TTK4190 Guidance and Control of Vehicles

Assignment 3

Written Fall 2020 By Magnus Dyre-Moe, Patrick Nitschke and Siawash Naqibi

1 Marine Craft Modeling

1.1 Problem a)

Let $m = \rho_m LBT$ with $L = 161$ m, $B = 21.8$ m and $T = 15.8$ m being the dimensions of the prism, respectively; length, width and height. ρ_m is not defined. However, given that the mass of the ship and the prism is the same, we have $m = 17.0677 \cdot 10^6$. With this, we can compute the density of the prism, which turns out to be $\rho_m = 307.7766$ kg/m^3.

Next, moment of inertia around yaw in center og gravity (CG) is given by Definition 3.1 in Fossen on page 55 and states:

$$I_z^{CG} = \int_V (x^2 + y^2)\rho_m dV \tag{1}$$

Given that we have a prism, the volume is defined as:

$$V = L \cdot B \cdot T \tag{2}$$

Resulting in

$$I_z^{CG} = \frac{m}{12} \cdot (L^2 + B^2) \tag{3}$$

When integrating w.r.t to the center of volume, meaning from $-\frac{L}{2}$ to $\frac{L}{2}$, etc.

Inserting with values, we arrive at $I_z^{CG} = 3.7544 \cdot 10^{10}$.

Given that the center of origin (CO) is located at the center of volume we know that CO is located at the center of the boat. We know this given the structure of the prism and a constant density. Knowing this, the prism is symmetric around the x and y axis. Hence, $I_{xy}^{CG} = 0$.

1.2 Problem b)

Transforming the inertia dyadic to the point CO can be done through the use of the Parallel-Axis Theorem, Eq. (3.36) in Fossen:

$$I_b^b = I_g^b + m \cdot ((r_g^b)^\top r_g^b I^3 - r_g^b (r_g^b)^\top) \tag{4}$$

Where r_g^b represents the vector from CG to CO, given as:

$$r_g^b = \begin{bmatrix} -3.7 \\ 0 \\ H/2 \end{bmatrix} \tag{5}$$

First, we found the inertia dyadic around CG by also using the method in a) for the x and y-axes. With this, we used the formula above to calculate the new inertia dyadic about CO. This was found to be $I_z^{CO} = 3.777 \cdot 10^{10}$. Taking the ratio of I_z^{CO} of the prism and vessel gives 1.7753, which is, to no surprise, equal to the ratio between the height of the prism and vessel.

1.3 Problem c)

From Fossen Equations 3.23 and 3.24 M_{RB}^{CG} and C_{RB}^{CG} are:

$$\begin{aligned} M_{RB}^{CG} &= \begin{bmatrix} m\mathbf{I}_3 & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \mathbf{I}_g^b \end{bmatrix} \\ C_{RB}^{CG} &= \begin{bmatrix} m\mathbf{S}(\omega_{nb}^b) & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & -\mathbf{S}(\mathbf{I}_g^b\omega_{nb}^b) \end{bmatrix} \end{aligned} \tag{6}$$

Expanding the matrices into center og origin gives us:

$$\begin{aligned} M_{RB}^{CO} &= \begin{bmatrix} m\mathbf{I}_3 & -m\mathbf{S}(\mathbf{r}_{bg}^b) \\ m\mathbf{S}(\mathbf{r}_{bg}^b) & I_g^b - m\mathbf{S}^2(\mathbf{r}_{bg}^b) \end{bmatrix} \\ C_{RB}^{CO} &= \begin{bmatrix} m\mathbf{S}(\omega_{nb}^b) & -m\mathbf{S}(\omega_{nb}^b)\mathbf{S}(\mathbf{r}_{bg}^b) \\ m\mathbf{S}(\mathbf{r}_{bg}^b)\mathbf{S}(\omega_{nb}^b) & -m\mathbf{S}(\mathbf{r}_{bg}^b)\mathbf{S}(\omega_{nb}^b)\mathbf{S}(\mathbf{r}_{bg}^b) - \mathbf{S}(\mathbf{I}_g^b\omega_{nb}^b) \end{bmatrix} \end{aligned} \tag{7}$$

For our problem we are looking at 3 degrees of freedom, namely, surge, sway and yaw. These er DOF 1,2,6. This results in

$$\begin{aligned} M_{RB}^{CO}\{1,2,6\} &= \begin{bmatrix} m & 0 & -mr_y \\ 0 & m & mr_x \\ -mr_y & mr_x & I_z + m(r_x^2 + r_y^2) \end{bmatrix} \\ C_{RB}^{CO}\{1,2,6\} &= \begin{bmatrix} 0 & -\omega_z & -m\omega_z r_x \\ \omega_z & 0 & m\omega_z r_y \\ m\omega_z r_x & -m\omega_z r_y & 0 \end{bmatrix} \end{aligned} \tag{8}$$

1.4 Problem d)

A skew-symmetric matrix is defined as $A^T = -A$. For C_{RB}^{CO} we observe that the diagonal matrices are skew symmetric. Also, by the definition of the skew-symmetric matrix we observe that C_{RB}^{CO} is indeed skew-symmetric.

Skew-symmetric is a desired property as it allows for stability proof using the Lyapunov direct method and the state-space equation for kinetics. Also, a skew-symmetric matrix has the property of $S(a) \cdot a = 0$.

1.5 Problem e)

The skew-symmetric matrix C_{RB}^{CO} also has a property known as: Linear velocity-independent parameterization. This states that the coriolis matrix does not change with ocean currents as ocean currents are modelled with linear velocity, whereas the coriolis matrix is dependent on the vector between CO and CG, r_{bg}^b, and the angular velocities, ω_{nb}^b.

2 Hydrostatics

2.1 Problem a)

The formula for buoyancy force is given by

$$B = \rho g \nabla \tag{9}$$

where ∇ is volume displacement. For a floating vehicle the buoyancy force is equal to the gravitational force of the vehicle. Hence, we can write the equation as

$$\begin{aligned} W &= B \\ mg &= \rho_w g \nabla \\ m &= \rho_w \nabla \end{aligned} \tag{10}$$

Resulting in $\nabla = \frac{m}{\rho_w} = 1.6651 \cdot 10^4$.

2.2 Problem b)

The area of the water plane, A_{wp} and the hydrostatic force in heave, Z_{hs} are related through

$$Z_{hs} \approx -\rho g A_{wp} z \tag{11}$$

With A_{wp} being the area of the water plane, which in our case is $L \times B$. Resulting in $A_{wp} = 3509.8m^2$. Using $z = T = 8.9$ m and $\rho = 1025$ yields the hydrostatic force in heave $Z_{hs} = -3.14{\cdot}10^8$.

2.3 Problem c)

Using equation 4.78 in Fossen we have

$$T_3 \approx 2 * \pi \sqrt{\frac{2T}{g}} \tag{12}$$

Where T_3 is the period in heave and T can be approximated to be

$$\nabla \approx A_{wp} T \tag{13}$$

This yields, $T \approx \frac{\nabla}{A_{wp}} = 4.7443$. Finally we can calculate the period in heave to be $T_3 = 2 * \pi \sqrt{\frac{2{\cdot}4.7443}{9.81}} = 6.1794s$.

2.4 Problem d)

The transversal and longtitudinal metacenters, GM_T and GM_L can be calculated through:

$$GM_T = BM_T - BG, \qquad GM_L = BM_L - BG \tag{14}$$

where BG represents the distance between CG and CB. BM_T and BM_L denotes the transverse and longitudinal radii of curvature and can be calculated through:

$$BM_T = \frac{I_T}{\nabla}, \qquad BM_L = \frac{I_L}{\nabla} \tag{15}$$

where I_T and I_L is:

$$I_T = \frac{1}{12} B^3 L, \qquad I_L = \frac{1}{12} L^3 B \tag{16}$$

Inserting values yield:

$$I_T = \frac{1}{12} \cdot 21.8^3 \cdot 161 \tag{17}$$

$$= 1.39 \cdot 10^5 \tag{18}$$

$$\tag{19}$$

and,

$$I_L = \frac{1}{12} \cdot 161^3 \cdot 21.8 \tag{20}$$

$$= 7.58 \cdot 10^6 \tag{21}$$

which gives:

$$BM_T = 8.348, \qquad BM_L = 455.23 \tag{22}$$

To calculate the metacenters we need the need the distance the between CG and CO:

$$BG = KG - KB \tag{23}$$

where,

$$KG = \frac{H}{2} \tag{24}$$
$$= 7.9 \tag{25}$$
$$\tag{26}$$
$$KB = \frac{1}{3} \cdot (\frac{5T}{2} - \frac{\nabla}{A_{wp}}) \tag{27}$$
$$= 5.34 \tag{28}$$

which gives $BG = 2.56$. Finally, this gives: $GM_T = 5.79$ and $GM_L = 452.67$.

2.5 Problem e)

The metacenters GM_T and GM_L are stable if they are located above CG. If a metacenter is located below CG the vessel will capsize. Because a boat naturally will be longer than it is wide, the transversal metacenter, GM_T, will be closer to CG than longitudinal metacenter, GM_L, will. From a physical viewpoint this makes sense as a boat is more likely to capsize in a longitudinal motion, rather than i transversal motion. That is, the boat is more likely to flip with a motion along the y-axis.

From the values we achieved from problem d) and definition 4.2 for metacenter stability, we see that the boat is stable for both axes. This being because $GM_T > 0.5$ m and $GM_L > 100$ m.

3 Added Mass and Coriolis

3.1 Problem a)

Using Definition 6.1 in Fossen for surge, sway and yaw, DOF 1,2,6, our added mass matrix will be

$$M_A\{1,2,6\} = -\begin{bmatrix} X_{\dot{u}} & 0 & 0 \\ 0 & Y_{\dot{v}} & Y_{\dot{r}} \\ 0 & N_{\dot{v}} & N_{\dot{r}} \end{bmatrix} \tag{29}$$

3.2 Problem b)

From Property 6.2 in Fossen we can write the coriolis matrix as

$$C_A = \begin{bmatrix} 0 & 0 & a_2 \\ 0 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{30}$$

Where a_1 and a_2 are defined as:

$$\begin{aligned} a_1 &= X_{\dot{u}} u_r \\ a_2 &= Y_{\dot{v}} v_r + Y_{\dot{r}} r \end{aligned} \tag{31}$$

Which is a simplification of the original equations given by (6.47) and (6.48) in Fossen. For a surface vessel we assume heave, pitch and roll negligible because of small motions. Given that we operate with DOF 1,2,6. The resulting coriolis matrix is

$$C_A = \begin{bmatrix} 0 & 0 & Y_{\dot{v}} v_r + Y_{\dot{r}} r \\ 0 & 0 & -X_{\dot{u}} u_r \\ -Y_{\dot{v}} v_r - Y_{\dot{r}} r & X_{\dot{u}} u_r & 0 \end{bmatrix} \tag{32}$$

This is also done in Example 6.1 in Fossen.

In MATLAB the coriolis matrix matrix can be found using $C = m2c(M, \nu)$ for the MSS-Toolbox. Here, M is the mass matrix, which can be rigid-body or added mass, and ν is the generalized velocity expressed in body. C is the coriolis matrix corresponding to the input mass matrix, M.

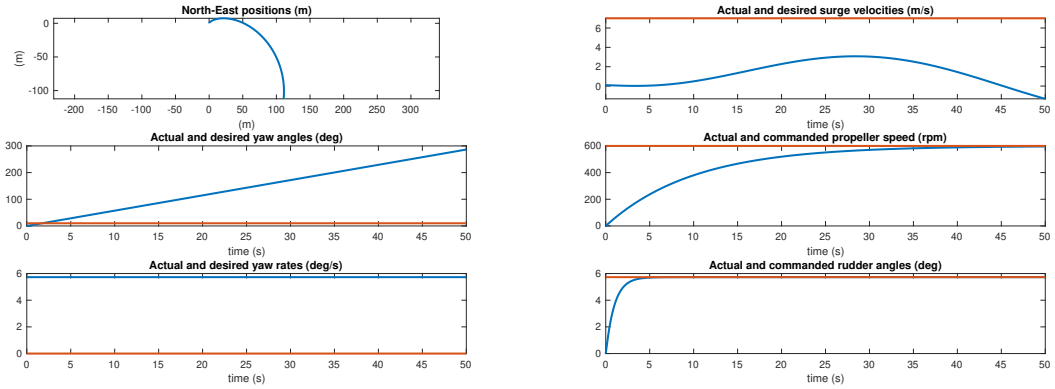

Figure 1: Ship states before added mass.

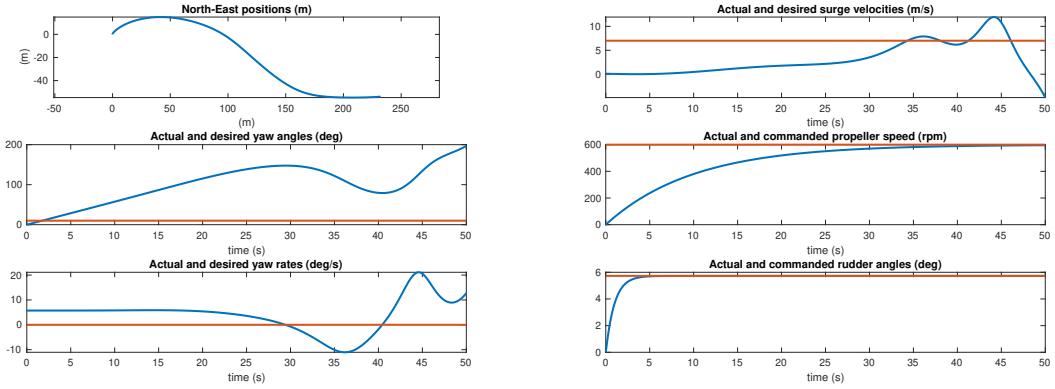

Figure 2: Ship states after added mass.

4 Implementing the Maneuvering Model in MATLAB

4.1 Problem a)

Finding M_A and C_A can be done with function $C = m2c(M, \nu)$ from the MSS-Toolbox.

M_A is time invariant. Hence, it can be placed outside the main loop. C_A is, however, time variant as it is dependant on surge and sway.

4.2 Problem b)

Here, we implement linear and non linear damping. Linear damping is added as

$$D = \begin{bmatrix} -X_u & 0 & 0 \\ 0 & -Y_v & 0 \\ 0 & 0 & -N_r \end{bmatrix} \tag{33}$$

Where the elements in D is as follows

$$\begin{aligned} -X_r &= \frac{m - X_{\dot{u}}}{T_1} \\ -Y_r &= \frac{m - Y_{\dot{v}}}{T_2} \\ -N_r &= \frac{I_z - N_{\dot{r}}}{T_6} \end{aligned} \tag{34}$$

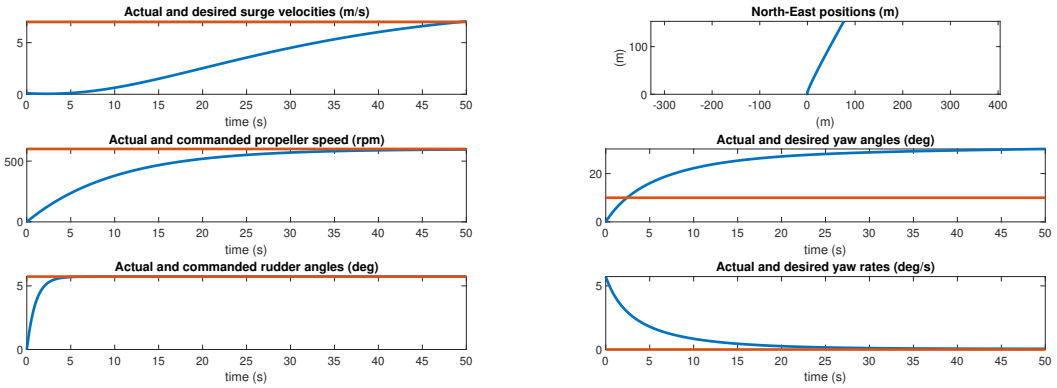

Figure 3: Ship states after added linear and nonlinear damping.

Where m is the mass, $X_{\dot{u}}$, Y_{dotv} and N_{dotr} are previously defined in M_A, I_z is inertia about the z-axis and T_i are time constants defined by the task.

Adding nonlinear surge damping was done through equation (6.76) and (6.77) in Fossen

$$\begin{aligned} X_n &= -\frac{1}{2}\rho S(1+k)C_f(u_r)|u_r|u_r \\ C_f(u_r) &= \frac{0.075}{(\log_{10}(R_n)-2)^2} + C_R \end{aligned} \tag{35}$$

Where k, u_r, C_R and ρ are known. S is the wetted surface area defined as $S = 2TB + 2TL$ and R_n is the Reynolds number, $R_n = u_r \cdot \frac{L}{\mathrm{v}}$, where v is the viscosity of water.

Next, adding cross-flow drag was done through equations (6.78) and (6.79) in Fossen

$$\begin{aligned} Y_n &= -\frac{1}{2}\rho \int_{-\frac{L_{pp}}{2}}^{\frac{L_{pp}}{2}} T(x)C_d^{2D}(x)|v_r + xr|(v_r + xr)dx \\ N_n &= -\frac{1}{2}\rho \int_{-\frac{L_{pp}}{2}}^{\frac{L_{pp}}{2}} T(x)C_d^{2D}(x)x|v_r + xr|(v_r + xr)dx \end{aligned} \tag{36}$$

Where $T(x)$ was assumed constant and C_d^{2D} was calculated with $Hoerner(B,T)$ from the MSS-Toolbox. The remaining variables are known.

Finally, adding damping to the ship dynamics

$$\dot{\nu} = M^{-1}(\tau - C\nu - D\nu + [Xn, Yn, Nn]^T) \tag{37}$$

Where $M = M_{RB} + M_A$, $C = C_{RB} + C_A$, D is linear damping and the vector $[Xn, Yn, Nn]^T$ is nonlinear surge damping and cross-flow drag.

4.3 Problem c)

The simulations with step commands in surge and yaw are seen in Fig 3.

A Matlab code

A.1 Problem 4 - project.m

```
% Project in TTK4190 Guidance and Control of Vehicles
%
% Author:           Magnus Dyre-Moe, Patrick Nitschke and Siawash Naqibi
% Study program:    Cybernetics and Robotics

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% USER INPUTS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
h  = 0.05;    % sampling time [s]
Ns = 1000;    % no. of samples

psi_ref = 10 * pi/180;  % desired yaw angle (rad)
u_ref   = 7;            % desired surge speed (m/s)

% ship parameters
m = 17.0677e6;          % mass (kg)
Iz = 2.1732e10;         % yaw moment of inertia (kg m^3)
xg = -3.7;              % CG x-ccordinate (m)
L = 161;                % length (m)
B = 21.8;               % beam (m)
T = 8.9;                % draft (m)
KT = 0.7;               % propeller coefficient (-)
Dia = 3.3;              % propeller diameter (m)
rho = 1025;             % density of water (m/s^3)

% rudder limitations
delta_max  = 40 * pi/180;        % max rudder angle      (rad)
Ddelta_max = 5  * pi/180;        % max rudder derivative (rad/s)

% added mass matrix
Xudot = -8.9830e5;
Yvdot = -5.1996e6;
Yrdot =  9.3677e5;
Nvdot =  Yrdot;
Nrdot = -2.4283e10;

% added mass matrix
MA = -[Xudot 0     0;
       0     Yvdot Yrdot;
       0     Nvdot Nrdot]

% rigid-body mass matrix
MRB = [ m 0    0
        0 m    m*xg
        0 m*xg Iz ];

% Inverted M to use to calculate nu
Minv = inv(MRB+MA);

% Linear damping
T1 = 20;
T2 = 20;
T6 = 10;
Xu = - (m - Xudot)/T1;
Yv = - (m - Yvdot)/T2;
Nr = - (Iz - Nrdot)/T6;

D = diag([-Xu, -Yv, -Nr])

% Constants
k = 0.1;
CR = 0;
epsilon = 0.001;
S = T*L*2 + T*B*2;   %Wettet surface area (m^2)
v = 1e-6;            %Kinematic viscosity (m/s^2)
```

```
66
67 %Cross flow drag coefficients
68 Cd_2D = Hoerner(B,T);
69 Yn = 0;
70 Nn = 0;
71
72 % input matrix
73 t_thr = 0.05;           % thrust deduction number
74 X_Δ2 = 0;           % rudder coefficients (Section 9.5)
75 Y_Δ = 0;
76 N_Δ = 1;
77 B = [ (1-t_thr)  X_Δ2
78         0        Y_Δ
79         0        N_Δ  ];
80
81 % initial states
82 eta = [0 0 0]';
83 nu  = [0.1 0 0.1]';
84 Δ = 0;
85 n = 0;
86
87 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
88 %% MAIN LOOP
89 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
90 simdata = zeros(Ns+1,14);               % table of simulation data
91
92 for i=1:Ns+1
93
94     t = (i-1) * h;                       % time (s)
95
96     % state-dependent time-varying matrices
97     CRB = m * nu(3) * [ 0 -1 -xg
98                         1  0  0
99                         xg 0  0  ];
100    % Added mass coriolis matrix
101    CA = [0                          0           Yvdot*nu(2)+Yrdot*nu(3);
102          0                          0           -Xudot*nu(1);
103          -Yvdot*nu(2)-Yrdot*nu(3)   Xudot*nu(1) 0];
104
105    C = CRB + CA
106
107    % Non linear surge damping
108    Rn = nu(1)*L/v;
109    Cf = 0.075/((log10(Rn) - 2)^2 + epsilon) + CR
110    Xn = -0.5*rho*S*(1+k)*Cf*abs(nu(1))*nu(1)
111
112    % Cross flow drag
113    dx = L/10;
114    for xL = -L/2:dx:L/2
115        Ucf = abs(nu(2) + xL *nu(3)) * (nu(2) + xL * nu(3));
116        Yn = Yn - 0.5 * rho * T * Cd_2D * Ucf * dx; % sway force
117        Nn = Nn - 0.5 * rho * T * Cd_2D * xL * Ucf * dx; % yaw moment
118    end
119
120    R = Rzyx(0,0,eta(3));
121
122    % reference models
123    psi_d = psi_ref;
124    r_d = 0;
125    u_d = u_ref;
126
127    % thrust
128    thr = rho * Dia^4 * KT * abs(n) * n;    % thrust command (N)
129
130    % control law
131    Δ_c = 0.1;            % rudder angle command (rad)
132    n_c = 10;                % propeller speed (rps)
133
134    % ship dynamics
135    u = [ thr Δ ]';
```

```
    tau = B * u;
    nu_dot = Minv * (tau - C * nu - D * nu + [Xn, Yn, Nn]');
    eta_dot = R * nu;

    % Rudder saturation and dynamics (Sections 9.5.2)
    if abs(δ_c) ≥ δ_max
        δ_c = sign(δ_c)*δ_max;
    end

    δ_dot = δ_c - δ;
    if abs(δ_dot) ≥ Dδ_max
        δ_dot = sign(δ_dot)*Dδ_max;
    end

    % propeller dynamics
    n_dot = (1/10) * (n_c - n);

    % store simulation data in a table (for testing)
    simdata(i,:) = [t n_c δ_c n δ eta' nu' u_d psi_d r_d];

    % Euler integration
    eta = euler2(eta_dot,eta,h);
    nu  = euler2(nu_dot,nu,h);
    δ   = euler2(δ_dot,δ,h);
    n   = euler2(n_dot,n,h);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOTS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t       = simdata(:,1);                 % s
n_c     = 60 * simdata(:,2);            % rpm
δ_c = (180/pi) * simdata(:,3);      % deg
n       = 60 * simdata(:,4);            % rpm
δ   = (180/pi) * simdata(:,5);      % deg
x       = simdata(:,6);                 % m
y       = simdata(:,7);                 % m
psi     = (180/pi) * simdata(:,8);      % deg
u       = simdata(:,9);                 % m/s
v       = simdata(:,10);                % m/s
r       = (180/pi) * simdata(:,11);     % deg/s
u_d     = simdata(:,12);                % m/s
psi_d   = (180/pi) * simdata(:,13);     % deg
r_d     = (180/pi) * simdata(:,14);     % deg/s

figure(1)
figure(gcf)
subplot(311)
plot(y,x,'linewidth',2); axis('equal')
title('North-East positions (m)'); xlabel('(m)'); ylabel('(m)');
subplot(312)
plot(t,psi,t,psi_d,'linewidth',2);
title('Actual and desired yaw angles (deg)'); xlabel('time (s)');
subplot(313)
plot(t,r,t,r_d,'linewidth',2);
title('Actual and desired yaw rates (deg/s)'); xlabel('time (s)');


figure(2)
figure(gcf)
subplot(311)
plot(t,u,t,u_d,'linewidth',2);
title('Actual and desired surge velocities (m/s)'); xlabel('time (s)');
subplot(312)
plot(t,n,t,n_c,'linewidth',2);
title('Actual and commanded propeller speed (rpm)'); xlabel('time (s)');
subplot(313)
plot(t,δ,t,δ_c,'linewidth',2);
title('Actual and commanded rudder angles (deg)'); xlabel('time (s)');
```

References