Project assignement in

Linear Systems TTK4115

Discrete Kalman Filtering Applied to a Ship Autopilot

[redacted]

[redacted]

November 19, 2016

NTNU – Trondheim
Norwegian University of
Science and Technology

Contents

1 Identification of the boat parameters

In this section we calculate the transfer function of the boat system and identify the time and gain constants. Furthermore we compare the calculated transfer function with the behaviour of the actual ship-plant.

1.1 Transfer function from δ to ψ

Let the $\psi - \delta$ relation be given as follows

$$\dot{\psi} = r \tag{1}$$

$$\ddot{\psi} = \dot{r} = -\frac{1}{T}r + \frac{K}{T}(\delta - b) \tag{2}$$

With no disturbances (b = 0), the transfer function from δ to ψ is then

$$\ddot{\psi} \xRightarrow{\mathcal{L}} s^2\psi(s) = -\frac{s}{T}\psi(s) + \frac{K}{T}\delta(s)$$

$$\frac{\psi}{\delta}(s) = \frac{K}{s(sT+1)} = H(s) \tag{3}$$

1.2 Identifying T and K

To estimate the parameters T and K, we started by determining the amplitude $|H(j\omega)|$ for the two frequencies $\omega_1 = 0.005$ and $\omega_2 = 0.05$. Using the noiseless plots shown in fig. 1, the respective amplitudes were calculated to be

$$|H(j\omega_1)| = A_1 = \frac{68.98 - 5}{2} = 32 \quad \Big\| \quad |H(j\omega_2)| = A_2 = \frac{4.48 - 2.91}{2} = 0.785$$

This results in

$$|H(j\omega_1)| = A_1 \Rightarrow K = A_1\omega_1\sqrt{\omega_1^2T^2 + 1}$$

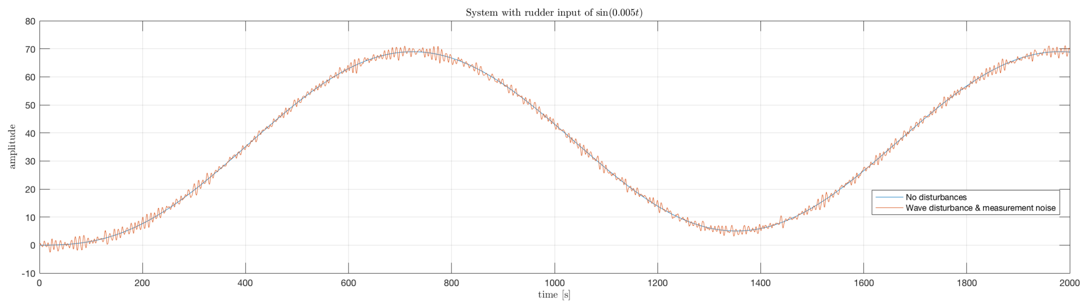
$$|H(j\omega_2)| = A_2 \Rightarrow T = \sqrt{\frac{K^2}{A_2^2\omega_2^4} - \frac{1}{\omega_2^2}}$$

Solving this set of equations gives the parameter values

$$T = \underline{74.4351} \quad \Big\| \quad \underline{K = 0.17072}$$

1.3 Estimating the model parameters with system noise

The 'noisy' plot is shown compared to the noiseless plot in fig. 1. It's fairly easy to see that it becomes difficult to give good parameter estimates, especially for higher frequency inputs. In the case of the lower frequency ($\sin(0.005t)$) it would be possible to 'ballpark' it, if one were to allow a certain tolerance interval.



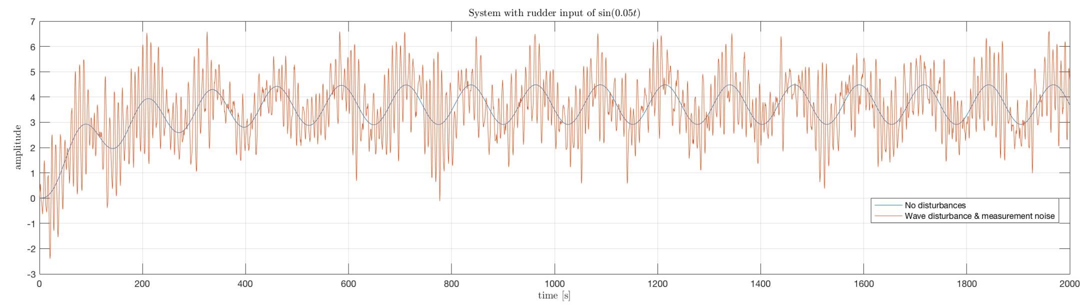


Figure 1: Simulation of the system with input $\sin(0.005t)$ and $\sin(0.05t)$ with and without disturbances.

1.4 Model-Ship comparison

The plot shown in fig. 2 shows that the transfer function model follows the ship model quite well for some time. After $t \approx 650$ it begins to veer off and gets a continuously growing error, and the model breaks down.

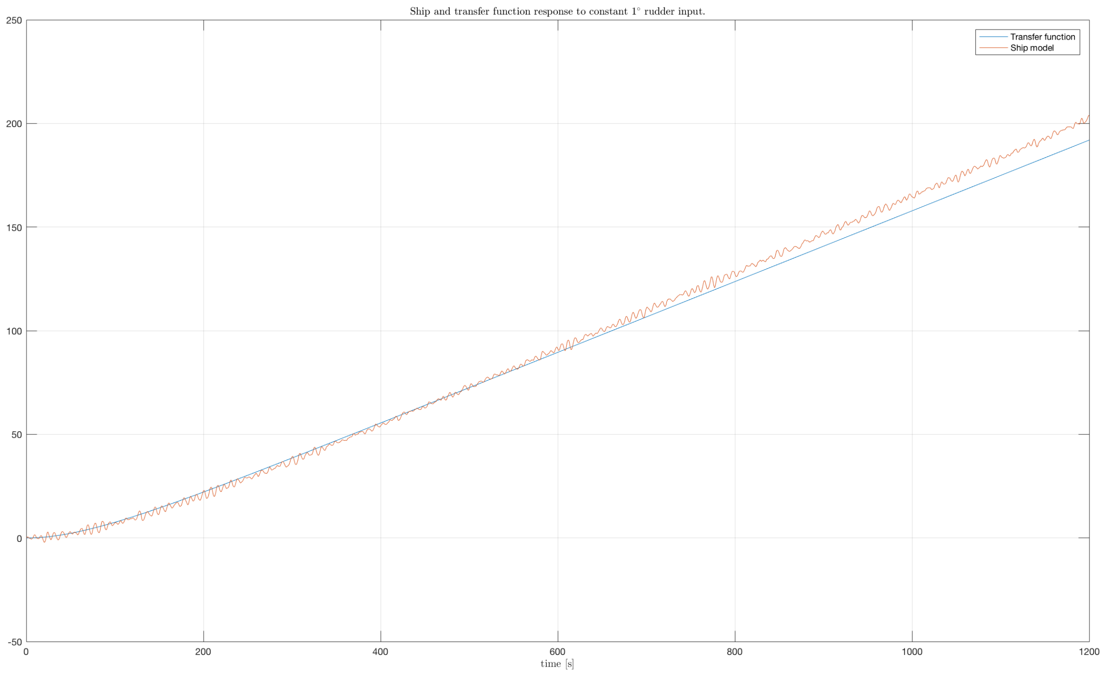


Figure 2: Plot comparing the system model with our transfer function.

2 Identification of wave spectrum model

In this section we estimate the Power Spectral Density of the wave system and use it to identify the parameters of the wave spectrum.

2.1 Estimate of the Power Spectral Density

The data file used (wave.mat) contained the time instants when the ψ_w elements are applied to the system in the first row, and the wave influences on the compass (in degrees) in the second. To calculate the PSD we used the MATLAB-function "pwelch" shown below, with window size 4096 and a sampling frequency of 10Hz.

```
    [pxx, f] = pwelch(psi_w(2,:)*pi/180,4096,[],[],10);
```

2.2 Transfer function from w_w to ψ_w and analytical PSD

We have given the relations

$$\dot{\xi}_w = \psi_w$$

$$\dot{\psi}_w = -\omega_0^2 \xi_w - 2\lambda\omega_0\psi_w + K_w w_w.$$

Taking the Laplace transform, we get

$$s^2\psi_w(s) = -\omega_0^2\xi_w(s) - 2\lambda\omega_0\psi_w + K_w\omega_w(s)$$
$$\xi_w = \frac{\psi_w}{s}$$

which after sorting gives

$$G_{\psi\omega} = \frac{\psi_w}{\omega_w}(s) = \frac{K_w s}{s^2 + 2\lambda\omega_0 s + \omega_0^2} \tag{4}$$

To find an analytical expression for the PSD of ψ_w we use the relation [2]

$$S_x(j\omega) = G_x(j\omega)G_x(-j\omega)S_f(j\omega)$$

where $S_x(j\omega)$ is the power density spectrum of the output signal, $G_x(j\omega)$ is the signal's transfer function and $S_f(j\omega)$ is the power spectral density of the input. In our case, assuming $S_f(j\omega) = 1$ (white noise), it then becomes

$$S_{\psi\omega}(j\omega) = G_{\psi\omega}(j\omega)G_{\psi\omega}(-j\omega) \tag{5}$$

Solving eq. (5) for our system gives the following analytical expression for the power spectrum density

$$P_{\psi_w} = \frac{-K_w^2 s^2}{s^4 + \omega_0^2 s^2 - 4\lambda^2\omega_0^2 s^2 + \omega_0^4} \tag{6}$$

2.3 Resonance frequency

The resonance frequency, ω_0, can be found directly from the estimated PSD as the frequency where of the power peak (resonance peak). From fig. 3 we then get

$$\omega_0 = 0.7823 \quad .$$

This shows that the wave disturbance have the biggest signal influence around a frequency of $\omega = 0.78$.

2.4 Wave damping factor λ

Let $K_w = 2\lambda\omega_0\sigma$, where σ^2 corresponds to the peak of S_{ψ_2}. λ was found by fitting the analytical PSD to the estimated using the MATLAB-command

```
    lambda = lsqcurvefit(P_psi_w_fun, x0, w, pxx);
```

where P_psi_w_fun equals $S_{\psi\omega}(j\omega)$, w is the radian-vector and pxx is the estimated PSD. With $\sigma = \sqrt{0.001484}$ and $\omega_0 = 0.7823$, this results in

$$\lambda = 0.0827 \quad ,$$

with the fitted analytical PSD shown in fig. 3.

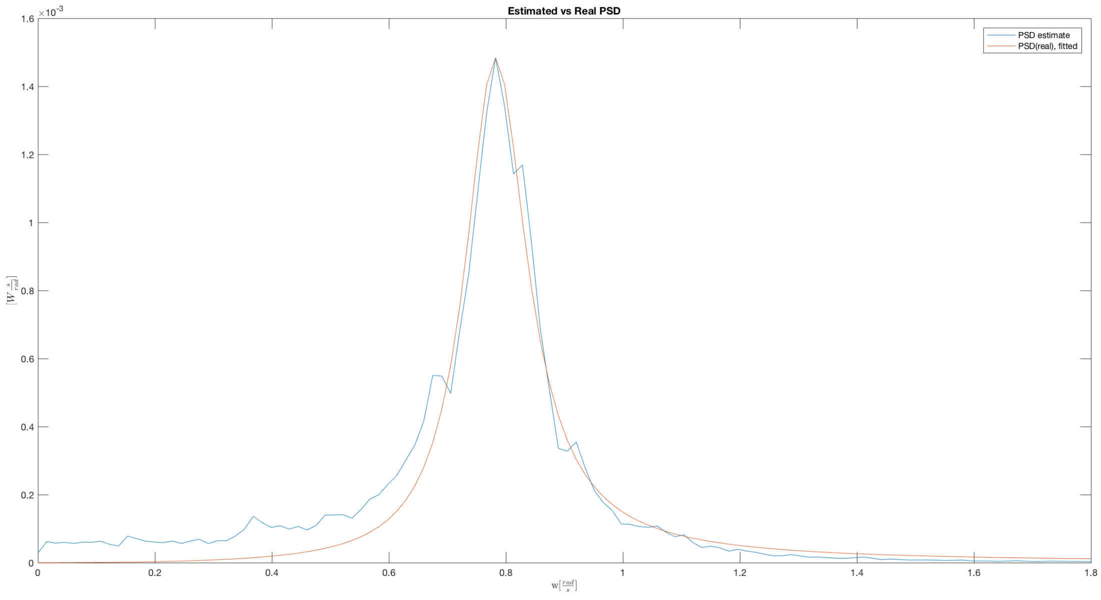


Figure 3: Comparison plot of the estimated and analytical Power Spectral Densities.

3 Control system design

In this section we tried to design and implement a simple ship autopilot to control the course angle ψ_r using a PD controller.

3.1 PD controller design

To design a PD controller such that $\omega_c = 0.10$ and $\phi_m = 50°$ for the open loop system, we use the basic PD transfer function

$$H_{pd} = K_{pd}\frac{1 + T_d s}{1 + T_f s} \quad .$$

The system is then given by

$$H_{sys} = H_{pd}H_{ship} = \frac{K_{pd}K}{s(1 + T_f s)} \quad ,$$

when $T_d = T$. To determine the PD-gain and T_f we let the real part of the system transfer function equal $\cos(\phi_m)$ and the imaginary part equal $\sin(\phi_m)$, and then evaluating at $\omega = \omega_c$. This gives

$$\Re(H_{sys}) = \frac{KT}{T^2\omega^2 + 1}\Big|_{\omega=\omega_c} = \cos(50°)$$
$$\Im(H_{sys}) = \frac{K}{T^2\omega^3 + \omega}\Big|_{\omega=\omega_c} = \sin(50°)$$

Solving these results in

$$K_{pd} = \frac{0.1305}{K} = \underline{\underline{0.765}} \quad \Big\| \quad T_f = \underline{\underline{8.39}}$$

The parameters are verified by the bode plot of the system transfer function shown in fig. 4.

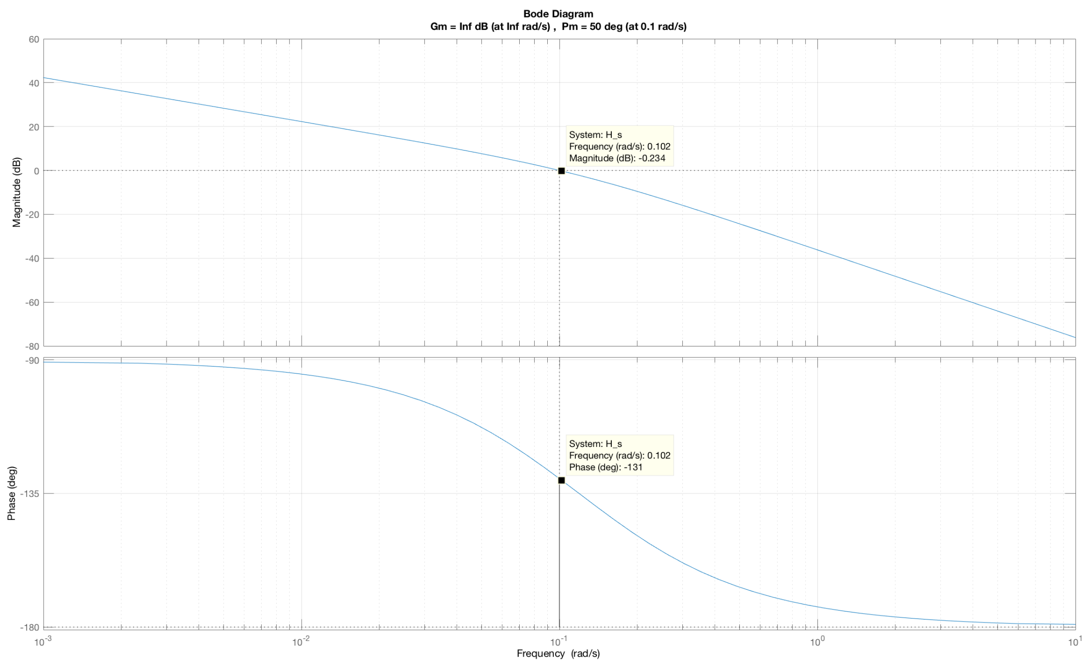


Figure 4: Bode plot of the system transfer function with PD controller with calculated parameters T_f and K_{pd}.

3.2 Autopilot without disturbances

Simulating the system without any disturbances gives a good and relatively fast response as shown in fig. 5. The accompanied rudder plot also shows that the rudder stabilizes itself around 0 with minute variations. This is to be expected and follows from the final value theorem, with reference R(s) as a step function:

$$H(s) = \frac{K_T}{T_f s^2 + s}$$

$$H_{ship}(s) = \frac{K}{Ts^2 + s}$$

$$E(s) = \frac{R(s)}{1 + H(s)}$$

$$\lim_{s \to 0} sE(s) = \lim_{s \to 0} \left(\frac{sR(s)}{\frac{KK_{pd}}{T_f s^2 + s} + 1} \right) = \underline{\underline{0}}$$

3.3 Autopilot with current disturbances

Simulating the system with the tuned PD controller and current disturbances gives a fairly poor response. In this case the autopilot fails to reach the reference heading at 30°, due to the nature of the current cisturbance, resulting in a steady-state error of approximately 4°. This error could however possibly be overcome by implementing a PID controller, integrating and eventually negating the deviance.

3.4 Autopilot with wave disturbances

Simulating the system again, now with only the wave disturbance, it is clear from the plots the system does not work satisfactory. From the plot fig. 5 one can see large oscillations in both the boat's heading and the rudder input. Given the nature of the derivative term of the regulator high frequency noise such as the wave disturbance will be amplified and causing such a result. Oscillations like these in the rudder input would cause strain on the rudders actuator causing wear.[1] The oscillations could be overcome by applying a low pass filter to system and attenuate the high frequency noise.

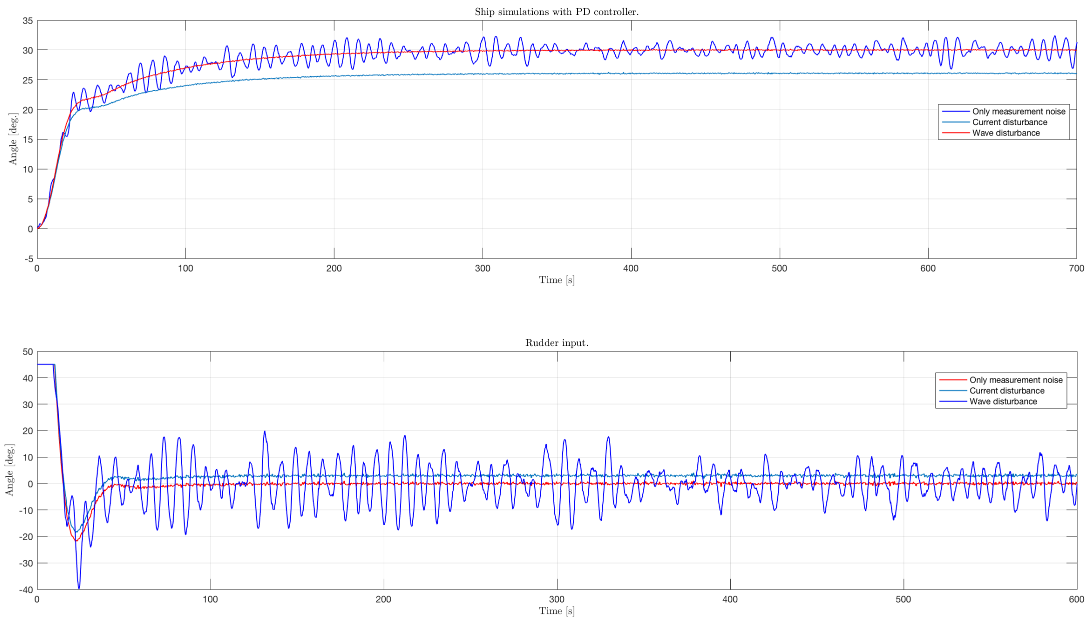


Figure 5: Plots of the compass course and rudder input with no disturbance, current disturbance and wave disturbance respectively.

[1]More practically it would make for an uncomfortable ride for potential passengers.

4 Observability

In this section we derive the state space model of the system and check the observability of the different parts of the system.

4.1 Matrices A, B, C, E

Using equations 13(a-f) in the assignment[1] we set

$$\begin{bmatrix} \dot{\xi}_w \\ \dot{\psi}_w \\ \dot{\psi} \\ \dot{r} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix}$$

thus

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= K_w w_w - 2\lambda\omega_0 x_1 \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \frac{K}{T}(\delta - x_5) - \frac{1}{T}x_4 \\ \dot{x}_5 &= w_b \\ y &= x_2 + x_3 + v \end{aligned}$$

by simply putting these equations into matrices we get the following

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

4.2 No disturbance

The system without any disturbance is the same as previous state space model with only the states x_3 and x_4, and is reduced to

$$A = \begin{bmatrix} 0 & 1 \\ 0 & \frac{1}{T} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{7}$$

with state vector $\begin{bmatrix} \dot{\psi} & \dot{r} \end{bmatrix}^T$, thus the observability matrix

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

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

which has $rank = 2$, thus the system is observable.

4.3 Current disturbance

To find the observability matrix corresponding to the system with current disturbance we removed the states corresponding to $x_1 = \xi_w$ and $x-_2 = \psi_w 2$ from A and C, and calculated the following matrix

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{T} & \frac{K}{T} \end{bmatrix} = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0.0134 & 0.0023 \end{bmatrix}$$

which has $rank = 3$, thus the system is observable.

4.4 Wave disturbance

To find the observability matrix corresponding to the system with wave disturbance we removed the states x_5, and calculated the following matrix

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -{\omega_0}^2 & -2\lambda\omega_0 & 0 & 1 \\ 2\lambda\omega_0^3 & 4\lambda^2\omega_0^2 - \omega_0^2 & 0 & \frac{1}{T^2} \\ \omega_0^4 - 4\lambda^2\omega_0^4 & 2\lambda\omega_0^3 - 2\lambda\omega_0(4\lambda^2\omega_0^2 - \omega_0^2) & 0 & \frac{1}{T^3} \end{bmatrix} = \begin{bmatrix} 0 & 1.0 & 1.0 & 0 \\ -0.6120 & -0.1294 & 0 & 1.0 \\ 0.0792 & -0.5953 & 0 & 0.0134 \\ 0.3643 & 0.1562 & 0 & 0.0002 \end{bmatrix}$$

which has $rank = 4$, thus the system is observable.

4.5 Entire system

Finding the observability matrix for the entire system is done by simply using the entire A matrix with C and gives the matrix

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ -{\omega_0}^2 & -2\lambda\omega_0 & 0 & 1 & 0 \\ 2\lambda\omega_0^3 & \omega_0^2(4\lambda^2 - 1) & 0 & \frac{1}{T^2} & \frac{K}{T} \\ \omega_0^4(1 - 4\lambda^2) & 2\lambda\omega_0^3(1 - (4\lambda^2 - 1)) & 0 & \frac{1}{T^3} & (\frac{K}{T})^2 \\ 2\lambda\omega_0^3(4\lambda^2\omega_0^2 - \omega_0^2) - 2\lambda\omega_0^5 & (4\lambda^2\omega_0^2 - \omega_0^2)^2 - 4\lambda^2\omega_0^4 & 0 & \frac{1}{T^4} & (\frac{K}{T})^3 \end{bmatrix}$$

$$\mathcal{O} = \begin{bmatrix} 01.01.000 \\ -0,6112 & -0,1293 & 0 & 1 & 0 \\ 0,0792 & -0,5953 & 0 & 0,0134 & 0,0023 \\ 0,3643 & 0,1562 & 0 & 0,0002 & 3,081 \cdot 10^{-5} \\ -0,0956 & 0,3441 & 0 & 2,424 \cdot 10^{-6} & 4,139 \cdot 10^{-7} \end{bmatrix}$$

which has $rank = 5$, thus also this is observable. This means that the system is observable with and without disturbances, and as such the Kalman filter can be applied to the system in all cases.

5 Discrete Kalman filter

In this section we implement the discrete Kalman filter to estimate the rudder bias and heading of the boat.

5.1 Discretization of state-space model

The exact discretized model was found using the MATLAB-command "c2d(A,B,Fs)", where Fs is the sample frequency set to 10 Hz, as shown below.

```
    [Ad, Bd] = c2d(A,B,0.1);
    [Ad, Ed] = c2d(A,E,0.1);
     Cd = C;
     Dd = D;
```

This results in the following discretized model:

$$\dot{x} = \begin{bmatrix} 0.9970 & 0.0993 & 0 & 0 & 0 \\ -0.0607 & 0.9841 & 0 & 0 & 0 \\ 0 & 0 & 1.0 & 0.0999 & 0 \\ 0 & 0 & 0 & 0.9987 & -0.0002 \\ 0 & 0 & 0 & 0 & 1.0 \end{bmatrix} \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \\ b \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0.0115 \cdot 10^{-3} \\ 0.2292 \cdot 10^{-3} \\ 0 \end{bmatrix} \cdot \delta + \begin{bmatrix} 0 & 0 \\ 0.0005 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.1 \end{bmatrix} \begin{bmatrix} w_w \\ w_b \end{bmatrix}$$

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

5.2 Estimate of the noise variance

An estimate for the variance of the measurement noise was found by applying zero input to the system and then using the MATLAB-command "var(x)" on the measured output. This gave us a variance of $\frac{0.002}{0.1} = 0.02$.

5.3 Kalman filter implementation

When implementing the Kalman filter we went for alternative two described in [1]; writing a Matlab function within a Matlab function-block in Simulink. The function is shown in appendix B. Figures 6 and 7 shows the Kalman estimated compass heading when the system is exposed to the different disturbances. In fig. 7 we can see that we get some initial oscillations from the Kalman filter. This is probably a result of our chosen filter implementation, where a memory-block is needed to initialize the filter. Thus the filter sees only the assorted disturbances on the input initially and responds accordingly (fig. 8). This could possibly be corrected by implementing the Kalman filter in another manner or maybe by adjusting the initial values of the memory-block(?).

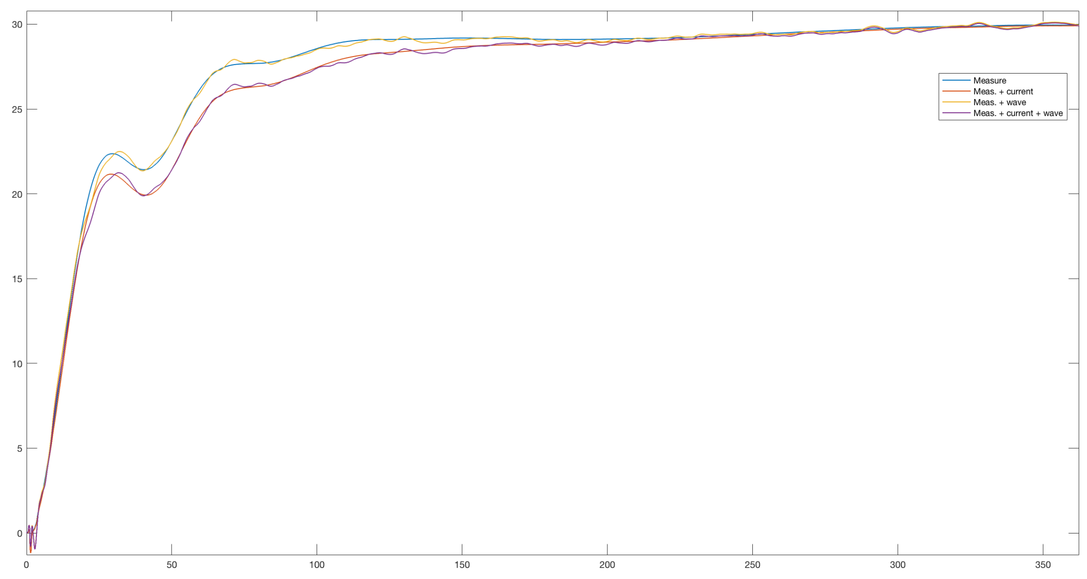


Figure 6: Plot showing the implemented Kalman filter when exposed to different disturbances.

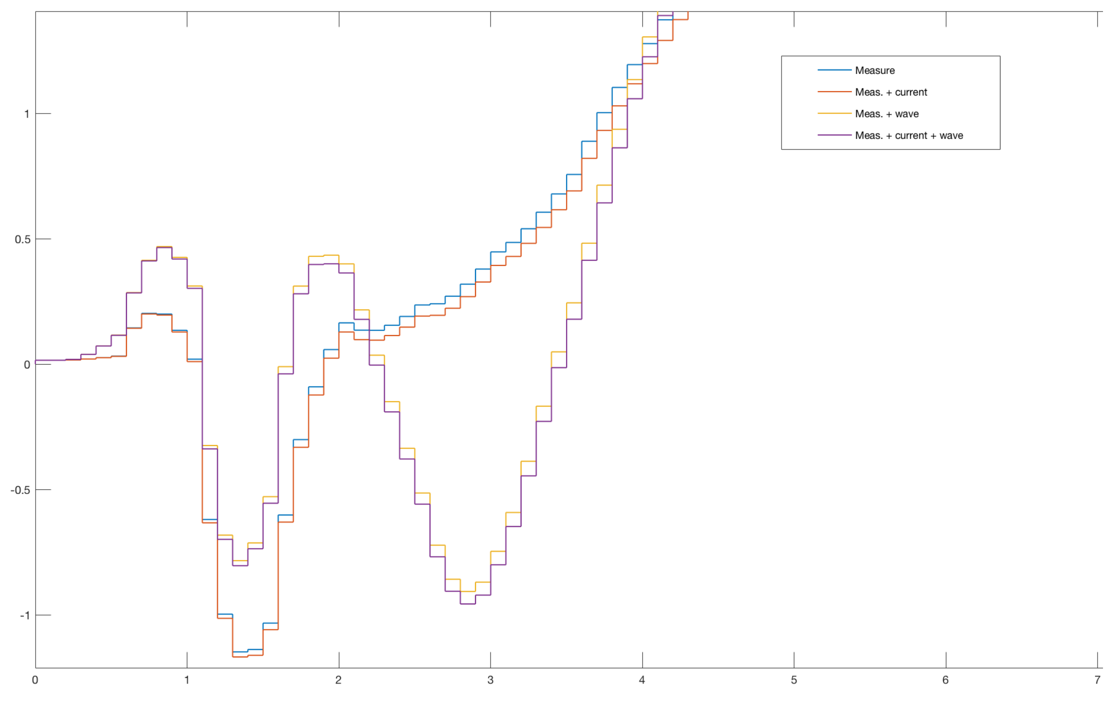


Figure 7: Plot showing the implemented Kalman filter when exposed to different disturbances at the start of simulation.

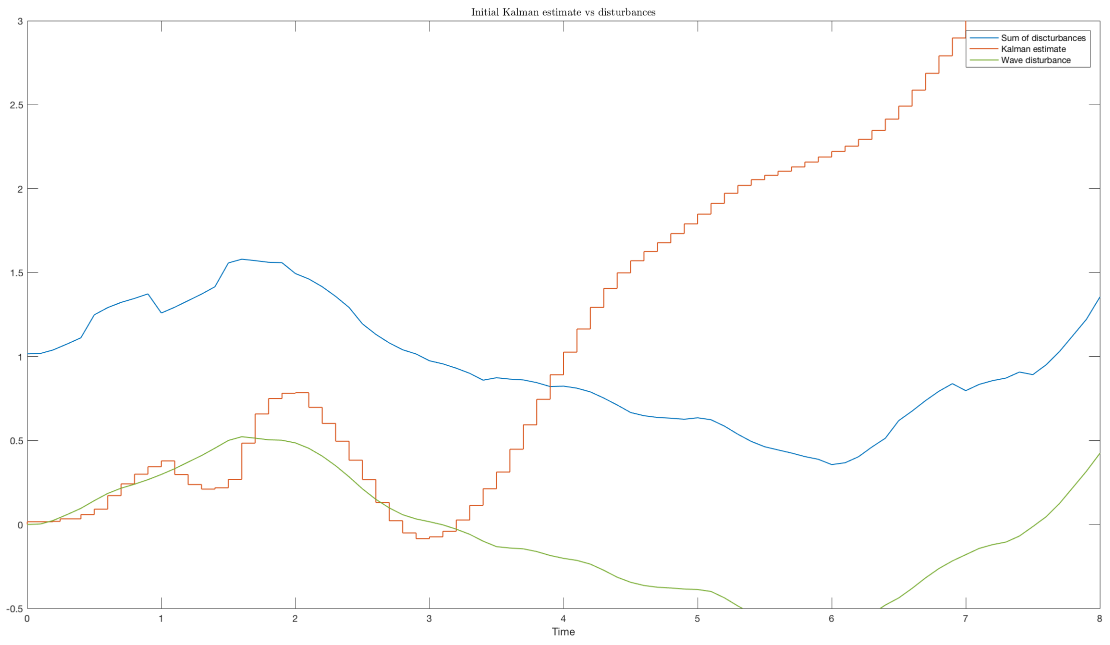


Figure 8: Plot showing the initial comparison between the Kalman estimate (exposed to all disturbances), wave disturbance and the total summed disturbance.

5.4 Estimated bias feedforward

Simulating the system using a bias feedforward, current disturbance and reference input-$\psi_r = 30°$ (shown in appendix C), and comparing it to the response from section 3.3, resulted in the plot shown in fig. 9. From this we can see that the autopilot has a much better performance with the Kalman filter, and actually reaches the reference heading without an added integrator element. This is due to the estimated bias feedforward, shown in fig. 10, which negates the current-induced stationary deviation.

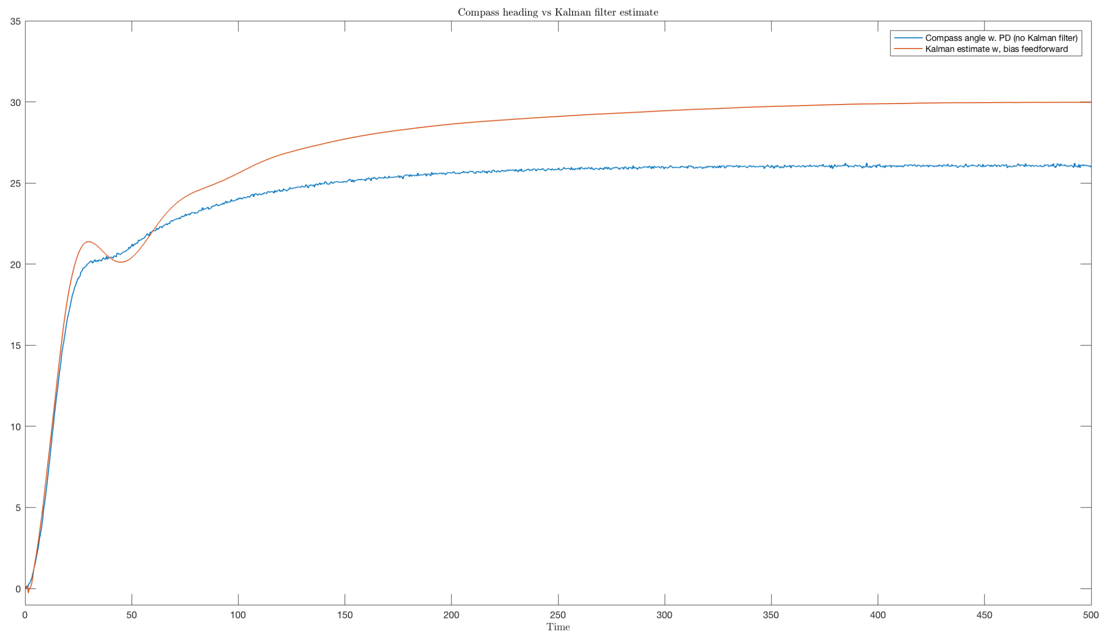


Figure 9: Comparison between the estimated Kalman heading with bias-feedforward and the measured compass heading using only a PD controller.

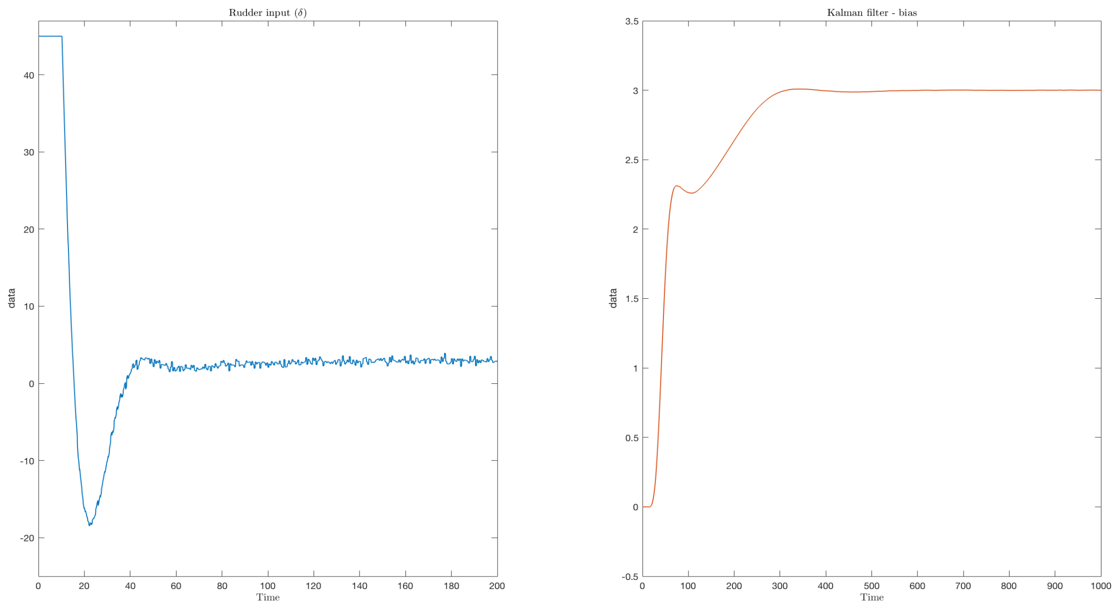


Figure 10: Plot of the rudder input and Kalman-estimated bias when simulating the system with measurement noise and current disturbance.

5.5 Using wave filtered ψ

Simulating the system again, using the wave filtered ψ (Kalman estimate) as feedback, resulted again in an improved autopilot compared to the response from section 3.3/3.4, since it again actually reaches the reference heading (due to bias feedforward). The comparison between the measured and estimated heading is shown in fig. 11, and shows that the Kalman filter gives a pretty good estimate, as expected. There is also a vast improvement in the rudder input compared to the result in fig. 10, where we saw very rapid oscillations. It keeps closer to a stationary value than and will cause less strain on the actuators in the system.

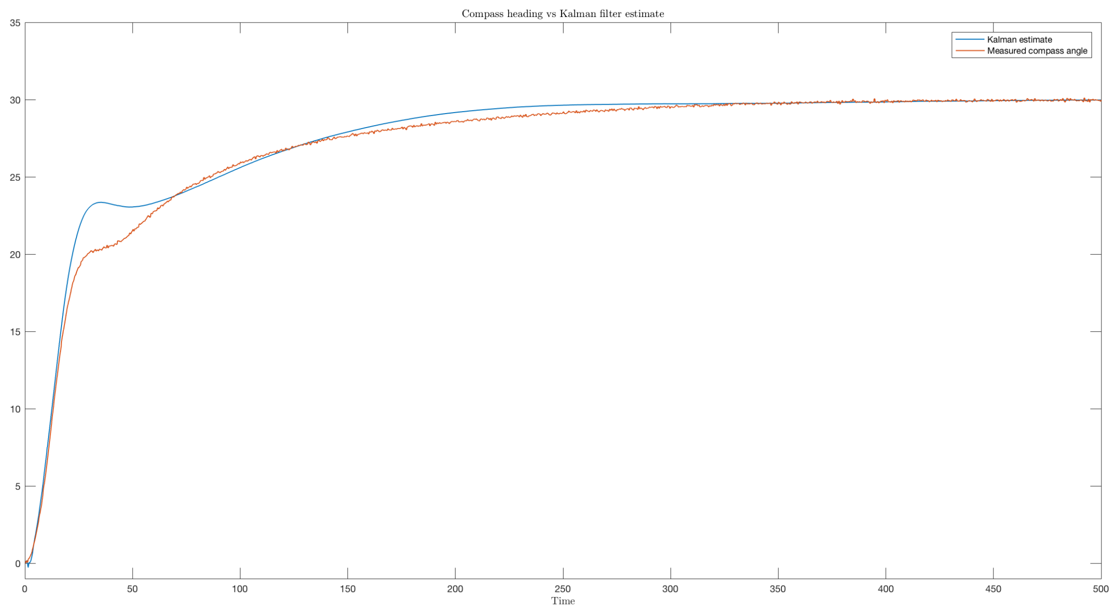
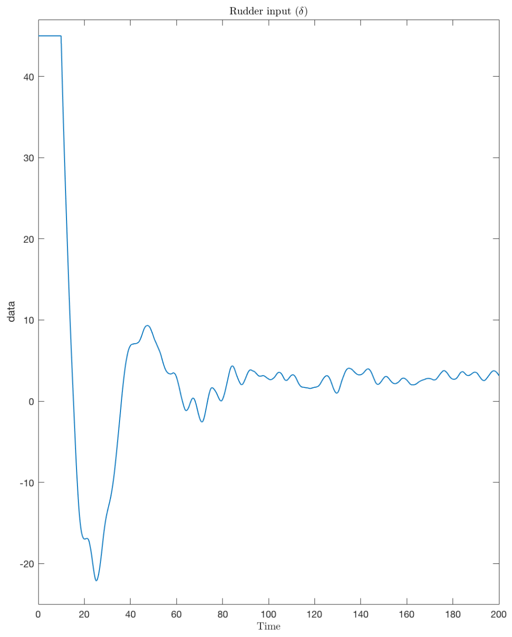


Figure 11: Comparison between the actual and estimated compass heading using bias-feedforward and wave filtered ψ as feedback.



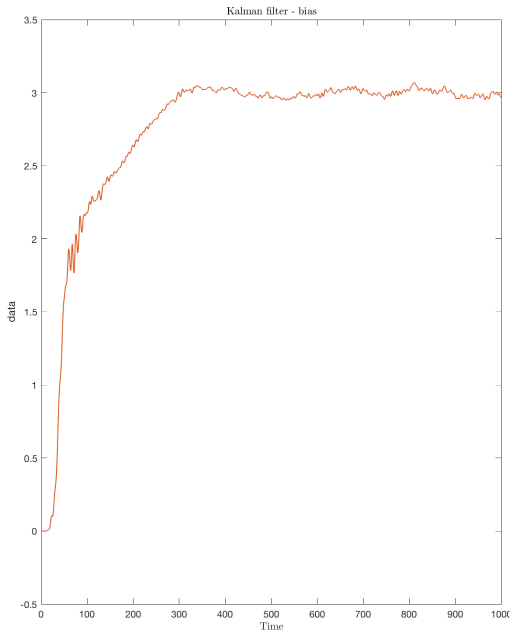
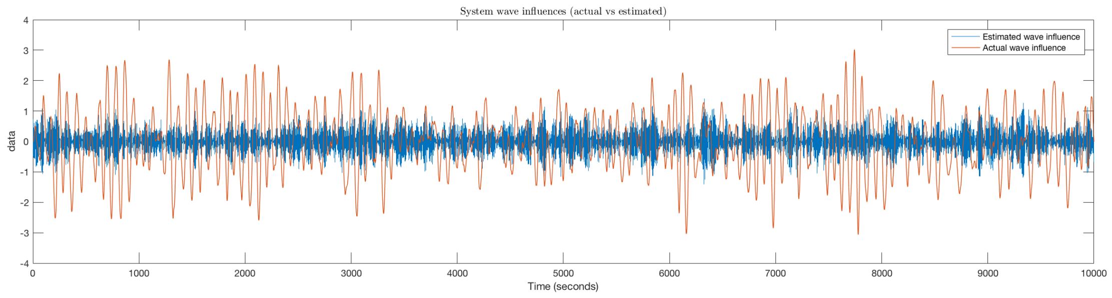


Figure 12: Plots of the rudder input and the estimated bias using wave filtered ψ with all disturbances.



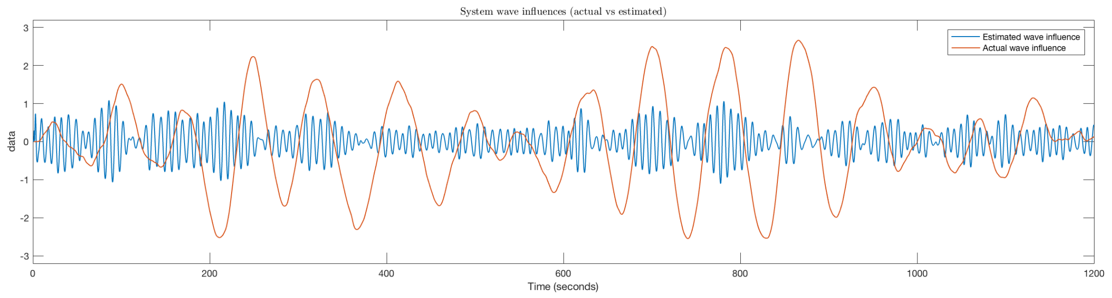


Figure 13: Plots of the actual and estimated wave influences.

References

[1] Department of Engineering Cybernetics, NTNU, *Project Assignement in TTK4115 - Linear systems, Discrete Kalman Filtering Applied to a Ship Autopilot*, Accessed: 30-9-2016

[2] David Brown, Patrick Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering, 4. Edition, 2012*, John Wiley and Sons Ltd

A Matlab: Initialisation and plotting

```
% Assorted code snippets for the TTK4115-Linear Systemstheory Boat
    assignement
%% Variable initialization
w0 = 0.7823;
T = 74.435;
K = 0.1707;
T_d = T;
T_f = 8.391;
K_pd = 0.7647;
phi = 30;
x = 0.0827;
sigma = sqrt(0.001484);
%
set(0,'DefaultTextInterpreter', 'latex')
%
%% Task 5.1 - Identification of the boat parameters
figure()
set(gcf,'color','w');
subplot(2,1,1); plot(noDist1); hold on; grid on;
                plot(waveNmeasure1);
                title('System with rudder input of $\sin(0.005t)$')
            legend('No disturbances','Wave disturbance & measurement
               noise')
                xlabel('time [s]','Interpreter','latex');
                ylabel('amplitude','Interpreter','latex')
subplot(2,1,2); plot(noDist2); hold on; grid on;
                plot(waveNmeasure2);
                title('System with rudder input of $\sin(0.05t)$')
            legend('No disturbances','Wave disturbance & measurement
               noise')
                xlabel('time [s]','Interpreter','latex')
                ylabel('amplitude','Interpreter','latex')

%% Task 5.1 d) - Transfer function
figure()
set(gcf,'color','w')
plot(transfer);     hold on;     grid on;
plot(ship);
title('Ship and transfer function response to constant rudder input.'
    )
legend('Transfer function','Ship model')
xlabel('time [s]','Interpreter','latex')

%% Task 5.2 - Identification of wave spectrum model
[pxx,f] = pwelch(psi_w(2,:)*pi/180,4096,[],[],10);

pxx = pxx/(2*pi);        w = f*2*pi;     % Scaling
```

```matlab

w0 = 0.7823;                        % Found from plot
sigma = sqrt(0.001484);             % --//--
L = 1;

P_psi_w_fun = @(L,w) (4.*L^2.*w0^2.*sigma^2.*w.^2)./(w.^4 + (4.*L^2-2).*w0^2*w.^2 + w0^4);

x = lsqcurvefit(P_psi_w_fun,L,w,pxx); % Using curvefit to find best lambda

P_psi_w = (4.*x^2.*w0^2.*sigma^2.*w.^2)./(w.^4 + (4.*x^2-2).*w0^2*w.^2 + w0^4);

figure()
plot(w,pxx);          axis([0 1.8 0 0.0016]);    hold on;
plot(w,P_psi_w);      legend('PSD␣estimate','PSD(real),␣fitted')
title('Estimated␣vs␣Real␣PSD')
xlabel('w$[\frac{{rad}}{{s}}]$','Interpreter','latex');
ylabel('$[W\frac{s}{rad}]$','Interpreter','latex')

%% Task 5.3 - Control system design
T = 74.435;
K = 0.1707;
T_d = T;
T_f = 8.391;          %Found by comparing the real and complex values of
K_pd = 0.7647;        % H_s(jw) with cos(phi) and sin(phi) respectively,
                      % with w = w_c = 0.1 and phi = 50 degrees.

H_pd = tf([T_d 1 0]);
H_ship = tf([T_f*T T_f+T 1 0]);
H_s = tf(K_pd*K*[1],[T_f 1 0])
figure()
set(gcf,'color','w')
margin(H_s); grid on;
%bode(feedback(H_s,1))

%% Task 5.3 b-d)
phi = 30;
figure()
set(gcf,'color','w');
% Compass plots
subplot(2,1,1); plot(compass_wave,'b');   grid on;    hold on;
                title('Compass␣with␣ref.␣30␣deg.')
                plot(compass_current);      plot(compass,'r')
legend('Only␣measurement␣noise', 'Current␣disturbance', 'Wave␣disturbance')
```

```
title('Ship␣simulations␣with␣PD␣controller.','Interpreter','latex')
xlabel('Time␣[s]', 'Interpreter', 'latex');
ylabel('Angle␣[deg.]', 'Interpreter', 'latex')
axis([0,800,-5,35]);
% Rudder input plots
subplot(2,1,2); plot(rudder,'r');   grid on;    hold on;    title('Rudder')
                plot(rudder_current);       plot(rudder_wave,'b');
legend('Only␣measurement␣noise', 'Current␣disturbance', 'Wave␣disturbance');
title('Rudder␣input.')
xlabel('Time␣[s]');     ylabel('Angle␣[deg.]');      axis([0,600,-50,230]);

%% Task 5.4 - Discrete State-space Model
A = [0 1 0 0 0;
     -w0^2 -2*x*w0 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 2*x*w0*sigma 0 0 0;
      0 0 0 0 1]';
 C = [0 1 1 0 0];
 D = 1;

 [Ad, Bd] = c2d(A,B,0.1);
 [Ad, Ed] = c2d(A,E,0.1);
 Cd = C;
 Dd = D;

 Q = [30 0;
      0 10^(-6)];
 P0_ = [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^(-4)];
 x0h_ = [0 0 0 0 0]';
 R = 0.002/0.1;
 P0 = P0_;
 I = eye(5);
 model_struct = struct('Ad',Ad, 'Bd',Bd, 'Ed',Ed, 'Cd',Cd, 'Q',Q, 'R',R, 'P0_',P0_, 'x0_h_',x0h_, 'I',I);
```

B Matlab: Kalman filter

```
    function output = DiscreteKalman( y, u, model_struct)
%DISCRETEKALMAN
%   - MATLAB function implementation of a discrete Kalman filter -
%   System model on the form:
%       x_dot = Ad*x + Bd*u + Ed*w
%       y = Cd*x + v (noise)

persistent init_flag P_pri x_est_pri
Ad = model_struct.Ad;
Bd = model_struct.Bd;
Ed = model_struct.Ed;
Cd = model_struct.Cd;
Q = model_struct.Q;
R = model_struct.R;
I = model_struct.I;

if isempty(init_flag)
    init_flag = 1;
    x_0_est_pri = [0;0;0;0;0];
    P_0_pri = [1 0 0 0 0;
               0 0.13 0 0 0;
               0 0 pi^2 0 0;
               0 0 0 1 0;
               0 0 0 0 2.5*10^(-4)];
    P_pri = P_0_pri;
    x_est_pri = x_0_est_pri;
end

% Kalman gain
L = P_pri * Cd' * inv(Cd * P_pri * Cd' + R);

% Update estimate
x_est_post = x_est_pri + L * (y - Cd * x_est_pri);

% Update error cov. matrix
P = (I - L*Cd) * P_pri * (I - L*Cd)' + L * R * L';

x_est_next_pri = Ad * x_est_post + Bd*u;
P_next_pri = Ad*P*Ad' + Ed*Q*Ed';

x_est_pri = x_est_next_pri;
P_pri = P_next_pri;

output = [x_est_post];
end
```

C Simulink diagrams

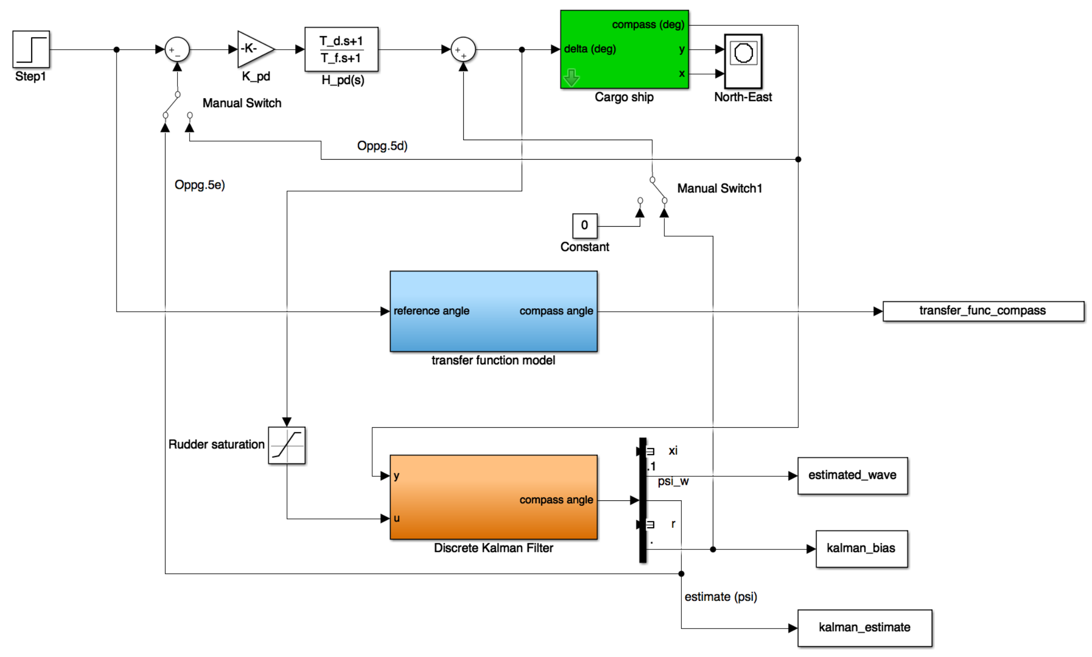


Figure 14: Simulink diagram of the final system including Kalman filter.

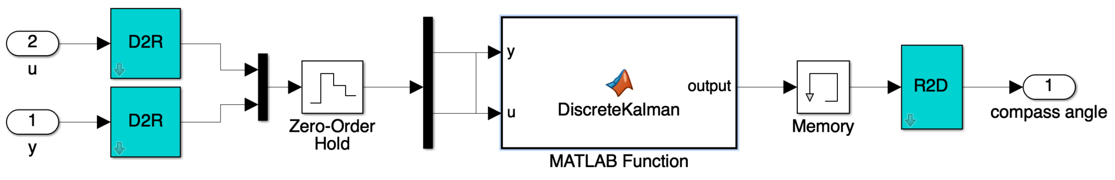


Figure 15: Simulink diagram of the kalman filter implementation using a MATLAB Function-block.