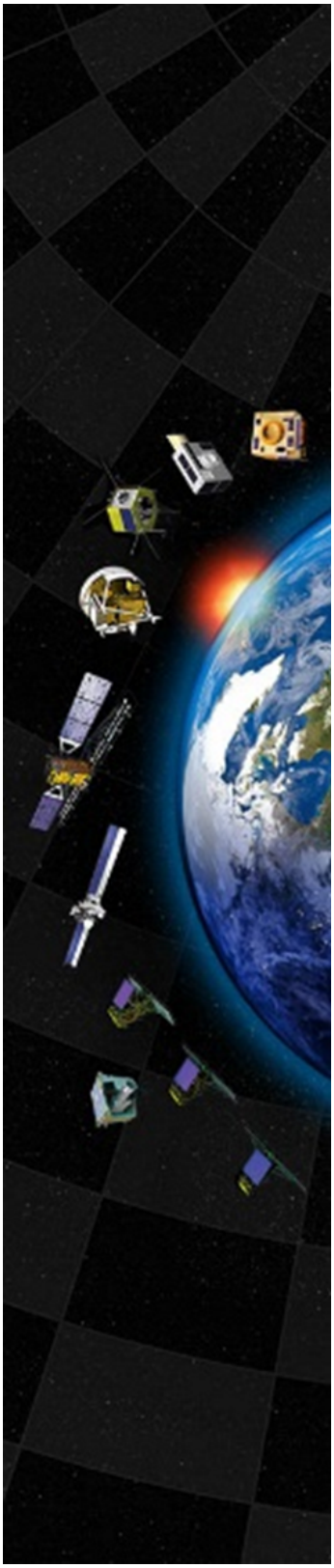
Attitude Dynamics: ASE 372K Professor Renato Zanetti

Final Project Report: LEO Imaging Satellite

Jaxon Liebeck (JDL4297)

Table of Contents Page

Introduction	(3)
Reference Mission	(4)
Complete Simulink Model	(6)
ICs and Orbit Trajectory	(7)
Dynamics Model	(8)
Sensors	(9)
Attitude Determination & Control System (ADCS)	(12)
Simulation Results	(13)
References	(21)
Appendix I: Simulink Code	(22)
Appendix II: Instructions for Running Code	(25)



Simulink Toolboxes Required to run Simulation:

1. Aerospace Toolbox
2. Aerospace Blockset

Introduction

This Project report outlines the creation and analysis of a realistic Low Earth Orbit (LEO) Imaging Satellite. This satellite is designed to be Earth pointing, so that the x-body axis is pointing opposite to Earth's nadir in order to capture images of outer space. An initial inclination of 30 degrees is chosen with the y-body axis pointing towards the velocity of the satellite, so as to cover most of the Earth after multiple orbits.

This satellite will be closely related to the Hubble Space Telescope Satellite which uses reaction wheels, as they are the best option for a satellite with a long designed life expectancy. This satellite will also use three reaction wheels with a similar inclination and inertia matrix as the Hubble Space Telescope. It will also have a similar attitude and hence period.

Hubble Telescope

Magnetic torques and the gravity gradient torque, due to the LEO position of the satellite, will be the two perturbation torques included. Appropriate random noise error will be chosen to most accurately mimic a similar satellite.

This report will demonstrate how the Attitude Determination and Control System (ADCS) software is compartmentalized to realistically control a satellite's orientation for desired orientation and performance.

The simulation will run three times, each for one full orbit. The first simulation will have all noise/error set to zero, and the torques set to zero. The second simulation will have the magnetic field and gravity gradient torques on, but the noise/error is off. The third simulation will have all torques and noise/error switched on to simulate realistic perturbations.

Reference Mission

For this report, an orbit trajectory similar to the Hubble Space Telescope has been chosen. The average altitude for the satellite is 550.00 km from the surface at the starting point. At this altitude, the period of the satellite will approximately be 95 minutes and 40 seconds with a radius from the center of the Earth to be 6928.14 km. For simplicity, the orbit will be treated as circular.

An inclination of 30.00 degrees will be used with an inertial position given by:

$$r^i(t) = \frac{\rho}{2}\begin{bmatrix} \sqrt{3}\cos(\omega t) \\ \sin(\omega t) \\ -\sqrt{3}\cos(\omega t) \end{bmatrix} \tag{1}$$

$$v^i(t) = \frac{\omega\rho}{2}\begin{bmatrix} -\sqrt{3}\sin(\omega t) \\ \cos(\omega t) \\ \sqrt{3}\sin(\omega t) \end{bmatrix} \tag{2}$$

$$\omega = \sqrt{\frac{\mu}{\rho^3}} \tag{3}$$

$$where$$

$$\rho = 6928.14\ km$$

$$\mu = 3.986 * 10^{14}\ \ m^3 s^{-2} \tag{4}$$

$$\hat{\omega}\ nominal\ estimation = \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \frac{rad}{s} \tag{5}$$

The satellite's nominal attitude is designed so that the body x-axis, the roll axis, is always pointing away from the center of the earth so the instruments are facing outer space. The body y-axis is designed to be aligned with the satellite's velocity vector with minimal error. The satellite's inertia matrix below was chosen by researching the inertia matrix of the Hubble Space Telescope, then simplifying its quantities for a circular orbit accurate for this satellite:

$$J_{sat}^{b} = \begin{bmatrix} 1500 & 0 & 0 \\ 0 & 450 & 0 \\ 0 & 0 & 700 \end{bmatrix} kg\ m^2 \tag{6}$$

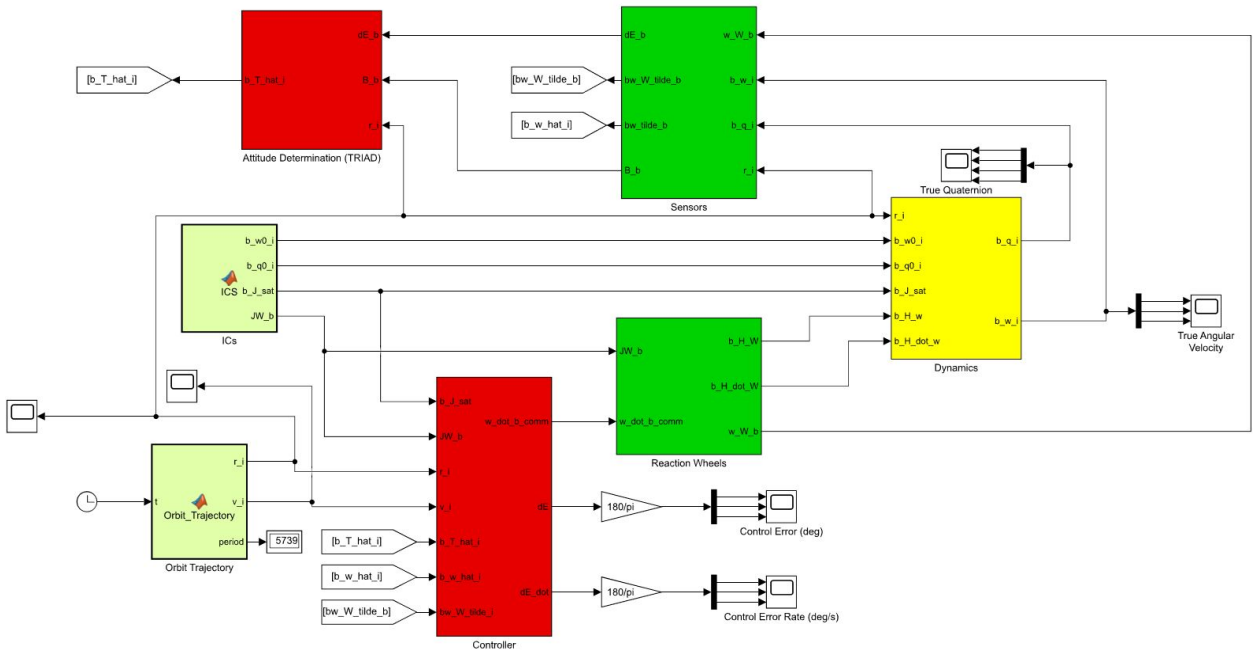
An initial angular velocity error in the body x-axis will be assumed to be 0.00255 rad/s, which has been chosen using the difference between the Hubble Space Telescope's initial angular velocity from NASA NTRS's report (see references) and the nominal angular velocity chosen for this satellite. An initial attitude pointing error will be assumed to be 1.35 degrees around the body y-axis, similar to the Hubble Space Telescope's report on pointing error reported.

The satellite will use three-axis attitude control reaction wheels (RW), as they do not require external applicators of torque. They also provide low pointing error so that the satellite can stay accurately pointed in the desired direction. Just like the Hubble Space Telescope, this satellite is designed to have a long lifespan, so reaction wheels are more appropriate than actuators that would need stored fuel. The mass of the satellite is assumed to be 830kg.

Earth's magnetic field can be modeled by a simple dipole model where:

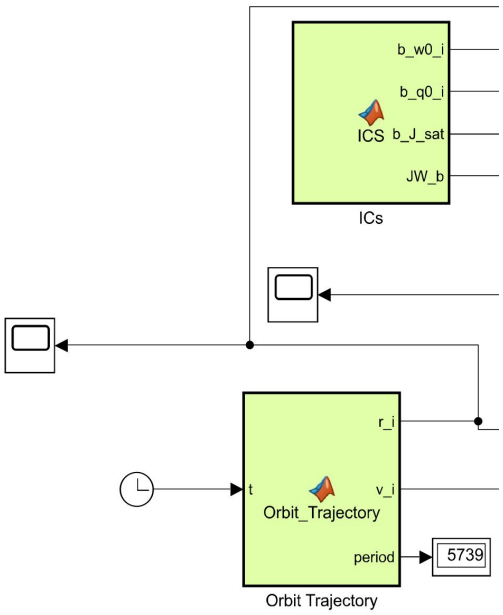
$$\begin{gathered} Magnetic\ Field\ at\ Equator \qquad B_0 = 3.12 * 10^{-5}\ Tesla \\ \lambda = latitude \\ Equitorial\ Radius \qquad R_e = 6371\ km \end{gathered} \tag{7}$$

Complete Simulink Model



The complete Simulink model shown above is compartmentalized using different colors to represent their role in stabilizing the satellite. All “Matlab function” blocks are shaded a light green to indicate they contain coding. This code can be seen in Appendix I. The Dynamics model is shaded yellow, while the sensor/actuator system is shaded green. The ADCS contains both the controller and attitude determination (TRIAD) and is shaded red. The model shown in this report has the noise switch off for simplicity, however the results when switched on are detailed in the Simulation section.

ICs and Orbit Trajectory



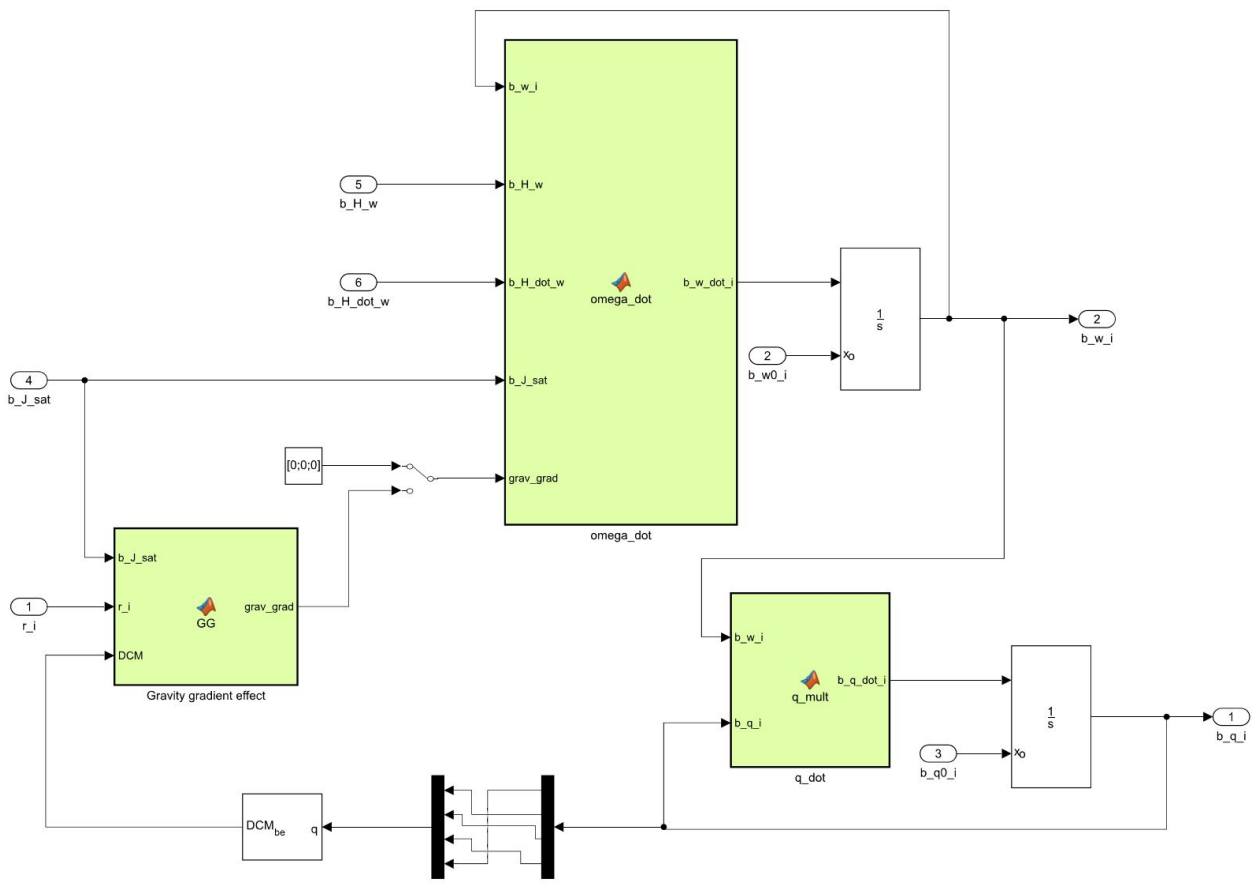
The ICs and orbit trajectory functions come directly from the reference mission section of this report. The instantaneous center of zero velocities (ICs) calculated the initial quaternion of the satellite in the body frame with respect to the inertial frame, by first calculating the DCM and then transforming it into quaternion form. This is to limit the number of restrictions while increasing attitude accuracy.

The inertia matrix of the satellite and angular velocity was assumed in the reference mission. The inertia of the wheel (JW_b) is an appropriate value picked to minimize the torque experienced in the controller and reaction wheel subsystems. This value is 1% of the satellite's weight.

The orbit trajectory function uses time as an input in seconds to calculate the position and velocity of the satellite for one full orbit at an inclination of 30 degrees. The period output displays the time taken, in seconds, for one full circular orbit. The period was calculated by:

$$Period = 2\pi\sqrt{\frac{a^3}{\mu}} \qquad where\ a = \rho \tag{8}$$

Dynamics Model



The dynamics model above calculates the actual quaternion and angular velocity vectors after each iteration. The rate of each is integrated using the initial angular velocity and quaternion to output the values. The dynamics subsystem is directly experiencing the torques and adjusting to provide the correct attitude to stay on track. The angular velocity can be calculated from:

$$\dot{\omega}_i^b = {J_{sat}^b}^{-1}\left(\omega_i^b \times J_{sat}^b \omega_i^b + H_w^b + M_{cm}\right) \tag{9}$$

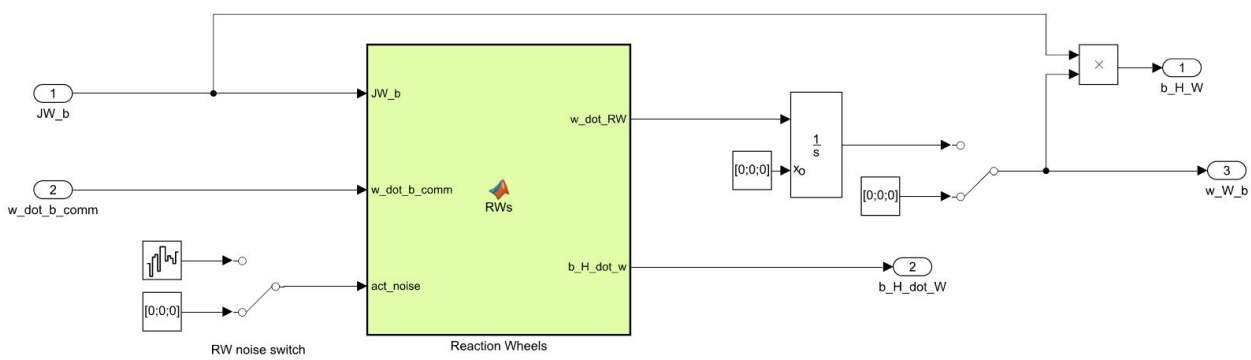
Where M_{cm} is the gravity gradient torque, and H is the momentum.

The gravity gradient torque has a large effect on the angular velocity which in turn has a large effect on the reaction wheels, as they have to keep the angular velocity correct to keep the satellite oriented correctly. The gravity gradient is constantly changing due to the satellites inclination and inertial position; it is critical that it is accounted for. The gravity gradient can be found by:

$$M_{cm} = 3\omega_i^{b^3}\hat{r}_i \times J_{sat}^b\hat{r}_i \tag{10}$$

Sensors

Reaction Wheels



The reaction wheels in this satellite are the primary mechanism responsible for correcting the satellite's attitude, instead of propellant, as they use momentum to orient the satellite. They are designed to operate at a constant speed to reduce the electrical input required. The velocity and momentum of the reaction wheels are:

$$\tilde{\dot{\omega}}_i^b = Act^N + \dot{\omega}_{comm}^b \tag{11}$$

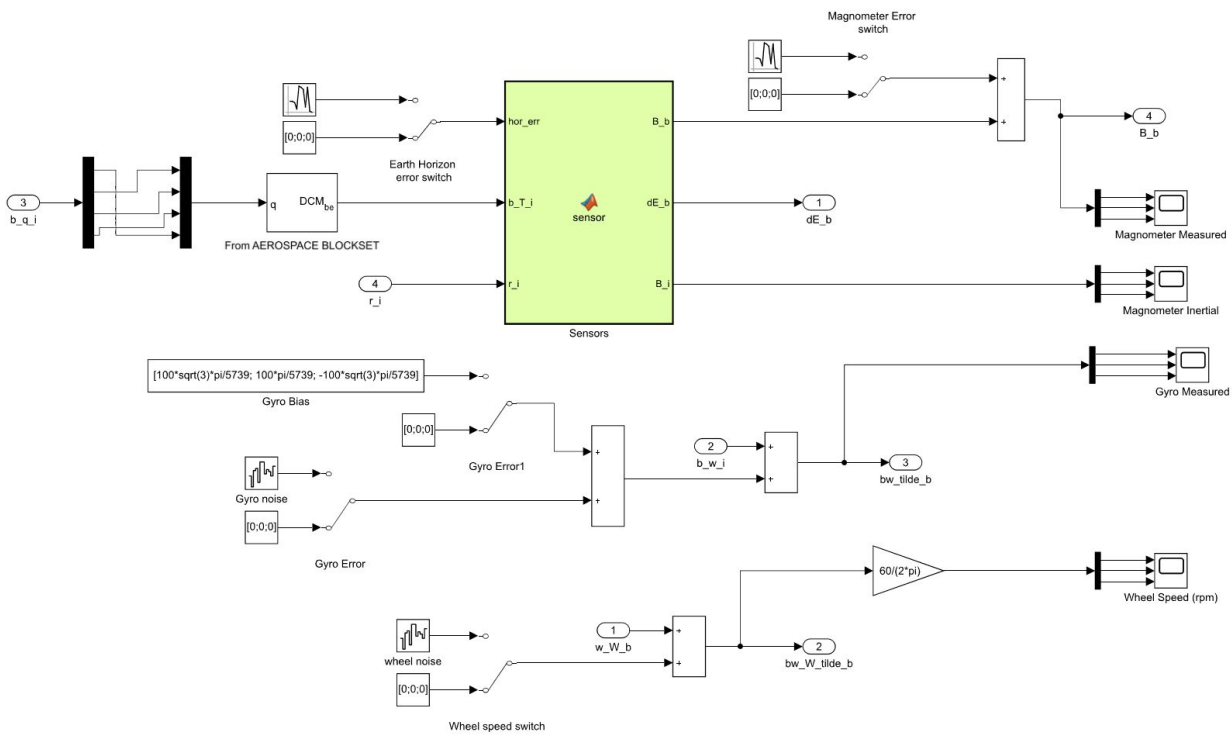
$$H_w^b = J_{sat}^b\tilde{\dot{\omega}}_i^b \tag{12}$$

Where ACT^N is the random actuator noise of the reaction wheels.

The reaction wheel noise error is displayed below:

- Actuator Noise: Power: 1E-5

Sensor Subsystem



The sensor subsystem creates all of the noise and error, incorporating it into the controller and hence to the dynamics and ADCS subsystems. All the noise above was chosen appropriately, but it was chosen to be smaller than the gravity gradient torque noise since this is a LEO orbit satellite. The sensors take the angular velocity, quaternion, and inertial position and add the noise to better simulate what a LEO satellite would experience in its orbit.

The body and inertial torques due to Earth's magnetic field can be calculated via:

$$\boldsymbol{B}^n = B_0 \left(\frac{R_e}{\|\boldsymbol{r}\|}\right)^3 \begin{bmatrix} \cos(\lambda) \\ 0 \\ -2\sin(\lambda) \end{bmatrix}$$
$$\boldsymbol{B}^i = (\boldsymbol{T}_i^n)^{\mathrm{T}}\, \boldsymbol{B}^n \tag{13}$$

$$\tilde{\boldsymbol{B}}^b(t) = \boldsymbol{T}_i^b(t)\, \boldsymbol{B}^i(t) + \boldsymbol{\nu}_{mag}(t) \tag{14}$$

Where V_{mag} is the magnetometer error.

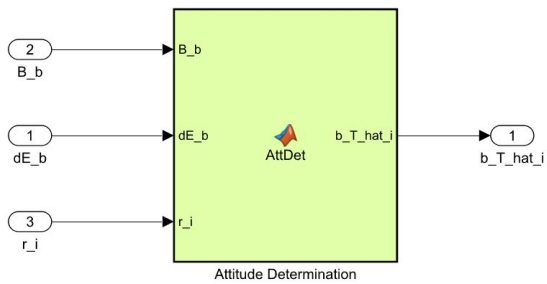
This subsystem has noise/error manual switches for the gyro, the magnetometer, and the reaction wheels, so that the induced noise can be compared to the simulation without noise. The difference between Run 1 and Run 3 is very evident. (See simulations for more information).

Below is a list of assumed noise errors to keep the satellite at the correct attitude:

- Magnetometer random number: variance = 1E-4 mean = 0
- Earth horizon random number: variance = 5E-4 mean = 0
- Gyro Bias: constant = [100*sqrt(3)*pi/5739; 100*pi/5739; -100*sqrt(3)*pi/5739]
- Gyro Noise: Power = [0.01;0.01;0.01]*.01
- Wheel Noise: Power = [0.01;0.01;0.01]*.01

ADCS

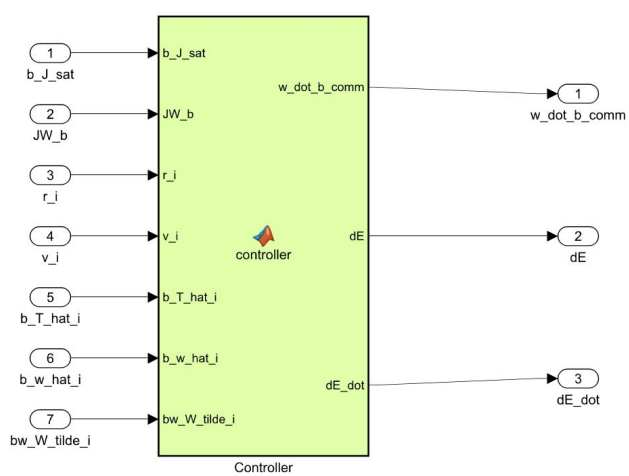
Attitude Determination



The attitude determination subsystem is relatively small compared to the rest of the simulation blocks. It's only composed of a Matlab function that uses the magnetic field torques and the inertial position to produce a DCM matrix for the controller to interpret. The magnetic field torque (B_b) and rate (dE_b) use the estimated quaternion and angular velocity in the sensor and output the AttDet Matlab function, so that the controller subsystem can orient the satellite correctly. Using equations (13) and (14), the unit DCM matrix can be found by:

$$\hat{T}_i^b = [x_b \quad y_b \quad z_b] \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \tag{15}$$

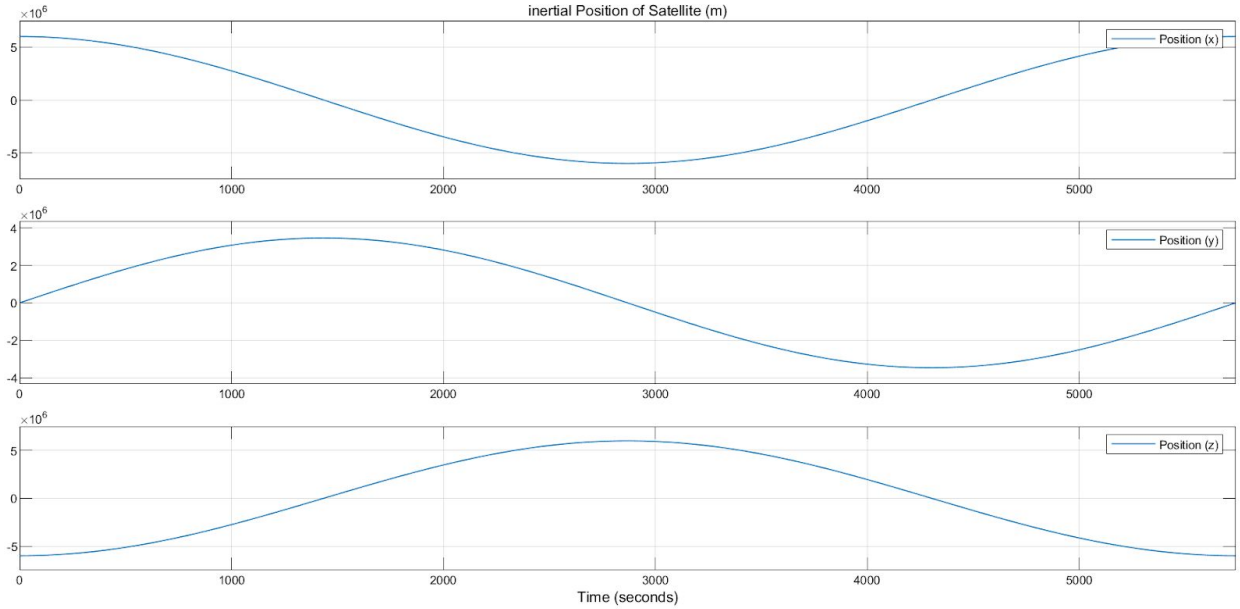
Controller

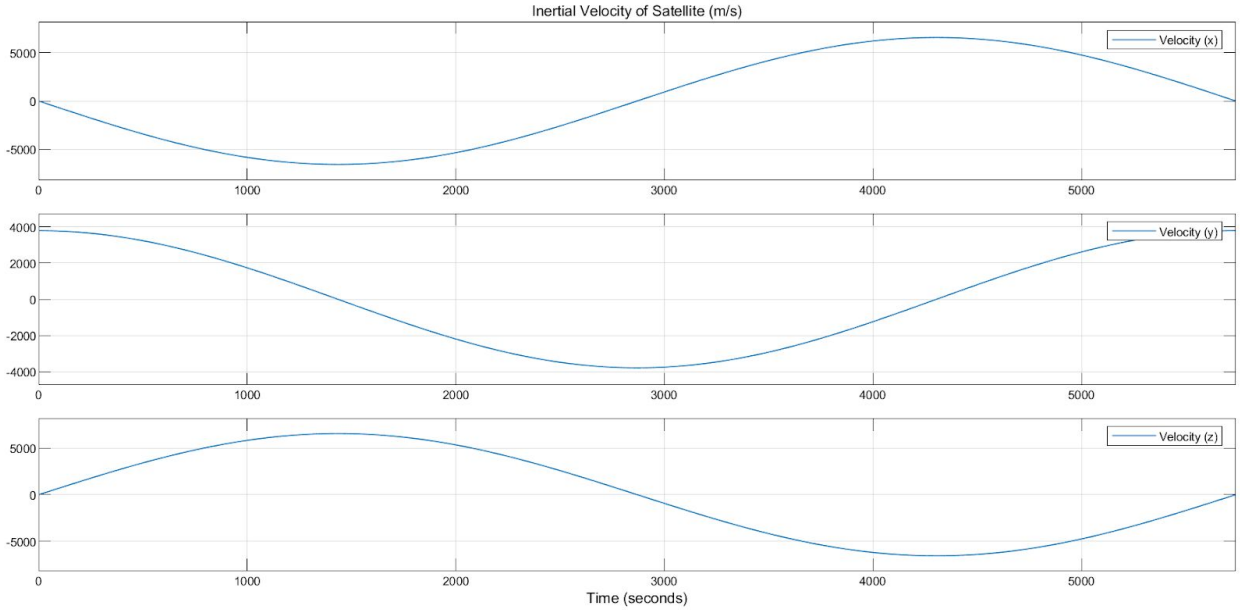


The controller subsystem also consists of one Matlab function block. The controller takes the satellite's inertial position and velocity, and it converts them into the position of the satellite in the body frame with respect to the inertial frame. Kd and Kp are constants both chosen to have a numerical value of 0.5, as those were the appropriate values tested to keep the satellite oriented correctly.

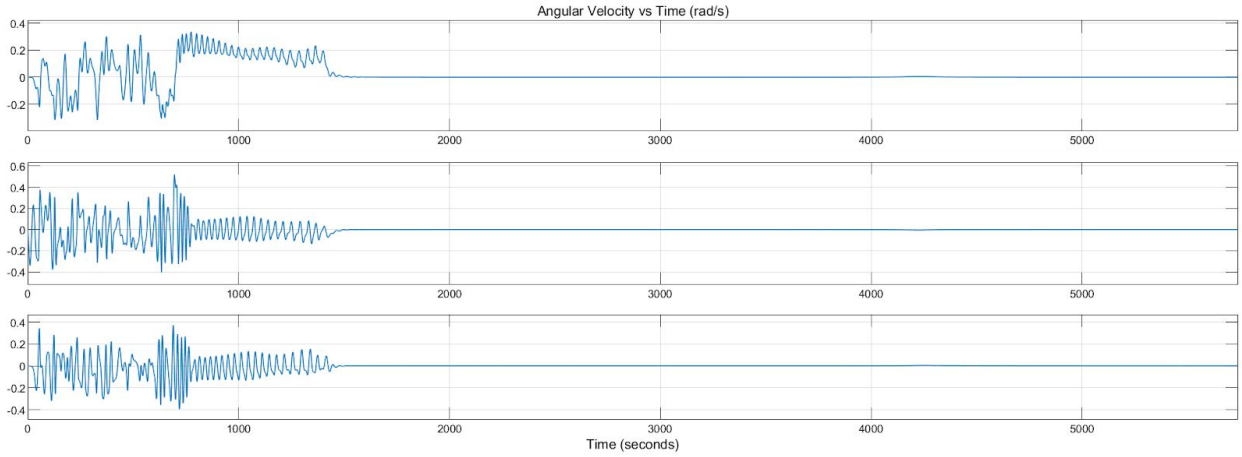
Simulation Results

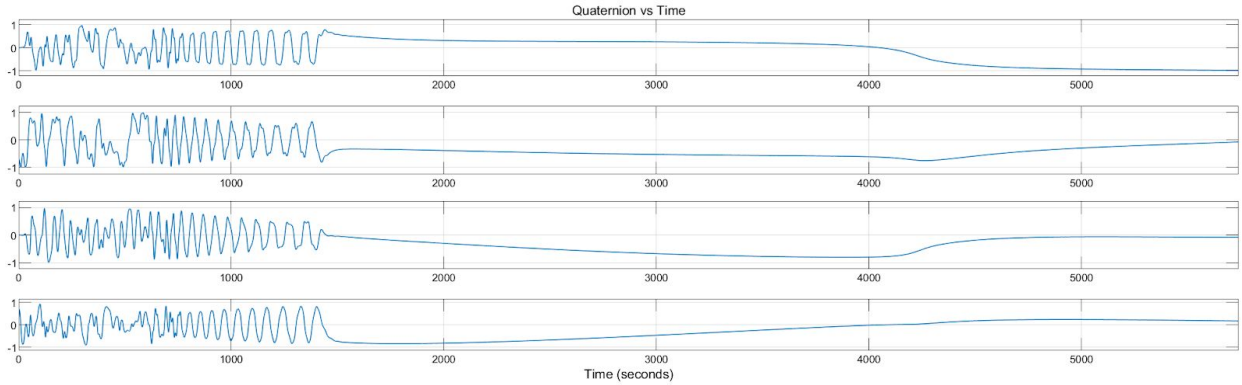
Orbit Trajectory



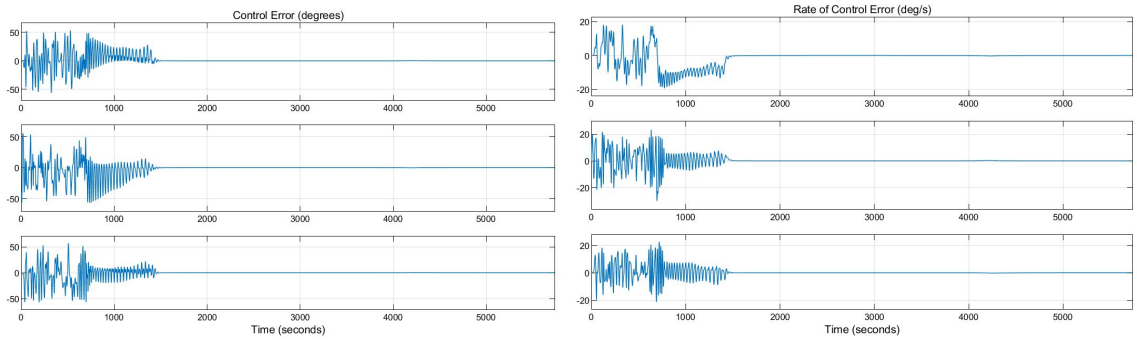


Simulation 1: No Noise, Torque, RW Velocity

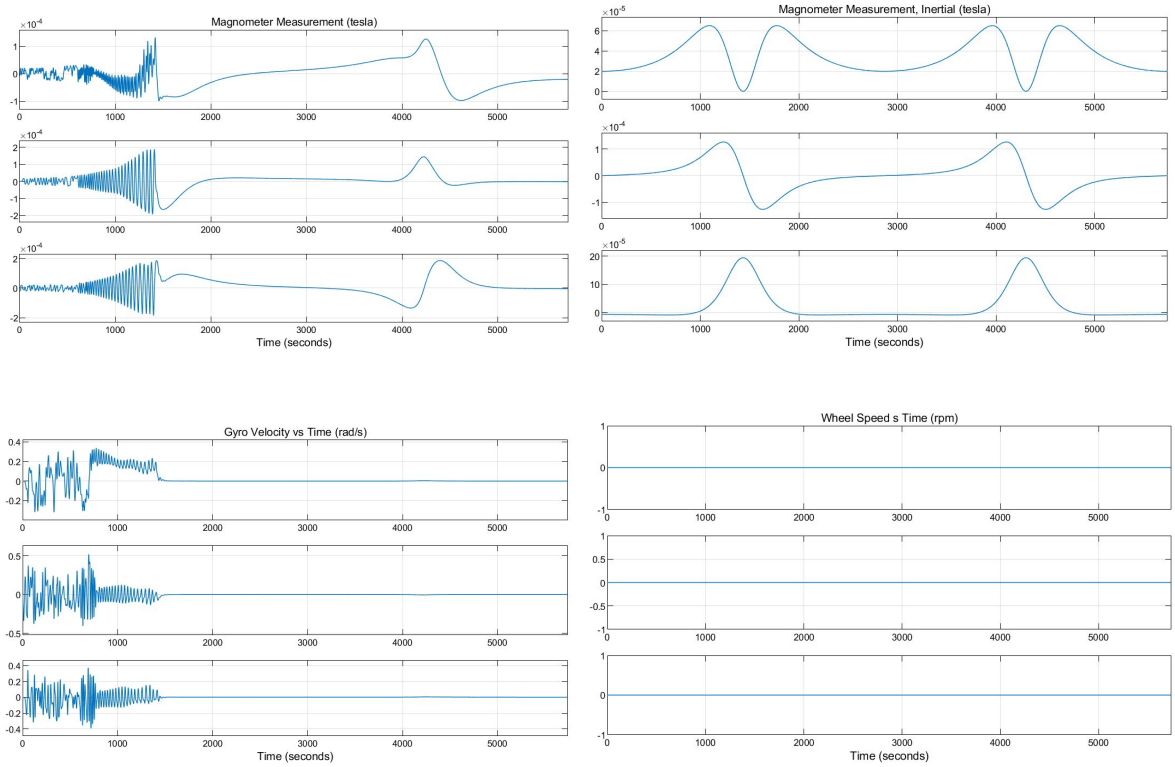




In Simulation 1, the angular velocity changes at a relatively quick rate until around 1500 seconds, when the desired attitude of the satellite is met.. The quaternion vector acts similarly to the angular velocity until around 4500 seconds, where it has a slight adjustment.

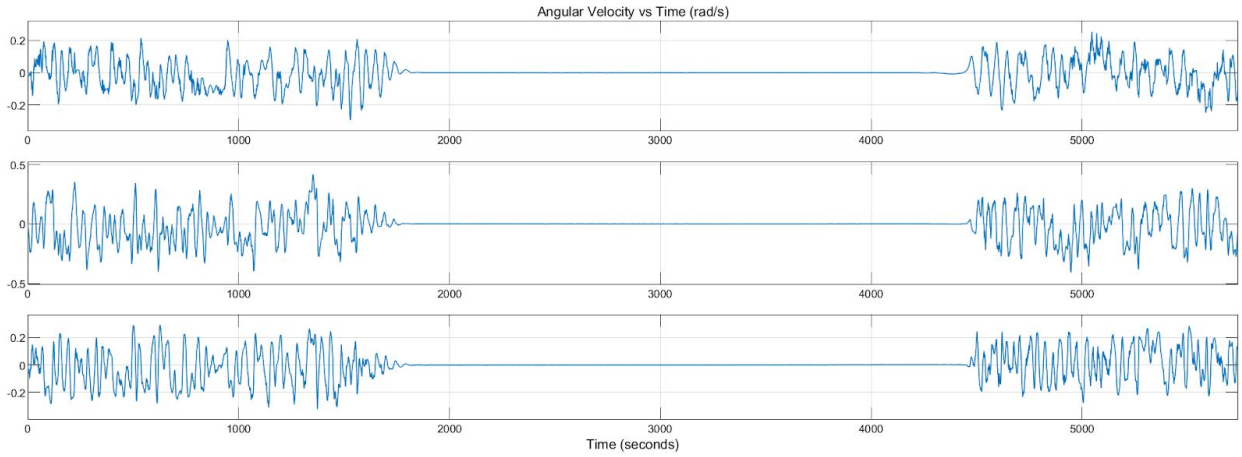


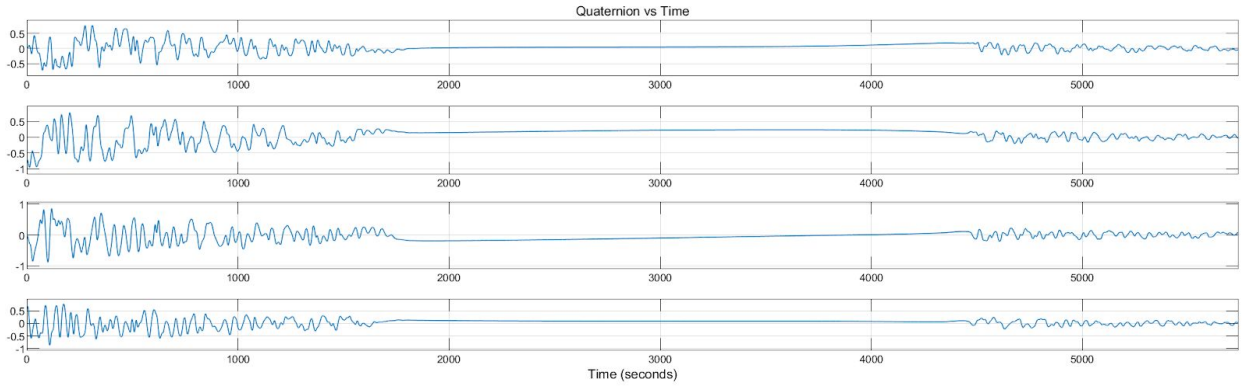
The control error and rate of control error again have the same change in movement as the angular velocity and quaternion, as the controller is trying to correct the attitude of the satellite. Any higher Kp and Kd values would have been too much natural momentum to overcome.



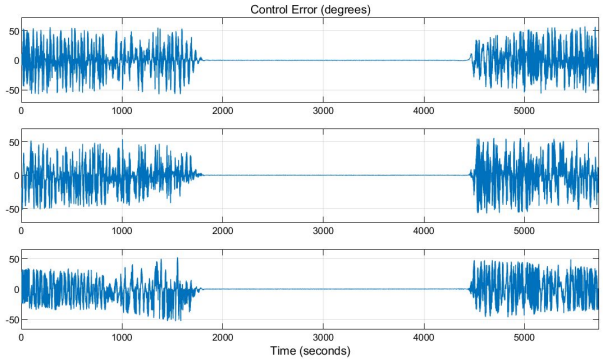
The magnetometer measurements and gyro velocity measurements follow the same trend as the first four graphs, as they are directly related to the satellite's actual angular velocity and quaternion from the dynamics subsystem. The inertial magnetometer measurement, however, does not change because it is in the inertial frame, not the body frame of the satellite itself, which is more dependent on the momentum of the center of mass of the satellite. The wheel speed measures zero, as for this simulation they were not accounted for to provide nominal results.

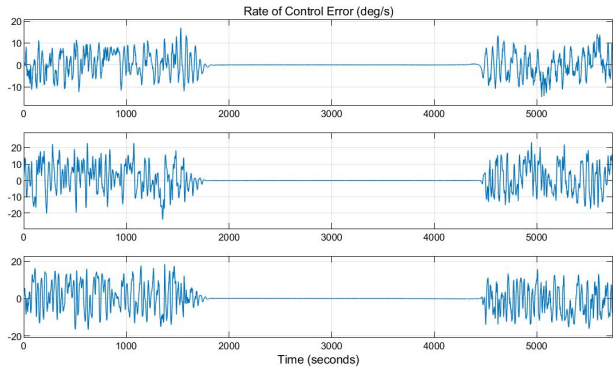
Simulation 2: Only Torque On



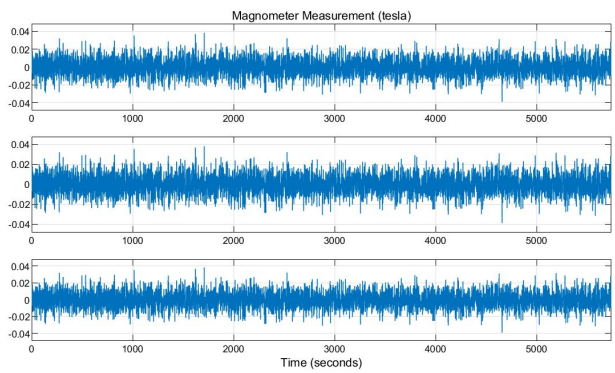


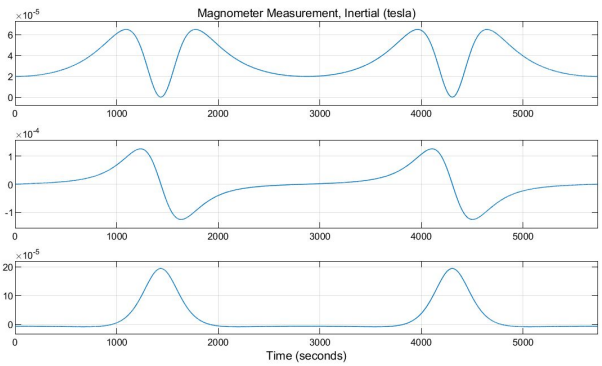
For this simulation, it is clear that torque plays a role in the satellite's quaternion orientation and angular velocity. While again the angular velocity and quaternion tend to be constant and smooth during the middle of the satellite's period, they change more rapidly at the start and end. This is most likely due to one period, over different parts of the Earth and its changing gravity gradient and magnetic field torques. If the simulation is run over multiple periods, this result is consistently oscillatory.

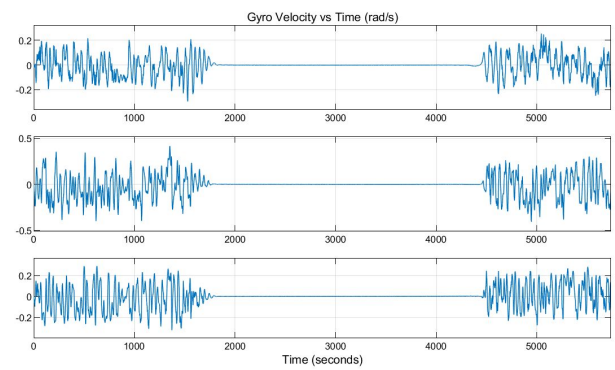


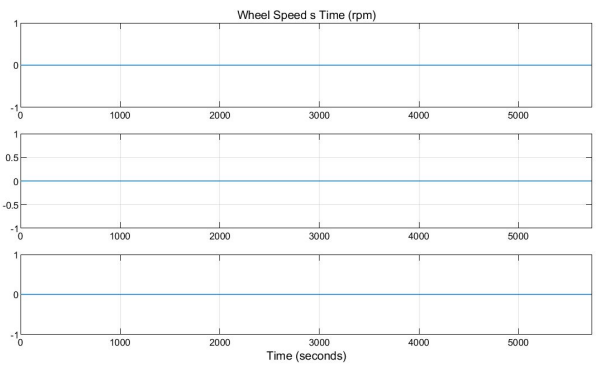


The controller error again follows the same behavior as the angular velocity and quaternion, suggesting that they are directly related. The control error and error rate have much more rapid value changes, however this is expected so that the satellite doesn't veer off course.



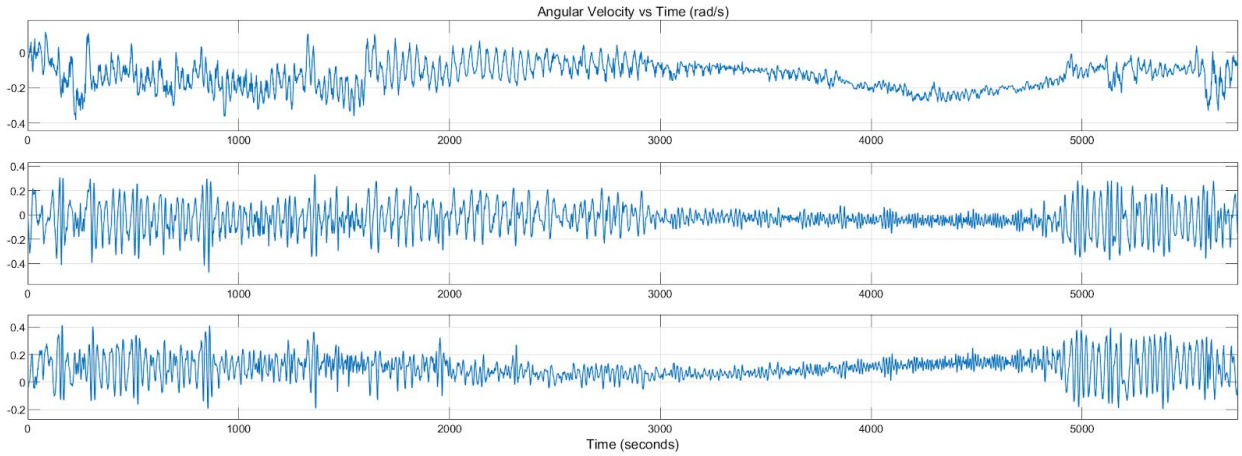


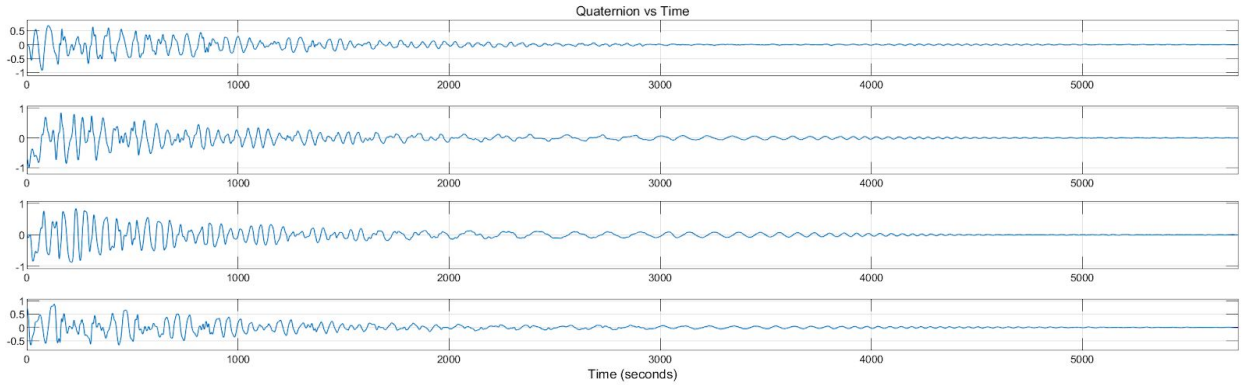




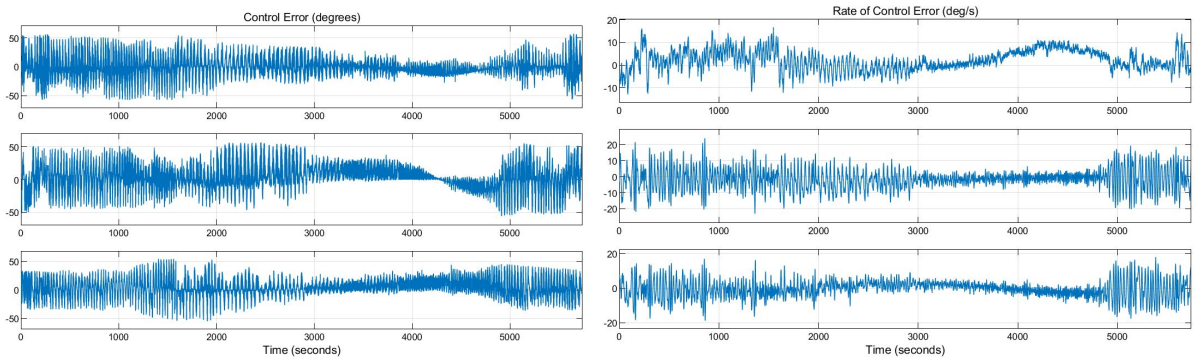
The magnetometer is directly related to the magnetic field torque due to the Earth, so it is expected for the measurements to be changing rapidly and consistently throughout the orbit. Again, the inertial magnetometer measurement does not change, which is also expected. The gyro velocity follows the same behavior as the angular velocity and quaternion.

Simulation 3: Whole Simulation

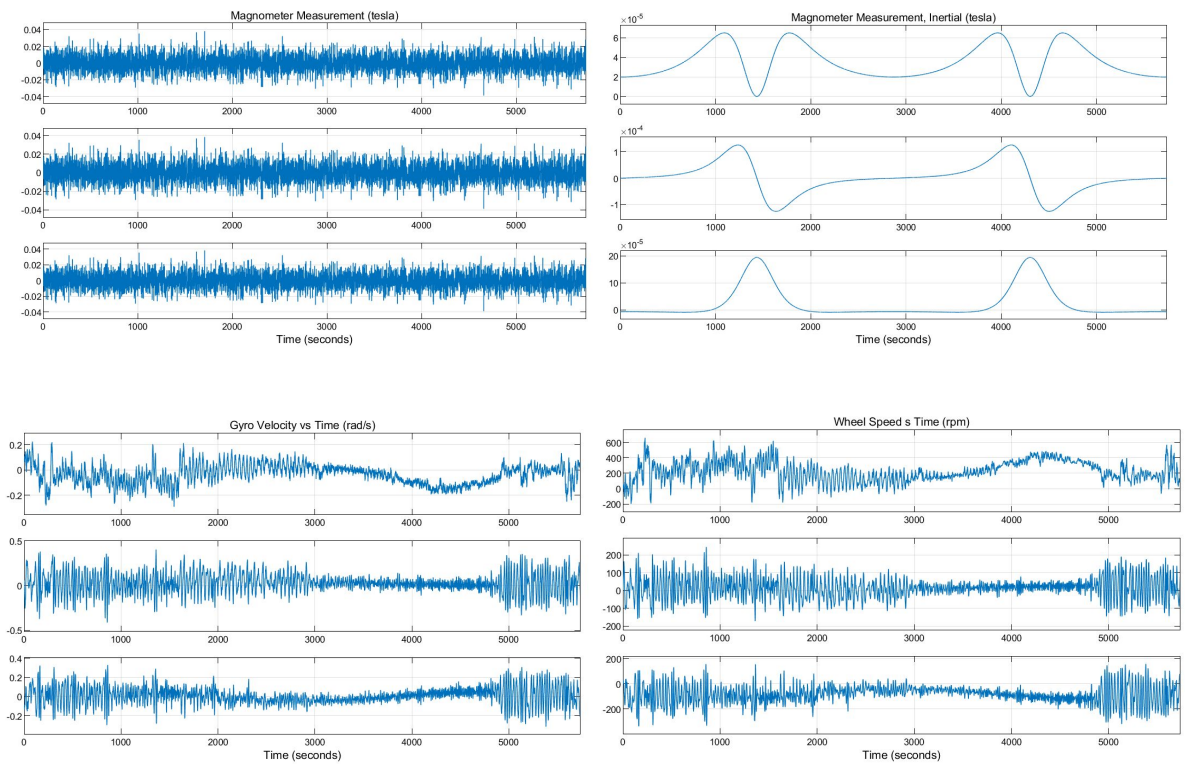




Simulation 3 displays more intense results due to all the added noise and error. The angular velocity can still be interpreted to follow the same motion and behavior as in Simulation 2, yet now noise is incorporated. Over multiple periods the angular velocity displays a behavior of oscillatory motion as expected. The quaternion better handles the noise than the angular velocity, as they are not directly related. The quaternion uses the angular velocity to better orient the satellite to its desired position.



Just as the angular velocity, the controller error and rate of error display the same motion and behavior, as they are not directly affected by all the noise. They display the error between the predicted and actual angular velocity and quaternion.



The magnetometer and inertial magnetometer are very similar to their graphs in Simulation 2. The only addition was the Earth horizon noise, which slightly offsets the measurements but not significantly enough. The gyro velocity displays the same shape as expected, as noise will only slightly alter its velocity. The reaction wheel speed is incorporated in

the bottom right graph above and shows a similar behavior to angular velocity. It has more rapid oscillations and higher amplitudes at the start and end of the period, however it seems to be very stable in the middle.

References

- Canadian Space Agency. (2020, January 27). *Satellites in our everyday lives*. Canada.Ca. https://www.asc-csa.gc.ca/eng/satellites/everyday-lives/default.asp
- Tamboli, I. (2020, April 9). *Medium and Large Satellite* [Photograph]. OpenPR. https://www.openpr.com/news/2005604/medium-and-large-satellite-market-2020-industry-future-trends
- NASA NTRS. (2005, January). Hubble Space Telescope Angular Velocity Estimation During the Robotic Servicing Mission. https://ntrs.nasa.gov/api/citations/20050210231/downloads/20050210231.pdf
- NASA NTRS. (2005, January). *Hubble Space Telescope Angular Velocity Estimation During the Robotic Servicing Mission*. https://ntrs.nasa.gov/api/citations/20050210231/downloads/20050210231.pdf
- NASA. (2018, December 18). *About the Hubble Space Telescope*. https://www.nasa.gov/mission_pages/hubble/story/index.html
- *Hubble overview*. (n.d.). [Photograph]. The European Space Agency. https://www.esa.int/Science_Exploration/Space_Science/Hubble_overview

Appendix I: Simulink Code

ICs

```
function [b_w0_i,b_q0_i,b_J_sat,JW_b] = ICS

mass_sat = 830; % kg
JW_b = 0.01*mass_sat; % assuming the required torque to
% remain stable is 1% of the mass of the satellite
b_J_sat = [1500 0 0; 0 450 0; 0 0 700];


rho = 6928.14E3; % earths radius + satellites average attitude in m
mu = 3.986E14; % earths gravitational constant

omega =sqrt(mu/rho^3); % inertial angular velocity in rad/s

r0_i = (rho/sqrt(2)).*[cos(omega*0); sqrt(2)*sin(omega*0); -cos(omega*0)];
% initial intertial position in m
v0_i = (omega*rho/sqrt(2)).*[-sin(omega*0); sqrt(2)*cos(omega*0); sin(omega*0)];
% initial inertial velocity in m/s

b_x_i = r0_i; % pointing away from earth
b_y_i = v0_i; % the direction of the velocity
b_z_i = cross(b_x_i, b_y_i); % completing the 3D body frame

b_Tnom_i = [b_x_i b_y_i b_z_i]; % original transformation matrix from b -> i

    function q = DCM_to_q(T) % turning the DCM (or tranformation) matrix
        % into the correct nominal quaternion
        q=zeros(4,1);

        q(1) = .5 * sqrt(abs(1+T(1,1)-T(2,2)-T(3,3)));
        q(2) = .5 * sqrt(abs(1-T(1,1)+T(2,2)-T(3,3)));
        q(3) = .5 * sqrt(abs(1-T(1,1)-T(2,2)+T(3,3)));
        q(4) = .5 * sqrt(abs(1+trace(T)));

        [~,imax] = max(q);

        switch imax(1)
            case 1
                q(2) = (T(1,2)+T(2,1))/4/q(1);
                q(3) = (T(1,3)+T(3,1))/4/q(1);
                q(4) = (T(2,3)-T(3,2))/4/q(1);
            case 2
                q(1) = (T(1,2)+T(2,1))/4/q(2);
                q(3) = (T(2,3)+T(3,2))/4/q(2);
                q(4) = (T(3,1)-T(1,3))/4/q(2);
            case 3
                q(1) = (T(1,3)+T(3,1))/4/q(3);
                q(2) = (T(2,3)+T(3,2))/4/q(3);
                q(4) = (T(1,2)-T(2,1))/4/q(3);
            case 4
                q(1) = (T(2,3)-T(3,2))/4/q(4);
                q(2) = (T(3,1)-T(1,3))/4/q(4);
                q(3) = (T(1,2)-T(2,1))/4/q(4);
        end
    end


    function q3 = q_mult(q1,q2) % for quaternion multiplication
        qv = q1(4)*q2(1:3) + q2(4)*q1(1:3) - cross(q1(1:3), q2(1:3));
        qs = q1(4)*q2(4) - dot(q1(1:3),q2(1:3));
        q3 = [qv; qs];



    end

q_nom = DCM_to_q(b_Tnom_i)
q0 = q_mult([0; sind(1); 0; cosd(1)], q_nom); % initial quaternion
b_q0_i =q0./norm(q0) ;
w_nom = [0; 0; omega];
b_w0_i = [0.00255; 0; 0] + w_nom;

end
```

Orbit Trajectory

```
 function [r_i,v_i,period] = Orbit_Trajectory(t)

rho = 6928.14E3; % earths radius + satellites average attitude in m
mu = 3.986E14; % earths gravitational constant

omega =sqrt(mu/rho^3); % inertial angular velocity in rad/s

r_i = (rho/2).*[sqrt(3)*cos(omega*t); sin(omega*t); -sqrt(3)*cos(omega*t)]; % intertial position in m

v_i = (omega*rho/2).*[-sqrt(3)*sin(omega*t); cos(omega*t); sqrt(3)*sin(omega*t)]; % inertial velocity in m/s

a = rho; % due to circular orbit assumption in m
period = 2*pi*sqrt(a^3/mu); % satellites period in seconds
```

Dynamics

```
function grav_grad = GG(b_J_sat, r_i, DCM)

rho = 6928.14E3; % earths radius + satellites average attitude in m
mu = 3.986E14; % earths gravitational constant

omega =sqrt(mu/rho^3); % inertial angular velocity in rad/s

m_cm = cross(3*omega^3*(r_i./norm(r_i)),b_J_sat*(r_i./norm(r_i)));

grav_grad = DCM*m_cm;
```

```
function b_w_dot_i  =  omega_dot(b_w_i, b_H_w, b_H_dot_w, b_J_sat, grav_grad)

b_w_dot_i = b_J_sat\(-cross(b_w_i,b_J_sat*b_w_i+b_H_w + grav_grad)-b_H_dot_w);
```

```
function b_q_dot_i = q_mult(b_w_i, b_q_i)


qv = b_q_i(1:3);
qs = b_q_i(4);

b_q_dot_i = 0.5*[qs.*b_w_i - cross(b_w_i,qv); -dot(qv,b_w_i)];
```

Sensor

```
function [B_b, dE_b, B_i] = sensor(hor_err, b_T_i, r_i)

lambda = 30*pi/180; % 30 degree inclination as stated in report


n_z_i = r_i./norm(r_i);
n_y_i = cross(n_z_i, [0;0;1])/norm(cross(n_z_i, [0;0;1])); %direction of angular and velocity
n_x_i = cross(n_y_i, n_z_i);

n_T_i = [n_x_i'; n_y_i'; n_z_i']; % DCM from NED to inertial frame

cos_lambda = norm(cross(n_z_i, [0;0;1])); % radians
sin_lambda = r_i(3)/norm(r_i); % radians

B0 = 3.12E-5; %Magentic Field at the equator (tesla)
Re = 6371E3; % radius of earth at equator (m)

B_n = (B0*(Re/norm(r_i))^3).*[cos_lambda; 0; -2*sin_lambda]; %mag field in NED frame
B_i = n_T_i' *B_n; % magnetic field in inertial frame
B_b = b_T_i*B_i; % magnetic field in body frame

%Now solve for dE_b

n_angle = norm(hor_err);
hor_err_hat = hor_err./n_angle;

skew_hor = [0 -hor_err_hat(3) hor_err_hat(2)
    hor_err_hat(3) 0 -hor_err_hat(1)
    -hor_err_hat(2) hor_err_hat(1) 0];   %skew matrix of the norm of earth horizon error

DCM = eye(3);

if n_angle > 1E-12 % tolerance level
    T_hor = DCM - sin(n_angle)*skew_hor + (1-cos(n_angle))*skew_hor^2;
else
    T_hor = eye(3);
end

dE_b = T_hor*b_T_i*n_z_i;

end
```

```
function [w_dot_RW, b_H_dot_w] = RWs(JW_b, w_dot_b_comm, act_noise)


w_dot_RW = act_noise + w_dot_b_comm; %rad/s^2

b_H_dot_w = JW_b.*w_dot_RW; % rate of momentum
```

ADCS

```
function b_T_hat_i = AttDet(B_b, dE_b, r_i)


lambda = 30*pi/180; % 30 degree inclination as stated in report


n_z_i = r_i./norm(r_i);
n_y_i = cross(n_z_i, [0;0;1])/norm(cross(n_z_i, [0;0;1])); %direction of angular and velocity
n_x_i = cross(n_y_i, n_z_i);

n_T_i = [n_x_i'; n_y_i'; n_z_i']; % DCM from NED to inertial frame

cos_lambda = norm(cross(n_z_i, [0;0;1])); % radians
sin_lambda = r_i(3)/norm(r_i); % radians

B0 = 3.12E-5; %Magentic Field at the equator (tesla)
Re = 6371E3; % radius of earth at equator (m)

B_n = (B0*(Re/norm(r_i))^3).*[cos_lambda; 0; -2*sin_lambda]; %mag field in NED frame
B_i = n_T_i' *B_n; % magnetic field in inertial frame

%essentially same code as the sensors function

x_i = n_z_i;
x_b = dE_b;
z_i = cross(x_i, B_i)/norm(cross(x_i, B_i));
z_b = cross(x_b, B_b)/norm(cross(x_b, B_b));
y_i = cross(z_i, x_i);
y_b = cross(z_b, x_b);

b_T_hat_i = [x_b y_b z_b]*[x_i y_i z_i]';




function [w_dot_b_comm,dE,dE_dot] = controller(b_J_sat, JW_b, r_i, v_i, b_T_hat_i, b_w_hat_i, bw_W_tilde_i)


rho = 6928.14E3; % earths radius + satellites average attitude in m
mu = 3.986E14; % earths gravitational constant

omega =sqrt(mu/rho^3); % inertial angular velocity in rad/s

b_x_i = r_i./norm(r_i);
b_z_i = cross(b_x_i,v_i)./norm(cross(b_x_i,v_i));
b_y_i = cross(b_z_i, b_x_i);

w_nom = [0;0;omega]; % nominal angular velocity
b_Tnom_i = [b_x_i b_y_i b_z_i]; % nominal DCM from b to i

%partial angular velocity and dcm
dw = w_nom - b_w_hat_i;
dT = b_Tnom_i*b_T_hat_i;

dE = -0.5.*[dT(3,2)-dT(2,3); dT(1,3)-dT(3,1); dT(2,1)-dT(1,2)];
dE_dot = -cross(w_nom, dE) + dw;

kd = 0.05; %chosen values
kp = 0.05;

u = kp.*dE+kd.*dE_dot;

M = b_J_sat*u; % torque on controller

w_dot_b_comm = -M./JW_b - cross(b_w_hat_i, bw_W_tilde_i); % reaction wheel control
```

Appendix II: Instructions for Running Code

1. Open the Simulink file provided in the submission folder.
2. To run Simulation 1, turn off <u>all</u> manual switches, to provide zero values, within the following subsystems:
    - Sensors (green)
    - Dynamics (yellow)
    - Reaction Wheels (green)
3. To run Simulation 2, turn on the "Earth Horizon Switch" within the Sensor subsystem block and the "grav_grad" switch within the dynamics subsystem.
4. To run Simulation 3, turn on all switches within the following subsystems:
    - Sensors (green)
    - Dynamics (yellow)
    - Reaction Wheels (green)
5. All scopes (graphs) are set up in vectors of three once clicked.