ASEN 3128 - Assignment 1

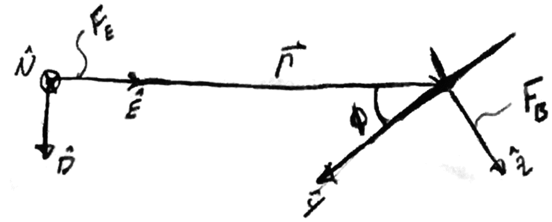
Seth Hill
Gregory Kondor
Connor Ott
Group 14 - Station 7b

January 29, 2018

University of Colorado - Boulder

Assignment 2 | 3128 | Connor Ott

Problem 1



ψ - Azimuth - rotation about $\hat{D}$, measured from $\hat{N}$

$\boxed{\psi = 180°}$

θ - Elevation - rotation about $\hat{E}$, $\boxed{\theta = 0°}$

ϕ - Bank - rotation about $\hat{N}$, $\boxed{\phi = -\phi°}$

Problem 2

$\dot{\psi}$ - rotation rate about $\hat{D}$

1 rotation in 1 period $\Rightarrow 360°/\tau$

$$\tau = \frac{2\pi}{0.1\ rad/s} = 62.83s$$

$$\dot{\psi} = \frac{360°}{62.83} = \boxed{5.73°/s = 0.1\ rad/s}$$

$\dot{\theta}$ - rotation rate about $\hat{E}$

$\boxed{\dot{\theta} = 0\ deg/s}$

$\dot{\phi}$ - rotation rate about $\hat{N}$

$$\dot{\phi} = \frac{d}{dt}\phi = \boxed{0\ deg/s = \dot{\phi}}$$

Problem 3

p - roll rate, the rate at which the plane rotates about its body-fixed $\hat{x}$ axis

$\boxed{p = 0\ deg/s}$

q - Pitch rate, the rate at which the plane rotates about its body-fixed $\hat{y}$ axis

$\boxed{\cancel{q = 0\ deg/s}}$

r - Yaw rate, the rate at which the plane rotates about its body-fixed $\hat{z}$ axis $\boxed{\cancel{r = 0\ deg/s}}$

Assignment 2 | 3128 | Connor Ott

Problem 4

$\dot{p}$ - $\hat{x}$ component of $\frac{d^B}{dt}\vec{\omega}^{EB} = \frac{d}{dt}(0) = 0 \ \text{rad}/s^2$

$\dot{q}$ - $\hat{y}$ component of $\frac{d^B}{dt}\vec{\omega}^{EB} = \frac{d}{dt}(0.1 \, s\sim) = 0 \ \text{rad}/s^2$

$\dot{r}$ - $\hat{z}$ component of $\frac{d^B}{dt}\vec{\omega}^{EB} = \frac{d}{dt}(0.1 \sim) = 0 \ \text{rad}/s^2$

Problem 5

$\vec{G} = \frac{d^E}{dt}\vec{h}^E$, the net moment about the c.o.m. of the aircraft expressed in Body frame coordinates

$$\vec{G} = \frac{d^E}{dt} I\vec{\omega}^{EB} = \bar{I}_B \frac{d}{dt}\vec{\omega}_B^{EB} + \vec{\omega}_B^{EB} \times \vec{h}_B^E$$

~~[illegible] angular velocity is constant~~

$$\vec{G} = \begin{bmatrix} (I_y - I_z)\,qr \\ -(x_z - I_x)\,\cancel{p}r \\ (I_x - I_y)\,\cancel{pq} \end{bmatrix} + \underbrace{\begin{bmatrix} \cancel{\dot{p}} \\ \cancel{\dot{q}} \\ \cancel{\dot{r}} \end{bmatrix}}_{\vec{0}}$$

$p = 0$

$$\vec{G} = -(I_y - I_z)qr$$

$$\boxed{\vec{G} = -0.1^2(I_y - I_z)\cos\phi\sin\phi\ \hat{x}}$$

- $\vec{G}$ is not fixed in the Inertial Frame since $\hat{x}$ is constantly changing direction
- $\vec{G}$ is fixed in the body frame since ϕ isn't changing

Assignment 2 | 3128 | Connor Ott

Problem 6.

$\vec{h}$ is the Inertial Angular momentum vector of the plane

$$\vec{h} = \bar{\bar{I}}\,\vec{\omega}^{\,EB}$$

$$= \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ 0.1 \end{bmatrix}$$

$$\boxed{\vec{h} = 0.1 I_z \hat{D}} = 0.1 I_z (\sin\phi\,\hat{y} + \cos\phi\,\hat{z})$$

$\vec{h}$ is not fixed in the inertial frame since $\vec{\omega}^{\,EB}$ is non-zero

$\vec{h}$ is fixed in the body frame since ϕ is not changing

Problem 7

Any value dependent on mass or moment of inertia would change

These would be #7 ($\vec{f}, \vec{f}_B, \vec{f}_E$) from assignment 1, + problems 4 ($\dot{p}, \dot{q}, \dot{r}$), 5 ($\vec{G}, \vec{G}_E, \vec{G}_B$) + 6 ($\vec{h}, \vec{h}_E, \vec{h}_B$)

Problem 8

Attitude will definitely induce translation on a quad copter since it will introduce some horizontal force

Translation does not necessarily cause changes in attitude, however

Problem 9

This problem asked for a simulation of a quad copter in steady, hovering flight. Figure 1 indicates that the simulation is working for level, steady hover.

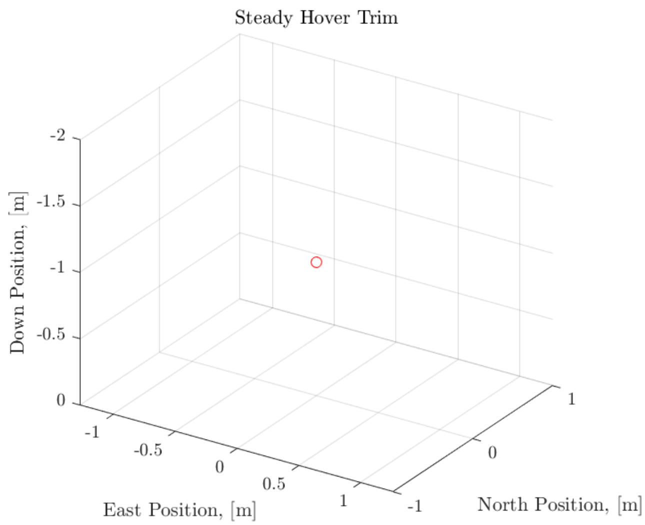


Figure 1: Steady hovering flight for quad copter.

Problem 10

This problem asked to determine trim forces and attitudes that would allow for steady flight travelling 5 m/s East. This was determined using a free body diagram and solving in MATLAB. The math is show in Figure 2. Figure 3 is a plot of the path of the quad copter.

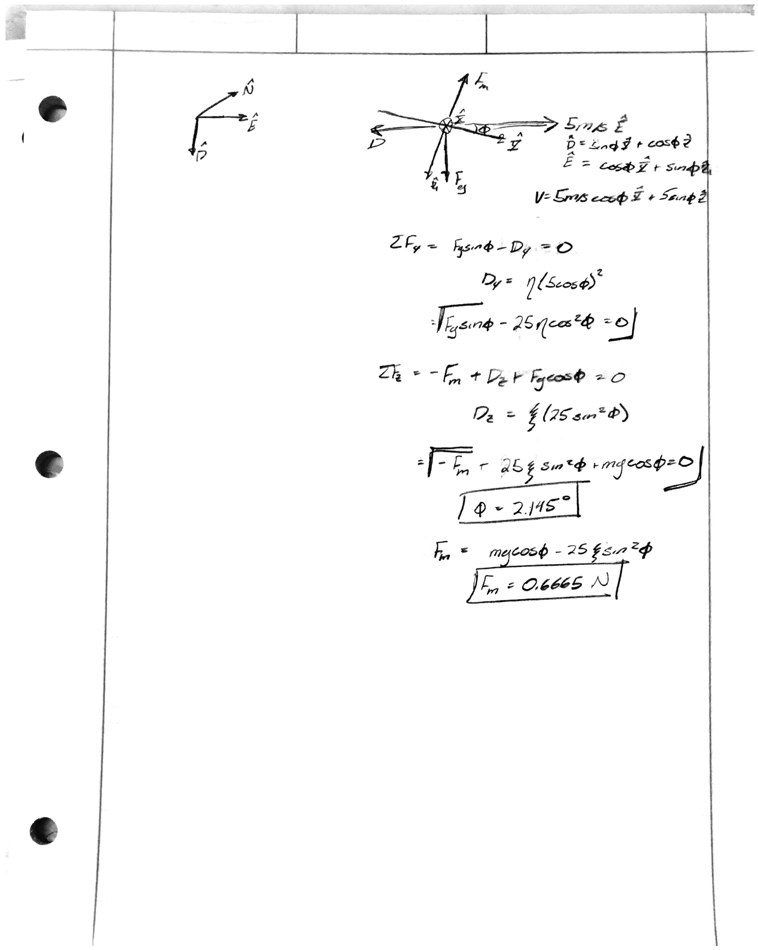


Figure 2: Math determining steady East trim forces and attitudes.

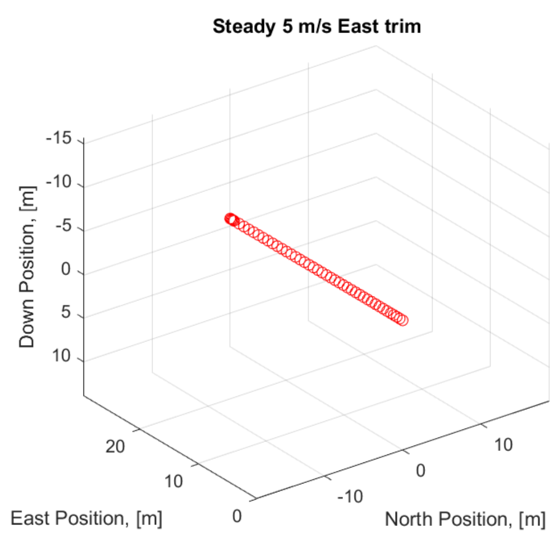


Figure 3: Steady East trim state.

For a 90 degree azimuth, the free body diagram and math remain mostly unchanged, the only difference being that the bank becomes a negative elevation angle. This case is shown in Figure 4.

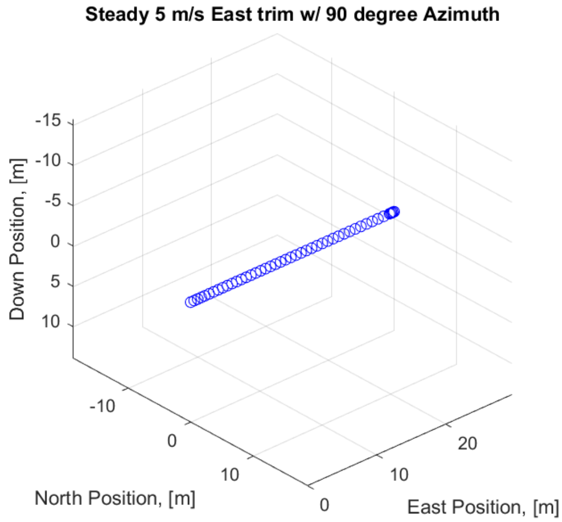


Figure 4: Steady East trim state w/ 90 degree azimuth.

Problem 11

This problem asked to determine if steady hovering flight is a stable state for the quad copter. It was seen to be unstable in both experiment and in simulation. In the simulation, a random moment was applied for 0.2 seconds on the steady hovering state, and the trajectory was plotted. This is shown in Figure 5. In experiment, the quad copter was set in a hover state and its feedback control was turned off. The experimental results are shown in Figure 6 the trajectory indicate the instability of the system without feedback control.

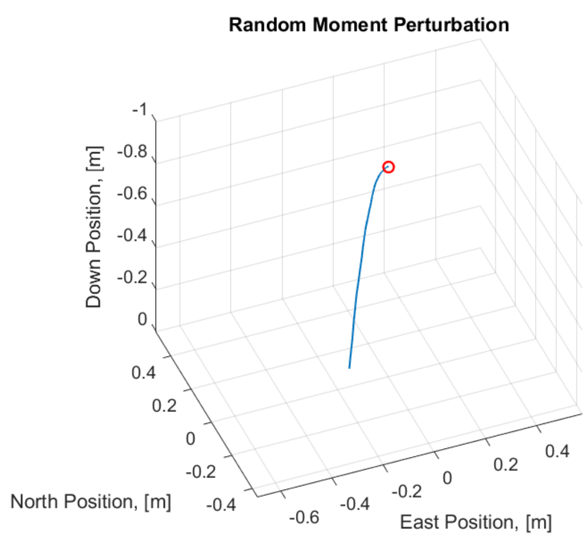


Figure 5: Simulation with random moment applied to it for 0.2 seconds.

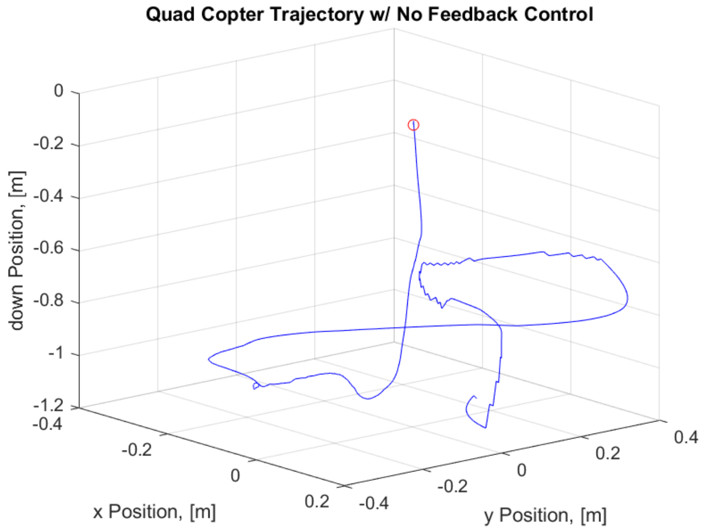


Figure 6: Experimental quad copter data.

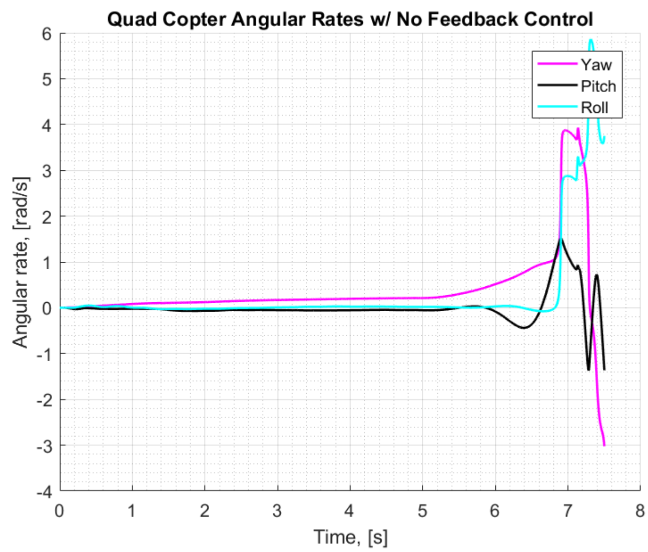


Figure 7: Experimental quad copter data.

MATLAB Code

```
%----------------------------------------------------------------------
% quadCopterSim simulates the dynamics of a small quad copter using
% numeical integration of Euler's Moment Equations
%
% Created: 1/30/18 - Connor Ott
% Last Modified: 1/30/18 - Connor Ott
%----------------------------------------------------------------------

clc; clear; close all;

%% Hover Trim
```

```matlab
ti = 0;
tf = 5; % s
momPert = zeros(1, 3); % No perturbation
analyticalTrim = ones(1, 4) * 0.068 * 9.81 / 4;
initConds = zeros(1, 12);
initConds(12) = -1;

options = odeset('Events', @termEvents, 'RelTol', 1e-8);
[t, F] = ode45(@(t, F)quadCopterODE(t, F, analyticalTrim, momPert), ...
                                                   [ti, tf], initConds, options);

figure
hold on; grid on; axis equal;
plot3(F(:, 10), F(:, 11), F(:, 12), 'ro')
set(gca, 'zdir', 'reverse')
zlabel('Down Position, [m]')
xlabel('East Position, [m]')
ylabel('North Position, [m]')
title('Steady Hover Trim')
set(gca, 'ticklabelinterpreter', 'latex');

%% 5 m/s East trim
ti = 0;
tf = 5; % [s]

momPert = zeros(1, 3); % No perturbation
V = 5;            % [m/s]
heta = 1e-3;      % [N/(m/s)^2]
xsi = 3e-3;       % [N/(m/s)^2]
m = 0.068;        % [kg]
g = 9.81;         % [m/s^2]
solveF = @(x) m*g*sin(x) - 25*heta*cos(x).^2;
phi = fzero(solveF, 0);
f_Mag = m*g*cos(phi) + 25*xsi*sin(phi).^2;

steadyTrim_East = ones(1, 4) * f_Mag / 4;
initConds = zeros(1, 12);
initConds(12) = -1;
initConds(2) = V*cos(phi);
initConds(3) = -V*sin(phi);
initConds(7) = phi;

options = odeset('Events', @termEvents, 'RelTol', 1e-8);
[t, F] = ode45(@(t, F)quadCopterODE(t, F, steadyTrim_East, momPert), ...
                                                   [ti, tf], initConds, options);

figure
hold on; grid on; axis equal;
plot3(F(:, 10), F(:, 11), F(:, 12), 'bo')
set(gca, 'zdir', 'reverse')
zlabel('Down Position, [m]')
ylabel('East Position, [m]')
xlabel('North Position, [m]')
title('Steady 5 m/s East trim w/ 90 degree Azimuth')

```

```matlab
%% Velocity Plots
figure
subplot(3, 1, 1)
hold on; grid on; grid minor;
plot(t, F(:, 1), 'b-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('North Velocity, [m/s]')

subplot(3, 1, 2)
hold on; grid on; grid minor;
plot(t, F(:, 2), 'b-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('East Velocity, [m/s]')

subplot(3, 1, 3)
hold on; grid on; grid minor;
plot(t, F(:, 3), 'b-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('Down Velocity, [m/s]')

%% Position Plots
figure
subplot(3, 1, 1)
hold on; grid on; grid minor;
plot(t, F(:, 10), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('North Position, [m/s]')

subplot(3, 1, 2)
hold on; grid on; grid minor;
plot(t, F(:, 11), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('East Position, [m/s]')

subplot(3, 1, 3)
hold on; grid on; grid minor;
plot(t, F(:, 12), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('Down Position, [m/s]')

%% Repeat of above with azimuth at 90 deg
% It travels north now, since the body frame has been rotated and the bank
% angle no longer causes motion in East direction.

ti = 0;
tf = 5; % [s]

momPert = zeros(1, 3); % No perturbation
u = 5;            % [m/s]
heta = 1e-3;      % [N/(rad/s)^2]
m = 0.068;        % [kg]
g = 9.81;         % [m/s^2]
theta = atan((heta^2 * u^2)/(m*g)); % [rad] - Bank
psi = pi/2;        % [rad] - Azimuth
f_Mag = sqrt((m*g)^2 + (heta^2*u^2)^2); %[N]
```

```
steadyTrim_East = ones(1, 4) * f_Mag / 4;
initConds = zeros(1, 12);
initConds(12) = -1;
initConds(1) = u;
initConds(8) = theta;
initConds(9) = psi;

options = odeset('Events', @termEvents, 'RelTol', 1e-8);
[t, F] = ode45(@(t, F)quadCopterODE(t, F, steadyTrim_East, momPert), ...
                                                [ti, tf], initConds, options);


figure
hold on; grid on; axis equal;
plot3(F(:, 10), F(:, 11), F(:, 12), 'ro')
set(gca, 'zdir', 'reverse')
zlabel('Down Position, [m]')
xlabel('East Position, [m]')
ylabel('North Position, [m]')
title('Steady 5 m/s East trim - 90 deg azimuth')

%% Position Plots
figure
subplot(3, 1, 1)
hold on; grid on; grid minor;
plot(t, F(:, 10), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('East Position, [m/s]')

subplot(3, 1, 2)
hold on; grid on; grid minor;
plot(t, F(:, 11), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('North Position, [m/s]')
axis([0, 5, -0.5, 0.5])

subplot(3, 1, 3)
hold on; grid on; grid minor;
plot(t, F(:, 12), 'm-', 'linewidth', 1.2)
xlabel('Time, [s]')
ylabel('Down Position, [m/s]')
axis([0, 5, -1.5, -0.5])

%% Simulate a perturbation in steady hover to see how quad copter reacts

ti = 0;
tf = 5; % s
analyticalTrim = ones(1, 4) * 0.068 * 9.81 / 4;
initConds = zeros(1, 12);
initConds(12) = -1;

figure
hold on; grid on; axis equal;
set(gca, 'zdir', 'reverse')
```

```
zlabel('Down Position, [m]')
xlabel('East Position, [m]')
ylabel('North Position, [m]')
title('Random Moment Perturbation')
plot3(0, 0, -1, 'ro', 'linewidth', 1.2)
momPert = randn(3, 1)./10;
options = odeset('Events', @termEvents, 'RelTol', 1e-8);
[t, F] = ode45(@(t, F)quadCopterODE(t, F, analyticalTrim, momPert), ...
                                          [ti, tf], initConds, options);

plot3(F(:, 10), F(:, 11), F(:, 12), '-', 'linewidth', 1.1)

%% Plotting experimental quad copter data
load('RSdata_Drone01_1330.mat');
times = rt_estim.time(:);
xdata = rt_estim.signals.values(:, 1);
ydata = rt_estim.signals.values(:, 2);
zdata = rt_estim.signals.values(:, 3);

figure
hold on; grid on;
plot3(xdata(1), ydata(1), zdata(1), 'ro')
plot3(xdata, ydata, zdata, 'b-')
title('Quad Copter Trajectory w/ No Feedback Control')
xlabel('x Position, [m]')
ylabel('y Position, [m]')
zlabel('down Position, [m]')
```

```
function dfdt = quadCopterODE(t, F, trim, momPert)
% Defines dynamics of quad copter


%% Physical properties and constants
alpha = 2e-6;   % [N/(m/s)^2]
beta = 1e-6;    % [N/(m/s)^2]
heta = 1e-3;    % [N/(rad/s)^2]
xsi = 3e-3;     % [N/(rad/s)^2]

I_x = 6.8e-5;   % [kg m^2]
I_y = 9.2e-5;   % ['']
I_z = 1.35e-4;  % ['']

m = 0.068;      % [kg]
d = 0.06;       % [m]
k = 0.0024;     % [~]
rad = d/sqrt(2);  % [m]
g = 9.81;       % [m/s^2]

%% Pulling from input, F and trim
% Inertial velocity in body coords
u = F(1);
v = F(2);
w = F(3);

% Inertial angular velocity in body coords
p = F(4);
```

```
q = F(5);
r = F(6);

% Euler angles
phi = F(7);
theta = F(8);
psi = F(9);

% Position vector from origin to COM in inertial coords
x_E = F(10);
y_E = F(11);
z_e = F(12);

f1 = trim(1); % motor forces
f2 = trim(2);
f3 = trim(3);
f4 = trim(4);

%% Aerodynamic and Control Moments and Forces

% Moments
L_a = - alpha^2 * p^2 * sign(p);
M_a = - alpha^2 * q^2 * sign(q);
N_a = - beta^2 * r^2 * sign(r);

L_c = ((f2 + f3) - (f1 + f4)) * rad;
M_c = ((f3 + f4) - (f1 + f2)) * rad;
N_c = (f2 + f4 - (f1 + f2)) * k;

L = L_a + L_c;
M = M_a + M_c;
N = N_a + N_c;

if t > 1 && t < 1.2 % Throw a moment perturbation in for 0.2 seconds
    L = L + momPert(1);
    M = M + momPert(2);
    N = N + momPert(3);
end

% Forces
X_a = - heta^2 * u^2 * sign(u);
Y_a = - heta^2 * v^2 * sign(v);
Z_a = - xsi^2 * w^2 * sign(w);

X_c = 0;
Y_c = 0;
Z_c = -sum(trim);

X = X_a + X_c;
Y = Y_a + Y_c;
Z = Z_a + Z_c;

% Determining Roll, Pitch, and Yaw rates of change
p_dot = (I_y - I_z)/I_x * q * r + 1/I_x * L;
q_dot = (I_z - I_x)/I_y * p * r + 1/I_y * M;
```

```
r_dot = (I_x - I_y)/I_z * p * q + 1/I_z * N;
dOmega_bdt = [p_dot, q_dot, r_dot]';

% Determining translation rates
u_dot = r*v - q*w - g*sin(theta) + 1/m * X;
v_dot = p*w - r*u + g*sin(phi)*cos(theta) + 1/m * Y;
w_dot = q*u - p*v + g*cos(phi)*cos(theta) + 1/m * Z;
dV_bdt = [u_dot, v_dot, w_dot]';

% Tranlating body to inertial coordinates
x_dot =  u * cos(theta)*cos(psi) + ...
         v * (sin(phi)*sin(theta)*cos(psi) - cos(phi)*sin(psi)) + ...
         w * (cos(phi)*sin(theta)*cos(psi) + sin(phi)*sin(psi));
y_dot =  u * cos(theta)*sin(psi) + ...
         v * (sin(phi)*sin(theta)*sin(psi) + cos(phi)*cos(psi)) + ...
         w * (cos(phi)*sin(theta)*sin(psi) - sin(phi)*cos(psi));
z_dot = -u * sin(theta) + ...
         v * sin(phi)*cos(theta) + ...
         w * cos(phi)*cos(theta);
dV_Edt = [x_dot, y_dot, z_dot]';

% Translating Angular Momentum to Euler angles
phi_dot   = p + (q*sin(phi)+ r*cos(phi))*tan(theta);
theta_dot = q*cos(phi) - r*sin(phi);
psi_dot   = q*sin(phi)*sec(theta) + r*cos(phi)*sec(theta);
dEuldt = [phi_dot, theta_dot, psi_dot]';

% Concatenating for output
dfdt = [dV_bdt; dOmega_bdt; dEuldt; dV_Edt];



end
```

```
function [value, isTerm, direction] = termEvents(t, F)
% Define events for quad copter ODE
value = F(12);
isTerm = 1;
direction = [];
end
```