Table of Contents

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% Author: Samuel Razumovskiy
% Date written: 9/20/19
% Date modified: 9/26/19
%
% Purpose: Modeling quadcopter flight and comparing three different
 models
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

set(groot, 'defaulttextinterpreter', 'latex');
set(groot, 'defaultAxesTickLabelInterpreter', 'latex');
set(groot, 'defaultLegendInterpreter', 'latex');
```

Problem 2

```
clear,clc,close all
dpsi = 5*pi/180;
dtheta = 5*pi/180;
dphi = 5*pi/180;
dp = 0.1;
dq = 0.1;
dr = 0.1;
letter = ['a','b','c','d','e','f'];
changes =
 [dphi,0,0,0,0,0;0,dtheta,0,0,0,0;0,0,dpsi,0,0,0;0,0,0,dp,0,0;...
    0,0,0,0,dq,0;0,0,0,0,0,dr];
[row,~] = size(changes);
for i = 1:row
    change = changes(i,:);
    % Non changing values
    mass = 0.068; % kg
    g = 9.81; % m/s^2
    weight = mass*g; % N B frame 3x1
    radius = 0.060; % m
    del_f1 = -weight/4; % N B frame 3x1
    del_f2 = -weight/4; % N B frame 3x1
    del_f3 = -weight/4; % N B frame 3x1
    del_f4 = -weight/4; % N B frame 3x1
    I_B = [6.8e-5,0,0;0,9.2e-5,0;0,0,1.35e-4]; % kg/m^2 B frame 3x3
```

```matlab
    tspan = [0 10];

    % Initial conditions for changing variables
    pos_N = 0; pos_E = 0; pos_D = -5; % m E frame
    velE_xB = 0; velE_yB = 0; velE_zB = 0; % m/s B frame
    y_trans = [pos_N; pos_E; pos_D; velE_xB; velE_yB; velE_zB];

    psi = change(3); theta = change(2); phi = change(1); % rad
    p = change(4); q = change(5); r = change(6); % rad/s B frame
    y_rot   = [psi; theta; phi; p; q; r];

    y = [y_trans;y_rot];

    opt = odeset('maxstep',0.001);
    [t,pos] =
ode45(@(t,y)quadcopter_ODE(t,y,mass,I_B,g,radius,del_f1,del_f2,del_f3,del_f4),tsp

    figure(i)
    subplot(2,1,1)
    hold on
    plot(t,pos(:,1))
    plot(t,pos(:,2))
    plot(t,pos(:,3))
    xlabel('Time (s)')
    ylabel('Position (m)')
    title(sprintf("Problem 2%s Position",letter(i)))
    legend('N','E','D','location','southeast')
    subplot(2,1,2)
    hold on
    plot(t,pos(:,7).*180./pi)
    plot(t,pos(:,8).*180./pi)
    plot(t,pos(:,9).*180./pi)
    xlabel('Time (s)')
    ylabel('Orientation (deg)')
    title(sprintf("Problem 2%s Orientation",letter(i)))
    legend('psi','theta','phi','location','southeast')

end
```
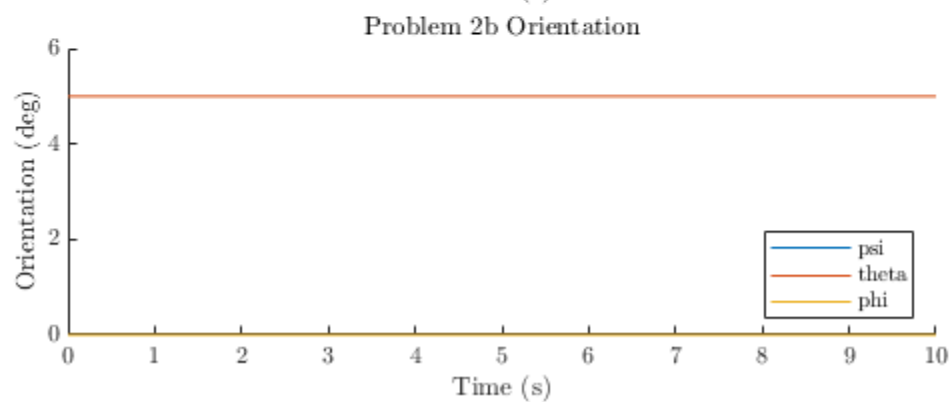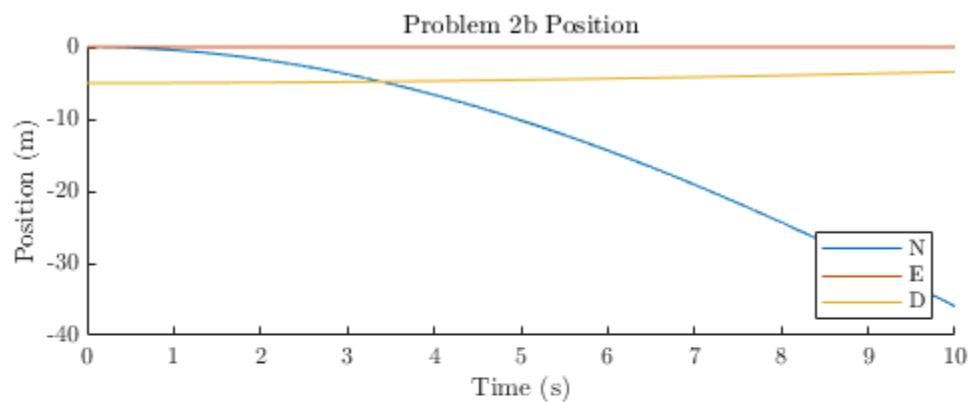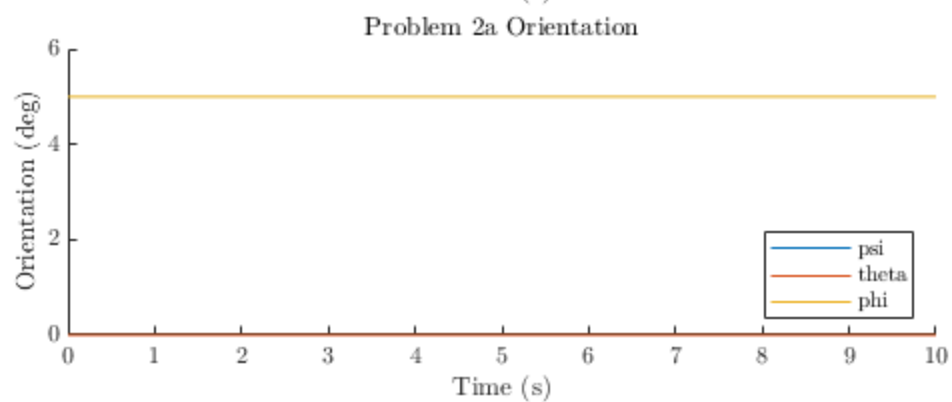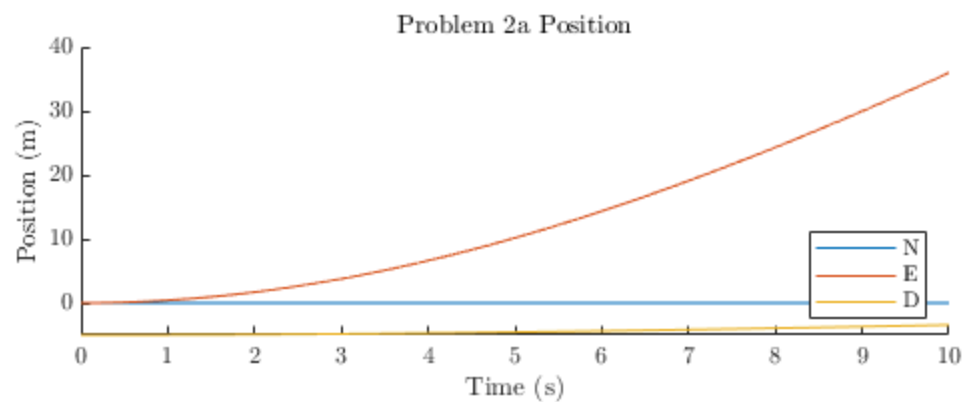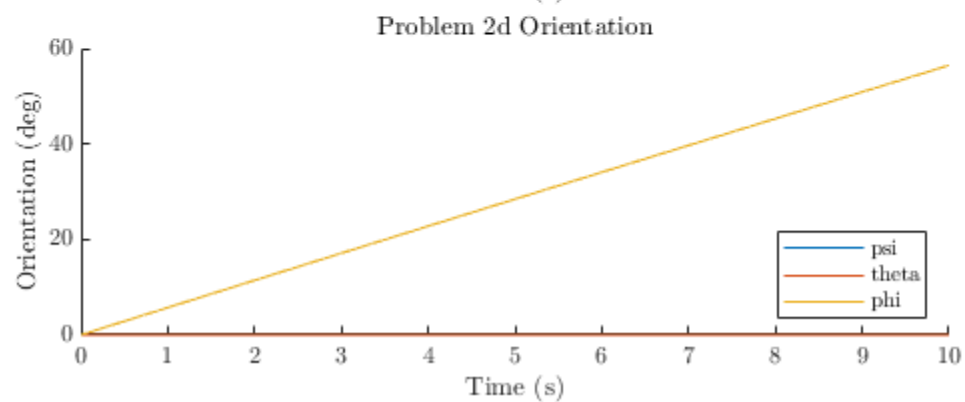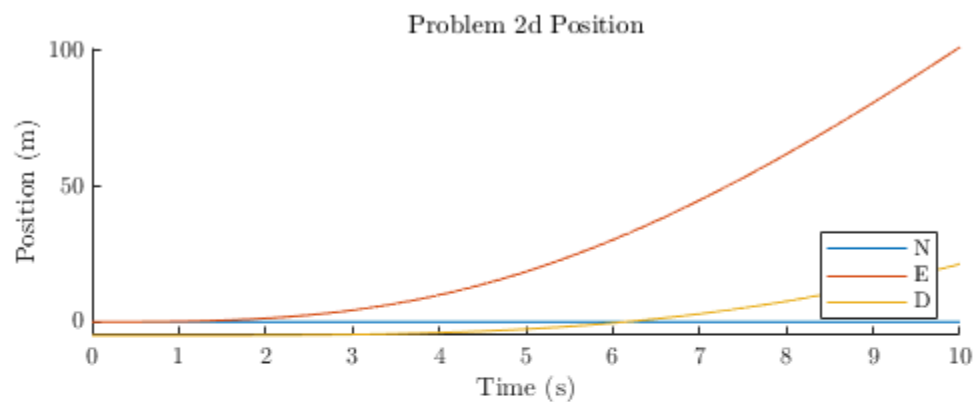
Problem 2a Position

Problem 2a Orientation

Problem 2b Position

Problem 2b Orientation

Problem 2c Position

Problem 2c Orientation

Problem 2d Position

Problem 2d Orientation

Problem 2e Position

Problem 2e Orientation

Problem 2f Position

Problem 2f Orientation
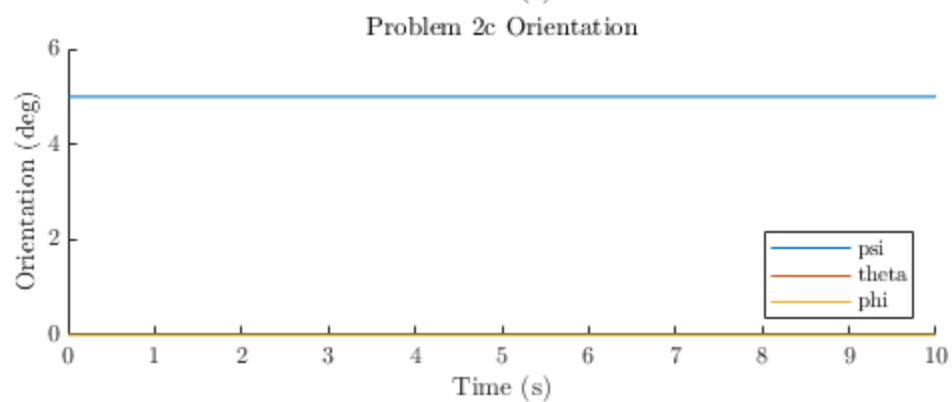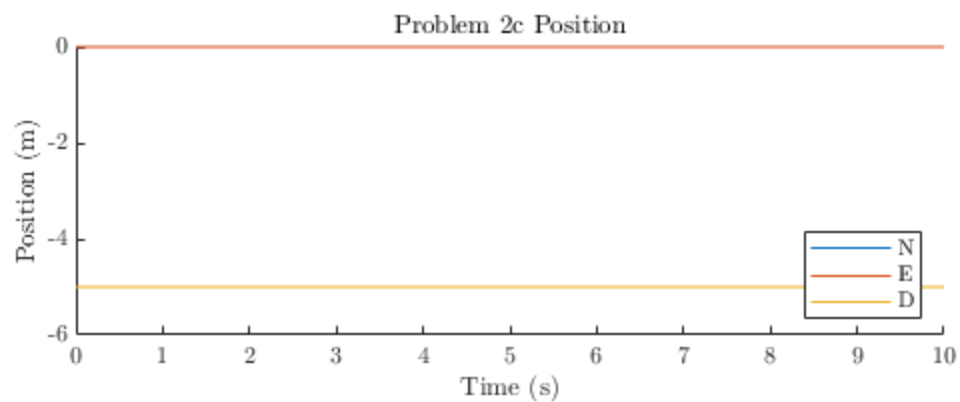
Problem 3

```matlab
clear,clc,close all
dpsi = 5*pi/180;
dtheta = 5*pi/180;
dphi = 5*pi/180;
dp = 0.1;
dq = 0.1;
dr = 0.1;
letter = ['a','b','c','d','e','f'];
changes =
 [dphi,0,0,0,0,0;0,dtheta,0,0,0,0;0,0,dpsi,0,0,0;0,0,0,dp,0,0;...
    0,0,0,0,dq,0;0,0,0,0,0,dr];
[row,~] = size(changes);
for i = 1:row
    change = changes(i,:);
    % Non changing values
    mass = 0.068; % kg
    g = 9.81; % m/s^2
    weight = mass*g; % N B frame 3x1
    radius = 0.060; % m
    del_f1 = 0; % N B frame 3x1
    del_f2 = 0; % N B frame 3x1
    del_f3 = 0; % N B frame 3x1
    del_f4 = 0; % N B frame 3x1
    I_B = [6.8e-5,0,0;0,9.2e-5,0;0,0,1.35e-4]; % kg/m^2 B frame 3x3
    tspan = [0 10];

    % Initial conditions for changing variables
    pos_N = 0; pos_E = 0; pos_D = -5; % m E frame
    velE_xB = 0; velE_yB = 0; velE_zB = 0; % m/s B frame
    y_trans = [pos_N; pos_E; pos_D; velE_xB; velE_yB; velE_zB];

    del_psi = change(3); del_theta = change(2); del_phi = change(1); %
rad
    del_p = change(4); del_q = change(5); del_r = change(6); % rad/s B
frame
    y_rot   = [del_psi; del_theta; del_phi; del_p; del_q; del_r];

    y = [y_trans;y_rot];

    opt = odeset('maxstep',0.001);
    [t,pos] =
ode45(@(t,y)linearized_quadcopter_ODE(t,y,mass,I_B,g,radius,del_f1,del_f2,del_f3,

    figure(7+i)
    subplot(2,1,1)
    hold on
    plot(t,pos(:,1))
    plot(t,pos(:,2))
    plot(t,pos(:,3))
    xlabel('Time (s)')
    ylabel('Position (m)')
```

```
        title(sprintf("Problem 2%s Position",letter(i)))
        legend('N','E','D','location','southeast')
        subplot(2,1,2)
        hold on
        plot(t,pos(:,7).*180./pi)
        plot(t,pos(:,8).*180./pi)
        plot(t,pos(:,9).*180./pi)
        xlabel('Time (s)')
        ylabel('Orientation (deg)')
        title(sprintf("Problem 2%s Orientation",letter(i)))
        legend('psi','theta','phi','location','southeast')
end
```
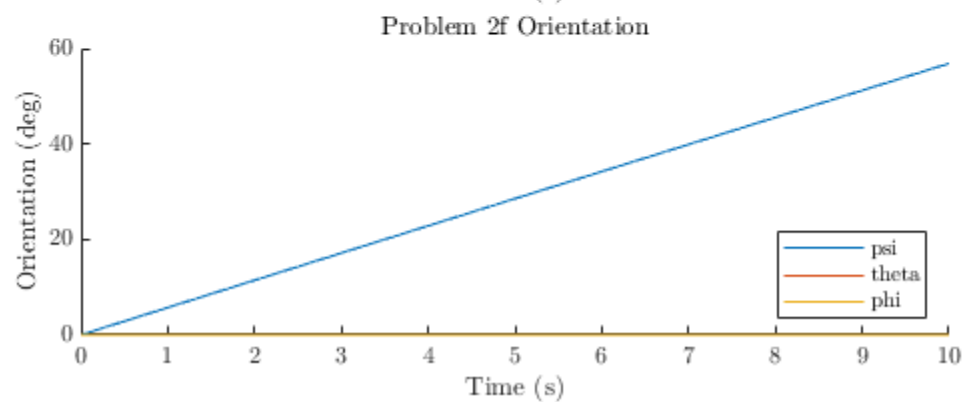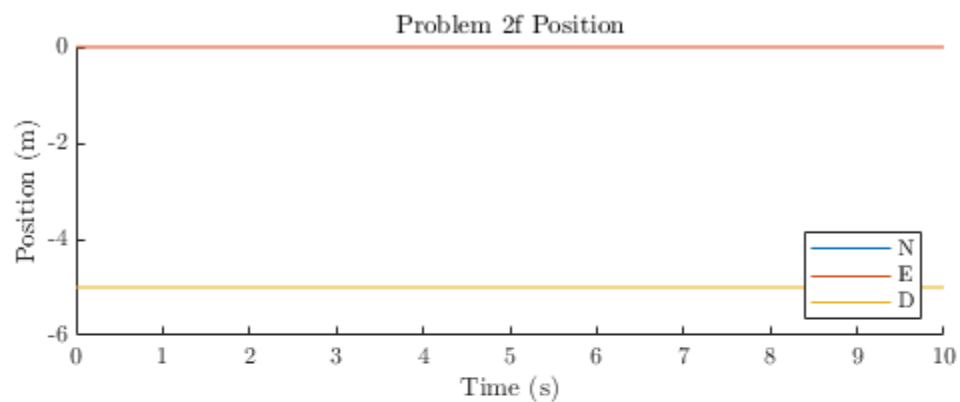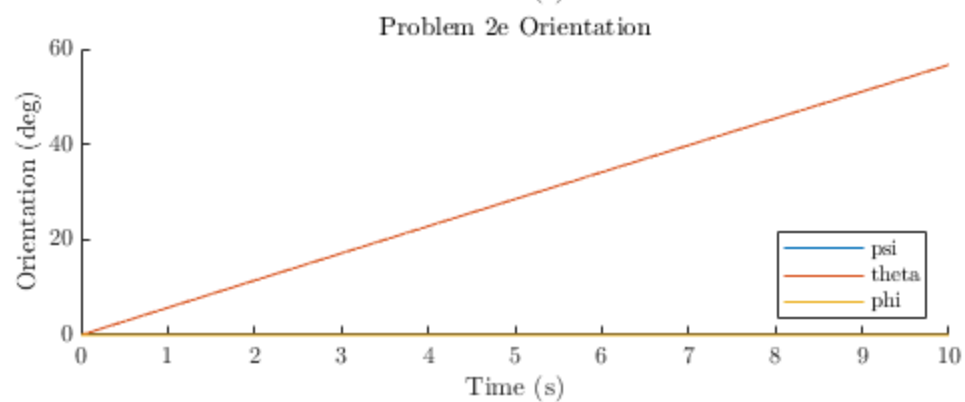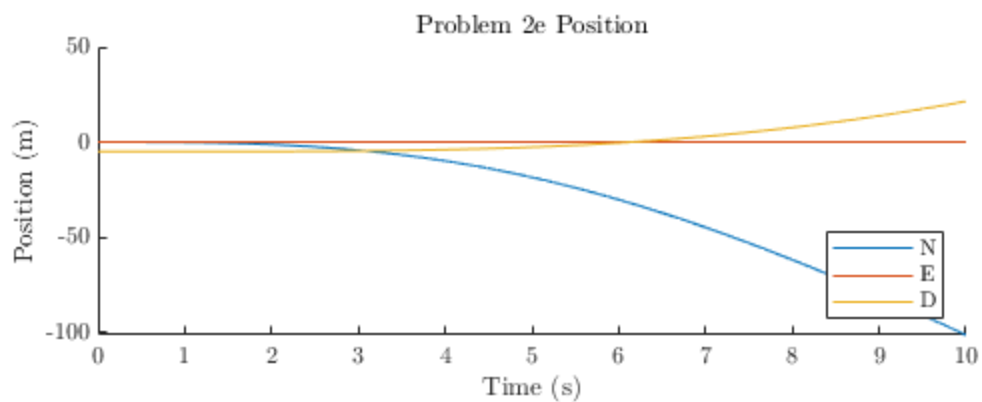
Problem 2b Position

Problem 2b Orientation

Problem 2c Position

Problem 2c Orientation

Problem 2d Position

Problem 2d Orientation

Problem 2e Position

Problem 2e Orientation

Problem 2f Position

Problem 2f Orientation
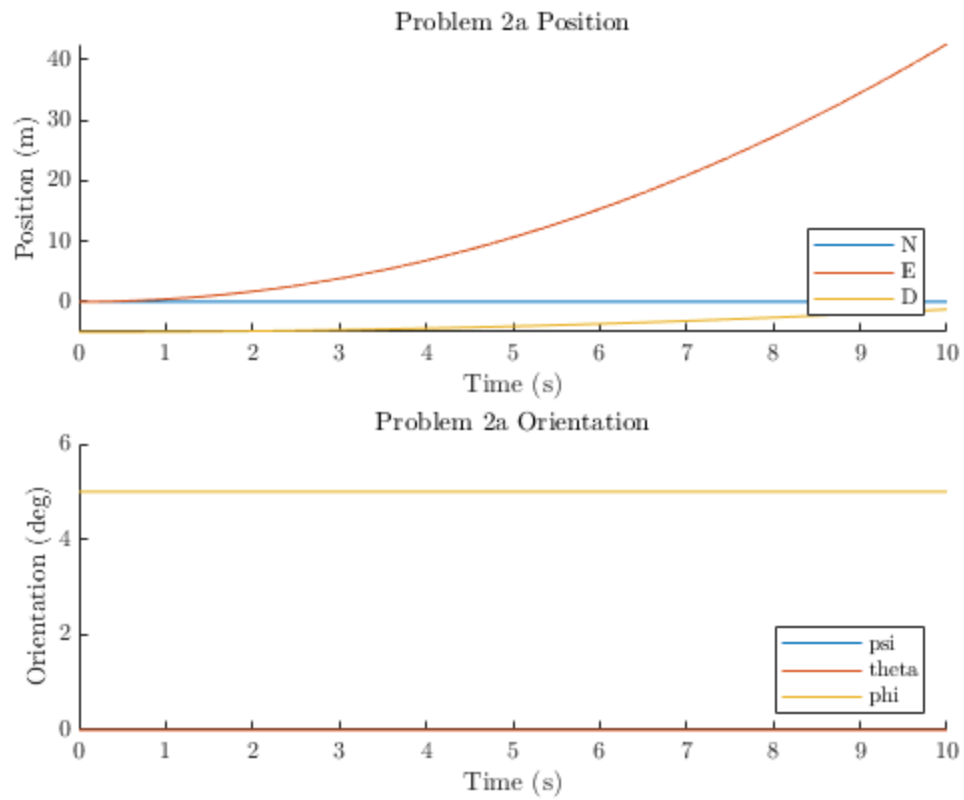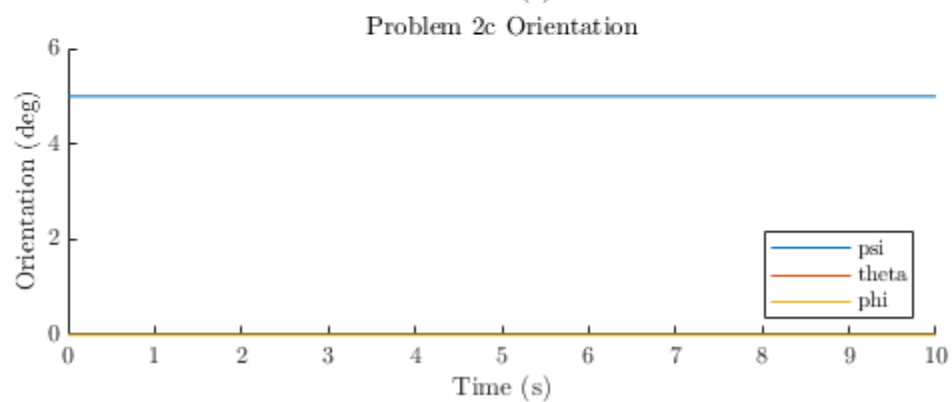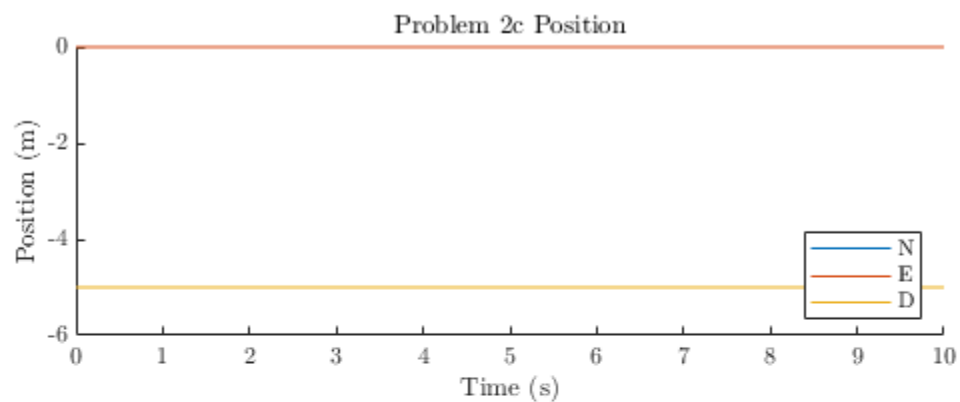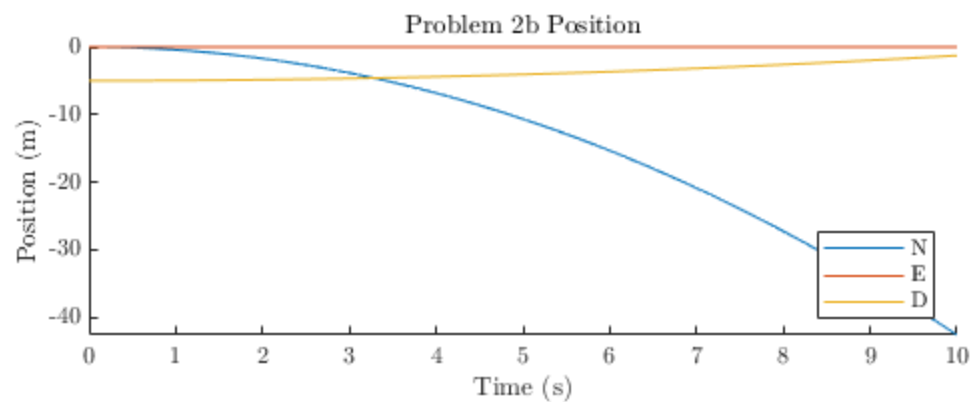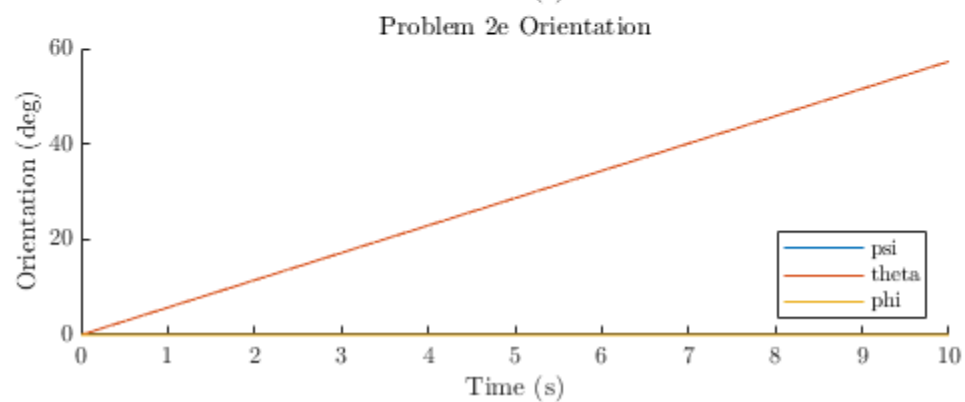
Problem 4

```
clear,clc,close all

% Non changing values
mass = 0.068; % kg
g = 9.81; % m/s^2
I_B = [6.8e-5,0,0;0,9.2e-5,0;0,0,1.35e-4]; % kg/m^2 B frame 3x3
tspan = [0 10];
dpsi = 5*pi/180;
dtheta = 5*pi/180;
dphi = 5*pi/180;
dp = 0.1;
dq = 0.1;
dr = 0.1;
k1 = 0.004; % Nm/(rad/s)
letter = ['d','e','f'];
changes = [0,0,0,dp,0,0;0,0,0,0,dq,0;0,0,0,0,0,dr];
[row,~] = size(changes);
for i = 1:row
    change = changes(i,:);
    % Initial conditions for changing variables
    pos_N = 0; pos_E = 0; pos_D = -5; % m E frame
    velE_xB = 0; velE_yB = 0; velE_zB = 0; % m/s B frame
    y_trans = [pos_N; pos_E; pos_D; velE_xB; velE_yB; velE_zB];
```

```matlab
    psi = change(3); theta = change(2); phi = change(1); % rad
    p = change(4); q = change(5); r = change(6); % rad/s B frame
    y_rot   = [psi; theta; phi; p; q; r];

    y = [y_trans;y_rot];

    opt = odeset('maxstep',0.001);
    [t,pos] =
 ode45(@(t,y)FB_quadcopter_ODE(t,y,mass,I_B,g,k1),tspan,y,opt);

    figure(14+i)
    subplot(2,1,1)
    hold on
    plot(t,pos(:,1))
    plot(t,pos(:,2))
    plot(t,pos(:,3))
    xlabel('Time (s)')
    ylabel('Position (m)')
    title(sprintf("Problem 2%s Position",letter(i)))
    legend('N','E','D','location','southeast')
    subplot(2,1,2)
    hold on
    plot(t,pos(:,7).*180./pi)
    plot(t,pos(:,8).*180./pi)
    plot(t,pos(:,9).*180./pi)
    xlabel('Time (s)')
    ylabel('Orientation (deg)')
    title(sprintf("Problem 2%s Orientation",letter(i)))
    legend('psi','theta','phi','location','southeast')
end
```
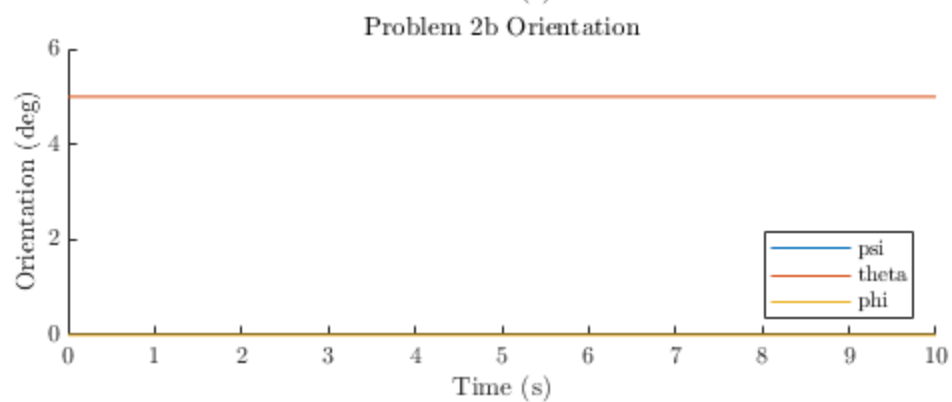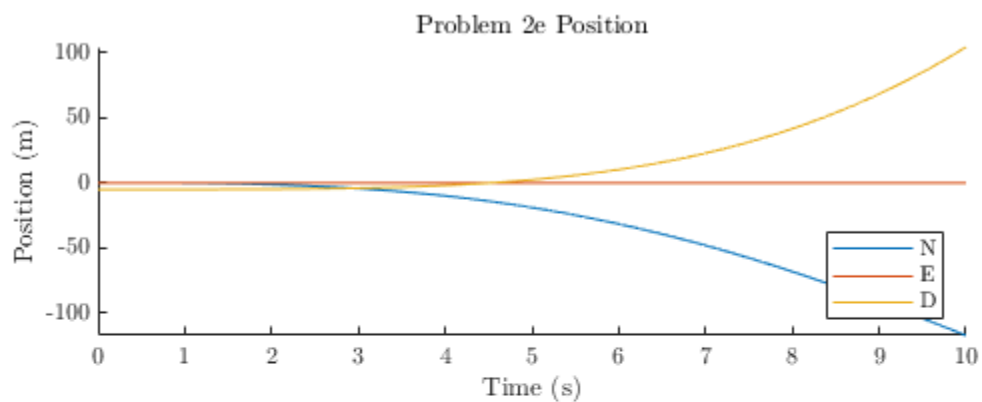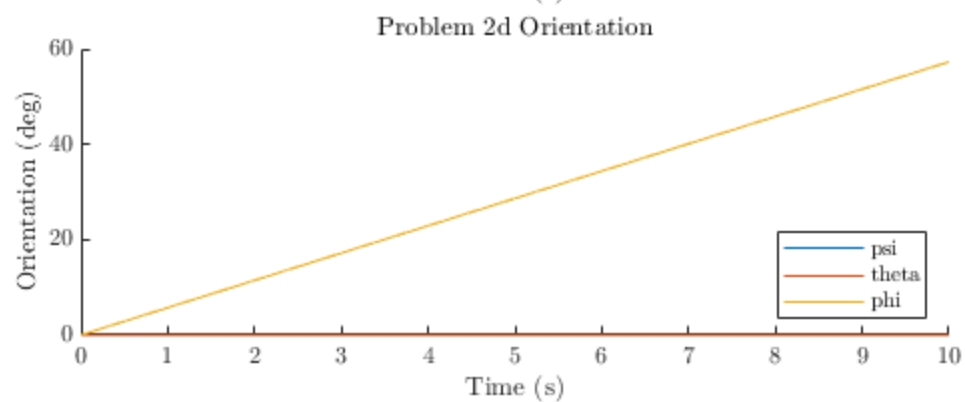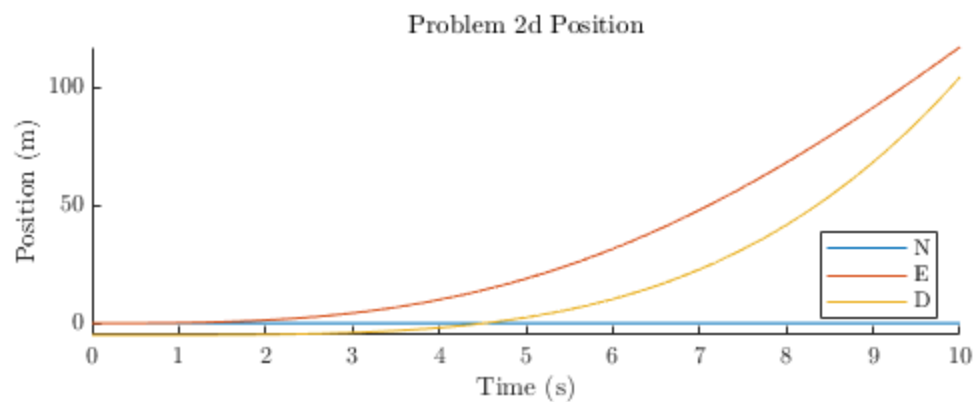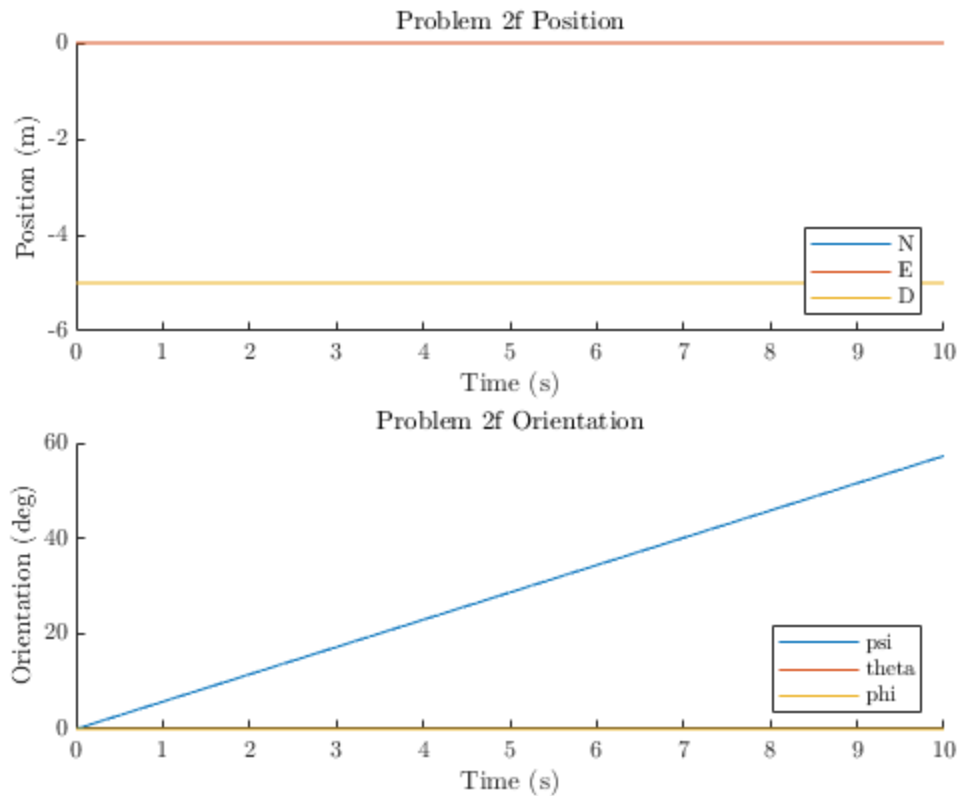
Problem 2d Position

Problem 2d Orientation

Problem 2e Position

Problem 2e Orientation

Problem 2f Position

Problem 2f Orientation
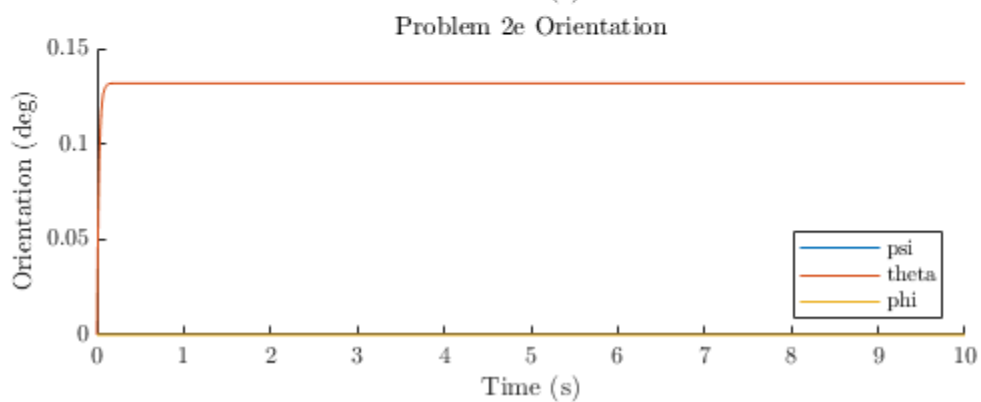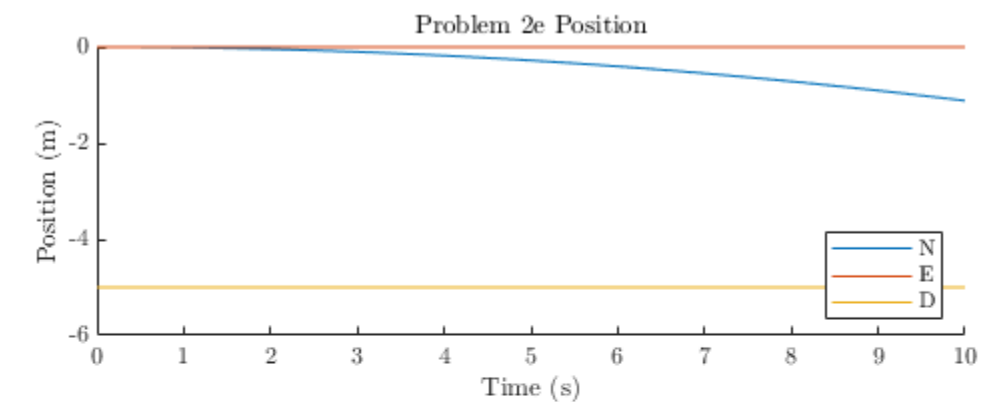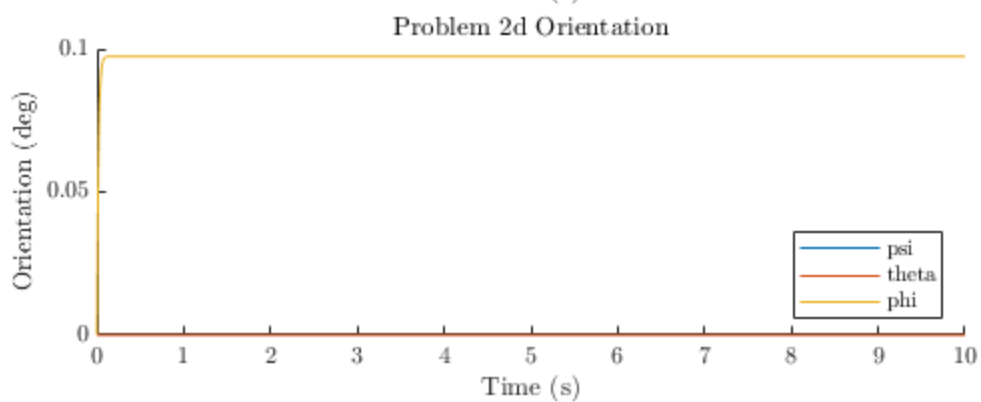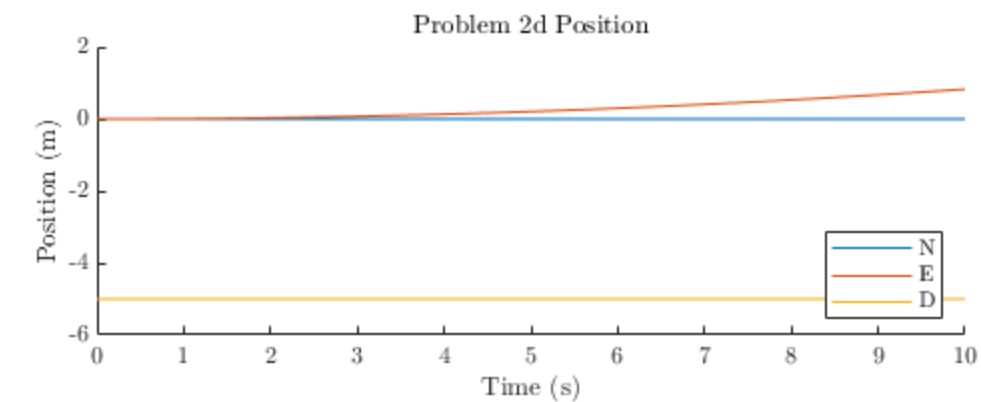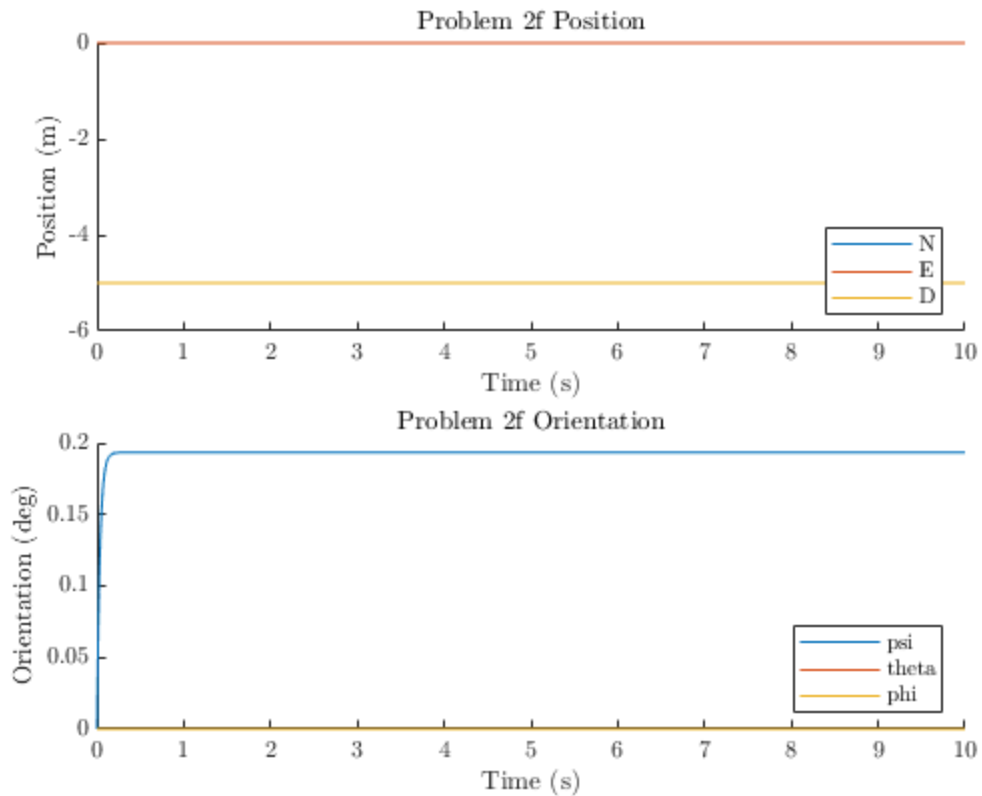
Problem 5

```
clear,clc,close all

load RSdata_White_1516.mat
times = rt_estim.time(:);
xdata = rt_estim.signals.values(:,1);
ydata = rt_estim.signals.values(:,2);
zdata = rt_estim.signals.values(:,3);
psi = rt_estim.signals.values(:,4);
theta = rt_estim.signals.values(:,5);
phi = rt_estim.signals.values(:,6);

figure(18)
subplot(2,1,1)
hold on
plot(times,xdata)
plot(times,ydata)
plot(times,zdata)
xlabel('Time (s)')
ylabel('Position (m)')
title("Problem 4 Position")
legend('x','y','z','location','southwest')

subplot(2,1,2)
```

```matlab
hold on
plot(times,psi.*180./pi)
plot(times,theta.*180./pi)
plot(times,phi.*180./pi)
xlabel('Time (s)')
ylabel('Orientation (deg)')
title("Problem 4 Orientation")
legend('psi','theta','phi','location','southwest')
```


Problem 4 Position

Problem 4 Orientation
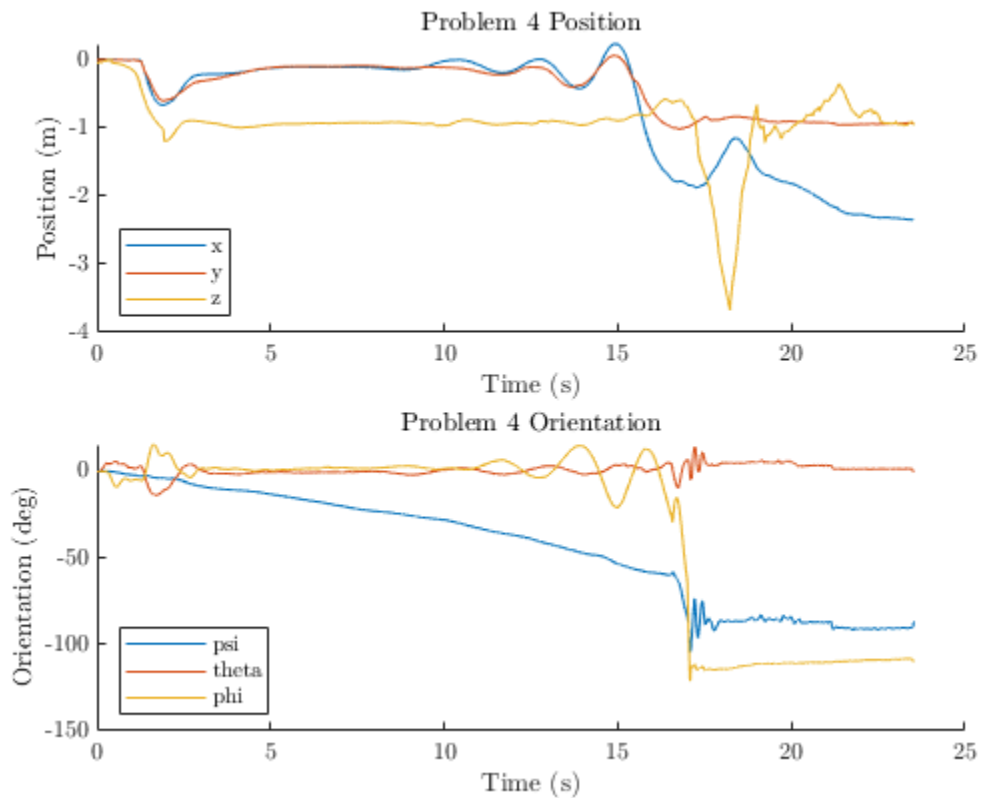
Functions Called

The following functions were built and called as part of this assignment.

```matlab
function [dydt] = quadcopter_ODE(t,y,mass,I_B,g,radius,f1,f2,f3,f4)
% quadcopter_ODE This function is used to simulate trimmed flight of a
% quadcopter with simplified kinematics and dynamics.

% Position in E frame
pos_N = y(1); pos_E = y(2); pos_D = y(3); % m
% Velocity in B frame
vel_xB = y(4); vel_yB = y(5); vel_zB = y(6); % m/s
psi = y(7); theta = y(8); phi = y(9); % rad
p = y(10); q = y(11); r = y(12); % rad/s
VE_B = [vel_xB;vel_yB;vel_zB];
wEB = [p;q;r];
```

```matlab
% Transfer matrix from B to E frame coordinates
L_EB = [cos(theta)*cos(psi),sin(phi)*sin(theta)*cos(psi)-
cos(phi)*sin(psi),...

 cos(phi)*sin(theta)*cos(psi)+sin(phi)*sin(psi);cos(theta)*sin(psi),...
    sin(phi)*sin(theta)*sin(psi)+cos(phi)*cos(psi),...
    cos(phi)*sin(theta)*sin(psi)-sin(phi)*cos(psi);...
    -sin(theta),sin(phi)*cos(theta),cos(phi)*cos(theta)];

% Weight in B frame coordinates
weight = L_EB\[0;0;mass*g];

% Drag in B frame coordinates
VEmag = norm(VE_B);
Aa = -1*10^-3.*VEmag.*VE_B;

% Forces from rotors and aerodynamic drag
Xc = 0;
Yc = 0;
Zc = f1+f2+f3+f4;
Ac = [Xc;Yc;Zc];
FB = weight+Aa+Ac;

% Acceleration in B frame coordinates
a_B = FB./mass-cross(wEB,VE_B);
uB_d = a_B(1);
vB_d = a_B(2);
wB_d = a_B(3);

% Velocity in E frame coordinates
VE_E = L_EB*VE_B;
vel_NE = VE_E(1);
vel_EE = VE_E(2);
vel_DE = VE_E(3);

% Finding the aerodynamic moments from rotors and drag
wmag = norm(wEB);
G_a = -2*10^-6*wmag.*[p;q;r];

Lc = radius/sqrt(2)*(f1+f2-f3-f4);
Mc = radius/sqrt(2)*(-f1+f2+f3-f4);
Nc = radius/sqrt(2)*(f1-f2+f3-f4);
G_c = [Lc;Mc;Nc];
G_B = G_a+G_c;

% Finding changes in psi,theta,phi from p,q,r
Rates = [1, sin(phi)*tan(theta), cos(phi)*tan(theta);...
         0, cos(phi),            -sin(phi);...
         0, sin(phi)*sec(theta), cos(phi)*sec(theta)]*[p;q;r];
phi_d = Rates(1);
theta_d = Rates(2);
psi_d = Rates(3);
```

```matlab
    % Finding change in p,q,r
    w_d = I_B\(cross(-wEB,I_B*wEB)+G_B);


    p_d = w_d(1);
    q_d = w_d(2);
    r_d = w_d(3);


    yd_trans = [vel_NE; vel_EE; vel_DE; uB_d; vB_d; wB_d];
    yd_rot  = [psi_d; theta_d; phi_d; p_d; q_d; r_d];


    dydt = [yd_trans;yd_rot];
    end



    function [dydt] =
     linearized_quadcopter_ODE(t,y,mass,I_B,g,radius,del_f1,del_f2,del_f3,del_f4)
    % quadcopter_ODE This function is used to simulate trimmed flight of a
    % quadcopter with simplified kinematics and dynamics.

    % Position in E frame
    del_pos_N = y(1); del_pos_E = y(2); del_pos_D = y(3); % m
    % Velocity in B frame
    del_vel_xB = y(4); del_vel_yB = y(5); del_vel_zB = y(6); % m/s
    del_psi = y(7); del_theta = y(8); del_phi = y(9); % rad
    del_p = y(10); del_q = y(11); del_r = y(12); % rad/s
    del_VE_B = [del_vel_xB;del_vel_yB;del_vel_zB];

    % Transfer matrix from B to E frame coordinates
    L_EB =
     [cos(del_theta)*cos(del_psi),sin(del_phi)*sin(del_theta)*cos(del_psi)-
    cos(del_phi)*sin(del_psi),...

     cos(del_phi)*sin(del_theta)*cos(del_psi)+sin(del_phi)*sin(del_psi);cos(del_theta)

     sin(del_phi)*sin(del_theta)*sin(del_psi)+cos(del_phi)*cos(del_psi),...
        cos(del_phi)*sin(del_theta)*sin(del_psi)-
    sin(del_phi)*cos(del_psi);...
        -
    sin(del_theta),sin(del_phi)*cos(del_theta),cos(del_phi)*cos(del_theta)];

    % Forces from rotors and aerodynamic drag
    del_Xc = 0;
    del_Yc = 0;
    del_Zc = del_f1+del_f2+del_f3+del_f4;
    del_Ac = [del_Xc;del_Yc;del_Zc];

    % Acceleration in B frame coordinates
    del_u_d = -g*del_theta;
    del_v_d = g*del_phi;
    del_w_d = 1/mass*del_Ac(3);

    % Velocity in E frame coordinates
    VE_E = L_EB*del_VE_B;
    vel_NE = VE_E(1);
```

```matlab
    vel_EE = VE_E(2);
    vel_DE = VE_E(3);

    % Finding the aerodynamic moments from rotors and drag
    del_Lc = radius/sqrt(2)*(del_f1+del_f2-del_f3-del_f4);
    del_Mc = radius/sqrt(2)*(-del_f1+del_f2+del_f3-del_f4);
    del_Nc = radius/sqrt(2)*(del_f1-del_f2+del_f3-del_f4);
    del_G_c = [del_Lc,del_Mc,del_Nc];

    % Finding change in p,q,r
    del_p_d = 1/I_B(1,1)*del_G_c(1);
    del_q_d = 1/I_B(2,2)*del_G_c(2);
    del_r_d = 1/I_B(3,3)*del_G_c(3);

    % Finding changes in psi,theta,phi from p,q,r
    del_psi_d = del_r;
    del_theta_d = del_q;
    del_phi_d = del_p;

    yd_trans = [vel_NE; vel_EE; vel_DE; del_u_d; del_v_d; del_w_d];
    yd_rot  = [del_psi_d; del_theta_d; del_phi_d; del_p_d; del_q_d;
     del_r_d];

    dydt = [yd_trans;yd_rot];
    end


    function [dydt] = FB_quadcopter_ODE(t,y,mass,I_B,g,k1)
    % quadcopter_ODE This function is used to simulate trimmed flight of a
    % quadcopter with simplified kinematics and dynamics.

    % Position in E frame
    pos_N = y(1); pos_E = y(2); pos_D = y(3); % m
    % Velocity in B frame
    vel_xB = y(4); vel_yB = y(5); vel_zB = y(6); % m/s
    psi = y(7); theta = y(8); phi = y(9); % rad
    p = y(10); q = y(11); r = y(12); % rad/s
    VE_B = [vel_xB;vel_yB;vel_zB];
    wEB = [p;q;r];

    % Transfer matrix from B to E frame coordinates
    L_EB = [cos(theta)*cos(psi),sin(phi)*sin(theta)*cos(psi)-
    cos(phi)*sin(psi),...

     cos(phi)*sin(theta)*cos(psi)+sin(phi)*sin(psi);cos(theta)*sin(psi),...
        sin(phi)*sin(theta)*sin(psi)+cos(phi)*cos(psi),...
        cos(phi)*sin(theta)*sin(psi)-sin(phi)*cos(psi);...
        -sin(theta),sin(phi)*cos(theta),cos(phi)*cos(theta)];

    % Weight in B frame coordinates
    weight = L_EB\[0;0;mass*g];

    % Drag in B frame coordinates
    VEmag = norm(VE_B);
```

```matlab
Aa = -1*10^-3.*VEmag.*VE_B;

% Forces from rotors and aerodynamic drag added together
Xc = 0;
Yc = 0;
Zc = -mass*g;
Ac = [Xc;Yc;Zc];
FB = weight+Aa+Ac;

% Acceleration in B frame coordinates
a_B = FB./mass-cross(wEB,VE_B);
uB_d = a_B(1);
vB_d = a_B(2);
wB_d = a_B(3);

% Velocity in E frame coordinates
VE_E = L_EB*VE_B;
vel_NE = VE_E(1);
vel_EE = VE_E(2);
vel_DE = VE_E(3);

% Finding the aerodynamic moments from rotors and drag
wmag = norm(wEB);
G_a = -2*10^-6*wmag.*[p;q;r];


Lc = -k1*p;
Mc = -k1*q;
Nc = -k1*r;
G_c = [Lc;Mc;Nc];
G_B = G_a+G_c;

% Finding changes in psi,theta,phi from p,q,r
Rates = [1, sin(phi)*tan(theta), cos(phi)*tan(theta);...
         0, cos(phi),            -sin(phi);...
         0, sin(phi)*sec(theta), cos(phi)*sec(theta)]*[p;q;r];
phi_d = Rates(1);
theta_d = Rates(2);
psi_d = Rates(3);

% Finding change in p,q,r
w_d = I_B\(cross(-wEB,I_B*wEB)+G_B);


p_d = w_d(1);
q_d = w_d(2);
r_d = w_d(3);

yd_trans = [vel_NE; vel_EE; vel_DE; uB_d; vB_d; wB_d];
yd_rot  = [psi_d; theta_d; phi_d; p_d; q_d; r_d];

dydt = [yd_trans;yd_rot];
end
```

Published with MATLAB® R2019b