Table of Contents

Roshan Jaiswal-Ferri & Stefan Rosu

```
%Section - 01
%Aero 421 FP6: 5/23/25
```

Workspace Prep

```
%warning off
format long     %Allows for more accurate decimals
close all;      %Clears all
clear all;      %Clears Workspace
clc;            %Clears Command Window
```

Note

```
% We could not find on canvas where the satellite was supposed to point so
% we just assumed nadir and converted from lvlh to eci each time step,
% which is why it might look different.
```

Mass properties for normal operations phase

```
zbar = 0.23438;
cm = [0; 0; zbar];
mw = 1;
total_mass = 640;
total_mass = total_mass + 3*mw;
J = [812.0396 0 0
0 545.3729 0
0 0 627.7083];
eps = [0;sqrt(0.5);0];
eta = sqrt(0.5);
q_c = [eps; eta];
I_w1 = diag([0.6, 0.6 ,1.2]); % z-axis
I_w2 = diag([0.6, 1.2 , 0.6]); % y-axis
I_w3 = diag([1.2, 0.6 ,0.6]); % x-axis

r_w1 = [0,0,10];
r_w2 = [0,10,0];
r_w3 = [10,0,0];
```

```
I_w1 = I_w1 - mw*vcross(r_w1)*vcross(r_w1);
I_w2 = I_w2 - mw*vcross(r_w2)*vcross(r_w2);
I_w3 = I_w3 - mw*vcross(r_w3)*vcross(r_w3);
I_s = diag([1.2, 1.2, 1.2]);
J = J + I_w1 + I_w2 + I_w3;
ts = 100; % settling time
zeta = 0.65; % Damping ratio
wn = log(0.02*sqrt(1-zeta^2))/-zeta/ts;
beta = atan(sqrt(1-zeta^2)/zeta);
tr = (pi-beta)/wn/sqrt(1-zeta^2);
syms Mp1
eqn = zeta == sqrt(log(Mp1)^2/(pi^2 + log(Mp1)^2));
Mp = double(solve(eqn, Mp1));
Kp = 2*J*eye(3)*wn^2;
Kd = J*eye(3)*2*zeta*wn;


% Spacecraft Orbit Properties (given)
mu = 398600; % km^3/s^2
h = 53335.2; % km^2/s
e = 0; % none
Omega = 0*pi/180; % radians
inc = 98.43*pi/180; % radians
omega = 0*pi/180; % radians
nu = 0*pi/180; % radians

a = h^2/mu/(1 - e^2);
orbital_period = 2*pi*sqrt(a^3/mu);

% Torque free scenario (Given)
T = [0;0;0];

% Set/Compute initial conditions
% intial orbital position and velocity
[r_ECI_0, v_ECI_0] = coes2rvd(a,e,rad2deg(inc),0,omega,nu,mu);

% Compute inital F_LVLH basis vectors in F_ECI components based on F_LVLH
% definition

rV = r_ECI_0; %Position Vector km
vV = v_ECI_0; %Vel Vector km/s

Zlvlh = -(rV/norm(rV));
Ylvlh = -(cross(rV,vV)/norm(cross(rV,vV)));
Xlvlh = cross(Ylvlh,Zlvlh);

%Creating Matrix with new vectors

C_LVLH_ECI_0 = [Xlvlh, Ylvlh, Zlvlh];

% Initial Euler angles relating F_body and F_LVLH (given)
phi_0 = 0;
theta_0 = 0;
psi_0 = 0;
```

```
E_b_LVLH_0 = [phi_0; theta_0; psi_0];

% Initial Quaternion relating F_body and F_LVLH (given)
q_b_LVLH_0 = [0; 0; 0; 1];

% Compute initial C_LVLH_ECI_0, C_b_LHVL_0, and C_b_ECI_0 rotaiton matrices
%C_LVLH_ECI_0 = [x_LVLH'; y_LVLH'; z_LVLH'];
C_b_LVLH_0 =
rotx(rad2deg(phi_0))'*roty(rad2deg(theta_0))'*rotz(rad2deg(psi_0))';
C_b_ECI_0 = C_b_LVLH_0*C_LVLH_ECI_0;

% Initial Euler angles relating body to ECI
E_b_ECI_0 = C2EulerAngles(C_b_ECI_0);

% Initial quaternion relating body to E
q_b_ECI_0 = rotm2quat(C_b_ECI_0);

% Initial body rates of spacecraft (given)
w_b_ECI_0 = [0.001; -0.001; 0.002];

% Set simulation time period
N = 5; % Number of Orbits
%tspan = orbital_period*N;
%tspan = 300;


for i = 1:2

    if i == 2
        tspan = orbital_period*N;
    else
        tspan = 300;
    end

    % Simulate!
    out = sim('proj6.slx');
```

```
Error using AERO421_FP6_RJFSR (line 127)
Model
'proj6.slx' does
not exist.

Error in evalmxdom>instrumentAndRun (line 116)
text = evalc(evalstr);
^^^^^^^^^^^^^^^^^^^^^^
Error in evalmxdom (line 21)
[data,text,laste] =
instrumentAndRun(file,cellBoundaries,imageDir,imagePrefix,options);
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^
```

Plot Results

```
figure('Name','Body to ECI')
subplot(3,1,1)
plot(out.w_b_ECI(:,1), out.w_b_ECI(:,2:4))
title('Angular Velocities')
ylabel('angular velocity (rad/sec)')
legend('\omega_x','\omega_y','\omega_z')
grid on

subplot(3,1,2)
plot(out.tout, out.q_b_ECI(:,2:5))
title('Quaternions')
ylabel('Quaternion Parameter')
legend('q_1','q_2','q_3','q_4')
grid on

subplot(3,1,3)
plot(out.tout, out.E_b_ECI(:,2:4))
title('Euler Angles')
xlabel('time (seconds)')
ylabel('Angle (rad)')
legend('\phi','\theta','\psi')
grid on

sgtitle('Body to ECI Dynamics and Kinematics')

q_b_LVLH = squeeze(out.q_b_LVLH.signals.values);
E_b_LVLH = squeeze(out.E_b_LVLH.signals.values);
w_b_LVLH = squeeze(out.w_b_LVLH.signals.values);


figure('Name','Body to LVLH')
subplot(3,1,1)
plot(out.tout, w_b_LVLH)
title('Angular Velocities')
ylabel('angular velocity (rad/sec)')
legend('\omega_x','\omega_y','\omega_z')
grid on

subplot(3,1,2)
plot(out.tout, q_b_LVLH)
title('Quaternions')
ylabel('Quaternion Parameter')
legend('q_1','q_2','q_3','q_4')
grid on

subplot(3,1,3)
plot(out.tout, E_b_LVLH)
title('Euler Angles')
xlabel('time (seconds)')
ylabel('Angle (rad)')
legend('\phi','\theta','\psi')
```

```
    grid on

    sgtitle('Body to LVLH Dynamics and Kinematics')

    wheelz = squeeze(out.RWheel.signals.values);
    torque = squeeze(out.M_c.signals.values);

    figure('Name','Reaction Wheels')
    subplot(2,1,1)
    plot(out.tout(:,1), wheelz)
    title('Angular Velocities')
    ylabel('angular velocity (rad/sec)')
    legend('\omega_x','\omega_y','\omega_z')
    grid on

    subplot(2,1,2)
    plot(out.tout(:,1), torque)
    title('Commanded Torque ')
    ylabel('Torque (N/m)')
    legend('T_x','T_y','T_z')
    grid on

end
```

Published with MATLAB® R2024b