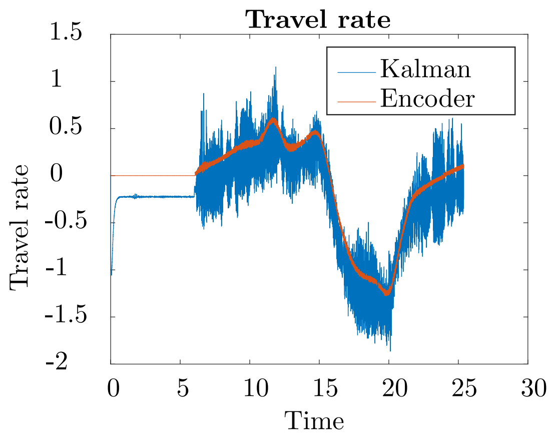


Figure 37: Travel rate reading from the Kalman filter without the Euler block.