Boat lab - 2018

Group 50
Kjetil Sekse Kristiansen - 477664
Asgeir Hunshamar - 478431

22.10.2018 - 25.11.2018

Department of Engineering Cybernetics

Contents

1 Introduction

The purpose of this project was to control and make an autopilot for a ship. The ship was first controlled through a PD-controller and then a Kalman filter was added in order to counteract the disturbances from the system.

2 Mathematical model of the system

The linearized model of the system is given by (1)

$$\begin{aligned}
\dot{\xi}_\omega &= \psi_\omega && (1a)\\
\dot{\psi}_\omega &= -\omega_0^2 \xi_w - 2\lambda\omega_0\psi_\omega + K_\omega \omega_\omega && (1b)\\
\dot{\psi} &= r && (1c)\\
\dot{r} &= -\frac{1}{T} r + \frac{K}{T}(\delta - b) && (1d)\\
\dot{b} &= \omega_b && (1e)\\
y &= \xi + \xi_\omega + v && (1f)
\end{aligned}$$

where

- ψ_ω - Is a high-frequency component due to the wave disturbance.
- ψ - Is the average heading, i.e without wave disturbance.
- r - Is the rate of change for the average heading, i.e without wave disturbance.
- b - Bias to the rudder angle.
- y - Is the measured heading.
- ω_b, ω_ω and v are white noise
- δ - Is the rudder angle relative to the body frame processes.

The system can be written as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) + \boldsymbol{E}\boldsymbol{w}(t), \quad y = \boldsymbol{C}\boldsymbol{x}(t) + v(t) \tag{2}$$

with

$$\boldsymbol{x} = \begin{bmatrix} \xi_\omega \\ \psi_\omega \\ \psi \\ r \\ b \end{bmatrix}, \quad u = \delta \text{ and } \boldsymbol{w} = \begin{bmatrix} \omega_\omega \\ \omega_b \end{bmatrix}$$

3 Identification of the boat parameters

3.1 Transfer function from δ to ψ

Assuming no disturbance, i.e $b = 0$, we want to find the transfer function $H(s) = \frac{\psi}{\delta}(s)$.

From (1) we have

$$
\begin{aligned}
\dot{\psi} &= r \\
\Rightarrow \ddot{\psi} &= \dot{r} = -\frac{1}{T}r + \frac{K}{T}(\delta - b), b = 0 \\
&\Downarrow \mathcal{L} \\
s^2\psi(s) &= -\frac{1}{T}s\psi(s) + \frac{K}{T}\delta(s) \\
\Rightarrow \psi(s)(s^2 + \frac{1}{T}s) &= \frac{K}{T}\delta(s)
\end{aligned}
$$

$$\Rightarrow H(s) = \frac{\psi}{\delta}(s) = \frac{\frac{K}{T}}{s^2 + \frac{1}{T}s} = \frac{K}{s(Ts+1)} \tag{3}$$

3.2 Identifying boat parameters in smooth weather conditions

We want to find the parameters K and T in smooth weather conditions. We turn off all disturbances in the model and apply sine inputs to system with amplitude 1 and frequency $\omega_1 = 0.005$ and $\omega_2 = 0.05$. $|H(jw)| = A$. The simulink setup for this is shown in figure 24 in B.1. We use this to get two expressions to solve for K and T by looking at the amplitudes of the output signal, A_1 and A_2, with their respective frequencies, as shown in (5)

The amplitude of the plots are found by using (4)

$$A = \frac{max - min}{2} \tag{4}$$

where the minimum and maximum values of the wave are found by using the min() and max() functions in MATLAB. The procedure for finding the amplitudes are seen in listing 1 in A.1. We find that $A_1 = 29.36$ and $A_2 = 0.83$

$$|H(j\omega_1)| = |\frac{K}{j0.005(Tj0.005+1)}| = A_1 \tag{5a}$$

$$|H(j\omega_2)| = |\frac{K}{j0.05(Tj0.05+1)}| = A_2 \tag{5b}$$

Solving for K and T we get |T| = 72.5216 and |K| = 0.1561 Graphs of the outputs to the sine waves with respective frequencies ω = 0.005 and ω = 0.05 can be seen in figures 1 and 2.

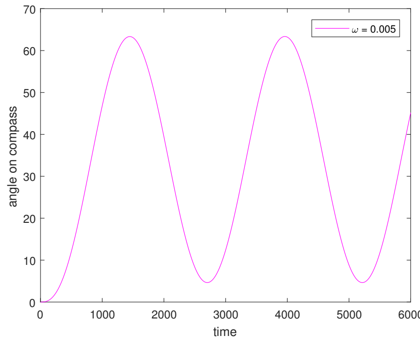

Figure 1: System response with ω_1 = 0.005 in smooth weather conditions

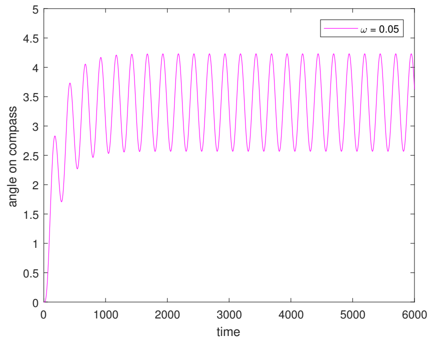

Figure 2: System response with ω_2 = 0.05 in smooth weather conditions

3.3 Identifying boat parameters in rough weather conditions

We use the same procedure to find K and T, but this time with waves and measurement noise on. The MATLAB code for identifying amplitude is shown in listing 2 in A.1. We find that $A_1 = 31.01$ and $A_2 = 2.8449$ By using (5) we find $|T| = 2.02$ and $|K| = 0.156617$.

Graphs of the outputs to the sine waves with different frequency are shown in figures 3 and 4.

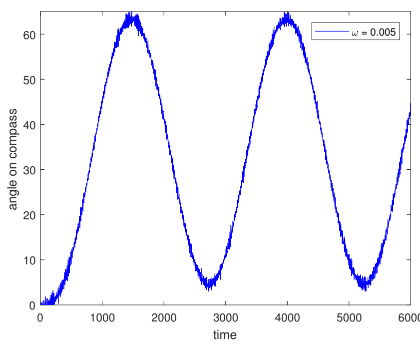

Figure 3: System response with ω_1 = 0.005 in rough weather conditions

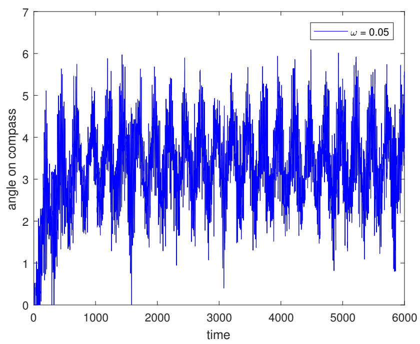

Figure 4: System response with ω_2 = 0.05 in rough weather conditions

As we can see from the results comparing to section 3.3, the approximated K value fits well with the system, but the calculated time constant deviated a lot from the time constant in section 3.3. We can see from the plots that noise affects the signal noticeably in the tops and bottoms of the curve for $\omega = 0.005$. Otherwise the output is not affected too much. For $\omega = 0.05$ we can see the signal being affected noticeably more by noise at all times, because of it's higher frequency. This results in higher 'peaks' and lower 'lows' which makes the calculated amplitude of the signal appear way larger than it actually is. This again affects the time constant severely. We conclude that it's not possible to get good estimates of the boat parameters in rough weather conditions using the two given frequencies.

3.4 Step response

We want to compare the response of our model with the response of the ship. We apply a step input of 1 degree to the rudder at $t = 0$ on both systems.

Our model is implemented by using the transfer function block in simulink with the transfer function in (3). The values for K and T calculated in section 3.2 are used. Simulink diagram of the system is shown in figure 25 in B.1, code for plotting and comparing the responses are shown in listing 3 in A.1.

As seen in figure 5 the response of the mathematical model is nearly identical to the response of the boat at the start of the simulation. After about 1000 seconds the models start to deviate noticeably from each other, and the deviation grows with time. Because our model acts very similar to the system for such a long time interval, we conclude that it is a good mathematical approximation for the system.

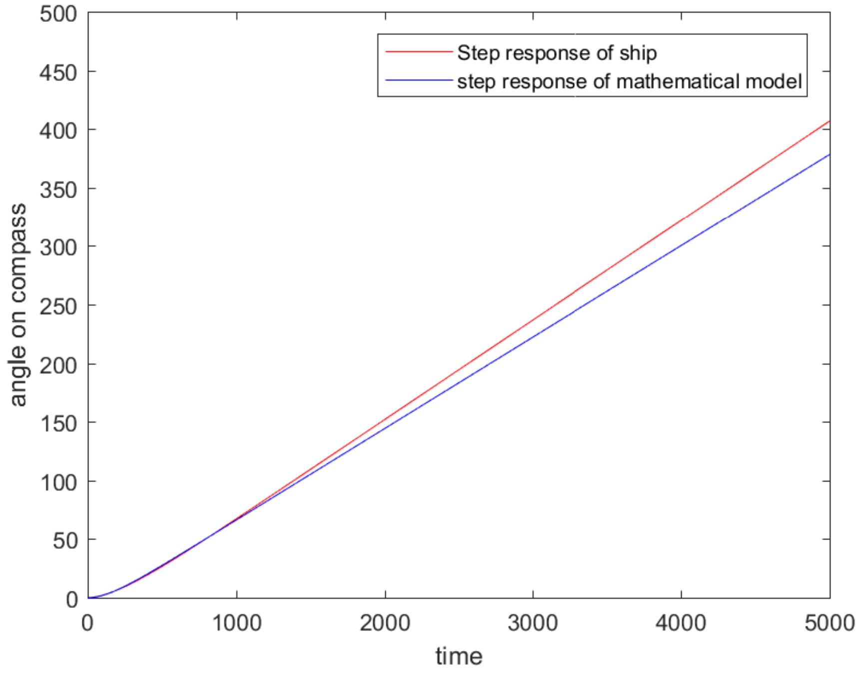

Figure 5: Step response of ship and modelled system

4 Identification of wave spectrum model

4.1 Estimate of PSD function

We want to estimate the Power Spectral Density of ψ_w from an array of samples given in a datafile. This is done in MATLAB using the command pwelch(). This command generates an estimate of the Power spectral density function based on Welch's method. The resulting estimate can be seen in figure 4.1 below. The MATLAB code used to generate the estimate can be found under section A.2 in listing 6

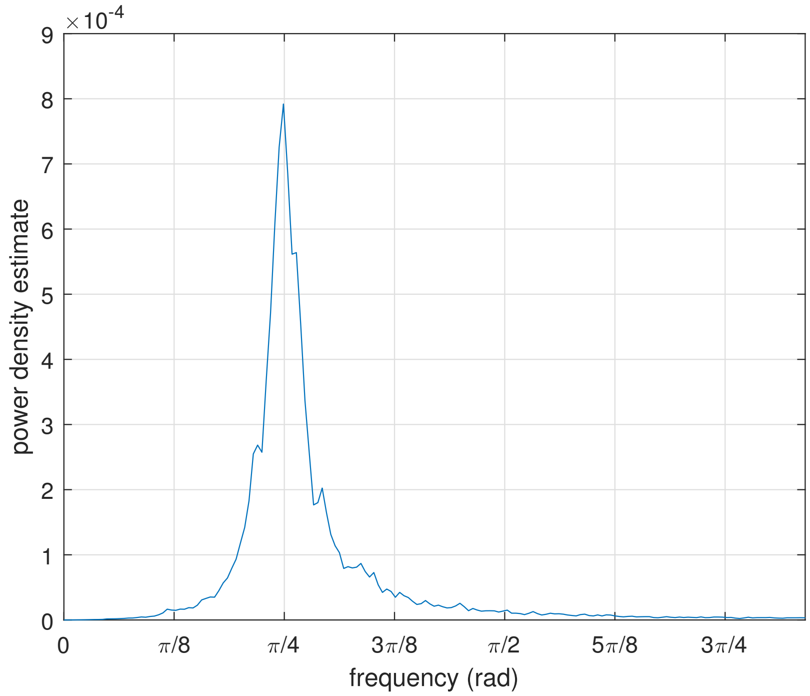

Figure 6: Power density spectrum estimate from samples using Welch's method

4.2 Transfer function from ω_ω to ψ_ω

In this task we want to find a transfer function from the wave disturbance ω_ω to the high frequency heading component ψ_ω We have from equation (1a) that $\dot{\xi}_\omega = \psi_\omega$. Inserting for ξ in equation (1b) we get

$$\dot{\psi_\omega} = -\omega_0^2 \int \psi_\omega - 2\lambda\omega_0\psi_\omega + K_\omega\omega_\omega$$
$$\Rightarrow s\psi_\omega = \frac{-\omega_0^2\psi_\omega}{s} - \lambda\omega_0\psi_\omega + K_\omega\omega_\omega$$

$$\frac{\psi_\omega}{\omega_\omega}(s) = \frac{K_\omega s}{s^2 + 2\lambda\omega_0 s + \omega_o^2} = H(s) \tag{6}$$

As we have found the transfer function from ψ_ω to ω_ω we can now find an analytical expression for the signal's power spectral density. We know that ω_ω is a zero mean white noise process with unity variance. This means that the variance of $\omega_m = 1$. The power spectral density of this signal is given by the formula

$$P_{\psi_w}(\omega) = P_{\omega_\omega}(j\omega) \cdot |H(j\omega)|^2 \tag{7}$$

We know ω_ω is a white noise process with zero mean and unity variance. This means that its know autocorrelation function $\gamma_{\omega\omega}(\tau) = 1 \cdot \delta(\tau) = \delta(\tau)$. The PSD equals the fourier transform of the autocorrelation function. We then get

$$P_{\psi_\omega}(\omega) = \mathcal{F}(\gamma_{\omega\omega}(\tau)) \cdot H(j\omega) \cdot H(-j\omega)$$

$$= \frac{K_\omega j\omega}{(j\omega)^2 + 2\lambda\omega_0 j\omega + \omega_o^2} \cdot \frac{K_\omega(-j\omega)}{(-j\omega)^2 - 2\lambda\omega_0 j\omega + \omega_o^2}$$

$$\Rightarrow P_{\psi_\omega}(\omega) = \frac{(\omega K_\omega)^2}{\omega^4 + \omega_0^4 + 2\omega\omega_0(2\lambda^2 - 1)} \tag{8}$$

4.3 Finding ω_0 and σ^2

A transfer function on the form as shown in equation 6 has a resonance top at $\omega = \omega_0$. This means that the Power spectral density estimate also has a maximum at $\omega = \omega_0$. If we can identify the maximum of the estimate we know that the frequency at which that maximum occurs is equal to ω_0. Code for finding x and y coordinate of the maximum can be seen in listing 5 in section A.2. This approach resulted in the values $\omega_0 = 0.7823$ and the peak variance $\sigma^2 = 0.000792$

4.4 Identifying damping factor, λ

Through trial and error with fitting the graph of $P_{\psi_m}(\omega)$ to ψ_{m_ω}, the best fit was achieved for $\lambda = 0.09$. Several values for λ got graphed and compared to estimate, but only a few of them were included in figure 7 for visual reasons. The MATLAB script used for generating the plot can be seen in listing 6 in section A.2

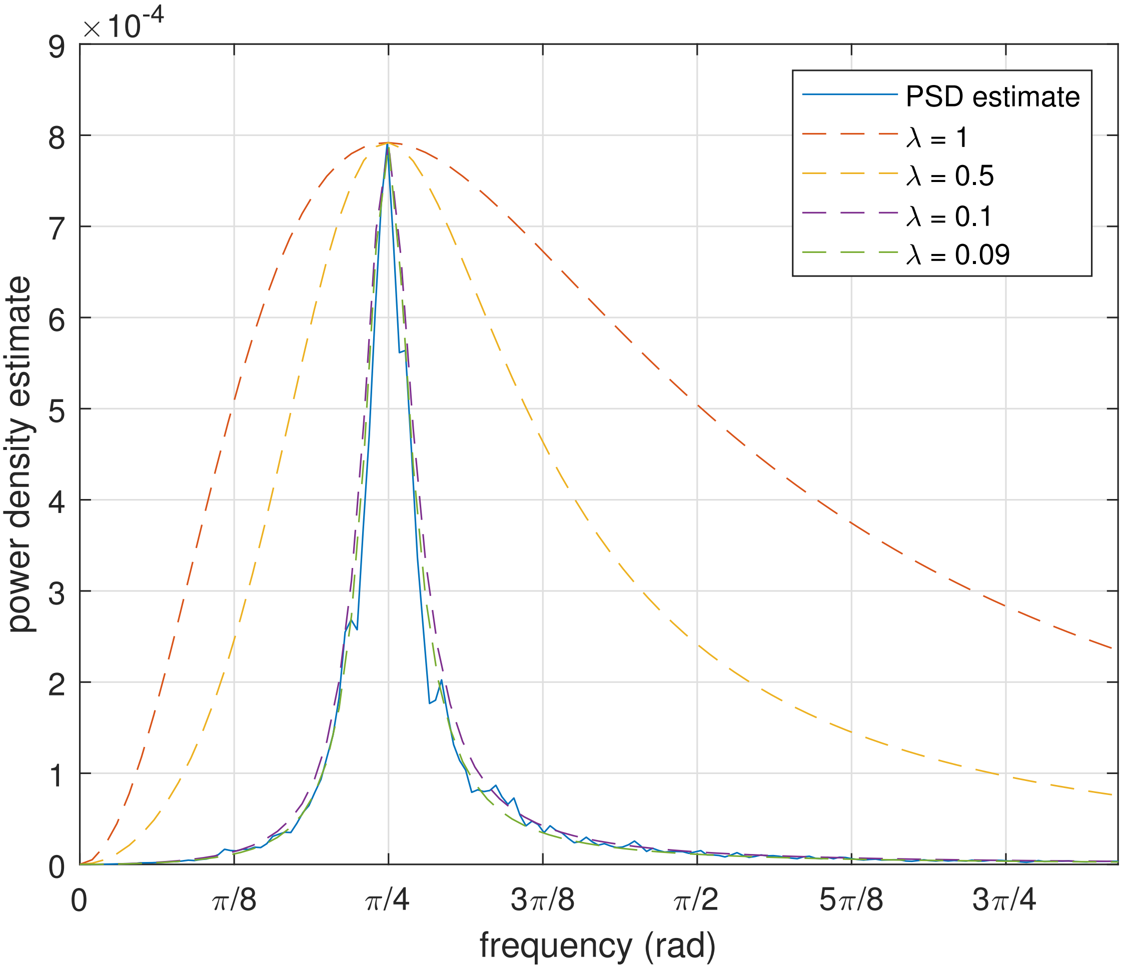

Figure 7: Power Density Spectrum for different values of λ

All components of the wave model have now been identified, and we have a complete model of the wave response.

5 Control system design

5.1 Design of PD controller

We want to design a PD controller for the system. The transfer function of the PD controller is given by (10)

$$H_{ship}(s) = \frac{K}{s(1 + Ts)} \tag{9}$$

$$H_{pd}(s) = K_{pd}\frac{1 + T_d s}{1 + T_f s} \tag{10}$$

By choosing derivative time constant, $T_d = T$ for the controller we cancel out the time constant of the ship transfer function, (9), and get the following expression for the open loop system

$$h_0 = H_{pd} \cdot H_{ship} = \frac{KK_{pd}}{(1 + T_f s)s} \tag{11}$$

We want the phase margin, PM, and the cross frequency, ω_c, to be 50° and 0.10 (rad/s) respectively. PM and ω_c are defined as follows

$$\begin{aligned}
PM &= \angle h_o(j\omega_c) - (-180^\circ) \\
\Rightarrow PM - 180^\circ &= \angle \frac{KK_{pd}}{(1 + T_f j\omega_c)j\omega_c} \\
&= \angle(KK_{pd}) - \angle(j\omega_c - T_f\omega_c^2) \\
&= 0 - \arctan(\frac{\omega_c}{-T_f\omega_c^2}) \\
\Rightarrow \tan(PM - 180^\circ) &= \frac{\omega_c}{-T_f\omega_c^2}
\end{aligned}$$

$$\Rightarrow T_f = \frac{-1}{\omega_c \tan(PM - 180^\circ)} \tag{12}$$

Inserting $\omega_c = 0.10$ (rad/s) and PM = 50° into equation (12) we get $T_f \approx 8.39$ (s)
At the cutoff frequency, ω_c, the magnitude of the transfer function, $h_0(s)$ is equal to 1. We use this to find K_{pd}

$$|h_0(j\omega_c)| = KK_{pd}\frac{1}{\sqrt{\omega_c^2 + (T_f\omega_c)^2}} = 1$$

$$\Rightarrow K_{pd} = \frac{\sqrt{\omega_c^2 + (T_f\omega_c^2)^2}}{K} \tag{13}$$

In section 3.2 we found the constant of the ship, K, to be 0.1561 without disturbances. By inserting this and the values for ω_c and T_f into (13) we get $K_{pd} \approx 0.8362$.

In the table below we have the summarized the values for the system parameters thus far

Table 1: Closed loop system parameters.

Symbol	Parameter	Value
K	Boat constant	0.1561
T	Boat time constant	72.5216
K_{pd}	Controller constant	0.8362
T_d	Controller derivative time constant	72.5216
T_f	derivative effect limit time constant	8.39

5.2 Simulation without disturbance

The simulink implementation of the controller is shown in figure 26 in B.3 . A saturation block is added to limit the input to values between -35° and 35°, as this is the psycial limit of the boat rudder. The system response in smooth weather conditions and containing measurement noise with a 30° reference input for the compass is shown in figure 8. The code for plotting is shown in listing 7 in A.3.

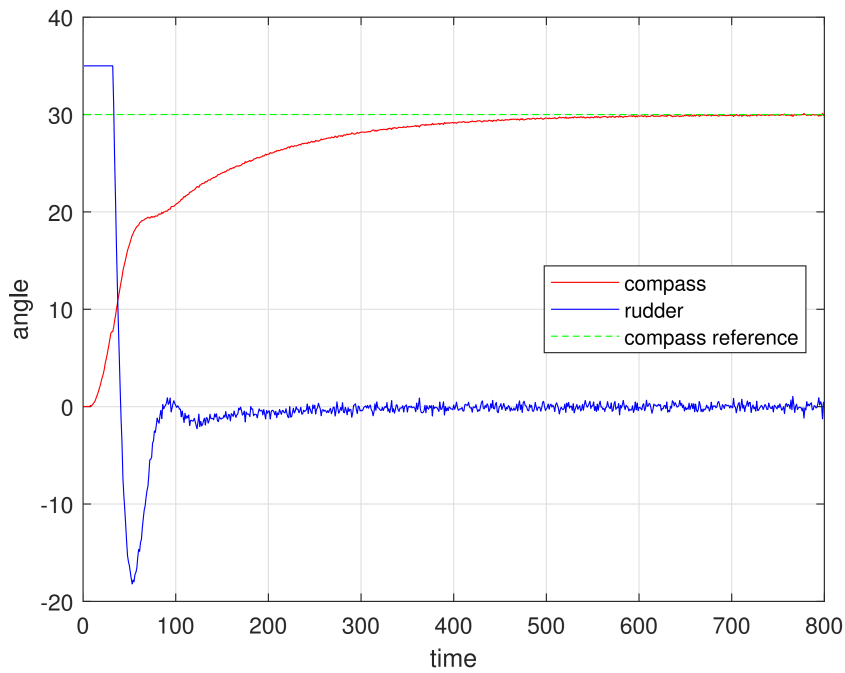

Figure 8: Step response of ship and modelled system with only measurement noise as disturbance

The autopilot works well, it reaches the input reference in about 600 seconds. We use a PD-controller instead of a P-controller here because of the massive momentum the boat has. The PD-output is a combination of how far the system is from the reference and how fast it is approaching the reference. This means that if the system is rapidly approaching the reference it will slow down. This is a must to avoid overshoot as the boat model has a massive moment of inertia. It is however not good for handling the wave disturbance as taking the derivative of noise is the same as amplifying it. We can see the system slowing down in practice in figure 8. The rudder angle changes directions long before the compass has reached steady state.

5.3 Simulation with current disturbance

We will now simulate the system with a current disturbance using the PD-controller designed previously. The system response can be seen below in figure 9

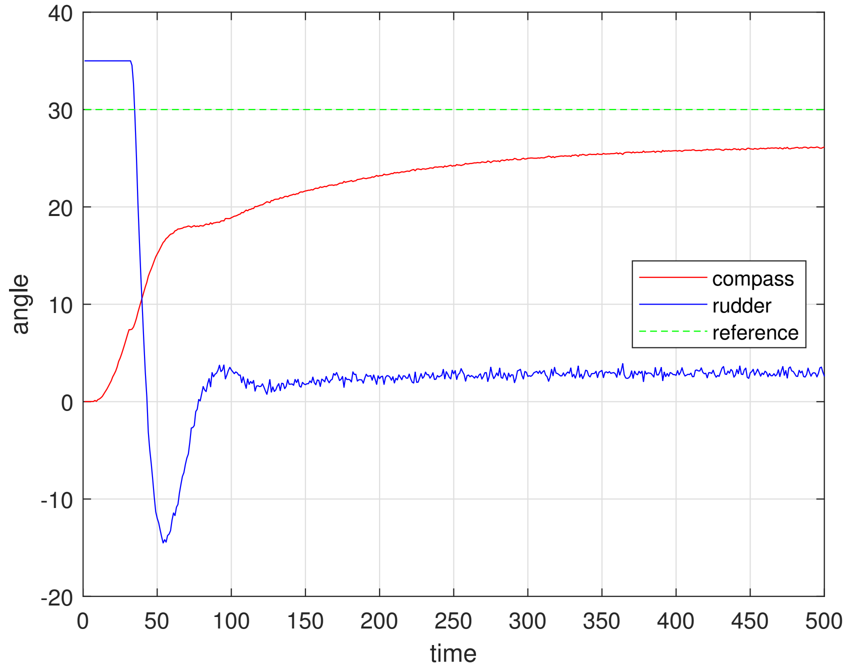

Figure 9: Response of the system with a current disturbance

We can see from the response in figure 9 that the autopilot does not work optimally. The system output has a large steady state error of 4° which will cause the ship to deviate more and more from the set waypoint as time goes by. The rudder angle is constantly around 2° to try and counteract the current induced steady-state error, but this is not sufficient to completely counteract the waves. This autopilot will not be be sufficient in a realistic setting.

5.4 Simulation with wave disturbance

The system is simulated with wave disturbance, and no current disturbance using the PD-controller. As we know the waves only affect the heading so the rudder is expected to move a lot to try and compensate for this. The resulting response can be seen below in figure 10

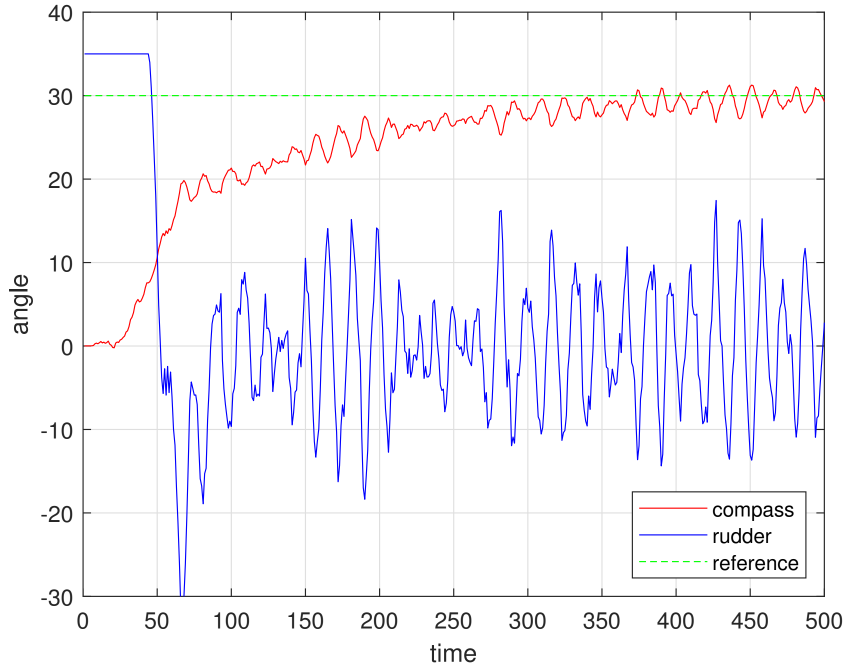

Figure 10: Response of the system with wave disturbance

From the response we see that the compass course is very oscillatory, but with a small amplitude. This translates to a lot of oscillations in the rudder as well. The rudder changes directions very fast. This is because passing noise through a PD-controller is the same as amplifying it. In its "steady state" the rudder can be seen varying between approximately -15° and 15°. This is not satisfactory as it will tear on the mechanical parts of the rudder way faster than normal. It will also cost a lot more fuel for the ship as it has to travel quite a bit longer than if it didn't change course so frequently. This will again result in longer travel times, and a lot more fuel expended. We therefore conclude that the autopilot is not reliable in waves nor with a current.

6 Observability

We want to optimize the autopilot to work in rough weather conditions. In order to do this the plan is to add a Kalman filter, but to do this we first have to check if the system is observable.

6.1 State space model

With $\mathbf{x} = \begin{bmatrix}\xi & \psi_w & \psi & r & b\end{bmatrix}^T$, u $= \delta$ and $\mathbf{w} = \begin{bmatrix}w_w & w_b\end{bmatrix}^T$ We obtain from the set of equations in (1) the state space matrices

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -\omega_0^2 & -2\lambda\omega_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{-1}{T} & \frac{-K}{T} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \\ 0 \end{bmatrix}$$

$$\mathbf{E} = \begin{bmatrix} 0 & 0 \\ K_\omega & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{14}$$

6.2 Observablity without disturbance

The observability matrix is given by the formula

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \vdots \\ \boldsymbol{CA^{n-1}} \end{bmatrix} \tag{15}$$

Without any disturbances we have that $\psi_w = 0$. Since $\dot{\xi} = \psi_w$ this means that $\xi =$ K, where K is a constant. For this task we assume K $= 0 \Rightarrow$ b is also equal to 0. Our state vector $\mathbf{x}$ then becomes $\begin{bmatrix}\psi & r\end{bmatrix}^T$. $\mathbf{w}$ is also equal to 0 now and $\mathbf{B}$ remains the same. We get the following state space model

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{T} \end{bmatrix} \begin{bmatrix} \psi \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{K}{T} \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}$$

The observability matrix for this system is given by the definition in equation

15

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{16}$$

The observability matrix has rank $= 2 =$ amount of states. We can therefore conclude that the system is indeed observable for this configuration.

6.3 Observablity with current disturbance

In this task we have taken a naive approach to the current disturbance in that it only affects the rudder angle. Therefore the rudder bias b $\neq 0$ when a current disturbance is present. We then get the state vector $\mathbf{x} = \begin{bmatrix} \psi & r & b \end{bmatrix}^T$.
, and the following state space representation

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \psi \\ r \\ b \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{K}{T} \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w_b$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \psi \\ r \\ b \end{bmatrix}$$

The observability matrix can be calculated as

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \boldsymbol{CA^2} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{1}{T} & -\frac{K}{T} \end{bmatrix} \tag{17}$$

The observability matrix has 3 linearly independent column vectors which means it has rank $= 3 =$ amount of states. We can therefore conclude that the system is observable with a current disturbance introduced.

6.4 Observablity with wave disturbance

If we introduce a wave disturbance our model gets the state vector $\mathbf{x} = \begin{bmatrix} \xi_w & \psi_w & \psi & r \end{bmatrix}^T$, and the following state space representation

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\omega_0^2 & -2\lambda\omega_0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{T} \end{bmatrix} \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \end{bmatrix} u + \begin{bmatrix} 0 \\ K_w \\ 0 \\ 0 \end{bmatrix} w_w$$

$$y = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \end{bmatrix}$$

The observability matrix can be calculated as

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \boldsymbol{CA^2} \\ \boldsymbol{CA^3} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -\omega_0^2 & -2\lambda\omega_0 & 0 & 1 \\ 2\lambda\omega_0^3 & 4\lambda^2\omega_0^2 - \omega_0^2 & 0 & -\frac{1}{T} \\ \omega_0^2(-4\lambda^2\omega_0^2 + \omega_0^2) & 2\lambda\omega_0^3 + 2\lambda\omega_0(-4\lambda^2\omega_0^2 + \omega_0^2) & 0 & \frac{1}{T^2} \end{bmatrix}$$

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -0.6120 & -0.1408 & 0 & 1 \\ 0.0862 & -0.5922 & 0 & -0.0138 \\ 0.3624 & 0.1696 & 0 & 0.0002 \end{bmatrix}$$

The rank of this matrix is equal to 4 = amount of states. The system is therefore observable with a wave disturbance introduced.

6.5 Observablity with current and wave disturbance

If we include both a wave and current disturbance in our system we can use the state space model from (14) to compute the observably matrix, $\boldsymbol{\mathcal{O}}$.

$$\boldsymbol{\mathcal{O}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \boldsymbol{CA^2} \\ \boldsymbol{CA^3} \\ \boldsymbol{CA^4} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ -0.612 & -0.1408 & 0 & 1 & 0 \\ 0.0862 & -0.5922 & 0 & 0.0138 & 0.0022 \\ 0.3624 & 0.1696 & 0 & 0.0002 & 0 \\ -0.1038 & 0.3385 & 0 & 0 & 0 \end{bmatrix}$$

From inspection and verification in MATLAB, the observability matrix has rank = 5. The system is therefore observable with a wave and current disturbance which means the system is observable for all possible configurations. Code for calculating observability matrix can be seen in listing 10 in appendix A.4.

6.6 Discussion

We have found out that the system is indeed observable with different forms of disturbances introduced. This means we can proceed with using an observer on the system, which will be done in the next task.

7 Discrete Kalman filter

In this part of the assignment we will use a Kalman filter to estimate all states. Of these the rudder bias **b**, the heading $\boldsymbol{\psi}$ and the wave induced motion of the heading $\boldsymbol{\psi_w}$ will be used in the estimation. The Kalman filter is an excellent estimation algorithm when statistical random signals are present in the system. It is an iterative algorithm whose outputs converges towards statistically optimal estimates of the system states after very few iterations compared to other algorithms.

7.1 Discretization

In order to implement a discrete Kalman filter in the system we first have to discretize the model given by (14). Exact discretization is used with a sampling frequency of 10 Hz.

The continuous system is on the following form

$$\begin{aligned} \dot{\boldsymbol{x}}\boldsymbol{(t)} &= \boldsymbol{Ax(t)} + \boldsymbol{Bu(t)} + \boldsymbol{Ew(t)} \\ \mathbf{y(t)} &= \boldsymbol{Cx(t)} + v(t) \end{aligned}$$

we want to transform it into a discrete system as on the form below

$$\begin{aligned} \boldsymbol{x}[k+1] &= \boldsymbol{A}_d\boldsymbol{x}[k] + \boldsymbol{B}_d\boldsymbol{u}[k] + \boldsymbol{E}_d\boldsymbol{w}[k] \\ \boldsymbol{y[k]} &= \boldsymbol{C}_d\boldsymbol{x}[k] + v[k] \end{aligned}$$

By using the MATLAB command **c2d(sys,Ts)** which converts a continuous state space model to a discrete one we get the following discretized matrices. The code for this is shown in shown in listing 11 in A.5.

$$\boldsymbol{A}_d = \begin{bmatrix} 0.997 & 0.0992 & 0 & 0 & 0 \\ -0.0607 & 0.9830 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.1 & 0 \\ 0 & 0 & 0 & 0.9986 & -0.0002 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{B}_d = \begin{bmatrix} 0 \\ 0 \\ 0.0108 \\ 0.2151 \\ 0 \end{bmatrix}$$

$$\mathbf{E}_d = \begin{bmatrix} 0 & 0 \\ 0.0004 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{C}_d = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{18}$$

7.2 Estimated variance of measurement noise

We want to find an estimate for the variance of the measurement noise. The simulink boat model is run with rudder reference = 0, to measure the noise. The MATLAB function var() is then used to find the variance of the measurement in deg^2. This was then converted to rad^2. The code for this is shown in listing 12 in A.5.

$$\sigma_m^2 \quad = \quad 6.1614 * 10^{-7}$$

7.3 Implementation

From the assignment we are given the following matrices where Q is the process noise covariance matrix, and R is the measurement noise variance.

$$\boldsymbol{w} = \begin{bmatrix} w_w & w_b \end{bmatrix}^T \quad E[\boldsymbol{w}\boldsymbol{w^T}] \ = \ \boldsymbol{Q} \ = \ \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix} \quad E[v^2] == \sigma_m^2 * F_s$$

The a priori error covariance estimate $\mathbf{P}_0^-$ and initial a priori state $\mathbf{x}_0^-$

$$\mathbf{P}_0^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.013 & 0 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-3} \end{bmatrix} \quad \mathbf{x}_0^- = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{19}$$

The Kalman filter algorithm uses the following equations

$$\begin{aligned} \mathbf{L}_k &= \mathbf{P}_k^- \mathbf{C}^T (\mathbf{C}\mathbf{P}_k^- \mathbf{C}^T + \bar{\mathbf{R}}_\mathbf{v})^{-1} \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \mathbf{L}_k (\mathbf{y}_k - \mathbf{C}\hat{\mathbf{x}}_k^-) \\ \mathbf{P}_k &= (\mathbb{I} - \mathbf{L}_k \mathbf{C}) \mathbf{P}_k^- (\mathbb{I} - \mathbf{L}_k \mathbf{C})^T + \mathbf{L}_k \bar{\mathbf{R}}_\mathbf{v} \mathbf{L}_k^T \\ \mathbf{P}_{k+1}^- &= \mathbf{A}\mathbf{P}_k^- \mathbf{A}^T + \mathbf{E}\mathbf{Q}_\mathbf{w}\mathbf{E}^T \\ \hat{\mathbf{x}}_{k+1}^- &= \mathbf{A}\hat{\mathbf{x}}_k + \mathbf{B}\mathbf{u}_k \end{aligned}$$

$\boldsymbol{L_k}$ is the Kalman observer gain
$\boldsymbol{\hat{x}_k}$ is the a posteriori state estimates
$\boldsymbol{P_k}$ is the a posteriori error covariance matrix
$\mathbf{P}_{k+1}^-$ is predicted error covariance matrix, and
$\hat{\mathbf{x}}_{k+1}^-$ is the a priori state estimates

The code for the MATLAB function simulink block can be seen in listing 13 in in A.5. The simulink model including the Kalman filter can be seen in the appendix in figure 27

7.4 Feed-forward

In this part of the task we make a feed forward from the Kalman estimated bias such that we cancel the bias on the output signal. The resulting simulink diagram can be seen in the appendix in figure 29

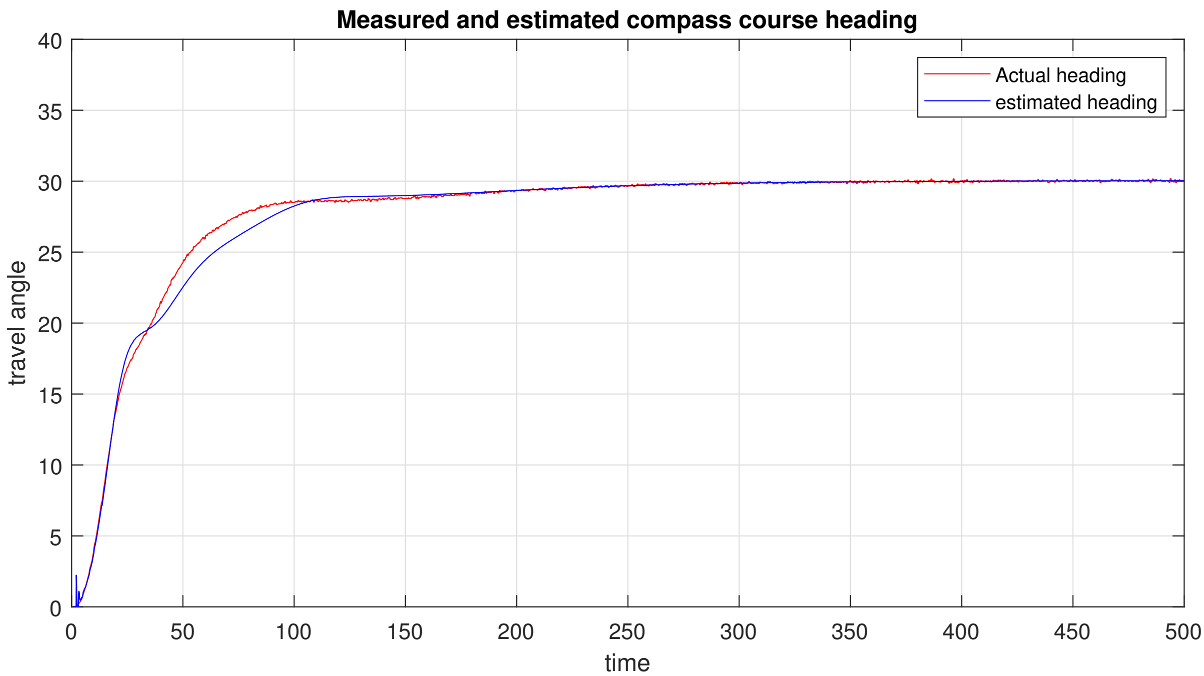

Figure 11: Response of the system with a current disturbance

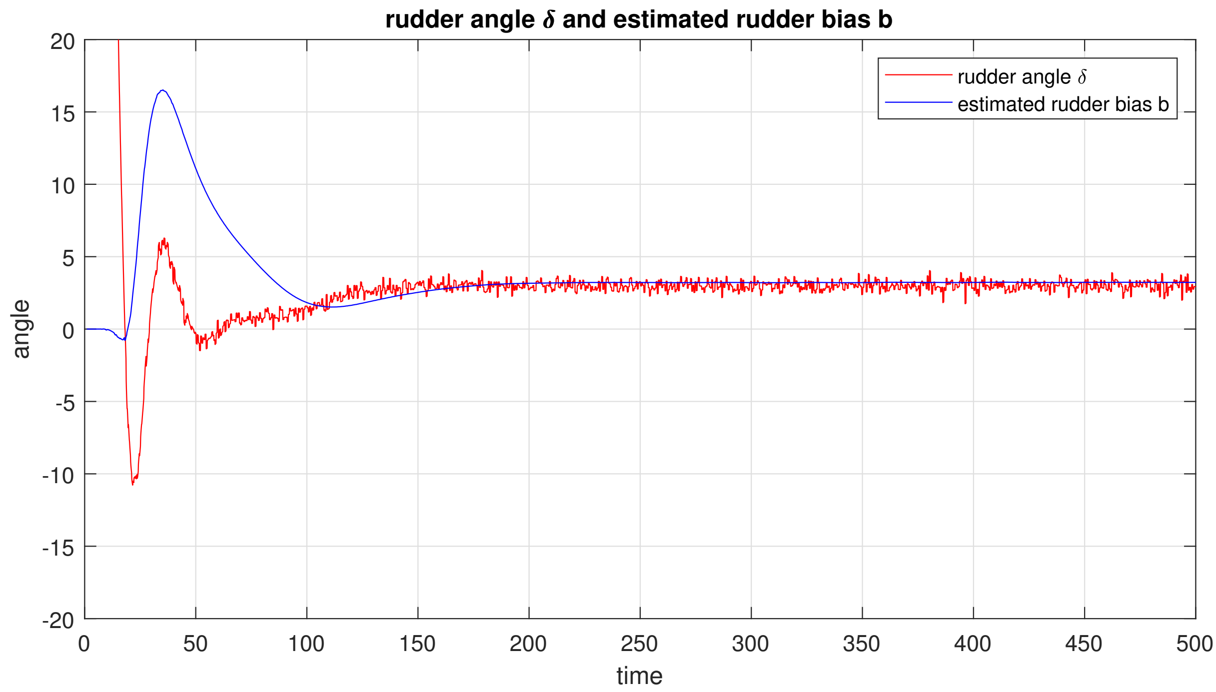

Figure 12: Rudder input and estimated bias

If we compare this to the PD-controller response in figure 9 we see a massive improvement. The steady-state error is eliminated and the compass heading reaches the reference in about half the time it did with only the PD-controller.

7.5 Wave-filter

We will now use the wave-filtered heading from the kalman filter instead of the compass heading as feedback in the outer loop of the control system.

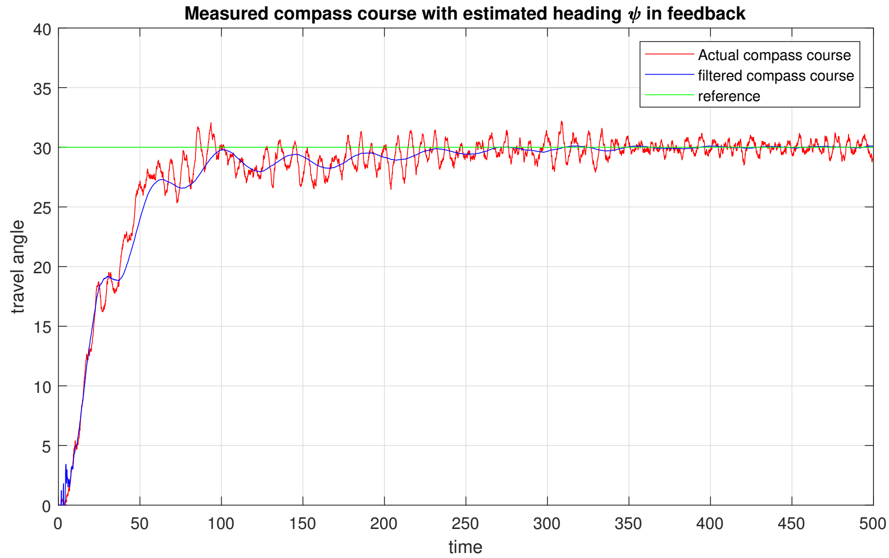

Figure 13: Response of system with heading estimate in feedback

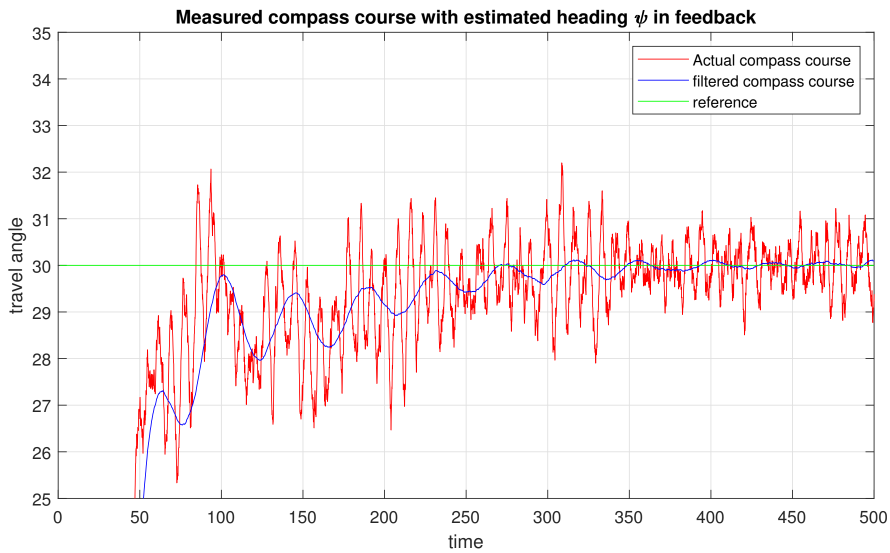

Figure 14: Zoomed in...

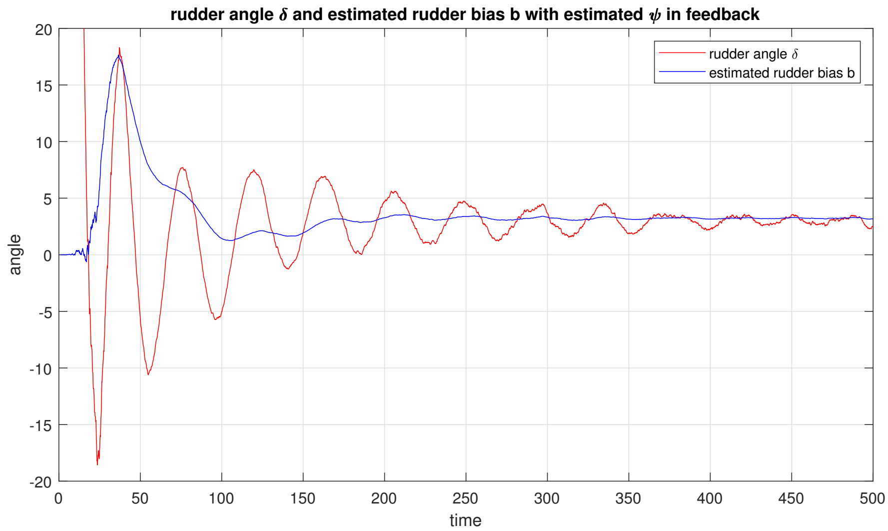

Figure 15: Rudder movement and estimated bias

Compared to the result in section **5.4** which can be seen in figure 10 we see a significant improvement. The compass course reaches the reference in about 250 seconds as can be seen in figure 13. The Kalman filter has shaved off nearly 200 seconds of the transient response time. The wave-filtered Kalman estimated heading has no signs of noise in it either and, works great as feedback. To measure the wave influence we set the reference compass course $= 0$ and measure the compass course of the ship and the estimated wave influence ψ_w. The actual and the estimated wave influence can be seen in figure 16 and 17

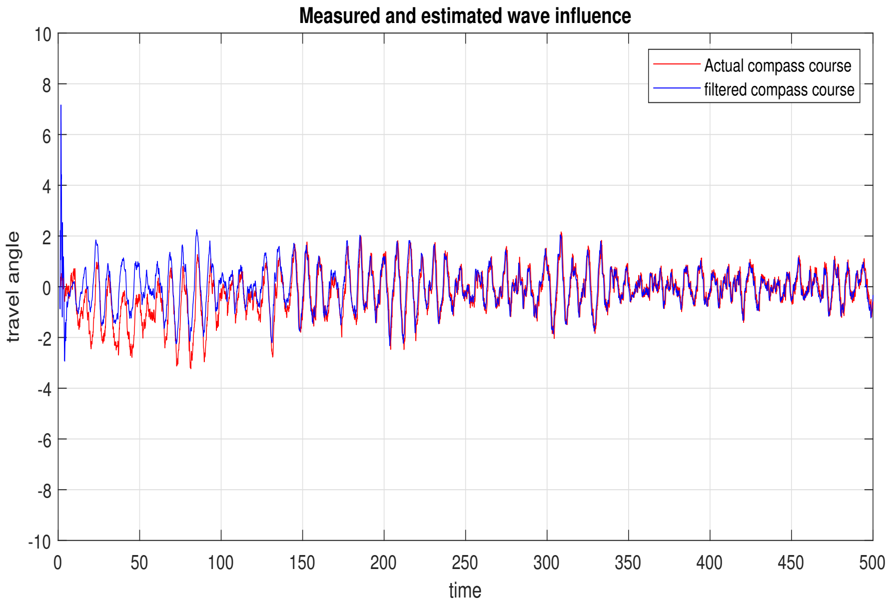

Figure 16: Comparison of actual and estimated wave influence

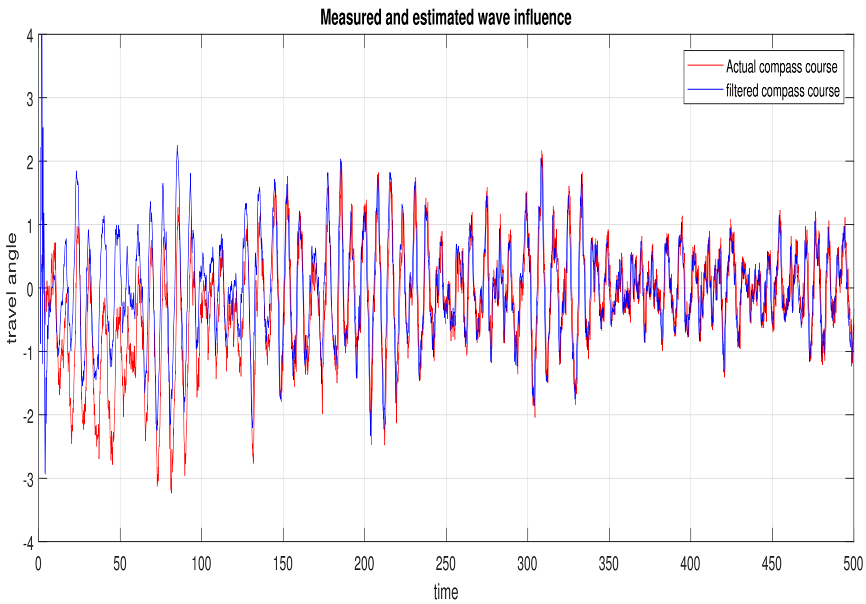

Figure 17: Zoomed in...

7.6 Effects of the Q-matrix

We know from 19 that the process noise covariance matrix Q is defined as

$$Q = E[\boldsymbol{w}\boldsymbol{w}^{\boldsymbol{T}}] = \begin{bmatrix} E[w_w^2] & E[w_w w_b] \\ E[w_w w_b] & E[w_b^2] \end{bmatrix} = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix}$$

From this we can see that the wave disturbance and current disturbance have 0 covariance and are therefore uncorrelated. This means that a change in the upper diagonal element will not affect the current, and a change in the lower diagonal element will not affect the waves as they do not covariate. Some plots are included below.

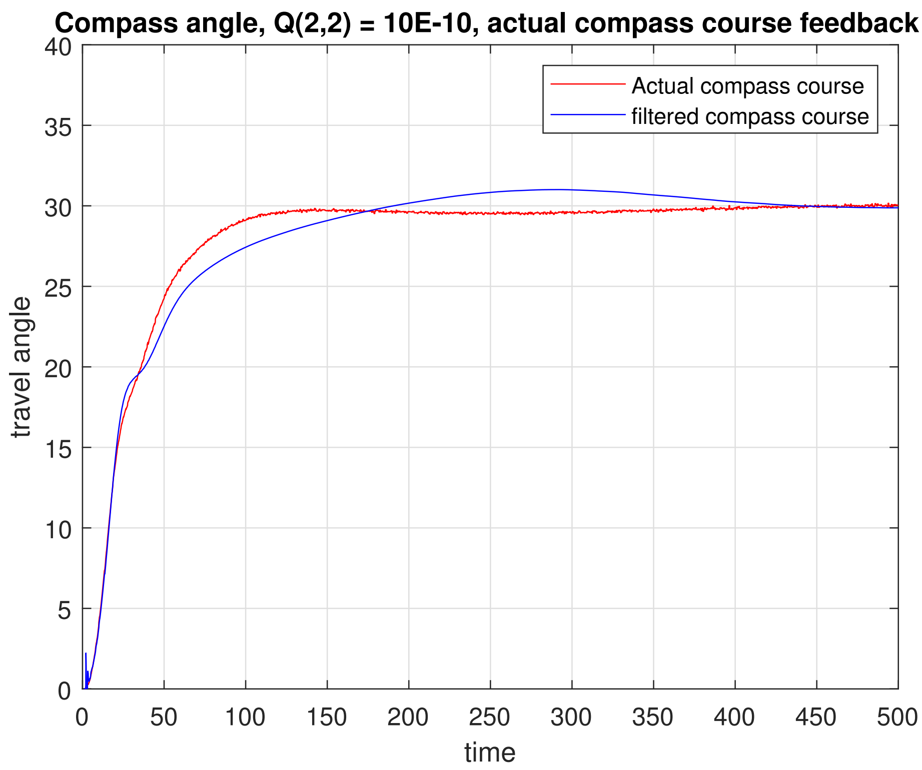

Figure 18: Response of system with estimated heading in feedback, no waves

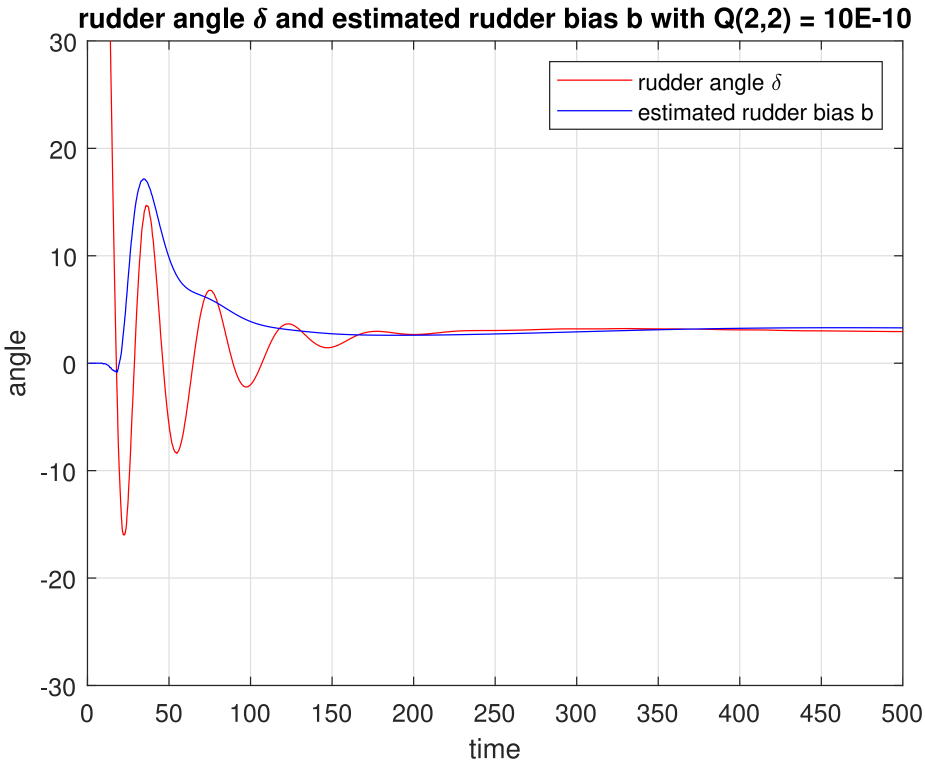

Figure 19: Response of system with heading estimate in feedback

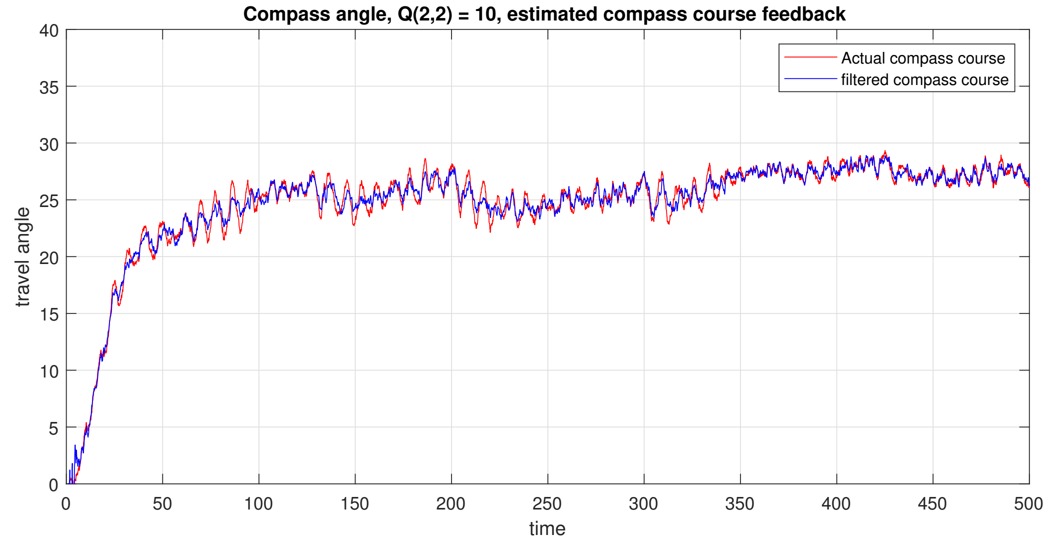

Figure 20: Compass course and estimate for high current covariance with wave disturbance

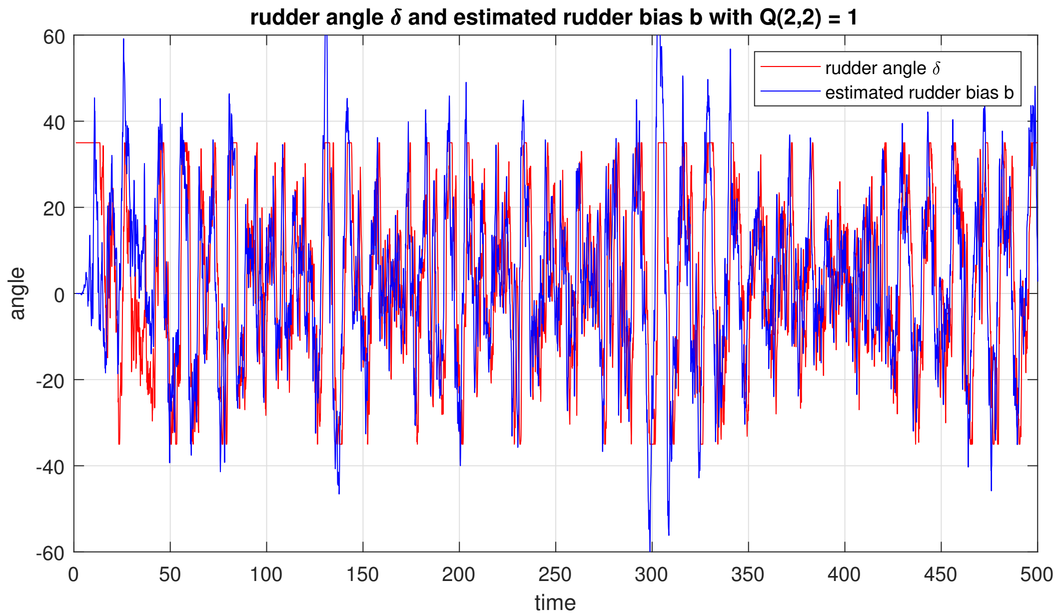

Figure 21: Rudder and bias for high current covariance with wave disturbance

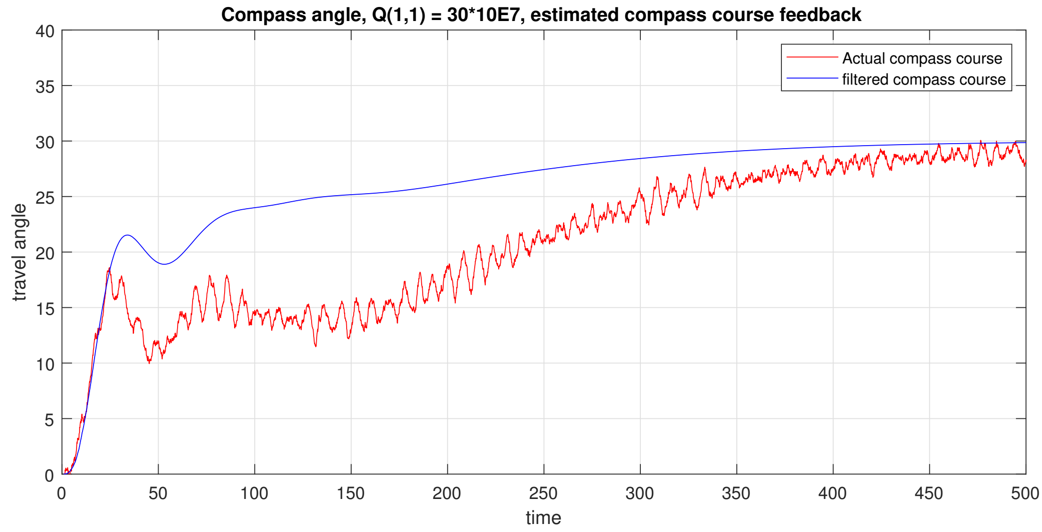

Figure 22: Actual and estimated heading for high wave covariance and all disturbances

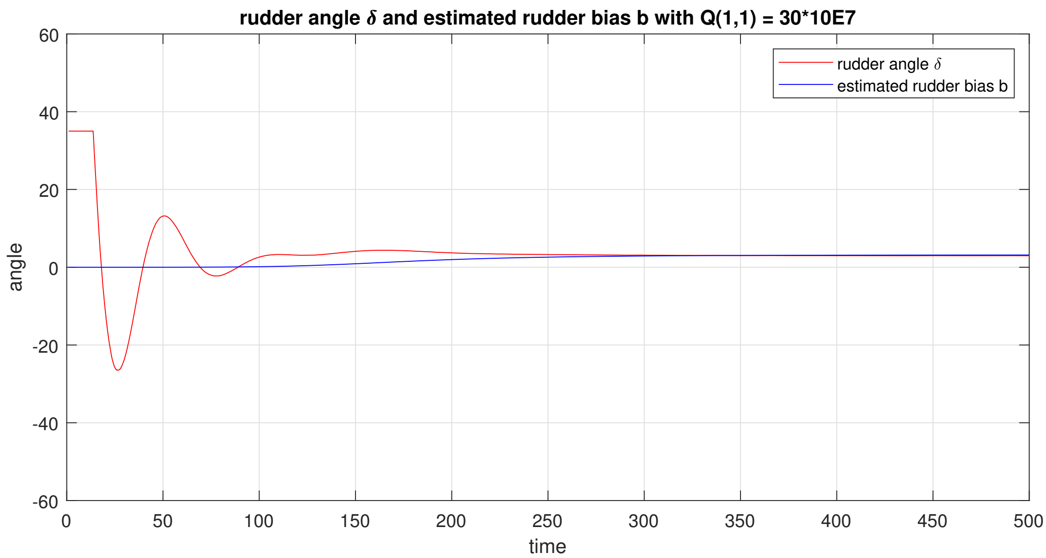

Figure 23: Rudder and bias for high wave covariance with wave disturbance

We can see several things from the plots. A high current covariance will increase the transient response time of the compass angle, but the overall compass course is good. The rudder angle and the bias on the other hand is hard to make anything off. They both look to be contaminated with noise and very oscillatory. A low current covariance gives super smooth rudder angle and estimated bias, but causes the compass course to overshoot on the reference and take a long time to reach steady-state. A high wave covariance makes the bias, b, react slow to the rudder angle. The actual compass course has huge erros for most of its transient period, but it reaches the steady state after a just over 500 seconds.

As stated earlier the process noise covariance matrix has two diagonal elements which represent the covariance of the wave signal and the covariance of the current signal. They are uncorrelated signals. As these values states how much a signal covariates with itself, increasing it will naturally lead to more uncertainty in the system. As the current variance gets bigger, the Kalman filter will have a harder time estimating an optimal bias for use in the feedfoward as the rudder will try to compensate for a heavily varying incoming current. This leads to rapid changes in the rudder angle at most times, and if the estimator is not fast enough the bias feed-forward will be practically useless. However it seems to be managing well with a high current covariance, but the performance is definitely worse than with the original configuration. A very high wave covariance will make the heading very hard to control as there wont be a clear pattern in the incoming wave disturbance. This means that the heading will use a long time to set and the rudder angle/ estimated bias will not act as agressively as usual.

8 Conclusion

In conclusion the control system for the ship works well. We have created a mathematical model of the system and performed a spectral analysis of the ships wave response. This was used to create and tune a PD controller to steer and autopilot the ship. As this controller was not sufficient in counteracting the noises and disturbances in the system a Kalman filter was also added to provide a feed-forward from the rudder angle bias, eliminating the steady state error of the autopilot, and a wave-filtered heading estimate to be used as feedback.

A MATLAB Code

A.1 Identification of the boat parameters

Listing 1: Code to plot the responses and find amplitudes

```
w_1 = importdata('51b1.mat'); %Import data from simulink
w_2 = importdata('51b2.mat'); %Import data from simulink

n=1:length(w_1(2,:));
m=1:length(w_2(2,:));

max_1 = max(w_1(2,3000:6000));
min_1 = min(w_1(2,3000:6000));
A_1 = (max_1-min_1)/2; %output amplitude for w_1

max_2 = max(w_2(2,3000:6000));
min_2 = min(w_2(2,3000:6000));
A_2 = (max_2-min_2)/2; %output amplitude for w_2


plot(m,w_1(2,n), 'color', 'm'); %plot of response with w_1
xlabel('time'); ylabel('angle on compass');
legend('\omega = 0.005', '\omega = 0.05');
axis([0 6000 0 70]);

figure();

plot(m,w_2(2,m), 'color', 'm'); %plot of response with w_2
xlabel('time'); ylabel('angle on compass');
legend('\omega = 0.005', '\omega = 0.05');
axis([0 6000 0 5]);
```

Listing 2: Code to plot the responses and find amplitude

```
w_1 = importdata('51c1.mat'); %Import data from simulink
w_2 = importdata('51c2.mat'); %Import data from simulink

n=1:length(w_1(2,:)); %Time steps for the x-axis
m=1:length(w_2(2,:));

max_1 = max(w_1(2,3000:6000));
min_1 = min(w_1(2,3000:6000));
```

```
A_1 = (max_1-min_1)/2; %output amplitude for w_1

max_2 = max(w_2(2,3000:6000));
min_2 = min(w_2(2,3000:6000));
A_2 = (max_2-min_2)/2; %output amplitude for w_2


plot(m,w_1(2,n), 'color', 'b'); %plot of response with w_1
xlabel('time'); ylabel('angle on compass');
legend('\omega = 0.005', '\omega = 0.05');
axis([0 6000 0 65]);

figure();

plot(m,w_2(2,m), 'color', 'b'); %plot of response with w_2
xlabel('time'); ylabel('angle on compass');
legend('\omega = 0.005', '\omega = 0.05');
axis([0 6000 0 7]);
```

Listing 3: Code to plot the step response of the ship and the modelled system

```
model = importdata('51dm.mat'); %Import data from simulink
system = importdata('51ds.mat'); %Import data from simulink

n=1:length(model(2,:));

K = 0.1561;
T = 72.5216;

plot(n,system(2,n), 'color', 'r'); %Plot boat system step response
hold on;
plot(n,model(2,n), 'color', 'b'); %Plot model step response

xlabel('time'); ylabel('angle on compass');
legend('Step response of ship', 'step response of mathematical model');
axis([0 5000 0 500]);
```

A.2 Identification of wave spectrum model

Listing 4: Code used for making power spectral density estimate

```
samples = importdata('yourlocation'); %% insert address of wave.mat fil

F_s = 10;
```

```
window = 4096;

[pxx,f] = pwelch(samples(2,:).*(pi/180), window, [], [], F_s);

%% converting from degrees to radians %%
omega = 2*pi.*f;
pxx = pxx./(2*pi);

%% plotting %%
plot(omega,pxx);
grid on;
xticks([ 0 pi/8 pi/4 3*pi/8 pi/2 5*pi/8 3*pi/4 7*pi/8 pi])
xticklabels({'0','\pi/8','\pi/4','3\pi/8', '\pi/2', '5\pi/8', '3\pi/4',
xlabel('frequency');
ylabel('power density estimate');
axis([0 pi-0.5 0 0.0009]);
```

Listing 5: Code for finding x y coordinates of maximum of power spectral density estimate

```
%% code for finding maximum of estimate plot %%
maxF = max(pxx);  % Find max value over all elements.
indexOfFirstMax = find(pxx == maxF, 1, 'first');  % Get first element t
% Get the x and y values at that index.
maxY = pxx(indexOfFirstMax);
maxX = omega(indexOfFirstMax);
```

Listing 6: Code for fitting real PSD to estimated PSD

```

%% constants %%
lambda = 0.09;
sigma = 0.02814;
omega_0 = 0.7823;
K_w = 2*lambda.*omega_0*sigma;
omega = 0:pi/100:2*pi;
%% plotting %%
num = omega.^2*K_w^2
den =(omega.^4 + omega_0^4 + 2*omega.^2*omega_0^2*(2*lambda.^2-1));
psd = rdivide(num,den);
hold on;
plot(omega,psd, '--');
hold on;
title('PSD estimate and actual PSD with different \lambda');
legend('PSD estimate', '\lambda = 1', '\lambda = 0.5', '\lambda = 0.1',
```

A.3 Control system design

Listing 7: Code to plot the step response of the ship and the modelled system

```
%Constants

K_pd = 0.8362;
T_f = 8.39;
psi_r = 30;

%Plotting
compass = importdata('53b_compass.mat'); %Import data from simulink
rudder = importdata('53b_rudder.mat'); %Import data from simulink
reference = importdata('53b_ref.mat'); %Import data from simulink

n=1:length(compass(2,:));


plot(n,compass(2,n), 'color', 'r'); %Plot compass
hold on;
plot(n,rudder(2,n), 'color', 'b'); %Plot rudder
hold on;
plot(n,reference(2,n), 'green--'); %Plot reference

grid on;
xlabel('time'); ylabel('angle');
legend('compass', 'rudder', 'compass reference', 'Location', 'East');
axis([0 800 -20 40]);
```

Listing 8: Code to plot the step response of the ship and the modelled system

```
%Constants

K_pd = 0.8362;
T_d = 72.5216;
T_f = 8.39;
psi_r = 30;

compass = importdata('53c_compass.mat'); %Import data from simulink
rudder = importdata('53c_rudder.mat'); %Import data from simulink
reference = importdata('53c_ref.mat'); %Import data from simulink
```

```
n=1:length(compass(2,:));

plot(n,compass(2,n), 'color', 'r'); %Plot compass
hold on;
plot(n,rudder(2,n), 'color', 'b'); %Plot rudder
hold on;
plot(n,reference(2,n), 'green--'); %Plot reference

grid on;

xlabel('time'); ylabel('angle');
legend('compass', 'rudder', 'reference', 'Location', 'East');
axis([0 500 -20 40]);
```

Listing 9: Code to plot the step response of the ship and the modelled system

```


%Constants

K_pd = 0.8362;
T_d = 72.5216;
T_f = 8.39;
psi_r = 30;

compass = importdata('53d_compass.mat'); %Import data from simulink
rudder = importdata('53d_rudder.mat'); %Import data from simulink
reference = importdata('53d_ref.mat'); %Import data from simulink

n=1:length(compass(2,:));

plot(n,compass(2,n), 'color', 'r'); %Plot compass
hold on;
plot(n,rudder(2,n), 'color', 'b'); %Plot rudder
hold on;
plot(n,reference(2,n), 'green--'); %Plot reference
grid on;
xlabel('time'); ylabel('angle');
legend('compass', 'rudder', 'reference', 'Location', 'Southeast');
axis([0 500 -30 40]);
```

A.4 Observability

Listing 10: Code find observably matrix and its rank in part 6.5

```
w = 0.7823;
h = 0.09;
T = 72.5216;
K = 0.1561;


A = [0 1 0 0 0; -w^2 -2*h*w 0 0 0; 0 0 0 1 0; 0 0 0 -1/T -K/T; 0 0 0 0

C = [0 1 1 0 0];

O = obsv(A,C);
```

A.5 Discrete Kalman filter

Listing 11: Code for finding matrices

```

w_0 = 0.7823;
w_c = 0.10;
K = 0.1561;
T = 72.5216;

K_pd = 0.8362;
T_d = 72.5216;
T_f = 8.39;
psi_r = 30;

lambda = 0.09;
sigma = 0.02814;
K_w = 2*lambda*w_0*sigma;
T_s = 0.10;

A = [0 1 0 0 0;
     -w_0^2 -2*lambda*w_0 0 0 0;
     0 0 0 1 0;
     0 0 0 -1/T -K/T;
     0 0 0 0 0;];
B = [0; 0; 0; K/T; 0];

E = [0 0; K_w 0; 0 0; 0 0; 0 1];

C = [0 1 1 0 0];


```

```
[A_d, B_d] = c2d(A,B, T_s);
[A_d, E_d] = c2d(A,E, T_s);
C_d = C;

Q = [30 0; 0 10^-6];

P_ = [1 0 0 0 0;
       0 0.013 0 0 0;
       0 0 pi^2 0 0;
       0 0 0 1 0;
       0 0 0 0 2.5*10^-3];

x_ =[0; 0; 0; 0; 0];
```

Listing 12: Code for finding variance

```

measurement_noise = importdata('measurement_noise.mat');

variance = var(deg2rad(measurement_noise(2,:)));
```

Listing 13: Code for Kalman filter S-function

```
function [b, psi, psi_w]= fcn(u,y)
persistent init_flag A_d B_d C_d E_d Q P_ x_ R

if isempty(init_flag)
    init_flag = 1;

    A_d = [0.997 0.0992 0 0 0;
           -0.0607 0.9830 0 0 0;
           0 0 1 0.0999 0;
           0 0 0 0.9986 -0.0002;
           0 0 0 0 1;
           ];
    B_d = 1.0e-03*[ 0
            0
            0.0108
            0.2151
            0
            ];
    C_d = [0 1 1 0 0];


    E_d = [0 0; 0.0004 0;
```

```
             0 0; 0 0; 0 0.1];

    Q = diag([30 10^(-7)]);

    P_ = diag([1 0.013 pi^2 1 0.0025]);

    x_ = [0; 0; 0; 0; 0];

    R = 6.1614 * 10^(-7)*10;
end



%compute kalman gain
L = P_*C_d'*(C_d*P_*C_d' + R)^(-1);

%update the estimate with the measurement
x = x_ + L*(y - C_d*x_);

%compute error covariance for updated estimator
I = diag([1 1 1 1 1 ]);
P = (I - L*C_d)*P_*(I-L*C_d)' + L*R*L';

%project ahead
P_ = A_d*P*A_d' + E_d*Q*E_d';
x_ = A_d*x + B_d*u;

psi = x(3);
b = x(5);
psi_w = x(2);
end
```

B Simulink Diagrams

B.1 Identification of the boat parameters

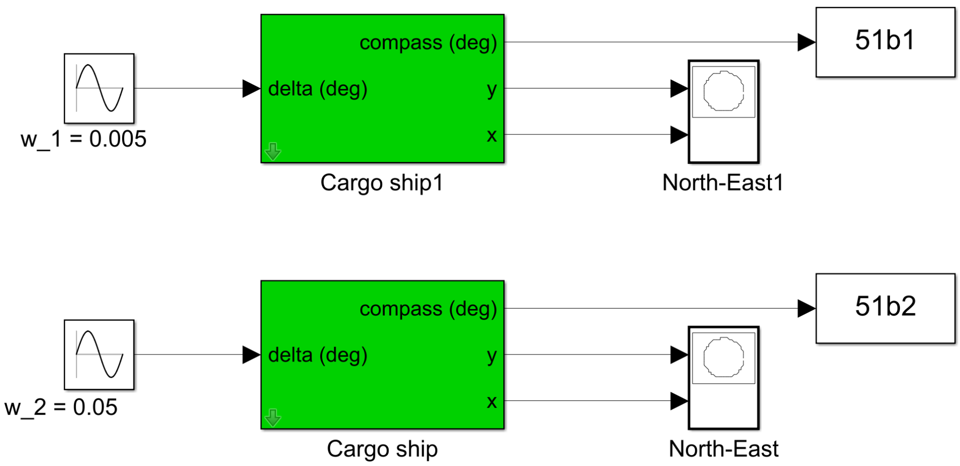

Figure 24: Simulink diagram of with sine inputs to find boat parameters

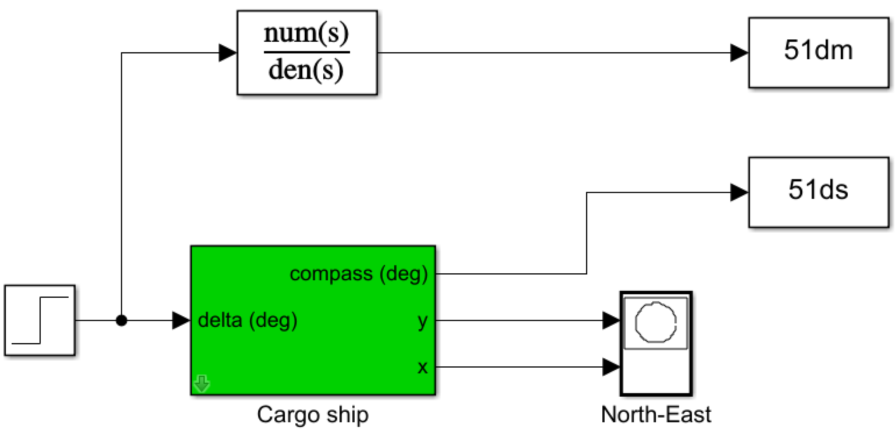

Figure 25: Simulink diagram of the system with step response

B.2 Identification of wave spectrum model

B.3 Control system design

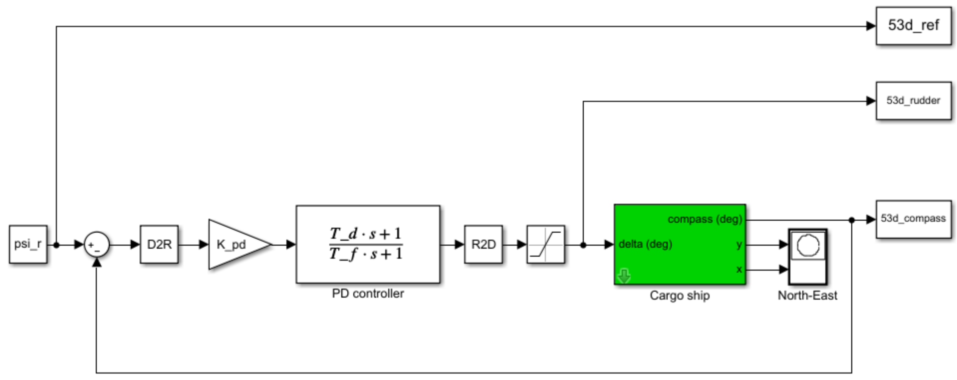

Figure 26: Simulink diagram of the system with a PD controller

B.4 Observability

B.5 Discrete Kalman filter

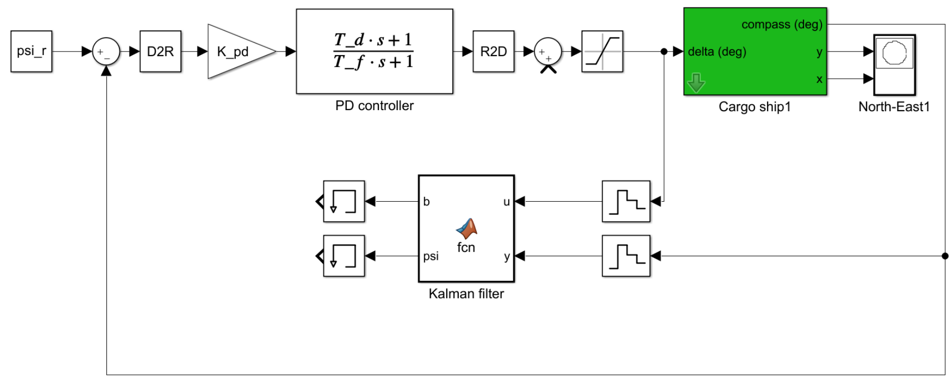

Figure 27: Simulink diagram of the system with Kalman filter

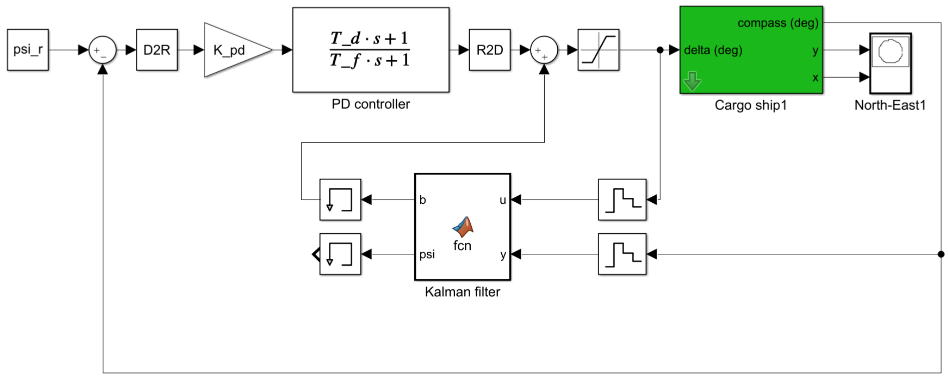

Figure 28: Simulink diagram of the system with Kalman filter with feed forward from the rudder angle bias

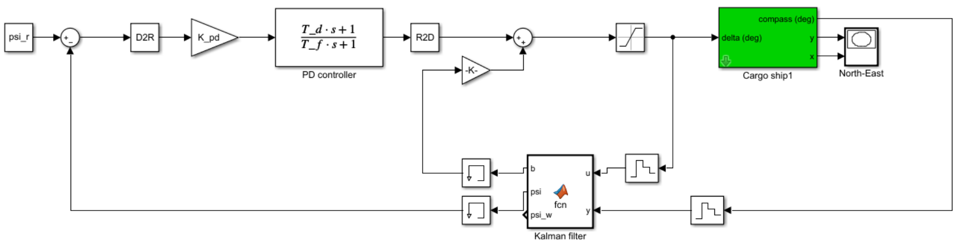

Figure 29: Simulink diagram of the system with Kalman filter using wave filtered heading in feedback