Table of Contents

Assignment 2

```
%Aero 560
%Morgan Yost
```

Set up for simulink

```
%clear all
ecc = 0;
a = 42000; %km
inc = 0; %degrees
raan = 0; %degrees
omega = 0; %degrees
theta =  180;
mu = 398600; %km^2 something
[rvect,vvect]=COES2rvd(a,ecc,inc,raan, omega, theta);
x = [1 0 0];
y = [0 1 0];
z = [0 0 1];
h_a = cross(rvect, vvect);
k_hat = -rvect/norm(rvect);
j_hat = -h_a/norm(h_a);
i_hat = cross(j_hat, k_hat);
Q= [i_hat j_hat k_hat]; %inertial to lvlh
DCM = [dot(x, i_hat) dot(x, j_hat) dot(x, k_hat);
       dot(y, i_hat) dot(y, j_hat) dot(y, k_hat);
       dot(z, i_hat) dot(z, j_hat) dot(z, k_hat)];
w =[.5e-7; 7.27e-7; 3e-7]; %rad/sec in body frame
wbn0 = DCM*w; %body to inertial assuming body is aligned with lvlh to
 start
%initial euler angles in ECI assuming s/c starts aligned with LVLH
J = [3500 0 0; 0 3200 0; 0 0 2500]; %kg m^2
Jinv = inv(J);
T = 2*pi*sqrt(a^3/mu); %sec
tprop = T; %one period
q = DCM2quat(DCM);
eta0 = q(1);
epsilon0 = q(2:4);
ts = 30; %s
zeta  = .65;
nf = -log(.02)/(ts*zeta);
kd = J*2*zeta*nf;
kp = J*2*nf^2;
n= 1/T;
```

Run Simulink

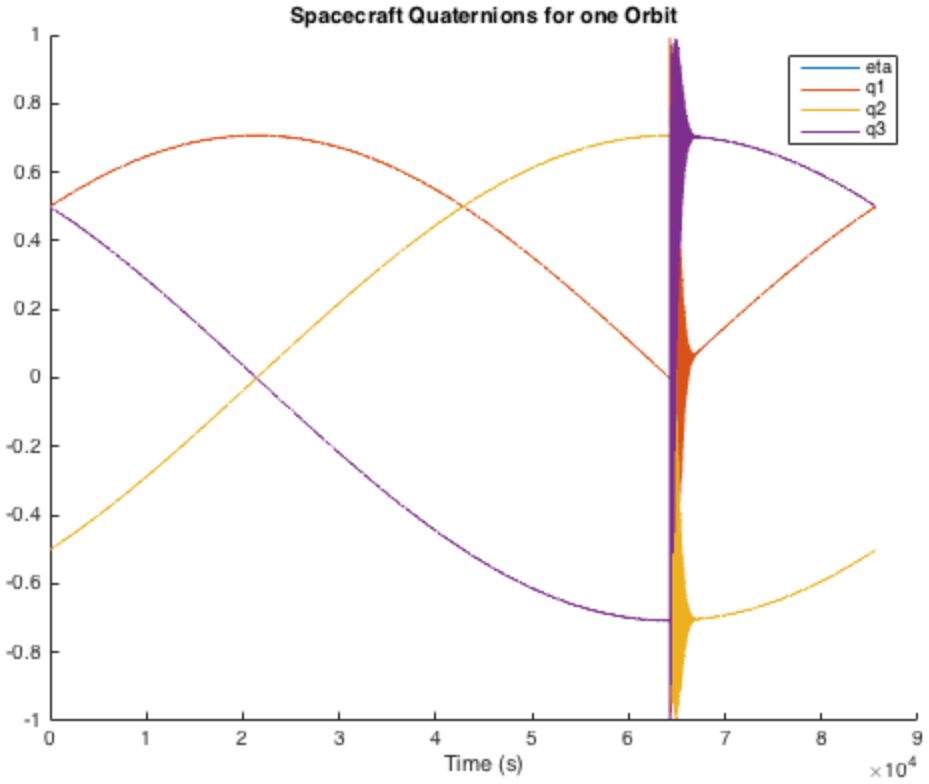
```
sim('satPropWThrusterControl');
```

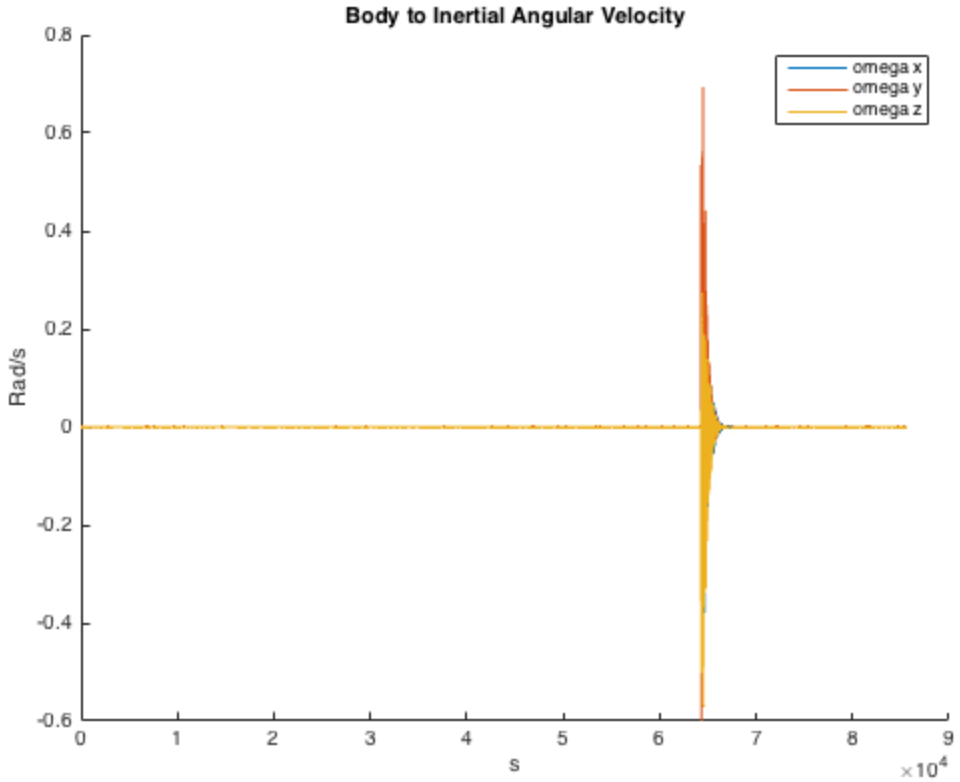
Make Plots

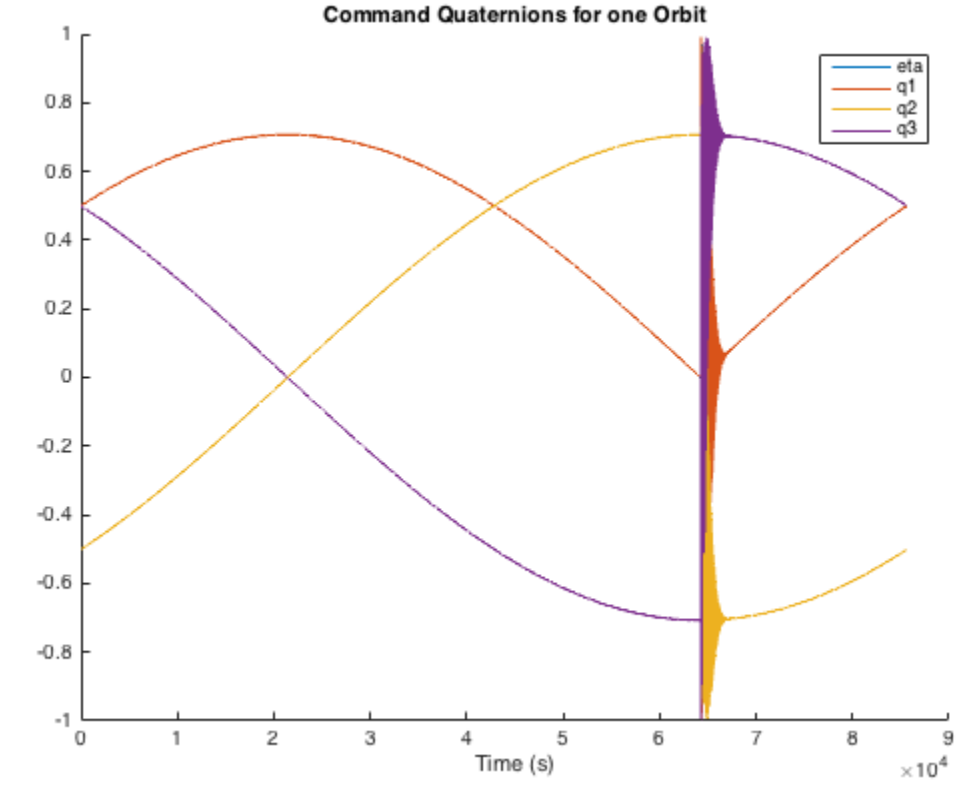
```
close all
figure(1)
hold on
plot(quat.time,quat.data(:, 1))
plot(quat.time,quat.data(:, 2))
plot(quat.time,quat.data(:, 3))
plot(quat.time,quat.data(:, 4))
legend('eta', 'q1', 'q2', 'q3')
title('Spacecraft Quaternions for one Orbit')
xlabel('Time (s)')

figure(2)
hold on
plot(wbn.time, wbn.Data(:,1))
plot(wbn.time, wbn.Data(:,2))
plot(wbn.time, wbn.Data(:,3))
ylabel('Rad/s')
xlabel('s')
title('Body to Inertial Angular Velocity')
legend('omega x', 'omega y', 'omega z')

figure(3)
hold on
plot(q_command.time,quat.data(:, 1))
plot(q_command.time,quat.data(:, 2))
plot(q_command.time,quat.data(:, 3))
plot(q_command.time,quat.data(:, 4))
legend('eta', 'q1', 'q2', 'q3')
title('Command Quaternions for one Orbit')
xlabel('Time (s)')

figure(4)
hold on
plot(wbn.time, wbn.Data(:,1))
plot(wbn.time, wbn.Data(:,2))
plot(wbn.time, wbn.Data(:,3))
ylabel('Nm')
xlabel('s')
title('Commanded torque')
legend('X', 'y', 'omega z')

len = length(Tc.time)/2;
fprintf('Max torque on X axis %f \n', max(Tc.data(:,1)))
fprintf('Max torque on Y axis %f \n', max(Tc.data(:,2)))
fprintf('Max torque on Z axis %f \n', max(Tc.data(:,3)))
fprintf('When we ignore the 360 deg swivel\n')
```

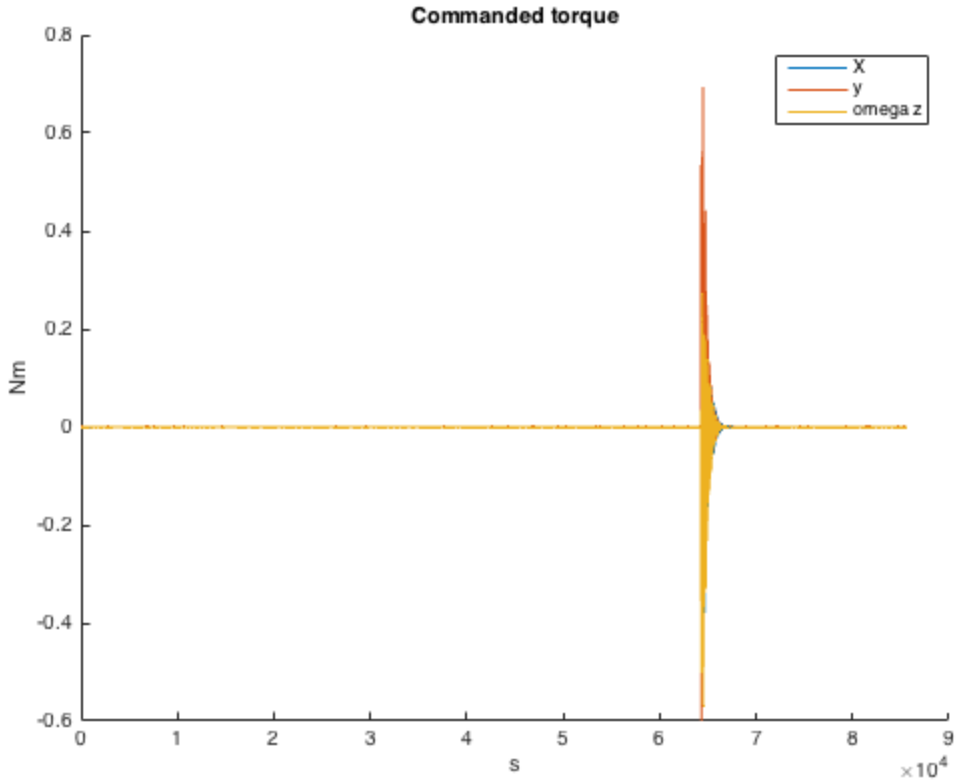
```
fprintf('Max torque on X axis %f \n', max(Tc.data(1:len,1)))
fprintf('Max torque on Y axis %f \n', max(Tc.data(1:len,2)))
fprintf('Max torque on Z axis %f \n', max(Tc.data(1:len,3)))
fprintf('An ion thruster could deliver this kind of torque because
\n');
fprintf('they are typically .5 N thrusters and a %f Nm torque would
 \n',...
    max(Tc.data(1:len,2)));
fprintf('be reasonable\n');
```

```
Max torque on X axis 504.877169
Max torque on Y axis 496.707258
Max torque on Z axis 415.319672
When we ignore the 360 deg swivel
Max torque on X axis 0.012825
Max torque on Y axis 0.004973
Max torque on Z axis 0.002692
An ion thruster could deliver this kind of torque because
they are typically .5 N thrusters and a 0.004973 Nm torque would
be reasonable
```

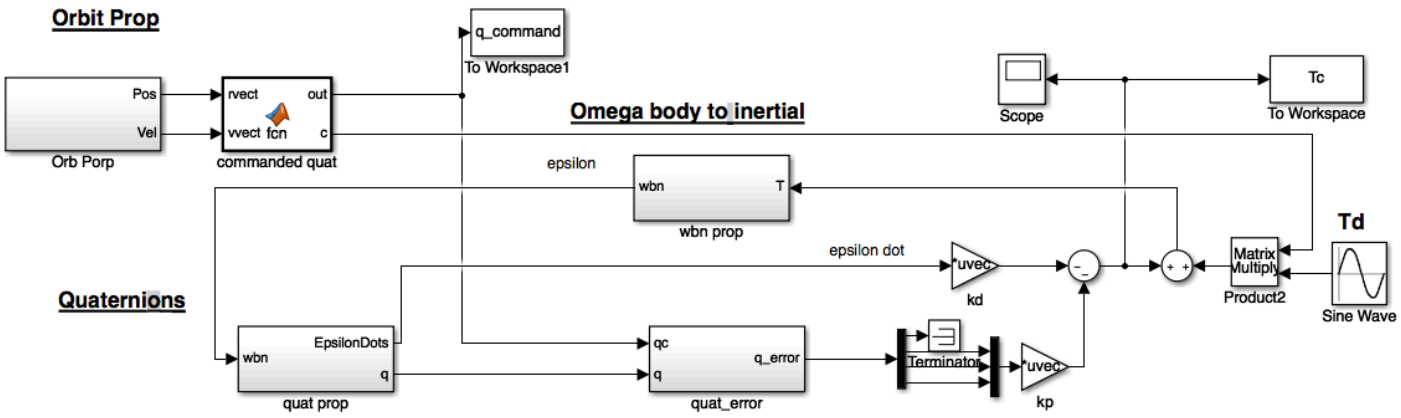


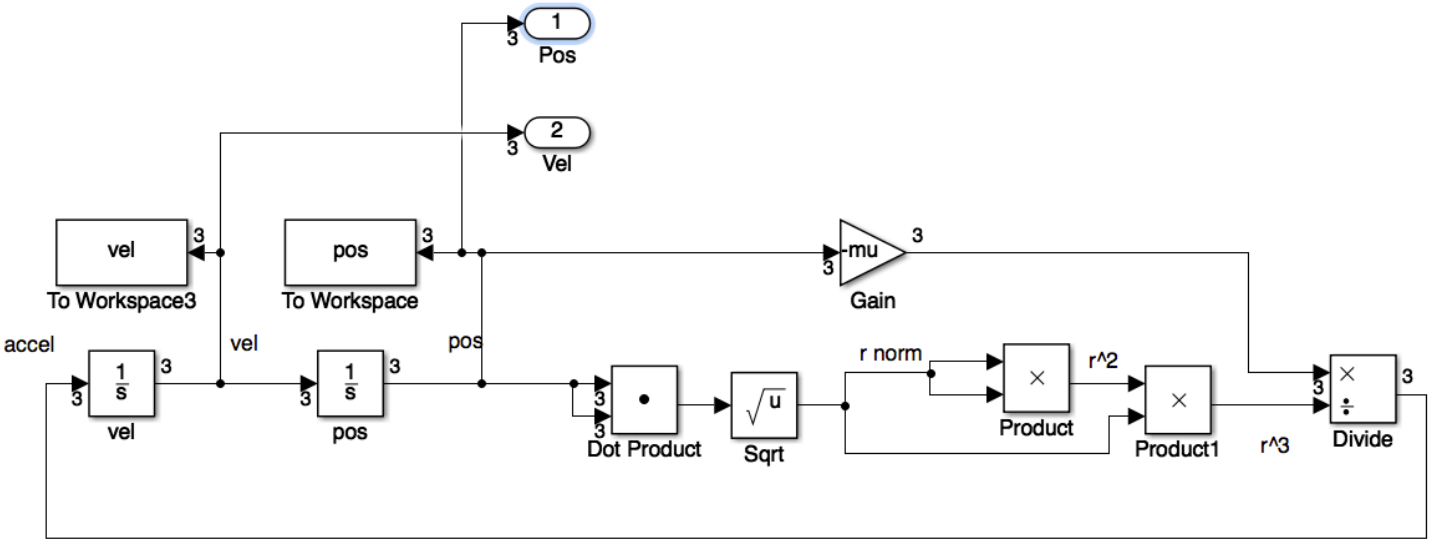

Body to Inertial Angular Velocity
omega x
omega y
omega z
Rad/s
s
×10^4

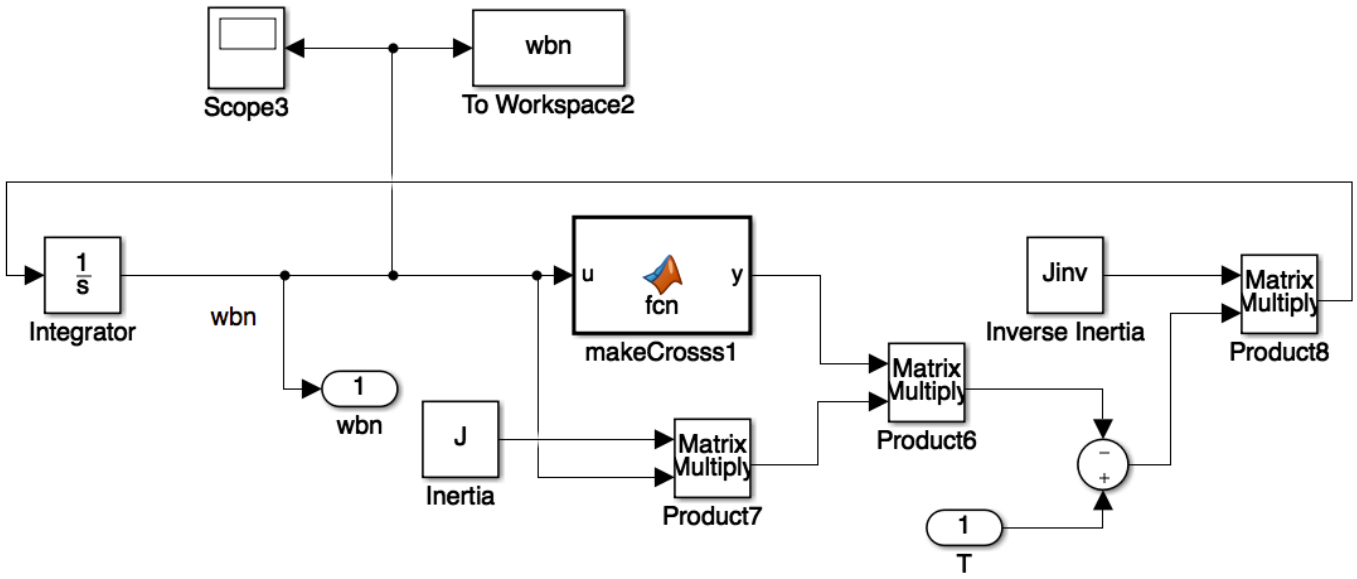

Command Quaternions for one Orbit
eta
q1
q2
q3
Time (s)
×10^4

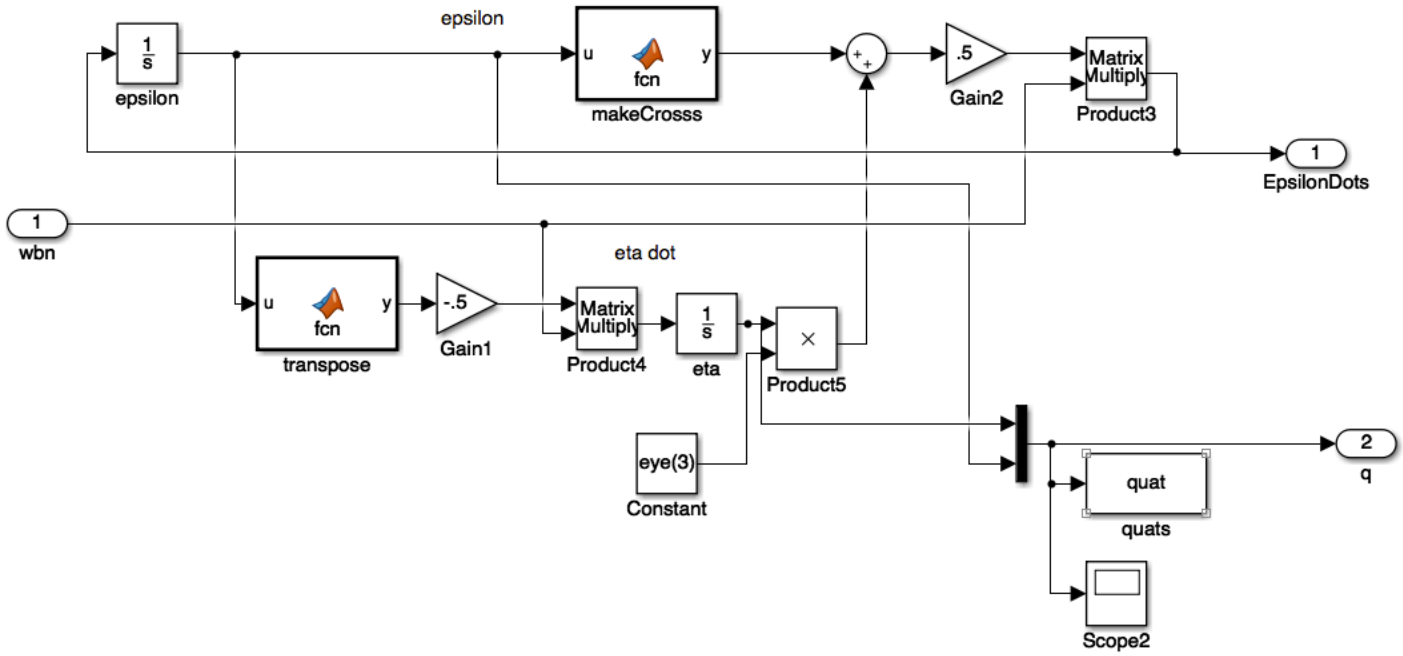


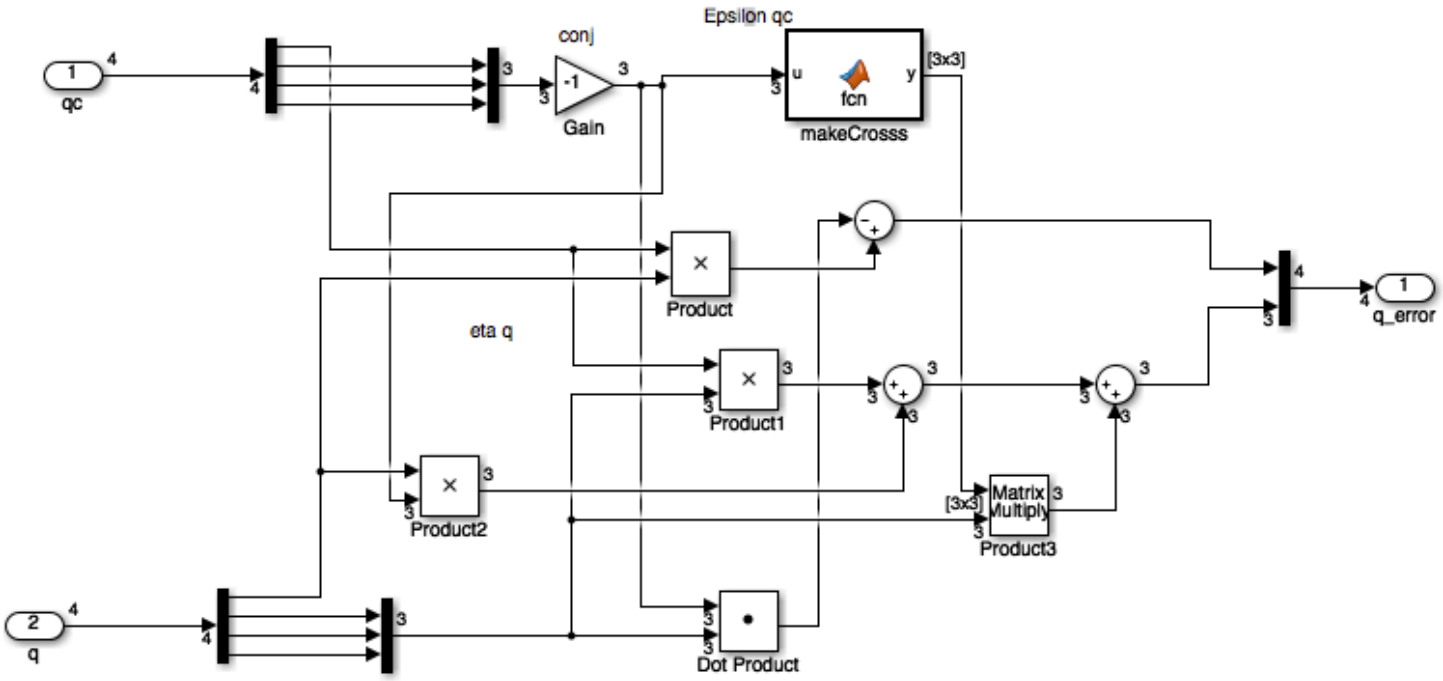
Published with MATLAB® R2015b


Orbit Prop
q_command
To Workspace1
Pos
Vel
Orb Porp
rvect
vvect
fcn
out
c
commanded quat
Omega body to inertial
Scope
Tc
To Workspace
epsilon
wbn
T
wbn prop
Td
epsilon dot
*uvec
kd
Matrix Multiply
Product2
Sine Wave
Quaternions
EpsilonDots
wbn
q
quat prop
qc
q
q_error
quat_error
Terminator
*uvec
kp


1
Pos
2
Vel
vel
To Workspace3
pos
To Workspace
mu
Gain
accel
vel
pos
r norm
r^2
r^3
Product
Product1
Divide
Dot Product
Sqrt
vel
pos


Scope3
wbn
To Workspace2
1
s
Integrator
wbn
1
wbn
J
Inertia
u
fcn
y
makeCrosss1
Matrix
Multiply
Product7
Matrix
Multiply
Product6
Jinv
Inverse Inertia
Matrix
Multiply
Product8
1
T


epsilon
1
s
epsilon
u
fcn
y
makeCrosss
.5
Gain2
Matrix
Multiply
Product3
1
EpsilonDots
1
wbn
eta dot
u
fcn
y
transpose
-.5
Gain1
Matrix
Multiply
Product4
1
s
eta
Product5
eye(3)
Constant
2
q
quat
quats
Scope2


1
qc
4
conj
-1
Gain
Epsilon qc
u
fcn
y
[3x3]
makeCrosss
Product
eta q
Product1
Product2
Matrix Multiply
Product3
2
q
Dot Product
q_error