1. $\dot{u} = \dfrac{X - mg\sin\theta - mqw + mrv}{m}$

$u_0 = v_0 = w_0 = 0 \quad L_0 = M_0 = N_0 = 0$

$p_0 = q_0 = r_0 = 0$

$\psi_0 = \theta_0 = \phi_0 = 0$

$X_0 = Y_0 = Z_0 = 0$

$\dot{u} = \underbrace{\left(\dfrac{X}{m} - g\sin\theta - qw + rv\right.}_{h_1}$

$\Delta\dot{u} = \dfrac{\partial h_1}{\partial X}\Big|_0 \cdot \Delta X + \dfrac{\partial h_1}{\partial q}\Big|_0 \cdot \Delta q + \dfrac{\partial h_1}{\partial w}\Big|_0 \cdot \Delta w + \dfrac{\partial h_1}{\partial r}\Big|_0 \cdot \Delta r + \dfrac{\partial h_1}{\partial v}\Big|_0 \cdot \Delta v + \dfrac{\partial h_1}{\partial \theta}\Big|_0 \cdot \Delta\theta$

$\Delta\dot{u} = \dfrac{\Delta X}{m} + g\cancel{\cos\theta_0}\Delta\theta - \cancel{w_0}\Delta q - \cancel{q_0}\Delta w + \cancel{v_0}\Delta r + \cancel{r_0}\Delta v$

$\Delta\dot{u} = \dfrac{\Delta X}{m} + g\Delta\theta \quad (1)$

$\dot{v} = \dfrac{Y + mg\cos\theta\sin\phi - mru + mpw}{m}$

$\dot{v} = \dfrac{Y}{m} + g\cos\theta\sin\phi - ru + pw$

$\dot{v} = \dot{v}_0^{\,0} + \Delta\dot{v}$

$= h_2(\qquad\qquad)$

$\Delta\dot{v} = \dfrac{\partial h_2}{\partial Y}\Big|_0 \Delta Y + \dfrac{\partial h_2}{\partial \theta}\Big|_0 \Delta\theta + \dfrac{\partial h_2}{\partial \phi}\Big|_0 \Delta\phi + \dfrac{\partial h_2}{\partial r}\Big|_0 \Delta r + \dfrac{\partial h_2}{\partial u}\Big|_0 \Delta u + \dfrac{\partial h_2}{\partial p}\Big|_0 \Delta p + \dfrac{\partial h_2}{\partial w}\Big|_0 \Delta w$

$\Delta\dot{v} = \dfrac{\Delta Y}{m} - \cancel{g\sin\theta_0\sin\phi_0\Delta\theta} + g\cancel{\cos\theta_0}\cos\phi_0\Delta\phi - \cancel{u_0}\Delta r - \cancel{r_0}\Delta u + \cancel{w_0}\Delta p + \cancel{p_0}\Delta w$

$\Delta\dot{v} = \dfrac{\Delta Y}{m} + g\Delta\phi \quad (2)$

$\dot{w} = \dfrac{Z + mg\cos\theta\cos\phi - mpv + mqu}{m}$

$\dot{w} = \dfrac{Z}{m} + g\cos\theta\cos\phi - pv + qu$

$= h_3(\qquad\qquad)$

$\Delta\dot{w} = \dfrac{\partial h_3}{\partial Z}\Big|_0 \Delta Z + \dfrac{\partial h_3}{\partial \theta}\Big|_0 \Delta\theta + \dfrac{\partial h_3}{\partial \phi}\Big|_0 \Delta\phi + \dfrac{\partial h_3}{\partial p}\Big|_0 \Delta p + \dfrac{\partial h_3}{\partial v}\Big|_0 \Delta v + \dfrac{\partial h_3}{\partial q}\Big|_0 \Delta q + \dfrac{\partial h_3}{\partial u}\Big|_0 \Delta u$

$\Delta\dot{w} = \dfrac{\Delta Z}{m} + g\cancel{\sin\theta}\cos\phi\,\Delta\theta - g\cos\theta\cancel{\sin\phi}\,\Delta\phi - \cancel{v_0}\Delta p^0 - \cancel{p_0}\Delta v_0 + \Delta q\,u_0^0 + \cancel{q_0 \Delta u}$

$\Delta\dot{w} = \dfrac{\Delta Z}{m} \quad (3)$

Scanned by CamScanner

$X = -\eta\sqrt{u^2+v^2+w^2}\cdot u$

$= g_1(\quad)$

$\Delta X = \frac{\partial g_1}{\partial u}\Big|_0 \Delta u + \frac{\partial g_1}{\partial v}\Big|_0 \Delta v + \frac{\partial g_1}{\partial w}\Big|_0 \Delta w$

$\Delta X = -\eta \frac{2u_0^2+\cancel{v_0^2+w_0^2}}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}}0 - \eta\frac{u_0 v_0^0}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}} + \frac{-\eta u_0 w_0^0}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}} = 0$

(1) $\boxed{\Delta\dot{u} = g\Delta\theta}$

$Y = -\eta\sqrt{u^2+v^2+w^2}\cdot v$

$\Delta Y = 0$

(2) $\boxed{\Delta\dot{v} = g\Delta\phi}$

$Z = \eta\sqrt{u^2+v^2+w^2}\cdot w + Z_c$

$= g_3($

$\Delta Z = \frac{\partial g_3}{\partial u}\Big|_0 \Delta u + \frac{\Delta g_3}{\partial v}\Big|_0 \Delta v + \frac{\Delta g_3}{\partial w}\Big|_0 \Delta w + \frac{\partial g_3}{\partial f_1}\Big|_0 \Delta f_1 \ldots \frac{\partial g_3}{\partial f_4}\Big|_0 \Delta f_4$

$\Delta Z = -\eta\frac{2u_0^2+\cancel{w_0^2+v_0^2}^{\,0}}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}} - \eta\frac{u_0 v_0^0}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}} - \eta\frac{u_0 w_0^0}{\sqrt{\cancel{u_0^2+v_0^2+w_0^2}}} + \Delta f_1 + \Delta f_2 + \Delta f_3 + \Delta f_4$

$\Delta Z = \Delta Z_c$

(3) $\boxed{\Delta\dot{w} = \frac{\Delta Z_c}{m}}$

Scanned by CamScanner

$$\dot p = \frac{L + I_{zx}\dot r + qr(I_z - I_y) - I_{zx} pq}{I_x} \qquad I_{zx}=0$$

$$\dot p = \frac{L}{I_x} + \frac{qr(I_z-I_y)}{I_x} = h_4(\quad) \qquad \Delta \dot p = \frac{\partial h_4}{\partial r}\Big|_0 \Delta r + \frac{\partial h_4}{\partial L}\Big|_0 \Delta L + \frac{\partial h_4}{\partial q}\Big|_0 \Delta q$$

$$\Delta \dot p = \cancel{\frac{q_0\Delta r(I_z-I_y)}{I_x}}^{\,0} + \frac{\Delta L}{I_x} + \cancel{\frac{\Delta q\, r_0 (I_z-I_y)}{I_x}}^{\,0} = \frac{\Delta L}{I_x} \qquad \Delta\dot p = \frac{\Delta L}{I_x} \ (4)$$

$$L = -\alpha\sqrt{p^2+q^2+r^2}\cdot p + L_c = g_4(\quad) \qquad \Delta L = \frac{\partial g_4}{\partial p}\Big|_0 \Delta p + \frac{\partial g_4}{\partial q}\Big|_0\Delta q + \frac{\partial g_4}{\partial r}\Big|_0 \Delta r + \frac{\partial g_4}{\partial L_c}\Big|_0 \Delta L_c$$

$$\Delta L = -\alpha \frac{2p_0^2 + \cancel{q_0^2 + r_0^2}}{\sqrt{\cancel{p_0^2+q_0^2+r_0^2}}}^{\,0}\Delta p - \alpha \cancel{\frac{p_0 q_0}{\sqrt{p_0^2+q_0^2+r_0^2}}}^{\,0} - \alpha\cancel{\frac{p_0 r_0}{\sqrt{p_0^2+q_0^2+r_0^2}}}^{\,0} + \Delta L_c$$

$$\Delta L_c = \frac{r}{\sqrt{2}}(\Delta f_1 + \Delta f_2 - \Delta f_3 - \Delta f_4)$$

$$\Delta L = \Delta L_c$$

$$\boxed{\Delta\dot p = \frac{\Delta L_c}{I_x}} \ (4)$$

$$\dot q = \frac{M - rp(I_x - I_z) - \cancel{I_{zx}}^{\,0}(p^2 - r^2)}{I_y} = \frac{M}{I_y} - rp\frac{(I_x-I_z)}{I_y} = h_5(\quad)$$

$$\Delta\dot q = \frac{\partial h_5}{\partial r}\Big|_0\Delta r + \frac{\partial h_5}{\partial p}\Big|_0\Delta p + \frac{\partial h_5}{\partial M}\Big|_0 \Delta M = -\Delta r\, \cancel{p_0}^{\,0} \frac{(I_x-I_z)}{I_y} - \cancel{r_0}^{\,0}\Delta p\frac{(I_x-I_z)}{I_y} + \frac{\Delta M}{I_y}$$

$$\Delta \dot q = \frac{\Delta M}{I_y} \ (5)$$

$$M = -\alpha\sqrt{p^2+q^2+r^2}\cdot q + M_c = g_5(\quad)$$

$$\Delta M = \frac{\partial g_5}{\partial p}\Big|_0 \Delta p + \frac{\partial g_5}{\partial q}\Big|_0 \Delta q + \frac{\partial g_5}{\partial r}\Big|_0 \Delta r + \frac{\partial g_5}{\partial M_c}\Big|_0 \Delta M_c$$

$$\Delta M = -\alpha\cancel{\frac{p_0 q_0}{\sqrt{p_0^2+q_0^2+r_0^2}}}^{\,0}\Delta p - \alpha \frac{\cancel{p_0^2}+2q_0^2+\cancel{r_0^2}}{\sqrt{\cancel{p_0^2}+q_0^2+\cancel{r_0^2}}}\Delta q - \alpha\cancel{\frac{q_0 r_0}{\sqrt{p_0^2+q_0^2+r_0^2}}}^{\,0}\Delta r + \Delta M_c$$

$$\Delta M = \Delta M_c$$

$$\boxed{\Delta\dot q = \frac{\Delta M_c}{I_y}} \ (5)$$

$$\dot{r} = \frac{N + \cancel{I_{zx}\dot{p}} - pq(I_y - I_x) + \cancel{I_{zx} qr}}{I_z} = \frac{N}{I_z} - \frac{pq(I_y - I_x)}{I_z} = h_6(\quad)$$

$$\Delta\dot{r} = \frac{\partial h_6}{\partial p}\Big|_0 \Delta p + \frac{\partial h_6}{\partial q}\Big|_0 \Delta q + \frac{\partial h_6}{\partial N}\Big|_0 \Delta N = -\cancel{\frac{\Delta p\, q_0 (I_y - I_x)}{I_z}} - \cancel{\frac{p_0 \Delta q (I_y - I_x)}{I_z}} + \frac{\Delta N}{I_z}$$

$$\Delta\dot{r} = \frac{\Delta N}{I_z} \quad (6)$$

$$N = -\alpha\sqrt{p^2 + q^2 + r^2}\, r + N_c = g_6(\quad)$$

$$\Delta N = \frac{\partial h_6}{p}\Big|_0 \Delta p + \frac{\partial h_6}{q}\Big|_0 \Delta q + \frac{\partial h_6}{\partial r}\Big|_0 \Delta r + \frac{\partial h_6}{\partial N_c}\Big|_0 \Delta N_c$$

$$\Delta N = -\alpha \cancel{\frac{p_0 r_0}{\sqrt{p_0^2 + q_0^2 + r_0^2}}} \Delta p - \alpha \cancel{\frac{q_0 r_0}{\sqrt{p_0^2 + q_0^2 + r_0^2}}} \Delta q - \alpha \cancel{\frac{p_0^2 + q_0^2 + 2r_0^2}{\sqrt{p_0^2 + q_0^2 + r_0^2}}} \Delta r + \Delta N_c$$

$$\Delta N = \Delta N_c$$

$$\boxed{\Delta\dot{r} = \frac{\Delta N_c}{I_z}} \quad (6)$$

$$\dot{\theta} = q\cos\phi - r\sin\phi = h_7(\quad)$$

$$\Delta\dot{\theta} = \frac{\partial h_7}{\partial q}\Big|_0 \Delta q + \frac{\partial h_7}{\partial \phi}\Big|_0 \Delta\phi + \frac{\partial h_7}{\partial r}\Big|_0 \Delta r = \cos\phi_0 \Delta q + \cancel{(q_0 \sin\phi_0 - r_0\cos\phi_0)\Delta\phi} - \cancel{\sin\phi_0 \Delta r}$$

$$\boxed{\Delta\dot{\theta} = \Delta q} \quad (7)$$

$$\dot{\phi} = p + (q\sin\phi + r\cos\phi)\tan\theta = h_8(\quad)$$

$$\Delta\dot{\phi} = \frac{\partial h_8}{\partial p}\Big|_0 \Delta p + \frac{\partial h_8}{\partial q}\Big|_0 \Delta q + \frac{\partial h_8}{\partial r}\Big|_0 \Delta r + \frac{\partial h_8}{\partial \phi}\Big|_0 \Delta\phi + \frac{\partial h_8}{\partial \theta}\Big|_0 \Delta\theta$$

$$\Delta\dot{\phi} = \Delta p + \cancel{\sin\phi_0 \tan\theta_0 \Delta q} + \cancel{\cos\phi_0 \tan\theta_0 \Delta r} + \cancel{(q_0\cos\phi_0 - r_0\sin\phi_0)\tan\theta\, \Delta\phi}$$
$$+ \cancel{(q_0 \sin\phi_0 + r_0\cos\phi_0)\sec^2\theta\, \Delta\theta}$$

$$\boxed{\Delta\dot{\phi} = \Delta p} \quad (8)$$

2. The results make some sense since hover is not a stable flight condition. Any small deviation in the flight conditions cause the aircraft to leave hover.

3. The linearized version seems very similar to the non-linearized results. There is a noticable difference in the rate changes since the rates in the linearized model don't have drag.

4. The control law makes the quad copter semi stable since it still deviats a little over time.

5. The control law doesn't do much in terms of stablizing the quadcopter. The copter starts oscillating for a few seconds before crashing.

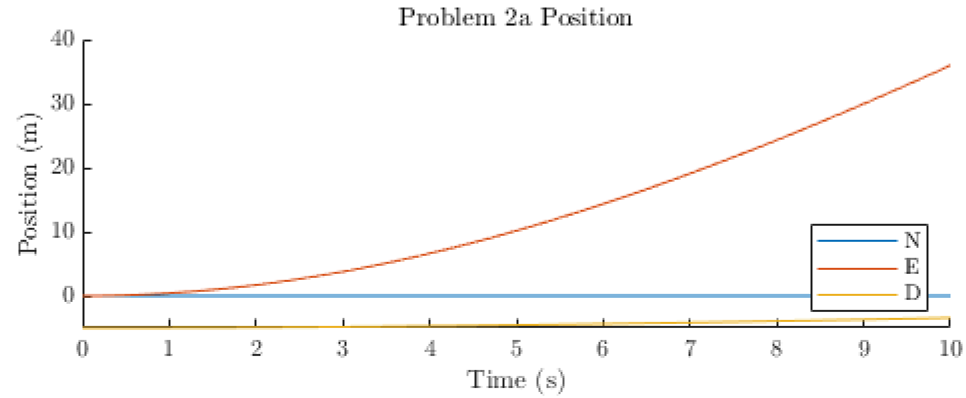
Table of Contents

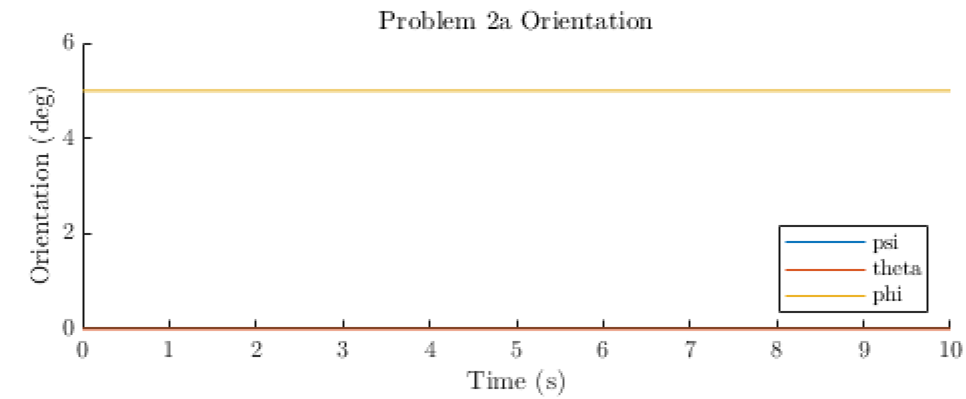

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% Author: Samuel Razumovskiy
% Date written: 9/20/19
% Date modified: 9/26/19
%
% Purpose: Modeling quadcopter flight and comparing three different
 models
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%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

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

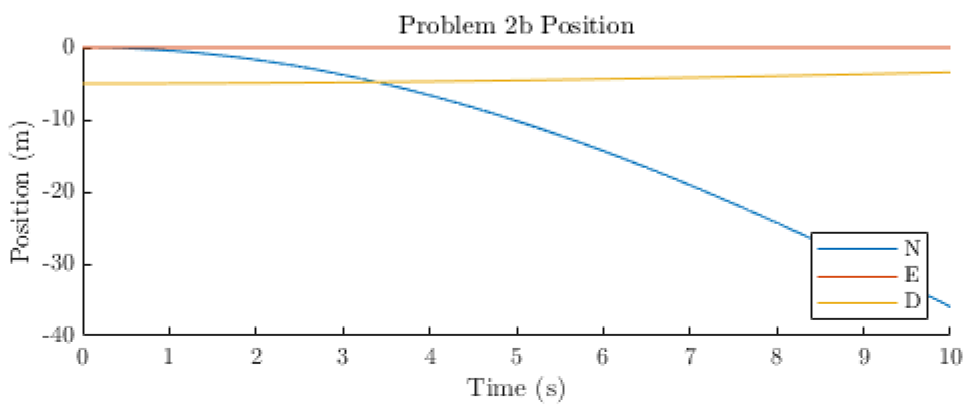
Problem 2

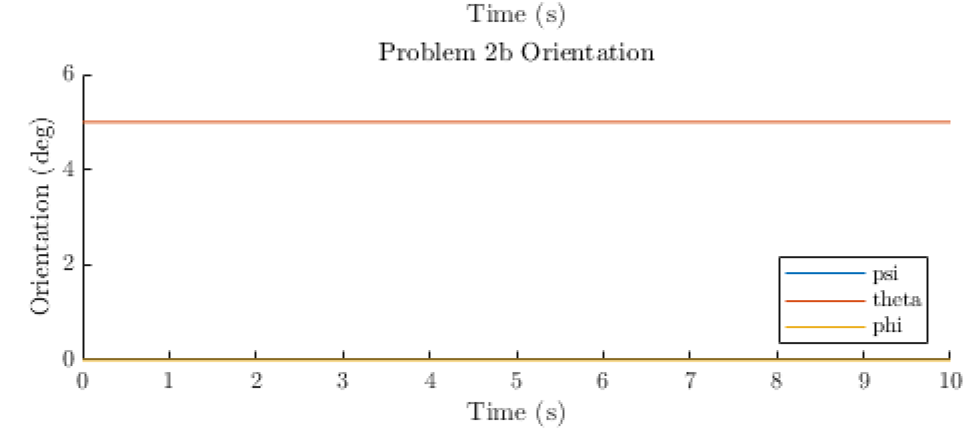
```
clear,clc,close all
dpsi = 5*pi/180;
dtheta = 5*pi/180;
dphi = 5*pi/180;
dp = 0.1;
dq = 0.1;
dr = 0.1;
letter = ['a','b','c','d','e','f'];
changes =
 [dphi,0,0,0,0,0;0,dtheta,0,0,0,0;0,0,dpsi,0,0,0;0,0,0,dp,0,0;...
    0,0,0,0,dq,0;0,0,0,0,0,dr];
[row,~] = size(changes);
for i = 1:row
    change = 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
```

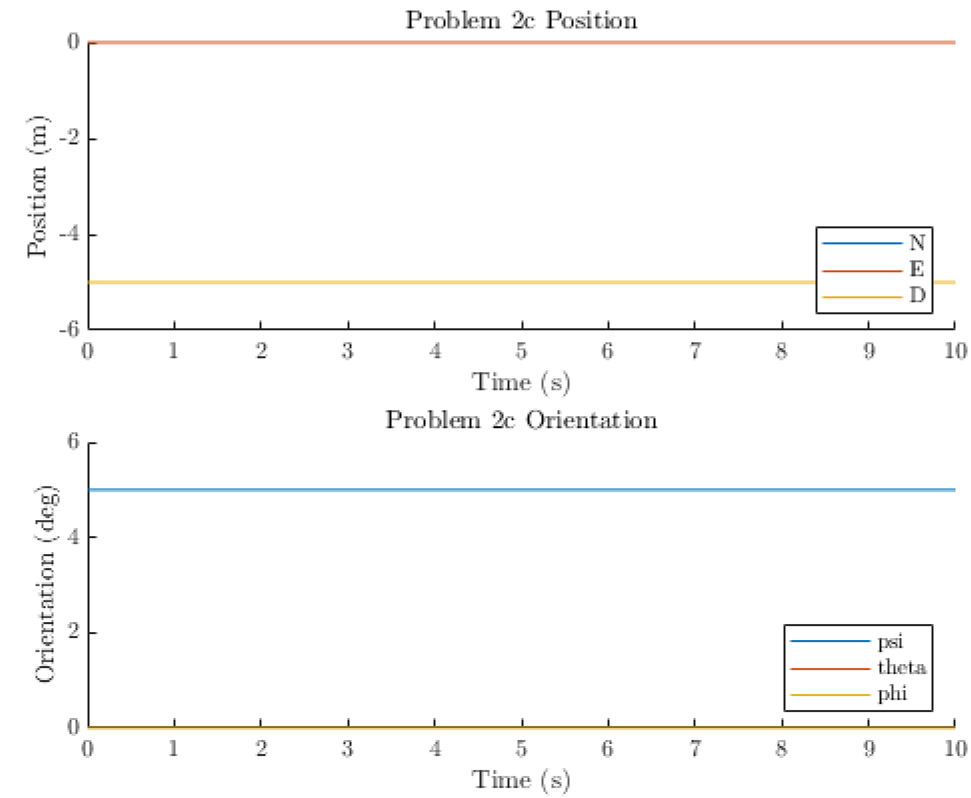
```
    tspan = [0 10];

    % 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
Position (m)
40
30
20
10
0
Time (s)
0 1 2 3 4 5 6 7 8 9 10
N
E
D

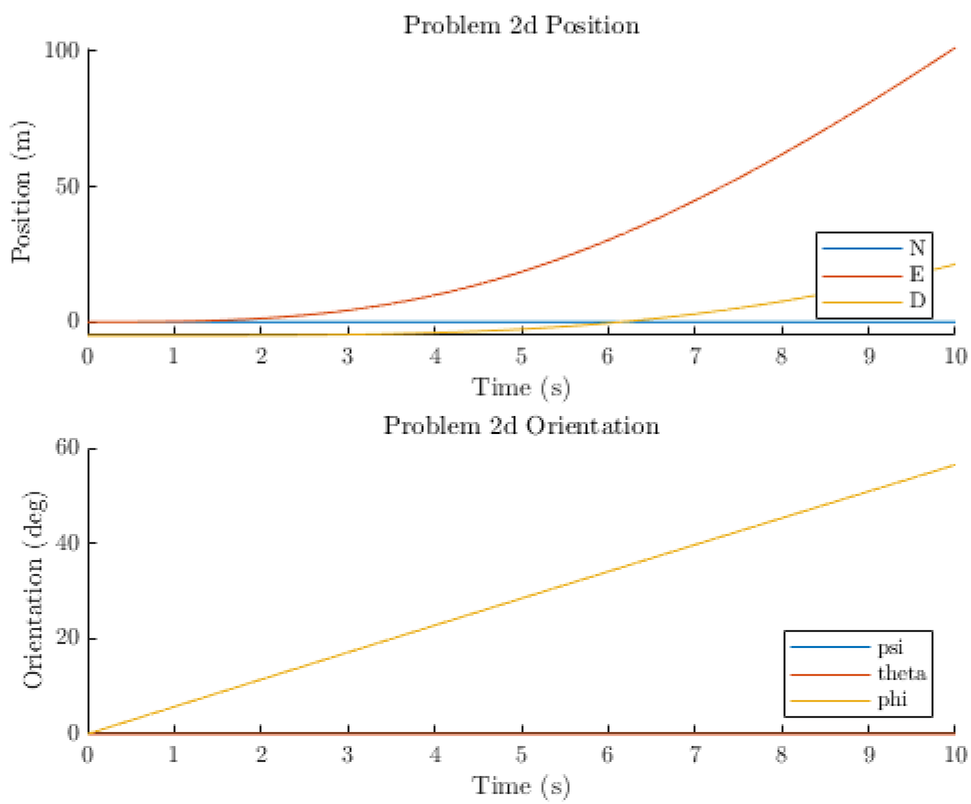

Problem 2a Orientation
Orientation (deg)
6
4
2
0
Time (s)
0 1 2 3 4 5 6 7 8 9 10
psi
theta
phi


Problem 2b Position
Position (m)
0
-10
-20
-30
-40
Time (s)
0 1 2 3 4 5 6 7 8 9 10
N
E
D

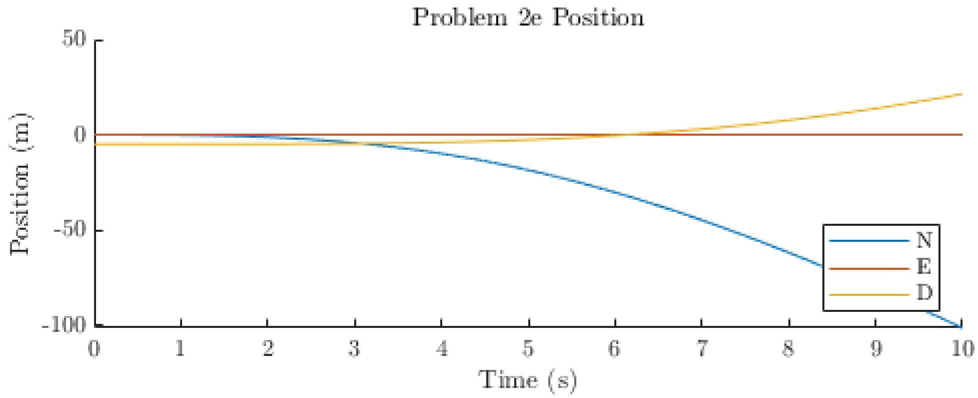

Problem 2b Orientation
Orientation (deg)
6
4
2
0
Time (s)
0 1 2 3 4 5 6 7 8 9 10
psi
theta
phi

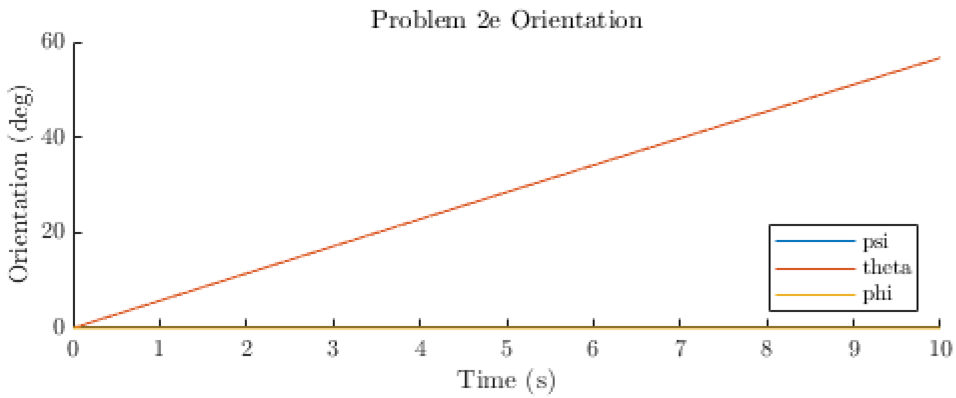


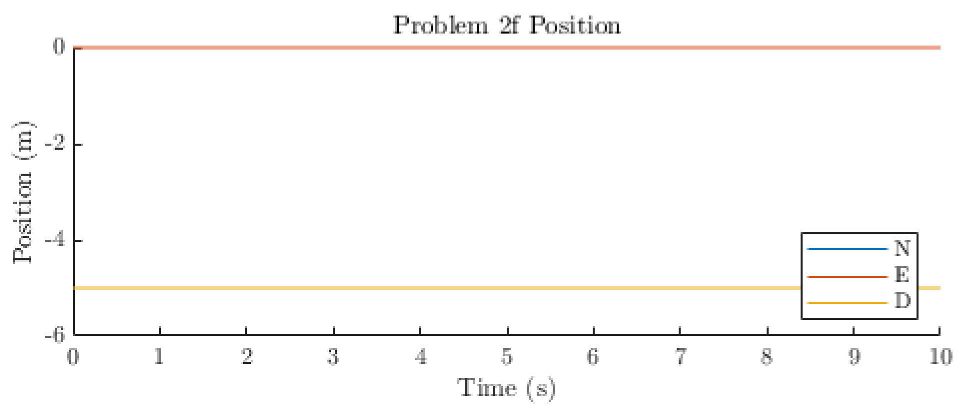
Problem 2c Position
Position (m)
Time (s)
N
E
D
Problem 2c Orientation
Orientation (deg)
Time (s)
psi
theta
phi

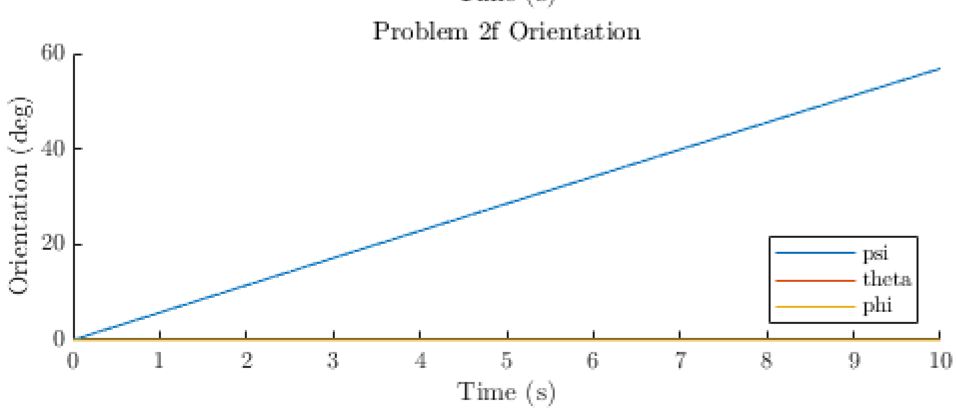


Problem 2d Position
Position (m)
Time (s)
N
E
D
Problem 2d Orientation
Orientation (deg)
Time (s)
psi
theta
phi


Problem 2e Position
Position (m)
50
0
-50
-100
0 1 2 3 4 5 6 7 8 9 10
Time (s)
N
E
D


Problem 2e Orientation
Orientation (deg)
60
40
20
0
0 1 2 3 4 5 6 7 8 9 10
Time (s)
psi
theta
phi

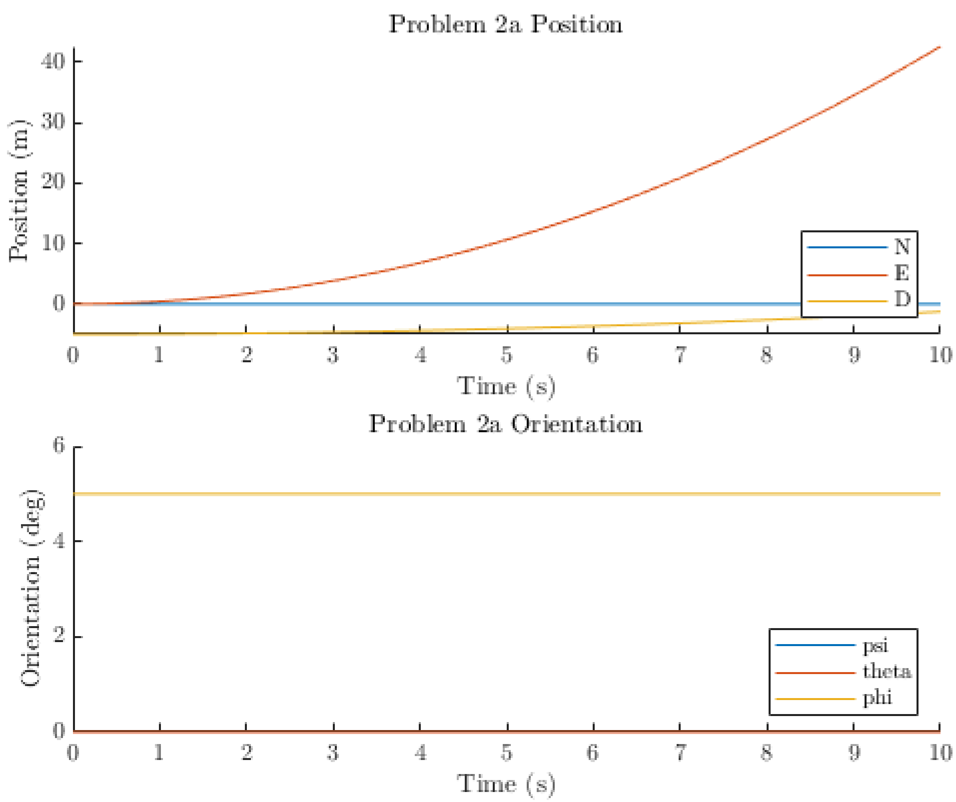

Problem 2f Position
Position (m)
0
-2
-4
-6
0 1 2 3 4 5 6 7 8 9 10
Time (s)
N
E
D

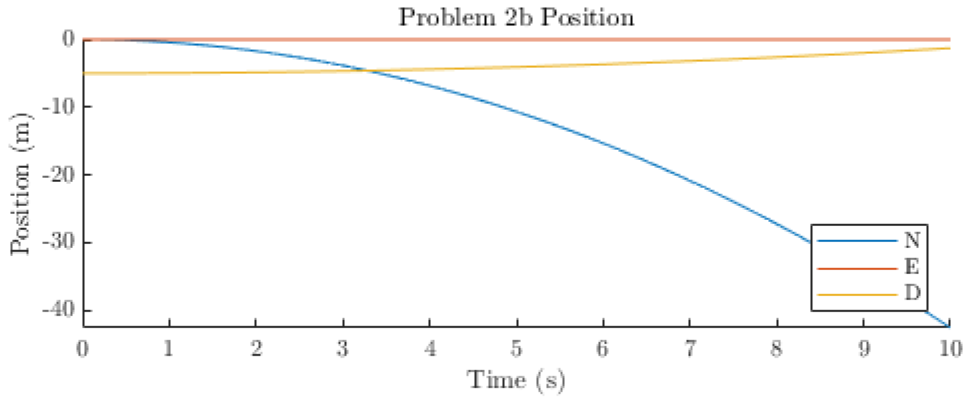

Problem 2f Orientation
Orientation (deg)
60
40
20
0
0 1 2 3 4 5 6 7 8 9 10
Time (s)
psi
theta
phi

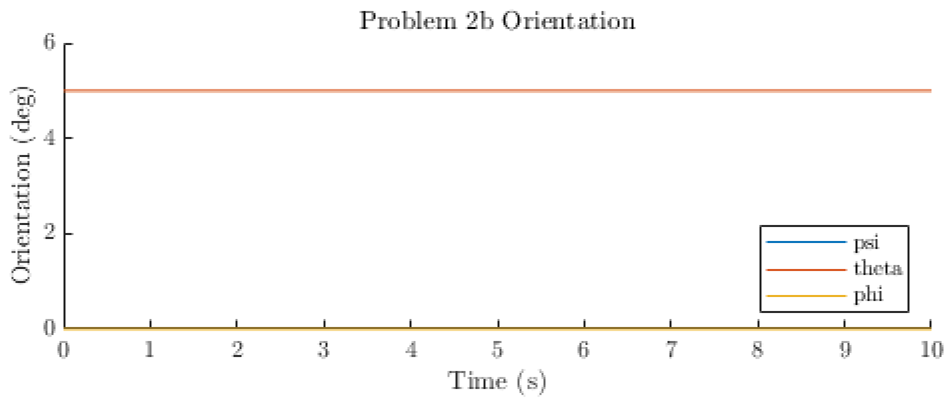
Problem 3

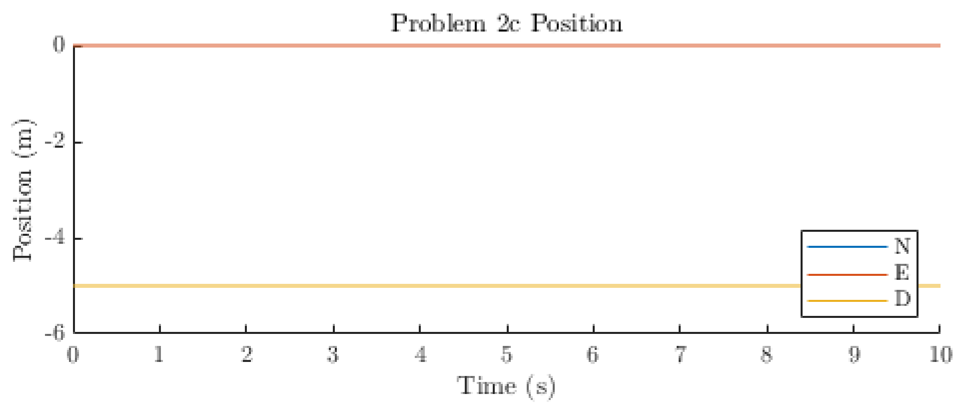
```
clear,clc,close all
dpsi = 5*pi/180;
dtheta = 5*pi/180;
dphi = 5*pi/180;
dp = 0.1;
dq = 0.1;
dr = 0.1;
letter = ['a','b','c','d','e','f'];
changes =
 [dphi,0,0,0,0,0;0,dtheta,0,0,0,0;0,0,dpsi,0,0,0;0,0,0,dp,0,0;...
    0,0,0,0,dq,0;0,0,0,0,0,dr];
[row,~] = size(changes);
for i = 1:row
    change = 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)')
```

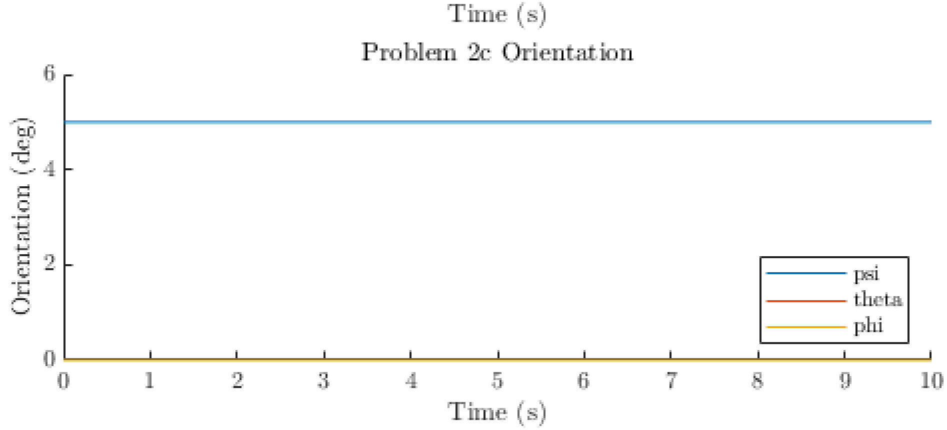
```
    title(sprintf("Problem 2%s Position",letter(i)))
    legend('N','E','D','location','southeast')
    subplot(2,1,2)
    hold on
    plot(t,pos(:,7).*180./pi)
    plot(t,pos(:,8).*180./pi)
    plot(t,pos(:,9).*180./pi)
    xlabel('Time (s)')
    ylabel('Orientation (deg)')
    title(sprintf("Problem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

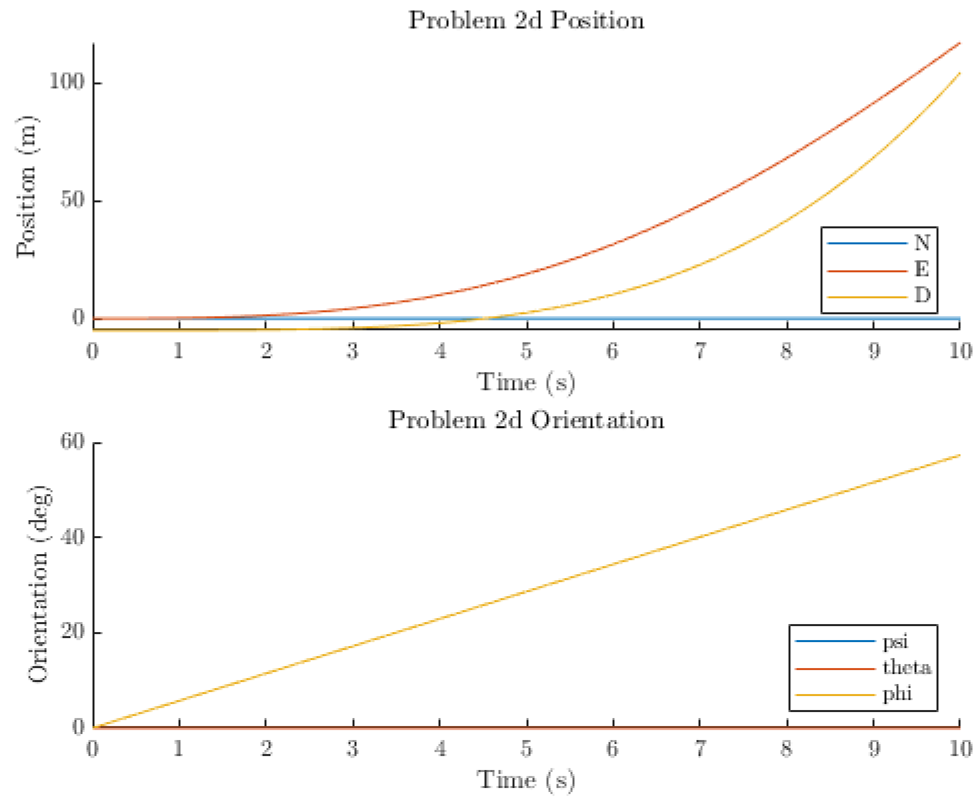


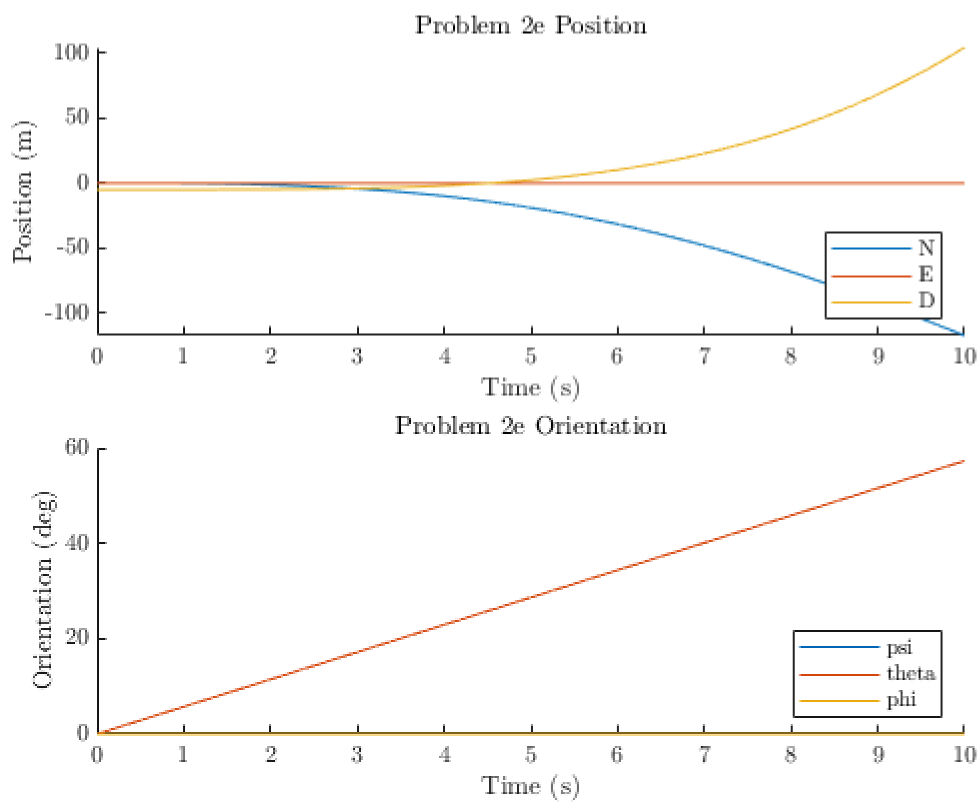

Problem 2b Position
Position (m)
Time (s)
N
E
D

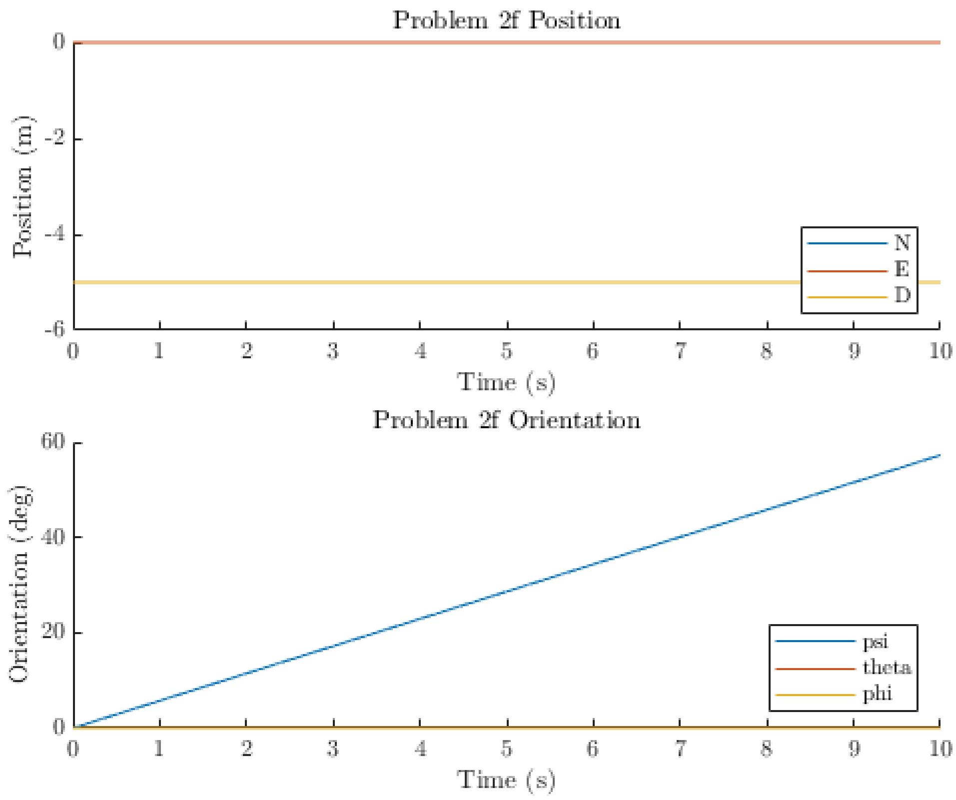

Problem 2b Orientation
Orientation (deg)
Time (s)
psi
theta
phi


Problem 2c Position
Position (m)
Time (s)
N
E
D


Problem 2c Orientation
Orientation (deg)
Time (s)
psi
theta
phi


Problem 2d Position
Position (m)
Time (s)
N
E
D
Problem 2d Orientation
Orientation (deg)
Time (s)
psi
theta
phi

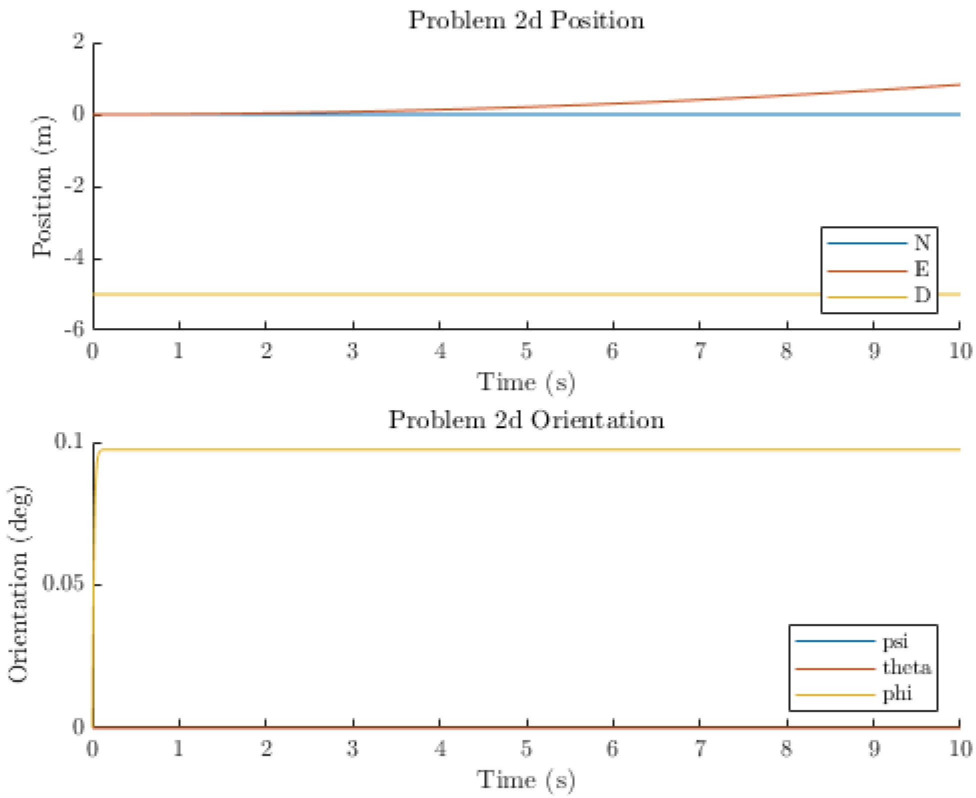

Problem 2e Position
Position (m)
Time (s)
N
E
D
Problem 2e Orientation
Orientation (deg)
Time (s)
psi
theta
phi

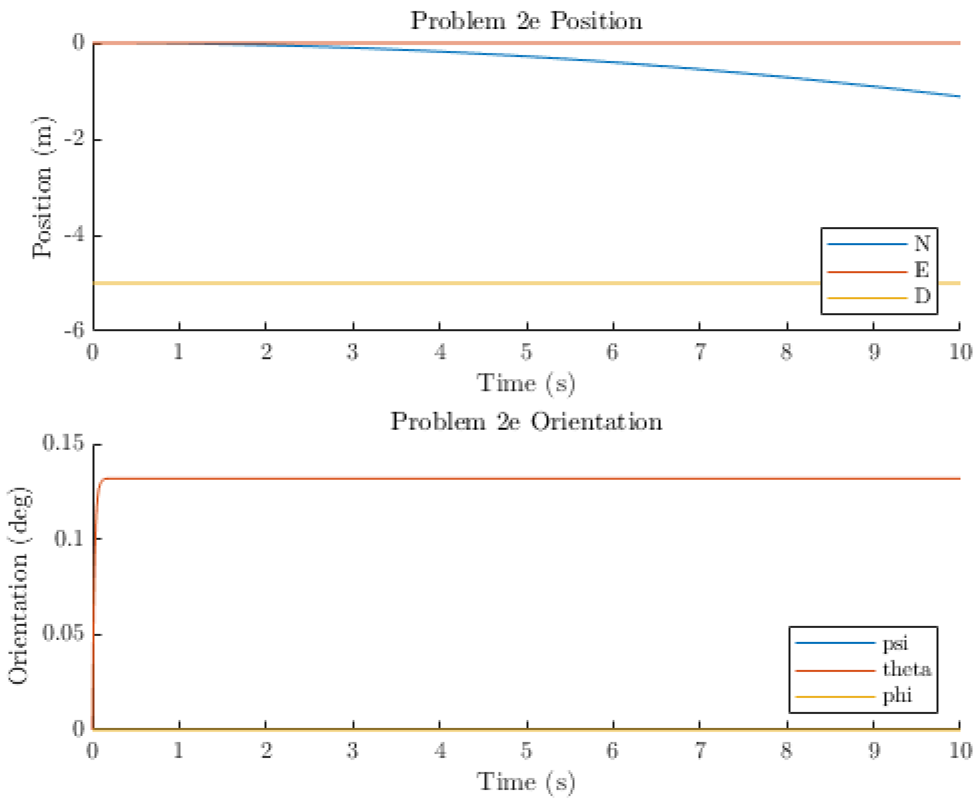


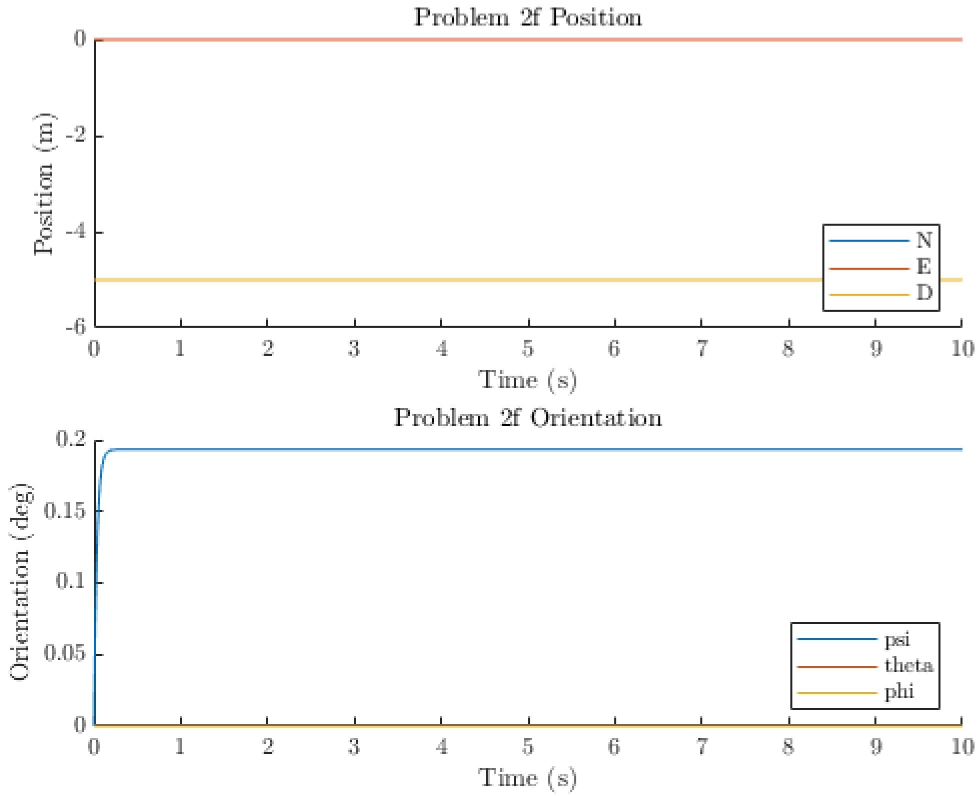
Problem 4

```
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];
```

```
    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
Position (m)
Time (s)
N
E
D
Problem 2d Orientation
Orientation (deg)
Time (s)
psi
theta
phi

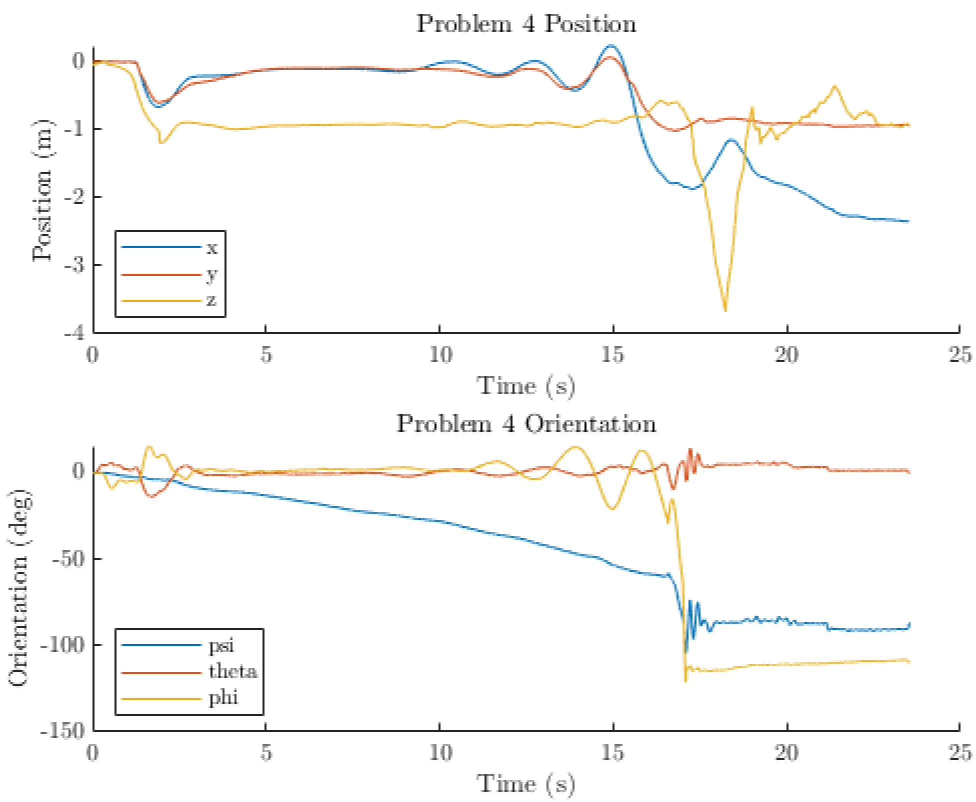

Problem 2e Position
Position (m)
Time (s)
N
E
D
Problem 2e Orientation
Orientation (deg)
Time (s)
psi
theta
phi



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)
```

```
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')
```



Functions Called

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

```
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];
```

```
% 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);
```

```
% 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);
```

```
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);
```

```
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