TTK4115 Linear System Theory
Lab Report

755488 and 677379
Group 39

November 1, 2016

Department of Engineering Cybernetics
Norwegian University of Science and Technology

Abstract

This project has give us the task to model, control and simulating a continuous system influenced by stochastic signals. More specific: make an autopilot for a ship experiencing waves, currents and measurement noise on the compass heading sensor. To overcome the effect of waves, current and measurement noise on the controller, a Kalman filter was implemented. Controller performance improved dramatically when working in pair with the Kalman filter.

Contents

1 Identification of the boat parameters

1.1 Introduction

The purpose of this assignment is to model and simulate a continuous system influenced by stochastic signals. Also, a simple autopilot will be designed, and a discrete Kalman filter will be used for wave filtering and disturbance estimation.

1.2 The stated space model

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

The system state space equation can be represented as

$$\dot{x} = Ax + Bu + Ew \tag{2a}$$
$$y = Cx + v \tag{2b}$$

This system will be used in the rest of the report.

1.3 Calculating the transfer function H(s)

Computing the transfer function as

$$\mathcal{L}\left\{\ddot{\psi}\right\} = \mathcal{L}\left\{-\frac{1}{T}r + \frac{K}{T}(\delta - b)\right\}$$

By substitute equation (1c) in to (1d)

$$\begin{aligned} s^2\psi &= -\frac{1}{T}s\psi + \frac{K}{T}\delta \\ \frac{\psi}{\delta} &= \frac{K}{Ts^2 + s} \\ H(s) &= \frac{K}{Ts^2 + s} \end{aligned}$$

The transfer function becomes

$$H(s)_{ship} = \frac{K}{Ts^2 + s} \tag{4}$$

1.4 Identifying the K and T parameters

To determine the K and T parameters, two sine inputs to the rudder with amplitude 1 and frequency $\omega_1 = 0.005$ and $\omega_2 = 0.05$ was applied with all disturbances turned off. The resulting compass heading amplitude is the equal to the magnitude of the transfer function $|H(s)|$, since the input is equal to 1. K and T can then be determined by using equation 5

$$A_{compass} = \frac{K}{\sqrt{T^2(j\omega)^4 + (j\omega)^2}} \tag{5}$$

Expressions for T and K in equation 6 and 7 are derived expressions 5 where the results from both sine inputs have been included.

$$T = \sqrt{\frac{A_2^2\omega_2^2 - A_1^2\omega_1^2}{A_1^2\omega_1^4 - A_2^2\omega_2^4}} \tag{6}$$

$$K = A_1\sqrt{T^2\omega_1^4 + \omega_1^2} \tag{7}$$

A_1 and A_2 is the amplitudes of the measured compass heading with the corresponding sine input. Values for A_1 and A_2 was found in the plots of the response given in figure 1 and 2

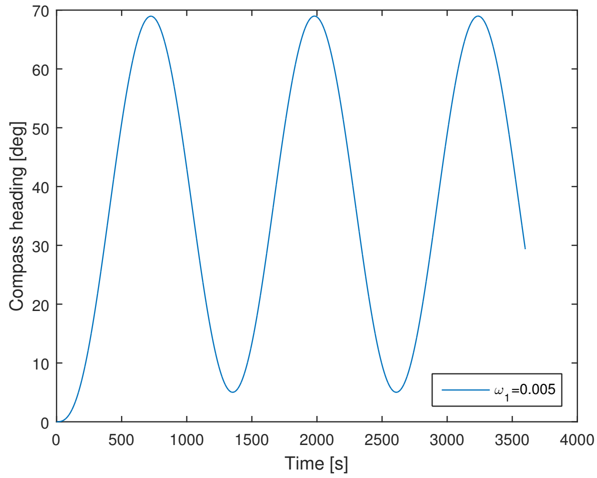

Figure 1: Response with sine input and $\omega_1 = 0.005$

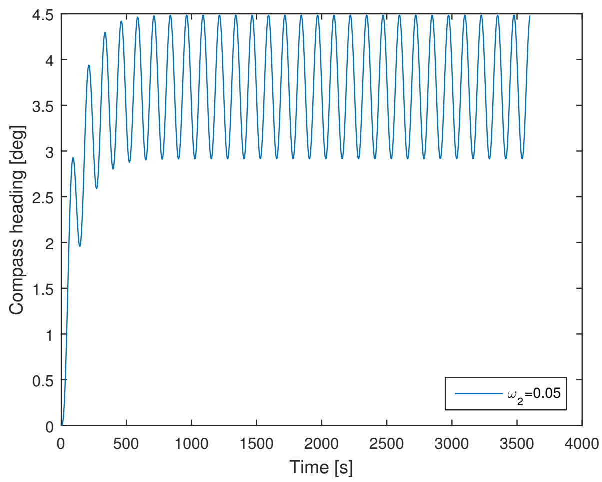

Figure 2: Response with sine input and $\omega_2 = 0.05$

A_1 was determined to be 31.9795 while A_2 was determined to be 0.7850. These amplitude values yields K=0.1742 and T=86.4855.

1.5 Identifying the K and T parameters with waves and measurement noise

The possibility to establish the K and T parameters from experiments with wave and measurement noise turned on, was also evaluated. With $\omega_1 = 0.005$ it is still possible to get a fairly good estimate of the amplitude as seen in figure 4. However for $\omega_2 = 0.05$ it is very difficult to get a good estimate for the amplitude as seen in figure 3. Determination of K and T with wave and measurement noise will require low input frequencies to be able to distinguish the effect of control input and the effect of wave and measurement noise.

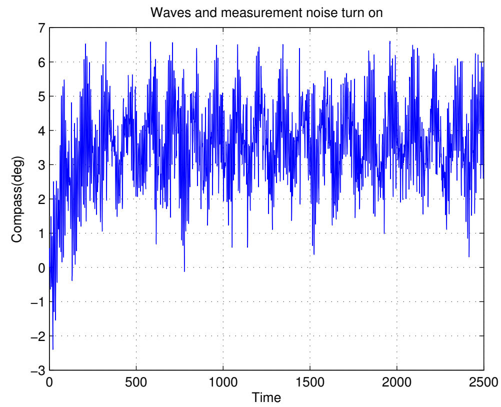

Figure 3: Plot with noise by amplitude 1 and frequency ω_1=0.005

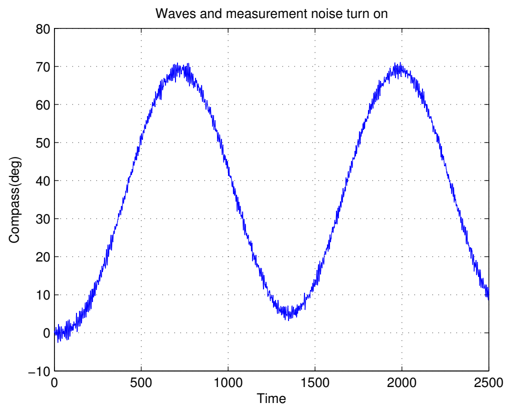

Figure 4: Plot with noise by amplitude 1 and frequency ω_2=0.05

1.6 Step response comparison

The ship transfer function and the more complex ship model has been compared by inputting a step response to both the transfer function and ship model. Plotting the resulting compass headings in figure 5 show how the compass heading deviates with time.

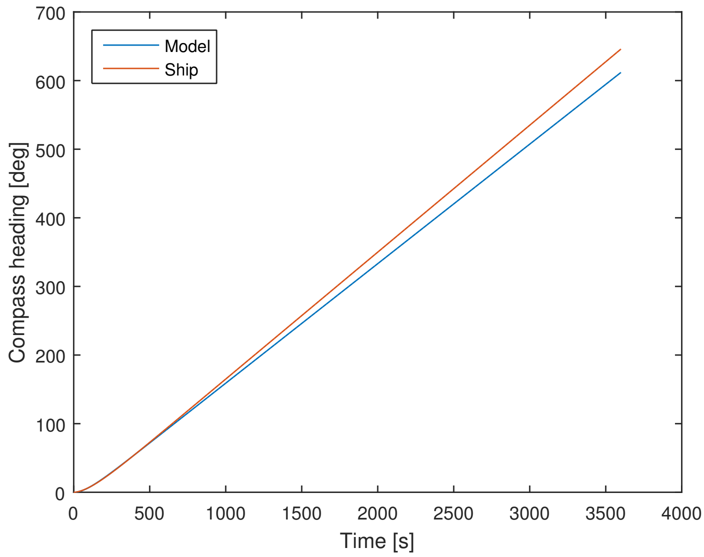

Figure 5: Comparison of unit step of model and ship

The model is a fairly good model for the ship, however as the model is not a perfect representation of the ship the error between ship heading and model heading increase with time. After an hour the error between the model and the ship is 34°. The difference is expected as the transfer function is a simplification of the real ship dynamics.

2 Identification of wave spectrum model

2.1 Finding an estimate of $S_{\psi\omega}$ using pwelch()

To analyse the wave distribution we used MATLAB and Welch's method. The function `pwelch` gives a power spectral estimate of the input signal. The wave input is available in the file wave.mat.We used a sampling frequency of 10 Hz and a window size of 4096. To get the outputs to a right scale we multiplied pxx and f with $\frac{1}{2\pi}$ and 2π respectively. Figure 6 presents the resulting power spectral estimate plot

Script 1: Finding an estimate of the PSD function

```
%Power Spectral Density estimation
load('wave');

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

plot(F,Pxx));
xlim([0 2]);
xlabel('rad/s'); ylabel('s/rad');
grid on;
```

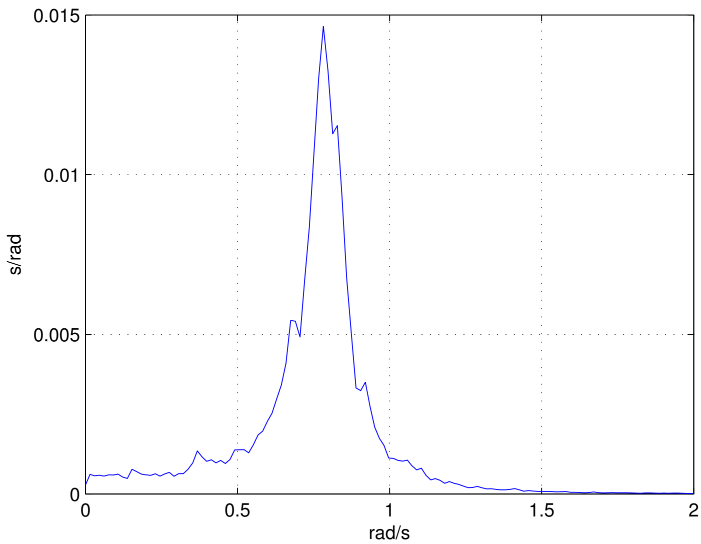

Figure 6: Resulting power spectral estimate based on data in wave.mat

2.2 Analytical expression for the transfer function of the wave response

The analytical expression of $P_{\psi_\omega}(\omega)$ is definde as spectral function:

$$P_{\psi_\omega}(j\omega) = |H(s)_{\frac{\psi_\omega}{\omega_\omega}}|^2 P_{\omega_\omega}(j\omega) \tag{8}$$

To find PSD, we have to find first a transfer function from ψ to ω and the PSD function for the white noise.

First Laplace transforming (1b)

$$\mathcal{L}\left\{\dot{\psi_\omega}\right\} = \mathcal{L}\left\{-\omega_0^2\xi_\omega - 2\lambda\omega_0\psi_\omega + K_\omega\omega_\omega\right\}$$

And substitute equation (1a) in to (1b)

$$\begin{aligned} s\psi_\omega &= -\omega_0^2\frac{\psi_\omega}{s} - 2\lambda\omega_0\psi_\omega + K_\omega\omega_\omega \\ \frac{\psi_\omega}{\omega_\omega} &= \frac{sK_\omega}{s^2 + 2\lambda\omega_0 s + \omega_0^2} \end{aligned}$$

The transfer function becomes

$$H(s)_{\frac{\psi_\omega}{\omega_\omega}} = \frac{sK_\omega}{s^2 + 2\lambda\omega_0 s + \omega_0^2} \tag{9}$$

Equation 9 is the shaping filter for the white noise input. To find the analytical power spectral function can be expressed as in equation 10. As the input is unity white noise, the power spectral function for the input S_f is 1.

$$P_{\psi_\omega} = S_f H(s)_{\frac{\psi_\omega}{\omega_\omega}} H(-s)_{\frac{\psi_\omega}{\omega_\omega}} \tag{10}$$

Inserting equation 9 into equation 10 yields

$$\begin{aligned} P_{\psi_\omega} &= 1\frac{j\omega K_\omega \cdot -j\omega K_\omega}{(-\omega^2 + j2\lambda\omega_0\omega + \omega_0^2)\cdot(-\omega^2 - j2\lambda\omega_0\omega + \omega_0^2)} = \\ &\frac{\omega^2 K_\omega^2}{\omega^4 + \cancel{j2\lambda\omega_0\omega^3} - \omega^2\omega_0^2 - \cancel{j2\lambda\omega_0\omega^3} + 4\lambda^2\omega_0^2\omega^2 + \cancel{j2\lambda\omega_0^3\omega} - \omega_0^2\omega^2 - \cancel{j2\lambda\omega_0^3\omega} + \omega_0^4} \\ &= \frac{\omega^2 K_\omega^2}{\omega^4 - 2\omega_0^2\omega^2 + 4\lambda^2\omega_0^2\omega^2 + \omega_0^4} = \frac{\omega^2 K_\omega^2}{\omega^4 + (4\lambda^2 - 2)\omega_0^2\omega^2 + \omega_0^4} \end{aligned}$$

The analytical expression for P_{ψ_ω} becommes

$$P_{\psi_\omega}(\omega) = \frac{\omega^2 K_\omega^2}{\omega^4 + (4\lambda^2 - 2)\omega_0^2\omega^2 + \omega_0^4} \tag{11}$$

2.3 Finding ω_0

After we have establishing S_{ϕ_ω} in MATLAB, we can extract ω_0 by using script 2. The frequency with the largest amplitude in the plot of the power spectral function is ω_0.

Script 2: Finding ω_0

```
[ maxValue , i] = max(pxx);
maxfreq = f(i);
w0 = maxfreq *2* pi
```

We get that $\omega_0 = 0.7823$

2.4 Identify the damping factor λ

K_ω is now defined as $2\lambda\omega_0\sigma$ where, σ^2 is found by determining the maximum value in the power spectral density results. Now the only unknown value is λ which can be determined by the function `lsqcurvefit`, which is a curve fitting function. The λ making P_{ψ_ω} fit best according to `lsqcurvefit` is $\lambda = 0.0827$. Matlab script used for this task is found in script 3 while the comparison of the estimated S_{ψ_ω} and P_{ψ_ω} is found in figure 7

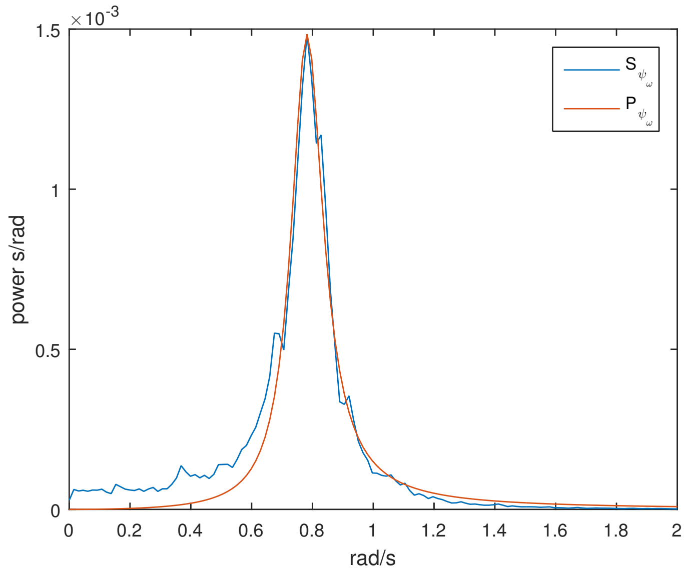

Figure 7: Comparison of data in wave.mat and fitted P_{ψ_ω}

Script 3: Finding the damping factor λ and comparing plot

```
%Determining w0 and sigma squared
[sigmasquared,pos]=max(Pxx);
w0=F(pos);
s=sqrt(sigmasquared);

%Curvefitting to find lambda
f = ...
    @(L,X)(X*2*L*w0*s).^2./(X.^4-X.^2*(2*w0^2-4*x^2*w0^2)+w0^4);
lambdaguess=1;
L=lsqcurvefit(f,lambdaguess,F,Pxx);

%Creating data for plotting of P
P=(F.*2*L*w0*s).^2./(F.^4-F.^2*(2*w0^2-4*L^2*w0^2)+w0^4);

plot(F,Pxx,F,P)
xlim([0 2]); xlabel('rad/s'); ylabel('power s/rad');
```

3 Control system design

The PD controller is given as, $H_{pd}(s) = K_{pd}\frac{1+T_d s}{1+T_f s}$. The controller is to be designed according to the following requirements: $\omega_c = 0.1$ (rad/s) and with a phase angel of 50 degrees. T_d is to be chosen so that it cancels out the time constant of the ship transfer function. The open loop system $H_{pd}(s) \cdot H_{ship}(s)$ is given in equation 12

$$H_{pd}(s) \cdot H_{ship}(s) = K_{pd}\frac{1+T_d s}{1+T_f s} \cdot \frac{K}{Ts^2+s} = \frac{K_{pd}K(1+T_d s)}{s(1+T_f s)(1+Ts)} \tag{12}$$

Setting $T_d = T$ result in the open loop transfer function given in equation 13

$$H_{OL}(s) = \frac{K_{pd}K}{s(1+T_f s)} \tag{13}$$

Based on the requirements we can set up the following equations 14 and 15

$$|H_{OL}| = \frac{K_{pd}K}{\sqrt{T_f^2\omega_c^4 + \omega_c^2}} = 1 \tag{14}$$

$$\angle H_{OL} = \tan^{-1}(\frac{\omega_c}{T_f\omega_c^2}) - (-\pi) = \frac{50 * \pi}{180} \tag{15}$$

Solving equation 15 for T_f gives $T_f = 8.3910$: Then solving equation 14 for K_{pd} gives $K_{pd} = 0.7481$. This was done in Matlab with script 4. The bode plot of the transfer function with the controller designed according to specifications are given in figure 8.

Script 4: Solving equation (14) and(15) and bode plot diagram

```
T=86.89;
Td=T;
K=0.1745;
wc=0.1;
W=50;
Tf=1/(tan(W*pi/180-pi)*wc)
Kpd=sqrt(Tf^2*wc^4+wc^2)/K
H=tf([K*Kpd],[Tf 1 0])
bode(H)
margin(H)
```

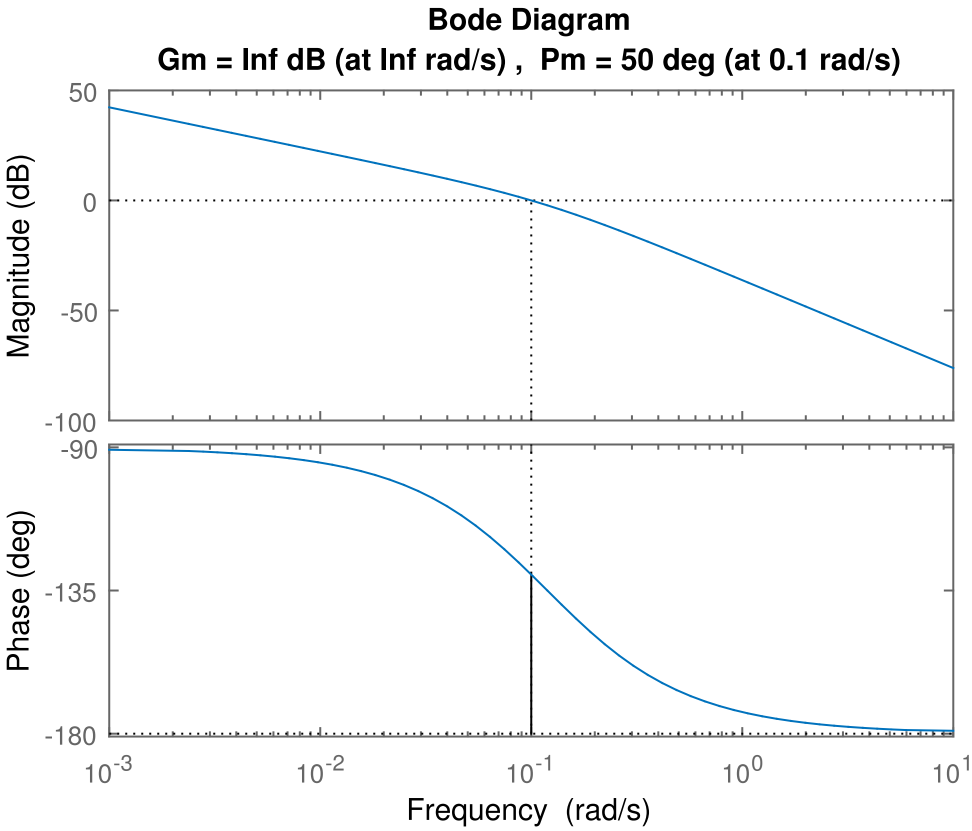

Figure 8: Bode plot of open loop transfer function

3.1 Simulation of system with only measurement noise

The PD controller has been tested with only measurement noise. The reference heading was set to 30°. Start heading is 0°. The simulation result is found in 9. The controller is able to change heading to 30°and remain at the required heading. There are some observable noise on the ruder angle curve which is caused by the measurement noise. This noise will cause ware and tare on the steering gear, however as the PD controller is able to change heading and remain at required heading we would say that the autopilot is working in the case where only measurement noise is present. As the measurement noise is of a much higher frequency than the heading changing frequency, including a low pass filter may be a sensible approach to remove measurement noise in this case.

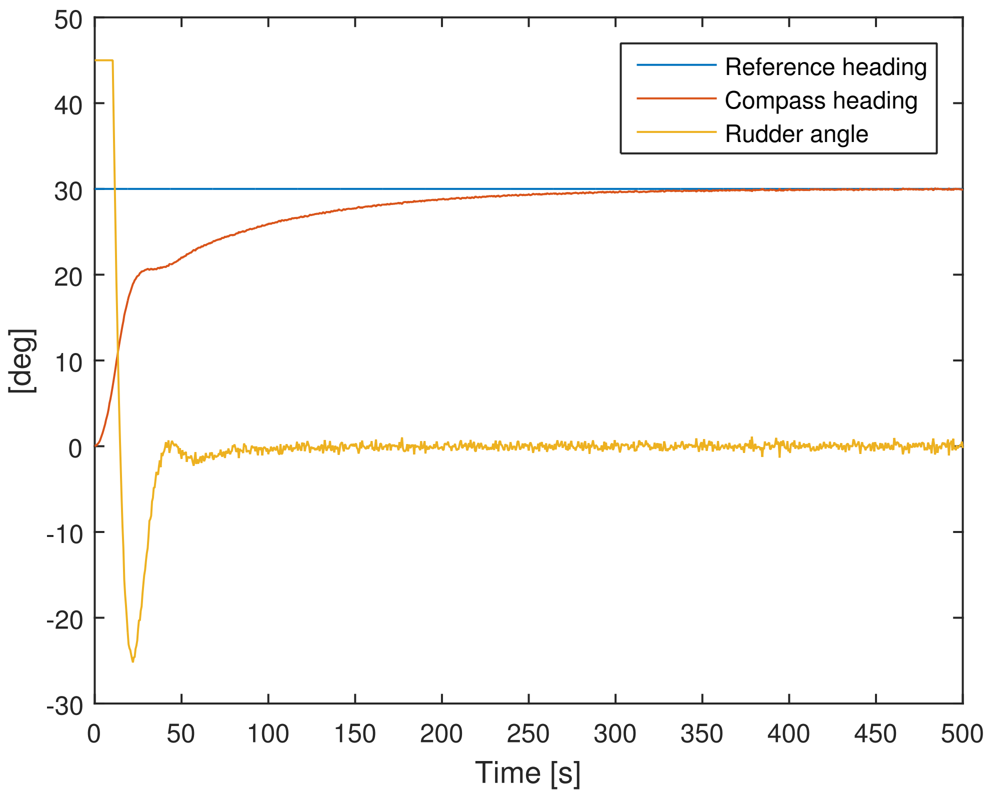

Figure 9: Simulation results for PD controller with measurement noise only

3.2 Simulation of system with measurement noise and current

Results of simulations with measurement noise and current disturbance is found in 10. When current disturbance is included, the PD controller is no longer able to get to the required heading. This problem is similar to the Helicopter lab problem where the LQR without integral effect would drift due to gravity. With current disturbance turned on, the PD controller does not work satisfactorily. As with the helicopter lab, including integral will enable the autopilot to reach the required heading even under current effect.

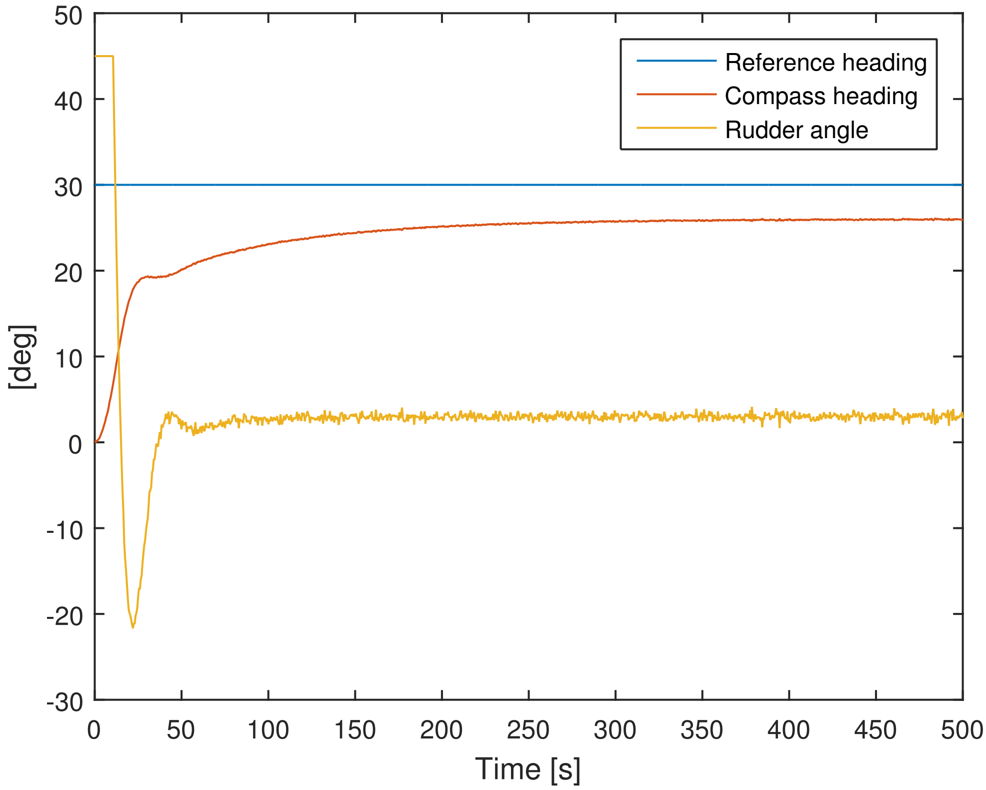

Figure 10: Simulation results for PD controller with measurement noise and current

3.3 Simulation of system with measurement noise and waves

Results of simulations with measurement noise and current disturbance is found in figure 11. When the wave disturbance is included, the PD controller is able to get to the required heading and continue heading in the required direction, although with a some deviations from the required course. However, the steering gear is working serious overtime and the ware and tare is significant. It should also be noted that this kind of rudder operation will significantly increase the drag of the ship, resulting in increased fuel consumption. So even if the autopilot is able to keep the ship heading in the right direction, the autopilot is not working satisfactory due to the amount of rudder input used.

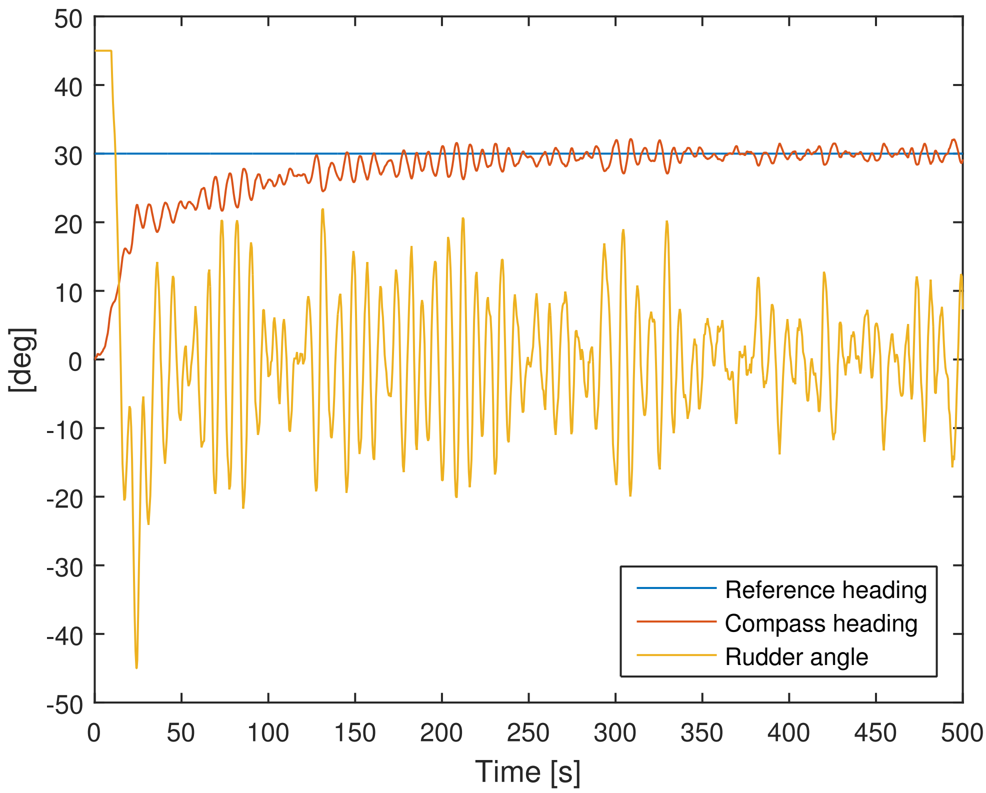

Figure 11: Simulation results for PD controller with measurement noise and waves

3.4 Suggestion for improvements

As the effect of waves and measurement noise on the rudder controller is of a high frequency compared to the frequency of heading change, one possible approach to improve the controller is to include a low pass filter on the measured compass heading.

To compensate for the bias including integral effect would be one method to overcome the effect of the current.

In figure 12 both a low pass filter and integral effect have been implemented to see if there are any potential for improvements. The result show that the low pass filter is able to remove a significant portion of the high frequency wave effect and that the autopilot is able to achieve required heading with integral effect. However the introduction of the low pass filter and integral effect has made the controller underdamped. More tuning is required to achieve critical damped course changes.

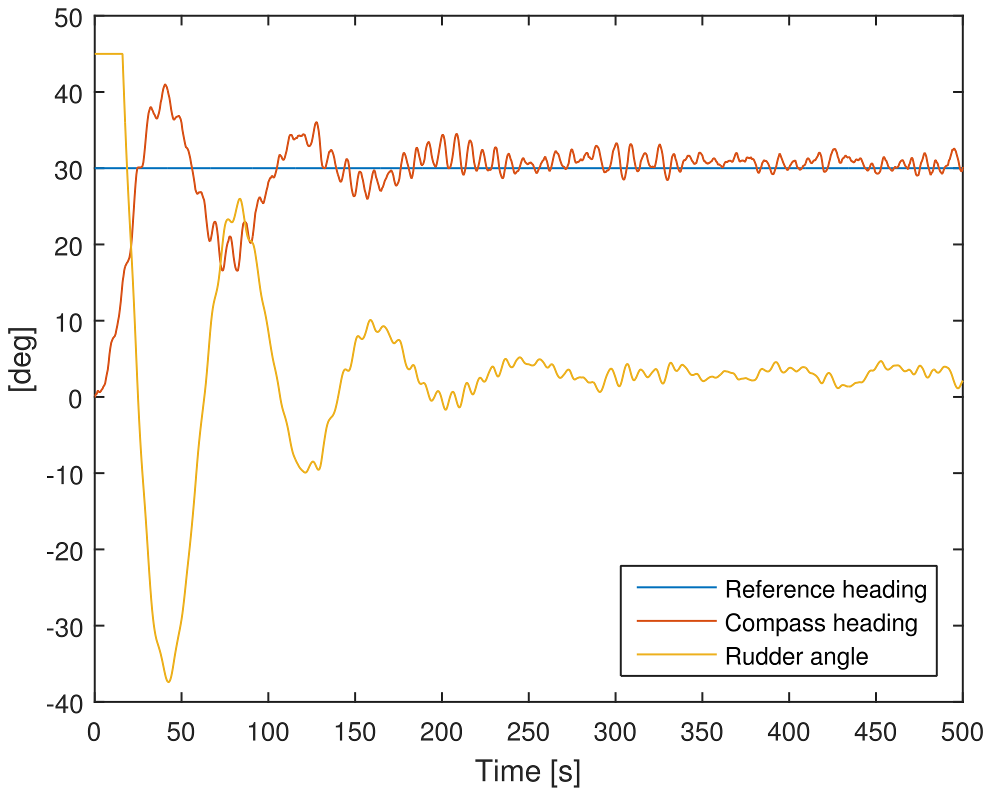

Figure 12: Autopilot operation with a low pass filter and integral effect

4 Observability

4.1 System formulation

Writing the system on the form

$$\dot{x} = Ax + Bu + Ew, \;\; y = Cx + v \tag{16}$$

For a given vector $x = [\xi_\omega \; \psi_\omega \; \psi \; r \; b]^T$, the input vector $u = [\delta]$ and the disturbance vector $w = [\omega \; \omega_b]^T$, where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -\omega_0^2 & -2\omega_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{17}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \\ 0 \end{bmatrix} \tag{18}$$

$$E = \begin{bmatrix} 0 & 0 \\ K_\omega & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{19}$$

and

$$C = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{20}$$

4.2 Obsevability

To find the observability of the different cases, the state space have been modified accordingly. Without any disturbances, only the heading dynamics is left etc. The observability of a system can be determined by finding the rank of the observability-matrix as shown in equation 21

$$rank(\mathcal{O}) = ran(\begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ . \\ CA^{m-1} \end{bmatrix}) \tag{21}$$

The rank of observability-matrix have to be equal to number of state m in the system, then we get a observable system. To evaluate if the system is observable we us MATLAB code

```
Obs=obsv(A,C);
unob = length(A)-rank(Obs)
```

If *unob* evaluates to 0, we have an observable system.

4.3 Obsevability without disturbances

In order to test the observability of the system without disturbances, the system can be simplified to a great extent. This way only equations 1c and 1d are used to derive the A and C matrices.

$$A = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{T} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{22}$$

We calculate and find that the system is observable without disturbances.

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} \rightarrow rank(\mathcal{O}) = dim(A) \tag{23}$$

```
%Problem 5.4 b
%-----Preparation of system matrices for system without ...
    disturbances----
Ab=zeros(2,2);
Bb=zeros(2,1);
Cb=zeros(1,2);

%Adding values to the system matrices
Ab(1,2)=1;
Ab(2,2)=-1/T;

Bb(2,1)=K/T;

Cb(1,1)=1;

%Observability
Obs_b=obsv(Ab,Cb);
unob_b = length(Ab)-rank(Obs_b)
```

4.4 Obsevability with current disturbance

To test the observability of the system with current disturbance equations 1c, 1d and 1e are used to derive the A and C matrices.

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{1}{T} & -\frac{K}{T} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{24}$$

After calculation the observability equation, we get that the system is observable with current disturbances.

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} \rightarrow rank(\mathcal{O}) = dim(A) \tag{25}$$

```
%Problem 5.4 c
%------Preparation of system matrices with current ...
    disturbance----------
Ac=zeros(3,3);
Bc=zeros(3,1);
Cc=zeros(1,3);
Ec=zeros(3,1);

%Adding values to the full system matrices
Ac(1,2)=1;
Ac(2,2)=-1/T;
Ac(2,3)=-K/T;

Bc(2,1)=K/T;

Cc(1,1)=1;

Ec(3,1)=1;

%Observability
Obs_c=obsv(Ac,Cc);
unob_c = length(Ac)-rank(Obs_c)
```

4.5 Obsevability with the wave disturbance

To test the observability of the system with wave disturbance equations (1a), (1b), (1c)and (1d)are used to derive the A and C matrices

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \omega_0^2 & -2\lambda\omega_0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{T} \end{bmatrix}, C = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \tag{26}$$

Again by calculate with the observability equation, we get a observable system with the wave disturbance.

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ CA^3 \end{bmatrix} \rightarrow rank(\mathcal{O}) = dim(A) \tag{27}$$

```
%Problem 5.4 d
%------Preparation of system matrices with wave ...
    disturbance----------
Ad=zeros(4,4);
Bd=zeros(4,1);
Cd=zeros(1,4);
Ed=zeros(4,1);

%Adding values to the full system matrices
Ad(1,2)=1;
Ad(2,1)=-w0^2;
Ad(2,2)=-2*lambda*w0;
Ad(3,4)=1;
Ad(4,4)=-1/T;

Bd(3,1)=K/T;

Cd(1,2)=1;
Cd(1,3)=1;

Ed(2,1)=Kw;

%Observability
Obs_d=obsv(Ad,Cd);
unob_d = length(Ad)-rank(Obs_d)
```

4.6 Obsevability with both current and wave disturbance

From matrix (17) and matrix(20) we substituting the matrixs into (21) and get

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ CA^3 \\ CA^4 \end{bmatrix} \rightarrow rank(\mathcal{O}) = dim(A) \tag{28}$$

The system is again observable. Only when the system are observable the system can be realized trough a Kalman filter. Therefor is importance of having an observable system, for realized trough a such filter.

```
%Problem 5.4 e
%------Preparation of system matrices for the full ...
    system----------
Ae=zeros(5,5);
Be=zeros(5,1);
Ce=zeros(1,5);
Ee=zeros(5,2);

%Adding values to the full system matrices
Ae(1,2)=1;
Ae(2,1)=-w0^2;
Ae(2,2)=-2*lambda*w0;
Ae(3,4)=1;
Ae(4,4)=-1/T;
Ae(4,5)=-K/T;

Be(4,1)=K/T;

Ce(1,2)=1;
Ce(1,3)=1;

Ee(2,1)=Kw;
Ee(5,2)=1;

%Observability
Obs_e=obsv(Ae,Ce);
unob_e = length(Ae)-rank(Obs_e)
```

5 Discrete Kalman filter

5.1 Discretization of continuous state space model

Following equations is use to discretization of the model

$$A_d = e^{AT} = \mathcal{L}^{-1}\left\{(sI - A)^{-1}\right\}_{t=T} \tag{29a}$$
$$B_d = (\int_{\tau=0}^{T} e^{A\tau} d\tau)B \tag{29b}$$
$$C_d = C \tag{29c}$$
$$E_d = (\int_{\tau=0}^{T} e^{A\tau} d\tau)E \tag{29d}$$

We get that T=1/fs, and our samling frequency $f_s = 10Hz$.The continous state space model was discretized by the Matlab command `c2d` as shown in script 5. With the method option `zoh`, zero order hold, the c2d command gives the exact discretization with the assumption that the control input are piecewise constant in the time intervals between the discrete time steps.

Script 5: Discretization of the model matrices

```
%Discrete system
%Matlab c2d method
opt = c2dOptions('Method', 'zoh');

sys1=ss(A,B,C,D);
sys1d=c2d(sys1,dt,opt);

sys2=ss(A,E,C,D);
sys2d=c2d(sys2,dt,opt);

Ad=sys1d.a;
Bd=sys1d.b;
Ed=sys2d.b;
Cd=C;
Dd=0;
```

The resulting matrices are given in equation 30, 31, 32 and 33.

$$A_d = \begin{bmatrix} 0.9970 & 0.0993 & 0 & 0 & 0 \\ -0.0607 & 0.9841 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.0999 & -0.0000 \\ 0 & 0 & 0 & 0.9988 & -0.0002 \\ 0 & 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{30}$$

$$B_d = \begin{bmatrix} 0 \\ 0 \\ 0.0101e-3 \\ 0.2013e-3 \\ 0 \end{bmatrix} \tag{31}$$

$$E_d = \begin{bmatrix} 0.0000 & 0 \\ 0.0005 & 0 \\ 0 & -0.0000 \\ 0 & -0.0000 \\ 0 & 0.1000 \end{bmatrix} \tag{32}$$

$$C_d = C \tag{33}$$

5.2 Finding an estimate for the measurement noise variance

To find the measurement noise variance the ship model was run without any rudder input and with only measurement noise activated. Compass heading was logged. The compass heading data now contains all the data we need to estimate the measurement noise variance. The test was run for a simulation time of 3600 seconds. For improved variance estimate, we could have collected data from several tests. Script 6was used to estimate the measurement variance. The resulting variance became $\sigma^2 = 6.0167e-07$

Script 6: Finding an estimate of the PSD function

```
load('TFdata');
V=var(ans(2,:)*pi/180)
```

5.3 Kalman filter implementation

We have from assignment that

$$P_0^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.013 & 0 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-4} \end{bmatrix} \tag{34}$$

$$\hat{x}_0^- = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{35}$$

$$w = \begin{bmatrix} \omega_\omega & \omega_b \end{bmatrix}^T \tag{36}$$

$$E\{ww^T\} = Q = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix} \tag{37}$$

The Kalman filter was implementation by using S-function block in Simulink. The block get input from the compass measurement y and the rudder input δ. The S-function calculate estimate compass measurement ψ, estimate current disturbance b and estimate wave disturbance ψ_ω. In figure 13 show how the implement of the Kalman filter loop work

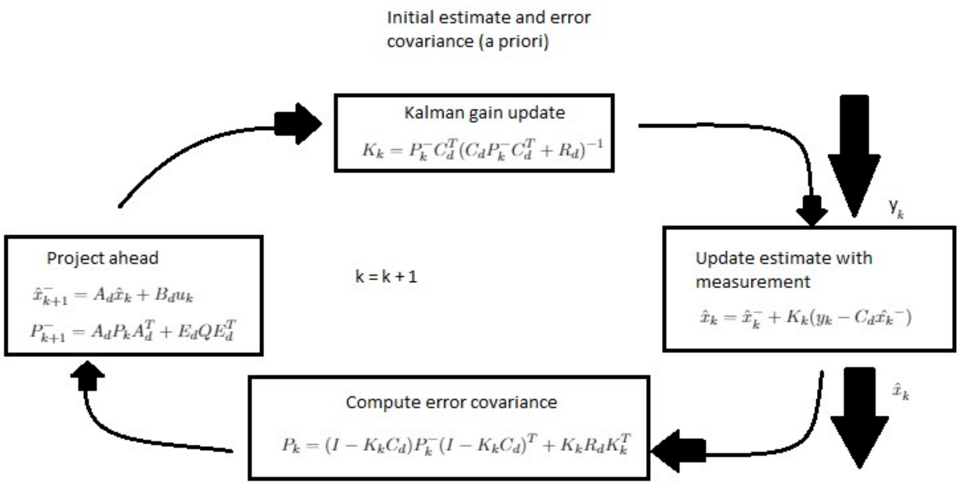

Figure 13: The Kalman filter loop

The algorithm of the filter have equation that are calculated in the following sequence

$$K_k = P_k^- C_d^T (C_d P_k^- C_d^T + R_d)^{-1} \tag{38}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C_d\hat{x}_k{}^-) \tag{39}$$

$$P_k = (I - K_k C_d)P_k^-(I - K_k C_d)^T + K_k R_d K_k^T \tag{40}$$

$$P_{k+1}^- = A_d P_k A_d^T + E_d Q E_d^T \tag{41}$$

$$\hat{x}_{k+1}^- = A_d\hat{x}_k + B_d u_k \tag{42}$$

The model and the script for the S-function are in appendix A.4

5.4 Bias feed forward

To compensate for the current influence on the heading, a bias feed forward was implemented into the controller. The estimated bias from the Kalman filter was used as feed forward input.

Resulting autopilot operation is presented in figure 14. The estimated bias is presented in figure 15

With the estimated bias as feed forward, the autopilot is able to achieve and hold the reference heading.

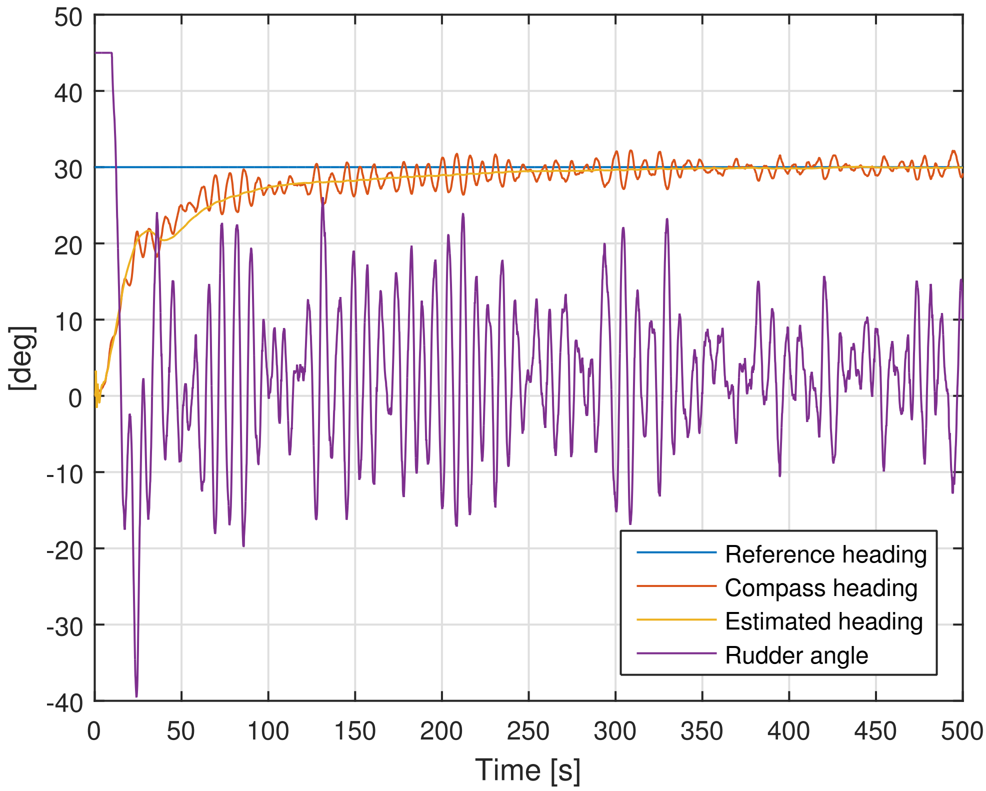

Figure 14: Autopilot performance with estimated bias feed forward

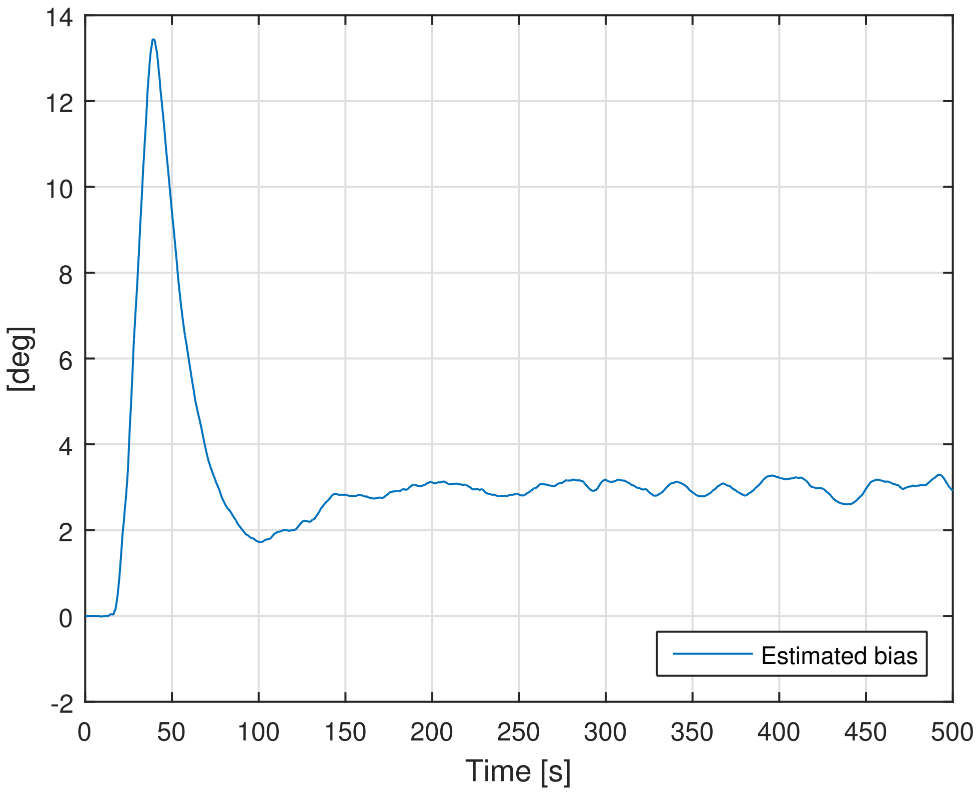

Figure 15: Estimated bias caused by current

5.5 Wave filtered heading feedback

To remove the high frequency input to the rudder controller, the wave filtered heading was used as input to the autopilot. Resulting autopilot performance is presented in figure 16. The estimated bias is presented in figure 17, while a comparison of estimated wave influence on heading and actual effect is presented in figure 18

With both bias feed forward and wave filtered heading as input to the rudder controller, the high frequency rudder action is eliminated. The autopilot is able to reach and maintain the designated heading with only minor oscillations.

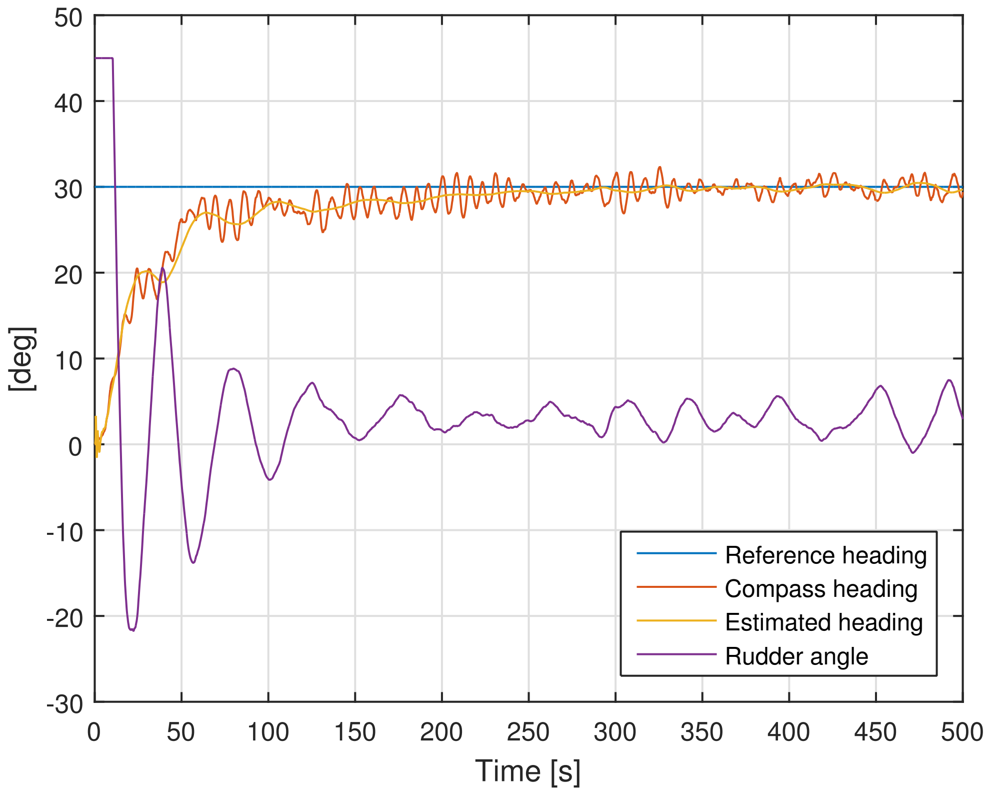

Figure 16: Autopilot performance with wave filtered heading as input

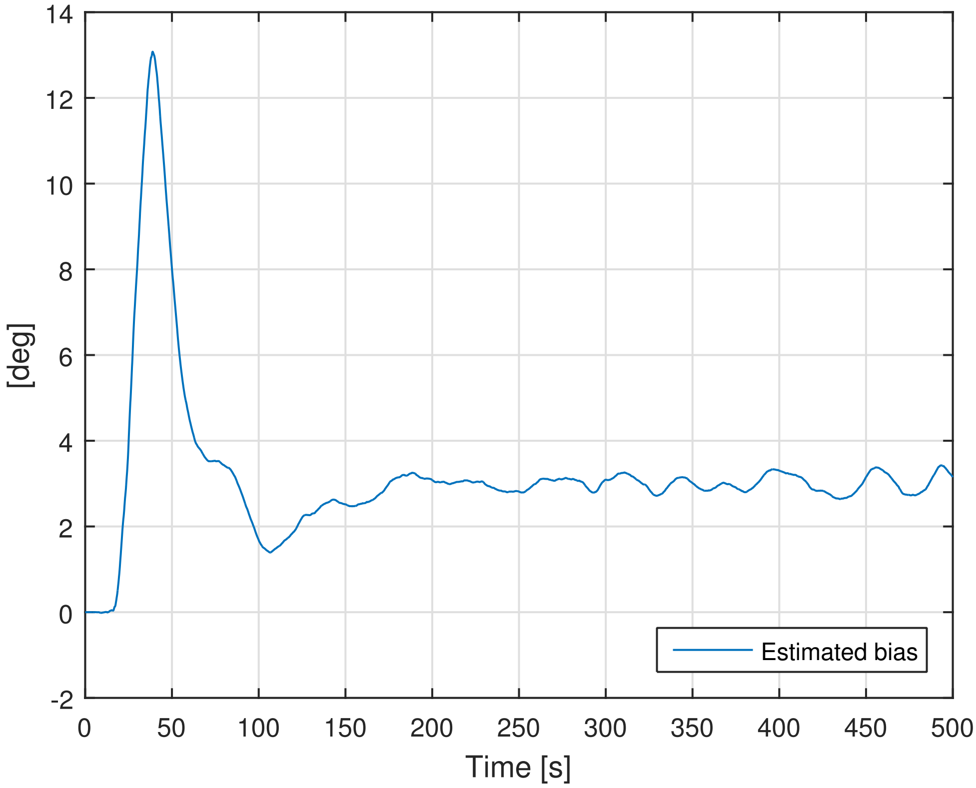

Figure 17: Estimated bias caused by current

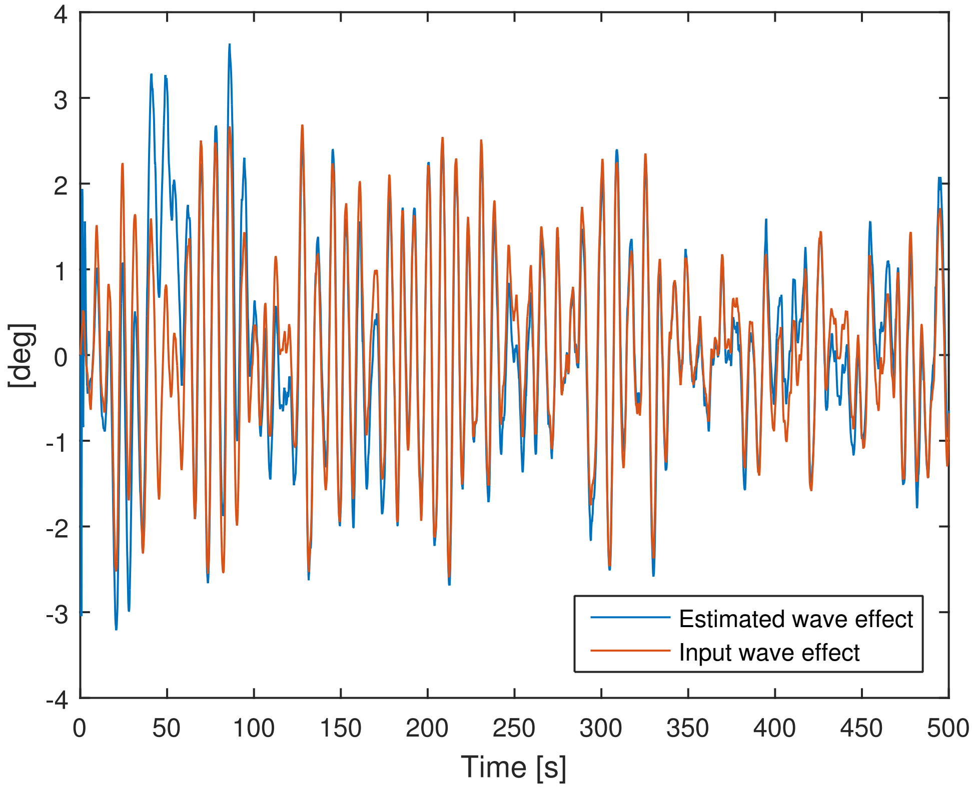

Figure 18: Comparison of estimated wave heading influence and actual wave heading influence

6 Conclusion

The purpose of this this assignment has been to implement an autopilot to control the ship compass course. The control system has been tested with various cases of wave and current influence combined with measurement noise.

With direct feedback from the compass heading measurement, the autopilot performance was unsatisfactory. The autopilot was trying to compensate for all effects of waves, and was not able to correct for the effect of current. Measurement noise was also applied directly as an input to the rudder.

Some mitigation was possible by introducing a low pass filter to remove the high frequency fractions of the compass heading signal and integral effect to compensate for the effect of current. However these additions did also affect the performance of the controller itself, requiring retuning of the autopilot.

To mitigate the short comings of the autopilot when working with stochastic signals, a Kalman filter was implemented. The Kalman filter allow us to filter the effect of waves, current and measurement noise based on their known statistical data. By using the predicted effect of current and heading without effect of waves the autopilot performance improved significantly without any need for redesign or retuning of the autopilot.

Appendices

A Simulink diagrams and m-files

A.1 Assignment 5.1

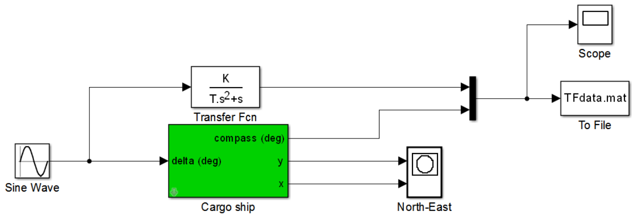

Figure 19: Simulink diagram used in assignment 5.1

A.2 Assignment 5.2

Script 7: Script for determining ω_0, σ^2 and λ

```
load('wave');

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

[sigmasquared,pos]=max(Pxx);
w0=F(pos);
sigma=sqrt(sigmasquared);

lambdaguess=1;
f = ...
    @(x,xdata)(xdata*2*x*w0*sigma).^2./(xdata.^4-xdata.^2*(2*w0^2-4*x^2*w0^2)+w0^4);

lambda=lsqcurvefit(f,lambdaguess,F,Pxx);


P=(F.*2*lambda*w0*sigma).^2./(F.^4-F.^2*(2*w0^2-4*lambda^2*w0^2)+w0^4);

limits=[0 2];

plot(F,Pxx,F,P)
xlim(limits)
xlabel('rad/s')
ylabel('power s/rad')
legend('S_{\psi_{\omega}}','P_{\psi_{\omega}}')
```

A.3 Assignment 5.3

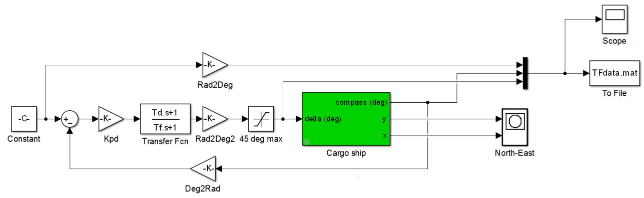

Figure 20: Simulink diagram used in assignment 5.3

Script 8: Script for determining controller parameters

```
A1=31.9795;
A2=0.785;
F1=0.005;
F2=0.05;

T=sqrt((A1^2*F1^2-A2^2*F2^2)/(A2^2*F2^4-A1^2*F1^4));
K=A1*sqrt(T^2*F1^4+F1^2);

Td=T;
wc=0.1;
W=50;
Tf=1/(tan(W*pi/180-pi)*wc)
Kpd=sqrt(Tf^2*wc^4+wc^2)/K
H=tf([K*Kpd],[Tf 1 0])
bode(H)
margin(H)
```

A.4 Assignment 5.5

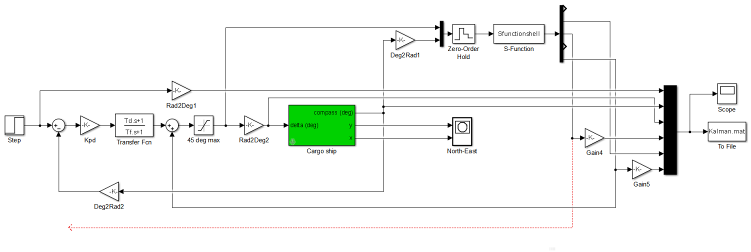

Figure 21: Simulink diagram used in assignment 5.5

Script 9: Initialation script for the Kalman filter

```
clear all
clc

%System description

dt=0.1; %Discretization time step

%Measured amplitudes for determination of constants T and K
A1=31.9795;      %Measured
A2=0.7850;       %Measured

%Frequency of sine used to get data for A1 and A2
F1=0.005;    %Experiment input
F2=0.05;     %Experiment input

%Calculation of ship model T and K
T=sqrt((A1^2*F1^2-A2^2*F2^2)/(A2^2*F2^4-A1^2*F1^4));
K=A1*sqrt(T^2*F1^4+F1^2);

%Shaping filter for waves
w0=0.7823;                     %Measured
lambda=0.0827;                 %Fitted
sigma=0.0385;                  %Measured
Kw=2*lambda*w0*sigma;          %Formula given

%Preparation of system matrices for the full system
A=zeros(5,5);
B=zeros(5,1);
C=zeros(1,5);
E=zeros(5,2);
D=0;

```

```matlab
%Adding values to the full system matrices
A(1,2)=1;
A(2,1)=-w0^2;
A(2,2)=-2*lambda*w0;
A(3,4)=1;
A(4,4)=-1/T;
A(4,5)=-K/T;

B(4,1)=K/T;

C(1,2)=1;
C(1,3)=1;

E(2,1)=Kw;
E(5,2)=1;


%Controller design
wc=0.1; %Given
Tf=1/(tan(50*pi/180-pi)*wc); %Based on phase shift equation
Kpd=sqrt(Tf^2*wc^4+wc^2)/K;
Td=T;

%Discrete system
%Matlab c2d method
opt = c2dOptions('Method', 'zoh');

sys1=ss(A,B,C,D);
sys1d=c2d(sys1,dt,opt);

sys2=ss(A,E,C,D);
sys2d=c2d(sys2,dt,opt);

Ad=sys1d.a;
Bd=sys1d.b;
Ed=sys2d.b;
Cd=C;
Dd=0;

%Determining Q and R
Q=[30,0;0,10^-6];

sigmasquared=6.0129e-07;
R=sigmasquared/dt;

%P0 matrix
P0=zeros(5,5);
P0(1,1)=1;
P0(2,2)=0.013;
P0(3,3)=pi^2;
P0(4,4)=1;
P0(5,5)=2.5*10^(-4);

%Initial values
```

```
x0=zeros(5,1);

%Making struct for input to s-function
data=struct('A',Ad(:)','B',Bd(:)','C',Cd,'E',Ed(:)','Q',...
Q(:)','R',R,'P0',P0(:)','x0',x0');
```

Script 10: S-function with implementation of Kalman filter according to method 2

```
function [sys,x0,str,ts] = DiscKal(t,x,u,flag,data) %if ... method 2 is used
% Shell for the discrete kalman filter assignment in
% TTK4115 Linear Systems.
%
% Author: JÃ¸rgen SpjÃ¸tvold
% 19/10-2003
%

switch flag,

  %%%%%%%%%%%%%%%%%%
  % Initialization %
  %%%%%%%%%%%%%%%%%%
  case 0,
    [sys,x0,str,ts]=mdlInitializeSizes(data); %if method 2 ... is used

  %%%%%%%%%%%%%
  % Outputs   %
  %%%%%%%%%%%%%

  case 3,
    sys=mdlOutputs(t,x,u,data);
  %%%%%%%%%%%%%
  % Terminate %
  %%%%%%%%%%%%%

  case 2,
    sys=mdlUpdate(t,x,u, data);

  case {1,4,}
    sys=[];

  case 9,
      sys=mdlTerminate(t,x,u);
  %%%%%%%%%%%%%%%%%%%%
  % Unexpected flags %
  %%%%%%%%%%%%%%%%%%%%
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end
```

```matlab

function [sys,x0,str,ts]=mdlInitializeSizes(data) %if ...
    method 2 is used
% This is called only at the start of the simulation.

sizes = simsizes; % do not modify

sizes.NumContStates  = 0; % Number of continuous states in ...
    the system, do not modify
sizes.NumDiscStates  = 35; % Number of discrete states in ...
    the system, modify.
sizes.NumOutputs     = 5; % Number of outputs, the hint ...
    states 2
sizes.NumInputs      = 2; % Number of inputs, the hint ...
    states 2
sizes.DirFeedthrough = 0; % 1 if the input is needed ...
    directly in the
% update part
sizes.NumSampleTimes = 1; % Do not modify

sys = simsizes(sizes); % Do not modify

x0  = [data.x0,data.x0,data.P0]; % Initial values for the ...
    discrete states, modify

str = []; % Do not modify

ts  = [-1 0]; % Sample time. [-1 0] means that sampling is
% inherited from the driving block and that it changes during
% minor steps.


function sys=mdlUpdate(t,x,u, data) %if method 2 is used
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Update the filter covariance matrix and state etsimates here.
% example: sys=x+u(1), means that the state vector after
% the update equals the previous state vector + input nr one.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A=reshape(data.A,sqrt(length(data.A)),sqrt(length(data.A)));
B=reshape(data.B,5,1);
H=data.C;
E=reshape(data.E,5,2);
Q=reshape(data.Q,sqrt(length(data.Q)),sqrt(length(data.Q)));
R=data.R;
P=reshape(x(11:35),5,5);
x_pre=x(1:5);

%Compute Kalman gain
K=P*H'/(H*P*H'+R);

%Update estimate with measurement
x_est=x_pre+K*(u(2)-H*x_pre);

%Update error covariance
```

```
P_upd=(eye(5)-K*H)*P*(eye(5)-K*H)'+K*R*K';

%Projecting ahead
x_new=A*x_est+B*u(1);
P_new=A*P_upd*A'+E*Q*E';


%Make vectors and matrices into array
sys=cat(1,x_new,x_est,P_new(:));

function sys=mdlOutputs(t,x,u,data) %if mathod 2 is used
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the outputs here
% example: sys=x(1)+u(2), means that the output is the ...
    first state+
% the second input.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Make vectors and matrices
%Calculate estimated heading and rudder bias

%Make vectors and matrices into array
sys=(eye(5)*x(6:10))';

function sys=mdlTerminate(t,x,u)
sys = [];
```