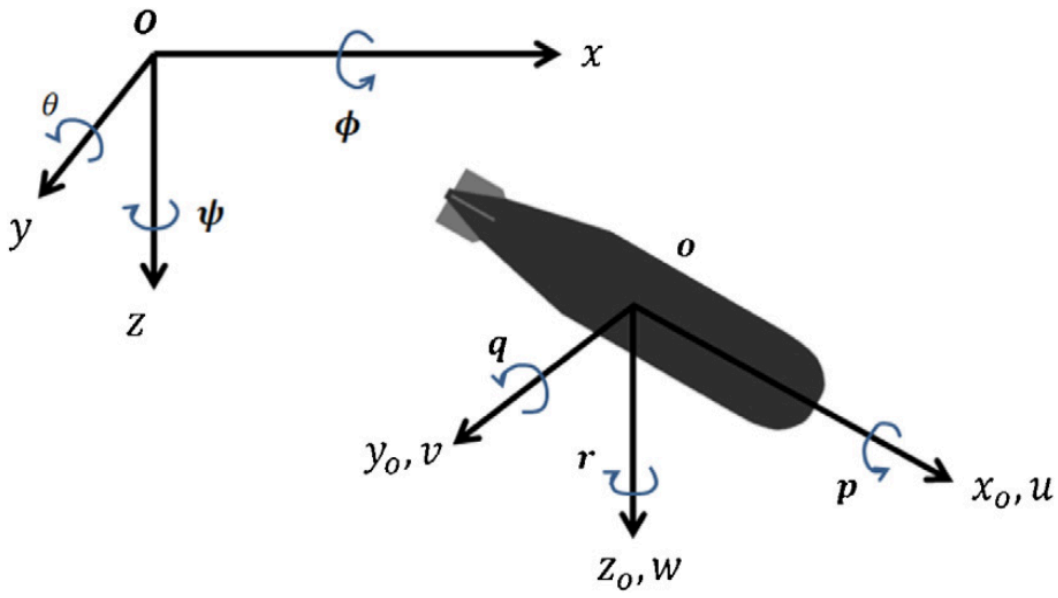
ALS Mid-Term Project

submitted by **: Karthik Balaji Keshavamurthi**

Problem 1. State Space Formulation:



Equations of motion:

To obtain the Dynamic equations of motion, we analyze all the forces and moments on the submarine at respective directions and then use Newton's Second law to obtain the following equations.

$$\left(m + X_{\dot{u}}\right)\ddot{x} = -X_u \dot{x} + X_T \qquad (1)$$

$$\left(m + Z_{\dot{w}}\right)\ddot{z} = -Z_{\dot{q}}\ddot{\theta} - Z_w \dot{z} + Z_q \dot{\theta} + Z_s + Z_b \qquad (2)$$

$$\left(I_y + M_{\dot{q}}\right)\ddot{\theta} = -M_{\dot{w}}\ddot{z} - M_q \dot{\theta} - M_w \dot{\theta} - x_s Z_s - x_b Z_b \qquad (3)$$

where the constants are:

$X_{\dot{u}}$ is the (-direction “added mass” from accelerating water (kg)

X_u is the (-direction hydrodynamic “drag” coefficient (N•s/m)

$Z_{\dot{w}}$ is the (-direction “added mass” from accelerating water (kg)

Z_w is the (-direction hydrodynamic “drag” coefficient (N•s/m)

$Z_{\dot{q}}$ is the (-direction “added mass” caused by rotation (kg•m)

Z_q is the (-direction hydrodynamic drag caused by rotation (kg•m/s)

$M_{\dot{q}}$ is the “added rotational inertia” about the (-axis (kg•m2)

M_q is the moment “drag” coefficient about the (-axis (N•m•s)

$M_{\dot{w}}$ is the “added rotational inertia” about the (-axis (kg•m)

M_w is the moment “drag” coefficient about the (-axis (N•s)

x_s is the position of the stern (rear) control surface in the (-direction (m)

x_b is the position of the bow (front) control surface in the (-direction (m)

Inputs:

The inputs to the system are the following:

- X_T - the thurst in the x_0 direction in newton.
- Z_s - the stern thrust in the $z_o$direction in newton
- Z_b - the blow thrust in the $z_0$direction in newton

Outputs:

The outputs that we measure using our sensors are the following:

- θ - The pitch angle in radians
- z - The depth in meters
- x - The position in meters

State Variables:

To construct the state space equation, the following is my choice of state variables.

$$x_1 = \dot{x}$$
$$x_2 = x$$
$$x_3 = \dot{z}$$
$$x_4 = z$$
$$x_5 = \dot{\theta}$$
$$x_6 = \theta$$

Using these state variables, we rearrange the Equations of motion to get the state and output equations.

State Equations:

$$\dot{x_1} = \left[\frac{-X_u}{(m + X_{\dot{u}})}\right] . x_1 + \left[\frac{1}{(m + X_{\dot{u}})}\right] . X_T \qquad (3)$$

$$\dot{x_2} = x_1 \tag{4}$$

$$\dot{x_3} = \left[\frac{\left(I_y Z_w + M_{\dot{q}} Z_w - M_w Z_{\dot{q}}\right)}{\xi}\right] x_3 + \left[\frac{\left(I_y Z_q + M_{\dot{q}} Z_q - M_q Z_{\dot{q}}\right)}{\xi}\right] x_5 + \left[\frac{\left(-I_y - M_{\dot{q}} - Z_{\dot{q}} x_s\right)}{\xi}\right] Z_s + \left[\frac{\left(-I_y - M_{\dot{q}} - Z_{\dot{q}} x_b\right)}{\xi}\right] Z_b \tag{5}$$

$$\dot{x_4} = x_3 \tag{6}$$

$$\dot{x_5} = \left[\frac{-\left(m + Z_{\dot{w}}\right)\left(I_y Z_w + M_{\dot{q}} Z_w - M_w Z_{\dot{q}}\right) - Z_w \xi}{Z_{\dot{q}} \xi}\right] x_3 + \left[\frac{-\left(m + Z_{\dot{w}}\right)\left(I_y Z_q + M_{\dot{q}} Z_q - M_q Z_{\dot{q}}\right) - Z_q \xi}{Z_{\dot{q}} \xi}\right] x_5 + \left[\frac{\left(I_y + M_{\dot{q}} + Z_{\dot{q}} x_s\right)\left(m + Z_{\dot{w}}\right) + \xi}{Z_{\dot{q}} \xi}\right] Z_s + \left[\frac{\left(I_y + M_{\dot{q}} + Z_{\dot{q}} x_b\right)\left(m + Z_{\dot{w}}\right) + \xi}{Z_{\dot{q}} \xi}\right] Z_b \tag{7}$$

$$\dot{x_6} = x_5 \tag{8}$$

where,

$$\xi = M_{\dot{w}} Z_{\dot{q}} - m I_y - m M_{\dot{q}} - Z_{\dot{w}} I_y - M_{\dot{q}} Z_{\dot{w}}$$

Output Equations:

The output equations are:

$$x = x_2 \tag{9}$$

$$z = x_4 \tag{10}$$

$$\theta = x_6 \tag{11}$$

```
clear
close all
clc
```

State Space Formulation

```
% I have used Alphabets to denote terms for simplicity during derivation of
% the State Space Equations.
syms A B C D E F G H J K XS XB XT ZS ZB M IY EPS
syms Xudot Xu Zwdot Zw Zqdot Zq Mqdot Mq Mwdot Mw Xs Xb XT Zs Zb m

EPS = J*E - M*IY - M*G - C*IY - G*C;
```

```
A1 = [(-B/(M+A)) 0 0 0 0 0
       1 0 0 0 0 0
       0 0 ((IY*D + G*D - K*E)/EPS) 0 ((IY*F + G*F - H*E)/EPS) 0
       0 0 1 0 0 0
       0 0 (-((M+C)/E)*((IY*D + G*D - K*E)/EPS) - (D/E)) 0 (-((M+C)/E)*((IY*F + G*F - H*E)/EPS)
       0 0 0 0 1 0];
A1 = subs(A1,[A,B,C,D,E,F,G,H,J,K,M],[Xudot,Xu,Zwdot,Zw,Zqdot,Zq,Mqdot,Mq,Mwdot,Mw,m])
```

A1 =

$$\begin{pmatrix} -\frac{\mathrm{Xu}}{\mathrm{Xudot}+m} & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{\sigma_2}{\sigma_1} & 0 & -\frac{\sigma_3}{\sigma_1} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{(\mathrm{Zwdot}+m)\,\sigma_2}{\mathrm{Zqdot}\,\sigma_1}-\frac{\mathrm{Zw}}{\mathrm{Zqdot}} & 0 & \frac{(\mathrm{Zwdot}+m)\,\sigma_3}{\mathrm{Zqdot}\,\sigma_1}-\frac{\mathrm{Zq}}{\mathrm{Zqdot}} & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

where

$$\sigma_1 = \mathrm{IY}\ \mathrm{Zwdot} + \mathrm{Mqdot}\ \mathrm{Zwdot} - \mathrm{Mwdot}\ \mathrm{Zqdot} + \mathrm{IY}\ m + \mathrm{Mqdot}\ m$$

$$\sigma_2 = \mathrm{IY}\ \mathrm{Zw} + \mathrm{Mqdot}\ \mathrm{Zw} - \mathrm{Mw}\ \mathrm{Zqdot}$$

$$\sigma_3 = \mathrm{IY}\ \mathrm{Zq} - \mathrm{Mq}\ \mathrm{Zqdot} + \mathrm{Mqdot}\ \mathrm{Zq}$$

```
B1 = [(1/(M+A)) 0 0
       0 0 0
       0 ((-IY - G - E*Xs)/EPS) ((-IY - G - E*Xb)/EPS)
       0 0 0
       0 (((IY+G+E*Xs)*(M+C) + EPS)/(EPS*E)) (((IY+G+E*Xb)*(M+C) + EPS)/(EPS*E))
       0 0 0];

B1 = subs(B1,[A,B,C,D,E,F,G,H,J,K,M],[Xudot,Xu,Zwdot,Zw,Zqdot,Zq,Mqdot,Mq,Mwdot,Mw,m])
```

B1 =

$$\begin{pmatrix} \frac{1}{\text{Xudot} + m} & 0 \\ 0 & 0 \\ 0 & \frac{\sigma_2}{\sigma_1} \\ 0 & 0 \\ 0 & \frac{\text{IY Zwdot} + \text{Mqdot Zwdot} - \text{Mwdot Zqdot} + \text{IY } m + \text{Mqdot } m - (\text{Zwdot} + m)\,\sigma_2}{\text{Zqdot } \sigma_1} \quad \frac{\text{IY Zw}}{} \\ 0 & 0 \end{pmatrix}$$

where

$$\sigma_1 = \text{IY Zwdot} + \text{Mqdot Zwdot} - \text{Mwdot Zqdot} + \text{IY } m + \text{Mqdot } m$$

$$\sigma_2 = \text{IY} + \text{Mqdot} + \text{Xs Zqdot}$$

$$\sigma_3 = \text{IY} + \text{Mqdot} + \text{Xb Zqdot}$$

```
C1 = [0 1 0 0 0 0
      0 0 0 1 0 0
      0 0 0 0 0 1]
```

```
C1 = 3×6
     0     1     0     0     0     0
     0     0     0     1     0     0
     0     0     0     0     0     1
```

```
  D1 = zeros(3)
```

```
D1 = 3×3
     0     0     0
     0     0     0
     0     0     0
```

Problem 2. Calculation of Parameters

```
% Let us update values for the constants
mass = 500;                              % in kg
L = 25;                                 % in m
Iy = (1/300) * mass * (L^2);             % in kg m^2
X_udot = mass/30;                         % in kg
X_u = 94;                                % in N s/m
Z_wdot = (mass/10);                       % in kg
Z_w = 4.7e2;                             % in N s/m
Z_qdot = mass/20;                         % in kg m
Z_q = 9.5e2;                             % in kg m/s
M_qdot = Iy/20;                          % in kg m^2
M_q = 1.1e3;                             % in N m s
M_wdot = Iy/40;                          % in kg m
```

```
M_w = 320;                              % in N s
xs = -L/3;                             % in m
xb = L/3;                              % in m
Umax = 3e6;                            % in N
```

Problem 3. State Space Representation of System:

```
A_matrix = double(subs(A1,[Xudot,Xu,Zwdot,Zw,Zqdot,Zq,Mqdot,Mq,Mwdot,Mw,m,IY],[X_udot,X_u,Z_wdo
```

```
A_matrix = 6×6
   -0.1819         0         0         0         0         0
    1.0000         0         0         0         0         0
         0         0   -0.8422         0   -1.6834         0
         0         0    1.0000         0         0         0
         0         0   -0.2725         0   -0.9656         0
         0         0         0         0    1.0000         0
```

```
B_matrix = double(subs(B1,[Xudot,Xu,Zwdot,Zw,Zqdot,Zq,Mqdot,Mq,Mwdot,Mw,m,IY,Xs,Xb],[X_udot,X_u
```

```
B_matrix = 6×3
    0.0019         0         0
         0         0         0
         0    0.0015    0.0022
         0         0         0
         0    0.0076   -0.0077
         0         0         0
```

```
C_matrix = C1
```

```
C_matrix = 3×6
     0     1     0     0     0     0
     0     0     0     1     0     0
     0     0     0     0     0     1
```

```
D_matrix = D1
```

```
D_matrix = 3×3
     0     0     0
     0     0     0
     0     0     0
```

Problem 4: Check for Minimum Realization:

To check for this, we need to create an LTI object and then use the matlab minreal function for checking for the minimum realisation.

From the Documentation, we can see that the minreal function filters out any redundant states and provides the least realization.

If the object returned due to this function is similar to the original LTI object, our system is a minimum realization system.

```
% Let us first create the LTI Object
original_system = ss(A_matrix,B_matrix,C_matrix,D_matrix);
Minimum_Realisation = minreal(original_system)
```

```
Minimum_Realisation =
```

```
A =
          x1       x2       x3       x4       x5       x6
  x1  -0.1819        0        0        0        0        0
  x2       1        0        0        0        0        0
  x3       0        0  -0.8422        0   -1.683        0
  x4       0        0        1        0        0        0
  x5       0        0  -0.2725        0  -0.9656        0
  x6       0        0        0        0        1        0

B =
             u1          u2          u3
  x1   0.001935           0           0
  x2          0           0           0
  x3          0    0.001473    0.002167
  x4          0           0           0
  x5          0    0.007584   -0.007671
  x6          0           0           0

C =
     x1  x2  x3  x4  x5  x6
  y1   0   1   0   0   0   0
  y2   0   0   0   1   0   0
  y3   0   0   0   0   0   1

D =
     u1  u2  u3
  y1   0   0   0
  y2   0   0   0
  y3   0   0   0

Continuous-time state-space model.
```

From above, you can see the State space matrices are the same as our original_system's matrices.

Hence, our State Space Model is a **Minimum Realization.**

Problem 5: Determine Controllability:

We will have to find the Controllability matrix and check for its rank. I have made a conditional statement for this purpose.

```
Q =  ctrb(A_matrix,B_matrix);

if rank(Q) == size(A_matrix,1)
    % This means that Matrix is controllable. If condition is satisfied,
    % the following statement would be the output
    disp('Matrix is Completely Controllable')
else
    % If the rank is not full, the following statement will be the output.
    disp('Matrix is not Completely Controllable')
end
```

```
Matrix is Completely Controllable
```

Problem 6: Compute Open Loop Poles:

We will use the damp function to obtain the Natural Frequency, Damping Ratios and Open loop poles.

Now, the Natual frequency values returned from this section will be in rad/s and we are asked to display in terms of Hz.

Hence, we will also convert them to the required format and use the matlab Table function to show the results.

```
[natural_frequency, damping_ratios, open_loop_poles] =damp(original_system);
natural_frequency = natural_frequency * (1/(2*pi));
table(open_loop_poles,natural_frequency,damping_ratios,'VariableNames',{'Open Loop Poles', 'Nat
```

ans = 6×3 table

	Open Loop Poles	Natural Frequency in Hz	Damping Ratios
1	0	0	-1
2	0	0	-1
3	0	0	-1
4	-0.1819	0.0290	1
5	-0.2238	0.0356	1
6	-1.5840	0.2521	1

Problem 7: Initial Response:

We are given initial a set of initial conditions. We will use the matlab function 'inital' to get the response of the function in open loop and plot the response in a subplot.

```
initial_condition_vector = [100,50,-300,350,0,45*pi/180]';
[open_loop_output, time, open_loop_state] = initial(original_system,initial_condition_vector);
```

Now we also have to find the settling time and pplace a marker on the response.

For this we will first use the lsiminfo command to find the settling time and then place a marker.

```
% The Default Settling Time for lsiminfo is 2%. To change it to 5%, we will
% need to add two extra arguments to lsiminfo other than output and time.
s = lsiminfo(open_loop_output,time,'SettlingTimeThreshold',0.05)
```

s = 3×1 struct

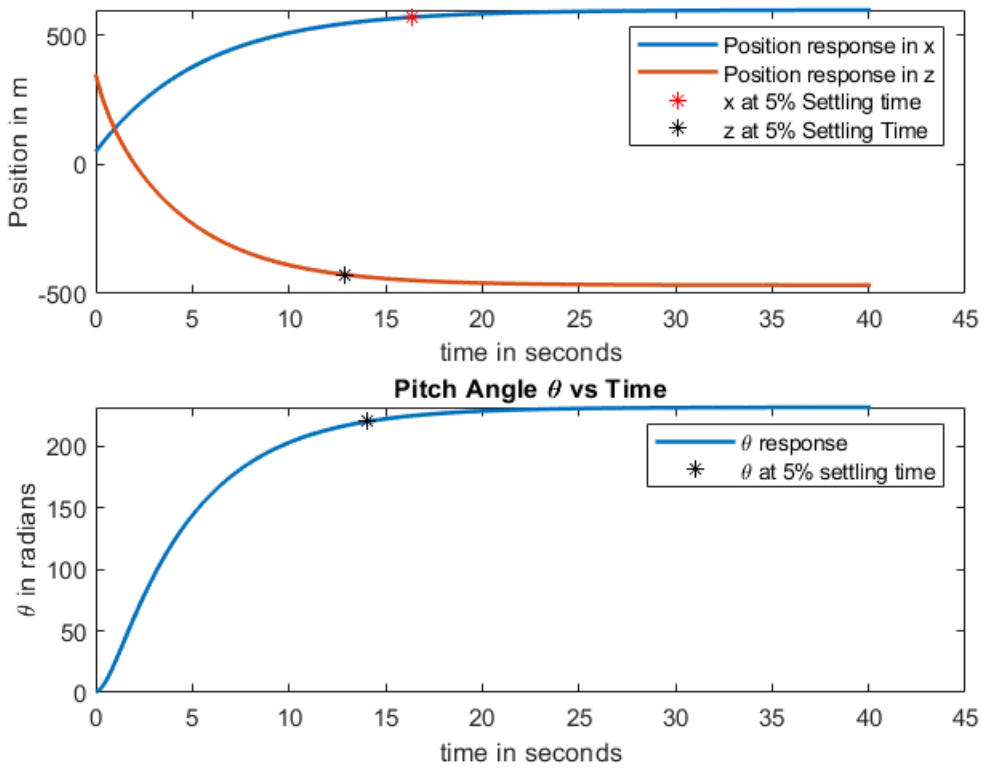
Fields	SettlingT...	Min	MinTime	Max	MaxTime
1	16.3958	50	0	599.2738	40.1203
2	12.8804	-467.1759	40.1203	350	0
3	14.0556	0.7854	0	231.4000	40.1203

```
% Now we have the three settling times. To place a marker, we need to find
% the index of that specific time.

[~,index_output1] = min(abs(time - s(1).SettlingTime));
[~,index_output2] = min(abs(time - s(2).SettlingTime));
[~,index_output3] = min(abs(time - s(3).SettlingTime));
```

```
% Now we plot the graphs. The x and z have the same units and hence will be
% plot in the same graph.
subplot(2,1,1)
title('Position vs Time')
plot(time, open_loop_output(:,1:2),'linewidth',1.5)
hold on
plot(time(index_output1,1),open_loop_output(index_output1,1),'r*')
hold on
plot(time(index_output2,1),open_loop_output(index_output2,2),'k*')
xlabel('time in seconds')
ylabel('Position in m')
legend('Position response in x', 'Position response in z','x at 5% Settling time','z at 5% Sett

subplot(2,1,2)
plot(time, open_loop_output(:,3),'linewidth',1.5)
hold on
plot(time(index_output3,1),open_loop_output(index_output3,3),'k*')
title('Pitch Angle \theta vs Time')
legend('\theta response','\theta at 5% settling time')
xlabel('time in seconds')
ylabel('\theta in radians')
```



Problem 8 - Full State Feedback Controller

Now we will start designing the full state feedback controller.