Dynamics and Control Of Spacecraft
Astrosat Controller Design

Kavitha Sivaraman - AE16M006
Prem Sagar S - AE14B021
Rishi - AE17D402
Daniel Satke - AE17F010
Abdul Azeez - AE14B028

November 22, 2017

Chapter 1

Kinematics and Dynamics of Astrosat

1.1 Introduction

We are planning to design a preliminary attitude control system for ASTROSAT satellite. Since this satellite is an inertial pointing satellite, we have chosen to stabilize the satellite using zero momentum biased stabilization method with 4 wheel tetrahedron configuration. Star sensors senses and provide the values of roll, pitch and yaw Euler angles along with gyroscopes. Our design also includes the momentum dumping, as and when any of the wheels reach their maximum angular rate capacity.

We have initially derived the kinematic equations of motion using 2-3-1 rotation sequence to get the Euler angles. Euler angles are then converted to quaternions before integration to avoid any singularities like gimbal lock phenomenon. Dynamic equations of motion for inertially pointing satellite, with zero momentum biased system configuration is derived using Euler equations. Gravity gradient torque is also considered as disturbance torque, apart from the disturbance torque given in the problem which is in the order of milli N-m. Since our's is an inertially pointing satellite, our roll,pitch and yaw equations of motion will be decoupled.

We have studied the motion of satellite, in roll, pitch and yaw, found that the solutions are unstable in all the 3 DOF in the presence of disturbances and hence we have concluded to go for active stabilization technique based on Proportional-Derivative (PD) controller using momentum wheels in tetrahedral configuration. The gains for our control system design are chosen in such a way that the maximum allowable steady state error does not exceed 0.005 deg about all the axes (zero momentum biased system).

1.2 Astrosat

It is an inertial pointing spacecraft in a 650km 6 deg inclined circular orbit; 4-wheel tetrahedron wheel system is used for momentum management; Actual MI properties of the s/c after deployment.

$$J_c = \begin{bmatrix} 1763 & -52 & -16 \\ -52 & 1591 & 25 \\ -16 & 25 & 1185 \end{bmatrix} kg - m^2$$

Mass of the s/c = 1542 Kg. [x,y,z] axes are yaw, roll and pitch respectively. Initially we have assumed that the cross product of inertias are negligible and designed the control system. Then once the control design is completed, we have used the actual inertia matrix to check the performance variations. The momentum dumping is achieved by using 60 A-m^2 torque rods about all the three axes. Disturbance torques are $T_x = T_z = 2 * 10^{-3}$ Nm and $T_y = 10^{-4}$ Nm and satellite's orbital angular velocity, $\omega_o = 1.0741 * 10^{-3}$rad/sec

1.3 Kinematics

We are choosing 2-3-1 (ψ,θ, ϕ) rotation for our kinematics. The angle rates and angular velocity can be related as below,

$$\begin{bmatrix}\dot{\phi}\\ \dot{\psi}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}1 & -cos\phi tan\theta & sin\phi tan\theta\\ 0 & cos\phi sec\theta & -sin\phi sec\theta\\ 0 & sin\phi & cos\phi\end{bmatrix}\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} \tag{1.3.1}$$

After small angle approximations,

$$\begin{bmatrix}\dot{\phi}\\ \dot{\psi}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} \tag{1.3.2}$$

1.4 Dynamics

Since we are using a zero momentum biased control system, the following holds true. Where $\vec{h_i}$ are the wheel angular momentum.

$$\vec{h}_1 + \vec{h}_2 + \vec{h}_3 + \vec{h}_4 = 0 \tag{1.4.1}$$

Also each of XYZ components must be zero.

$$h_{wx} = \frac{h_1}{\sqrt{3}} - \frac{h_2}{\sqrt{3}} - \frac{h_3}{\sqrt{3}} + \frac{h_4}{\sqrt{3}} = 0 \tag{1.4.2}$$

$$h_{wy} = \frac{h_1}{\sqrt{3}} - \frac{h_2}{\sqrt{3}} + \frac{h_3}{\sqrt{3}} - \frac{h_4}{\sqrt{3}} = 0 \tag{1.4.3}$$

$$h_{wz} = \frac{h_1}{\sqrt{3}} + \frac{h_2}{\sqrt{3}} - \frac{h_3}{\sqrt{3}} - \frac{h_4}{\sqrt{3}} = 0 \tag{1.4.4}$$

The dynamics of the wheels is governed by,

$$\frac{d}{dt}\left(\vec{H}_w\right) = -\vec{T}_c$$

where $\vec{T}_c$ is the control torque.
The Euler equations for the spacecraft system can be rewritten as below,

$$J_c\dot{\omega} + \omega \times (J_c\omega) = T_c + T_d \tag{1.4.5}$$

$$\begin{aligned} &J_x\dot{\omega}_x + \frac{I_w\dot{\omega}_1}{\sqrt{3}} - \frac{I_w\dot{\omega}_2}{\sqrt{3}} + \frac{I_w\dot{\omega}_3}{\sqrt{3}} - \frac{I_w\dot{\omega}_4}{\sqrt{3}}\\ &-\omega_z\left(J_y\omega_y + \frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} + \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right)\\ &+\omega_y\left(J_z\omega_z + \frac{I_w\omega_1}{\sqrt{3}} + \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0 \end{aligned} \tag{1.4.6}$$

$$\begin{aligned} &J_y\ddot{\omega}_y + \frac{I_w\dot{\omega}_1}{\sqrt{3}} + \frac{I_w\dot{\omega}_2}{\sqrt{3}} - \frac{I_w\dot{\omega}_3}{\sqrt{3}} + \frac{I_w\dot{\omega}_4}{\sqrt{3}}\\ &+\omega_z\left(J_x\omega_x + \frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} + \frac{I_w\omega_4}{\sqrt{3}}\right)\\ &-\omega_x\left(J_z\omega_z + \frac{I_w\omega_1}{\sqrt{3}} + \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0 \end{aligned} \tag{1.4.7}$$

$$
\begin{aligned}
&J_z\ddot{\omega}_z + \frac{I_w\dot{\omega}_1}{\sqrt{3}} - \frac{I_w\dot{\omega}_2}{\sqrt{3}} - \frac{I_w\dot{\omega}_3}{\sqrt{3}} + \frac{I_w\dot{\omega}_4}{\sqrt{3}} \\
&-\omega_y\left(J_x\omega_x + \frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} + \frac{I_w\omega_4}{\sqrt{3}}\right) \\
&+\omega_x\left(J_y\omega_y + \frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} + \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0
\end{aligned} \tag{1.4.8}
$$

After doing small angle approximations the euler equations can be written as follows,

$$
\begin{aligned}
&J_x\ddot{\phi} + \frac{I_w\dot{\omega}_1}{\sqrt{3}} - \frac{I_w\dot{\omega}_2}{\sqrt{3}} + \frac{I_w\dot{\omega}_3}{\sqrt{3}} - \frac{I_w\dot{\omega}_4}{\sqrt{3}} \\
&-\dot{\theta}\left(\frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} + \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) \\
&+\dot{\psi}\left(\frac{I_w\omega_1}{\sqrt{3}} + \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0
\end{aligned} \tag{1.4.9}
$$

$$
\begin{aligned}
&J_y\ddot{\psi} + \frac{I_w\dot{\omega}_1}{\sqrt{3}} + \frac{I_w\dot{\omega}_2}{\sqrt{3}} - \frac{I_w\dot{\omega}_3}{\sqrt{3}} + \frac{I_w\dot{\omega}_4}{\sqrt{3}} \\
&+\dot{\theta}\left(\frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} + \frac{I_w\omega_4}{\sqrt{3}}\right) \\
&-\dot{\phi}\left(\frac{I_w\omega_1}{\sqrt{3}} + \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0
\end{aligned} \tag{1.4.10}
$$

$$
\begin{aligned}
&J_z\ddot{\theta} + \frac{I_w\dot{\omega}_1}{\sqrt{3}} - \frac{I_w\dot{\omega}_2}{\sqrt{3}} - \frac{I_w\dot{\omega}_3}{\sqrt{3}} + \frac{I_w\dot{\omega}_4}{\sqrt{3}} \\
&-\dot{\psi}\left(\frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} - \frac{I_w\omega_3}{\sqrt{3}} + \frac{I_w\omega_4}{\sqrt{3}}\right) \\
&+\dot{\phi}\left(\frac{I_w\omega_1}{\sqrt{3}} - \frac{I_w\omega_2}{\sqrt{3}} + \frac{I_w\omega_3}{\sqrt{3}} - \frac{I_w\omega_4}{\sqrt{3}}\right) = 0
\end{aligned} \tag{1.4.11}
$$

Chapter 2

Controller Design

We will be using the modified Proportional Derivative (PD) controller.

2.0.1 Modified PD controller

Control Torque

For a general moment of inertia matrix, after including control torques, disturbance torques and the gravity gradient torques, the dynamic equations become:

$$\begin{aligned} &I_{xx}\dot{\omega}_x + I_{xy}\dot{\omega}_y + I_{xz}\dot{\omega}_z + \\ &\omega_y \left(I_{zx}\omega_x + I_{zy}\omega_y + I_{zz}\omega_z\right) \\ &-\omega_z \left(I_{yx}\omega_x + I_{yy}\omega_y + I_{yz}\omega_z\right) = T_{cx} + T_{dx} + T_{gx} \end{aligned} \tag{2.0.1}$$

$$\begin{aligned} &I_{yx}\dot{\omega}_x + I_{yy}\dot{\omega}_y + I_{yz}\dot{\omega}_z + \\ &\omega_z \left(I_{xx}\omega_x + I_{xy}\omega_y + I_{xz}\omega_z\right) \\ &-\omega_x \left(I_{zx}\omega_x + I_{zy}\omega_y + I_{zz}\omega_z\right) = T_{cy} + T_{dy} + T_{gy} \end{aligned} \tag{2.0.2}$$

$$\begin{aligned} &I_{zx}\dot{\omega}_x + I_{zy}\dot{\omega}_y + I_{zz}\dot{\omega}_z + \\ &\omega_x \left(I_{yx}\omega_x + I_{yy}\omega_y + I_{yz}\omega_z\right) \\ &-\omega_y \left(I_{xx}\omega_x + I_{xy}\omega_y + I_{xz}\omega_z\right) = T_{cz} + T_{dz} + T_{gz} \end{aligned} \tag{2.0.3}$$

$$T_c = -\begin{bmatrix} K_{px}(\phi - R_\phi) + K_{dx}\dot{\phi} \\ K_{py}(\psi - R_\psi) + K_{dy}\dot{\psi} \\ K_{pz}(\theta - R_\theta) + K_{dz}\dot{\theta} \end{bmatrix} \tag{2.0.4}$$

2.0.1.1 Gravity Gradient torque:

$$r_b = C_{bo} r \tag{2.0.5}$$

$$\begin{aligned} &= \begin{bmatrix} 1 & \theta & -\psi \\ -\theta & 1 & \phi \\ \psi & -\phi & 1 \end{bmatrix} \begin{bmatrix} -r \\ 0 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} -r \\ \theta r \\ -\psi r \end{bmatrix} \end{aligned} \tag{2.0.6}$$

$$
\begin{aligned}
T_g &= \frac{3\mu}{r^5} r_b^x I r_b && (2.0.7)\\
&= \frac{3\mu}{r^5} \begin{bmatrix} 0 & \psi r & \theta r \\ -\psi r & 0 & r \\ -\theta r & -r & 0 \end{bmatrix} I \begin{bmatrix} -r \\ \theta r \\ -\psi r \end{bmatrix} \\
&= \frac{3\mu}{r^3} \begin{bmatrix} 0 \\ (I_x - I_z)\,\psi \\ (I_x - I_y)\,\theta \end{bmatrix} \\
&= 3\omega_0^2 \begin{bmatrix} 0 \\ (I_x - I_z)\,\psi \\ (I_x - I_y)\,\theta \end{bmatrix} && (2.0.8)
\end{aligned}
$$

2.0.2 Wheel Dynamics

The individual wheel torques are:

$$T_1 = \frac{1}{4}\left(\sqrt{3}T_{cx} + \sqrt{3}T_{cy} + \sqrt{3}T_{cz}\right) \tag{2.0.9}$$

$$T_2 = \frac{1}{4}\left(-\sqrt{3}T_{cx} - \sqrt{3}T_{cy} + \sqrt{3}T_{cz}\right) \tag{2.0.10}$$

$$T_3 = \frac{1}{4}\left(-\sqrt{3}T_{cx} + \sqrt{3}T_{cy} - \sqrt{3}T_{cz}\right) \tag{2.0.11}$$

$$T_4 = \frac{1}{4}\left(\sqrt{3}T_{cx} - \sqrt{3}T_{cx} - \sqrt{3}T_{cx}\right) \tag{2.0.12}$$

The dynamics of wheel **i** are described by the equations below:

$$I\dot{\omega}_i = T_i \tag{2.0.13}$$

$$I\dot{\omega}_i + IC_{w_ib}\dot{\omega} + (\omega_i + C_{w_ib}\omega) \times (I\omega_i + IC_{w_ib}\omega) = T_i$$

$$I\dot{\omega}_i = T_i - IC_{w_ib}\dot{\omega} - (\omega_i + C_{w_ib}\omega) \times (I\omega_i + IC_{w_ib}\omega) \tag{2.0.14}$$

where I is the moment of inertia of the whole spacecraft.

2.0.3 Estimates for K_d and K_p

The system can be represented as a transfer function:

$$G_p(s) = \frac{1}{Is^2 + K_d s} \tag{2.0.15}$$

Assuming a critically damped system,

$$\xi = 1$$

$$K_d^2 = 4IK_p \tag{2.0.16}$$

Closed loop system pole:

$$1 + G_c(s) = 0$$

$$1 + \frac{K_p}{Is^2 + K_d s} = 0$$
$$Is^2 + K_d s + K_p = 0$$

So our spacecraft is a Type 1 system.

$$s^2 + \frac{K_d}{I}s + \frac{K_p}{I} = 0 \tag{2.0.17}$$

$$s^2 + 2\xi\omega_n s + \omega_n^2 = 0 \tag{2.0.18}$$

From this equation, since we know that damping and frequency are positive, K_p and K_d are positive. This is also verified using Routh's stability criterion.

$$\omega_n^2 = \frac{K_p}{I} \tag{2.0.19}$$

$$2\xi\omega_n = \frac{K_d}{I}$$

$$\xi = \frac{K_d}{2\omega_n I} \tag{2.0.20}$$

$$\xi^2 = \frac{K_d^2}{4\frac{K_p}{I}I^2} = \frac{K_d^2}{4K_p I}$$

2.0.4 Steady State errors

Without consideration of disturbance torque, the steady state error is given by,

$$lim_{s\to 0}\frac{sR(s)}{1 + G_0(s)}$$

$$\lim_{s\to 0}\frac{sR(s)}{1 + \left(\frac{1}{Is^2+K_d s}\right)G_c(s)} = \lim_{s\to 0}\frac{s(Is^2 + K_d s)}{Is^2 + K_d s + G_c(s)}R(s)$$
$$= \lim_{s\to 0}\frac{Is + K_d}{I + \frac{K_d}{s} + \frac{G_c(s)}{s^2}}R(s)$$

Test signals $R(s) = \frac{1}{s}$, $R(s) = \frac{1}{s^2}$,$R(s) = \frac{1}{s^3}$,

2.0.4.1 Step Input

$$\lim_{s\to 0}\frac{Is + K_d}{I + \frac{K_d}{s} + \frac{G_c(s)}{s^2}}\frac{1}{s} = \lim_{s\to 0}\frac{Is + K_d}{Is + K_d + \frac{G_c(s)}{s}}$$
$$= \lim_{s\to 0}\frac{Is + K_d}{Is + K_d + \frac{K_p}{s}}$$
$$= 0$$

2.0.4.2 Ramp Input

$$\lim_{s\to 0}\frac{Is + K_d}{I + \frac{K_d}{s} + \frac{G_c(s)}{s^2}}\frac{1}{s^2} = \lim_{s\to 0}\frac{Is + K_d}{Is^2 + K_d s + K_p}$$
$$= \frac{K_d}{K_p}$$

2.0.4.3 Accelerating Input

$$\lim_{s\to 0} \frac{Is + K_d}{I + \frac{K_d}{s} + \frac{G_c(s)}{s^2}} \frac{1}{s^3} = \lim_{s\to 0} \frac{Is + K_d}{Is^3 + K_d s^2 + sK_p}$$
$$= K_d$$

2.1 Momemtum dumping

There is a maximum capacity that each of the wheel is designed for. If by applying control torques multiple times over the operation period, this capacity is reached, the wheel cannot be used and thus, requires the momentum of the entire spacecraft to be dumped using thrusters in order to restore the wheel to working configuration.

- **Criteria 1: Maximum cap on angular rate in the wheel frame.**

Consider the wheel torque on a wheel **i**. In the wheel frame, the moment of inertia of the entire spacecraft is I_t.

$$T_i = I_t \frac{d\omega_t}{dt}$$

$$\omega_t = \omega_i + C_{w_i b}\omega$$

$$\begin{aligned} I_t \frac{d\omega_{tw}}{dt} &= I_t \frac{d\omega_t}{dt} + \omega_t \times I_t \omega_t \\ &= I_t \left(\dot{\omega}_i + C_{w_i b}\dot{\omega}_b + \dot{C}_{w_i b}\omega_b \right) \\ &\quad + (\omega_i + C_{wb}\omega_b) \times [I_t (\omega_i + C_{wb}\omega_b)] \\ &= I_t\dot{\omega}_i + I_t C_{w_i b}\dot{\omega}_b + \omega_i \times I_t (\omega_i + C_{wb}\omega_b) \\ &\quad + C_{wb}\omega_b \times I_t (\omega_i + C_{wb}\omega_b) \end{aligned}$$

All quantities quantities on RHS can be known at any instant from the dynamics. The dumping is done if the angular rate so obtained exceeds the cutoff.

- **Criteria 2: Maximum cap on Torque**

From our wheel dynamics, for a given control torque, each of the component torques given to every wheel is known. If it exceeds the maximum allowable torque rating of the wheel, momentum dumping is done.

- **Criteria 2: Maximum cap on wheel angular momentum**

Again, from our wheel dynamics, the applied wheel torque changes the angular speed of the wheel which has a maximum capacity. The resulting angular speed from a given torque must be estimated beforehand to see if it exceeds the capacity. If it is so, dumping of momentum is done.

2.2 Results

2.2.1 Excluding Gravity Gradient

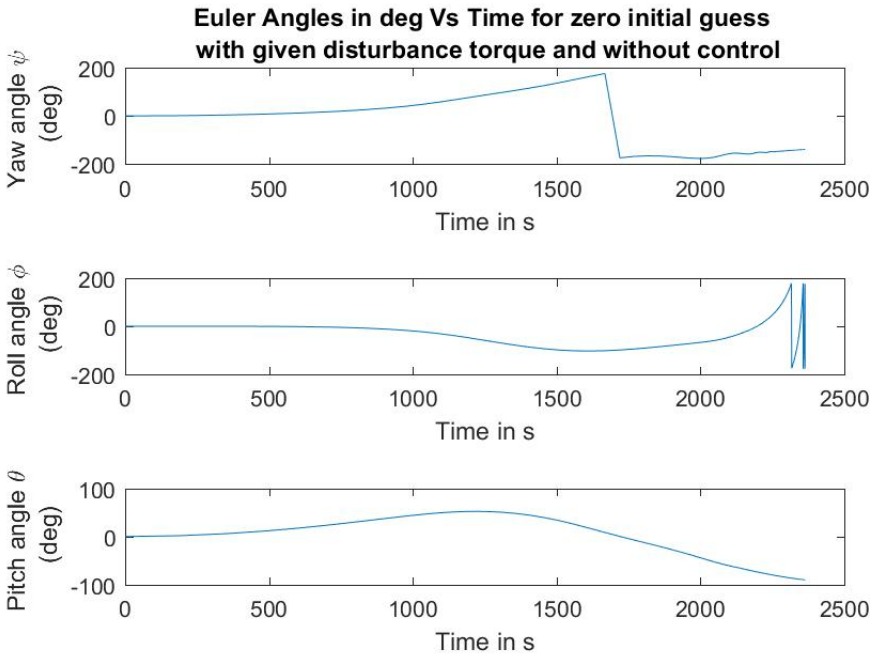

Figure 2.2.1:

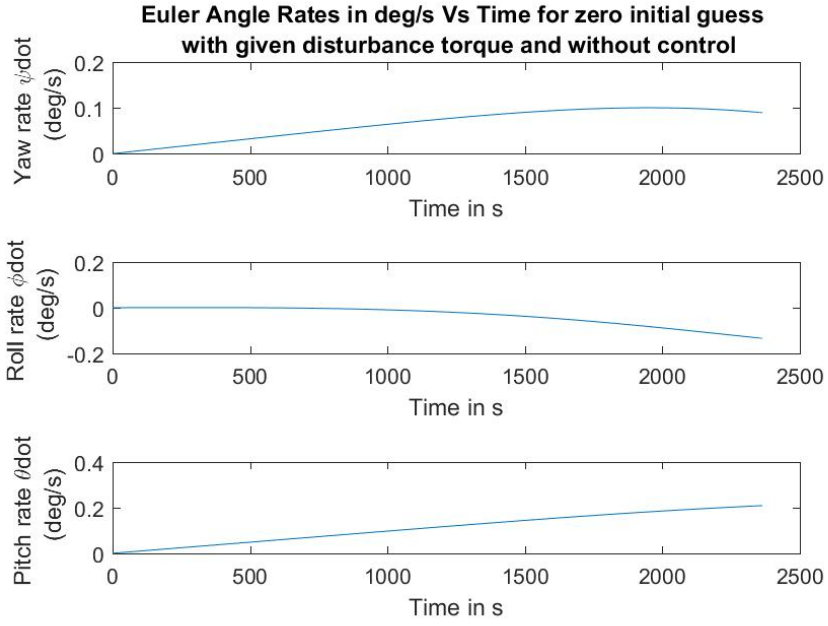

Figure 2.2.2:

Without the application of control torques, in the presence of disturbances, the Euler angles and Euler rates keep diverging. Hence active control system is required for Inertial pointing spacecrafts.

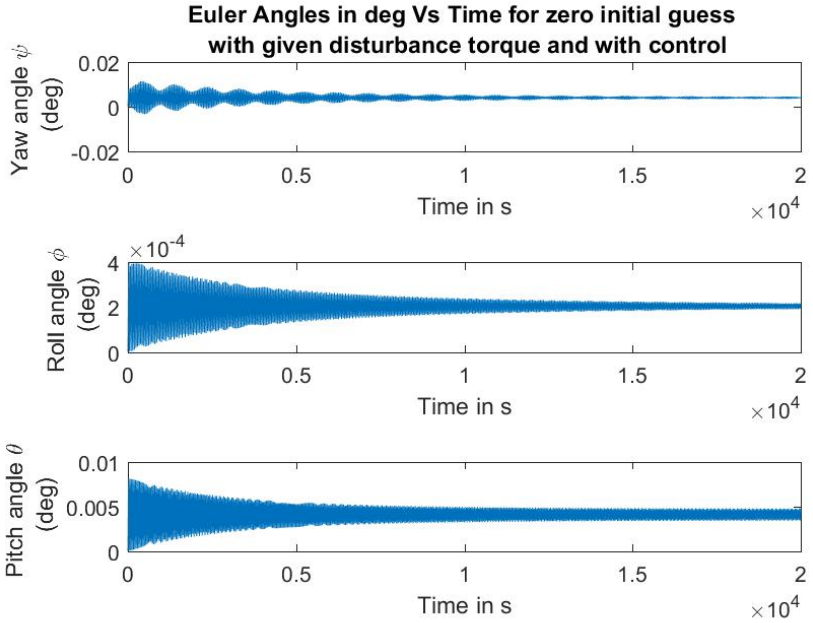

Figure 2.2.3:

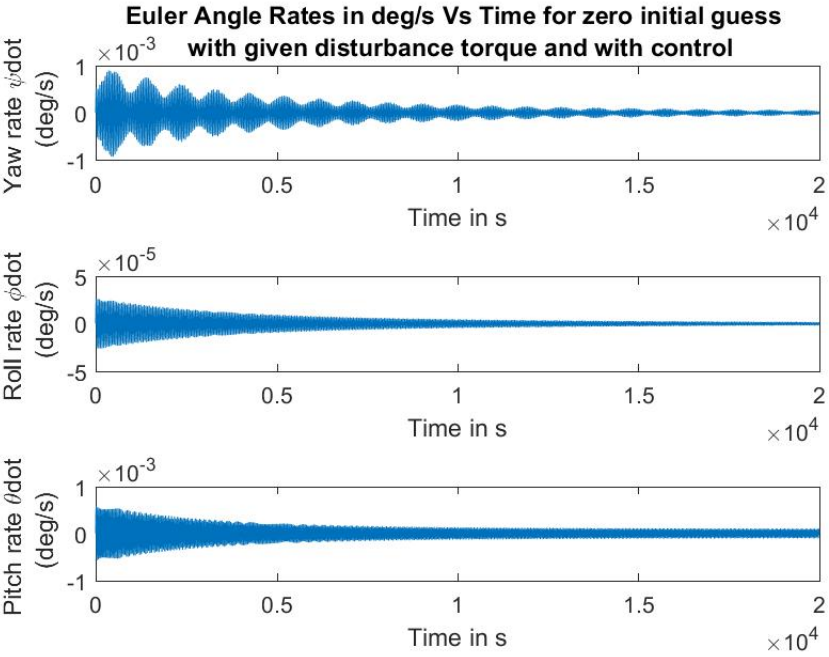

Figure 2.2.4:

As the control torques are applied in x,y and z directions, it is evident from the plots that the Euler angle and Euler rates settle down, such that the steady state error and error rates are well within 0.005 deg and 0.005 deg/sec,

as per the requirement of the problem. The values of Kp and Kd are chosen as 28 and 0.95 after several analysis, for all the axes, to achieve the required steady state error.

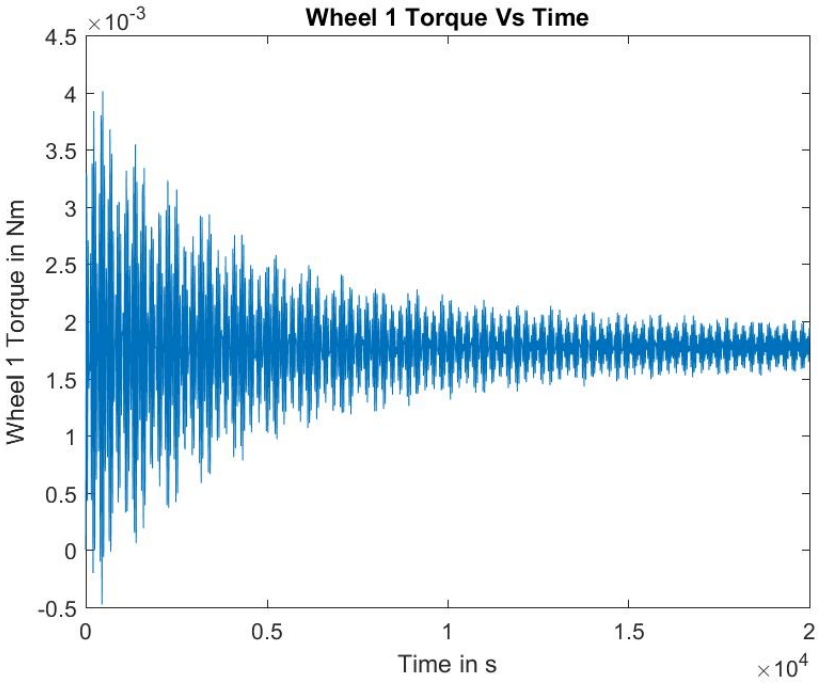

Figure 2.2.5:

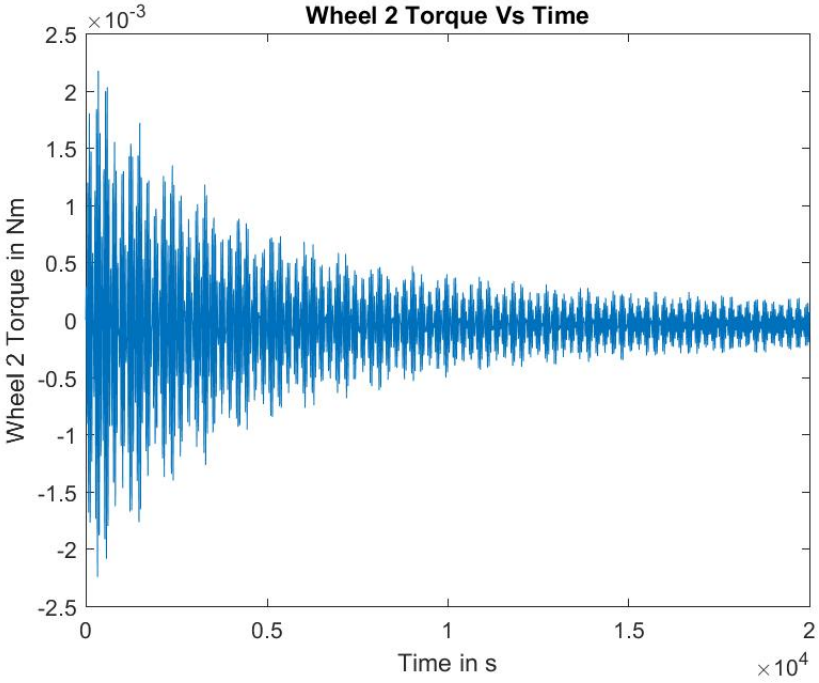

Figure 2.2.6:

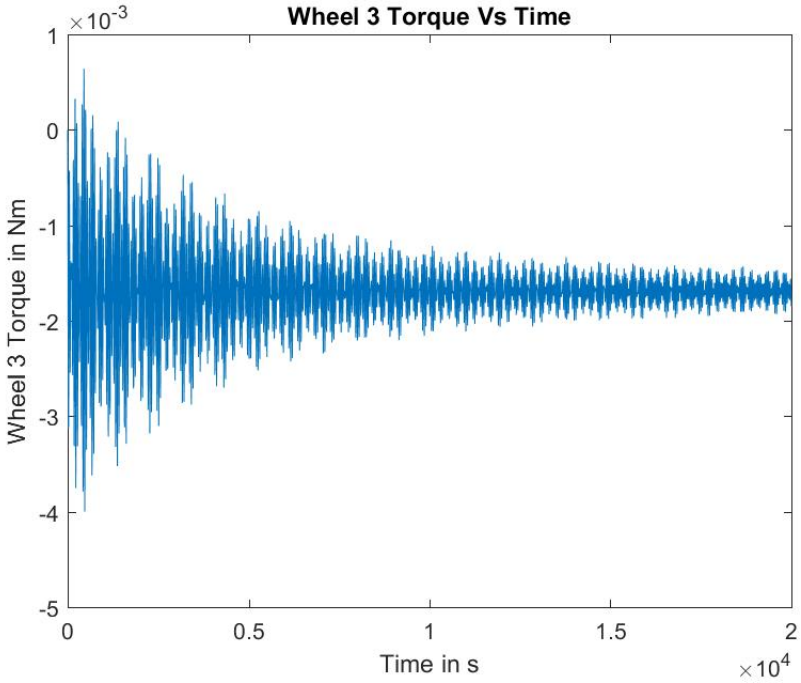

Figure 2.2.7:

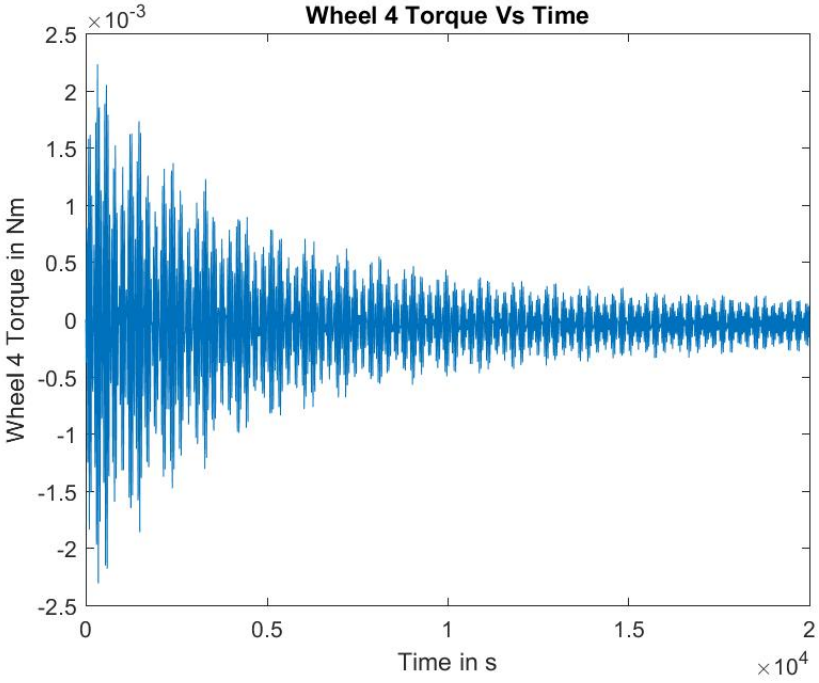

Figure 2.2.8:

All the above plots indicate the variation of Control Toruqe applied with time. As the control torque operate on the error dynamics of the system, eventually as the error reduces over time, the torque reaches zero, but the wheels will have change in angular speed and settle in new angular speed.

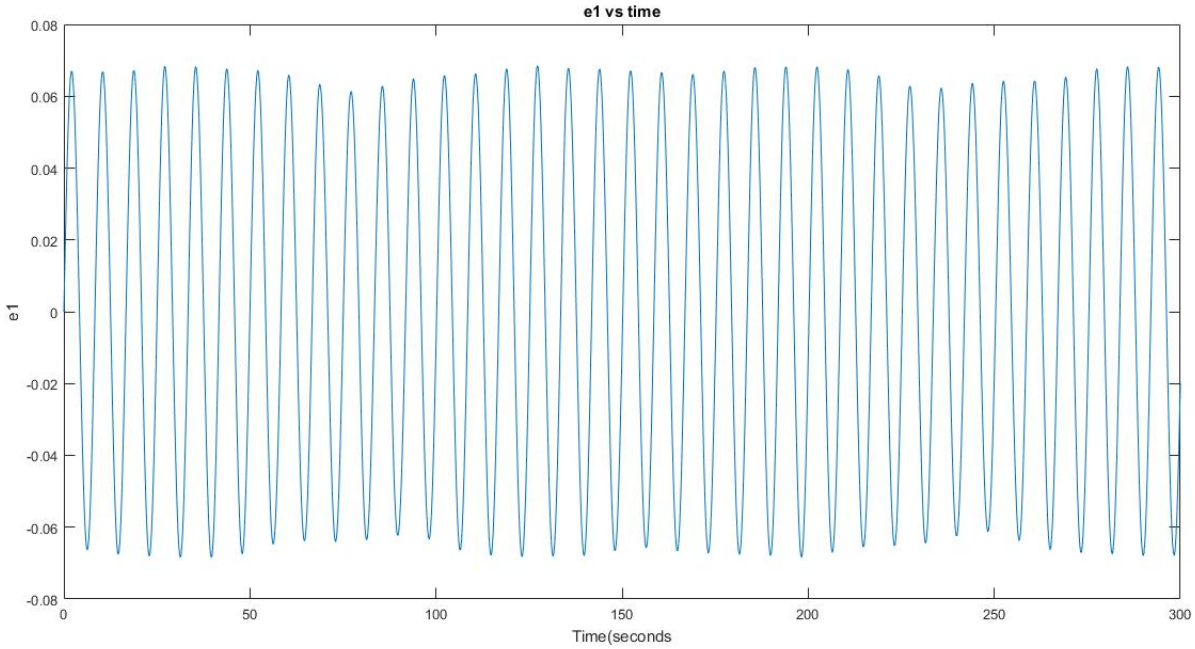

Figure 2.2.9:

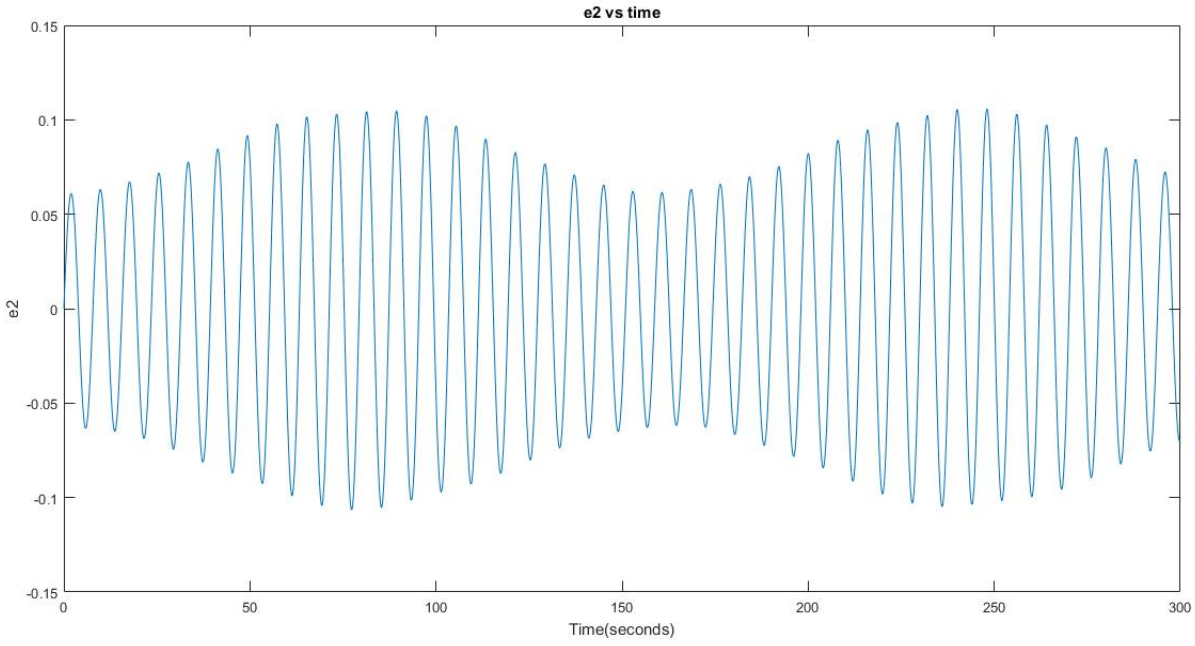

Figure 2.2.10:

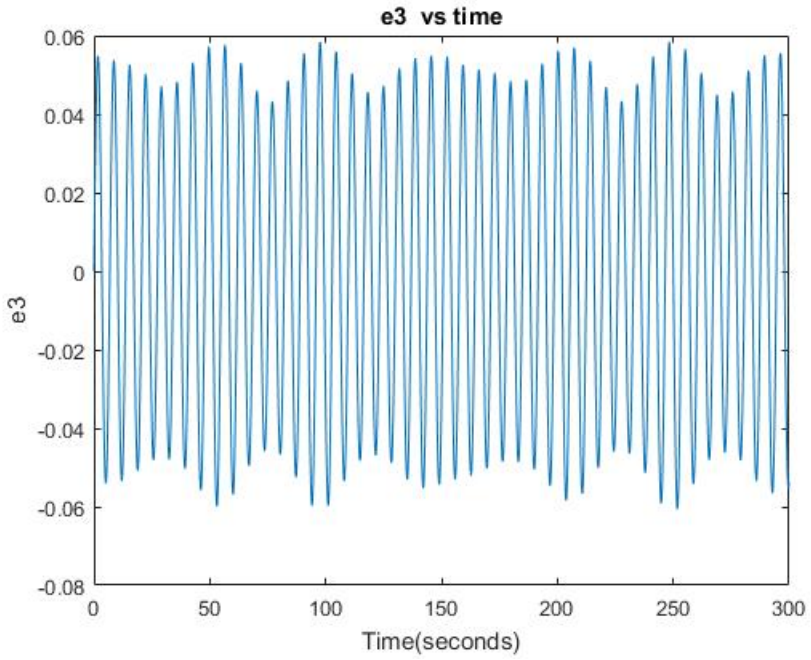

Figure 2.2.11:

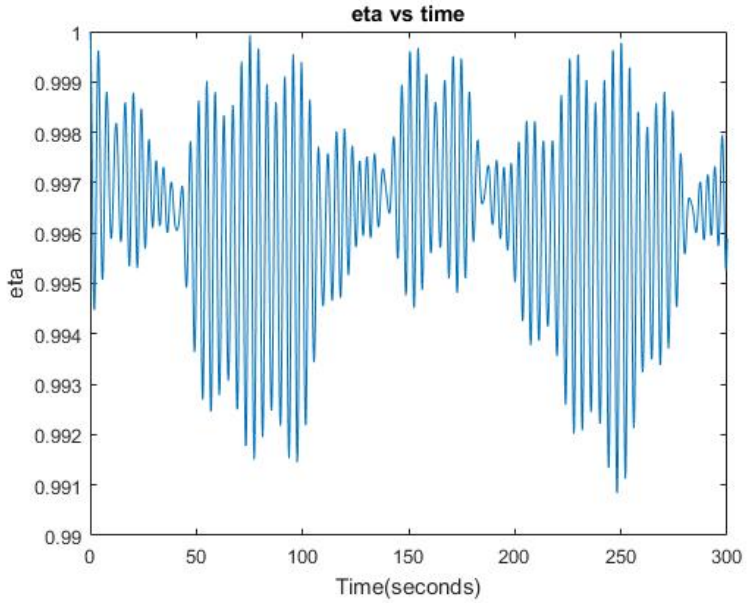

Figure 2.2.12:

The above plots show variation of quaternion in time without including gravity gradient. It can be clearly seen they reach a stable value.

2.2.2 Including Gravity gradient

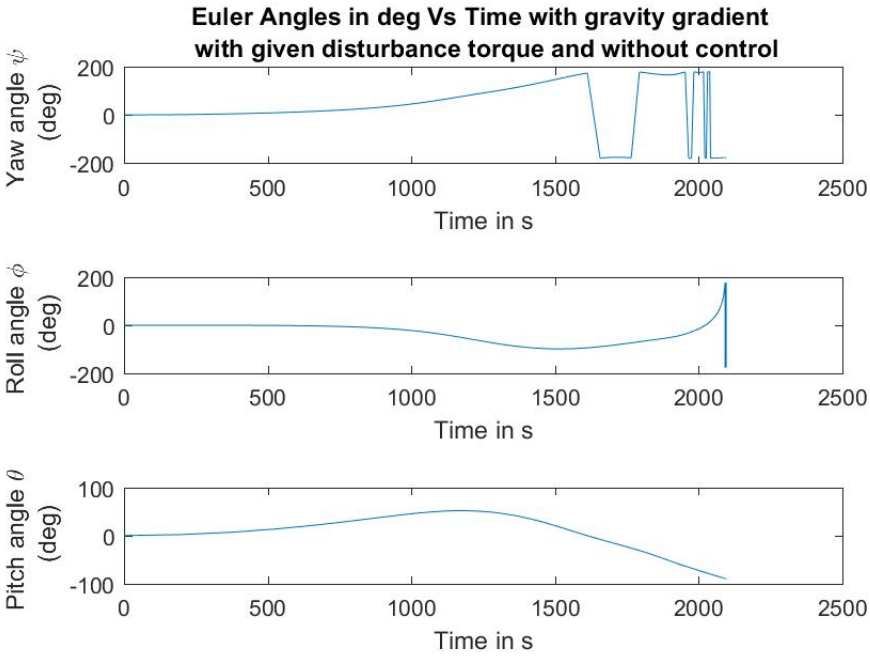

Figure 2.2.13:

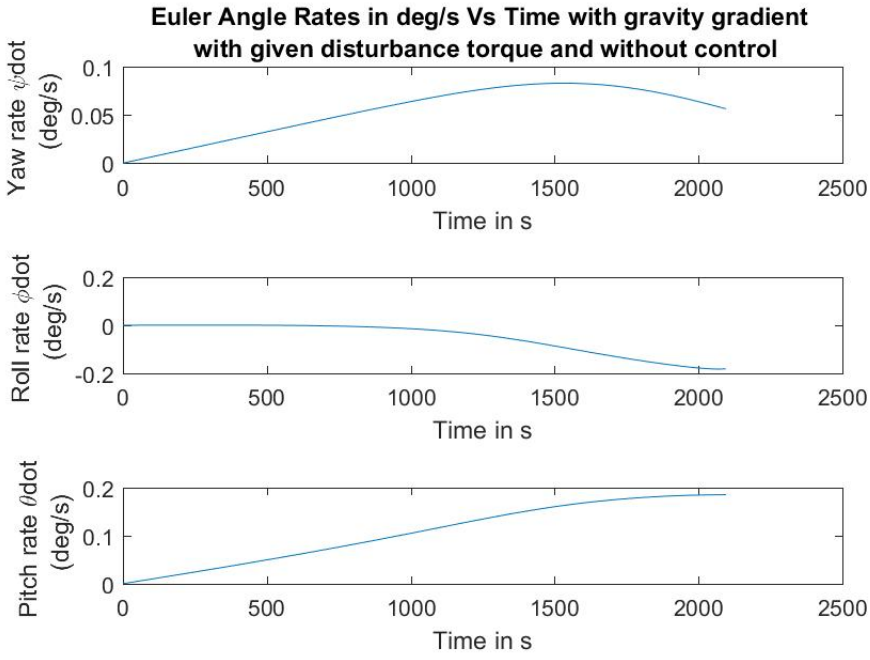

Figure 2.2.14:

As the gravity gradient torques are applied along with the given disturbances, euler angles and rates both diverge with time as indicated by above plots.

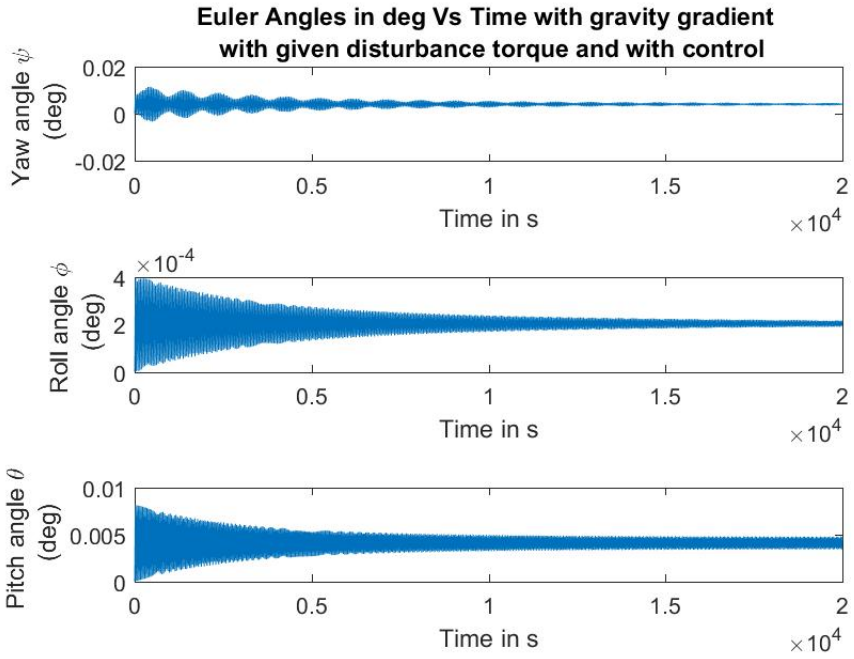

Figure 2.2.15:

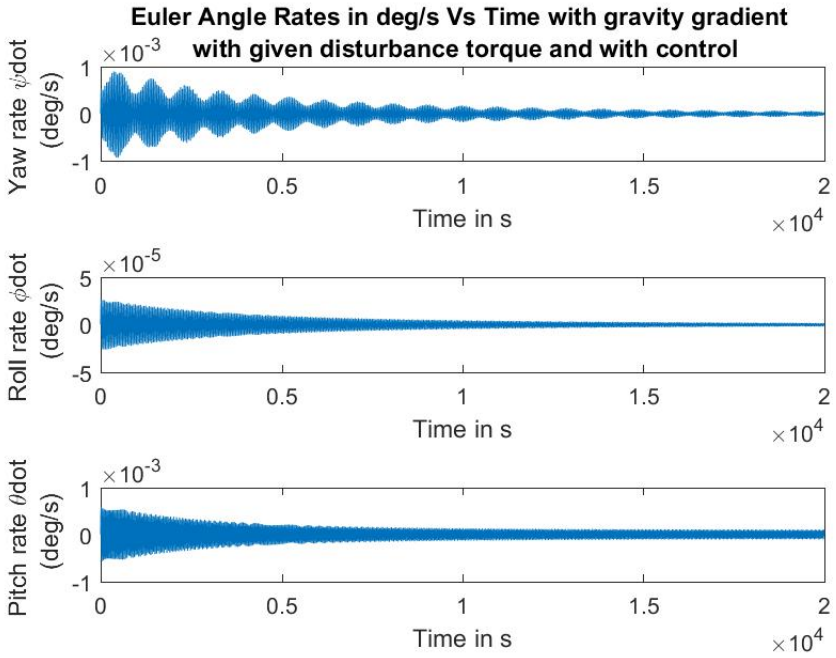

Figure 2.2.16:

Even in the presence of gravity gradient torque and external disturbances, the steady state errors of Euler angles

and Rates are well within 0.005 deg and 0.005 deg/s.

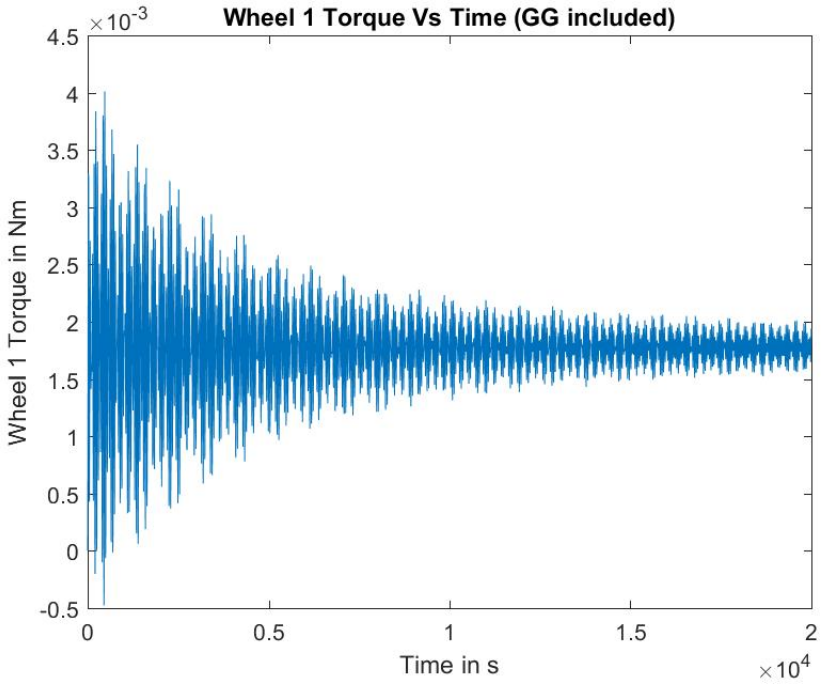

Figure 2.2.17:

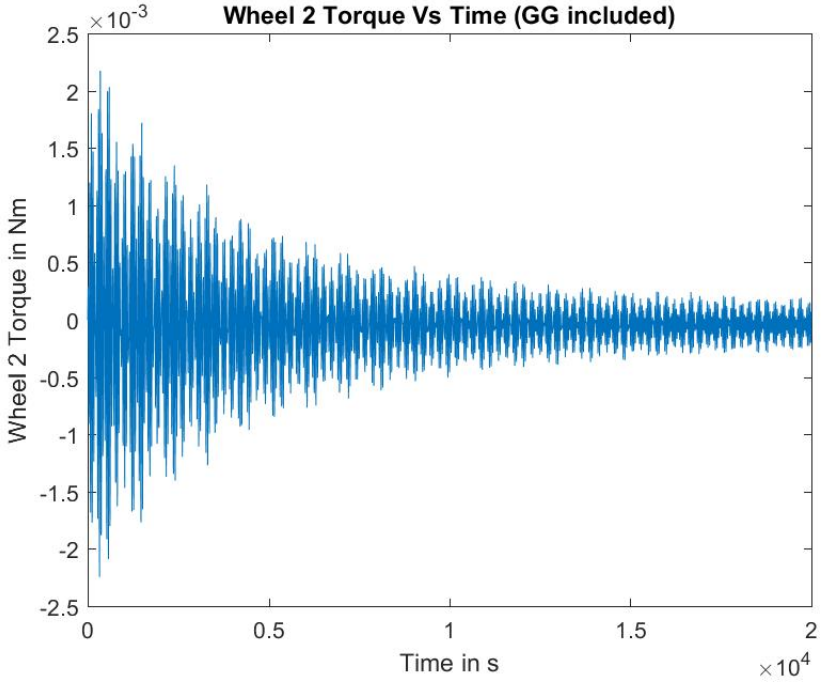

Figure 2.2.18:

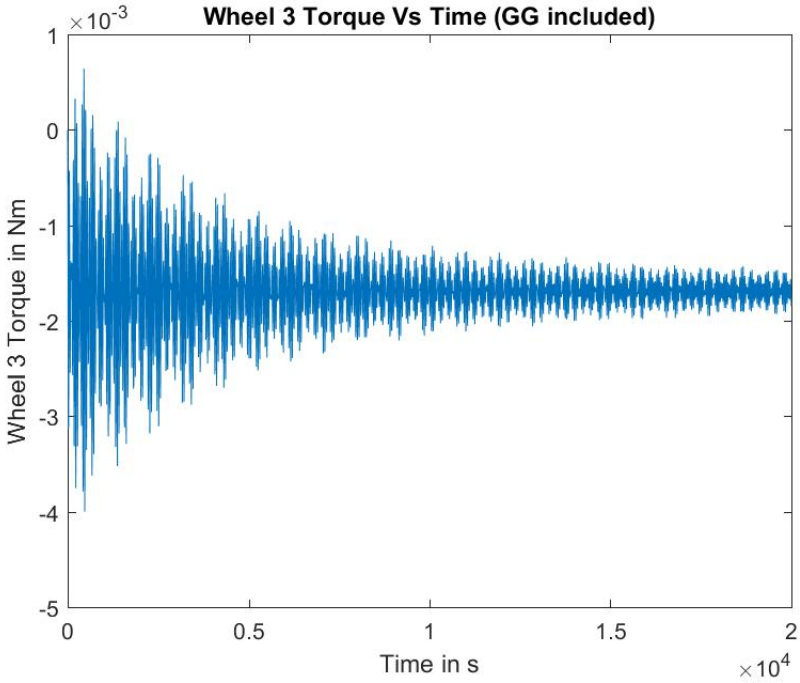

Figure 2.2.19:

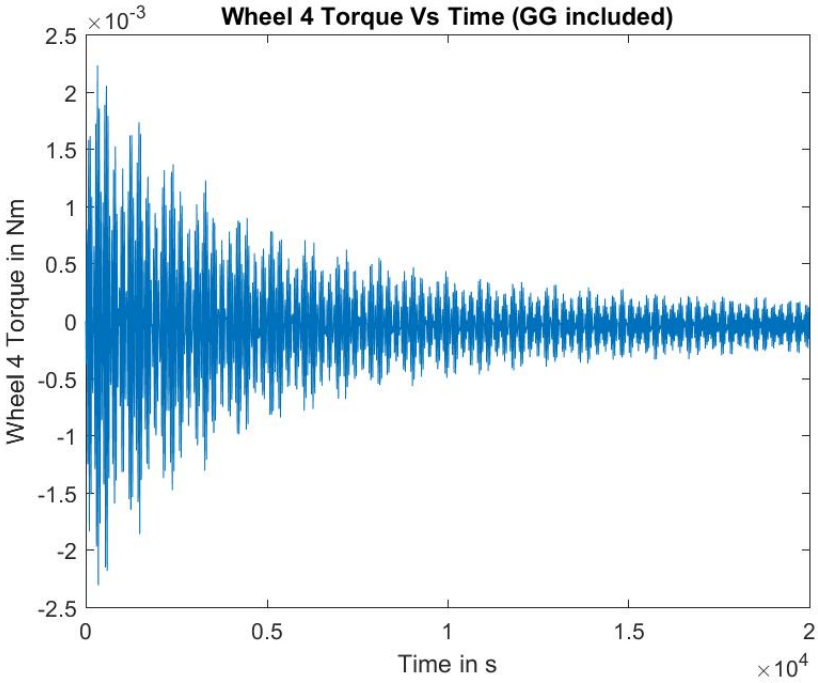

Figure 2.2.20:

From the above plots it is very clear that the torque level for none of the wheels reached 0.5 Nm, hence there is no requirement of momentum dumping even in the presence of gravity gradient torques and external disturbances.

2.3 Conclusion

A modified PD controller was used to control Astrosat to make it stable given a small disturbance (zero momentum biased active control system). After several analysis, the gains K_p and K_d were set so that the steady state errors in the euler angles do not exceed 0.005 degrees. Quaternions and momentum dumping strategies are also addresed in this report apart from Euler angles approach.

Bibliography

[1] Spacecraft Dynamics and Control - Marcel J.Sidis

[2] Spacecraft Dynamics and Control - Anton de Ruiter, Christopher Dameren, James Forbes

Appendix

```
clc;
close all;
w0 = zeros(6,1);

M=1542; %%Mass of the spacecraft
J=[1763 -52 -16;-52 1591 25;-16 25 1185];%%Moment of inertia matrix of the
    entire SpaceCraft
Kpx=28; Kpy=28; Kpz=28;       %% Gain values Proportional
Kdx=0.95; Kdy=0.95; Kdz=0.95;    %% Gain values Derivative
Kp = [Kpx;Kpy;Kpz];
Kd = [Kdx;Kdy;Kdz];

%%case 1 zero initial guess with given disturbance torque without control%%
w0 = [0; 0; 0; 0; 0; 0;];
Tdx=2e-3; Tdy=1e-4; Tdz=2e-3;
Td = [Tdx;Tdy;Tdz];
c_on = 0;
tspan = [0 40000];
g_on = 0;
[tout,wout] = rkf45(@(t,w) AstroSatDynamics(t,w,Td,c_on,M,J,Kp,Kd,g_on),tspan,
    w0,0.0000001);
figure
for k=1:3
    subplot(3,1,k)
    wout(:,k) = wrapToPi(wout(:,k));
    plot(tout,wout(:,k)/pi()*180)
    if k == 1
        str = sprintf('Yaw rate %s \n (deg/s)','\psidot');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angle Rates in deg/s Vs Time for zero initial
            guess \n with given disturbance torque and without control');
        title(str);
    elseif k ==2
        str = sprintf('Roll rate %s \n (deg/s)','\phidot');
        ylabel(str)
```

```
        xlabel('Time in s')
    else
        str = sprintf('Pitch rate %s \n (deg/s)','\thetadot');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_rate_zero_ig_wo_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')

figure
for k=1:3
subplot(3,1,k)
wout(:,k+3) = wrapToPi(wout(:,k+3));
plot(tout,wout(:,k+3)/pi()*180)
    if k == 1
        str = sprintf('Yaw angle %s \n (deg)','\psi');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angles in deg Vs Time for zero initial guess \n
           with given disturbance torque and without control');
        title(str);
    elseif k ==2
        str = sprintf('Roll angle %s \n (deg)','\phi');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch angle %s \n (deg)','\theta');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_angle_zero_ig_wo_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%case 2 zero initial guess with given disturbance torque with control%%

w0 = [0; 0; 0; 0; 0; 0;]
Tdx=2e-3; Tdy=1e-4; Tdz=2e-3;
Kpx=28; Kpy=28; Kpz=28;       %% Gain values Proportional
Kdx=0.95; Kdy=0.95; Kdz=0.95;    %% Gain values Derivative
Kp = [Kpx;Kpy;Kpz];
Kd = [Kdx;Kdy;Kdz];
Td = [Tdx;Tdy;Tdz];
c_on = 1;
```

```
tspan = [0 20000]
[tout,wout] = rkf45(@(t,w) AstroSatDynamics(t,w,Td,c_on,M,J,Kp,Kd,g_on),tspan,
   w0,0.0000001);
wout_no_gg = wout;
figure
for k=1:3
    subplot(3,1,k)
    wout(:,k) = wrapToPi(wout(:,k));
    plot(tout,wout(:,k)/pi()*180)
    if k == 1
        str = sprintf('Yaw rate %s \n (deg/s)','\psidot');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angle Rates in deg/s Vs Time for zero initial
           guess \n with given disturbance torque and with control');
        title(str);
    elseif k ==2
        str = sprintf('Roll rate %s \n (deg/s)','\phidot');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch rate %s \n (deg/s)','\thetadot');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_rate_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')

figure
for k=1:3
subplot(3,1,k)
wout(:,k+3) = wrapToPi(wout(:,k+3));
plot(tout,wout(:,k+3)/pi()*180)
    if k == 1
        str = sprintf('Yaw angle %s \n (deg)','\psi');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angles in deg Vs Time for zero initial guess \n
           with given disturbance torque and with control');
        title(str);
    elseif k ==2
        str = sprintf('Roll angle %s \n (deg)','\phi');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch angle %s \n (deg)','\theta');
        ylabel(str)
        xlabel('Time in s')
    end
```

```
end

str = sprintf('Euler_angle_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
Rphi = 0;
Rpsy = 0;
Rtheta = 0;
Tcx=Kpx*(-Rphi+wout(:,4))+Kdx*wout(:,1);
    Tcy=Kpy*(-Rpsy+wout(:,5))+Kdy*wout(:,2);
    Tcz=Kpz*(-Rtheta+wout(:,6))+Kdz*wout(:,3);
T1=(sqrt(3)*(Tcx)+sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T2=(-sqrt(3)*(Tcx)-sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T3=(-sqrt(3)*(Tcx)+sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;
T4=(sqrt(3)*(Tcx)-sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;
figure
plot(tout,T1(:,1));
ylabel('Wheel 1 Torque in Nm')
xlabel('Time in s')
title('Wheel 1 Torque Vs Time')
str = sprintf('Wheel_1_torque_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T2(:,1));
ylabel('Wheel 2 Torque in Nm')
xlabel('Time in s')
title('Wheel 2 Torque Vs Time')
str = sprintf('Wheel_2_torque_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T3(:,1));
ylabel('Wheel 3 Torque in Nm')
xlabel('Time in s')
title('Wheel 3 Torque Vs Time')
str = sprintf('Wheel_3_torque_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T4(:,1));
ylabel('Wheel 4 Torque in Nm')
xlabel('Time in s')
title('Wheel 4 Torque Vs Time')
str = sprintf('Wheel_4_torque_zero_ig_w_c')
str1  = sprintf('%s%s',str,'.fig')
```

```
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')

g_on = 1;

% %%case 3 initial guess zero and gravity torque included in disturbance
    without control%%
w0 = [0; 0; 0; 0; 0; 0;]
Tdx=2e-3; Tdy=1e-4; Tdz=2e-3;
Td = [Tdx;Tdy;Tdz];
c_on = 0;
tspan = [0 20000]

[tout,wout] = rkf45(@(t,w) AstroSatDynamics(t,w,Td,c_on,M,J,Kp,Kd,g_on),tspan,
    w0,0.0000001)

figure
for k=1:3
    subplot(3,1,k)
    wout(:,k) = wrapToPi(wout(:,k));
    plot(tout,wout(:,k)/pi()*180)
    if k == 1
        str = sprintf('Yaw rate %s \n (deg/s)','\psidot');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angle Rates in deg/s Vs Time with gravity gradient
            \n with given disturbance torque and without control');
        title(str);
    elseif k ==2
        str = sprintf('Roll rate %s \n (deg/s)','\phidot');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch rate %s \n (deg/s)','\thetadot');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_rate_zero_ig_gg_wo_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')

figure
for k=1:3
subplot(3,1,k)
wout(:,k+3) = wrapToPi(wout(:,k+3));
plot(tout,wout(:,k+3)/pi()*180)
    if k == 1
        str = sprintf('Yaw angle %s \n (deg)','\psi');
        ylabel(str)
```

```
        xlabel('Time in s')
        str = sprintf('Euler Angles in deg Vs Time with gravity gradient \n
            with given disturbance torque and without control');
        title(str);
    elseif k ==2
        str = sprintf('Roll angle %s \n (deg)','\phi');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch angle %s \n (deg)','\theta');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_angle_zero_ig_gg_wo_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%case 4 initial guess zero and gravity torque included in disturbance with
    control%%
w0 = [0; 0; 0; 0; 0; 0;]
Tdx=2e-3; Tdy=1e-4; Tdz=2e-3;
Td = [Tdx;Tdy;Tdz];
Kpx=28; Kpy=28; Kpz=28;       %% Gain values Proportional
Kdx=0.95; Kdy=0.95; Kdz=0.95;    %% Gain values Derivative
Kp = [Kpx;Kpy;Kpz];
Kd = [Kdx;Kdy;Kdz];
Td = [Tdx;Tdy;Tdz];
c_on = 1;
tspan = [0 20000]

[tout,wout] = rkf45(@(t,w) AstroSatDynamics(t,w,Td,c_on,M,J,Kp,Kd,g_on),tspan,
    w0,0.0000001)
wout_w_gg = wout;
figure
for k=1:3
    subplot(3,1,k)
    wout(:,k) = wrapToPi(wout(:,k));
    plot(tout,wout(:,k)/pi()*180)
    if k == 1
        str = sprintf('Yaw rate %s \n (deg/s)','\psidot');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angle Rates in deg/s Vs Time with gravity gradient
            \n with given disturbance torque and with control');
        title(str);
    elseif k ==2
        str = sprintf('Roll rate %s \n (deg/s)','\phidot');
        ylabel(str)
```

```
        xlabel('Time in s')
    else
        str = sprintf('Pitch rate %s \n (deg/s)','\thetadot');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_rate_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')

figure
for k=1:3
subplot(3,1,k)
wout(:,k+3) = wrapToPi(wout(:,k+3));
plot(tout,wout(:,k+3)/pi()*180)
    if k == 1
        str = sprintf('Yaw angle %s \n (deg)','\psi');
        ylabel(str)
        xlabel('Time in s')
        str = sprintf('Euler Angles in deg Vs Time with gravity gradient \n
            with given disturbance torque and with control');
        title(str);
    elseif k ==2
        str = sprintf('Roll angle %s \n (deg)','\phi');
        ylabel(str)
        xlabel('Time in s')
    else
        str = sprintf('Pitch angle %s \n (deg)','\theta');
        ylabel(str)
        xlabel('Time in s')
    end
end

str = sprintf('Euler_angle_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
Rphi = 0;
Rpsy = 0;
Rtheta = 0;
Tcx=Kpx*(-Rphi+wout(:,4))+Kdx*wout(:,1);
    Tcy=Kpy*(-Rpsy+wout(:,5))+Kdy*wout(:,2);
    Tcz=Kpz*(-Rtheta+wout(:,6))+Kdz*wout(:,3);
T1=(sqrt(3)*(Tcx)+sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T2=(-sqrt(3)*(Tcx)-sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T3=(-sqrt(3)*(Tcx)+sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;
T4=(sqrt(3)*(Tcx)-sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;
figure
plot(tout,T1(:,1));
```

```
ylabel('Wheel 1 Torque in Nm')
xlabel('Time in s')
title('Wheel 1 Torque Vs Time (GG included)')
str = sprintf('Wheel_1_torque_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T2(:,1));
ylabel('Wheel 2 Torque in Nm')
xlabel('Time in s')
title('Wheel 2 Torque Vs Time (GG included)')
str = sprintf('Wheel_2_torque_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T3(:,1));
ylabel('Wheel 3 Torque in Nm')
xlabel('Time in s')
title('Wheel 3 Torque Vs Time (GG included)')
str = sprintf('Wheel_3_torque_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')
figure
plot(tout,T4(:,1));
ylabel('Wheel 4 Torque in Nm')
xlabel('Time in s')
title('Wheel 4 Torque Vs Time (GG included)')
str = sprintf('Wheel_4_torque_zero_ig_gg_w_c')
str1  = sprintf('%s%s',str,'.fig')
savefig(str1);
saveas(gcf, str, 'jpg')
saveas(gcf, str, 'eps')


function wdot = AstroSatDynamics(t,w,Td,c_on,M,J,Kp,Kd,g_on)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
wdot=zeros(6,1);

Jxx=J(1,1); Jxy=J(1,2); Jxz=J(1,3); Jyx=J(2,1); Jyy=J(2,2); Jyz=J(2,3); Jzx=J
   (3,1); Jzy=J(3,2); Jzz=J(3,3);
                                    %%Mass of the SpaceCraft
Rphi=0;                                    %%Reference input signal for Phi angle
Rtheta=0;                                  %%Reference input signal for theta
   angle
Rpsy=0;                                    %%Reference input signal for psy angle

Tdx  = Td(1);
Tdy = Td(2);
```

```
Tdz = Td(3);
if c_on == 1
    Kpx = Kp(1);
    Kpy = Kp(2);
    Kpz = Kp(3);
    Kdx = Kd(1);
    Kdy = Kd(2);
    Kdz = Kd(3);
    Tcx=Kpx*(Rphi-w(4))-Kdx*w(1);
    Tcy=Kpy*(Rpsy-w(5))-Kdy*w(2);
    Tcz=Kpz*(Rtheta-w(6))-Kdz*w(3);
else
    Tcx= 0;
    Tcy = 0;
    Tcz = 0;
end

%%%Wheel torgues (in wheel frame)

T1=(sqrt(3)*(Tcx)+sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T2=(-sqrt(3)*(Tcx)-sqrt(3)*(Tcy)+sqrt(3)*(Tcz))/4;
T3=(-sqrt(3)*(Tcx)+sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;
T4=(sqrt(3)*(Tcx)-sqrt(3)*(Tcy)-sqrt(3)*(Tcz))/4;

%% Wheel configuration details(You can use any general wheel config,
%%but here  it is coded assuming a tetrahedron wheel configuration)

%%%for the purpose of momentum dumping
k1=[1/sqrt(3);1/sqrt(3);1/sqrt(3)];
k2=[-1/sqrt(3);-1/sqrt(3);1/sqrt(3)];
k3=[-1/sqrt(3);1/sqrt(3);-1/sqrt(3)];
k4=[1/sqrt(3);-1/sqrt(3);-1/sqrt(3)];
a1=cross([1;0;0],k1)/norm(cross([1;0;0],k1));
a2=cross([1;0;0],k2)/norm(cross([1;0;0],k2));
a3=cross([1;0;0],k3)/norm(cross([1;0;0],k3));
a4=cross([1;0;0],k4)/norm(cross([1;0;0],k4));
phi1=acos(dot(k1,[1;0;0])/norm(k1)*norm([1;0;0]));
phi2=acos(dot(k2,[1;0;0])/norm(k2)*norm([1;0;0]));
phi3=acos(dot(k3,[1;0;0])/norm(k3)*norm([1;0;0]));
phi4=acos(dot(k4,[1;0;0])/norm(k4)*norm([1;0;0]));
I=[1 0 0;0 1 0;0 0 1];
a1star=[0 -a1(3) a1(2);a1(3) 0 -a1(1);-a1(2) a1(1) 0];
a2star=[0 -a2(3) a2(2);a2(3) 0 -a2(1);-a2(2) a2(1) 0];
a3star=[0 -a3(3) a3(2);a3(3) 0 -a3(1);-a3(2) a3(1) 0];
a4star=[0 -a4(3) a4(2);a4(3) 0 -a4(1);-a4(2) a4(1) 0];
C1=I*cos(phi1)+(1-cos(phi1))*(a1*a1')-sin(phi1)*a1star;
C2=I*cos(phi2)+(1-cos(phi2))*(a2*a2')-sin(phi2)*a2star;
C3=I*cos(phi3)+(1-cos(phi3))*(a3*a3')-sin(phi3)*a3star;
C4=I*cos(phi4)+(1-cos(phi4))*(a4*a4')-sin(phi4)*a4star;
Iw=[0.1 0 0;0 0.1 0; 0 0 0.1];          %% Moment of inertia matrix for the
   wheels
Ix=Iw(1,1); Iy=Iw(2,2); Iz=Iw(3,3);
a1=C1(1,1); b1=C1(1,2); c1=C1(1,3); p1=C1(2,1); q1=C1(2,2); r1=C1(2,3); x1=C1
   (3,1); y1=C1(3,2); z1=C1(3,3);
```

```
a2=C2(1,1); b2=C2(1,2); c2=C2(1,3); p2=C2(2,1); q2=C2(2,2); r2=C2(2,3); x2=C2
    (3,1); y2=C2(3,2); z2=C2(3,3);
a3=C3(1,1); b3=C3(1,2); c3=C3(1,3); p3=C3(2,1); q3=C3(2,2); r3=C3(2,3); x3=C3
    (3,1); y3=C3(3,2); z3=C3(3,3);
a4=C4(1,1); b4=C4(1,2); c4=C4(1,3); p4=C4(2,1); q4=C4(2,2); r4=C4(2,3); x4=C4
    (3,1); y4=C4(3,2); z4=C4(3,3);

%%%%%Gravity Gradient%%%%
if g_on == 1
     Tgy = 3*(1.0741e-3)^2*(Jxx-Jzz)*w(5);
     Tgz = 3*(1.0741e-3)^2*(Jxx-Jyy)*w(6);
else
     Tgy = 0;
     Tgz = 0;
end
X=Tcx+Tdx+(w(3)*((Jyx*w(1))+(Jyy*w(2))+(Jyz*w(3))))-(w(2)*((Jzx*w(1))+(Jzy*w(2)
    )+(Jzz*w(3))));
Y=Tcy+Tdy+Tgy-(w(3)*((Jxx*w(1))+(Jxy*w(2))+(Jxz*w(3))))+(w(1)*((Jzx*w(1))+(Jzy*
    w(2))+(Jzz*w(3))));
Z=Tcz+Tdz+Tgz+(w(2)*((Jxx*w(1))+(Jxy*w(2))+(Jxz*w(3))))-(w(1)*((Jyx*w(1))+(Jyy*
    w(2))+(Jyz*w(3))));

%%Dynamic equations are written separating the SpaceCraft and the Wheel
P=(X-(Jxy*wdot(2))-(Jxz*wdot(3)))/Jxx;
Q=(Y-(Jyx*wdot(1))-(Jyz*wdot(3)))/Jyy;
R=(Z-(Jzx*wdot(1))-(Jzy*wdot(2)))/Jzz;
wdot(1) = P
wdot(2) = Q
wdot(3) = R
wdot(4)=w(1)-((cos(w(4)*tan(w(6))))*w(2))+((sin(w(4)*tan(w(6))))*w(3));
wdot(5)=((cos(w(4))*sec(w(6)))*w(2))-((sin(w(4))*sec(w(6)))*w(3));
wdot(6)= ((sin(w(4)))*w(2))+((cos(w(4)))*w(3));

end


function wdot = AstroSatQuaternions(t,w)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
wdot=zeros(11,1);
J=[1763 -52 -16;-52 1591 25;-16 25 1185];%%Moment of inertia matrix of the
    SpaceCraft
Jxx=J(1,1); Jxy=J(1,2); Jxz=J(1,3); Jyx=J(2,1); Jyy=J(2,2); Jyz=J(2,3); Jzx=J
    (3,1); Jzy=J(3,2); Jzz=J(3,3);
M=1542;                                    %%Mass of the SpaceCraft
Re1=0;                                    %%Reference input signal for Phi angle
Re2=0;                                  %%Reference input signal for theta angle
Re3=0;                                    %%Reference input signal for psy angle
Kpx=100; Kpy=100; Kpz=100;                        %% Gain values Proportional
Kdx=10; Kdy=10; Kdz=10;                       %% Gain values Derivative
Tcx=2*Kpx*(Re1-w(4))*(w(11))+Kdx*wdot(1);
Tcy=2*Kpy*(Re2-w(5))*(w(11))+Kdy*wdot(2);
Tcz=2*Kpz*(Re3-w(6))*(w(11))+Kdz*wdot(3);
Tdx=0; Tdy=0; Tdz=0;                         %% Disturbance Torques
%% Wheel configuration details(You can use any general wheel config, but here
```

```
   it is coded assuming a tetrahedron wheel configuration)
k1=[1/sqrt(3);1/sqrt(3);1/sqrt(3)];
k2=[-1/sqrt(3);-1/sqrt(3);1/sqrt(3)];
k3=[-1/sqrt(3);1/sqrt(3);-1/sqrt(3)];
k4=[1/sqrt(3);-1/sqrt(3);-1/sqrt(3)];
A1=cross([1;0;0],k1)/norm(cross([1;0;0],k1));
A2=cross([1;0;0],k2)/norm(cross([1;0;0],k2));
A3=cross([1;0;0],k3)/norm(cross([1;0;0],k3));
A4=cross([1;0;0],k4)/norm(cross([1;0;0],k4));
phi1=acos(dot(k1,[1;0;0])/norm(k1)*norm([1;0;0]));
phi2=acos(dot(k2,[1;0;0])/norm(k2)*norm([1;0;0]));
phi3=acos(dot(k3,[1;0;0])/norm(k3)*norm([1;0;0]));
phi4=acos(dot(k4,[1;0;0])/norm(k4)*norm([1;0;0]));
I=[1 0 0;0 1 0;0 0 1];
A1star=[0 -A1(3) A1(2);A1(3) 0 -A1(1);-A1(2) A1(1) 0];
A2star=[0 -A2(3) A2(2);A2(3) 0 -A2(1);-A2(2) A2(1) 0];
A3star=[0 -A3(3) A3(2);A3(3) 0 -A3(1);-A3(2) A3(1) 0];
A4star=[0 -A4(3) A4(2);A4(3) 0 -A4(1);-A4(2) A4(1) 0];
C1=I*cos(phi1)+(1-cos(phi1))*(A1*A1')-sin(phi1)*A1star;
C2=I*cos(phi2)+(1-cos(phi2))*(A2*A2')-sin(phi2)*A2star;
C3=I*cos(phi3)+(1-cos(phi3))*(A3*A3')-sin(phi3)*A3star;
C4=I*cos(phi4)+(1-cos(phi4))*(A4*A4')-sin(phi4)*A4star;
Iw=[0.1 0 0;0 0.05 0; 0 0 0.05];          %% Moment of inertia matrix for the
   wheels
Ix=Iw(1,1); Iy=Iw(2,2); Iz=Iw(3,3);
a1=C1(1,1); b1=C1(1,2); c1=C1(1,3); p1=C1(2,1); q1=C1(2,2); r1=C1(2,3); x1=C1
   (3,1); y1=C1(3,2); z1=C1(3,3);
a2=C2(1,1); b2=C2(1,2); c2=C2(1,3); p2=C2(2,1); q2=C2(2,2); r2=C2(2,3); x2=C2
   (3,1); y2=C2(3,2); z2=C2(3,3);
a3=C3(1,1); b3=C3(1,2); c3=C3(1,3); p3=C3(2,1); q3=C3(2,2); r3=C3(2,3); x3=C3
   (3,1); y3=C3(3,2); z3=C3(3,3);
a4=C4(1,1); b4=C4(1,2); c4=C4(1,3); p4=C4(2,1); q4=C4(2,2); r4=C4(2,3); x4=C4
   (3,1); y4=C4(3,2); z4=C4(3,3);
X=Tcx+Tdx+(w(3)*((Jyx*w(1))+(Jyy*w(2))+(Jyz*w(3))))-(w(2)*((Jzx*w(1))+(Jzy*w(2)
   )+(Jzz*w(3))));
Y=Tcy+Tdy-(w(3)*((Jxx*w(1))+(Jxy*w(2))+(Jxz*w(3))))+(w(1)*((Jzx*w(1))+(Jzy*w(2)
   )+(Jzz*w(3))));
Z=Tcz+Tdz+(w(2)*((Jxx*w(1))+(Jxy*w(2))+(Jxz*w(3))))-(w(1)*((Jyx*w(1))+(Jyy*w(2)
   )+(Jyz*w(3))));

%%Dynamic equations are written separating the SpaceCraft and the Wheel
wdot(1)=(((Y*Jzy)-(Z*Jyy))-((Jyz*Jzy)-(Jzz*Jyy))*wdot(3))/((Jyx*Jzy)-(Jzx*Jyy))
   ;
wdot(2)=(X-(Jxz*wdot(3))-(Jxx*wdot(1)))/Jyy;
wdot(3)=(((X*Jyy)-(Y*Jxy))*((Jyx*Jzy)-(Jzx*Jyy))-((Y*Jzy)-(Z*Jyy))*((Jxx*Jyy)-(
   Jyx*Jxy)))/(((Jyy*Jxz)-(Jyz*Jxy))*((Jyx*Jzy)-(Jzx*Jyy))-((Jyz*Jzy)-(Jzz*Jyy)
   )*((Jxx*Jyy)-(Jyx*Jxy)));
wdot(4)=0.5*(w(1)*w(11)+w(3)*w(5)-w(2)*w(6));
wdot(5)=0.5*(w(2)*w(11)-w(3)*w(4)+w(1)*w(6));
wdot(6)=0.5*(w(3)*w(11)+w(2)*w(4)-w(1)*w(5));
T1=(sqrt(3)*(-Tcx)+sqrt(3)*(-Tcy)+sqrt(3)*(-Tcz))/4;
T2=(-sqrt(3)*(-Tcx)-sqrt(3)*(-Tcy)+sqrt(3)*(-Tcz))/4;
T3=(-sqrt(3)*(-Tcx)+sqrt(3)*(-Tcy)-sqrt(3)*(-Tcz))/4;
T4=(sqrt(3)*(-Tcx)-sqrt(3)*(-Tcy)-sqrt(3)*(-Tcz))/4;
```

```
wdot(7)=(T1-(Ix*((a1*wdot(1))+(b1*wdot(2))+(c1*wdot(3))))+(((x1*w(1))+(y1*w(2))
    +(z1*w(3)))*(Iy*((p1*w(1))+(q1*w(2))+(r1*w(3)))))-(((p1*w(1))+(q1*w(2))+(r1*
    w(3)))*(Iz*((x1*w(1))+(y1*w(2))+(z1*w(3))))))/Ix;
wdot(8)=(T2-(Ix*((a2*wdot(1))+(b2*wdot(2))+(c2*wdot(3))))+(((x2*w(1))+(y2*w(2))
    +(z2*w(3)))*(Iy*((p2*w(1))+(q2*w(2))+(r2*w(3)))))-(((p2*w(1))+(q2*w(2))+(r2*
    w(3)))*(Iz*((x2*w(1))+(y2*w(2))+(z2*w(3))))))/Ix;
wdot(9)=(T3-(Ix*((a3*wdot(1))+(b3*wdot(2))+(c3*wdot(3))))+(((x3*w(1))+(y3*w(2))
    +(z3*w(3)))*(Iy*((p3*w(1))+(q3*w(2))+(r3*w(3)))))-(((p3*w(1))+(q3*w(2))+(r3*
    w(3)))*(Iz*((x3*w(1))+(y3*w(2))+(z3*w(3))))))/Ix;
wdot(10)=(T4-(Ix*((a4*wdot(1))+(b4*wdot(2))+(c4*wdot(3))))+(((x4*w(1))+(y4*w(2)
    )+(z4*w(3)))*(Iy*((p4*w(1))+(q4*w(2))+(r4*w(3)))))-(((p4*w(1))+(q4*w(2))+(r4
    *w(3)))*(Iz*((x4*w(1))+(y4*w(2))+(z4*w(3))))))/Ix;

end
```