Attitude Final Project

Author: Eric Johnson

Introduction: 2

Reference Mission: 2

Dynamics Model: 2

Sensors: 3

Attitude Determination: 4

Control System: 4

Actuators: 5

Simulation Results: 6

References: 10

Appendix: 11

Introduction:

The purpose of this attitude project, is to develop and verify flight code that is tasked to follow a given attitude profile. This flight code will be verified via simulation containing all of the relevant dynamics, sensors, perturbations and sensor error associated with operation in space. Upon completion of the project, flight code capable of deployment to real hardware should be demonstrated.

Reference Mission:

A satellite is in a perfectly circular orbit with position and velocity in inertial coordinates given by

$$\rho = 6,600 \text{ km}$$
$$\omega = \sqrt{\mu/\rho^3}$$
$$\boldsymbol{r}^i(t) = \frac{\rho}{\sqrt{2}} \begin{bmatrix} -\cos\omega t \\ \sqrt{2}\sin\omega t \\ \cos\omega t \end{bmatrix}$$
$$\dot{\boldsymbol{r}}^i(t) = \frac{\omega\rho}{\sqrt{2}} \begin{bmatrix} \sin\omega t \\ \sqrt{2}\cos\omega t \\ -\sin\omega t \end{bmatrix}$$

where μ is Earth's gravitational constant. The satellite's nominal attitude is such that its body x-axis (= b_x, the roll axis) is always pointing towards the center of Earth, while the body y-axis (= b_y, the pitch axis) is always aligned with the velocity vector. The satellite's inertial matrix $\mathbf{J}^b_{sat}$ is given by

$$\boldsymbol{J}^b_{sat} = \begin{bmatrix} 1000 & 0 & 0 \\ 0 & 500 & 0 \\ 0 & 0 & 600 \end{bmatrix} \text{ kg m}^2$$

At time t = 0 the satellite has an attitude pointing error of 2 degrees around the pitch axis (body y-axis), and an angular velocity error of 0.005 rad/s around the roll axis (body x-axis). The satellite is equipped with three orthogonal Reaction Wheels (RWs), the rotation axes of the wheels are aligned with the three body axes

Dynamics Model:

This block takes as inputs the RWs angular momentum $\mathbf{H}^b_w(t)$ and its rate ${}^b\dot{\mathbf{H}}^b_w(t)$ and produces the satellite's attitude and angular velocity. The angular velocity and attitude of the satellite evolve as

$${}^b\dot{\boldsymbol{\omega}}^b_{b/i}(t) = \boldsymbol{J}^{-1}_{cg}\Big(-\boldsymbol{\omega}^b_{b/i}(t) \times (\boldsymbol{J}^b_{cg}\boldsymbol{\omega}^b_{b/i}(t)) - {}^b\dot{\boldsymbol{H}}^b_W(t) - \boldsymbol{\omega}^b_{b/i} \times \boldsymbol{H}^b_W(t)\Big)$$

$$\dot{\bar{q}}_i^b = \frac{1}{2}\begin{bmatrix}\boldsymbol{\omega}_{b/i}^b \\ 0\end{bmatrix} \otimes \bar{q}_i^b$$

To help verify the viability of the control system the perturbations from aerodynamic drag and the magnetic charge of the spacecraft were modeled. The following equations are used to model the aerodynamic drag torque.

$$\boldsymbol{a}_D = -\frac{1}{2}\frac{C_D A}{m}\rho\, v_{rel}\, \boldsymbol{v}_{rel}, \qquad \boldsymbol{v}_{rel} = \boldsymbol{v}_{s/c} - \boldsymbol{\omega}_{\oplus} \times \boldsymbol{r}_{s/c}$$

$$F_d = m * \alpha_D$$

$$M_d = r_{/cg} \times F_d$$

The following equation gives the density at various altitudes.

$$\rho(r) = \rho_0 \exp\left(-\frac{(r - r_{ellp}) - h_0}{H}\right)$$

h0 is a reference altitude and H is the corresponding scaling factor for the reference altitude. These values are found from table 8-4 in Fundamentals of Astrodynamics and Applications by David Vallado. Since most of the atmosphere resides below 1000 km and our orbital radius is 6000 km, h0 is chosen to be 1000 [km] and H to be 268. The magnetic torque from the charge of the spacecraft is modeled as follows

$$Mmag = m \times B$$

Where m is the charge of the spacecraft. M is chosen to be [100; 0; 0] [A m^2]. This value was chosen to create a significant torque on the vehicle for which the attitude control system is needed to counteract.
B is found from

$$\boldsymbol{B}^n = B_0\left(\frac{R_e}{\|\boldsymbol{r}\|}\right)^3 \begin{bmatrix}\cos(\lambda) \\ 0 \\ -2\sin(\lambda)\end{bmatrix}$$

$$\boldsymbol{B}^i = (\boldsymbol{T}_i^n)^{\mathrm{T}}\, \boldsymbol{B}^n$$

Sensors:

The sensors block takes as inputs the satellite's attitude $\mathbf{q}^b_i(t)$ and angular velocity $\boldsymbol{\omega}^b_{b/i}(t)$, as well as its position $\mathbf{r}^i(t)$. The outputs are the gyro measurement $\tilde{\boldsymbol{\omega}}^b_{b/i}(t)$, the magnetometer measurement $\tilde{\mathbf{B}}^b(t)$ and the Earth Horizon sensor measurement $\tilde{\mathbf{d}}^b(t)$

$$\tilde{\omega}^b_{b/i}(t) = \omega^b_{b/i}(t) + b + \nu_{gyro}(t)$$
$$\tilde{B}^b(t) = T^b_i(t)\, B^i(t) + \nu_{mag}(t)$$
$$\tilde{d}^b(t) = T(\nu_{Hor})\, T^b_i(t)\, \frac{-r^i(t)}{\|r^i(t)\|}$$
$$T^b_i = I_{3\times3} - 2q_s[q_v\times] + 2[q_v\times]^2$$

gyro bias (b) was chosen based on the LN-200s IMU from northrop grumman. The gyro bias is 1 degree per hour and the random walk (vgyro(t)) is .07 degree per the square root of hour. The Magnetometer selected is the Mesisei 3-axis magnetometer for small satellite. This magnetometer has a measurement accuracy of +- 1 deg (vmag(t)). The earth horizon sensor selected is the MAI-SES Static Earth Sensor. This sensor has a resolution of .25 degrees.

Attitude Determination:

This block takes as inputs the sensors measurements $\tilde{\boldsymbol{\omega}}^b_{b/i}(t)$, $\tilde{\mathbf{B}}^b(t)$, $\tilde{\mathbf{B}}^i(t)$, $\tilde{\mathbf{d}}^b(t)$ as well as the satellite's position $\mathbf{r}^i(t)$ and produces as outputs the estimated angular velocity $\tilde{\boldsymbol{\omega}}^b_{b/i}(t)$ and attitude $\check{\mathbf{T}}^{\hat{}b}_i(t)$. We are not studying how to estimate the gyro bias, therefore a small bias is chosen and the estimated angular velocity $\hat{\omega}^b_{b/i}(t)$ set equal to the measured one $\tilde{\boldsymbol{\omega}}^b_{b/i}(t)$. The TRIAD algorithm is used to estimate the satellite's attitude as a **DCM** $\check{\mathbf{T}}^{\hat{}b}_i(t)$.

$$\mathbf{x}^b = \tilde{\mathbf{d}}^b_1; \qquad \mathbf{x}^i = \mathbf{d}^i_1$$
$$\mathbf{z}^b = \frac{\tilde{\mathbf{d}}^b_1 \times \tilde{\mathbf{d}}^b_2}{|\tilde{\mathbf{d}}^b_1 \times \tilde{\mathbf{d}}^b_2|} \qquad \mathbf{z}^i = \frac{\mathbf{d}^i_1 \times \mathbf{d}^i_2}{|\mathbf{d}^i_1 \times \mathbf{d}^i_2|}$$
$$\mathbf{y}^b = \mathbf{z}^b \times \mathbf{x}^b; \qquad \mathbf{y}^i = \mathbf{z}^i \times \mathbf{x}^i$$

$$\hat{T}^b_i = \left[\mathbf{x}^b \quad \mathbf{y}^b \quad \mathbf{z}^b\right] \left[\mathbf{x}^i \quad \mathbf{y}^i \quad \mathbf{z}^i\right]^{\mathrm{T}}$$

Where $\bar{d}^b_1$ is set equal the vector formulated from the horizon sensor. d^i_1 is set equal to the vector formulated from the magnetometer reading.

Control System:

This block takes as inputs the estimated angular velocity $\hat{\omega}^b_{b/i}(t)$, attitude $\check{\mathbf{T}}^b_i(t)$, position $\mathbf{r}^i(t)$, and velocity $\dot{r}^i(t)$, as well as the RWs angular velocities $\tilde{\boldsymbol{\omega}}_1$, $\tilde{\boldsymbol{\omega}}_2$, $\tilde{\boldsymbol{\omega}}_3$, and produce as output the RWs commanded rates $\dot{\omega}_{1,comm}$ $\dot{\omega}_{2,comm}$ and $\dot{\omega}_{3,comm}$. The actual rate of the wheels are affected by a small error δαb. The error for the PD controller is estimated as follows

$$\delta \boldsymbol{T}(t) = \hat{\boldsymbol{T}}_i^b(t)\, \bar{\boldsymbol{T}}_i^b(t)^{\mathrm{T}}$$
$$\boldsymbol{\epsilon}^b(t) \simeq -0.5 \begin{bmatrix} \delta T_{32} - \delta T_{23} \\ \delta T_{13} - \delta T_{31} \\ \delta T_{21} - \delta T_{12} \end{bmatrix}$$

The change in ε is modeled as

$$\dot{\epsilon}(t) \simeq -[\bar{\omega}_{b/i}^b \times]\, \epsilon(t) + \delta\omega^b(t)$$

From this the commanded angular velocities of the reaction wheels are solved for as follows

$$^b\dot{\boldsymbol{\omega}}_{W,comm}^b = \begin{bmatrix} \dot{\omega}_{1,comm} \\ \dot{\omega}_{2,comm} \\ \dot{\omega}_{3,comm} \end{bmatrix} = \frac{\boldsymbol{J}_{sat}^b \boldsymbol{K}_d}{C_W} \boldsymbol{\epsilon}^b + \frac{\boldsymbol{J}_{sat}^b \boldsymbol{K}_p}{C_W} \dot{\boldsymbol{\epsilon}}^b - \hat{\boldsymbol{\omega}}_{b/i}^b \times \begin{bmatrix} \tilde{\omega}_1 \\ \tilde{\omega}_2 \\ \tilde{\omega}_3 \end{bmatrix}$$

The proportional and derivative gains are chosen to both be 2. This value was chosen from manually testing various gains until the spacecraft was able to hold it’s profile.

Actuators:

Reaction wheels were chosen for the spacecraft’s control actuators. These were chosen because they are simplistic to model as the spacecraft’s moment of inertia matrix is unchaged from actuation. This block takes as input the commanded wheel rate $^b\dot{\omega}^b_{w,comm}$ and produces as output the RWs angular momentum $\mathbf{H}^{\mathbf{b}}_w(t)$ and its rate $^b\dot{\mathbf{H}}^b_w(t)$, as well as the wheels measured angular velocities $\tilde{\boldsymbol{\omega}}_1, \tilde{\boldsymbol{\omega}}_2, \tilde{\boldsymbol{\omega}}_3$. The actual rate of the wheels is not be exactly the commanded rate but be affected by a small error $\delta\boldsymbol{\alpha}^b_w(t)$. The actuators are modeled as follows.

$$^b\dot{\boldsymbol{\omega}}_W^b(t) = {}^b\dot{\boldsymbol{\omega}}_{W,comm}^b(t) + \delta\boldsymbol{\alpha}_W^b(t)$$
$$\boldsymbol{\omega}_W^b(t) = \begin{bmatrix} \tilde{\omega}_1 \\ \tilde{\omega}_2 \\ \tilde{\omega}_3 \end{bmatrix} = \int \dot{\boldsymbol{\omega}}_W^b(t)\, dt$$
$$^b\dot{\boldsymbol{H}}_W^b(t) = C_W\, {}^b\dot{\boldsymbol{\omega}}_W^b(t)$$
$$\boldsymbol{H}_W^b(t) = C_W\, \boldsymbol{\omega}_W^b(t)$$

The reaction wheels chosen for the spacecraft are the RW8s from Blue Canyon Technologies. In addition to the above equations the reaction wheel angular velocity and angular acceleration are limited by the hardware capabilities of the reaction wheels. The angular velocity and acceleration limits are found from the data sheets’ specified max momentum and max torque.

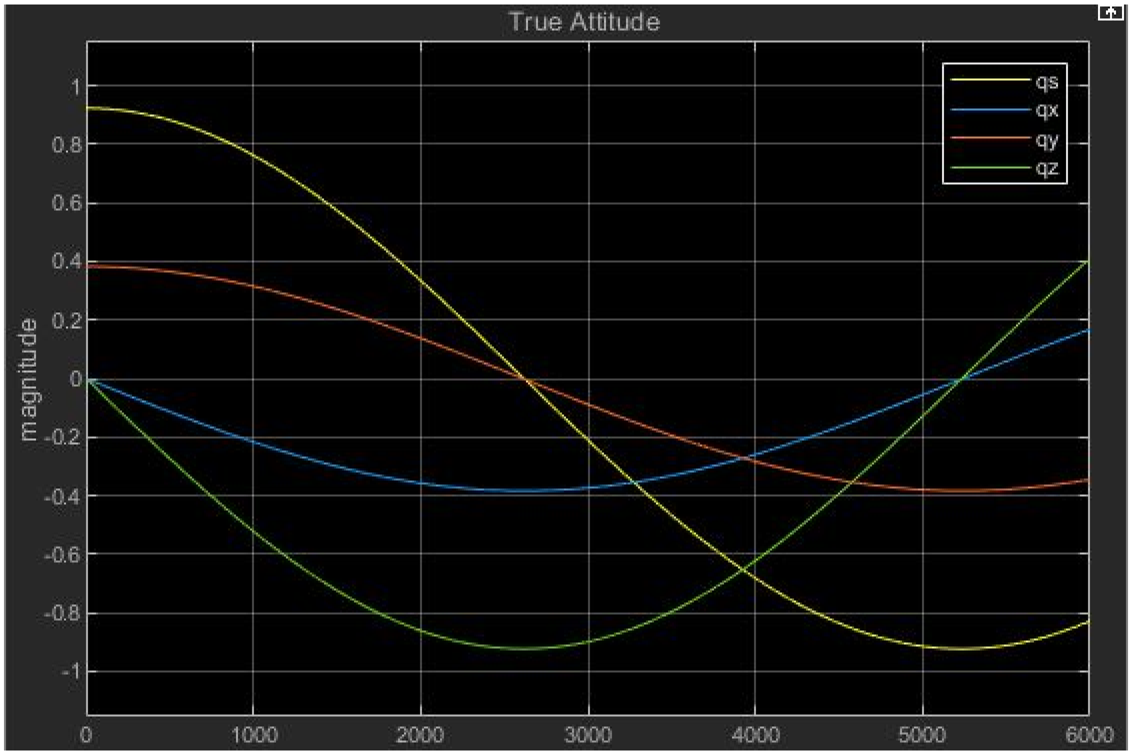
$$\omega_{max} = H_{max}/C$$
$$\alpha_{max} = T_{max}/C$$

Where $H_{max} = 8[Nms]$ and $T_{max} = .3\ [Nm]$ Originally, the RW4s were chosen, but it was soon discovered the required output momentums were too large for their capabilities. Choosing more massive reaction wheels allowed the spacecraft to effectively counteract the torques from the perturbations.

Simulation Results:

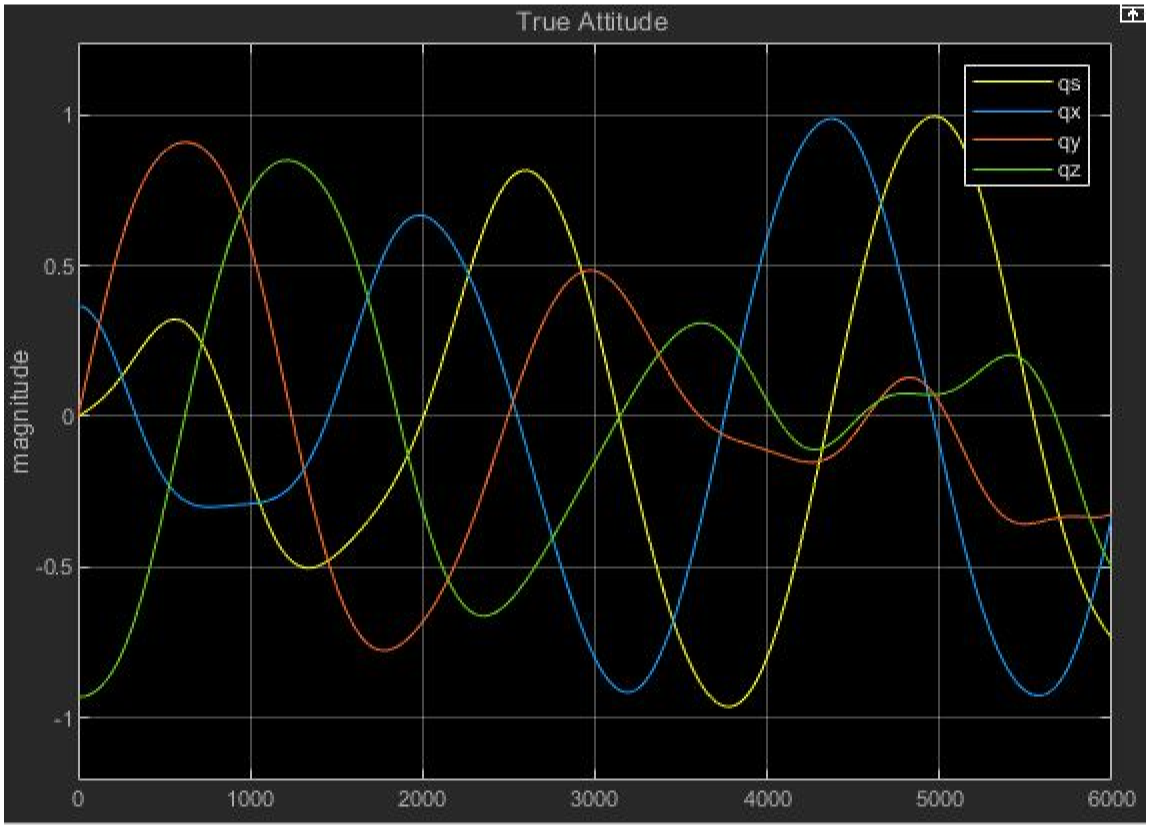
Reference Profile

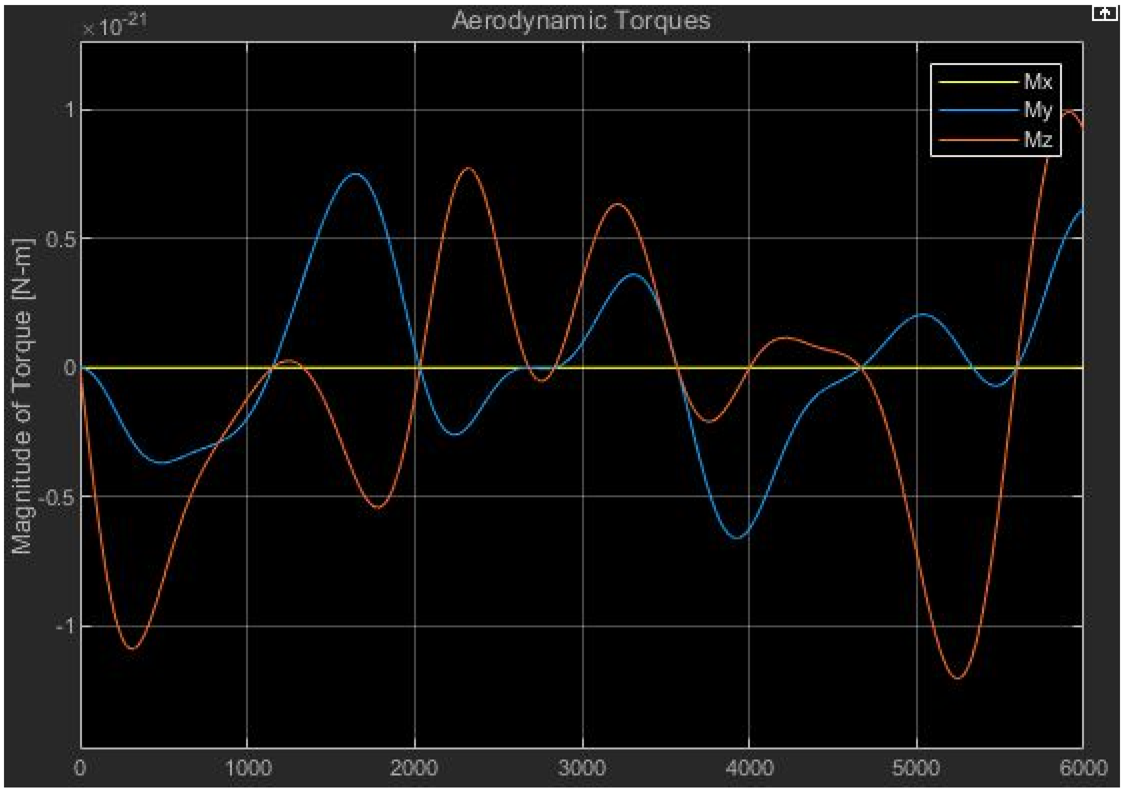
- No perturbations
- no initial pointing error
- no attitude controller

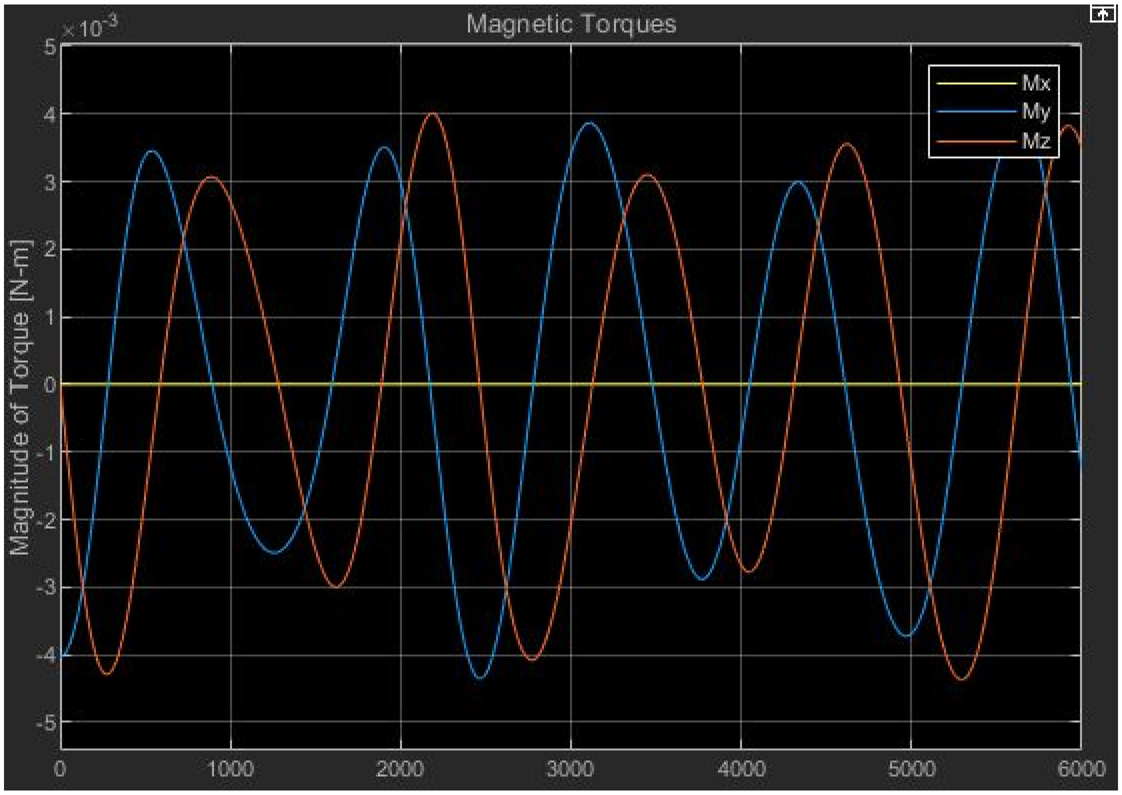


Mission 1:

- Yes perturbations
- Yes initial pointing error.
- No Control loop


True Attitude
qs
qx
qy
qz
magnitude
1
0.5
0
-0.5
-1
0
1000
2000
3000
4000
5000
6000

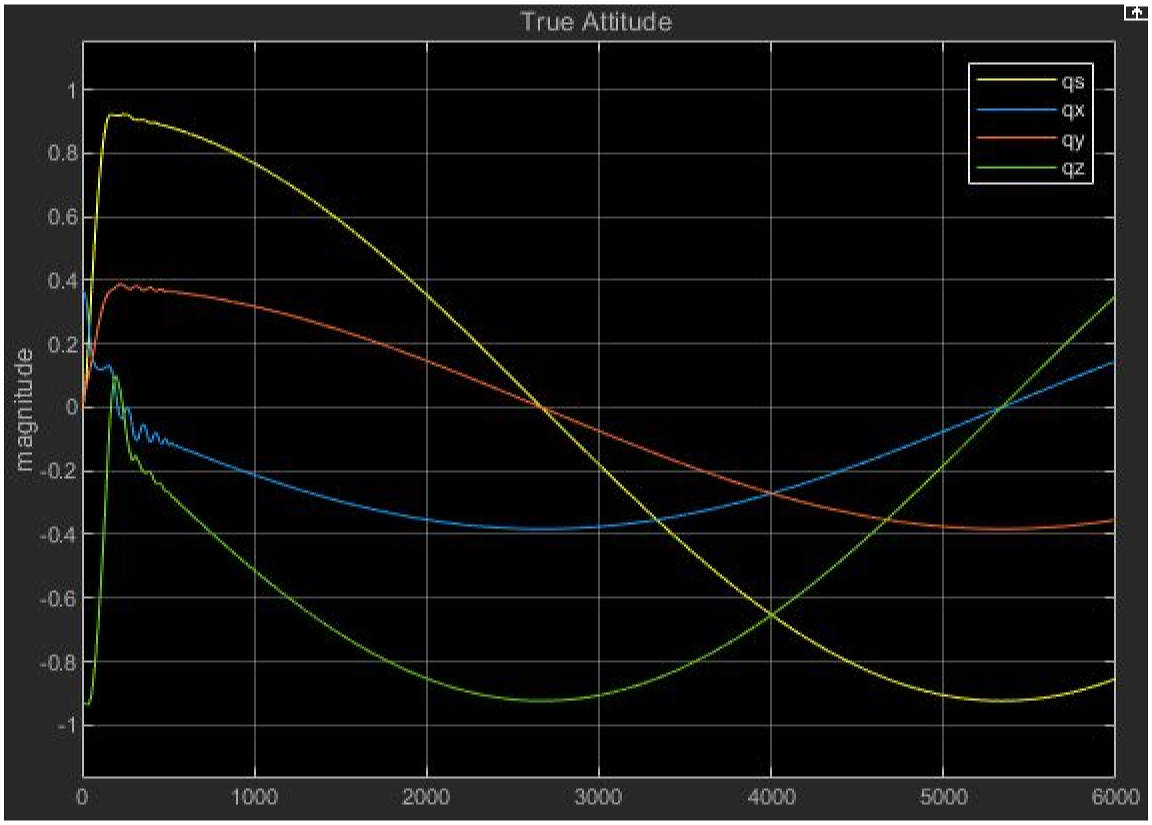

Aerodynamic Torques
×10^-21
Mx
My
Mz
Magnitude of Torque [N-m]
1
0.5
0
-0.5
-1
0
1000
2000
3000
4000
5000
6000



The plots from mission 1 prove that the system requires attitude control.

Mission 2:

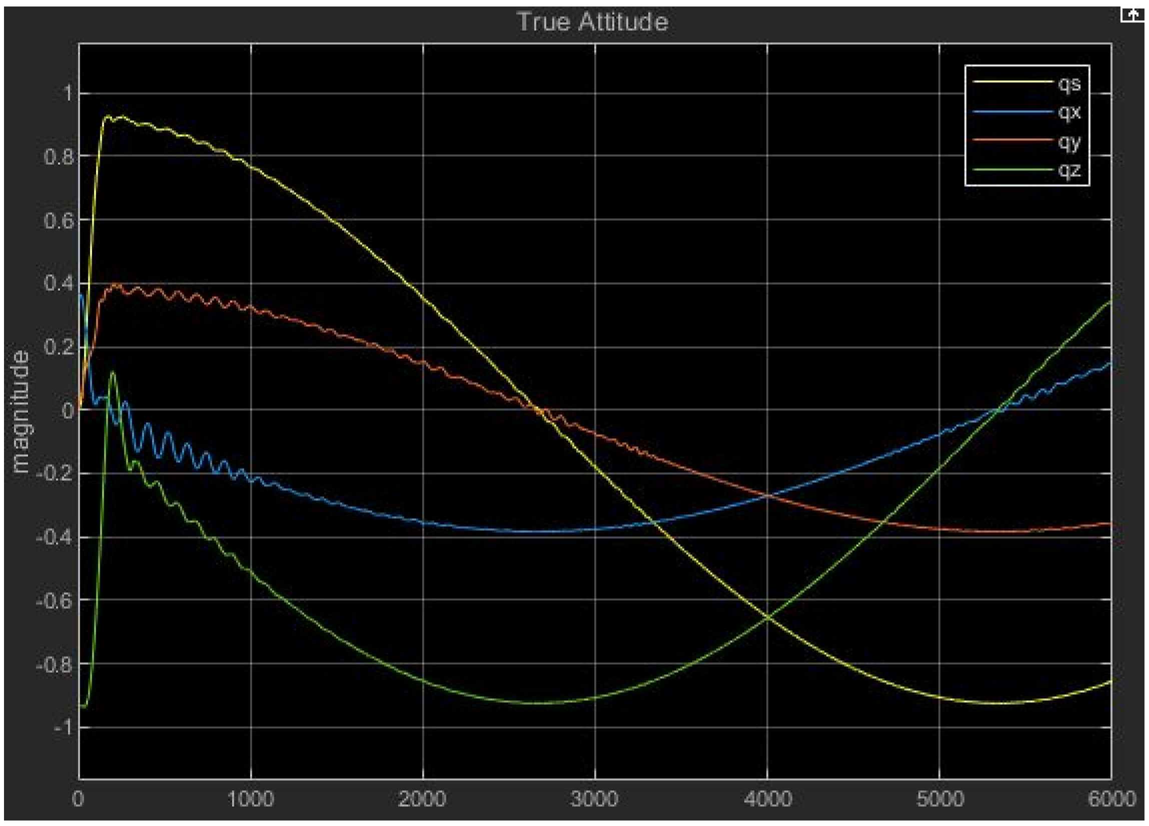
- Yes perturbations
- Yes initial pointing error.
- No sensor error
- Yes Controller on



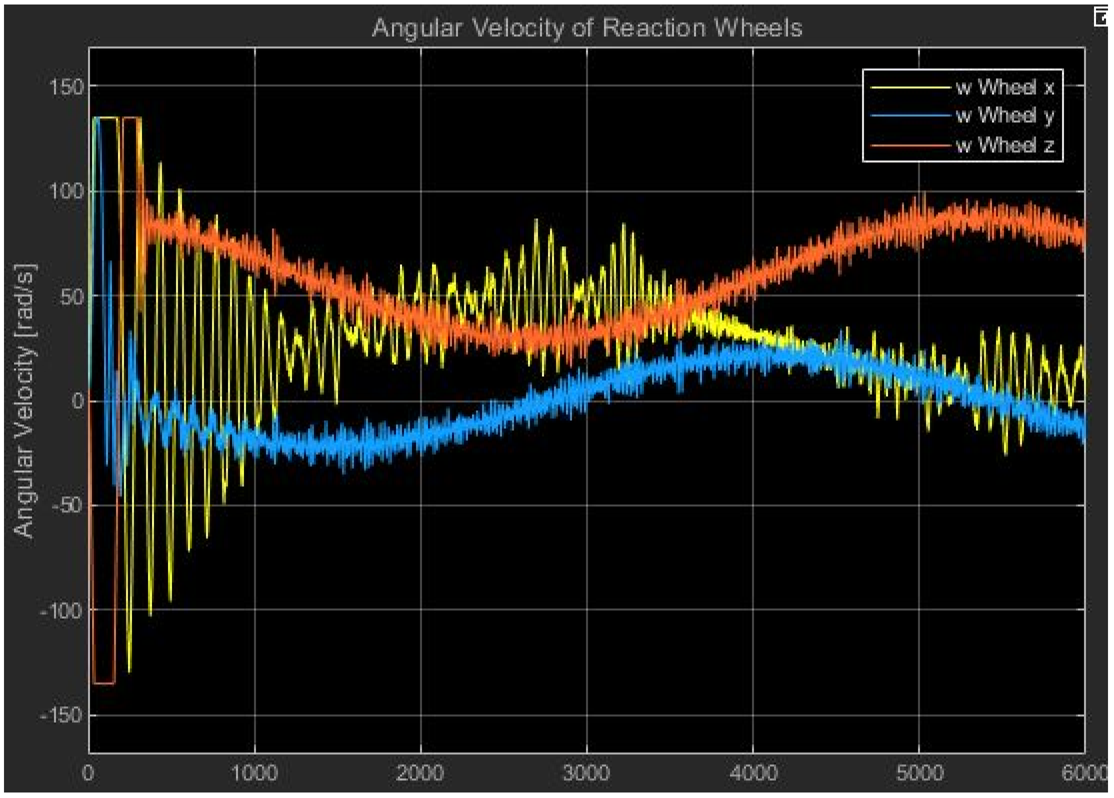
The orientation plot from mission 2 shows that the control system can achieve proper guidance.

<u>Full Mission:</u>

- Yes perturbations
- Yes initial pointing error
- Yes sensor measurement noise
- Yes Controller



The orientation plot from the full mission proves that the sensor package will provide accurate enough data to hold the profile



The angular velocity of the reaction wheels plot illustrates the mechanical limitations of the wheels in the first 200 secs. We can observe flat lines at the peaks because of the maximum deliverable momentum limit being reached.

References:

Imu - https://www.northropgrumman.com/Capabilities/LN200FOG/Documents/ln200s.pdf

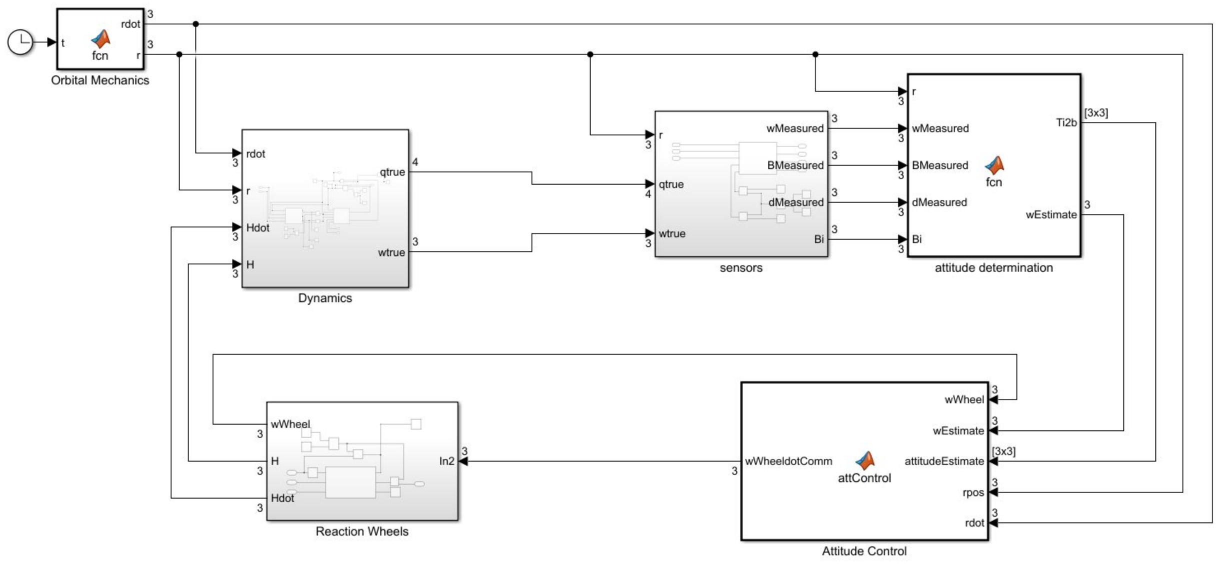
Magnetometer -
http://www.meisei.co.jp/english/products/space/magnetic_sensor_for_spacecraft.html

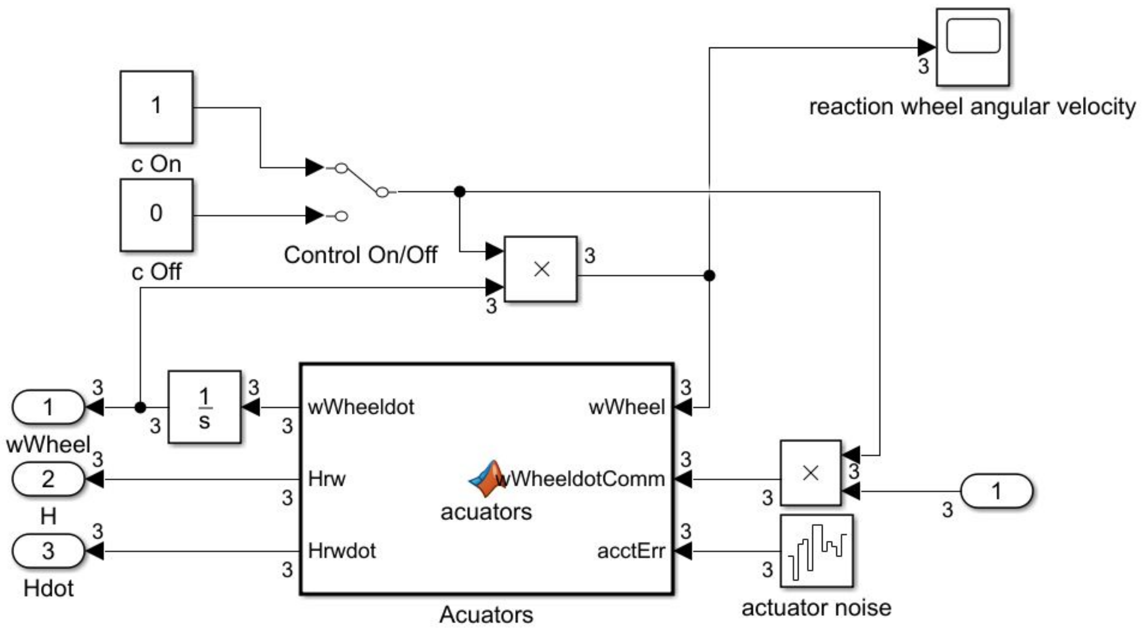
Horizon sensor - https://www.adcolemai.com/ir-earth-horizon-sensor

Reaction Wheels - http://bluecanyontech.com/rw8/

Fundamentals of Astrodynamics and Applications by David Vallado

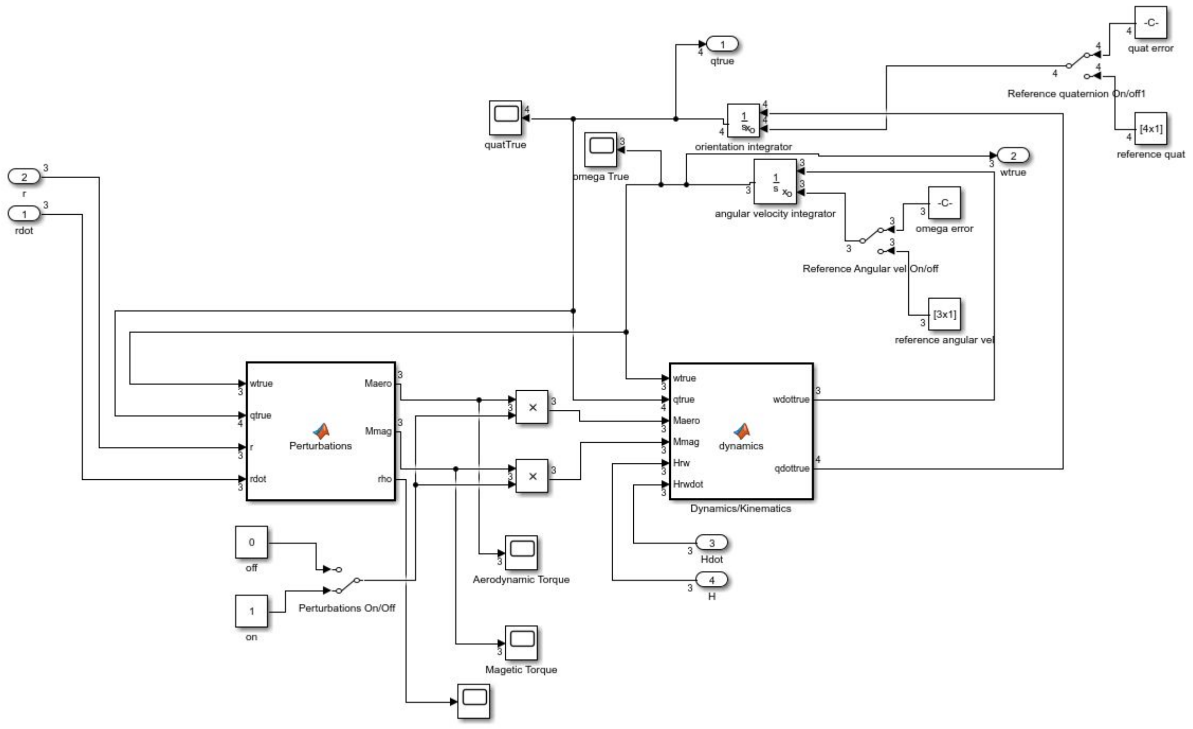
Appendix:





```
function [wWheeldot, Hrw, Hrwdot] = acuators( wWheel, wWheeldotComm, acctErr)
%reaction wheel specs
%http://bluecanyontech.com/rw8/
mw = 4.1; %kg
r = .17; %m
h = .07; %m
C = .5*mw*r^2; %kg m^2
maxMomentum = 8; %Nms = kg(m^2/s)
maxwWheel = maxMomentum/C;
maxTorque = .3;
maxAlpha = maxTorque/C;

%check to see if the commanded value is out of the reaction wheel's
%capabilities
for i=1:3
   if wWheel(i) > maxwWheel
       wWheel(i) = maxwWheel;
       if wWheeldotComm(i) > 0
           wWheeldotComm(i) = 0;
       end
   end
   if wWheel(i) < -maxwWheel
       wWheel(i) = -maxwWheel;
       if wWheeldotComm(i) < 0
           wWheeldotComm(i) = 0;
       end
   end
```

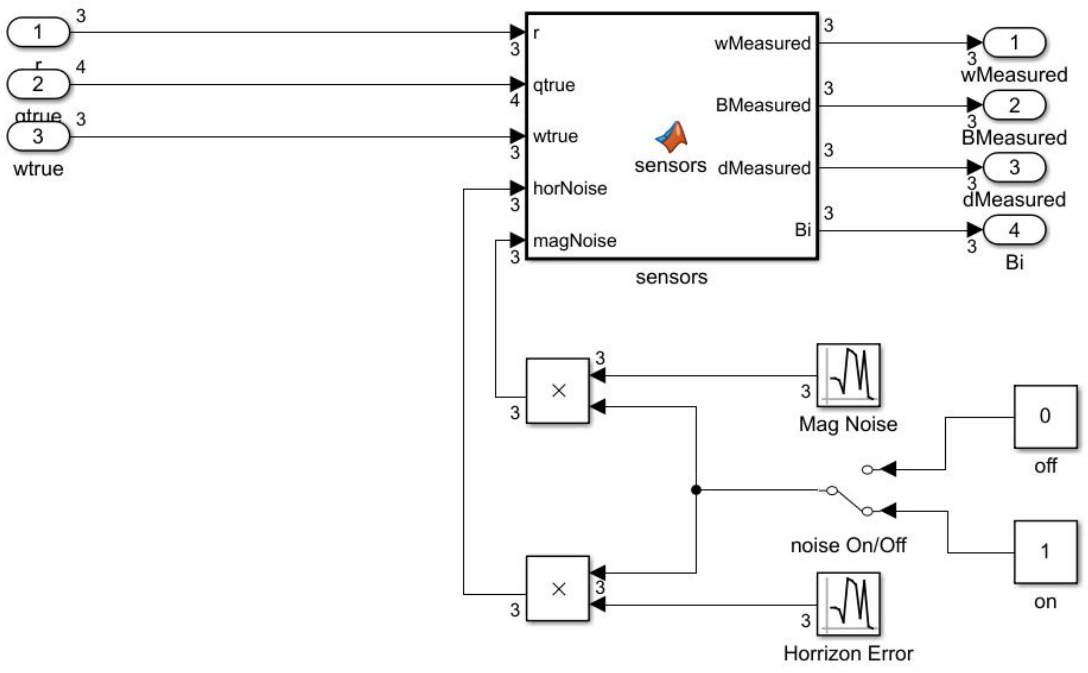
```
end

%check to see if maximum allowable torque is being exceeded
for i=1:3
   if wWheeldotComm(i) > maxAlpha
       wWheeldotComm(i) = maxAlpha;
   end
   if wWheeldotComm(i) < -maxAlpha
       wWheeldotComm(i) = -maxAlpha;
   end
end

wWheeldot = wWheeldotComm + acctErr;
Hrw = C*wWheel;
Hrwdot = C*wWheeldot;
```



```
%
%AERODYNAMIC TORQUE
%
rSwrtCG = [1.2; 0; 0]; %[m]
A = 10; %m^2
Cd = 5;
wE = [0; 0; 7.2921158553E-5];
v = rdot*1000 + cross(wtrue,rSwrtCG) - cross(wE, r*1000); %m/s
Nhat = r/norm(r);
vhat = v/norm(v);

%Exponential Atmospheric Model
H = 268;
```

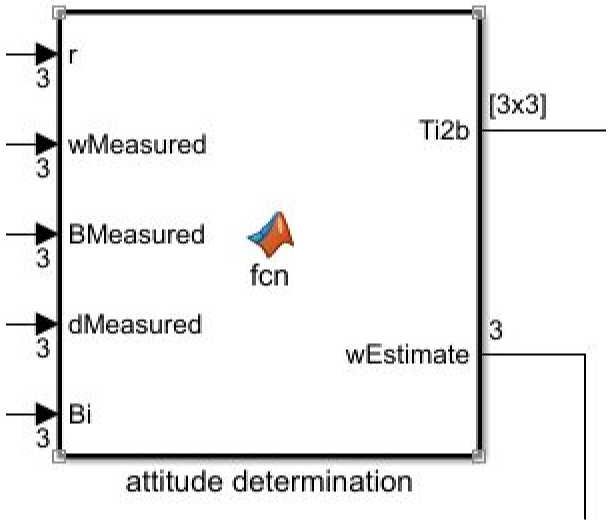
```
rho0 = 3.019E-15;
rho = rho0*exp(-(norm(r)-1000)/H);

%torque calculation
fAero = -.5*Cd*rho*(norm(v)^2)*dot(Nhat,vhat)*vhat*A;
Maero = cross(rSwrtCG, fAero);

%
%MAGNETIC TORQUE
%
mSC = [100;0;0]; %magnetic feild of the spacecraft
Bo = 3.12E-5;
Re = 6371; %km

qv = qtrue(2:4);
qs = qtrue(1);

skewv = [0, -qv(3), qv(2);
         qv(3), 0, -qv(1);
         -qv(2), qv(1), 0];

Ti2b = eye(3) - 2*qs*skewv + 2*skewv*skewv;

%DCM inertial to NEZ;
nz = -r/norm(r);
ny = (cross(nz, [0; 0; 1]))/norm(cross(nz, [0; 0; 1]));
nx = cross(ny, nz);
Ti2ned = [ nx'; ny'; nz'];
lambda = asin(r(3)/norm(r));

%magnetic field calculation
Bn = Bo*((Re/norm(r))^3)*[cos(lambda); 0; -2*sin(lambda)];
Bi = Ti2ned'*Bn;
Bb = Ti2b*Bi;

Mmag = cross(mSC,Bb);
end

function [ wdottrue, qdottrue] = dynamics(wtrue, qtrue, Maero, Mmag, Hrw, Hrwdot)
Jcg = [1000, 0, 0; 0, 500, 0; 0, 0, 600];


%
%ATTITUDE PROPAGATION
%
wdottrue = inv(Jcg)*(-cross(wtrue,Jcg*wtrue+Hrw) - Hrwdot + Maero + Mmag);

qwtrue = [0; wtrue];
qdottrue = .5*quatMult(qtrue, qwtrue);
```

```
end
```

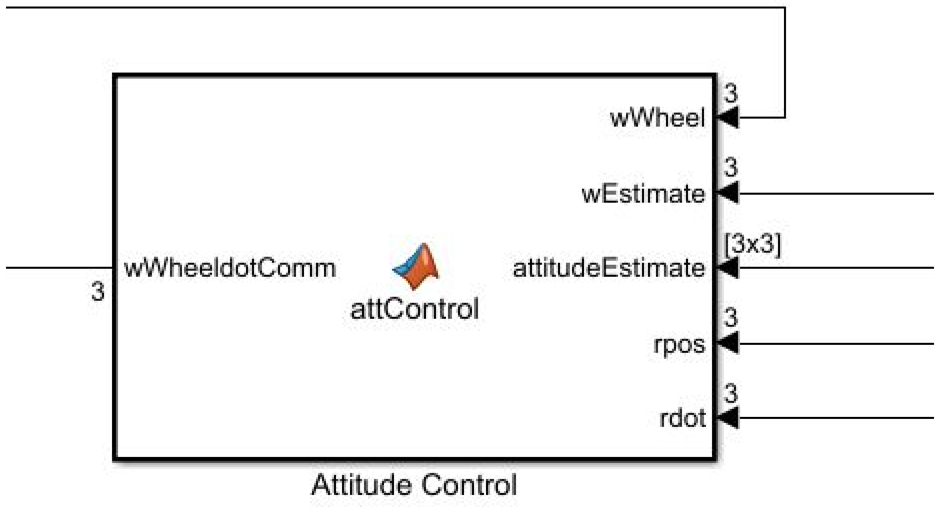


```
function [wMeasured, BMeasured, dMeasured, Bi] = sensors(r, qtrue, wtrue, horNoise, magNoise)

%
%GYRO
%
%https://www.northropgrumman.com/Capabilities/LN200FOG/Documents/ln200s.pdf
%gyro bias = 1degree/hour = 9.696E-6 rad/s
gyroBias = 9.696E-6*[1; 1; 1];
gyroWalk = (.0012*pi/180)^2*[1;1;1];
wMeasured = wtrue + gyroBias + gyroWalk;
qv = qtrue(2:4);
qs = qtrue(1);

skewv = [0, -qv(3), qv(2);
         qv(3), 0, -qv(1);
         -qv(2), qv(1), 0];

Ti2b = eye(3) - 2*qs*skewv + 2*skewv*skewv;

%
%MAGNETOMETER
%
Bo = 3.12E-5;
Re = 6371; %km

%DCM inertial to NEZ;
nz = -r/norm(r);
```

```
ny = (cross(nz, [0; 0; 1]))/norm(cross(nz, [0; 0; 1]));
nx = cross(ny, nz);
Ti2ned = [ nx'; ny'; nz'];
lambda = asin(r(3)/norm(r));

%magnetic field calculation
Bn = Bo*((Re/norm(r))^3)*[cos(lambda); 0; -2*sin(lambda)];
Bi = Ti2ned'*Bn;
BMeasured = Ti2b*Bi + magNoise;
BMeasured =  BMeasured/norm(BMeasured);

%
%Horizon
%
horErrT = theta2T(horNoise);
dMeasured = horErrT*Ti2b*(-r/norm(r));
end

%from dr. zenetti's HW 10
function T = theta2T(theta)

nt = norm(theta);
T = eye(3);

if nt > 1e-10
    S = skew(theta/nt);
    T = T - sin(nt)*S + (1-cos(nt))*S^2;
end
end
%from dr. zenetti's HW 10
function S = skew(v)
S = [0 -v(3) v(2);
     v(3) 0 -v(1);
    -v(2) v(1) 0];
end
```

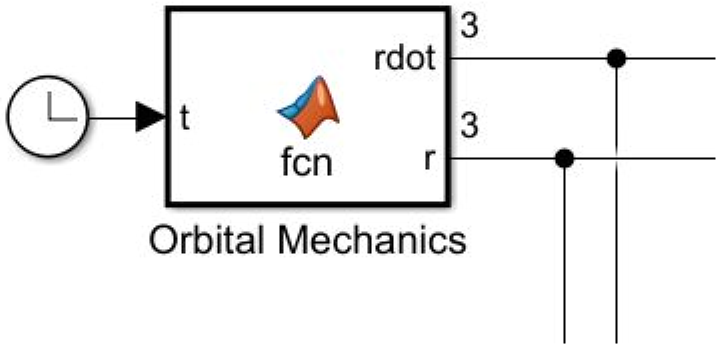


```
function [Ti2b, wEstimate] = fcn(r, wMeasured, BMeasured, dMeasured, Bi)


%
%GYRO
%

wEstimate = wMeasured;

%
%TRIAD
%


d1b = dMeasured;
d2b = BMeasured;

xb = d1b;
zb = cross(d1b, d2b)/norm(cross(d1b, d2b));
yb = cross(zb, xb);

d1i = -r/norm(r);
d2i = Bi/norm(Bi);

xi = d1i;
zi = cross(d1i, d2i)/norm(cross(d1i, d2i));
yi = cross(zi, xi);

Ti2b = [xb, yb, zb]*[xi, yi, zi]';
```



```
function wWheeldotComm  = attControl( wWheel, wEstimate, attitudeEstimate, rpos, rdot)
%reaction wheel specs
%http://bluecanyontech.com/rw8/
mw = 4.1; %kg
r = .17; %m
C = .5*mw*r^2;

JcgS = [1000, 0, 0; 0, 500, 0; 0, 0, 600];

%
%DESIRED ATTITUDE AND RATE
%
mu = 398600.4415; %km
rho = 6600; %km
w = sqrt(mu/rho^3);
wdesired = [0; 0; -w];

xbar = -rpos/norm(rpos);
zbar = cross(xbar, rdot);
zbar = zbar/norm(zbar);
ybar = cross(zbar, xbar);

desiredAtt = [xbar'; ybar'; zbar'];


%
%REACTION WHEEL COMMAND
%

attdot = attitudeEstimate*desiredAtt';
```

```
err = -.5*[attdot(3,2) - attdot(2,3);
           attdot(1,3) - attdot(3,1);
           attdot(2,1) - attdot(1,2)];

dw =  wEstimate - wdesired;
skewWDesired = [0,         -wdesired(3), wdesired(2);
                wdesired(3), 0,         -wdesired(1);
                -wdesired(2), wdesired(1), 0];
errdot = -skewWDesired*err + dw;
Kd = .8;
Kp = .8;

wWheeldotComm = ((Kp*JcgS)/C)*err + ((Kd*JcgS)/C)*errdot - cross(wEstimate, wWheel);
```



```
function [rdot, r] = fcn(t)
mu = 398600.4415;
rho = 6600; %km

w = sqrt(mu/rho^3);
r = (rho/sqrt(2))*[-cos(w*t); sqrt(2)*sin(w*t); cos(w*t)];
rdot = ((w*rho)/(sqrt(2)))*[sin(w*t); sqrt(2)*cos(w*t); -sin(w*t)];
```