3-22-2007

Satellite Attitude Control Using Atmopheric Drag

David B. Guettler

Satellite Attitude Control Using Atmospheric Drag

THESIS

David B. Guettler, Captain, USAF

AFIT/GA/ENY/07-M10

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

AFIT/GA/ENY/07-M10

Satellite Attitude Control Using Atmospheric Drag

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Astronautical Engineering

David B. Guettler, BS

Captain, USAF

March 2007

AFIT/GA/ENY/07-M10

Satellite Attitude Control Using Atmospheric Drag

David B. Guettler, BS

Captain, USAF

Approved:

/signed/	22 Mar 2007
Dr. William E. Wiesel (Chairman)	date
/signed/	22 Mar 2007
Nathan A. Titus, Lt Col, USAF (Member)	date
/signed/	22 Mar 2007
Eric D. Swenson, Maj, USAF (Member)	date

AFIT/GA/ENY/07-M10

Abstract

Attitude control is a requirement for most satellites. Many schemes have been devised over the years including control moment gyros, reaction wheels, spin stabilization and gravity gradient stabilization. For low Earth orbits, the Earth's atmosphere can have an affect on a satellite's orbit and attitude. Using the atmosphere to control spacecraft attitude has been researched in the past however very little research has been done using an active feedback control system to maintain spacecraft attitude.

This research effort examines the feasibility of using the atmosphere to actively control a spacecraft's attitude using drag panels. Several variables affect the drag force, of which, projected area is the only variable that can be changed easily. Adding controllable drag panels to a satellite gives the ability to change the projected area as well as the location of the projected area. The result of manipulating the projected areas is a force that is not aligned with the center of gravity, resulting in an external torque on the spacecraft. Although these torques are very small, on the scale of micro-Newton meters and smaller, over time these torques can be used to change the spacecraft's attitude.

A linear computer model was created using a proportional controller. This model was used to evaluate the effectiveness of using drag panels for attitude control. Results from the simulation show that the spacecraft can recover from disturbance torques that may cause a change in attitude very effectively especially at low altitudes (200-300km). At 200km, the satellite is able to recover from a disturbance in less than one hour. As the altitude increases, these settling times increase exponentially. At 600km it takes approximately 2 weeks to stabilize. Other factors affect the settling

time such as mass, and the geometric dimensions of the satellite's control panels and control arms.

Acknowledgements

First and foremost, I owe a large debt of gratitude to Dr. Wiesel for his help. I would also like to thank fellow students for their help and finally, I would like to thank my family, especially my wife for her support.

David B. Guettler

Table of Contents

Page

Page

List of Figures

List of Tables

List of Symbols

Symbol		Page
V	Tangential velocity	23
C_D	Drag coefficient	23
A_p	Projected area	23
μ	Standard gravitational parameter	23
r_{orbit}	Orbital radius	23
$\phi_{RightAil}$	Right aileron control angle	26
$\phi_{LeftAil}$	Left aileron control angle	26
ϕ_{Wedge}	Wedge angle	26
ϕ_{Arm}	Control arm angles	27
$\hat{v} =$	Velocity unit vector	29
L_{Arm}	Length of control arms	32
L_{Panel}	Length of sides for square control panels	33
$\phi_{0_{Arm}}$	Linearization angle for the control arms	34
$\phi_{0_{Ail}}$	Linearization angle for the ailerons	34
C_{RAil}	Linearized torque constant multiplying $\delta\phi_{RightAil}$	34
C_{LAil}	Linearized torque constant multiplying $\delta\phi_{LeftAil}$	34
C_{TArm}	Linearized torque constant multiplying $\delta\phi_{TopArm}$	34
C_{BArm}	Linearized torque constant multiplying $\delta\phi_{BotArm}$	34
C_{LArm}	Linearized torque constant multiplying $\delta\phi_{LeftArm}$	34
C_{RArm}	Linearized torque constant multiplying $\delta\phi_{RightArm}$	34
C_{θ_2}	Linearized torque constant multiplying θ_2	34
C_{θ_3}	Linearized torque constant multiplying θ_3	34
L_{Ail}	Length spanwise of the aileron surface	36
W_{Ail}	Length chordwise of the aileron surfaces	36
L_{panel}	Length of drag panel	36
W_{panel}	Width of drag panel	36
L_{b_1}	Length of satellite body along the $\hat{b}_1$ axis	36

Symbol		Page
L_{b_2}	Length of satellite body along the $\hat{b}_2$ axis	36
L_{b_3}	Length of satellite body along the $\hat{b}_3$ axis	36
$\delta\phi_{TopArm}$	Deflection angle from $\phi_{0_{Arm}}$ for the top arm	36
$\delta\phi_{BotArm}$	Deflection angle from $\phi_{0_{Arm}}$ for the bottom arm	36
$\delta\phi_{LeftArm}$	Deflection angle from $\phi_{0_{Arm}}$ for the left arm	36
$\delta\phi_{RightArm}$	Deflection angle from $\phi_{0_{Arm}}$ for the right arm	36
$\delta\phi_{LeftAil}$	Deflection angle from $\phi_{0_{Ail}}$ for the left aileron	36
$\delta\phi_{RightAil}$	Deflection angle from $\phi_{0_{Ail}}$ for the right aileron	36
C_{X_1}	Constant multiplying $\dot{\theta}$	36
C_{X_2}	Constant multiplying θ	36
C_{X_3}	Constant multiplying control outputs	36
Λ	Eigenvalues	39
ϵ	Eigenvectors	40
τ	Time Constant (τ)	59
Amp_0	Initial Amplitude)	59

List of Abbreviations

Abbreviation		Page
LEO	Low Earth Orbit	1
CG	Center of Gravity	2
GAMES	Gravity and Magnetic Earth Surveyor	6
CP	Center of Pressure	6
LVLH	Local Vertical Local Horizontal	14
EOM	Equations of Motion	18
MOI	Moment of Inertia	21
LEO	Low Earth Orbit	144

Satellite Attitude Control Using Atmospheric Drag

I. Introduction

Over the last several decades since Sputnik became the first artificial satellite to orbit the Earth in 1957, satellites have become an integral part of our lives. Satellites serve many useful purposes such as communications, weather and remote sensing to name a few. As their missions differ, so do their orbits. Many of these missions require the spacecraft to be in a low Earth orbit (LEO)

Some advantages for putting a spacecraft in LEO are:

- Higher resolution for Earth sensing satellites.
- Allows for smaller/lower mass payloads.
- Less costly to get to LEO.
- Shorter orbital periods for rapid revisits.
- No fuel needed for placement into a graveyard orbit.

There are also disadvantages for putting spacecraft in LEO:

- Atmospheric drag.
- Finite orbit lifetime.
- Attitude control issues.
- Additional fuel for orbit/attitude maintenance.

The atmosphere is one of the dominating contributors to orbit and attitude perturbations for spacecraft in LEO and accounts for most of the disadvantages.

The atmosphere is the dominating contributor to these perturbations from altitudes ranging from 150-600 km [9, page 81]. The research presented here only focuses on these altitudes where the Earth's atmosphere has a significant effect on spacecraft.

Measurements show that atmospheric density tends to decrease exponentially with increasing altitude and can be easily modeled (see Appendix C). Since the atmosphere doesn't end abruptly, there are fewer air molecules at LEO altitudes. As the spacecraft orbits the Earth, it hits these molecules and loses some of its orbital energy. Although the energy loss is extremely small, over time the losses add up causing the orbit to decay thus leading to a finite orbital lifetime. There is no way to stop orbit decay other than having onboard thrusters for orbit maintenance. On the other hand, collisions with atmospheric particles can affect the spacecraft's attitude. If a satellite has a center of pressure that is not in line with it's center of mass or center of gravity (CG) a small torque will be produced. If the spacecraft utilizes an attitude control system other than gravity gradient, it will most likely use thrusters for attitude control or for momentum dumping of reaction wheels. The torques caused by the atmosphere will increase the spacecraft's fuel consumption thus reducing it's life. Gyros and reaction wheels can also be prone to failure which can render a satellite useless.

1.1 Research Objectives

The exponential decay of the density of the atmosphere as a function of increasing altitude is always figured into satellite design for LEO spacecraft. Most attitude control systems used to overcome perturbations tend to be complex and expensive. The mission specifies the altitude of the orbit, from which orbital velocity and atmospheric density can be calculated. The only variables we can easily control are the projected areas and the location of the projected area relative to the spacecraft's center of gravity. By changing the size and location of the projected area, torques can be produced from the drag the projected areas experience. This research looks

into the feasibility of using drag panels to produce torques on a spacecraft to damp oscillations and control the spacecraft's attitude. Since the forces on the drag panels are very small, the drag panels can be as simple as having a piece of foil mounted to a stiff lightweight frame, which will then be connected to an actuator. An attitude control system using drag panels would be less complex and less prone to failure.

II. Background

2.1 Literature Review

Atmospheric drag has had an effect on most satellites starting with Sputnik. Sputnik 1 was launched on 4 October 1957 with an initial apogee of approximately 950 km and a perigee of approximately 230 km. At its perigee, the atmosphere had a significant affect on the satellite which eventually circularized its orbit (apogee was approximately 600 km by 9 December 1957). Sputnik's orbit eventually decayed to the point of re-entry after 92 days in orbit on 4 January 1958. Since then, many satellites, weather balloons, and sounding rockets have been launched to study the atmosphere. Some of the studies that have been done include both active and passive attitude stabilization techniques using atmospheric drag. Most of the research in satellite aero stabilization deals with passive stabilization. Little research has been done using active attitude stabilization using atmospheric drag.

2.1.1 Paddlewheel Satellites. By 1968, about a dozen satellites, such as Explorer VI (see Figure 2.1), have been launched to study solar wind and the magnetosphere. While these satellites were in orbit, they revealed information about the aerodynamic interaction of air molecules with the satellite surfaces. According to the data, air molecules experience nearly diffuse reflections. Maxwell's classical model best approximated this interaction. Maxwell's model accounts for both specular and diffuse reflections. Scientific analyses of the behavior of paddlewheel satellites led to a good approximation of the accommodation coefficients from which an approximated drag coefficient could be determined. [4]

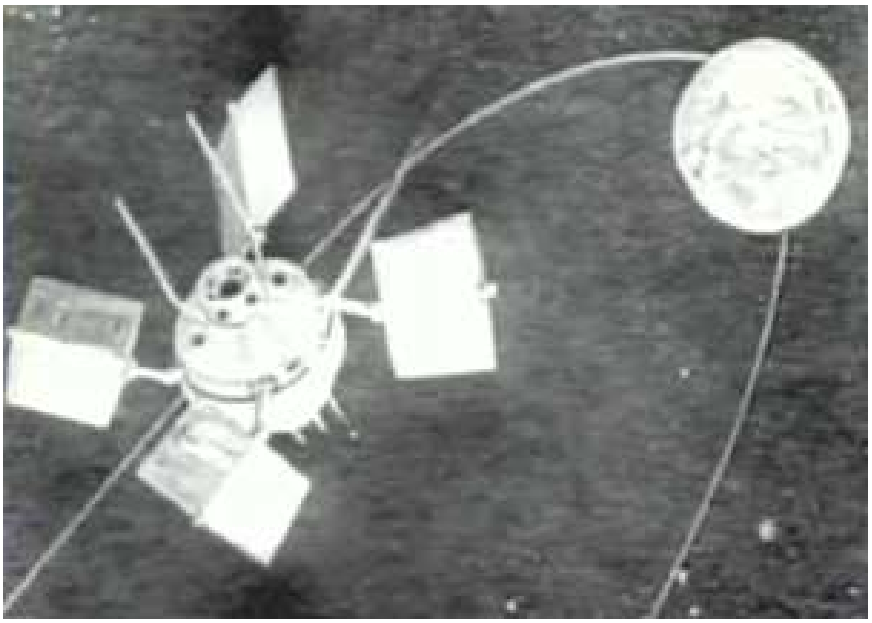


Figure 2.1: Explorer VI

2.1.2 Nanosatellite Passive Attitude stabilization. Psiaki performed a study using passive attitude stabilization for a nanosatellite [5]. This satellite was a cubesat with dimensions of 0.1 m for each side and mass of 1 kg. This satellite was designed to use passive drag torques to stabilize the roll, pitch and yaw axes and provide magnetic damping on both the pitch and yaw axes. The satellite resembles a shuttlecock used in badminton (see Figure 2.2). The conclusion of this study determined that the nanosatellite performed reasonably well in simulations.

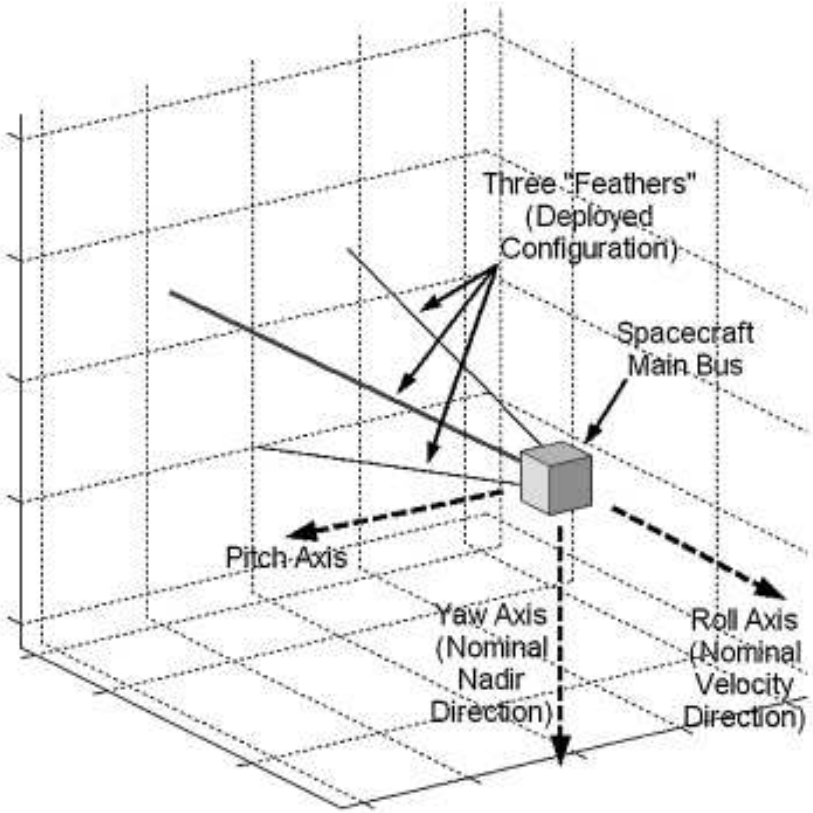
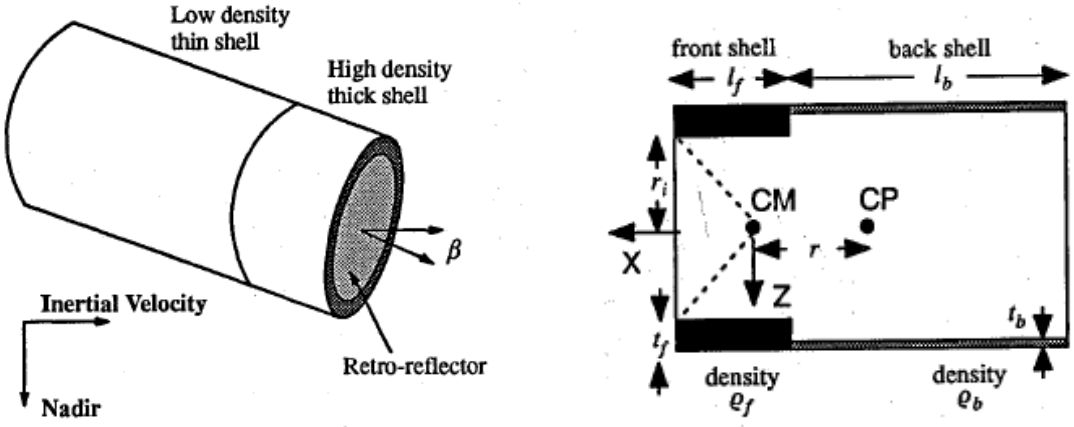


Figure 2.2: Nanosatellite Passive Attitude Stabilization Design

2.1.3 Shuttle Hitchhiker Passive Aerostabilization. Another study focused on purely passive stabilization with magnetic damping. This was a NASA study for the feasibility for a low cost, low weight, and long life spacecraft for the gravity and magnetic Earth surveyor (GAMES) mission . It is very similar to the satellite pictured in Figure 2.2 as far as size, mass and method used for aero-stabilization. Both use magnetic damping but the physical design is different. The shuttle hitchhiker is a cylindrical design vs. a cubesat with "feathers" (see Figure 2.3). In this design, the center of pressure (CP) is aft of the CG which tends to keep it pointed in the direction tangent to the orbit. [7, 8]



(a) (b)

Figure 2.3: (a) Shuttle Hitchhiker schematic(b) Cross section view

2.1.4 Active Aerodynamic Attitude Stabilization. Very little research has been done with active stabilization using atmospheric drag. Ravindran and Hughes [6] researched an Earth oriented satellite using drag panels for attitude control. Their research was closely related to the research presented here, but their satellite design was very different. Their design consisted of long cylindrical body where the axis through the cylinder is aligned with the roll axis (see Figure 2.4). The panels are turned so they are in the most streamlined position when no control is necessary. For pitch and yaw controls, the appropriate panels would be rotated 90° to increase the drag which is offset from the CG thus producing a torque. For roll control, they would all be rotated a specified angle turning the spacecraft into a propeller like configuration. Results from their analysis are similar to results obtained from this research. [6]

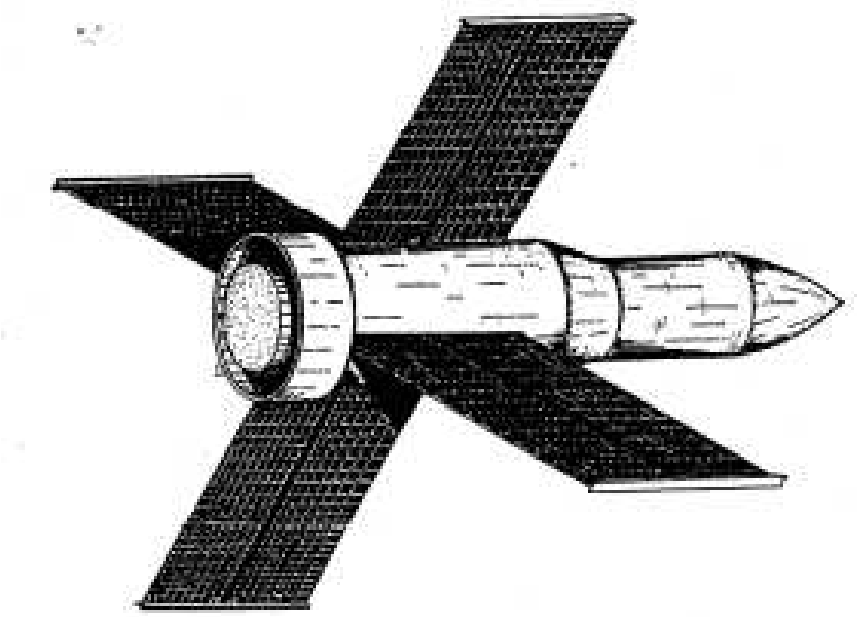


Figure 2.4: Active Control Design

2.2 The Atmosphere

The atmosphere has been studied for quite some time. Dozens of satellites, weather balloons and sounding rockets have been launched to answer the question "How does the pressure, temperature and density change as altitude changes?" Answers to this question have been crucial to the study of aeronautics and astronautics. Most people believe that space is empty and void. The truth is, space is mostly void and empty however there are particles flying around. According to the 1976 standard atmospheric model [3], the atmospheric density can be approximated by using an exponential model. See Appendix C for details on how the density was modeled. The models used for atmospheric density only predict an average value based on solar activity. Solar activity can have a significant affect on the actual density by heating and expanding the atmosphere to higher altitudes. Although the atmospheric density model only approximates the density, the dynamic model should still accurately predict the dynamic behavior of the linearized satellite.

2.3 Design Concepts

As progress was made with this research, the satellite design was altered to simplify the linearization process for the torque equations. Three design iterations were performed and the satellite evolved into the final version as presented in this section as Design 3. This section summarizes these design variations and why they changed.

2.3.1 Design 1: 4 Drag Panels to control Roll, Pitch and Yaw. The first design had 4 drag panels, each on a control arm with two degrees of freedom for each drag panel. This design enabled 3 axis control about the roll ($\hat{b}_1$), pitch ($\hat{b}_2$), and yaw ($\hat{b}_3$) axes (See Figure 2.5). The panels could be rotated into the air stream about the $\hat{b}_2$ axis for pitch control (top and bottom arms) and the $\hat{b}_3$ axis for yaw control (left and right arms). For roll control, all control arms could be rotated into the air stream and each panel rotated about each arm for left and right roll control. See Figures 2.6 (a) and (b), and 2.7 (a) and (b). Design 1 is very similar in design as the Explorer VI but with moveable panels. One drawback is that the control actuators become more complex having to rotate about two different axes. This design was abandoned since linearizing the equations of motino were became unmanageable. The nonlinear equation derivations for this design are included in Appendix F.

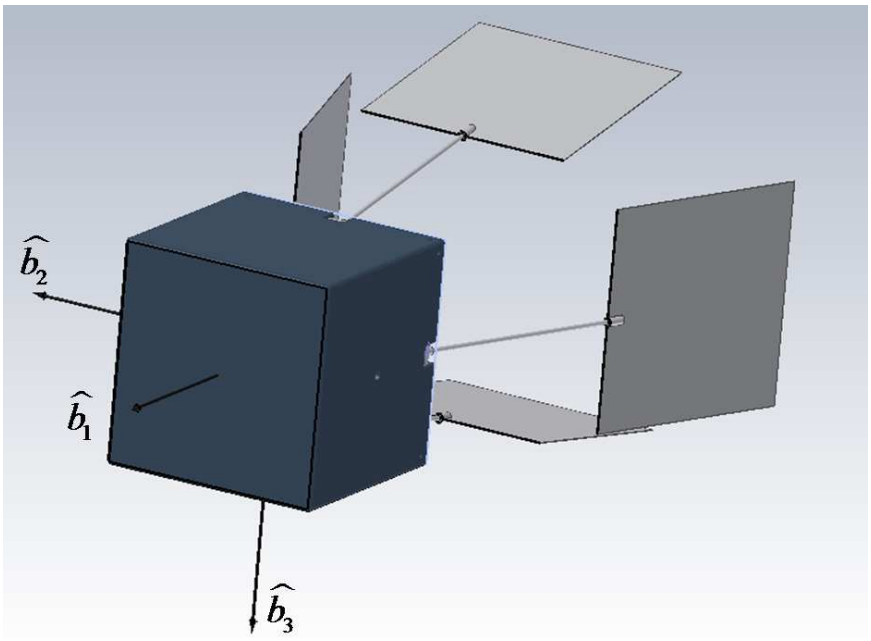
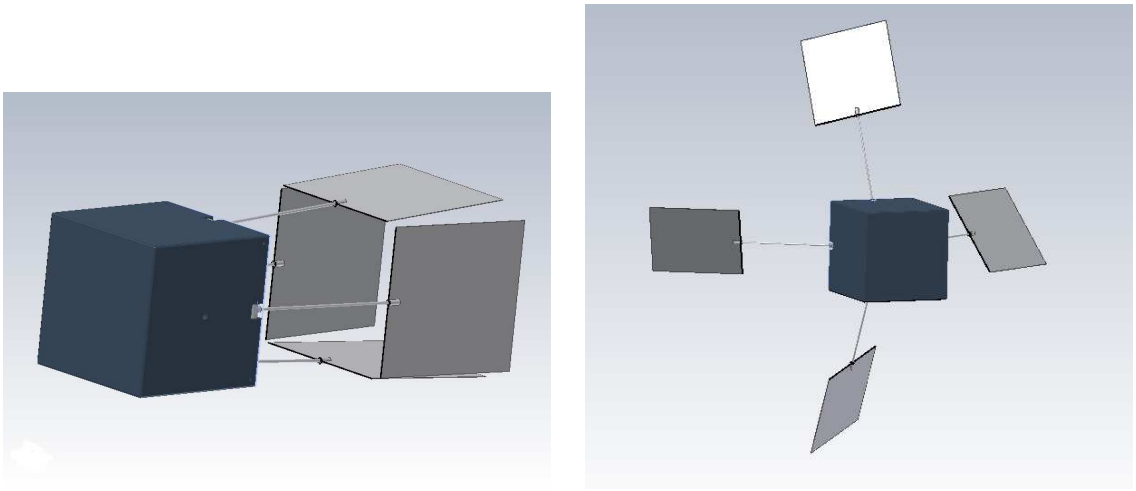
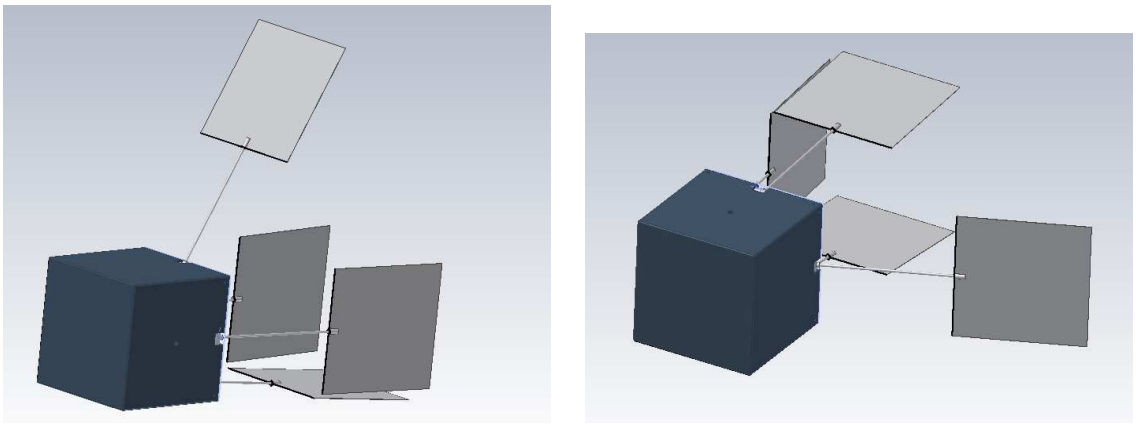


Figure 2.5: Roll ($\hat{b}_1$), Pitch ($\hat{b}_2$)and Yaw ($\hat{b}_3$) Axes



(a) (b)

Figure 2.6: (a) Panels retracted for normal flight.
(b) Panels extended and rotated for left rolling maneuver.



(a) (b)

Figure 2.7: (a) Top panel extended for a pitch up maneuver.
(b) Left panel extended for a left yaw maneuver.

2.3.2 Design 2: Ailerons to control Roll, 4 Drag Panels to Control Pitch and Yaw. Due to the complexity of the linearization of the torque equations in Design 1, a revision to the original design was made to simplify the equations. This design was similar to the first design. Pitch and yaw were controlled using the same drag panels. In this case, the control panels only had one degree of freedom and now only control pitch and yaw. "Ailerons" for lack of a better term, were added for roll control. If an actual satellite were to be built and tested, either this design or the previous design would be the best choice. Figure 2.8 shows the addition of ailerons to the sides of the spacecraft bus.

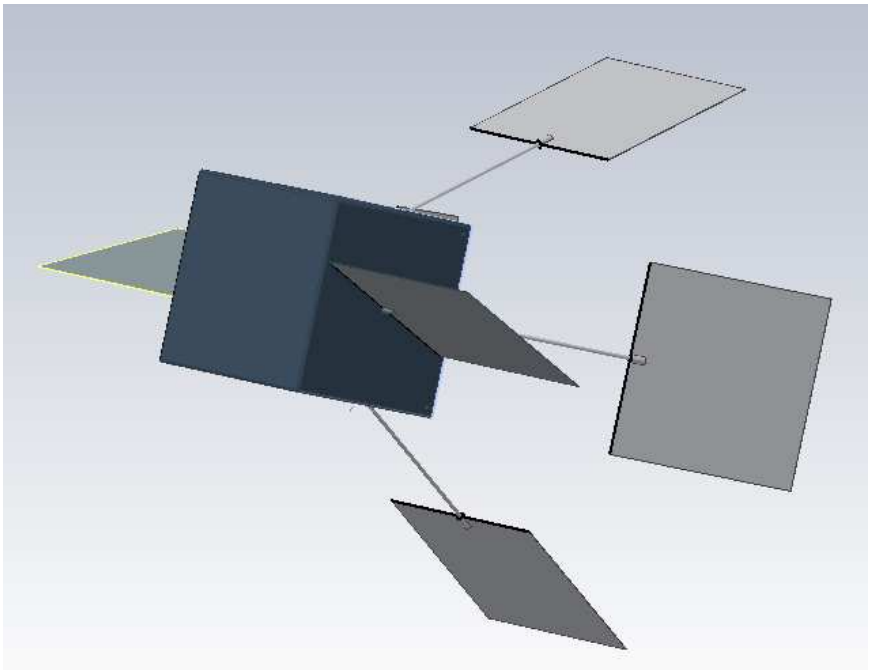


Figure 2.8: Satellite shown with ailerons in a right rolling maneuver.

2.3.3 Design 3: The third and final design was very similar to the previous design using ailerons for roll control. The only difference between the two models is that the ailerons were changed from flat plates to wedge shapes. This simplified the linearization by eliminating negative angles for the aileron surfaces. This design

is shown in Figure 2.9. All simulations presented in this research are based on the wedge shaped aileron design.

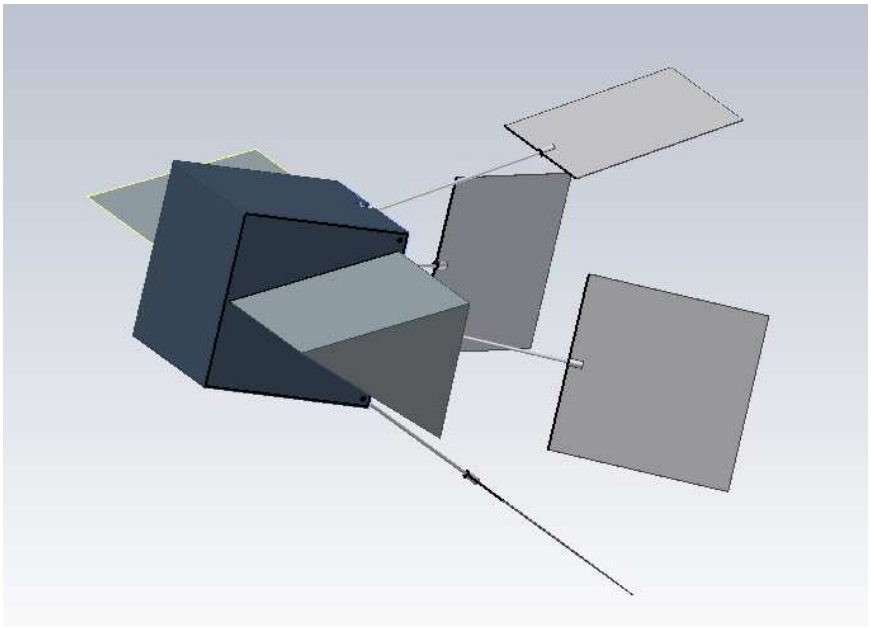


Figure 2.9: Satellite shown with wedges for ailerons.

III. Methodology

The equations of motion and torque equations depend on a satellite's geometry, physical dimensions and orbit. When deriving the equations, all terms were kept as variables to give the model more flexibility while running simulations.

3.1 Coordinate Systems

Three reference frames are used for the rotational kinematic equations. The "B frame", a body fixed reference frame is aligned with the principal moments of inertia, or the principal axes for the satellite, where $\hat{b}_1$ is the roll axis, $\hat{b}_2$ is the pitch axis and $\hat{b}_3$ is the yaw axis. The "A frame" is the local vertical, local horizontal (LVLH) frame aligned with the orbit where $\hat{a}_1$ is along the orbit direction, $\hat{a}_2$ is perpendicular to the orbit plane and $\hat{a}_3$ is toward the center of the Earth. Finally the inertial "I frame" is an Earth fixed frame with its origin at the center of the Earth. (See Figure 3.1) [10, page 366].

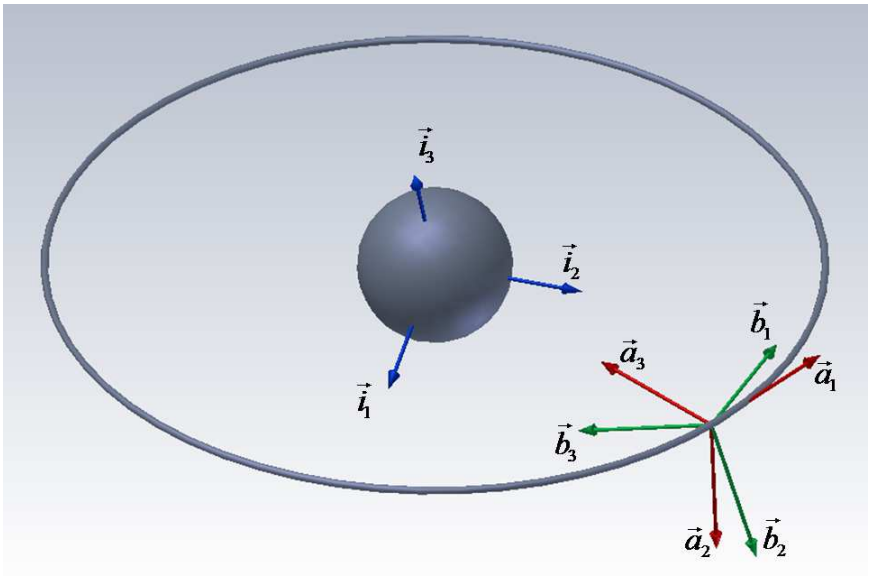


Figure 3.1: Coordinate frames for a rigid body in circular orbit

3.2 Rotation Matrices Between Coordinate Frames

The first step in deriving the equations of motion is to derive the rotation matrices. A 3-2-1 (yaw, pitch, roll) rotation sequence was used (see Figure 3.2) where each elementary rotation has the following rotation matrices associated with it. Note $c(\theta_i)= \cos(\theta_i)$, $s(\theta_i)= \sin(\theta_i)$, and θ_i are the rotation angles for each of the roll, pitch and yaw axes [10, page 309].

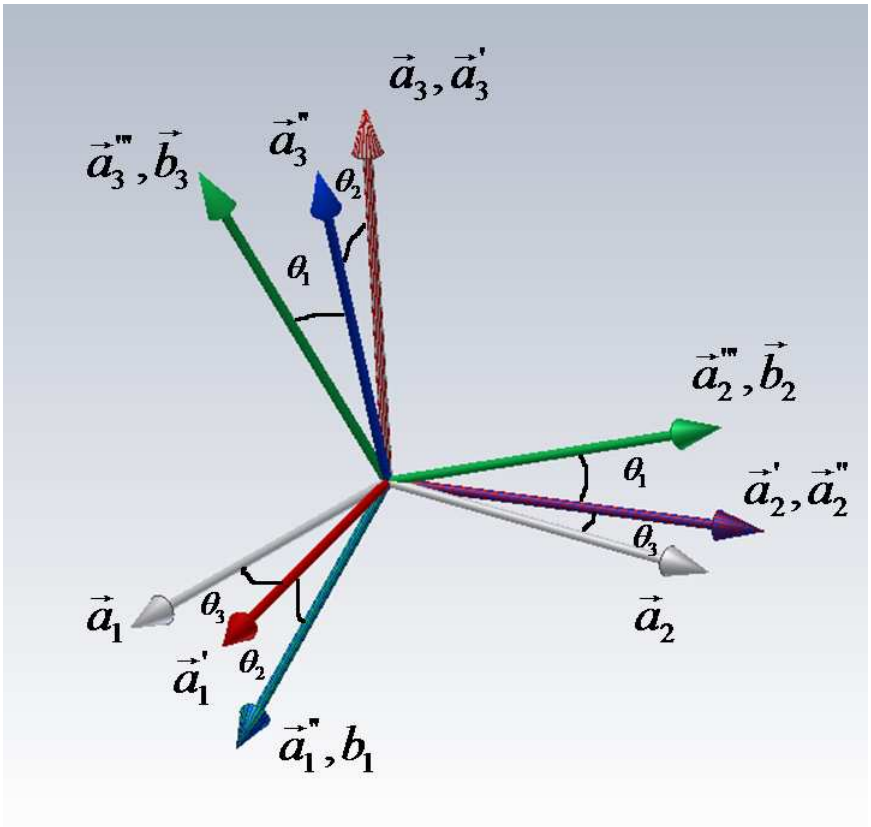


Figure 3.2: Euler Angle Rotations.

$$C_3(\theta_3) = \begin{bmatrix} c(\theta_3) & s(\theta_3) & 0 \\ -s(\theta_3) & c(\theta_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1a}$$

$$C_2(\theta_2) = \begin{bmatrix} c(\theta_2) & 0 & -s(\theta_2) \\ 0 & 1 & 0 \\ s(\theta_2) & 0 & c(\theta_2) \end{bmatrix} \tag{3.1b}$$

$$C_1(\theta_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\theta_1) & s(\theta_1) \\ 0 & -s(\theta_1) & c(\theta_1) \end{bmatrix} \tag{3.1c}$$

The rotation matrix to the B frame from the A frame is defined as $R^{BA} = C_1(\theta_1)C_2(\theta_2)C_3(\theta_3)$ and when expanded becomes [10, page 311]:

$$R^{BA} = \begin{bmatrix} c(\theta_2)c(\theta_3) & c(\theta_2)s(\theta_3) & -s(\theta_2) \\ s(\theta_1)s(\theta_2)c(\theta_3) - c(\theta_1)s(\theta_3) & s(\theta_1)s(\theta_2)s(\theta_3) + c(\theta_1)c(\theta_3) & s(\theta_1)c(\theta_2) \\ c(\theta_1)s(\theta_2)c(\theta_3) + s(\theta_1)s(\theta_3) & c(\theta_1)s(\theta_2)s(\theta_3) - s(\theta_1)c(\theta_3) & c(\theta_1)c(\theta_2) \end{bmatrix} \tag{3.2}$$

To get the rotation matrix to the A frame from the B frame the transpose must be taken since $R^{AB} = (R^{BA})^{-1} = (R^{BA})^T$, [10, page 311]. R^{AB} becomes:

$$R^{AB} = \begin{bmatrix} c(\theta_2)c(\theta_3) & s(\theta_1)s(\theta_2)c(\theta_3) - c(\theta_1)s(\theta_3) & c(\theta_1)s(\theta_2)c(\theta_3) + s(\theta_1)s(\theta_3) \\ c(\theta_2)s(\theta_3) & s(\theta_1)s(\theta_2)s(\theta_3) + c(\theta_1)c(\theta_3) & c(\theta_1)s(\theta_2)s(\theta_3) - s(\theta_1)c(\theta_3) \\ -s(\theta_2) & s(\theta_1)c(\theta_2) & c(\theta_1)c(\theta_2) \end{bmatrix} \tag{3.3}$$

Finally, to get to the B frame from the A frame, or to the A frame from the B frame, the equations are as follows [10, pages 365-366].

$$\begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix} = R^{BA} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{bmatrix} \tag{3.4}$$

$$\begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{bmatrix} = R^{AB} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix} \tag{3.5}$$

Since $\hat{a}_i$ contains only orthonormal unit vectors, $\hat{a} \cdot \hat{a}'$ becomes the identity matrix and equation 3.4 can be solved for R^{BA}:

$$R^{BA} = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix} \begin{bmatrix} \hat{a}_1 & \hat{a}_2 & \hat{a}_3 \end{bmatrix} = \begin{bmatrix} \hat{b}_1\hat{a}_1 & \hat{b}_1\hat{a}_2 & \hat{b}_1\hat{a}_3 \\ \hat{b}_2\hat{a}_1 & \hat{b}_2\hat{a}_2 & \hat{b}_2\hat{a}_3 \\ \hat{b}_3\hat{a}_1 & \hat{b}_3\hat{a}_2 & \hat{b}_3\hat{a}_3 \end{bmatrix} \tag{3.6}$$

3.3 Equations of Motion of a Rigid Body in Circular Orbit

The next step was to derive the equations of motion (EOM). This derivation is extracted from Wie [10, pages 365-369]. Several assumptions were made to simplify the problem and these assumptions would have little impact on the problem.

- Spacecraft is in a circular equatorial orbit (constant orbital rate).
- Uniform gravitational field, (gravity gradient torques are neglected).
- Spacecraft is in low Earth orbit (200-600 km)
- Atmospheric density is taken as an average over the orbit (no fluctuations in density).

3.3.1 Angular Velocity Vector. From the coordinate system shown in Figure 3.1, the angular velocity between the B frame and the Earth frame I frame) is:

$$\vec{\omega} \equiv \vec{\omega}^{BI} = \vec{\omega}^{BA} + \vec{\omega}^{AI} \tag{3.7}$$

and written in the body frame

$$\vec{\omega} = \omega_1 \hat{b}_1 + \omega_2 \hat{b}_2 + \omega_3 \hat{b}_3 = \begin{bmatrix} \hat{b}_1 & \hat{b}_2 & \hat{b}_3 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \tag{3.8}$$

Where $\vec{\omega}^{BI}$ is the angular velocity vector of the Earth fixed "I" frame to the body frame, $\vec{\omega}^{AI}$ is the angular velocity of the I frame to the A frame, and $\vec{\omega}^{BA}$ is the angular velocity from the A frame to the B frame.

Since the spacecraft is assumed to be in a circular orbit, $\vec{\omega}^{AI} = -n\hat{a}_2$, and therefore equation 3.7 becomes:

$$\vec{\omega}^{BI} = \vec{\omega}^{BA} - n\hat{a}_2 \tag{3.9}$$

where n is the constant orbital rate due to a circular orbit.

From equation 3.5, $\hat{a}_2$ can be written as:

$$\hat{a}_2 = \begin{bmatrix} \hat{b}_1 & \hat{b}_2 & \hat{b}_3 \end{bmatrix} \begin{bmatrix} c(\theta_2)s(\theta_3) \\ s(\theta_1)s(\theta_2)s(\theta_3) + c(\theta_1)c(\theta_3) \\ c(\theta_1)s(\theta_2)s(\theta_3) - s(\theta_1)c(\theta_3) \end{bmatrix} \tag{3.10}$$

To find $\vec{\omega}^{BA}$, a yaw, pitch, roll or 3-2-1 sequence was used. The first rotation about the yaw ($\hat{a}_3$) axis goes from the A axis to the A' axis. The second rotation about the $\hat{a}'_2$ axis takes us from the A' to the A'' axis and finally the third rotation about the $\hat{a}''_1$ axis goes from the A'' to the B frame (See Fig. 3.2). From the rotations, the angular velocity vector from the A frame to the B frame becomes [10, pages 324-326]:

$$\omega^{A'A} = \dot{\theta}_3 \vec{a}_3 = \dot{\theta}_3 \vec{a}'_3 \tag{3.11}$$

$$\omega^{A''A'} = \dot{\theta}_2 \vec{a}'_2 = \dot{\theta}_2 \vec{a}''_2 \tag{3.12}$$

$$\omega^{BA''} = \dot{\theta}_1 \vec{a}''_1 = \dot{\theta}_1 \vec{b}_1 \tag{3.13}$$

Where $\dot{\theta}_i$ is the angular rate of the roll (i=1), pitch (i=2) and yaw (i=3) axes. The total angular velocity vector then becomes:

$$\vec{\omega}^{BA} = \vec{\omega}^{BA''} + \vec{\omega}^{A''A'} + \vec{\omega}^{A'A} = \dot{\theta}_3 \vec{a}'_3 + \dot{\theta}_2 \vec{a}''_2 + \dot{\theta}_1 \vec{b}_1 \tag{3.14}$$

By substituting equations 3.11, 3.12, and 3.13 into 3.14, we get:

$$\vec{\omega}^{BA} = \begin{bmatrix} \hat{b}_1 & \hat{b}_2 & \hat{b}_3 \end{bmatrix} \left[\begin{bmatrix} \dot{\theta}_1 \\ 0 \\ 0 \end{bmatrix} + c_1(\theta_1) \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ 0 \end{bmatrix} + c_1(\theta_1)c_2(\theta_2) \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_3 \end{bmatrix} \right] \tag{3.15}$$

By combining equations 3.7 and 3.8, and substituting in equations 3.1b, 3.1c, 3.10, and 3.15, the vector $\begin{bmatrix}\hat{b}_1 & \hat{b}_2 & \hat{b}_3\end{bmatrix}$ cancels and the equation becomes [10, page 368]:

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} = \begin{bmatrix}1 & 0 & -s(\theta_2)\\ 0 & c(\theta_1) & s(\theta_1)c(\theta_2)\\ 0 & -s(\theta_1) & c(\theta_1)c(\theta_2)\end{bmatrix}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix} - n\begin{bmatrix}c(\theta_2)s(\theta_3)\\ s(\theta_1)s(\theta_2)s(\theta_3)+c(\theta_1)c(\theta_3)\\ c(\theta_1)s(\theta_2)s(\theta_3)-s(\theta_1)c(\theta_3)\end{bmatrix} \tag{3.16}$$

By solving for $\dot{\vec{\theta}}_1$, the result is the kinematic differential equation for an orbiting rigid body [10, page 368]:

$$\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix} = \frac{1}{c(\theta_2)}\begin{bmatrix}c(\theta_2) & s(\theta_1)s(\theta_2) & c(\theta_1)s(\theta_2)\\ 0 & c(\theta_1)c(\theta_2) & -s(\theta_1)c(\theta_2)\\ 0 & s(\theta_1) & c(\theta_1)\end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} + \frac{n}{c(\theta_2)}\begin{bmatrix}s(\theta_3)\\ c(\theta_2)c(\theta_3)\\ s(\theta_2)s(\theta_3)\end{bmatrix} \tag{3.17}$$

The following equation is the equation 3.16 expanded:

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} = \begin{bmatrix}\dot{\theta}_1 - s(\theta_2)\dot{\theta}_3 + c(\theta_2)s(\theta_3)n\\ c(\theta_1)\dot{\theta}_2 + s(\theta_1)c(\theta_2)\dot{\theta}_3 + ns(\theta_1)s(\theta_2)s(\theta_3) + nc(\theta_1)c(\theta_3)\\ -s(\theta_1)\dot{\theta}_2 + c(\theta_1)c(\theta_2)\dot{\theta}_3 + nc(\theta_1)s(\theta_2)s(\theta_3) - ns(\theta_1)c(\theta_3)\end{bmatrix} \tag{3.18}$$

Since θ_i is very small, equation 3.18 can be linearized by letting $\cos(\theta_i) = 1$, $\sin(\theta_i) = \theta_i$, $\sin(\theta_i)\sin(\theta_i) \approx 0$ and $\sin(\theta_i)\dot{\theta}_i \approx 0$. The linearized equation becomes [10, page 369].

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} = \begin{bmatrix}\dot{\theta}_1 - n\theta_3\\ \dot{\theta}_2 - n\\ \dot{\theta}_3 + n\theta_1\end{bmatrix} \tag{3.19}$$

Next we take the derivative of equation 3.19 to get $\dot{\vec{\omega}}_i$.

$$\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} = \begin{bmatrix} \ddot{\theta}_1 - n\dot{\theta}_3 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 + n\dot{\theta}_1 \end{bmatrix} \tag{3.20}$$

3.3.2 Time Derivatives of the Angular Momentum Vector. This derivation follows Wie [10, pages 340-342]. A rigid body has an angular momentum associated with it, which is expressed as $\vec{H} = \hat{J} \cdot \vec{\omega}^{BI}$. By taking the derivative of $\vec{H}$ the rotational equation motion can be found where:

$$\left\{ \frac{d\vec{H}}{dt} \right\}_I = \left\{ \frac{d\vec{H}}{dt} \right\}_B + \vec{\omega}^{BI} \times \vec{H} \tag{3.21}$$

By using the transport theorem equation 3.21 becomes:

$$\hat{J} \cdot \dot{\vec{\omega}} + \vec{\omega} \times \hat{J} \cdot \vec{\omega} = \vec{M} \tag{3.22}$$

where $\hat{J}$ is the moment of inertia matrix (MOI), $\vec{M}$ is the external moment vector, $\vec{\omega}$ is the angular rate vector and $\dot{\vec{\omega}}$ is the angular acceleration vector. Equation 3.22 in matrix form becomes:

$$\begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} + \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \tag{3.23}$$

Since it is assumed that the B frame is aligned to the principal axes, equation 3.23 becomes:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} \begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} + \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \tag{3.24}$$

Where A is the MOI about the roll axis, B is the MOI about the pitch axis, C is the MOI about the yaw axis, and when multiplied out becomes:

$$\begin{bmatrix} A\dot{\omega}_1 - (B-C)\omega_2\omega_3 \\ B\dot{\omega}_2 - (C-A)\omega_1\omega_3 \\ C\dot{\omega}_3 - (A-B)\omega_1\omega_2 \end{bmatrix} = \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \tag{3.25}$$

By substituting equations 3.19 and 3.20 into equation 3.25, linearizing and simplifying, the equation becomes:

$$\begin{bmatrix} A\ddot{\theta}_1 + (B-A-C)n\dot{\theta}_3 + (B-C)n^2\theta_1 \\ B\ddot{\theta}_2 \\ C\ddot{\theta}_3 + (C+A-B)n\dot{\theta}_1 + (B-A)n^2\theta_3 \end{bmatrix} = \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \tag{3.26}$$

Finally by solving for $\ddot{\theta}$ and putting into matrix form, the equation becomes:

$$\begin{aligned} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & \frac{C+A-B}{A} * n \\ 0 & 0 & 0 \\ \frac{B-A-C}{C} * n & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \\ &+ \begin{bmatrix} \frac{C-B}{A} \cdot n^2 0 & 0 & \\ 0 & 0 & 0 \\ 0 & 0 & \frac{A-B}{C} \cdot n^2 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} + \begin{bmatrix} \frac{M_1}{A} \\ \frac{M_2}{B} \\ \frac{M_3}{C} \end{bmatrix} \end{aligned} \tag{3.27}$$

3.4 External Torques from Atmospheric Drag (Nonlinear)

Satellites in LEO experience an aerodynamic drag force given by [9, page 329]:

$$F_{drag} = \frac{1}{2}\rho V^2 C_D A_p \tag{3.28}$$

where F_{drag}= drag force (N), ρ= atmospheric density (kg/m^3), V= velocity (m/s), C_D= drag coefficient and A_p= projected area (m^2).

Since the satellite is assumed to be in a circular orbit, the tangential velocity V is constant which is given by [11, page 70]:

$$V = \sqrt{\frac{\mu}{r_{orbit}}} \tag{3.29}$$

where the standard gravitational parameter μ= $398600\frac{km^3}{s^2}$ and the orbital radius r_{orbit}= $6378.135km + Altitude$. Note, the velocity vector is in the $-\hat{a}_1$ direction.

The density ρ depends on altitude and is calculated by the `Matlab`® code in Appendix C. From Moe's research, a good experimental approximation for C_D is 2.2 [4, page 4]. The projected areas A_p and their locations with respect to the CG are the only parameters that can be controlled after the spacecraft is in orbit.

Altitudes above 125 km are in the free molecular flow regime [1, page 316]. In the free molecular flow regime, particles are typically modeled either as specular or diffuse reflections. A specular reflection assumes that molecules are perfectly elastic where the tangential velocity is constant and the normal velocity is equal and opposite before and after reflection. The diffuse model assumes the molecules are reflected in a diffuse manner and have no memory of previous velocities. Either model imparts a force normal to the surface and the reflections are wrapped up into the drag coefficient which as stated earlier was determined experimentally. Spacecraft in LEO always experience a small drag torque since it is not possible to construct a

spacecraft that has its CG at the exact geometric center. Some assumptions made for the following drag torque equations are:

- The drag panels are visible to the incoming molecules on the front/outward facing surfaces.
- The drag coefficient was assumed to be constant at 2.2 [4, page 4].
- Atmospheric density is constant at a constant altitude and averaged over the orbit.(neglecting solar effects and atmospheric perturbations).
- The incoming molecular velocity is equal and opposite the orbital velocity (non-rotating atmosphere).
- The mass of the control panels and arms are negligible compared to the mass of the satellite.

3.4.1 Spacecraft Configuration. The linearized model consists of a basic cube shaped satellite bus with drag panels that can be extended into the airstream for pitch and yaw control. Two wedges acting as "ailerons", to produce torques about the roll axis (See Figure 2.9). Wedges were chosen instead of flat panels to simplify the linearization of the equations of motion. When linearizing the torque equations about zero degrees for the ailerons, all variables would become higher order terms since the projected areas and linearization angles were zero. Since Higher order terms were to be neglected, there were no variables left in the equations. By modelling the spacecraft with wedge shaped ailerons, the linearization problem goes away as does the heaviside function in projected area equations.

3.4.2 Drag Effects from Spacecraft Body. The spacecraft body doesn't produce any torques due to drag unless the CG is not at the geometric center (assuming a symmetric spacecraft). Although, in the linearized torque equations, the CG location is taken to be at the geometric center thus drag effects from the spacecraft body are neglected in the linearized model. In the nonlinear equations in Appendix F, the

CG location can be moved off-center to produce small torques. No further analysis was done with drag from the spacecraft body. The remaining analysis focused on the control panels.

3.4.3 Drag Effects from Spacecraft Drag Panels. The first step was to derive the rotation matrix to transform the drag panel reference frames (the F_i frame) to the body frame. Each drag panel rotation was based on equation 3.30

$$R^{FB} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_1) & s(\phi_1) \\ 0 & -s(\phi_1) & c(\phi_1) \end{bmatrix} \begin{bmatrix} c(\phi_2) & 0 & -s(\phi_2) \\ 0 & 1 & 0 \\ s(\phi_2) & 0 & c(\phi_2) \end{bmatrix} \begin{bmatrix} c(\phi_3) & s(\phi_3) & 0 \\ -s(\phi_3) & c(\phi_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.30}$$

Each drag panel has only one degree of freedom, therefore equation 3.30 is simplified requiring one rotation per panel. Each panel's rotation matrix contains a single rotation about its respective axis. Figure 3.3 shows how each arm rotates with respect to the satellite bus, i is either the top, bottom, left, or right arm. Equation 3.30 for each panel simplifies to

$$R^{FB}{}_{Top} = \begin{bmatrix} c(-\phi_{TopArm}) & 0 & -s(-\phi_{TopArm}) \\ 0 & 1 & 0 \\ s(-\phi_{TopArm}) & 0 & c(-\phi_{TopArm}) \end{bmatrix} \tag{3.31a}$$

$$R^{FB}{}_{Bot} = \begin{bmatrix} c(\phi_{BotArm}) & 0 & -s(\phi_{BotArm}) \\ 0 & 1 & 0 \\ s(\phi_{BotArm}) & 0 & c(\phi_{BotArm}) \end{bmatrix} \tag{3.31b}$$

$$R^{FB}{}_{Left} = \begin{bmatrix} c(\phi_{LeftArm}) & s(\phi_{LeftArm}) & 0 \\ -s(\phi_{LeftArm}) & c(\phi_{LeftArm}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.31c}$$

$$R^{FB}{}_{Right} = \begin{bmatrix} c(-\phi_{RightArm}) & s(-\phi_{RightArm}) & 0 \\ -s(-\phi_{RightArm}) & c(-\phi_{RightArm}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.31d}$$

For the wedges, the angles of the faces are found by adding or subtracting the wedge

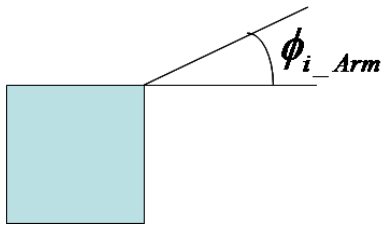


Figure 3.3: Angles for each control arm with respect to body frame

angle to get the total angle. The angles are defined as follows:

$$\phi_{LeftAilTop} = \phi_{LeftAil} - \phi_{Wedge} \tag{3.32a}$$

$$\phi_{LeftAilBot} = \phi_{LeftAil} + \phi_{Wedge} \tag{3.32b}$$

$$\phi_{RightAilTop} = \phi_{RightAil} - \phi_{Wedge} \tag{3.32c}$$

$$\phi_{RightAilBot} = \phi_{RightAil} + \phi_{Wedge} \tag{3.32d}$$

where the aileron control angles are $\phi_{RightAil} = -\phi_{LeftAil}$, and ϕ_{Wedge} is half the wedge angle. See Figure 3.4. The rotations for each surface then become

$$R^{FB}{}_{LeftAilTop} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(0) & s(0) \\ 0 & -s(0) & c(0) \end{bmatrix} \begin{bmatrix} c(\phi_{LeftAilTop}) & 0 & -s(\phi_{LeftAilTop}) \\ 0 & 1 & 0 \\ s(\phi_{LeftAilTop}) & 0 & c(\phi_{LeftAilTop}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33a}$$

$$R^{FB}{}_{LeftAilBot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(0) & s(0) \\ 0 & -s(0) & c(0) \end{bmatrix} \begin{bmatrix} c(\phi_{LeftAilBot}) & 0 & -s(\phi_{LeftAilBot}) \\ 0 & 1 & 0 \\ s(\phi_{LeftAilBot}) & 0 & c(\phi_{LeftAilBot}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33b}$$

$$R^{FB}{}_{RightAilTop} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(0) & s(0) \\ 0 & -s(0) & c(0) \end{bmatrix} \begin{bmatrix} c(\phi_{RightAilTop}) & 0 & -s(\phi_{RightAilTop}) \\ 0 & 1 & 0 \\ s(\phi_{RightAilTop}) & 0 & c(\phi_{RightAilTop}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33c}$$

$$R^{FB}{}_{RightAilBot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(0) & s(0) \\ 0 & -s(0) & c(0) \end{bmatrix} \begin{bmatrix} c(\phi_{RightAilBot}) & 0 & -s(\phi_{RightAilBot}) \\ 0 & 1 & 0 \\ s(\phi_{RightAilBot}) & 0 & c(\phi_{RightAilBot}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33d}$$

where the control arm angles ϕ_{Arm} are ϕ_{TopArm}, ϕ_{BotArm}, $\phi_{LeftArm}$, and $\phi_{RightArm}$, are the top, bottom, left and right arms that are connected to the drag panels. The angles between the $\hat{b}_1$ direction and the arms range from $0 - 90°$. In equations 3.31a and 3.31d, the arm angles have a negative sign so that all angles will be positive to make it easier to remember that all angles to be input to the controller are positive. Two of the rotation angles in each equation are always zero since there is only one degree of freedom for each panel.

The rotations from the A frame to the F frame are used to determine normal force for each panel due to the drag in the A frame. This is found by $R^{FA} = R^{FB}R^{BA}$ and thus becomes:

$$R^{FA}{}_{Top} = R^{FB}{}_{Top}R^{BA} \tag{3.34a}$$

$$R^{FA}{}_{Bot} = R^{FB}{}_{Bot}R^{BA} \tag{3.34b}$$

$$R^{FA}{}_{Left} = R^{FB}{}_{Left}R^{BA} \tag{3.34c}$$

$$R^{FA}{}_{Right} = R^{FB}{}_{Right}R^{BA} \tag{3.34d}$$

$$R^{FA}{}_{LeftAilTop} = R^{FB}{}_{LeftAilTop}R^{BA} \tag{3.34e}$$

$$R^{FA}{}_{LeftAilBot} = R^{FB}{}_{LeftAilBot}R^{BA} \tag{3.34f}$$

$$R^{FA}{}_{RightAilTop} = R^{FB}{}_{RightAilTop}R^{BA} \tag{3.34g}$$

$$R^{FA}{}_{RightAilBot} = R^{FB}{}_{RightAilBot} R^{BA} \tag{3.34h}$$

The angles between the incoming molecules and the inward normal directions are:

$$\cos(\alpha_{Top}) = (-\hat{a}_1 \cdot f_3) = -R^{FA}{}_{Top(3,1)} \tag{3.35a}$$

$$\cos(\alpha_{Bot}) = (-\hat{a}_1 \cdot -f_3) = R^{FA}{}_{Bot(3,1)} \tag{3.35b}$$

$$\cos(\alpha_{Left}) = (-\hat{a}_1 \cdot f_2) = -R^{FA}{}_{Left(2,1)} \tag{3.35c}$$

$$\cos(\alpha_{Right}) = (-\hat{a}_1 \cdot -f_2) = R^{FA}{}_{Right(2,1)} \tag{3.35d}$$

$$\cos(\alpha_{LeftAilTop}) = (-\hat{a}_1 \cdot f_3) = -R^{FA}{}_{LeftAilTop(3,1)} \tag{3.35e}$$

$$\cos(\alpha_{LeftAilBot}) = (-\hat{a}_1 \cdot -f_3) = R^{FA}{}_{LeftAilBot(3,1)} \tag{3.35f}$$

$$\cos(\alpha_{RightAilTop}) = (-\hat{a}_1 \cdot f_3) = -R^{FA}{}_{RightAilTop(3,1)} \tag{3.35g}$$

$$\cos(\alpha_{RightAilBot}) = (-\hat{a}_1 \cdot -f_3) = R^{FA}{}_{RightAilBot(3,1)} \tag{3.35h}$$

The projected areas from the panels become:

$$A_{Top} = L^2 H(\cos(\alpha_{Top}))\cos(\alpha_{Top}) \tag{3.36a}$$

$$A_{Bot} = L^2 H(\cos(\alpha_{Bot}))\cos(\alpha_{Bot}) \tag{3.36b}$$

$$A_{Left} = L^2 H(\cos(\alpha_{Left}))\cos(\alpha_{Left}) \tag{3.36c}$$

$$A_{Right} = L^2 H(\cos(\alpha_{Right}))\cos(\alpha_{Right}) \tag{3.36d}$$

$$A_{LeftAilTop} = L_{Ail} W_{Ail} H(\cos(\alpha_{LeftAilTop}))\cos(\alpha_{LeftAilTop}) \tag{3.36e}$$

$$A_{LeftAilBot} = L_{Ail} W_{Ail} H(\cos(\alpha_{LeftAilBot}))\cos(\alpha_{LeftAilBot}) \tag{3.36f}$$

$$A_{RightAilTop} = L_{Ail} W_{Ail} H(\cos(\alpha_{RightAilTop}))\cos(\alpha_{RightAilTop}) \tag{3.36g}$$

$$A_{RightAilBot} = L_{Ail} W_{Ail} H(\cos(\alpha_{RightAilBot}))\cos(\alpha_{RightAilBot}) \tag{3.36h}$$

where L is the side length of each square drag panel, L_{Ail} is the length (span) of each aileron and W_{Ail} is the width of each aileron. Calculate forces from the drag panels in the A frame:

$$F_{Top_A} = \frac{1}{2} C_D A_{Top} \rho V^2 \hat{v} \tag{3.37a}$$

$$F_{Bot_A} = \frac{1}{2} C_D A_{Bot} \rho V^2 \hat{v} \tag{3.37b}$$

$$F_{Left_A} = \frac{1}{2} C_D A_{Left} \rho V^2 \hat{v} \tag{3.37c}$$

$$F_{Right_A} = \frac{1}{2} C_D A_{Right} \rho V^2 \hat{v} \tag{3.37d}$$

$$F_{LeftAilTop_A} = \frac{1}{2} C_D A_{LeftAilTop} \rho V^2 \hat{v} \tag{3.37e}$$

$$F_{LeftAilBot_A} = \frac{1}{2} C_D A_{LeftAilBot} \rho V^2 \hat{v} \tag{3.37f}$$

$$F_{RightAilTop_A} = \frac{1}{2} C_D A_{RightAilTop} \rho V^2 \hat{v} \tag{3.37g}$$

$$F_{RightAilBot_A} = \frac{1}{2} C_D A_{RightAilBot} \rho V^2 \hat{v} \tag{3.37h}$$

where $\hat{v} =$ is the velocity unit vector and is defined as:

$$\hat{v} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{3.38}$$

Calculate force from drag panels in the F frame and pull out the inward normal component:

$$F_{Top_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} R^{FA}{}_{Top} F_{Top_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39a}$$

$$F_{Bot_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} {R^{FA}}_{Bot} F_{Bot_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39b}$$

$$F_{Left_F} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} {R^{FA}}_{Left} F_{Left_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.39c}$$

$$F_{Right_F} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} {R^{FA}}_{Right} F_{Right_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.39d}$$

$$F_{LeftAilTop_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} {R^{FA}}_{LeftAilTop} F_{LeftAilTop_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39e}$$

$$F_{LeftAilBot_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} {R^{FA}}_{LeftAilBot} F_{LeftAilBot_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39f}$$

$$F_{RightAilTop_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} {R^{FA}}_{RightAilTop} F_{RightAilTop_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39g}$$

$$F_{RightAilBot_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} {R^{FA}}_{RightAilBot} F_{RightAilBot_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39h}$$

Convert the inward normal component of each surface in the F frame to the B frame:

$$F_{Top_B} = ({R^{FB}}_{Top})^T F_{Top_F} \tag{3.40a}$$

$$F_{Bot_B} = ({R^{FB}}_{Bot})^T F_{Bot_F} \tag{3.40b}$$

$$F_{Left_B} = (R^{FB}{}_{Left})^T F_{Left_F} \tag{3.40c}$$

$$F_{Right_B} = (R^{FB}{}_{Right})^T F_{Right_F} \tag{3.40d}$$

$$F_{LeftAilTop_B} = (R^{FB}{}_{LeftAilTop})^T F_{LeftAilTop_F} \tag{3.40e}$$

$$F_{LeftAilBot_B} = (R^{FB}{}_{LeftAilBot})^T F_{LeftAilBot_F} \tag{3.40f}$$

$$F_{RightAilTop_B} = (R^{FB}{}_{RightAilTop})^T F_{RightAilTop_F} \tag{3.40g}$$

$$F_{RightAilBot_B} = (R^{FB}{}_{RightAilBot})^T F_{RightAilBot_F} \tag{3.40h}$$

Find r from the CG to the center of pressure for each panel. (See Figure 3.4 for aileron details.) (Note: CG is at the geometric center for the linearized model, i.e. CG=0):

$$r_{Top} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{TopArm}) \\ 0 \\ -0.5L - (L_{Arm} + 0.5L_{panel})\sin(\phi_{TopArm}) \end{bmatrix} - CG \tag{3.41a}$$

$$r_{Bot} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{BotArm}) \\ 0 \\ 0.5L + (L_{Arm} + 0.5L_{panel})\sin(\phi_{BotArm}) \end{bmatrix} - CG \tag{3.41b}$$

$$r_{Left} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{LeftArm}) \\ -0.5L - (L_{Arm} + 0.5L_{panel})\sin(\phi_{LeftArm}) \\ 0 \end{bmatrix} - CG \tag{3.41c}$$

$$r_{Right} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{RightArm}) \\ 0.5L + (L_{Arm} + 0.5L_{panel})\sin(\phi_{RightArm}) \\ 0 \end{bmatrix} - CG \tag{3.41d}$$

$$r_{LeftAilTop} = \begin{bmatrix} -0.5W_{Ail} \sin\phi_{Wedge} \sin\phi_{LeftAil} \\ -0.5(L_{Side} + L_{Ail}) \\ -0.5W_{Ail} \sin\phi_{Wedge} \cos\phi_{LeftAil} \end{bmatrix} - CG \tag{3.41e}$$

$$r_{LeftAilBot} = \begin{bmatrix} 0.5W_{Ail} \sin\phi_{Wedge} \sin\phi_{LeftAil} \\ -0.5(L_{Side} + L_{Ail}) \\ 0.5W_{Ail} \sin\phi_{Wedge} \cos\phi_{LeftAil} \end{bmatrix} - CG \tag{3.41f}$$

$$r_{RightAilTop} = \begin{bmatrix} -0.5W_{Ail} \sin\phi_{Wedge} \sin\phi_{RightAil} \\ 0.5(L_{Side} + L_{Ail}) \\ -0.5W_{Ail} \sin\phi_{Wedge} \cos\phi_{RightAil} \end{bmatrix} - CG \tag{3.41g}$$

$$r_{RightAilBot} = \begin{bmatrix} 0.5W_{Ail} \sin\phi_{Wedge} \sin\phi_{RightAil} \\ 0.5(L_{Side} + L_{Ail}) \\ 0.5W_{Ail} \sin\phi_{Wedge} \cos\phi_{RightAil} \end{bmatrix} - CG \tag{3.41h}$$

where L_{Side} is the side length of the spacecraft body, L_{Arm} is the length of the

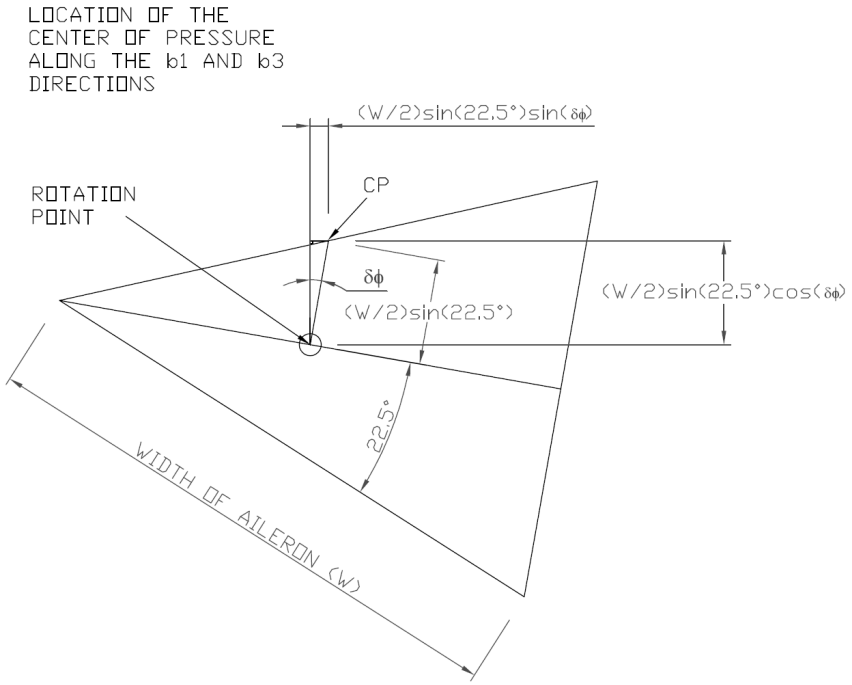


Figure 3.4: Aileron details

control arm, L_{Panel} is the side length of the square panels, ϕ_{LArm}, ϕ_{RArm}, ϕ_{BArm} and ϕ_{TArm} are the angles of the arms from their retracted positions, $\phi_{LeftAil}$ and $\phi_{RightAil}$ are the angles of the ailerons, and ϕ_{Wedge} is half the angle of the wedge.

Finally, the torque equations for the drag panels with all the substitutions become:

$$M_{Top} = r_{Top} \mathrm{x} F_{Top_B} \tag{3.42a}$$

$$M_{Bot} = r_{Bot} \mathrm{x} F_{Bot_B} \tag{3.42b}$$

$$M_{Left} = r_{Left} \mathrm{x} F_{Left_B} \tag{3.42c}$$

$$M_{Right} = r_{Right} \mathrm{x} F_{Right_B} \tag{3.42d}$$

$$M_{LeftAilTop} = r_{LeftAilTop} \mathrm{x} F_{LeftAilTop_B} \tag{3.42e}$$

$$M_{LeftAilBot} = r_{LeftAilBot} \mathrm{x} F_{LeftAilBot_B} \tag{3.42f}$$

$$M_{RightAilTop} = r_{RightAilTop} \mathrm{x} F_{RightAilTop_B} \tag{3.42g}$$

$$M_{RightAilBot} = r_{RightAilBot} \mathrm{x} F_{RightAilBot_B} \tag{3.42h}$$

To get the total torques for the drag control panels, equations 3.42 a-h) are added:

$$\begin{aligned} M_{Panels} &= M_{Top} + M_{Bot} + M_{Left} + M_{Right} + M_{LeftAilTop} \\ &+ M_{LeftAilBot} + M_{RightAilTop} + M_{RightAilBot} \end{aligned} \tag{3.43}$$

If the torques from the body were included, they would be added here, however in this case, $M_{SatBody} = 0$:

$$M_{Total} = M_{SatBody} + M_{Panels} \tag{3.44}$$

3.4.4 Linearized Torques from Control Panels. Linear analysis required the nonlinear torque equations to be linearized. By having the ailerons as wedges

and assuming small angles, the outward facing control surfaces were always facing the incoming molecules. Thus, the heaviside function was no longer necessary for the linear model. Each control arm for the pitch and yaw controls were linearized about $\phi_{0_{Arm}}$, which is the linearization angle of the control arms. The ailerons were linearized about $\phi_{0_{Ail}}$, where $\phi_{0_{Ail}} = 0^\circ$ (aligned with the $\hat{b}_1$ axis). After using both MathCad and `Matlab`® symbolic solvers, the combined and linearized torque equations were found:

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} = \begin{bmatrix} C_{RAil}\delta\phi_{RightAil} + C_{LAil}\delta\phi_{LeftAil}) \\ C_{BArm}\delta\phi_{BotArm} + C_{TArm}\delta\phi_{TopArm} + C_{\theta_2}\theta_2 \\ C_{LArm}\delta\phi_{LeftArm} + C_{RArm}\delta\phi_{RightArm} + C_{\theta_3}\theta_3 \end{bmatrix} \tag{3.45}$$

where C_{RAil}, C_{LAil}, C_{TArm}, C_{BArm}, C_{LArm}, C_{RArm}, C_{θ_2}, and C_{θ_3} are defined as follows:

$$\begin{aligned} C_{RAil} &= [0.5{L_{Ail}}^2 W_{Ail}\sin(\phi_{Wedge}) \\ &+ 0.5L_{Ail}W_{Ail}L_{b_2}\sin(\phi_{Wedge}) \\ &- 1.5{L_{Ail}}^2 W_{Ail}\sin(\phi_{Wedge})\cos^2(\phi_{Wedge}) \\ &- 1.5L_{Ail}W_{Ail}L_{b_2}\sin(\phi_{Wedge})\cos^2(\phi_{Wedge})]C_D\rho V^2 \end{aligned} \tag{3.46a}$$

$$C_{LAil} = -C_{RAil} \tag{3.46b}$$

$$\begin{aligned} C_{TArm} &= [0.5{L_{panel}}^2 W_{panel}\cos(\phi_{0_{Arm}})\sin(\phi_{0_{Arm}}) \\ &+ L_{panel}W_{panel}L_{Arm}\cos(\phi_{0_{Arm}})\sin(\phi_{0_{Arm}}) \\ &+ 0.75L_{panel}W_{panel}L_{b_1}\cos^2(\phi_{0_{Arm}})\sin(\phi_{0_{Arm}}) \\ &- 0.75L_{panel}W_{panel}L_{b_3}\cos^3(\phi_{0_{Arm}}) \\ &+ 0.75L_{panel}W_{panel}L_{b_3}\cos(\phi_{0_{Arm}}) \\ &- 0.25L_{panel}W_{panel}L_{b_1}\sin(\phi_{0_{Arm}})]C_D\rho V^2 \end{aligned} \tag{3.46c}$$

$$C_{BArm} = -C_{LArm} \tag{3.46d}$$

$$\begin{aligned} C_{LArm} &= [-0.5{L_{panel}}^2 W_{panel} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- L_{panel} W_{panel} L_{Arm} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- 0.75 L_{panel} W_{panel} L_{b_1} \cos^2(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &+ 0.75 L_{panel} W_{panel} L_{b_2} \cos^3(\phi_{0_{Arm}}) \\ &- 0.75 L_{panel} W_{panel} L_{b_2} \cos(\phi_{0_{Arm}}) \\ &+ 0.25 L_{panel} W_{panel} L_{b_1} \sin(\phi_{0_{Arm}})] C_D \rho V^2 \end{aligned} \tag{3.46e}$$

$$C_{RArm} = -C_{LArm} \tag{3.46f}$$

$$\begin{aligned} C_{\theta_2} &= [2 L_{Ail} {W_{Ail}}^2 \cos^3(\phi_{Wedge}) \sin(\phi_{Wedge}) \\ &- 2 L_{Ail} {W_{Ail}}^2 \cos(\phi_{Wedge}) \sin(\phi_{Wedge}) \\ &- {L_{panel}}^2 W_{panel} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- 2 L_{panel} W_{panel} L_{Arm} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- L_{panel} W_{panel} L_{b_1} \cos^2(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &+ L_{panel} W_{panel} L_{b_3} \cos^3(\phi_{0_{Arm}}) \\ &- L_{panel} W_{panel} L_{b_3} \cos(\phi_{0_{Arm}})] C_D \rho V^2 \end{aligned} \tag{3.46g}$$

$$\begin{aligned} C_{\theta_3} &= [L_{panel} W_{panel} L_{b_2} \cos^3(\phi_{0_{Arm}}) \\ &- 2 L_{panel} W_{panel} L_{Arm} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- L_{panel} W_{panel} L_{b_1} \cos^2(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- {L_{panel}}^2 W_{panel} \cos(\phi_{0_{Arm}}) \sin(\phi_{0_{Arm}}) \\ &- L_{panel} W_{panel} L_{b_3} \cos(\phi_{0_{Arm}})] C_D \rho V^2 \end{aligned} \tag{3.46h}$$

L_{Ail} is the spanwise length of the aileron surface, W_{Ail} is the chord length of the aileron surfaces, L_{panel} is the length of the drag panels, W_{panel} is the width of the drag panels, L_{b_1} is the length of the satellite body in the $\hat{b}_1$ direction, L_{b_2}, is the length of the satellite body in the $\hat{b}_2$ direction, and L_{b_3}, is the length of the satellite body in the $\hat{b}_3$ direction.

3.5 Combined Linearized Equations of motion

The final step in deriving the EOM was to combine all the equations. By plugging the linearized torque equations 3.45 into the overall equations of motion 3.27 and simplifying, the equations become:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} = C_{X_1} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} + C_{X_2} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} + C_{X_3} \cdot \begin{bmatrix} \delta\phi_{TopArm} \\ \delta\phi_{BotArm} \\ \delta\phi_{LeftArm} \\ \delta\phi_{RightArm} \\ \delta\phi_{LeftAil} \\ \delta\phi_{RightAil} \end{bmatrix} \tag{3.47}$$

where $\delta\phi_i$ is the deflection angles for the respective control arms and ailerons; $\delta\phi_{TopArm}$, $\delta\phi_{BotArm}$, $\delta\phi_{LeftArm}$, $\delta\phi_{RightArm}$, $\delta\phi_{LeftAil}$, $\delta\phi_{RightAil}$, and C_{X_1}, C_{X_2}, and C_{X_3} are defined as follows:

$$C_{X_1} = \begin{bmatrix} \frac{1}{A} & 0 & 0 \\ 0 & \frac{1}{B} & 0 \\ 0 & 0 & \frac{1}{C} \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & (C+A-B)n \\ 0 & 0 & 0 \\ (B-A-C)n & 0 & 0 \end{bmatrix} \tag{3.48}$$

$$C_{X_2} = \begin{bmatrix} \frac{1}{A} & 0 & 0 \\ 0 & \frac{1}{B} & 0 \\ 0 & 0 & \frac{1}{C} \end{bmatrix} \cdot \begin{bmatrix} (C-B)n^2 0 & 0 & \\ 0 & C_{\theta_2} & 0 \\ 0 & 0 & (A-B)n^2 + C_{\theta_3} \end{bmatrix} \tag{3.49}$$

$$C_{X_3} = \begin{bmatrix} \frac{1}{A} & 0 & 0 \\ 0 & \frac{1}{B} & 0 \\ 0 & 0 & \frac{1}{C} \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & C_{LAil} & C_{RAil} \\ C_{TArm} & C_{BArm} & 0 & 0 & 0 & 0 \\ 0 & 0 & C_{LArm} & C_{RArm} & 0 & 0 \end{bmatrix} \tag{3.50}$$

Equation 3.47 with all the substitutions is the final equation to be modeled in Simulink®.

IV. Dynamic Model

4.1 Dynamic Model Overview

The linearized dynamic equations 3.47 were modeled in Simulink®. All terms were left as variables. Inputs so different scenarios could be easily simulated by changing the input variables. If more details of a satellite are known, such as it's dimensions, mass and orbital altitude, this model can be easily updated to simulate a variety of satellites. The only restriction is that the satellite has to have the same basic geometric design.

4.2 Dynamic Model Validation

The following is the validation of the open loop and closed loop model by using eigenvalues and eigenvectors, where the eigenvalues are the frequencies (cycles/orbit) of this system and the eigenvectors determine initial conditions that will predict system response. The results of this analysis show how the general shape of the response. Such responses could be pure oscillatory or exponential. In retrospect, the eigenvectors could have been scaled so the amplitude of the responses remained small, however, having a larger amplitude response does not alter the outcome of this section. All simulations performed in this section were run at 300km altitude and the density was set to zero for the no torque case and the density was calculated for 300km altitude for the case with torques.

4.2.1 Dynamic Model Validation Without Torques. The first step was to set all external torques of equation 3.27 to zero. The rest of equation 3.27 was

rewritten in a state space ($\dot{X} = A \cdot X$) format as follows.

$$\begin{bmatrix} \dot{\theta_1} \\ \dot{\theta_2} \\ \dot{\theta_3} \\ \ddot{\theta_1} \\ \ddot{\theta_2} \\ \ddot{\theta_3} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{C-B}{A}n^2 & 0 & 0 & 0 & 0 & \frac{C+A-B}{A}n \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{A-B}{C}n^2 & \frac{B-A-C}{C}n & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_2} \\ \dot{\theta_3} \end{bmatrix} \tag{4.1}$$

From the state space equation (equation 4.2), the pitch axis (θ_2) has no influence on the other axes, hence it is decoupled from the roll (θ_1) and yaw (θ_3) axes. θ_2 was shown to be stable based on test simulations. Since θ_2 is stable and decoupled, all references to θ_2 were removed leaving the following equation.

$$\begin{bmatrix} \dot{\theta_1} \\ \dot{\theta_3} \\ \ddot{\theta_1} \\ \ddot{\theta_3} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{C-B}{A}n^2 & 0 & 0 & \frac{C+A-B}{A}n \\ 0 & \frac{A-B}{C}n^2 & \frac{B-A-C}{C}n & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_3} \end{bmatrix} \tag{4.2}$$

The next step was to test the model with different combinations of major, minor and intermediate axes for the principal moments of inertia. If the spacecraft is nadir pointing, the pitch axis will always be rotating with the orbit. If the pitch axis is either the minor or major axis of rotation, the spacecraft will be stable. If the pitch axis is the minor axis, the spacecraft will be in an unstable configuration. Several combinations of MOI were simulated where the moments of inertia about the roll, pitch and yaw axes are A, B, and C respectively. For the first simulation, the $\hat{b}_2$ axis was set as the minor axis, where $A = 20$, $B = 10$, and $C = 30$. The calculated eigenvalues λ_i are arranged in a diagonal matrix of eigenvalues (Λ) and the calculated

eigenvectors are the columns of (ϵ).

$$\Lambda = \begin{bmatrix} i \cdot n & 0 & 0 & 0 \\ 0 & -i \cdot n & 0 & 0 \\ 0 & 0 & i \cdot n & 0 \\ 0 & 0 & 0 & -i \cdot n \end{bmatrix} \tag{4.3}$$

$$\epsilon = \begin{bmatrix} -i & i & -0.865i & 0.865i \\ 1 & 1 & 1 & 1 \\ n & n & 0.500n & 0.500n \\ i \cdot n & -i \cdot n & 0.576n & -0.576n \end{bmatrix} \tag{4.4}$$

The initial conditions (θ_i and $\dot{\theta}_i$) for the first and second complex conjugate pair of eigenvectors can be found by:

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta}_1 \\ \dot{\theta}_3 \end{bmatrix} = (1+i) \begin{bmatrix} -i \\ 1 \\ n \\ i \cdot n \end{bmatrix} + (1-i) \begin{bmatrix} i \\ 1 \\ n \\ -i \cdot n \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ n \\ -n \end{bmatrix} \tag{4.5}$$

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta}_1 \\ \dot{\theta}_3 \end{bmatrix} = (1+i) \begin{bmatrix} -0.865i \\ 1 \\ 0.5n \\ 0.576i \cdot n \end{bmatrix} + (1-i) \begin{bmatrix} 0.865i \\ 1 \\ 0.5n \\ -0.576i \cdot n \end{bmatrix} = \begin{bmatrix} 1.730 \\ 2 \\ n \\ -1.156n \end{bmatrix} \tag{4.6}$$

When using the initial conditions from equation 4.5 in the `Matlab`® code in Appendix B, the response for each eigenvalue and associated eigenvectors should be purely oscillatory (since it has only imaginary components) and have a period of 1 cycle per orbit.

Figures 4.1, 4.2 and 4.3, shows the angular position, angular rate and angular acceleration of this system. The system response shows that the oscillations are purely sinusoidal with a period equal to the orbital period of 5431 seconds. Recall that the roll, pitch and yaw axes are θ_1, θ_2, and θ_3 respectively.

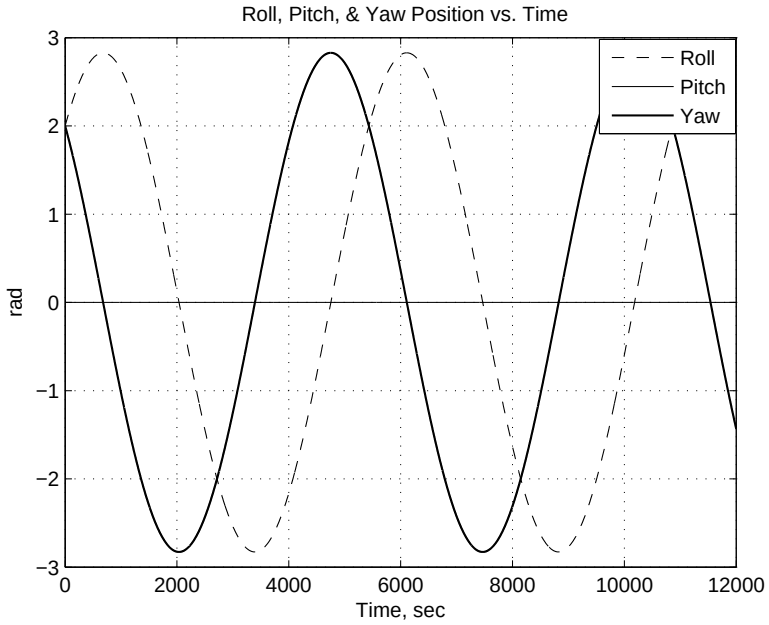


Figure 4.1: Angular Position or first pair of eigenvectors with $A = 20$,$B = 10$, and $C = 30$

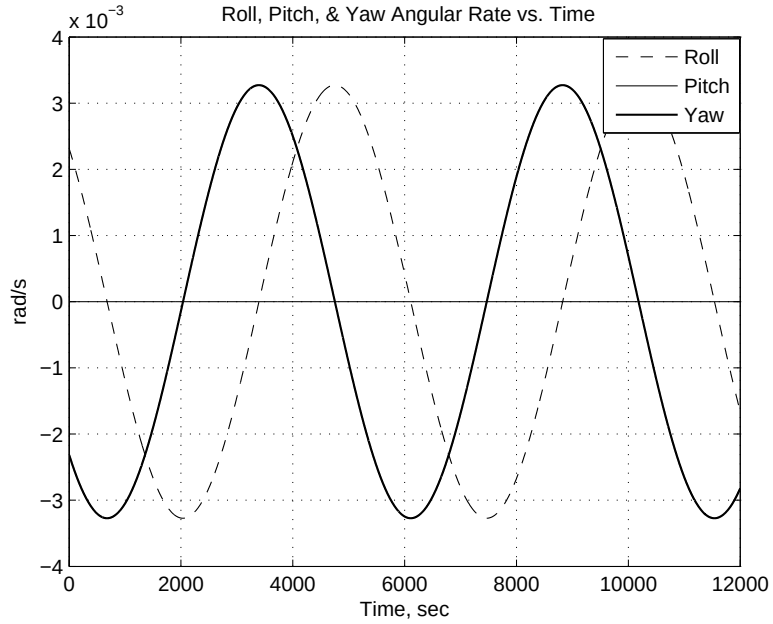


Figure 4.2: Angular Rate for first pair of eigenvectors with $A = 20$, $B = 10$, and $C = 30$

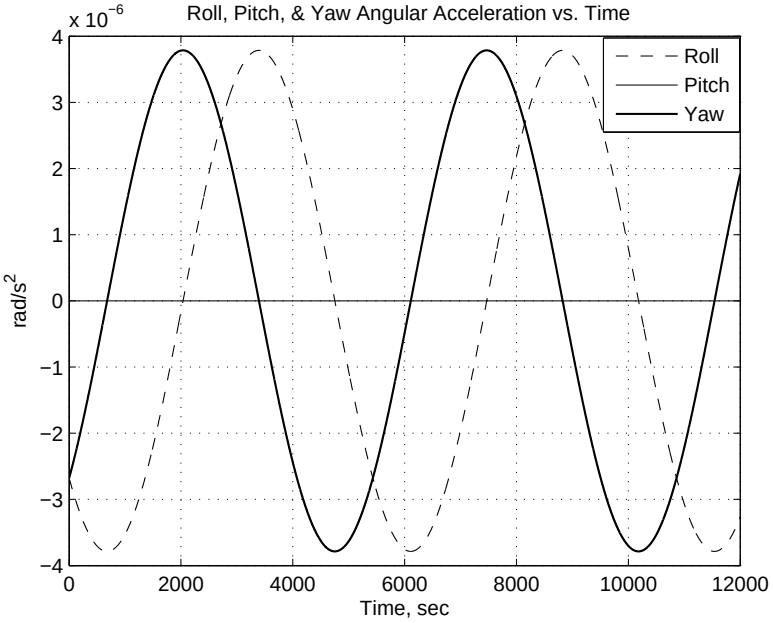


Figure 4.3: Angular Acceleration for first pair of eigenvectors with $A = 20$, $B = 10$, and $C = 30$

For the second set of complex conjugate eigenvectors, there should be 0.576 cycles per orbital period. The period of oscillations is therefore 9429 seconds which can be seen from Figures 4.4, 4.5 and 4.6.

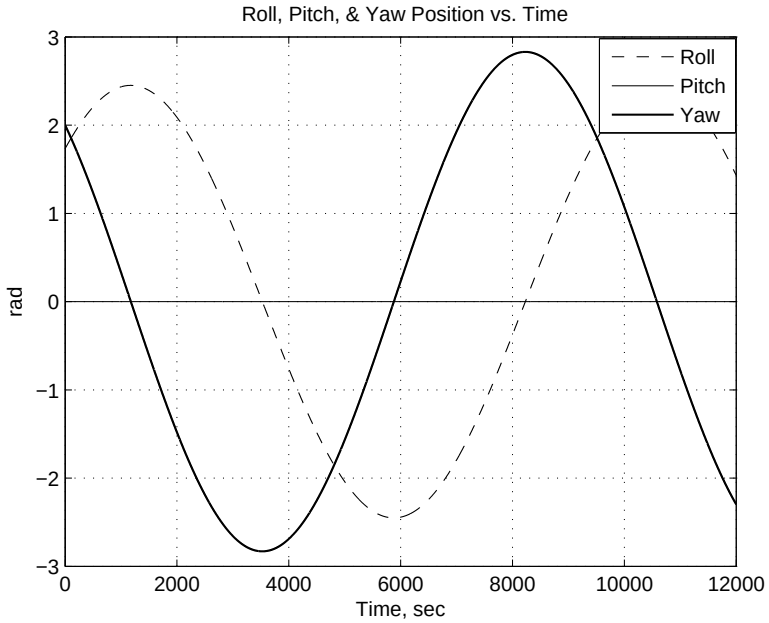


Figure 4.4: Angular Position for second pair of eigenvectors with $A = 20$, $B = 10$, and $C = 30$

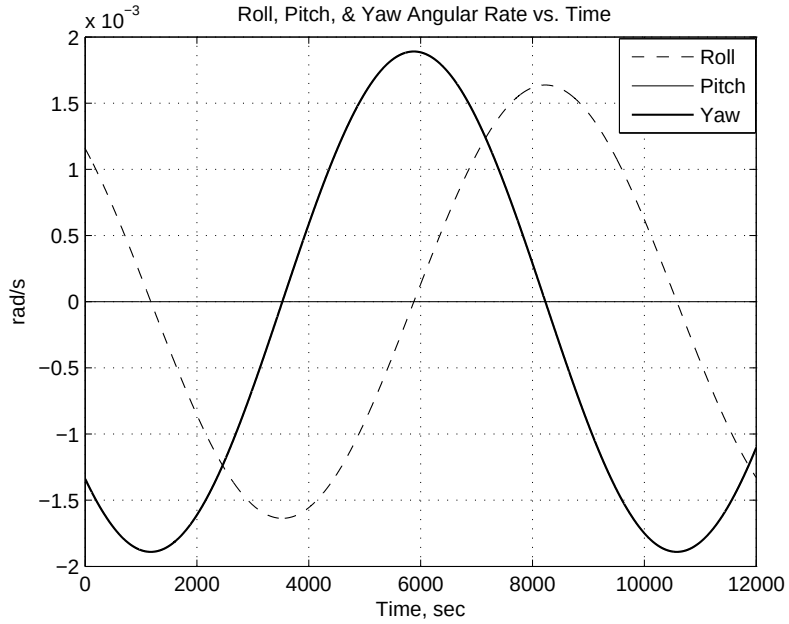


Figure 4.5: Angular Rate for second pair of eigenvectors with $A = 20$, $B = 10$, and $C = 30$

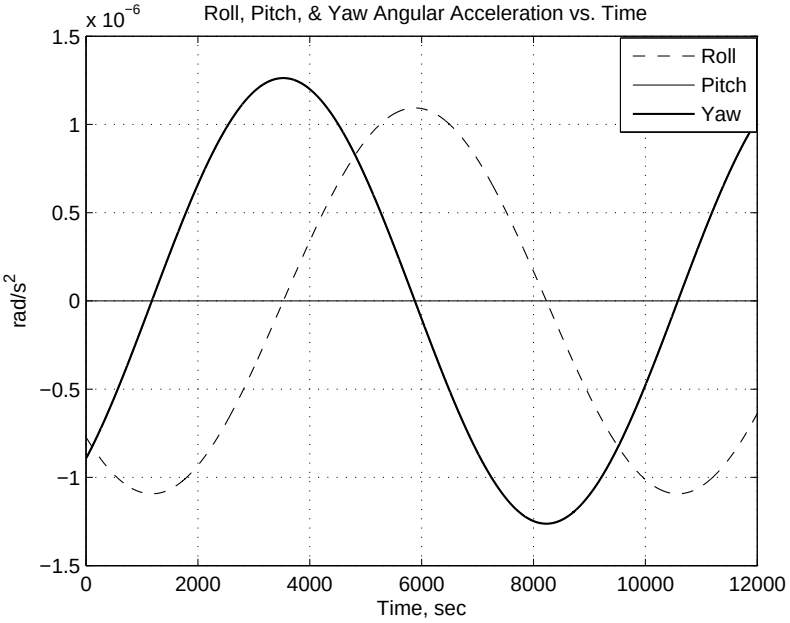


Figure 4.6: Angular Acceleration for second pair of eigenvectors with $A = 20$, $B = 10$, and $C = 30$

Another run was done with $\hat{b}_2$ as the intermediate axis, where $A = 10$, $B = 20$, and $C = 30$. In this configuration, the satellite will be unstable. This was tested to evaluate different combinations of the MOI to test different possibilities. The eigenvalues and eigenvectors for this particular MOI are:

$$\lambda = \begin{bmatrix} i \cdot n & 0 & 0 & 0 \\ 0 & -i \cdot n & 0 & 0 \\ 0 & 0 & 0.576n & 0 \\ 0 & 0 & 0 & -0.576n \end{bmatrix} \tag{4.7}$$

$$\epsilon = \begin{bmatrix} -i & i & -1.73 & 1.73 \\ 1 & 1 & 1 & 1 \\ n & n & -n & -n \\ i \cdot n & -i \cdot n & 0.576n & -0.576n \end{bmatrix} \tag{4.8}$$

Once again, the same procedure was done to obtain the initial conditions for this set of eigenvalues and eigenvectors. The initial conditions become

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_3} \end{bmatrix} = (1+i) \begin{bmatrix} -i \\ 1 \\ n \\ i \cdot n \end{bmatrix} + (1-i) \begin{bmatrix} i \\ 1 \\ n \\ -i \cdot n \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ n \\ -n \end{bmatrix} \tag{4.9}$$

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_3} \end{bmatrix} = \begin{bmatrix} -1.5 \\ 0.865 \\ -0.865n \\ 0.5n \end{bmatrix} + \begin{bmatrix} -1.5 \\ 0.865 \\ 0.865n \\ 0.5n \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ 0 \\ n \end{bmatrix} \tag{4.10}$$

The case where the initial conditions from equation 4.9 are used, the outputs for θ_1 and θ_3 in position, angular rate and angular acceleration should be sinusoidal

oscillations at one cycle per orbit due to the eigenvectors containing only imaginary components. The sinusoidal oscillations can be seen in Figures 4.7, 4.8, and 4.9

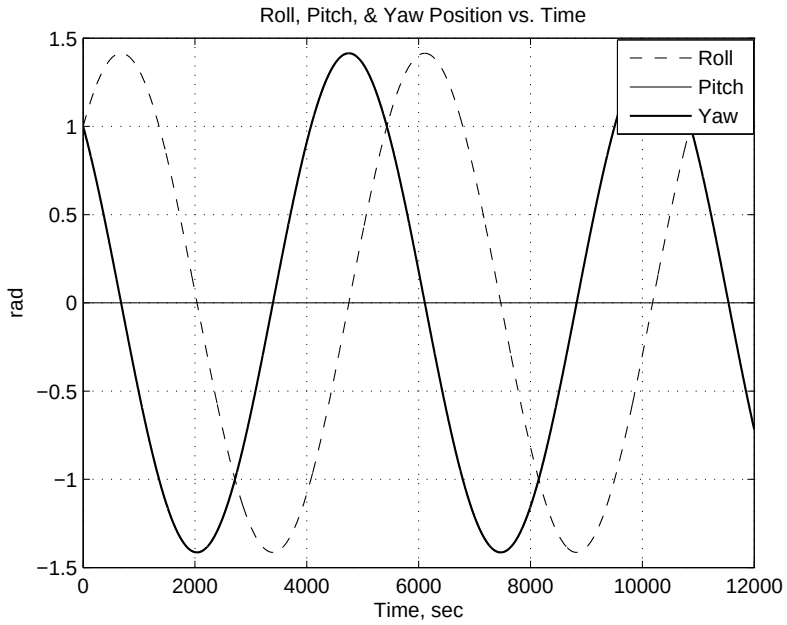


Figure 4.7: Angular Position for first pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

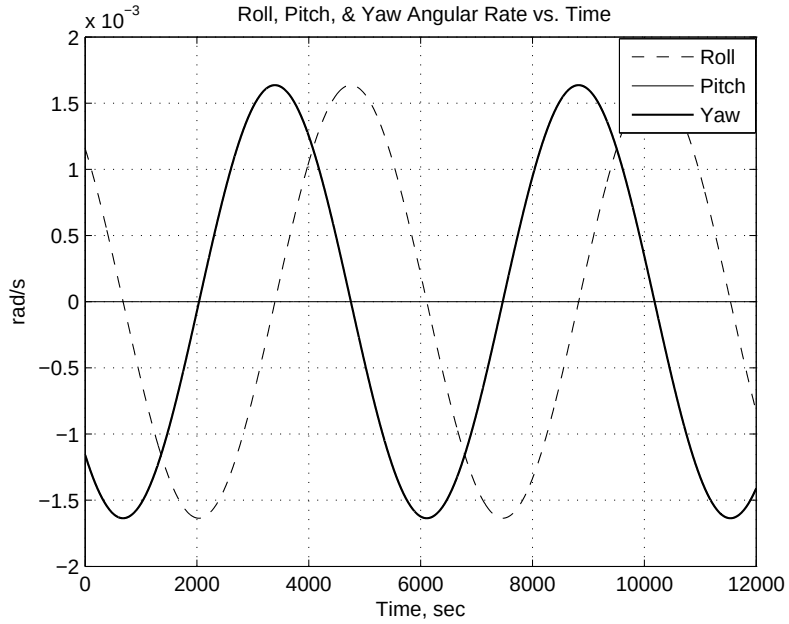


Figure 4.8: Angular Rate for first pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

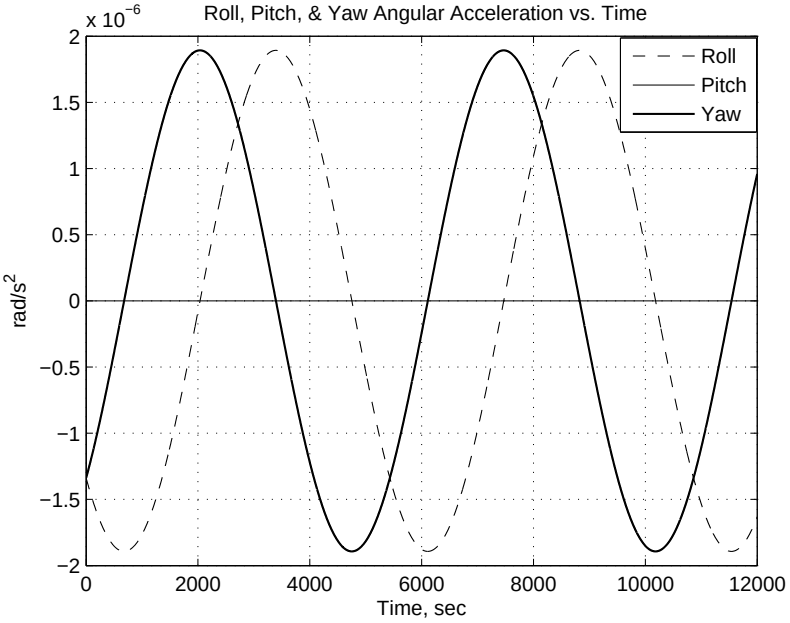


Figure 4.9: Angular Acceleration for first pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

The second set of eigenvalues and eigenvectors have no imaginary parts, therefore the θ's will grow exponentially. Exponential growth for the angles is due to the spacecraft being in an unstable configuration since the pitch axis is the intermediate MOI axis. The results of this run can be seen in Figures 4.10, 4.11, and 4.12.

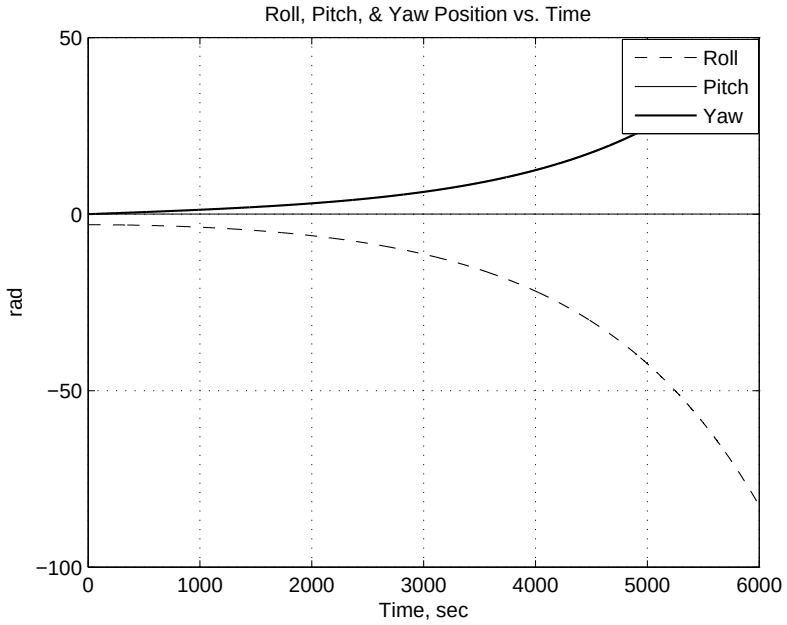


Figure 4.10: Angular Position for second pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

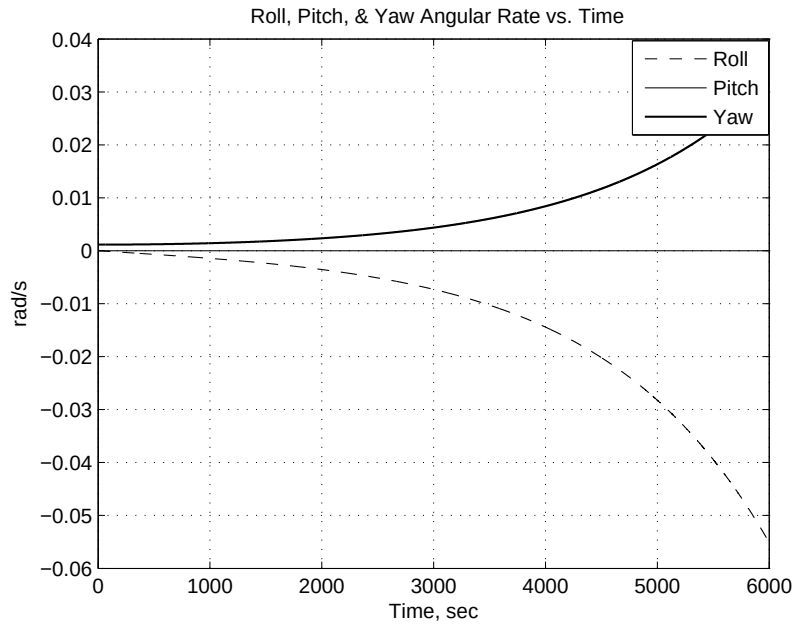


Figure 4.11: Angular Rate for second pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

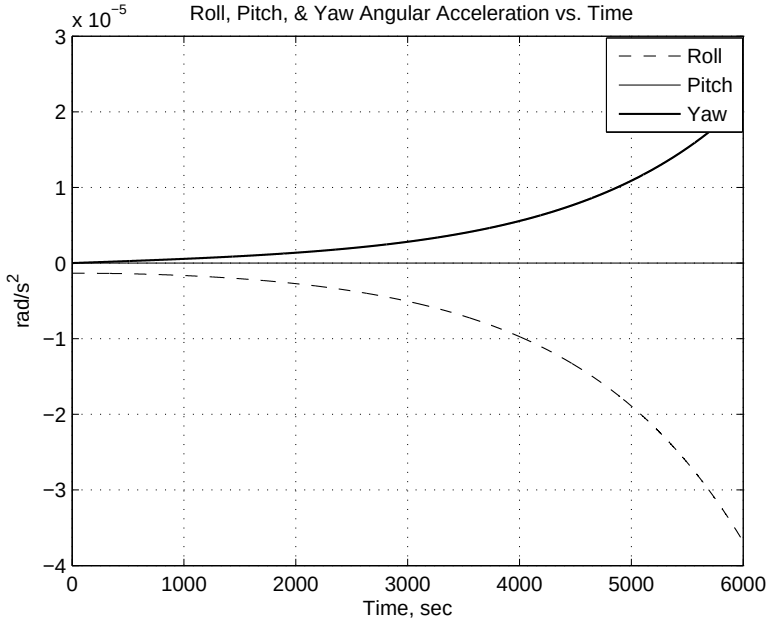


Figure 4.12: Angular Acceleration for second pair of eigenvectors with $A = 10$, $B = 20$, and $C = 30$

4.2.2 Dynamic Model Validation With Torques. The next step was to add the atmospheric density to the model and perform a similar eigenvalue/eigenvector analysis to see how the model responds. The same procedure was carried out as in the previous section. Equation 3.47 including in state space format ($\dot{X} = A \cdot x + B \cdot u$) including torques is as follows:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{C-B}{A}n^2 & 0 & 0 & 0 & 0 & \frac{C+A-B}{A}n \\ 0 & \frac{C_{\theta_2}}{B} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{(A-B)n^2+C_{\theta_3}}{C}n^2 & \frac{B-A-C}{C}n & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$
$$+ \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{C_{L_{Ail}}}{A} & \frac{C_{R_{Ail}}}{A} \\ \frac{C_{T_{Arm}}}{B} & \frac{C_{B_{Arm}}}{B} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{C_{L_{Arm}}}{C} & \frac{C_{R_{Arm}}}{C} & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \delta\phi_{TopArm} \\ \delta\phi_{BotArm} \\ \delta\phi_{LeftArm} \\ \delta\phi_{RightArm} \\ \delta\phi_{LeftAil} \\ \delta\phi_{RightAil} \end{bmatrix} \quad (4.11)$$

Once again, since θ_2 is totally decoupled from $\dot{\theta}_1$, $\dot{\theta}_3$, $\ddot{\theta}_1$ and $\ddot{\theta}_3$ all references to θ_2 were removed leaving the following A matrix to do the eigenvalue/eigenvector analysis.

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_3 \\ \ddot{\theta}_1 \\ \ddot{\theta}_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{C-B}{A}n^2 & 0 & 0 & \frac{C+A-B}{A}n \\ 0 & \frac{(A-B)n^2-4.414\rho V^2}{C}n^2 & \frac{B-A-C}{C}n & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta}_1 \\ \dot{\theta}_3 \end{bmatrix} \quad (4.12)$$

The eigenvalues and eigenvectors from Equation 4.12 were calculated using the `Matlab`® code in Appendix B. This simulation was also done at 300km altitude however, atmospheric density was $\rho = 2.4$ x 10^{-11} $\frac{kg}{m^3}$. Note, the most stable satellite configuration is with $\hat{b}_2$ as the major MOI axis, $\hat{b}_3$ as the minor axis and $\hat{b}_1$ as the intermediate axis. The following analysis was done with $A = 20$, $B = 30$, and $C = 10$.

$$\lambda = \begin{bmatrix} 0.00116 \cdot i & 0 & 0 & 0 \\ 0 & -0.00116 \cdot i & 0 & 0 \\ 0 & 0 & 0.0253 \cdot i & 0 \\ 0 & 0 & 0 & -0.0253 \cdot i \end{bmatrix} \tag{4.13}$$

$$\epsilon = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0.00116 \cdot i & -0.00116 \cdot i & 0 & 0 \\ 0 & 0 & 0.0253 \cdot i & -0.0253 \cdot i \end{bmatrix} \tag{4.14}$$

Once again, by taking each complex conjugate pair of eigenvectors, the initial conditions can be found.

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_3} \end{bmatrix} = (1-i) \begin{bmatrix} 1 \\ 0 \\ 0.00116 \cdot i \\ 0 \end{bmatrix} + (1+i) \begin{bmatrix} 1 \\ 0 \\ -0.00116 \cdot i \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0.00232 \\ 0 \end{bmatrix} \tag{4.15}$$

$$\begin{bmatrix} \theta_1 \\ \theta_3 \\ \dot{\theta_1} \\ \dot{\theta_3} \end{bmatrix} = (1-i) \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0.0253 \cdot i \end{bmatrix} + (1+i) \begin{bmatrix} 0 \\ 1 \\ 0 \\ -0.0253 \cdot i \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0.0506 \end{bmatrix} \tag{4.16}$$

Upon running the Simulink® model with the initial conditions computed from equation 4.15, the results indicate that the model is correct since the frequency of oscillation should again be one cycle per orbit. See Figures 4.13, 4.14, and 4.15

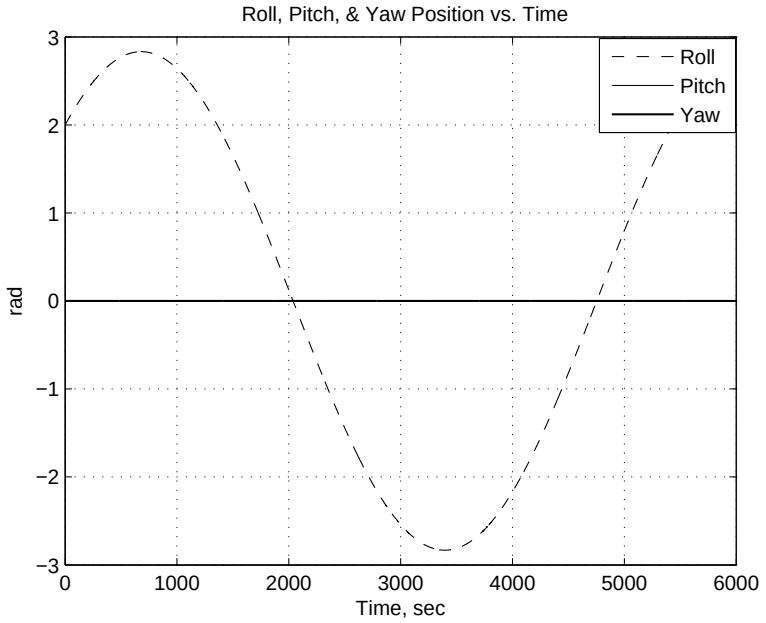


Figure 4.13: Angular Position with torque for first pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

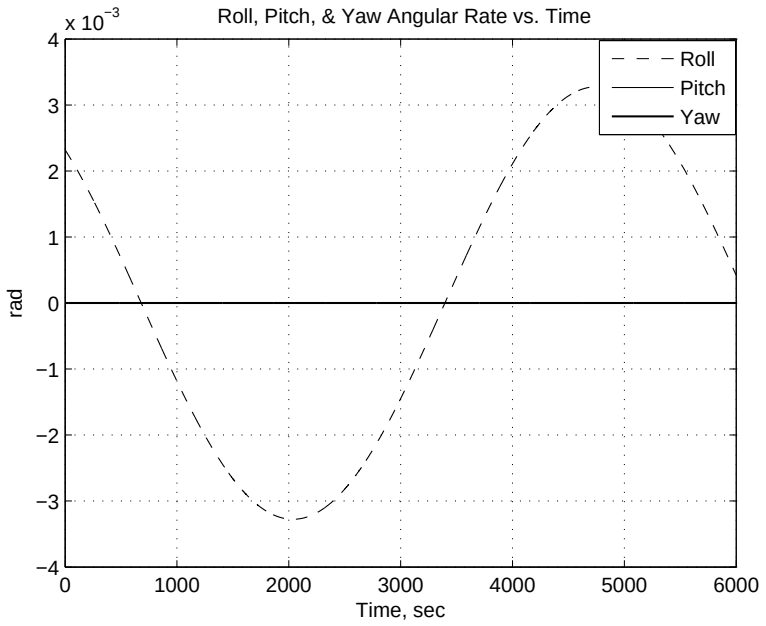


Figure 4.14: Angular Rate with torque for first pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

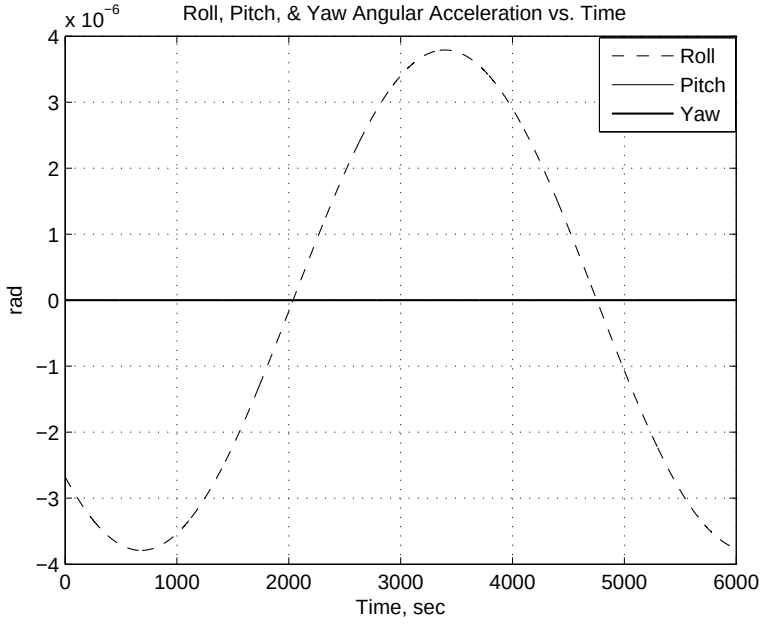


Figure 4.15: Angular Acceleration with torque for first pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

The same analysis was done on the remaining set of initial conditions from equation 4.16. Figures 4.16, 4.17, and 4.18, show that the cyclic period of oscillations is reduced (higher frequency). Since the period is approximately $315s$, the frequency can be calculated from $\omega = \frac{2\pi}{315s} = 0.0199\frac{rad}{s}$, which is close enough to the eigenvalue 0.0253 for model validation.

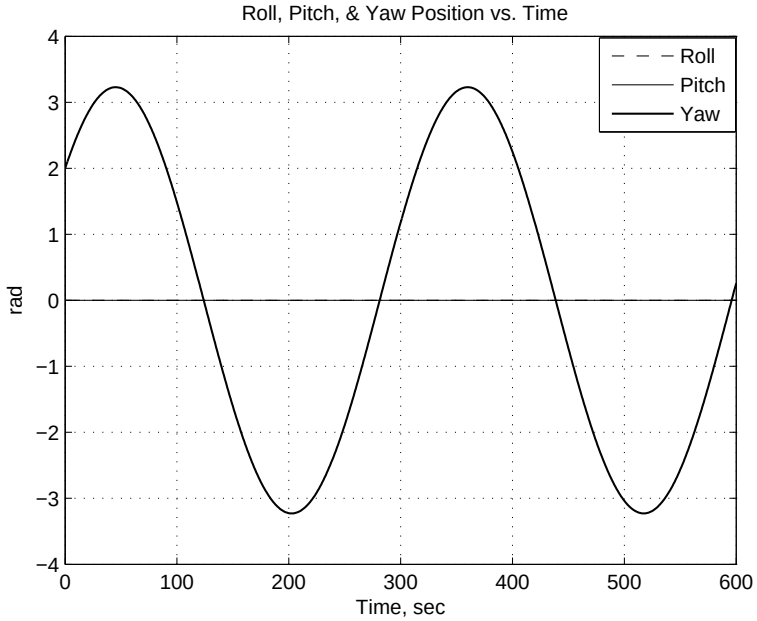


Figure 4.16: Angular Position with torque for second pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

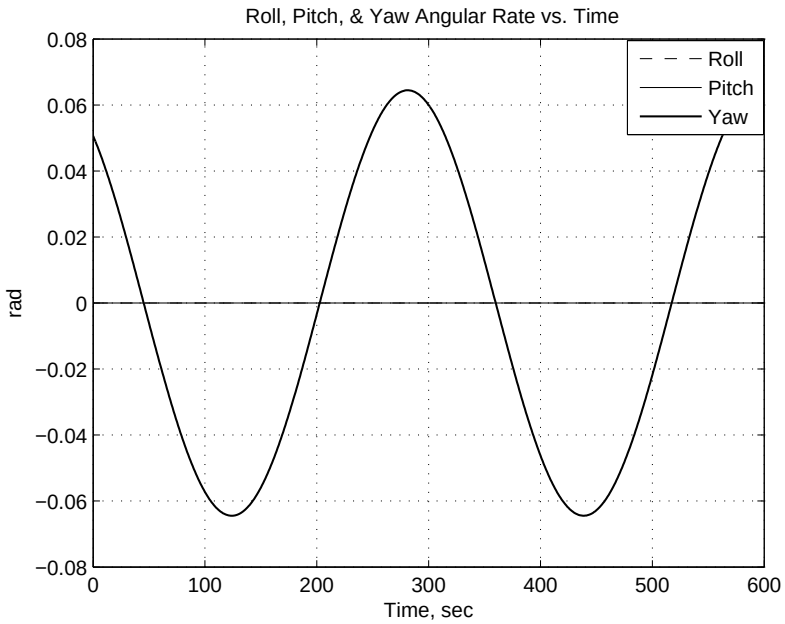


Figure 4.17: Angular Rate with torque for second pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

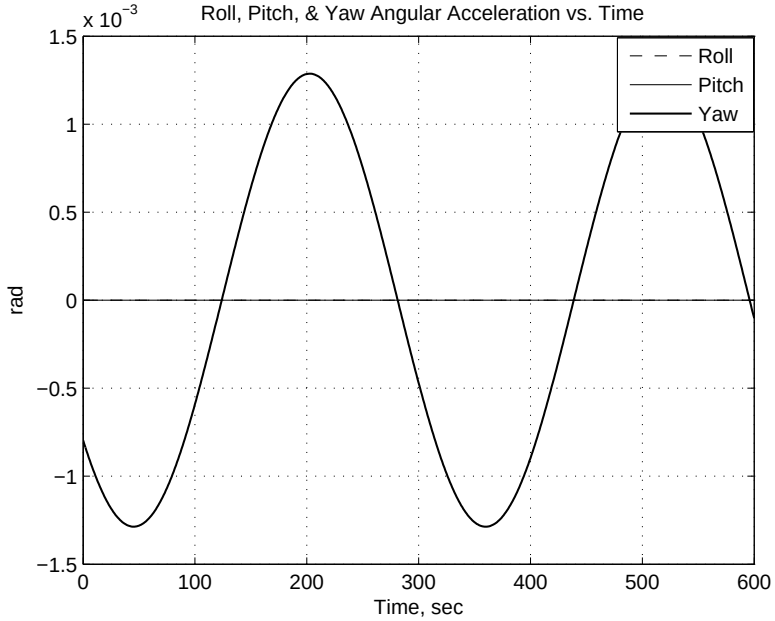


Figure 4.18: Angular Acceleration with torque for second pair of eigenvectors with $A = 20$, $B = 30$, and $C = 10$

4.3 Proportional Derivative Controller

Since the Simulink® model is linearized, the control deflections are kept reasonably small. The controller is a proportional controller with the constants selected in such a way that the control deflections remain small due to the linearized torque equations. As discussed in chapter V, the control deflections do in fact remain small thus giving a more realistic result. With the controller, the roll, pitch and yaw axes will eventually settle down in the correct satellite orientation. Originally, the controller was set up such that the control arm outputs were equal to some constant multiplying the angular rates. The roll axis seemed to take a while to return to zero degrees. The only reason the roll axis would return to zero was because it was coupled with the yaw axis and the dynamics of the system forced both the roll and yaw axis to work together to align the body frame to the orbit frame. Because of the long settling time, the controller was modified so that each control arm inputs for each axis was equal to a constant multiplying the angular rates plus another constant multiplying the angle between the body and orbit reference frames (see equation 4.17. Adding the constant times the anglular position brought the roll axis back to zero degrees in a more reasonable time frame.

$$\delta\phi = C\dot{\theta} + 0.1\theta \tag{4.17}$$

V. Results

5.1 Altitude Variations

Based on equation 3.28, drag is proportional to the square of the velocity and is proportional to the atmospheric density. A spacecraft in LEO will experience orbital decay due to drag which will lower its orbit over time. As orbital altitude is reduced, both the density and velocity increase thus having a significant increase on the drag force. Therefore, the most significant variable that affects drag is altitude. For the following subsections, the simulations were run at altitudes varying from 200km to 600km in 100km increments. At these altitudes, the density, velocity and orbital rate were calculated and are shown in Table 5.1. Table 5.2 shows all the inputs to the model for all the simulations where altitude is varied. The MOI used for this section are shown in equation 5.1 (See Appendix D.3 for details on how this was determined). Table 5.2 shows the inputs to the Simulink® model. Note, only a few plots were included in this chapter, however, all the plots have been included in Appendix E.

Table 5.1: Density, velocity and orbital rate calculated from Simulink®.

Altitude (km)	Calculated Density ($\frac{kg}{m^3}$)	Velocity ($\frac{m}{s}$)	Orbital Rate ($\frac{rad}{s}$)
200	$2.79\text{x}10^{-10}$	7784	0.001183
300	$2.42\text{x}10^{-11}$	7726	0.001157
400	$3.73\text{x}10^{-12}$	7669	0.001131
500	$6.97\text{x}10^{-13}$	7613	0.001107
600	$1.45\text{x}10^{-13}$	7558	0.001083

$$MOI = \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} = \begin{bmatrix} 92 & 0 & 0 \\ 0 & 107 & 0 \\ 0 & 0 & 68 \end{bmatrix} kg \cdot m^2 \tag{5.1}$$

The proportional derivative controller constants multiplying the angular rates are 50 for roll, 20 for pitch and 20 for yaw. The constants multiplying the angle between the

Table 5.2: Variables input into the Simulink® model for all the altitude variation simulations.

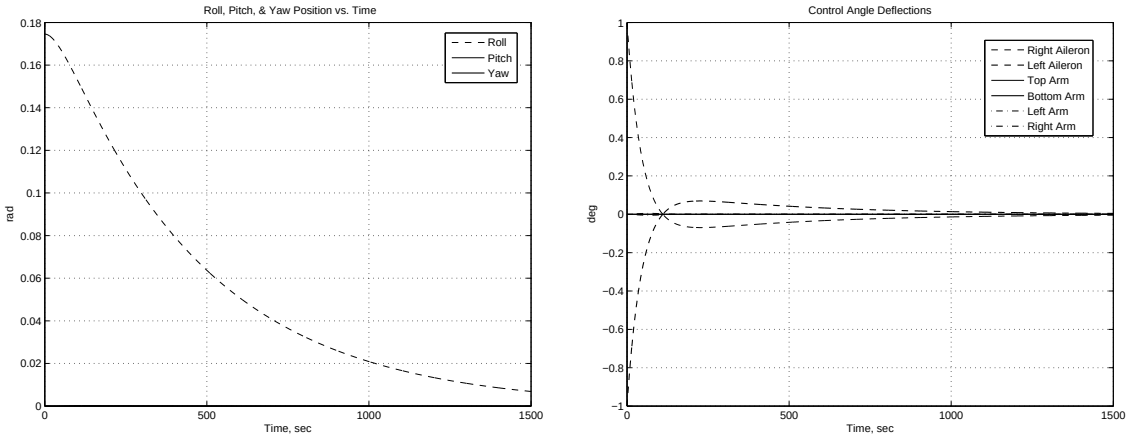
Variable	Value
L_{Panel}	$1m$
W_{Panel}	$1m$
L_{Arm}	$1m$
$\phi 0_{Arm}$	$45\circ$
L_{Ail}	$1m$
W_{Ail}	$1m$
L_{b_1}	$1m$
L_{b_2}	$1m$
L_{b_3}	$1m$
ϕ_{Wedge}	22.5°
$Mass$	$500kg$
C_D	2.2

body and orbit frames are 0.1. For the initial conditions, the angular displacement of the satellite body with respect to the orbital frame are chosen to be offset 10° and the angular rates set to $0\frac{rad}{s}$. The linearization angle for the control arms was chosen to be 45°. Several factors were considered in selecting the angle. First, when the panels are extended, only the windward facing sides are subject to being hit by molecules. Since small angles are assumed, the back side of the panels will not be exposed to the incoming molecules. The small angle assumption between the orbit and body frames simplified the linearization since the heaviside function was no longer needed. Another reason is that at smaller linearization angles, the wedge shaped ailerons would shadow the left and right panels used for yaw control. The ailerons for the nonlinear case would not have this problem since the ailerons would consist of flat panels, not wedges.

5.1.1 Simulation at 200 km Altitude. For this simulation, the inputs to the modeled are represented in Table 5.2 with an altitude of $200km$. The density, orbital velocity and orbital rate for this altitude are summarized in Table 5.1. The first simulation only had the roll axis displaced, the 2nd was pitch and the 3rd was

the yaw axis. Then a fourth simulation was run where all three axes of roll pitch and yaw were displaced simultaneously. By comparing Figures 5.1, and 5.4, it can be seen that roll and yaw are coupled and do have some affect on each other. When both yaw and roll are offset simultaneously as shown in Figure 5.6, the coupling is more pronounced.

5.1.1.1 200km Altitude, Roll Offset by $10°$. The first simulation was done where roll was offset by $10°$. To compare the settling times for the various altitudes, the time constant must be found. The time constant τ is the time it takes for the amplitude to decay by $\frac{Amp_0}{e}$, where, Amp_0 is the initial amplitude or initial θ_i and $e \approx 2.71$. The time constant can be approximated directly from the plots once the initial amplitude is known. For the simulations, the initial conditions determine Amp_0, and thus $Amp_0 = 10°$ or $0.175rad$. The amplitude for one time constant is $\frac{Amp_0}{e} = 0.064$.

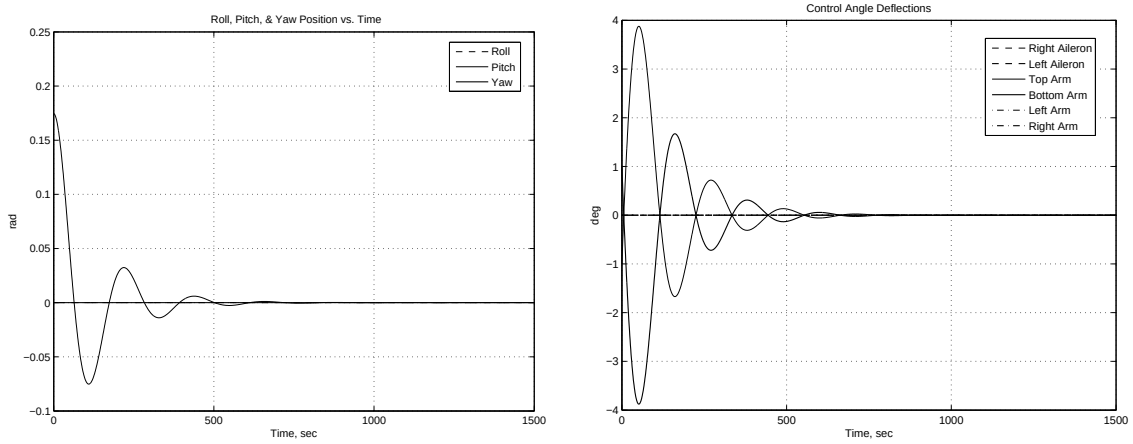


(a) (b)

Figure 5.1: (a) Angular displacement with roll offset $10°$.
(b) Control angles with roll $10°$.

Since the roll doesn't oscillate, the time constant can be read from the plot where $\tau \approx 500s$

5.1.1.2 200km Altitude, Pitch Offset by 10°. The pitch is decoupled from the roll and yaw axes therefore the only oscillations should be in the pitch axis. Figure 5.2(a) shows this to be true.



(a) (b)

Figure 5.2: (a) Angular displacement with pitch offset 10°.
(b) Control angles with pitch 10°.

The next step was to pick points on the plot, and fit them to a curve. The time constant for the best fit can then be determined. Since the pitch axis oscillates, the locations of the peaks were used for the curve fit algorithm in Appendix B. Assuming an exponential decay, the plot should fit the following equation:

$$Y = [exp(\frac{-T}{\tau})] * a \tag{5.2}$$

The plot shown in Figure 5.3 closely approximates the data points and the time constant is shown to be $\approx 130s$

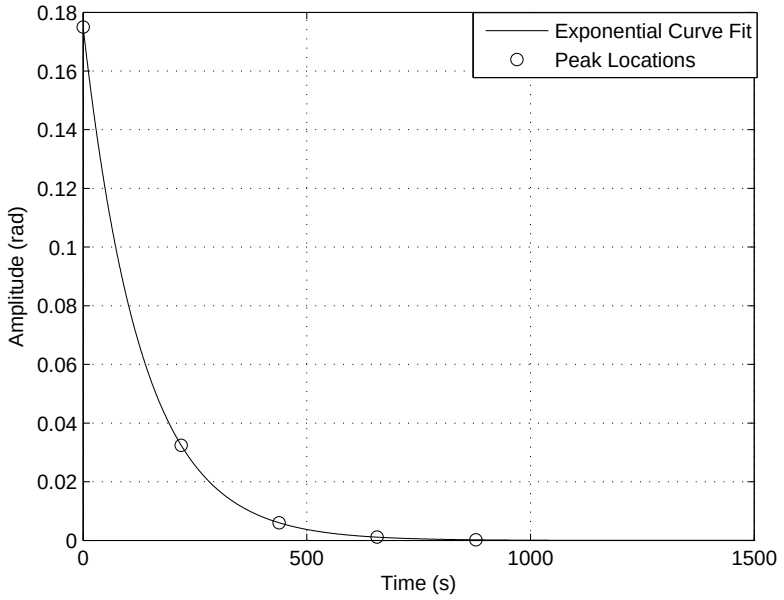
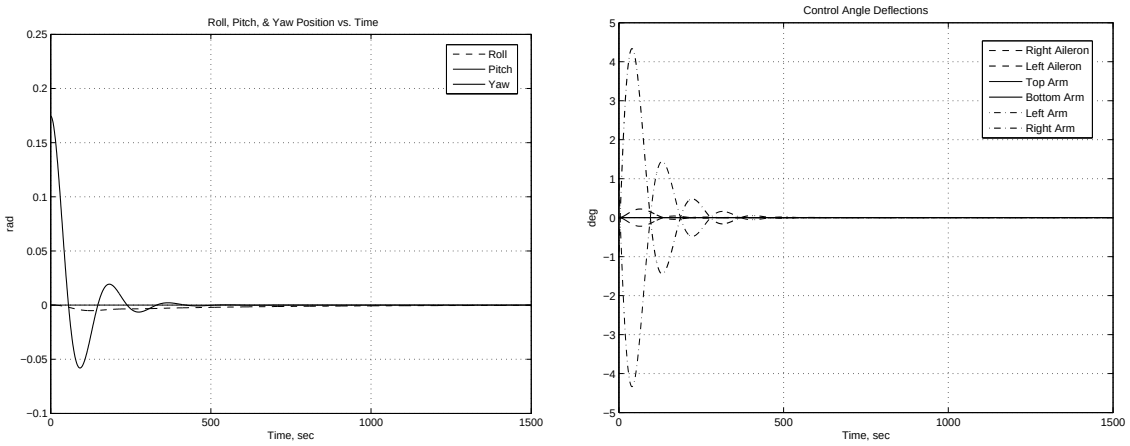


Figure 5.3: Curve fit for 200km altitude with pitch offset 10°

5.1.1.3 200km Altitude, Yaw Offset by 10°. The same procedure performed in the previous section was applied to the simulation where the yaw was offset by 10°. The results are shown in Figure 5.4 (a) and 5.5. Here, the time constant is $\approx 85s$.



(a) (b)

Figure 5.4: (a) Angular displacement with yaw offset 10°.
(b) Control angles with yaw 10°.

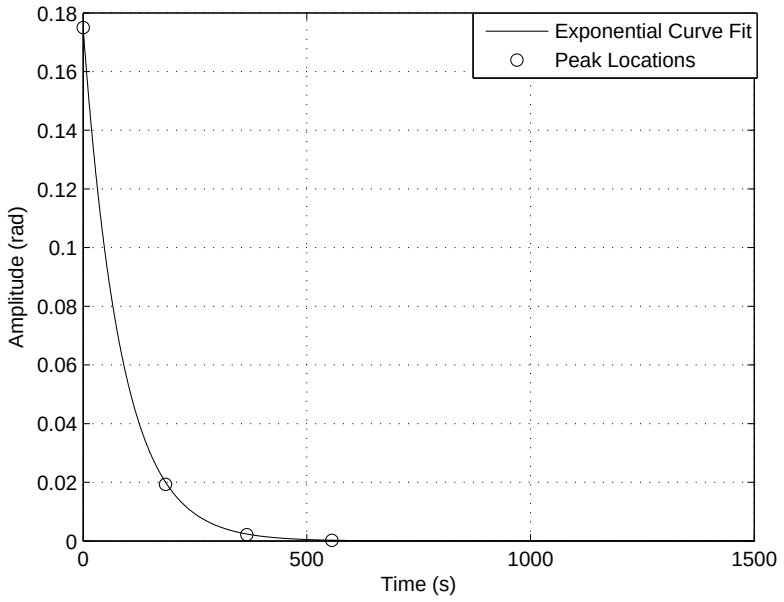
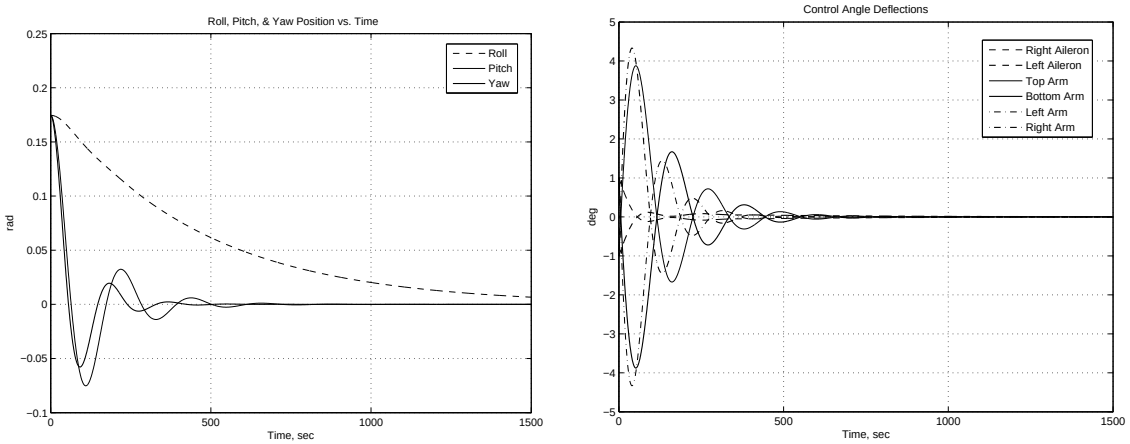


Figure 5.5: Curve fit for 200km altitude with yaw offset 10°

5.1.1.4 200km Altitude, Roll, Pitch and Yaw Offset by 10°. For this simulation, all 3 axes were offset simultaneously which would be a more realistic scenario. The reason for running them separately at first was to see how each axis compared to the more realistic simulation where all three were off from the orbital frame. Notice on all simulations, the control deflections are relatively small, $\delta\phi_i < 5°$. It is important that these angles remain small since this is a linearized model. At higher altitudes the maximum control angle deflections actually decrease since the controller is a proportional controller. At higher altitudes with the same initial conditions, the rotational rate decreases due to a smaller force imparted on the control panels.



(a) (b)

Figure 5.6: (a) Angular displacement with roll, pitch and yaw offset 10°.
(b) Control angles with roll, pitch and yaw offset 10°.

5.1.2 Altitudes Above 200km. Since we are interested in determining how altitude affects the settling time, the same procedure was done for altitudes of 300km, 400km, 500km and 600km. See Appendix E for the plots not shown in this chapter. For each simulation, the time constants were found and are summarized in Tables 5.3 and 5.4. Table 5.3 is the case where each axis was deflected separately and Table 5.4 is the case where all three axes were deflected simultaneously. In either case,

there isn't much difference between both tables. By using the same exponential curve fit algorithm as done in section 5.1.1.2, an exponential curve was fit to the time constants in Table 5.4 for the roll, pitch and yaw axes. Figures 5.7, 5.8 and 5.9 show how the time constants grow exponentially with altitude.

Table 5.3: Time constants for each axis at the various altitudes where roll, pitch and yaw were offset separately.

Altitude (km)	Time Constant (s)		
	Roll Axis	Pitch Axis	Yaw Axis
200	500	130	85
300	975	1550	950
400	6600	10200	6200
500	40000	55000	30000
600	275000	265000	260000

Table 5.4: Time constants for each axis at the various altitudes where roll, pitch and yaw were deflected simultaneously.

Altitude (km)	Time Constant (s)		
	Roll Axis	Pitch Axis	Yaw Axis
200	481	130	85
300	960	1550	950
400	6800	10200	6000
500	40000	55000	30000
600	265000	265000	260000

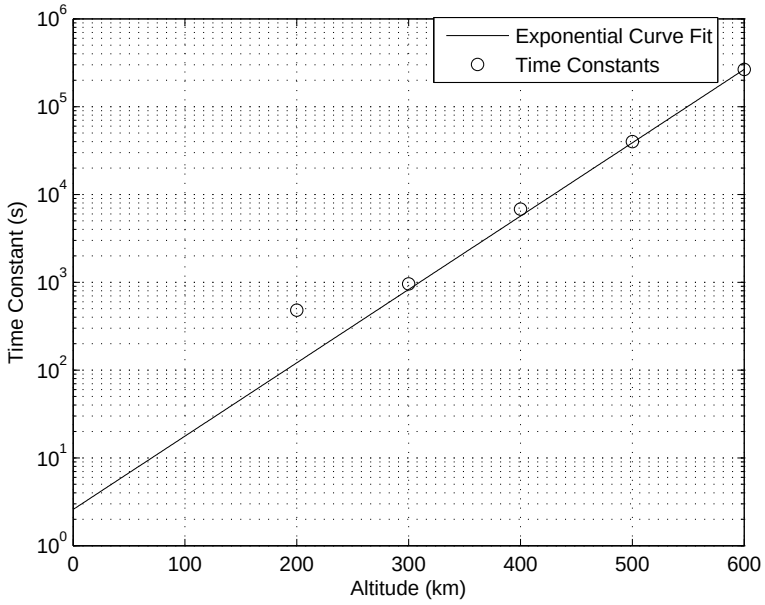


Figure 5.7: Time constants for the roll axis increase exponentially with increasing altitude.

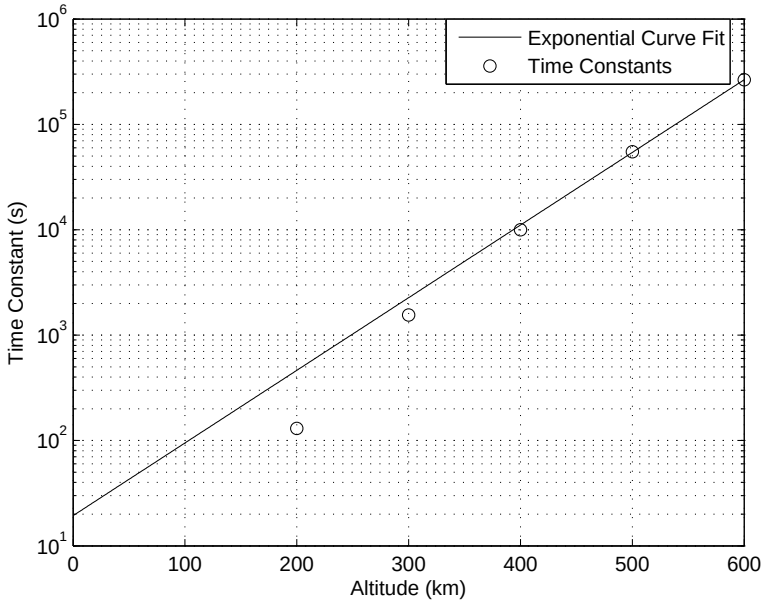


Figure 5.8: Time constants for the pitch axis increase exponentially with increasing altitude.

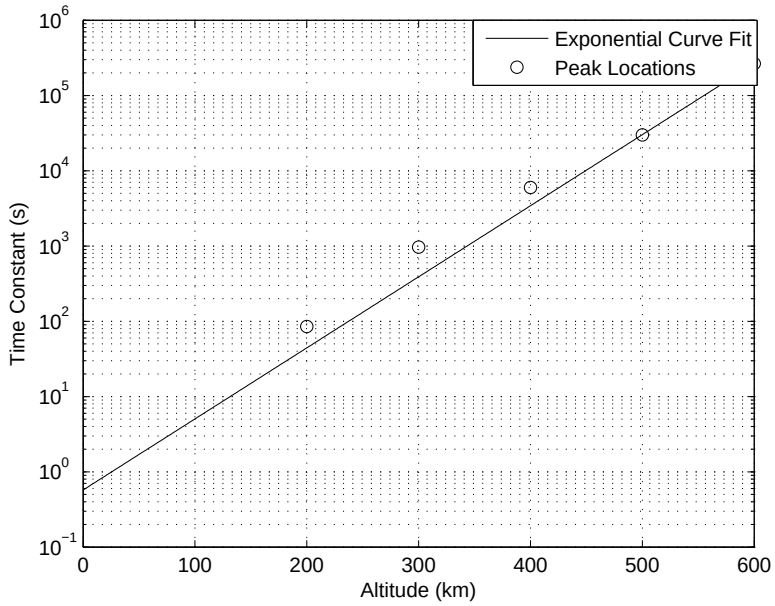


Figure 5.9: Time constants for the yaw axis increase exponentially with increasing altitude.

5.2 Pitch as the Intermediate Moment of Inertia Axis

The satellite would be in its most stable configuration with the pitch axis as the major or minor MOI axis. That may not always be the case, so the pitch axis was set as the intermediate axis which would make the attitude unstable (see equation 5.3 for the MOI matrix). For this simulation, theta was offset by 1 degree for each axis, the altitude was set to 400km and density was set to zero. This shows the satellite's attitude is unstable with pitch as the intermediate axis as can be seen in Figure 5.10.

$$MOI = \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} = \begin{bmatrix} 107 & 0 & 0 \\ 0 & 92 & 0 \\ 0 & 0 & 68 \end{bmatrix} kg \cdot m^2 \tag{5.3}$$

When the atmospheric density is put back into the simulation, the controller is able to return the satellite back to flying at its intended attitude. See figure 5.11

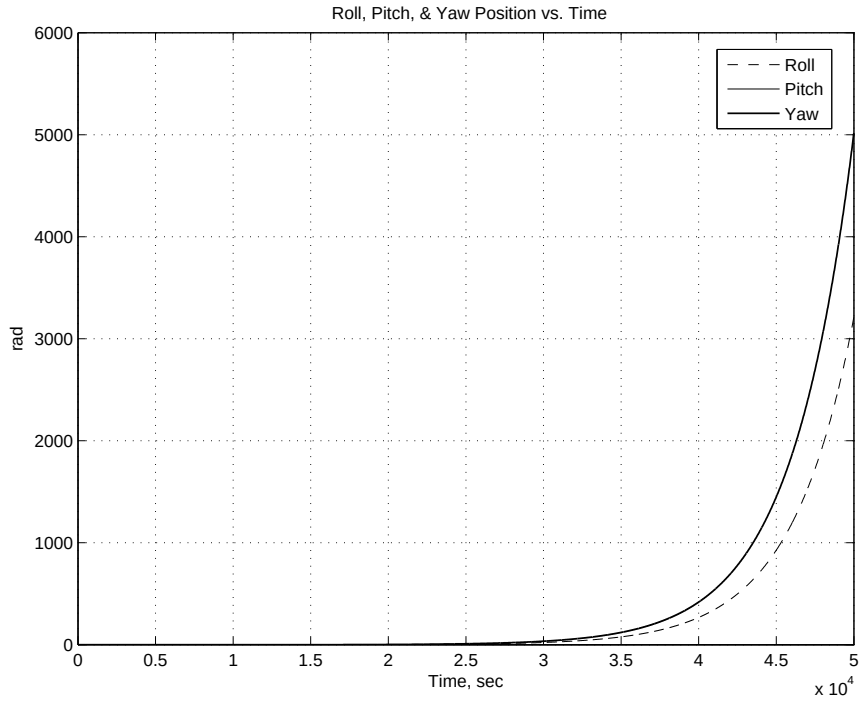


Figure 5.10: Pitch as the intermediate MOI axis and atmospheric density $\rho = 0$.

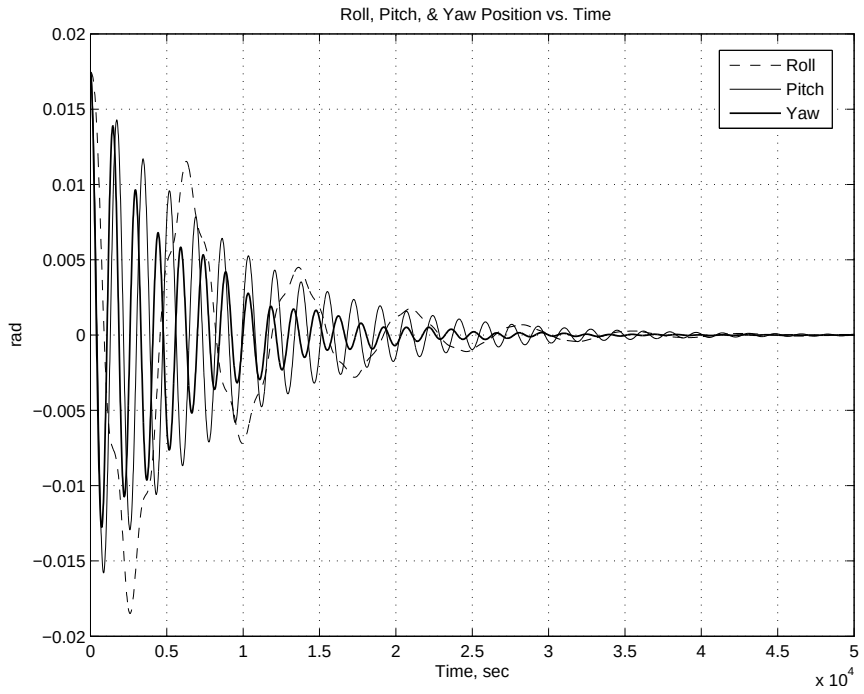


Figure 5.11: Pitch as the intermediate MOI axis and atmospheric density shown in Table 5.1.

5.3 Roll, Pitch, and Yaw Having the Same MOI

Since the pitch and yaw axes have the same proportional constant for the controller and are identical in dimensions, they should have the same time constant. This is not the case as can be seen in Figure 5.12, where the amplitude of the yaw oscillations end up being smaller than the pitch axis. This is due to the roll and yaw axis being coupled. The yaw and roll axes work work together thus having more control authority. Therefore the yaw axis decreases in amplitude at a faster rate than the pitch axis.

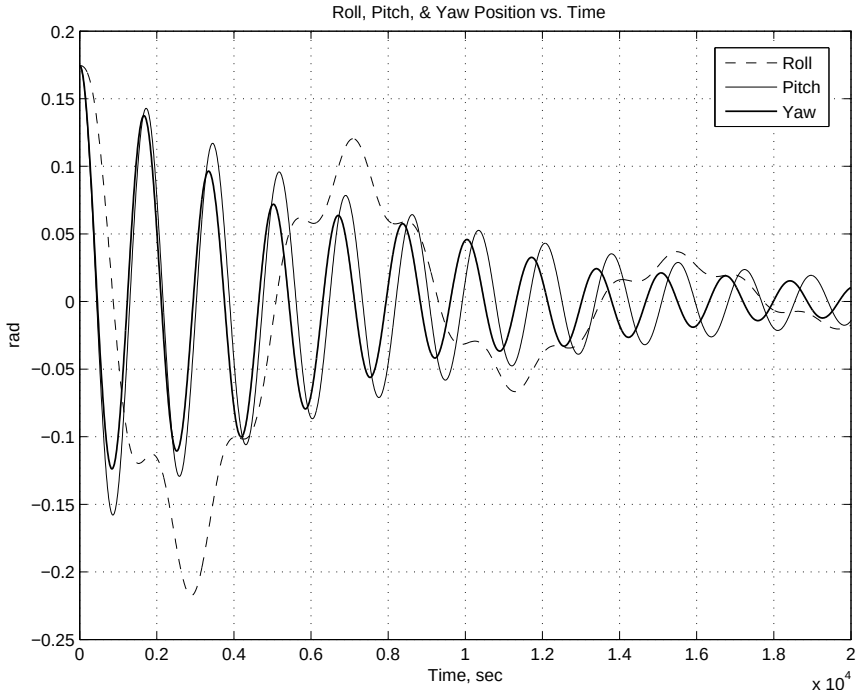


Figure 5.12: Angular position and all three axes having the same MOI.

VI. Conclusions and Recommendations

6.1 Conclusions

To be able to launch a satellite to orbit, and have the spacecraft orient itself automatically with a very simple attitude control system is very desirable and has many advantages. A simple control system would be less costly and less prone to failure than complex systems such as control moment gyros, reaction wheels or thrusters. Since most satellites have Earth sensing missions, they are typically nadir pointing and require some form of attitude control. These satellites typically have their body reference frame aligned with the orbit reference frame and are usually in LEO. Satellites are put in LEO for various reasons such as payload limitations and cost issues and have to compensate for atmospheric effects. Using the atmosphere to control a satellite's attitude was investigated and the results presented here show that it is a feasible alternative to the more expensive and complex methods mentioned earlier for satellites in LEO.

The atmospheric density decreases exponentially with altitude and therefore an atmospheric drag type control system has an altitude limitation where it no longer has enough control authority to overcome disturbance torques. The effects of increasing altitude can be seen by comparing the settling time at different altitudes. When a satellite's attitude is perturbed from its alignment with the orbital reference frame, it takes time for the attitude control system to reduce the amplitude of oscillations and bring it back to its proper attitude. At altitudes of 200 km, the settling time is on the order of minutes as compared to a 600km altitude where the settling time is on the order of weeks. Settling times are also affected by the satellites physical dimensions and mass. An increase in drag panel area, lengthening the control arms or decreasing mass will reduce settling time, however, the settling times would still be approximately the same order of magnitude. The linear analysis performed here has a small control angle limitation. A nonlinear analysis would

allow the control panels to be fully extended so the projected areas were maximized which will also reduce settling time.

According to the simulation results, the oscillations decay exponentially but will never quite decay to zero. In reality, there is a limit to the accuracy of the attitude sensors and how finely the drag panel angles can be controlled, therefore, it can be expected that the satellite will continue to oscillate about the orbital reference frame at plus or minus some small angle. Depending on the pointing requirements of the payload, these oscillations may not be acceptable. If so, a stabilized payload could be added to the spacecraft to counter these oscillations.

The bottom line is that atmospheric drag can effectively be used to control a spacecraft attitude in LEO. Disturbances in attitude can be overcome resulting in a stable attitude. The spacecraft's control system would consist of a less costly, simple control system with simple actuators less prone to failure. The research presented here is the foundation for future study which can eventually lead to an actual flight test of a drag controlled satellite.

6.2 Recommendations for Future Research

The research presented here can be very useful in several areas of continued research. Future research should be built upon eventually leading to the flight test of an actual aerodynamically controlled satellite. Below are several ideas to build upon this research.

6.2.1 Higher Fidelity Nonlinear Computer Model. The next step with this topic would be to add estimation algorithms into the model and eventually do a nonlinear analysis. A nonlinear computer model would better represent the dynamics of an actual aerodynamically controlled satellite system. The final step would be to set the spacecraft in a tumble with the nonlinear model and see if it can recover. Other factors should be added to the model such as a rotating

atmosphere, elliptical orbits, non-equatorial orbits, drag panels with mass, random external torques from solar winds, oblate Earth, and an orbit decay algorithm to determine the approximate orbit lifetime for a satellite of a given mass, projected area and altitude. This way, the optimum orbit could be selected based on lifetime and payload capabilities/requirements. From this point, satellite dimensions could be determined for the most effective control. The altitudes where the control authority is not enough to overcome perturbations can also be determined. Note, orbit decay calculations are only approximate since there are so many random variables that affect the atmospheric density.

6.2.2 Solar Drag Panels. Since satellites usually have solar panels to generate electrical power, the drag panels should be replaced with solar panels. In this research, the mass of the panels was neglected however, a solar panel mass should be included in the model. Including the mass of the panels in the model would also require some vibrational analysis to determine potential problems with oscillations of the panels. As the panels are being moved, toques will have an effect on the satellites attitude and should be accounted for. Finally, research the feasibility of tracking the sun with one or more panels while maintaining attitude.

6.2.3 Other Related Research. Some students have done research on maintaining satellite formations. This model could be used with their research to control both the satellites attitude and station keeping at the same time. Other designs using atmospheric drag for attitude control should be considered and compared with this and other research to determine the best and most cost effective design. Finally, research should be done to determine if drag panels could be used for momentum dumping of reaction wheels.

Appendix A. MathCad Worksheets

A.1 MathCad Nonlinear Torque Calculation Worksheet

Drag Torques on Sat due to control panels (NON LINEAR CASE):

$$CG := \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \cdot m \qquad \theta_1 := 0 \cdot deg \qquad L_{arm} := 1 \cdot m$$

$$\theta_2 := 0 \cdot deg \qquad L_{side} := 1 \cdot m$$

$$\theta_3 := 0 \cdot deg \qquad L_{panel} := 1 \cdot m$$

Angles of arms

$\phi_{TopArm} := 45 \cdot deg$ 0 to 90

$\phi_{BotArm} := 46 deg$ 0 to 90

$\phi_{LeftArm} := 45 \cdot deg$ 0 to 90

$\phi_{RightArm} := 45 \cdot deg$ 0 to 90

$\phi_{RightAil} := 0 \cdot deg$

$\phi_{LeftAil} := -\phi_{RightAil}$

$\phi_{Wedge} := 22.5 \cdot deg$ Wedge angle

$r_{earth} := 6370000 \cdot m$ $altitude := 300000 \cdot m$ $C_D := 2$

$r_{orbit} := r_{earth} + altitude$

$\mu := 3.98600 \cdot 10^{14} \cdot \frac{m^3}{s^2}$ Standard Gravitational Parameter

$V := \sqrt{\frac{\mu}{r_{orbit}}}$ Circular Orbital Velocity

$V = 7.73 \times 10^3 \frac{m}{s}$

$\rho := 10^{-13} \cdot \frac{kg}{m^3}$ ApproxAtmospheric Density at 300 km altitude

$v_{hat} := \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$ Velocity Unit Vector in the A frame

The orientation of the Top WRT the A frame is given by:

$$\begin{pmatrix} b1 \\ b2 \\ b3 \end{pmatrix} = R_{ba} \cdot \begin{pmatrix} a1 \\ a2 \\ a3 \end{pmatrix} \quad \text{Where} \quad R_{BA} = \begin{pmatrix} b1 \\ b2 \\ b3 \end{pmatrix} \cdot (a1 \;\; a2 \;\; a3) \rightarrow R_{BA} = \begin{pmatrix} b1 \cdot a1 & b1 \cdot a2 & b1 \cdot a3 \\ b2 \cdot a1 & b2 \cdot a2 & b2 \cdot a3 \\ b3 \cdot a1 & b3 \cdot a2 & b3 \cdot a3 \end{pmatrix}$$

THIS FILE CONTAINES THE NON-LINEAR TORQUE EQUATIONS FOR THE NEW SAT DESIGN WITH **WEDGIES FOR AILERONS**. THIS ALSO INCLUDES ALL DIMENSIONS OF SAT IN VARIABLES NOTE THE 4 CONTROL PANELS ARE ONLY USED FOR PITCH AND YAW.

$L_{Ail} := 1 \cdot m$ Length (span b2 direction) of each aileron

$W_{Ail} := 1 \cdot m$ Width (chord b1 direction) of each aileron

$A_{Ail} := L_{Ail} \cdot W_{Ail}$ Area for Ailerons

$L_{panel} := 1 \cdot m$ Length of each panel b1 front to back

$W_{panel} := 1 \cdot m$ Width of each panel

$A_{pan} := L_{panel} \cdot W_{panel}$ Area for Control panels

$L_{b1} := 1 \cdot m$ Body length in b1 direction

$L_{b2} := 1 \cdot m$ Body length in b1 direction

$L_{b3} := 1 \cdot m$ Body length in b1 direction

$L_{arm} := 2 \cdot m$ Length of control arm

First find Rba using the definition and by using 3-2-1 Euler Angle rotations

$$R_{BA} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_1) & \sin(\theta_1) \\ 0 & -\sin(\theta_1) & \cos(\theta_1) \end{pmatrix} \cdot \begin{pmatrix} \cos(\theta_2) & 0 & -\sin(\theta_2) \\ 0 & 1 & 0 \\ \sin(\theta_2) & 0 & \cos(\theta_2) \end{pmatrix} \cdot \begin{pmatrix} \cos(\theta_3) & \sin(\theta_3) & 0 \\ -\sin(\theta_3) & \cos(\theta_3) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{BA} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The orientation of the top flap F_t frame WRT the body frame is given by

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} = R_{FB} \cdot \begin{pmatrix} b1 \\ b2 \\ b3 \end{pmatrix}$$

Rotation from the B frame to the F_{top} frame

Using a 3-2-1 Euler Angle sequence, the rotation matrix from B to F is:

$$R_{FB} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi_1) & \sin(\phi_1) \\ 0 & -\sin(\phi_1) & \cos(\phi_1) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_2) & 0 & -\sin(\phi_2) \\ 0 & 1 & 0 \\ \sin(\phi_2) & 0 & \cos(\phi_2) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_3) & \sin(\phi_3) & 0 \\ -\sin(\phi_3) & \cos(\phi_3) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Need rotations for each flap (top, bot, left, right)

$$R_{FA} = \begin{pmatrix} f1 \\ f2 \\ f3 \end{pmatrix} \cdot (a1 \quad a2 \quad a3) \rightarrow R_{FA} = \begin{pmatrix} f1 \cdot a1 & f1 \cdot a2 & f1 \cdot a3 \\ f2 \cdot a1 & f2 \cdot a2 & f2 \cdot a3 \\ f3 \cdot a1 & f3 \cdot a2 & f3 \cdot a3 \end{pmatrix}$$

$$R_{FB.top} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(-\phi_{TopArm}) & 0 & -\sin(-\phi_{TopArm}) \\ 0 & 1 & 0 \\ \sin(-\phi_{TopArm}) & 0 & \cos(-\phi_{TopArm}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{FB.bot} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{BotArm}) & 0 & -\sin(\phi_{BotArm}) \\ 0 & 1 & 0 \\ \sin(\phi_{BotArm}) & 0 & \cos(\phi_{BotArm}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{FB.left} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & 0 & -\sin(0) \\ 0 & 1 & 0 \\ \sin(0) & 0 & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{LeftArm}) & \sin(\phi_{LeftArm}) & 0 \\ -\sin(\phi_{LeftArm}) & \cos(\phi_{LeftArm}) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{FB.right} := \left[\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & 0 & -\sin(0) \\ 0 & 1 & 0 \\ \sin(0) & 0 & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(-\phi_{RightArm}) & \sin(-\phi_{RightArm}) & 0 \\ -\sin(-\phi_{RightArm}) & \cos(-\phi_{RightArm}) & 0 \\ 0 & 0 & 1 \end{pmatrix} \right]$$

$$R_{FB.top} = \begin{pmatrix} 0.707 & 0 & 0.707 \\ 0 & 1 & 0 \\ -0.707 & 0 & 0.707 \end{pmatrix} \quad R_{FB.bot} = \begin{pmatrix} 0.695 & 0 & -0.719 \\ 0 & 1 & 0 \\ 0.719 & 0 & 0.695 \end{pmatrix} \quad R_{FB.left} = \begin{pmatrix} 0.707 & 0.707 & 0 \\ -0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R_{FB.right} = \begin{pmatrix} 0.707 & -0.707 & 0 \\ 0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Wedges: $\phi_{LeftAilTop} := \phi_{LeftAil} - \phi_{Wedge}$

$\phi_{LeftAilBot} := \phi_{LeftAil} + \phi_{Wedge}$

$\phi_{RightAilTop} := \phi_{RightAil} - \phi_{Wedge}$

$\phi_{RightAilBot} := \phi_{RightAil} + \phi_{Wedge}$

$$R_{FB.leftAilTop} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{LeftAilTop}) & 0 & -\sin(\phi_{LeftAilTop}) \\ 0 & 1 & 0 \\ \sin(\phi_{LeftAilTop}) & 0 & \cos(\phi_{LeftAilTop}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R_{FB.leftAilTop} = \begin{pmatrix} 0.924 & 0 & 0.383 \\ 0 & 1 & 0 \\ -0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FB.leftAilBot} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{LeftAilBot}) & 0 & -\sin(\phi_{LeftAilBot}) \\ 0 & 1 & 0 \\ \sin(\phi_{LeftAilBot}) & 0 & \cos(\phi_{LeftAilBot}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R_{FB.leftAilBot} = \begin{pmatrix} 0.924 & 0 & -0.383 \\ 0 & 1 & 0 \\ 0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FB.rightAilTop} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{RightAilTop}) & 0 & -\sin(\phi_{RightAilTop}) \\ 0 & 1 & 0 \\ \sin(\phi_{RightAilTop}) & 0 & \cos(\phi_{RightAilTop}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R_{FB.rightAilTop} = \begin{pmatrix} 0.924 & 0 & 0.383 \\ 0 & 1 & 0 \\ -0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FB.rightAilBot} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(0) & \sin(0) \\ 0 & -\sin(0) & \cos(0) \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi_{RightAilBot}) & 0 & -\sin(\phi_{RightAilBot}) \\ 0 & 1 & 0 \\ \sin(\phi_{RightAilBot}) & 0 & \cos(\phi_{RightAilBot}) \end{pmatrix} \cdot \begin{pmatrix} \cos(0) & \sin(0) & 0 \\ -\sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R_{FB.rightAilBot} = \begin{pmatrix} 0.924 & 0 & -0.383 \\ 0 & 1 & 0 \\ 0.383 & 0 & 0.924 \end{pmatrix}$$

Go to F from A

$R_{FA} = R_{FB} \cdot R_{BA}$

Flaps

$R_{FA.top} := R_{FB.top} \cdot R_{BA}$ $\quad R_{FA.top} = \begin{pmatrix} 0.707 & 0 & 0.707 \\ 0 & 1 & 0 \\ -0.707 & 0 & 0.707 \end{pmatrix}$

$R_{FA.bot} := R_{FB.bot} \cdot R_{BA}$ $\quad R_{FA.bot} = \begin{pmatrix} 0.695 & 0 & -0.719 \\ 0 & 1 & 0 \\ 0.719 & 0 & 0.695 \end{pmatrix}$

$R_{FA.left} := R_{FB.left} \cdot R_{BA}$ $\quad R_{FA.left} = \begin{pmatrix} 0.707 & 0.707 & 0 \\ -0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

$R_{FA.right} := R_{FB.right} \cdot R_{BA}$ $\quad R_{FA.right} = \begin{pmatrix} 0.707 & -0.707 & 0 \\ 0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Wedges

$$R_{FA.leftAilTop} := R_{FB.leftAilTop} \cdot R_{BA} \qquad R_{FA.leftAilTop} = \begin{pmatrix} 0.924 & 0 & 0.383 \\ 0 & 1 & 0 \\ -0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FA.leftAilBot} := R_{FB.leftAilBot} \cdot R_{BA} \qquad R_{FA.leftAilBot} = \begin{pmatrix} 0.924 & 0 & -0.383 \\ 0 & 1 & 0 \\ 0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FA.rightAilTop} := R_{FB.rightAilTop} \cdot R_{BA} \qquad R_{FA.rightAilTop} = \begin{pmatrix} 0.924 & 0 & 0.383 \\ 0 & 1 & 0 \\ -0.383 & 0 & 0.924 \end{pmatrix}$$

$$R_{FA.rightAilBot} := R_{FB.rightAilBot} \cdot R_{BA} \qquad R_{FA.rightAilBot} = \begin{pmatrix} 0.924 & 0 & -0.383 \\ 0 & 1 & 0 \\ 0.383 & 0 & 0.924 \end{pmatrix}$$

The inward normal is in the b2 direction and velocity of the wind is in the -a1 direction, then the projected area is:

$\cos(\alpha) = v \cdot n$

$$c\alpha_{top} := -R_{FA.top_{3,1}} \qquad c\alpha_{top} = 0.707 \qquad (-a_1 \cdot f_3)$$

$$c\alpha_{bot} := R_{FA.bot_{3,1}} \qquad c\alpha_{bot} = 0.719 \qquad (-a_1 \cdot -f_3)$$

$$c\alpha_{left} := -R_{FA.left_{2,1}} \qquad c\alpha_{left} = 0.707 \qquad (-a_1 \cdot f_2)$$

$$c\alpha_{right} := R_{FA.right_{2,1}} \qquad c\alpha_{right} = 0.707 \qquad (-a_1 \cdot -f_2)$$

$$c\alpha_{LAilTop} := -R_{FA.leftAilTop_{3,1}} \qquad c\alpha_{LAilTop} = 0.383 \qquad (-a_1 \cdot f_3)$$

$$c\alpha_{LAilBot} := R_{FA.leftAilBot_{3,1}} \qquad c\alpha_{LAilBot} = 0.383 \qquad (-a_1 \cdot -f_3)$$

$$c\alpha_{RAilTop} := -R_{FA.rightAilTop_{3,1}} \qquad c\alpha_{RAilTop} = 0.383 \qquad (-a_1 \cdot f_3)$$

$$c\alpha_{RAilBot} := R_{FA.rightAilBot_{3,1}} \qquad c\alpha_{RAilBot} = 0.383 \qquad (-a_1 \cdot -f_3)$$

Calc projected areas in the A frame:

$A_{top} := A_{pan} \cdot \Phi(c\alpha_{top}) \cdot c\alpha_{top}$ $\quad A_{top} = 0.707\ m^2$

$A_{bot} := A_{pan} \cdot \Phi(c\alpha_{bot}) \cdot c\alpha_{bot}$ $\quad A_{bot} = 0.719\ m^2$

$A_{left} := A_{pan} \cdot \Phi(c\alpha_{left}) \cdot c\alpha_{left}$ $\quad A_{left} = 0.707\ m^2$

$A_{right} := A_{pan} \cdot \Phi(c\alpha_{right}) \cdot c\alpha_{right}$ $\quad A_{right} = 0.707\ m^2$

$A_{LAilTop} := A_{Ail} \cdot \Phi(c\alpha_{LAilTop}) \cdot c\alpha_{LAilTop}$ $\quad A_{LAilTop} = 0.383\ m^2$

$A_{LAilBotl} := A_{Ail} \cdot \Phi(c\alpha_{LAilBot}) \cdot c\alpha_{LAilBot}$ $\quad A_{LAilBotl} = 0.383\ m^2$

$A_{RAilTop} := A_{Ail} \cdot \Phi(c\alpha_{RAilTop}) \cdot c\alpha_{RAilTop}$ $\quad A_{RAilTop} = 0.383\ m^2$

$A_{RAilBotl} := A_{Ail} \cdot \Phi(c\alpha_{RAilBot}) \cdot c\alpha_{RAilBot}$ $\quad A_{RAilBotl} = 0.383\ m^2$

Calc Forces in the A frame:

$F_{topA} := \frac{1}{2} \cdot C_D \cdot A_{top} \cdot \rho \cdot V^2 \cdot v_{hat}$ $\quad F_{topA} = \begin{pmatrix} -4.226 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$

$F_{botA} := \frac{1}{2} \cdot C_D \cdot A_{bot} \cdot \rho \cdot V^2 \cdot v_{hat}$ $\quad F_{botA} = \begin{pmatrix} -4.299 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$

$F_{leftA} := \frac{1}{2} \cdot C_D \cdot A_{left} \cdot \rho \cdot V^2 \cdot v_{hat}$ $\quad F_{leftA} = \begin{pmatrix} -4.226 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$

$F_{rightA} := \frac{1}{2} \cdot C_D \cdot A_{right} \cdot \rho \cdot V^2 \cdot v_{hat}$ $\quad F_{rightA} = \begin{pmatrix} -4.226 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$

Calc Forces in the A frame for the Ailerons:

$$F_{LAilTopA} := \frac{1}{2} \cdot C_D \cdot A_{LAilTop} \cdot \rho \cdot V^2 \cdot v_{hat} \qquad F_{LAilTopA} = \begin{pmatrix} -2.287 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$$

$$F_{LAilBotA} := \frac{1}{2} \cdot C_D \cdot A_{LAilBotl} \cdot \rho \cdot V^2 \cdot v_{hat} \qquad F_{LAilBotA} = \begin{pmatrix} -2.287 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$$

$$F_{RAilTopA} := \frac{1}{2} \cdot C_D \cdot A_{RAilTop} \cdot \rho \cdot V^2 \cdot v_{hat} \qquad F_{RAilTopA} = \begin{pmatrix} -2.287 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$$

$$F_{RAilBotA} := \frac{1}{2} \cdot C_D \cdot A_{RAilBotl} \cdot \rho \cdot V^2 \cdot v_{hat} \qquad F_{RAilBotA} = \begin{pmatrix} -2.287 \times 10^{-6} \\ 0 \\ 0 \end{pmatrix} N$$

Calculate Forces in F frame (Note use only the inward normal component vector):

$$F_{topF} := (0 \ 0 \ 1) \cdot R_{FA.top} \cdot F_{topA} \cdot (0 \ 0 \ 1)^T \qquad F_{topF} = \begin{pmatrix} 0 \\ 0 \\ 2.988 \times 10^{-6} \end{pmatrix} N$$

$$F_{botF} := (0 \ 0 \ 1) \cdot R_{FA.bot} \cdot F_{botA} \cdot (0 \ 0 \ 1)^T \qquad F_{botF} = \begin{pmatrix} 0 \\ 0 \\ -3.092 \times 10^{-6} \end{pmatrix} N$$

$$F_{leftF} := (0 \ 1 \ 0) \cdot R_{FA.left} \cdot F_{leftA} \cdot (0 \ 1 \ 0)^T \qquad F_{leftF} = \begin{pmatrix} 0 \\ 2.988 \times 10^{-6} \\ 0 \end{pmatrix} N$$

$$F_{rightF} := (0 \ 1 \ 0) \cdot R_{FA.right} \cdot F_{rightA} \cdot (0 \ 1 \ 0)^T \qquad F_{rightF} = \begin{pmatrix} 0 \\ -2.988 \times 10^{-6} \\ 0 \end{pmatrix} N$$

$$F_{LAilTopF} := (0\ \ 0\ \ 1)\cdot R_{FA.leftAilTop}\cdot F_{LAilTopA}\cdot(0\ \ 0\ \ 1)^{T} \qquad F_{LAilTopF} = \begin{pmatrix} 0 \\ 0 \\ 8.752\times10^{-7} \end{pmatrix} N$$

$$F_{LAilBotF} := (0\ \ 0\ \ 1)\cdot R_{FA.leftAilBot}\cdot F_{LAilBotA}\cdot(0\ \ 0\ \ 1)^{T} \qquad F_{LAilBotF} = \begin{pmatrix} 0 \\ 0 \\ -8.752\times10^{-7} \end{pmatrix} N$$

$$F_{RAilTopF} := (0\ \ 0\ \ 1)\cdot R_{FA.rightAilTop}\cdot F_{RAilTopA}\cdot(0\ \ 0\ \ 1)^{T} \qquad F_{RAilTopF} = \begin{pmatrix} 0 \\ 0 \\ 8.752\times10^{-7} \end{pmatrix} N$$

$$F_{RAilBotF} := (0\ \ 0\ \ 1)\cdot R_{FA.rightAilBot}\cdot F_{RAilBotA}\cdot(0\ \ 0\ \ 1)^{T} \qquad F_{RAilBotF} = \begin{pmatrix} 0 \\ 0 \\ -8.752\times10^{-7} \end{pmatrix} N$$

Convert to B frame:

$$F_{topB} := R_{FB.top}{}^{T}\cdot F_{topF} \qquad F_{topB} = \begin{pmatrix} -2.113\times10^{-6} \\ 0 \\ 2.113\times10^{-6} \end{pmatrix} N \qquad F_{leftB} := R_{FB.left}{}^{T}\cdot F_{leftF} \qquad F_{leftB} = \begin{pmatrix} -2.113\times10^{-6} \\ 2.113\times10^{-6} \\ 0 \end{pmatrix} N$$

$$F_{botB} := R_{FB.bot}{}^{T}\cdot F_{botF} \qquad F_{botB} = \begin{pmatrix} -2.224\times10^{-6} \\ 0 \\ -2.148\times10^{-6} \end{pmatrix} N \qquad F_{rightB} := R_{FB.right}{}^{T}\cdot F_{rightF} \qquad F_{rightB} = \begin{pmatrix} -2.113\times10^{-6} \\ -2.113\times10^{-6} \\ 0 \end{pmatrix} N$$

Convert Ailerons to B frame from F:

$$F_{LAilTopB} := R_{FB.leftAilTop}{}^{T} \cdot F_{LAilTopF} \qquad F_{LAilTopB} = \begin{pmatrix} -3.349 \times 10^{-7} \\ 0 \\ 8.085 \times 10^{-7} \end{pmatrix} N$$

$$F_{LAilBotB} := R_{FB.leftAilBot}{}^{T} \cdot F_{LAilBotF} \qquad F_{LAilBotB} = \begin{pmatrix} -3.349 \times 10^{-7} \\ 0 \\ -8.085 \times 10^{-7} \end{pmatrix} N$$

$$F_{RAilTopB} := R_{FB.rightAilTop}{}^{T} \cdot F_{RAilTopF} \qquad F_{RAilTopB} = \begin{pmatrix} -3.349 \times 10^{-7} \\ 0 \\ 8.085 \times 10^{-7} \end{pmatrix} N$$

$$F_{RAilBotB} := R_{FB.rightAilBot}{}^{T} \cdot F_{RAilBotF} \qquad F_{RAilBotB} = \begin{pmatrix} -3.349 \times 10^{-7} \\ 0 \\ -8.085 \times 10^{-7} \end{pmatrix} N$$

r is the position vector from the s/c cg to the centroid of the panels B Frame:

Torque = r x f where:

$$r_{top} := \begin{bmatrix} -.5 \cdot L_{b1} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \cos\left(\phi_{TopArm}\right) \\ 0 \\ -.5 \cdot L_{b3} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \sin\left(\phi_{TopArm}\right) \end{bmatrix} - CG \qquad r_{top} = \begin{pmatrix} -2.268 \\ 0 \\ -2.268 \end{pmatrix} m$$

$$r_{bot} := \begin{bmatrix} -.5 \cdot L_{b1} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \cos\left(\phi_{BotArm}\right) \\ 0 \\ .5 \cdot L_{b3} + \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \sin\left(\phi_{BotArm}\right) \end{bmatrix} - CG \qquad r_{bot} = \begin{pmatrix} -2.237 \\ 0 \\ 2.298 \end{pmatrix} m$$

$$r_{left} := \begin{bmatrix} -.5 \cdot L_{b1} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \cos\left(\phi_{LeftArm}\right) \\ -.5 \cdot L_{b2} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \sin\left(\phi_{LeftArm}\right) \\ 0 \end{bmatrix} - CG \qquad r_{left} = \begin{pmatrix} -2.268 \\ -2.268 \\ 0 \end{pmatrix} m$$

$$r_{right} := \begin{bmatrix} -.5 \cdot L_{b1} - \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \cos\left(\phi_{RightArm}\right) \\ .5 \cdot L_{b2} + \left(L_{arm} + 0.5 \cdot L_{panel}\right) \cdot \sin\left(\phi_{RightArm}\right) \\ 0 \end{bmatrix} - CG \qquad r_{right} = \begin{pmatrix} -2.268 \\ 2.268 \\ 0 \end{pmatrix} m$$

$$r_{LAilTop} := \begin{bmatrix} -\frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \sin\left(\phi_{LeftAil}\right) \\ -\left(.5 \cdot L_{b2} + 0.5 \cdot L_{Ail}\right) \\ -\frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \cos\left(\phi_{LeftAil}\right) \end{bmatrix} - CG \qquad r_{RAilTop} := \begin{bmatrix} -\frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \sin\left(\phi_{RightAil}\right) \\ \left(.5 \cdot L_{b2} + 0.5 \cdot L_{Ail}\right) \\ -\frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \cos\left(\phi_{RightAil}\right) \end{bmatrix} - CG$$

$$r_{LAilBot} := \begin{bmatrix} \frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \sin\left(\phi_{LeftAil}\right) \\ -\left(.5 \cdot L_{b2} + 0.5 \cdot L_{Ail}\right) \\ \frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \cos\left(\phi_{LeftAil}\right) \end{bmatrix} - CG \qquad r_{RAilBot} := \begin{bmatrix} \frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \sin\left(\phi_{RightAil}\right) \\ \left(.5 \cdot L_{b2} + 0.5 \cdot L_{Ail}\right) \\ \frac{W_{Ail}}{2} \cdot \sin\left(\phi_{Wedge}\right) \cdot \cos\left(\phi_{RightAil}\right) \end{bmatrix} - CG$$

$$r_{LAilTop} = \begin{pmatrix} 0 \\ -1 \\ -0.191 \end{pmatrix} m \qquad r_{LAilBot} = \begin{pmatrix} 0 \\ -1 \\ 0.191 \end{pmatrix} m \qquad r_{RAilTop} = \begin{pmatrix} 0 \\ 1 \\ -0.191 \end{pmatrix} m \qquad r_{RAilBot} = \begin{pmatrix} 0 \\ 1 \\ 0.191 \end{pmatrix} m$$

Calc Torque in the body frame:

$$M_{topB} := r_{top} \times F_{topB} \qquad M_{topB} = \begin{pmatrix} 0 \\ 9.583 \times 10^{-6} \\ 0 \end{pmatrix} N \cdot m$$

$$M_{botB} := r_{bot} \times F_{botB} \qquad M_{botB} = \begin{pmatrix} 0 \\ -9.917 \times 10^{-6} \\ 0 \end{pmatrix} N \cdot m$$

$$M_{leftB} := r_{left} \times F_{leftB} \qquad M_{leftB} = \begin{pmatrix} 0 \\ 0 \\ -9.583 \times 10^{-6} \end{pmatrix} N \cdot m$$

$$M_{rightB} := r_{right} \times F_{rightB} \qquad M_{rightB} = \begin{pmatrix} 0 \\ 0 \\ 9.583 \times 10^{-6} \end{pmatrix} N \cdot m$$

$$M_{LAilTopB} := r_{LAilTop} \times F_{LAilTopB} \qquad M_{LAilTopB} = \begin{pmatrix} -8.085 \times 10^{-7} \\ 6.408 \times 10^{-8} \\ -3.349 \times 10^{-7} \end{pmatrix} N \cdot m$$

$$M_{LAilBotB} := r_{LAilBot} \times F_{LAilBotB} \qquad M_{LAilBotB} = \begin{pmatrix} 8.085 \times 10^{-7} \\ -6.408 \times 10^{-8} \\ -3.349 \times 10^{-7} \end{pmatrix} N \cdot m$$

$$M_{RAilTopB} := r_{RAilTop} \times F_{RAilTopB} \qquad M_{RAilTopB} = \begin{pmatrix} 8.085 \times 10^{-7} \\ 6.408 \times 10^{-8} \\ 3.349 \times 10^{-7} \end{pmatrix} N \cdot m$$

$$M_{RAilBotB} := r_{RAilBot} \times F_{RAilBotB} \qquad M_{RAilBotB} = \begin{pmatrix} -8.085 \times 10^{-7} \\ -6.408 \times 10^{-8} \\ 3.349 \times 10^{-7} \end{pmatrix} N \cdot m$$

Total Torque due to flap panels in Body frame:

$$M_{total} := M_{topB} + M_{botB} + M_{leftB} + M_{rightB} + M_{LAilTopB} + M_{LAilBotB} + M_{RAilTopB} + M_{RAilBotB} \qquad M_{total} = \begin{pmatrix} 0 \\ -3.341 \times 10^{-7} \\ 0 \end{pmatrix} N \cdot m$$

Appendix B. Matlab® Code

B.1 Matlab® Eigenvalues and Eigenvectors with Torques

Listing B.1: This Matlab® code calculates the eigenvalues and eigenvectors for the model validation.
(appendix2/EigsEOMwTorques.m)

```
% D. Guettler
clc
clear all
Alt=300;  %km
syms A B C n Rho Vel

% MAT1=[0 0 0 1 0 0; 0 0 0 0 1 0; 0 0 0 0 0 1;...
%       ((C-B)/A)*n^2 0 0 0 0 ((C+A-B)/A)*n;...
%       0 -4.621*Rho*Vel^2/B 0 0 0 0;...
%       0 0 ((A-B)*n^2-4.414*Rho*Vel^2)/C ((B-A-C)/C)*n 0 0]

%MAT2 is the 4x4 matrix of MAT1
MAT2=[0 0 1 0; 0 0 0 1;...
      ((C-B)/A)*n^2 0 0 ((C+A-B)/A)*n;...
      0 ((A-B)*n^2-4.414*Rho*Vel^2)/C ((B-A-C)/C)*n 0]

%Orbital rate for 300km altitude in rad/sec
[n,T,freq,Vel] = OrbRate(Alt) %n = 0.001157 %for 300km alt

%Calculate Density
Rho = atmos_exp(Alt)
%Rho=0;

%MOI  B is always major axis, C is always minor axis
A=20.
B=30.
C=10.

% MAT1=[0 0 0 1 0 0; 0 0 0 0 1 0; 0 0 0 0 0 1;...
%       ((C-B)/A)*n^2 0 0 0 0 ((C+A-B)/A)*n;...
%       0 -4.621*Rho*Vel^2/B 0 0 0 0;...
%       0 0 ((A-B)*n^2-4.414*Rho*Vel^2)/C ((B-A-C)/C)*n 0 0]

%MAT2 is the 4x4 matrix of MAT1
MAT2=[0 0 1 0; 0 0 0 1;...
      ((C-B)/A)*n^2 0 0 ((C+A-B)/A)*n;...
      0 ((A-B)*n^2-4.414*Rho*Vel^2)/C ((B-A-C)/C)*n 0]

% %From 6x6 Matrix
% [EigVec1,EigVal1]=eig(MAT1);
```

```
% EigVal1 = simplify(EigVal1);
% EigVal1 = vpa(EigVal1,3)
% EigVec1 = simplify(EigVec1);
% EigVec1 = vpa(EigVec1,3)

%From 4x4 matrix
[EigVec2,EigVal2]=eig(MAT2);
EigVal2 = simplify(EigVal2);
EigVal2 = vpa(EigVal2,3)
EigVec2 = simplify(EigVec2);
EigVec2 = vpa(EigVec2,3)

%Find ICs for each run
EigValCase1=EigVal2(1,1)
IC1=(1+i)*[EigVec2(1,1) EigVec2(2,1) EigVec2(3,1) EigVec2(4,1)...
   ]'+...
    (1-i)*[EigVec2(1,2) EigVec2(2,2) EigVec2(3,2) EigVec2(4,2)]'

EigValCase2=EigVal2(3,3)
IC2=(1+i)*[EigVec2(1,3) EigVec2(2,3) EigVec2(3,3) EigVec2(4,3)...
   ]'+...
    (1-i)*[EigVec2(1,4) EigVec2(2,4) EigVec2(3,4) EigVec2(4,4)]'
```

B.2 Matlab® Curve Fit Algorithm

This algorithm uses a peak finding function to locate all the peaks of the oscillations of the angles between the orbit and body frame. Once the peaks were located, the main code fit an exponential curve to the points based on an approximated time constant. The time constant was then varied until an exponential curve closely matched the peaks. This code was used to fit an exponential curve to the decaying oscillations for the time constant calculations in Chapter V.

Listing B.2: This Matlab® code fits the peak amplitudes to an exponentially decaying function.
(appendix2/TimeConstCalc.m)

```
% peaks = function peakfinder(roll,time)
% D. Guettler
clear;clc;
load thetaout.mat
%************************************************
%Pick tau as time constant compare plot
%************************************************
tau=38000
```

```
%*************************************************
%*************************************************
% ROLL
%peaks  = peakfinder(theta(2,:)',theta(1,:)');
% PITCH
%peaks = peakfinder(theta(3,:)',theta(1,:)');
% YAW
peaks   = peakfinder(theta(4,:)',theta(1,:)');
%*************************************************
%*************************************************
%*************************************************
%Create vectors for time calculation
%*************************************************
t=peaks(:,1)
y=peaks(:,2)
%*************************************************
%create plot
%*************************************************
X = [exp(-t/tau)];
% Calculate model coefficients
a = X\y

T = (0:1:200000)';
Y = [exp(-T/tau)]*a;

% Create figure
figure1 = figure;
% Create axes
axes1 = axes('Parent',figure1);
box('on');
grid('on');
hold('all');
% Create plot
plot1 = plot(T,Y,'Parent',axes1), grid on
% Create plot
plot2 = plot(t,y,'LineStyle','none','Marker','o',...
    'Parent',axes1);
% Create xlabel
xlabel('Time (s)');
% Create ylabel
ylabel('Amplitude (rad)');
% Create legend
legend1 = legend(axes1,{'Exponential Curve Fit',...
    'Peak Locations'},'Position',...
    [0.5827 0.8245 0.3232 0.1002]);
```

Listing B.3: This Matlab® code determines the positive peak amplitudes of oscillation.
(appendix2/PeakFinder.m)

```
% peakfinder.m
% D. Guettler

function peaks = peakfinder(roll,time)

%Need roll, time as column vectors of equal dimension
%Includes initial and final points

[cols,rows]=size(roll);


if roll(1,1)>0
    rollpeaknums(1)=1;
    j=2;
else j=1;
end

for i=2:(cols-1)
    if roll(i,1)>roll(i-1,1) & roll(i,1)>roll(i+1,1)...
            & roll(i,1)>0
        rollpeaknums(j)=i;
        j=j+1;
    end
end
rollpeaknums(j)=cols;

for i=1:j
    peaks(i,1)=time(rollpeaknums(i),1);
    peaks(i,2)=roll(rollpeaknums(i),1);
end
```

Appendix C. The Atmosphere

C.1 Altitude vs Density Plot (NASA) [3, page 62]

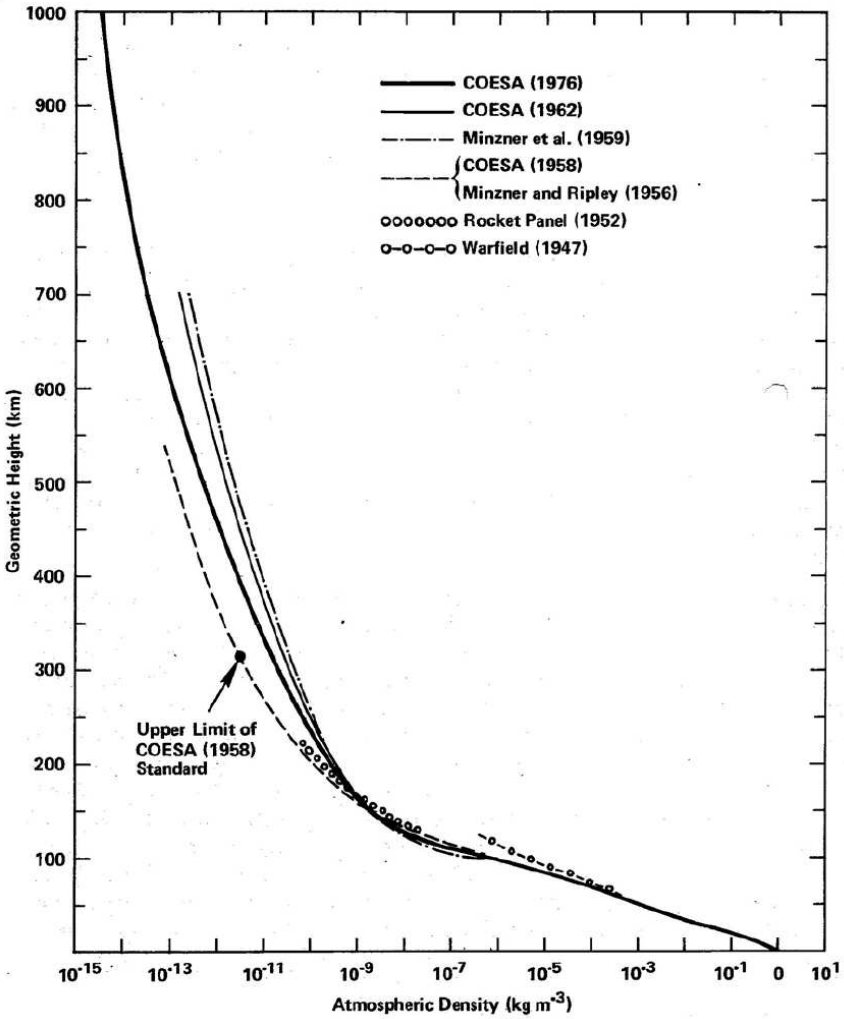


Figure C.1: Atmospheric Density/Altitude plot

C.2 Matlab® Density Calculations)

For the computer simulations (see Appendix C), an exponential atmospheric model was used. This model was tested at the following altitudes and compared to the density plot, Figure C.1. The results are displayed in Table C.1 exponential atmospheric density calculations in Matlab® compare with those found in the plot.

Table C.1: Altitude and Density Results from Matlab® .

Altitude	Calculated Density	Density from Figure C.1
150	$2.07\text{x}10^{-9}$	$2\text{x}10^{-9}$
200	$2.79\text{x}10^{-10}$	$2\text{x}10^{-10}$
250	$7.25\text{x}10^{-11}$	$7\text{x}10^{-11}$
300	$2.42\text{x}10^{-11}$	$1\text{x}10^{-11}$
350	$9.52\text{x}10^{-12}$	$7\text{x}10^{-12}$
400	$3.73\text{x}10^{-12}$	$3\text{x}10^{-12}$
450	$1.59\text{x}10^{-12}$	$1\text{x}10^{-12}$
500	$6.97\text{x}10^{-13}$	$5\text{x}10^{-13}$
550	$3.18\text{x}10^{-13}$	$2\text{x}10^{-13}$
600	$1.45\text{x}10^{-13}$	$1\text{x}10^{-13}$

Appendix D. Simulink® Model

D.1 Simulink® Model Top Level

The following figures illustrate the computer model used to simulate the dynamics of using atmospheric drag for attitude control. The software used is Simulink®. Figure D.1 is the overall Simulink® model. It is laid out such that the inputs are along the left side, the outputs are on the right. This model represents equation 3.47 with all the variables on the input side. The model outputs the control angles, angular accelerations, angular rates and attitude angle with respect to the orbital frame. All satellite dimensions are input as meters, angles are input as degrees and altitude is input as km. All the outputs containing angles are in degrees.

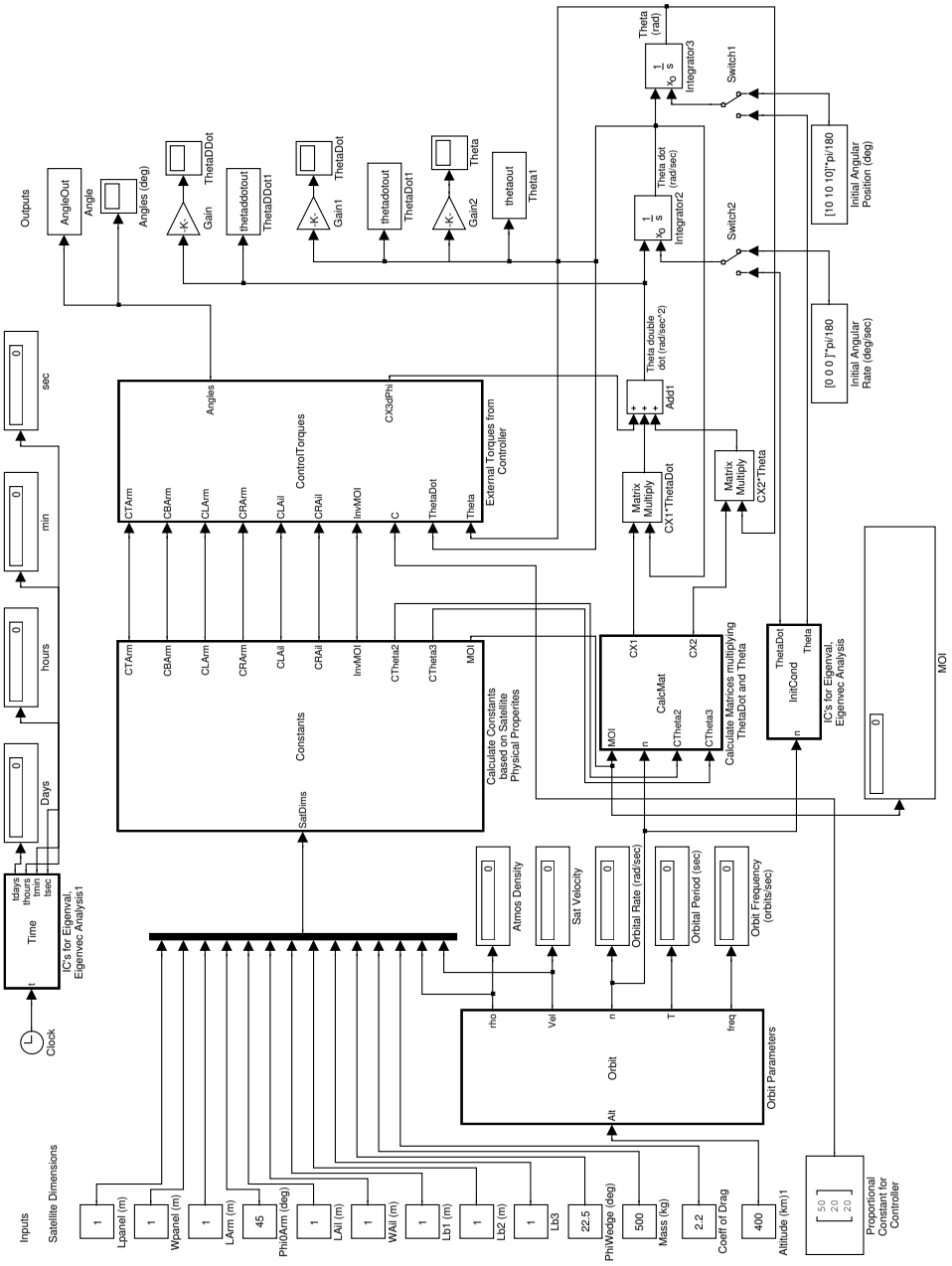


Figure D.1: Overall Simulink® Model

Listing D.1: This Matlab® function calculates the approximate density of the atmosphere at a given altitude, tangential orbital velocity for a circular orbit, orbital rate, orbital period and orbital frequency.
(appendix4/Orbit.m)

```
function [rho, Vel,n,T,freq]=Orbit(Alt)

% Exponential Atmospheric Model
% Accepts as INPUT the orbital radius of the
% satellite; OUTPUTS the atmospheric density
% for that altitude based on the mean equatorial
% radius of Earth (density units: kg/m^3).
% NOTE: really only valid for altitudes less than
% 1000 km.  Taken from Vallado, pg 537.
% Distances in km

%*****************************************************
% Calculate Orbit properites (D. Guettler)
%*****************************************************
mu = 3.986*10^14  ; %(km^3/s^2) Standard Gravitational
                    %Parameter
REarth = 6378135      ; %(km) Radius of Earth in
ROrbit = REarth+Alt*1000   ; %Radius from Earth's
                             %center to orbit

OrbRate = sqrt(mu/ROrbit^3);  %OrbRate is the orbital
                              %rate in rad/sec
Vel = sqrt(mu/ROrbit)      ;  %Orbit Velocity
n = OrbRate                ;  %Orbital rate (rad/sec)
T = 2*pi*(1/OrbRate)       ;  %T=orbital (s)
freq=1/T                   ;  %Frequency (orbits/sec)
%*****************************************************
% Calculate Density (B. Hajovsky)
%*****************************************************
% Term Definitions:
% h0 = Base Altitude (km)
% rho0 = Nominal Density (kg/m^3)
% H = Scale Height (km)
% Define the Earth radius to calculate the altitude:
R=Alt;
r = R;
rho0=0;
h0=0;
H=0;
if r>=0 & r<25
    h0 = 0;
    rho0 = 1.225;
    H = 7.249;
```

```
elseif r>=25 & r<30
    h0 = 25;
    rho0 = 3.899e-2;
    H = 6.349;
elseif r>=30 & r<40
    h0 = 30;
    rho0 = 1.774e-2;
    H = 6.682;
elseif r>=40 & r<50
    h0 = 40;
    rho0 = 3.972e-3;
    H = 7.554;
elseif r>=50 & r<60
    h0 = 50;
    rho0 = 1.057e-3;
    H = 8.382;
elseif r>=60 & r<70
    h0 = 60;
    rho0 = 3.206e-4;
    H = 7.714;
elseif r>=70 & r<80
    h0 = 70;
    rho0 = 8.770e-5;
    H = 6.549;
elseif r>=80 & r<90
    h0 = 80;
    rho0 = 1.905e-5;
    H = 5.799;
elseif r>=90 & r<100
    h0 = 90;
    rho0 = 3.396e-6;
    H = 5.382;
elseif r>=100 & r<110
    h0 = 100;
    rho0 = 5.297e-7;
    H = 5.877;
elseif r>=110 & r<120
    h0 = 110;
    rho0 = 9.661e-8;
    H = 7.263;
elseif r>=120 & r<130
    h0 = 120;
    rho0 = 2.438e-8;
    H = 9.473;
elseif r>=130 & r<140
    h0 = 130;
    rho0 = 8.484e-9;
    H = 12.636;
elseif r>=140 & r<150
    h0 = 140;
    rho0 = 3.845e-9;
```

```
        H = 16.149;
    elseif r>=150 & r<180
        h0 = 150;
        rho0 = 2.070e-9;
        H = 22.523;
    elseif r>=180 & r<200
        h0 = 180;
        rho0 = 5.464e-10;
        H = 29.740;
    elseif r>=200 & r<250
        h0 = 200;
        rho0 = 2.789e-10;
        H = 37.105;
    elseif r>=250 & r<300
        h0 = 250;
        rho0 = 7.248e-11;
        H = 45.546;
    elseif r>=300 & r<350
        h0 = 300;
        rho0 = 2.418e-11;
        H = 53.628;
    elseif r>=350 & r<400
        h0 = 350;
        rho0 = 9.518e-12;
        H = 53.298;
    elseif r>=400 & r<450
        h0 = 400;
        rho0 = 3.725e-12;
        H = 58.515;
    elseif r>=450 & r<500
        h0 = 450;
        rho0 = 1.585e-12;
        H = 60.828;
    elseif r>=500 & r<600
        h0 = 500;
        rho0 = 6.967e-13;
        H = 63.822;
    elseif r>=600 & r<700
        h0 = 600;
        rho0 = 1.454e-13;
        H = 71.835;
    elseif r>=700 & r<800
        h0 = 700;
        rho0 = 3.614e-14;
        H = 88.667;
    elseif r>=800 & r<900
        h0 = 800;
        rho0 = 1.170e-14;
        H = 124.64;
    elseif r>=900 & r<1000
        h0 = 900;
```

```
        rho0 = 5.245e-15;
        H = 181.05;
    elseif r>1000
        h0 = 1000;
        rho0 = 3.019e-15;
        H = 268.00;
    else
    end

rho = rho0*exp(-(r-h0)/H);
%rho=0;
```

D.3 Function: Constants.m

Listing D.2: This Matlab® function calculates the constants in equations 3.46 and also calculates the MOI matrix based on a given satellite mass. (appendix4/Constants.m)

```
function [CTArm,CBArm,CLArm,CRArm,CLAil,CRAil,InvMOI,...
    CTheta2,CTheta3,MOI]  = Constants(SatDims)
% This block calculates the constants for the
% linearized torques and contains all the satellite
% dimensions.   See MathCad File for explanation.

% D. Guettler
%*******************************************************
% Delcare Variables
%*******************************************************
Lpanel= SatDims(1);%Length of drag panel
                   %(front to back when retracted) (m)
Wpanel= SatDims(2);%Width of panel (m)
LArm =  SatDims(3);%Length of control arm (m)
Phi0Arm = SatDims(4)*pi/180; %Linearization angle for
                             %arms (radians)
LAil =  SatDims(5);%Length of Aileron spanwise  (m)
WAil =  SatDims(6);%Width of Aileron chordwise (m)
Lb1 =   SatDims(7);%cubesat length along b1 axis (m)
Lb2 =   SatDims(8);%Cubesat width along b2 axis (m)
Lb3 =   SatDims(9);%cubesat Height along b3 axis (m)
PhiWedge = SatDims(10)*pi/180; %one half the total
                               %wedge angle (Radians)
Mass = SatDims(11); %Mass of sat in kg
                    %(used to calc MOI matrix)
CD   = SatDims(12); %Drag Coefficient
Rho = SatDims(13);
V = SatDims(14);

%*******************************************************
% Calculate Constants
```

```
%*****************************************************
% Const multiplying deltaPhiTopArm
CTArm=(.5* Lpanel^2*Wpanel*cos(Phi0Arm)*sin(Phi0Arm) ...
    + Lpanel*Wpanel*LArm*cos(Phi0Arm)*sin(Phi0Arm) ...
    + .75*Lpanel*Wpanel*Lb1 *cos(Phi0Arm)^2*sin(Phi0Arm) ...
    - .25*Lpanel*Wpanel*Lb1*sin(Phi0Arm) ...
    + .75*Lpanel*Wpanel*Lb3*cos(Phi0Arm)                ...
    - .75*Lpanel*Wpanel*Lb3*cos(Phi0Arm)^3)*CD*Rho*V^2;
% Const multiplying deltaPhiBotArm
CBArm=-CTArm;
% Const multiplying deltaPhiLeftArm
CLArm=(.25 *Lpanel*Wpanel*Lb1*sin(Phi0Arm) ...
    -.75*Lpanel*Wpanel*Lb2*cos(Phi0Arm)...
    +.75*Lpanel*Wpanel*Lb2*cos(Phi0Arm)^3...
    -1*Lpanel*Wpanel*LArm*cos(Phi0Arm)*sin(Phi0Arm)...
    -.5*Lpanel^2*Wpanel*cos(Phi0Arm)*sin(Phi0Arm)...
    -.75*Lpanel*Wpanel*Lb1*cos(Phi0Arm)^2*sin(Phi0Arm))*CD*Rho*V...
       ^2;
% Const multiplying deltaPhiRightArm
CRArm=-CLArm;
% Const multiplying deltaPhiLeftAileron
CLAil=(1.5*LAil^2*WAil*cos(PhiWedge)^2 *sin(PhiWedge) ...
    -.5*LAil*WAil*Lb2*sin(PhiWedge) ...
    +1.5*LAil*WAil*Lb2*cos(PhiWedge)^2 *sin(PhiWedge) ...
    -.5*LAil^2*WAil*sin(PhiWedge))*CD*Rho*V^2;
% Const multiplying deltaPhiRightAileron
CRAil=-CLAil;
% Const multiplying Theta 2 term
CTheta2=(-1 *Lpanel*Wpanel *Lb1*cos(Phi0Arm)^2*sin(Phi0Arm)  ...
        -1 *Lpanel^2*Wpanel*cos(Phi0Arm)*sin(Phi0Arm)  ...
        +1 *Lpanel*Wpanel*Lb3*cos(Phi0Arm)^3                ...
        -2 *WAil^2*LAil*cos(PhiWedge)*sin(PhiWedge) ...
        -2 *Lpanel*Wpanel*LArm *cos(Phi0Arm)*sin(Phi0Arm)  ...
        +2 *WAil^2*LAil*cos(PhiWedge)^3*sin(PhiWedge) ...
        -1 *Lpanel*Wpanel*Lb3*cos(Phi0Arm))*CD*Rho*V^2;
% Const multiplying Theta 2 term
CTheta3=(-1 *Lpanel^2*Wpanel*cos(Phi0Arm)*sin(Phi0Arm) ...
        -2 *Lpanel*Wpanel*LArm*cos(Phi0Arm)*sin(Phi0Arm) ...
        +1 *Lpanel*Wpanel*Lb2*cos(Phi0Arm)^3                ...
        -1 *Lpanel*Wpanel*Lb1*cos(Phi0Arm)^2*sin(Phi0Arm) ...
        -1 *Lpanel*Wpanel*Lb2*cos(Phi0Arm))*CD*Rho*V^2;

%*****************************************************
% Calc MOI Where pitch is major, Roll is intermediate,
% Yaw is minor
%*****************************************************
Length = 1;
Width  = .8;
Height = 1.25;
Iroll=(1/12)*Mass*(Width^2+Height^2);
Ipitch=(1/12)*Mass*(Length^2+Height^2);
```

```
Iyaw=(1/12)*Mass*(Length^2+Width^2);
%xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
%MOI=[Iroll 0 0;0 Ipitch 0;0 0 Iyaw]; %(kg*m^2)
MOI=[10 0 0;0 20 0;0 0 30];
%xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
%******************************************************
% Calc Inv MOI
%******************************************************
InvMOI=MOI^-1;
```

D.4 Function: ControlTorques.m

Listing D.3: This Matlab® function calculates C_{X_3} (equation 3.50)and multiplies it with the deflections of the control surfaces.
(appendix4/ControlTorques.m)

```
function [Angles,CX3dPhi] = ControlTorques(CTArm,CBArm,CLArm,...
    CRArm,CLAil,CRAil,InvMOI,C,ThetaDot,Theta)
% This block calculates the torques a satellite
% experiences when in LEO, from Theta (the angles
% between the B-frame and the A-frame), PhiArms
% (the angle of each control arm from the sat
% body) and PhiFlaps (the angle of the flaps
% rotating about the arms).  it is assumed the
% inward pointing normal direction for the panels
% never sees the wind.

%D. Guettler

%xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
%EIG TEST  Comment out for control
% RollControl=0;
% PitchControl=0;
% YawControl=0;
%Uncomment for control
RollControl=C(1)*ThetaDot(1);
PitchControl=C(2)*ThetaDot(2);
YawControl=C(3)*ThetaDot(3);
%xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
dPhiRightAil=RollControl;
dPhiLeftAil=-RollControl;

dPhiTopArm=-PitchControl;
dPhiBotArm=PitchControl;

dPhiLeftArm=YawControl;
dPhiRightArm=-YawControl;
Angles=[dPhiRightAil dPhiLeftAil dPhiTopArm dPhiBotArm...
```

```
    dPhiLeftArm dPhiRightArm]*180/pi';

CX3dPhi=InvMOI*[CLAil*dPhiLeftAil+CRAil*dPhiRightAil;
    CTArm*dPhiTopArm+CBArm*dPhiBotArm;
    CLArm*dPhiLeftArm+CRArm*dPhiRightArm];
```

D.5 Function: CalcMat.m

Listing D.4: This Matlab® function calculates the matrices C_{X_1} and C_{X_1} in equations 3.48 and 3.49.
(appendix4/CalcMat.m)

```
function [CX1,CX2] = CalcMat(MOI,n,CTheta2,CTheta3)
% This block calculates the matrix multiplied by
% the ThetaDot vector. See the help menu for details.

%*************************************************************
% Calc Inv MOI
%*************************************************************
InvMOI=MOI^-1;

%*************************************************************
%Calculate the matrix that multiplies the ThetaDot Vector
%*************************************************************
CX1 = InvMOI*[0 0 (MOI(3,3)+MOI(1,1)-MOI(2,2));
    0 0 0; (MOI(2,2)-MOI(1,1)-MOI(3,3)) 0 0]*n;

%*************************************************************
%Calculate the matrix that multiplies the ThetaDot Vector
%*************************************************************
CX2 = InvMOI*[(MOI(3,3)-MOI(2,2))*n^2 0 0;
    0          CTheta2 0;
    0 0 (MOI(1,1)-MOI(2,2))*n^2+CTheta3];
```

D.6 Function: InitCond.m

Listing D.5: This Matlab® function inputs the initial conditions that were calculated from the eigenvector analysis for model validation. Once the model has been validated, these can be turned off so the actual simulations can be run.
(appendix4/InitCond.m)

```
function [ThetaDot,Theta] = InitCond(n)
% this block is to test the model by the eigenval
% eigenvec method. See the help menu for details.
```

```
% D. Guettler
%******For the Linear no torques case******
% For the cases where A=20 B=10 C=30 using the
% 4x4 matrix to get eigvecs
%Case 1:  The first pair of complex conj eigvecs
%Theta=[2 0 2]';
%ThetaDot=[2*n 0 -2*n]';
%Case 2  The second pair of complex conj eigvecs
%Theta=[1.734 0 2]';
%ThetaDot=[n 0 -1.156*n]';

%******For the Linear no torques case******
% For the cases where A=10 B=20 C=30 using the
% 4x4 matrix to get eigvecs
%Case 1:  The first pair of complex conj eigvecs
%Theta=[1 0 1]';
%ThetaDot=[n 0 -n]';
%Case 2  The second pair of complex conj eigvecs
Theta=[-3 0 0]';
ThetaDot=[0 0 n]';

%Theta=[2 0 0]';
%ThetaDot=[.00232 0 0]';
%Theta=[0 0 2]';
%ThetaDot=[0 0 .00232]';
%Theta=[1.414 0 1.414]';
%ThetaDot=[.00163599 0 -.00163599]';
%Theta=[-1.31 0 1.512]';
%ThetaDot=[.000874 0 .101]';

% For the cases where A=20 B=10 C=30 using the
% 6x6 matrix to get eigvecs
%Case 1:  The first pair of complex conj eigvecs
%Theta=[2 0 2]';
%ThetaDot=[2*n 0 -2*n]';
%Case 2  The second pair of complex conj eigvecs
%Theta=[-3/n 0 3.46/n]';
%ThetaDot=[1.73 0 2]';
%Theta=[0 2 2]';
%ThetaDot=[-2*n 0 0]';


%For the linear with torques case
%A=20, B=30, C=10 and using the 4x4 matrix to get eigs
%Case 1 (first pair of complex conj eigvecs)
%Theta=[2 0 0]';
%ThetaDot=[.00232 0 0]';
%Case 2 (2nd pair of complex conj eigvecs)
%Theta=[0 0 2]';
%ThetaDot=[0 0 .0506]';
```

Appendix E. Simulation Plots

E.1 Altitude Variation plots

This Appendix contains all the plots for angular displacement, angular rate, angular acceleration, and control angles for the simulations determine the effects altitude has on attitude stabilization. All inputs are summed up in the following tables.

Table E.1: Density, velocity and orbital rate calculated from Simulink®.

Altitude (km)	Calculated Density ($\frac{kg}{m^3}$)	Velocity ($\frac{m}{s}$)	Orbital Rate ($\frac{rad}{s}$)
200	2.79x10^{-10}	7784	0.001183
300	2.42x10^{-11}	7726	0.001157
400	3.73x10^{-12}	7669	0.001131
500	6.97x10^{-13}	7613	0.001107
600	1.45x10^{-13}	7558	0.001083

Table E.2: Variables input into the Simulink® model for all the altitude variation simulations.

Variable	Value
L_{Panel}	$1m$
W_{Panel}	$1m$
L_{Arm}	$1m$
$\phi_{0_{Arm}}$	$45\circ$
L_{Ail}	$1m$
W_{Ail}	$1m$
L_{b_1}	$1m$
L_{b_2}	$1m$
L_{b_3}	$1m$
ϕ_{Wedge}	22.5°
$Mass$	$500kg$
C_D	2.2

$$MOI = \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} = \begin{bmatrix} 92 & 0 & 0 \\ 0 & 107 & 0 \\ 0 & 0 & 68 \end{bmatrix} kg \cdot m^2 \tag{E.1}$$

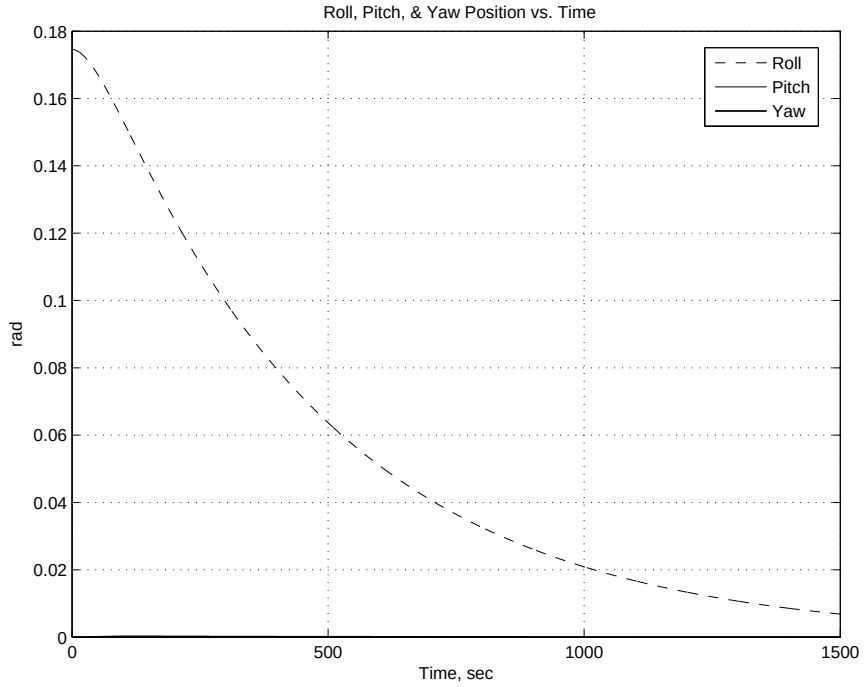


Figure E.1: Angular displacement at 200km with roll offset 10°.

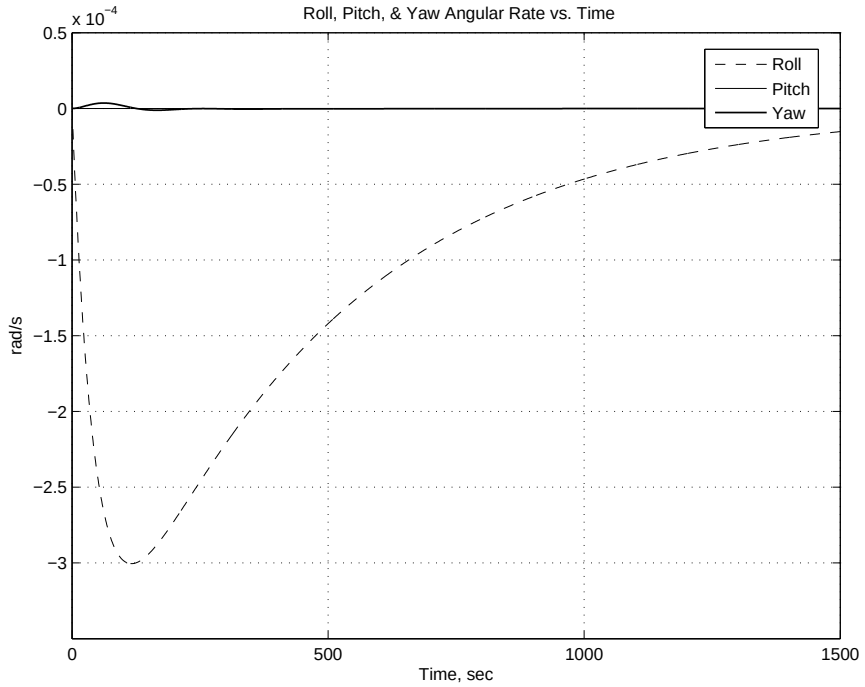


Figure E.2: Angular rate at 200km with roll offset 10°.

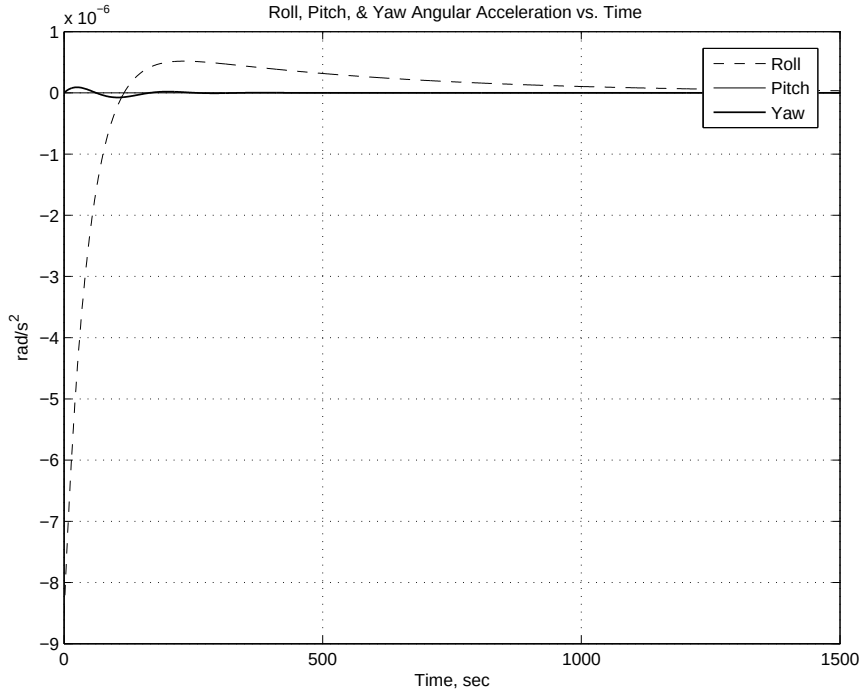


Figure E.3: Angular Acceleration at 200km with roll offset 10°.

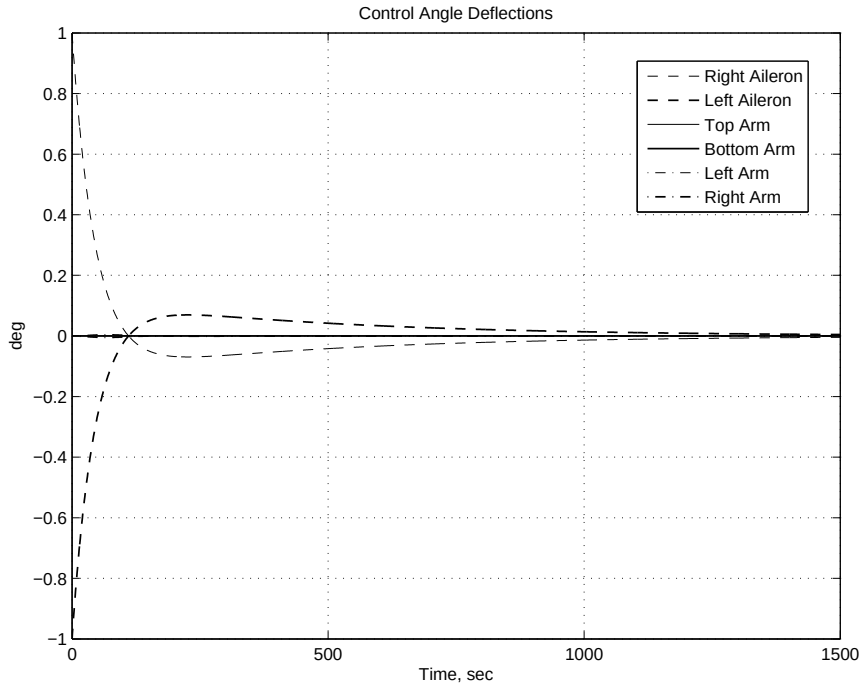


Figure E.4: Control angle deflections 200 km with roll offset 10°.

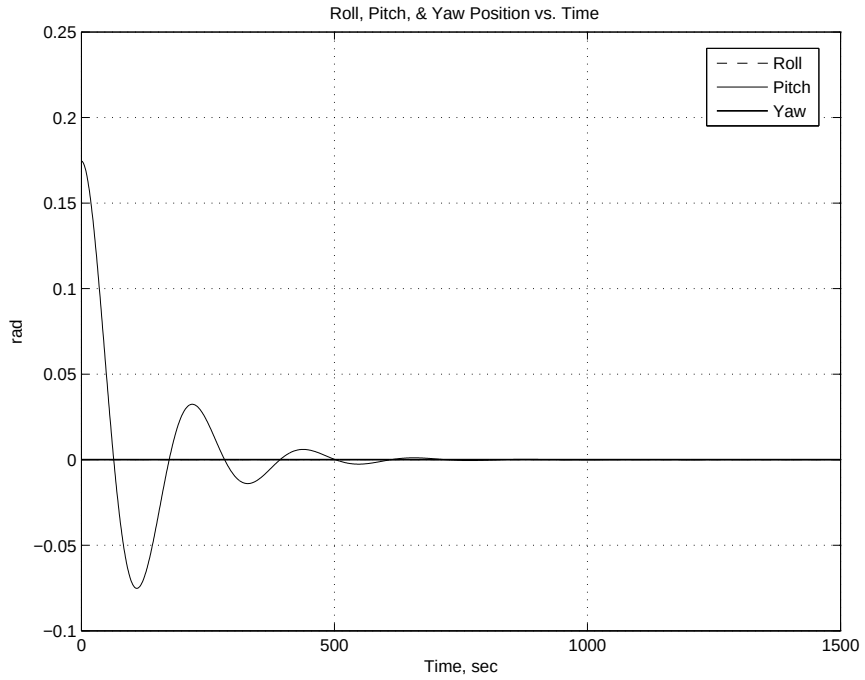


Figure E.5: Angular displacement at 200km with pitch offset 10°.

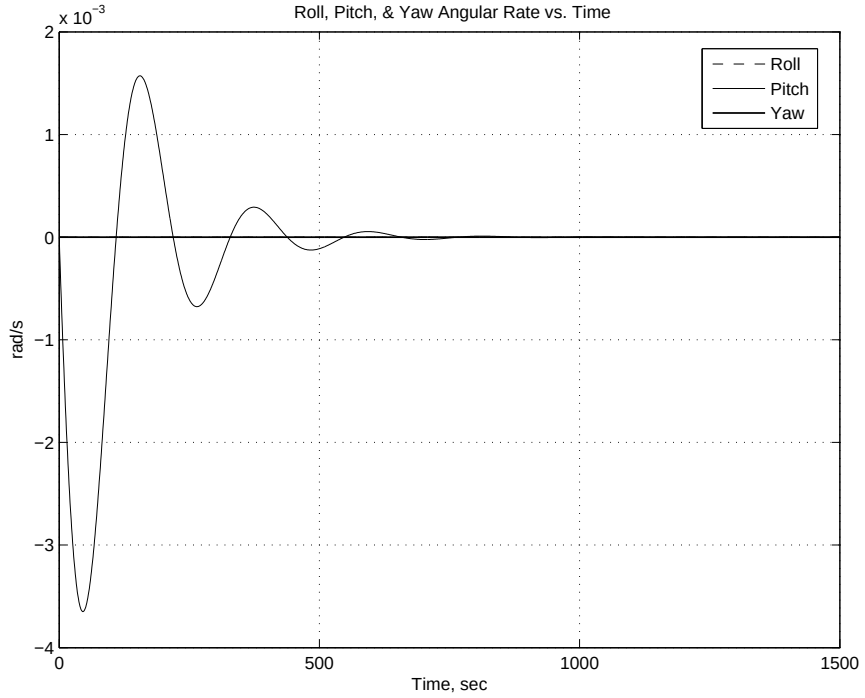


Figure E.6: Angular rate at 200km with pitch offset 10°.

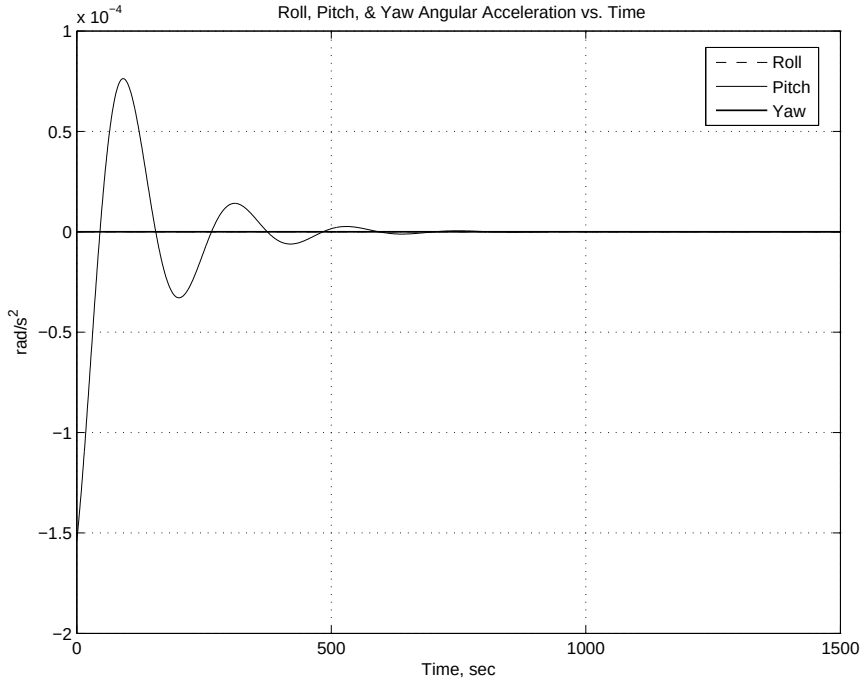


Figure E.7: Angular Acceleration at 200km with pitch offset 10°.

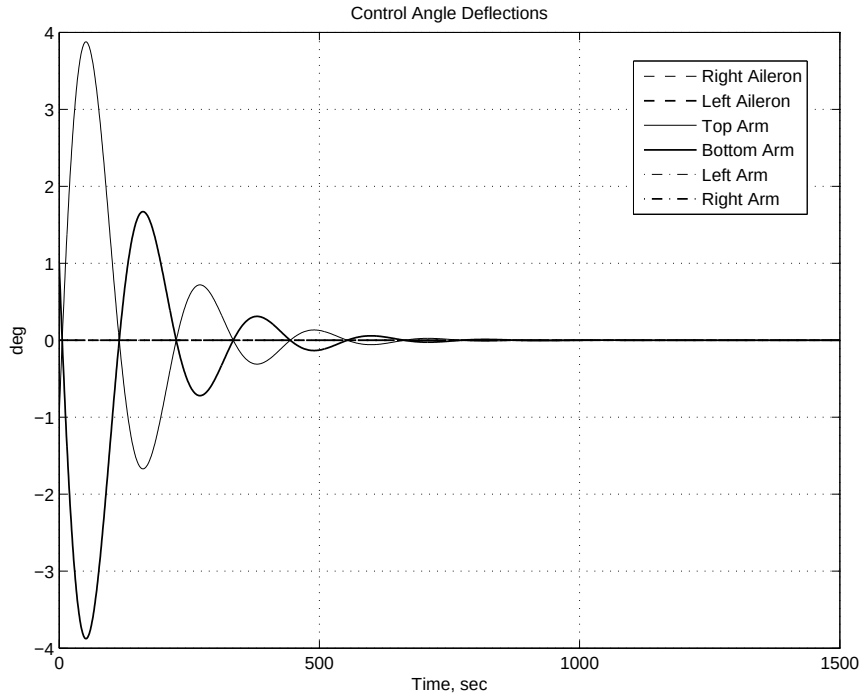


Figure E.8: Control angle deflections 200 km with pitch offset 10°.

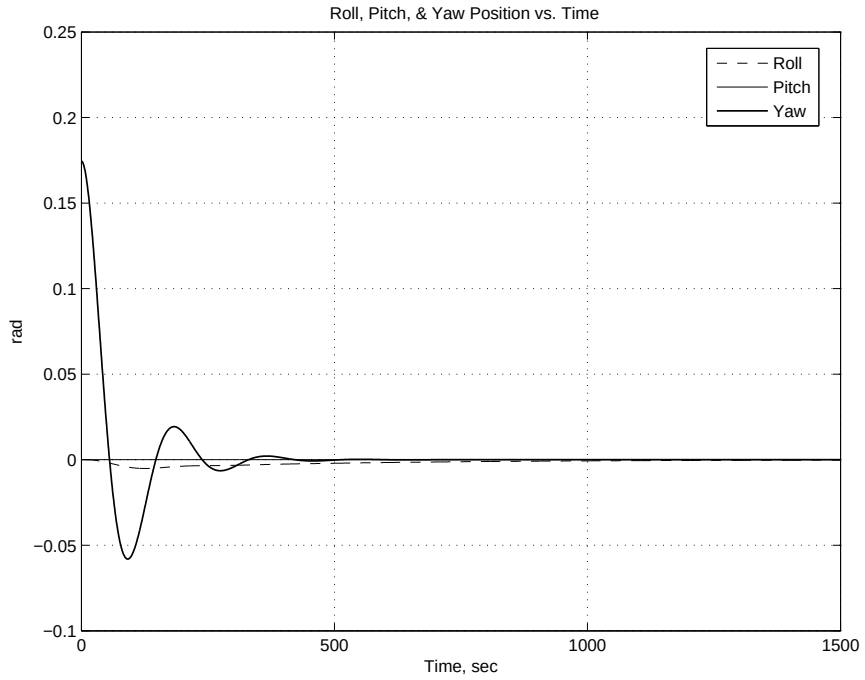


Figure E.9: Angular displacement at 200km with yaw offset 10°.

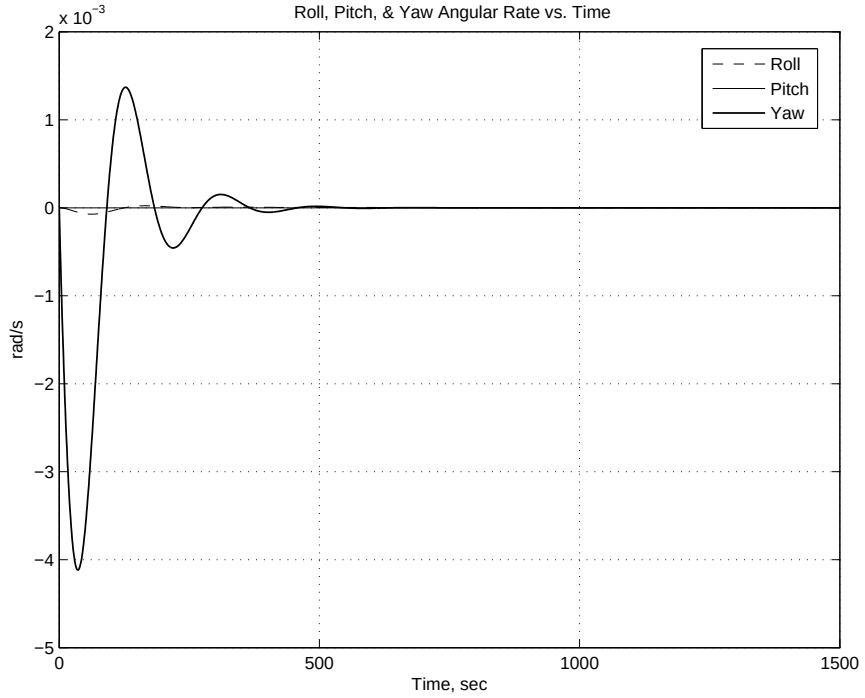


Figure E.10: Angular rate at 200km with yaw offset 10°.

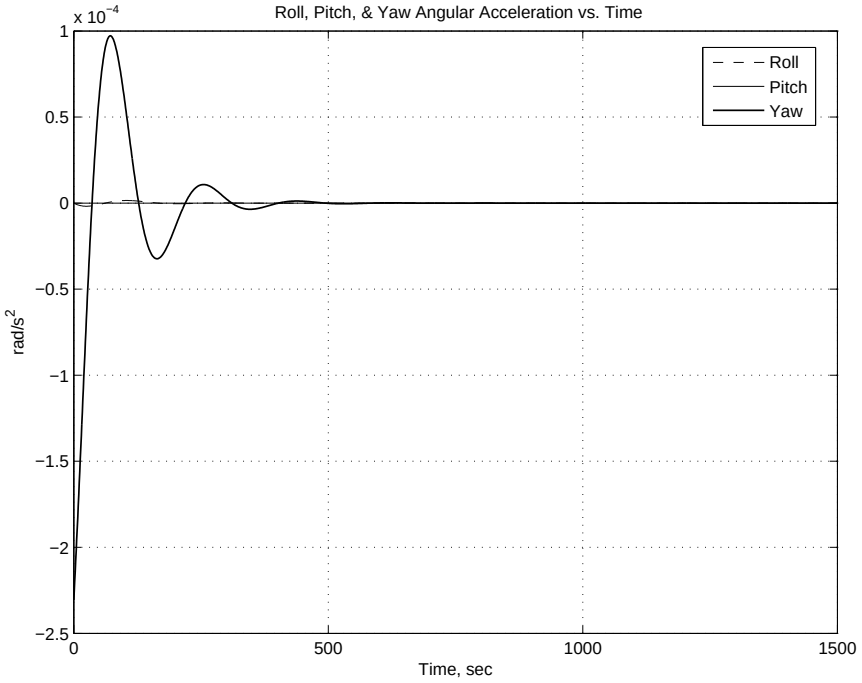


Figure E.11: Angular Acceleration at 200km with yaw offset 10°.

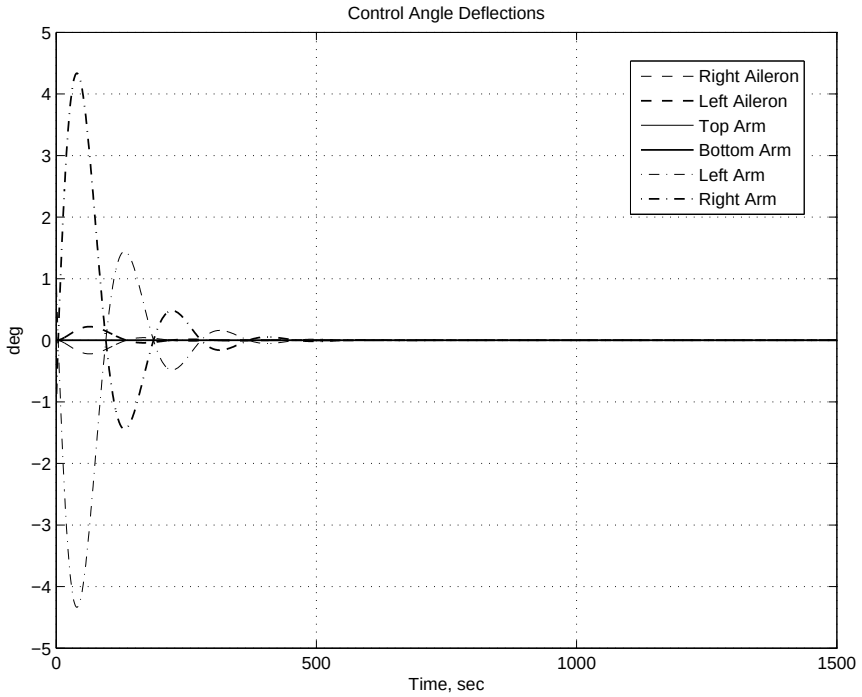


Figure E.12: Control angle deflections 200 km with yaw offset 10°.

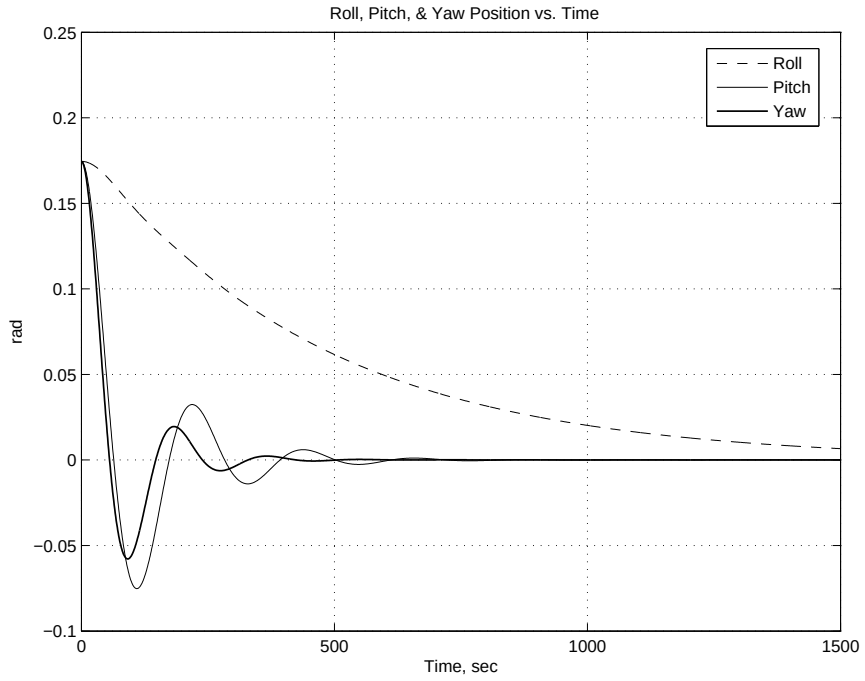


Figure E.13: Angular displacement at 200km with roll, pitch and yaw offset 10°.

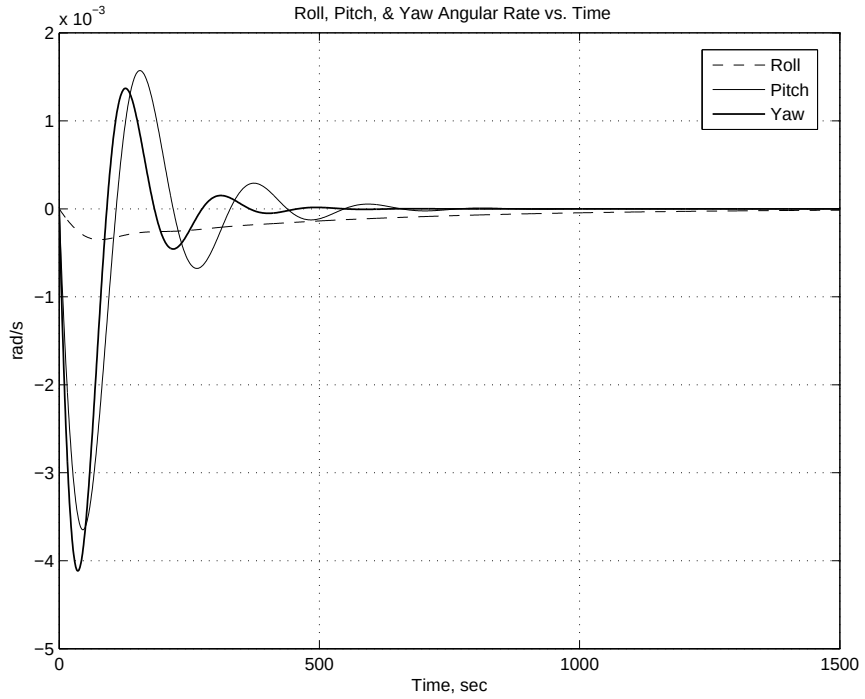


Figure E.14: Angular rate at 200km with roll, pitch and yaw offset 10°.

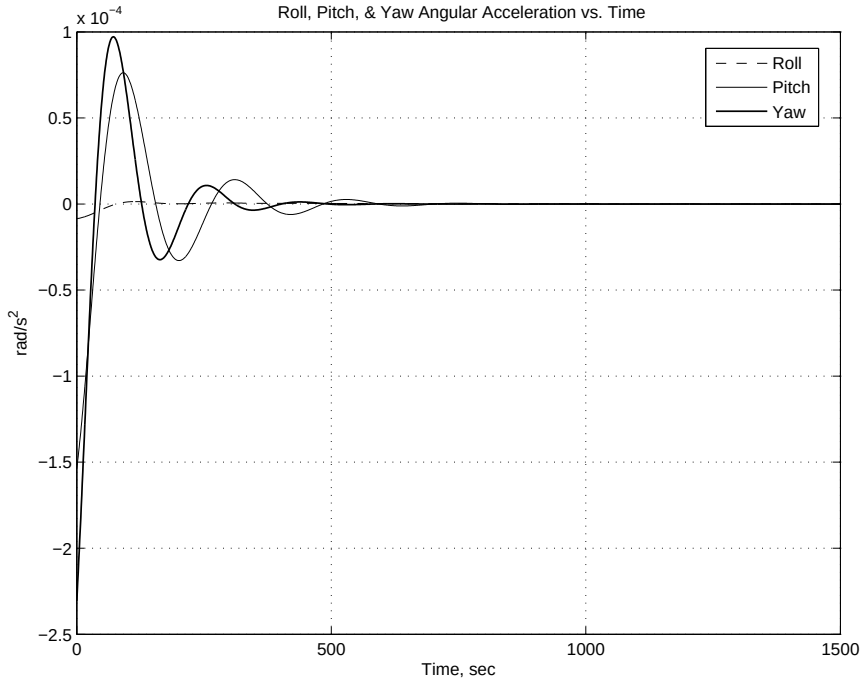


Figure E.15: Angular Acceleration at 200km with roll, pitch and yaw offset 10°.

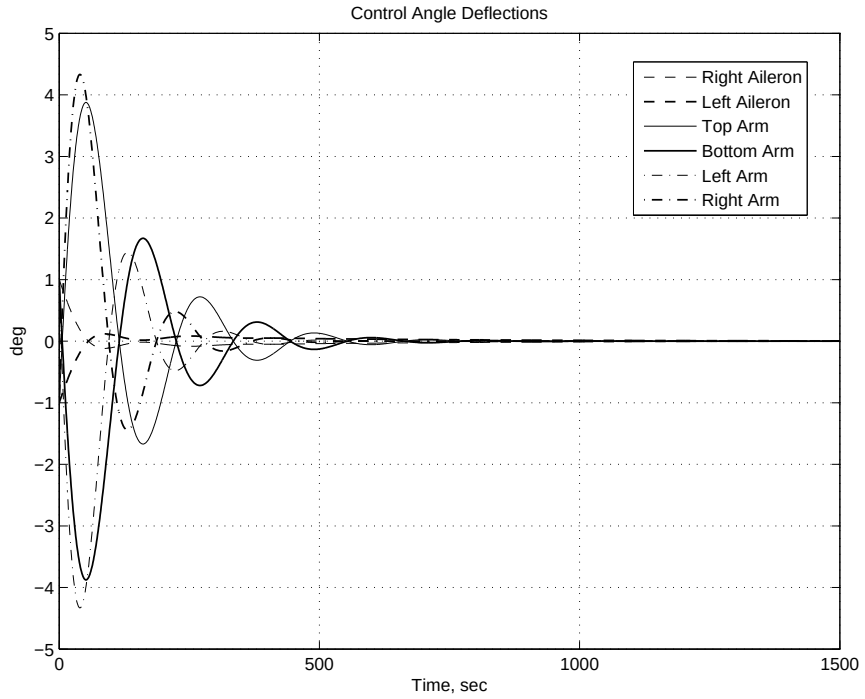


Figure E.16: Control angle deflections 200 km with roll, pitch and yaw offset 10°.

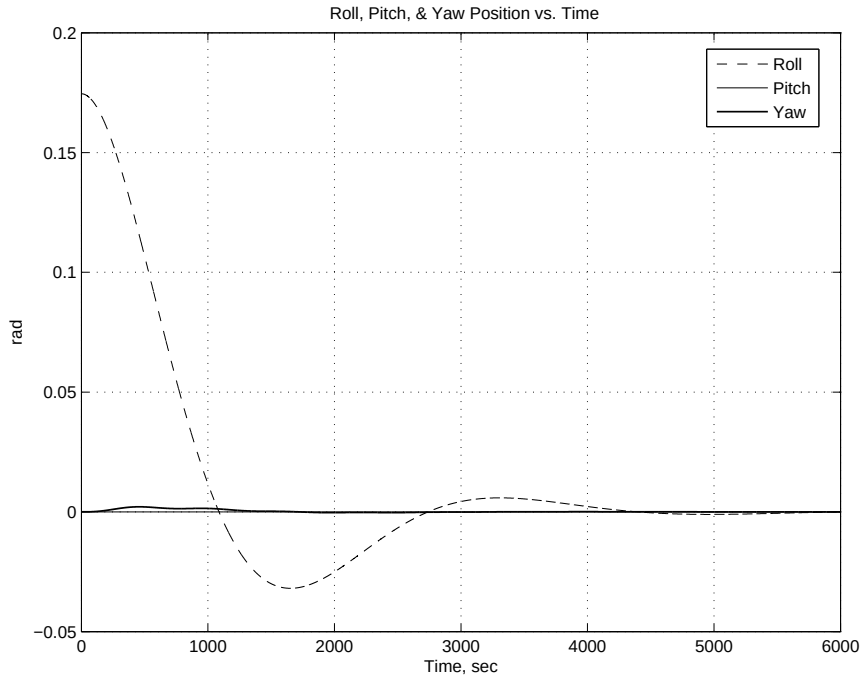


Figure E.17: Angular displacement at 300km with roll offset 10°.

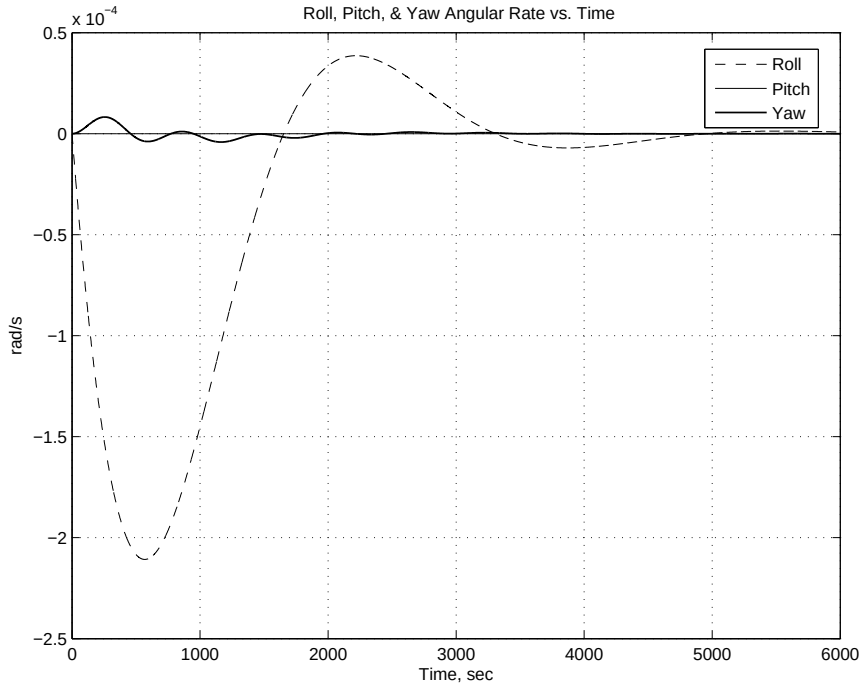


Figure E.18: Angular rate at 300km with roll offset 10°.

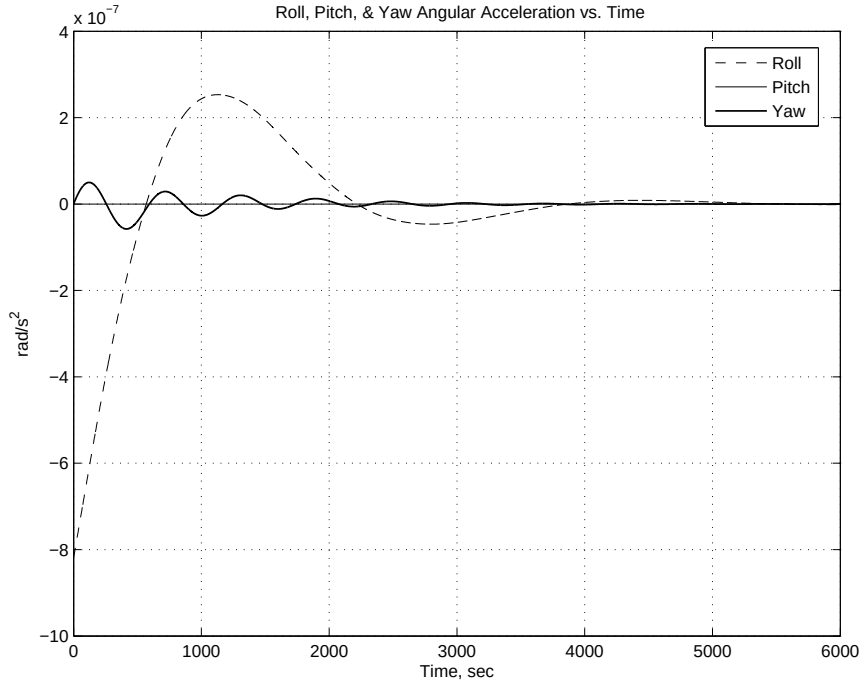


Figure E.19: Angular Acceleration at 300km with roll offset 10°.

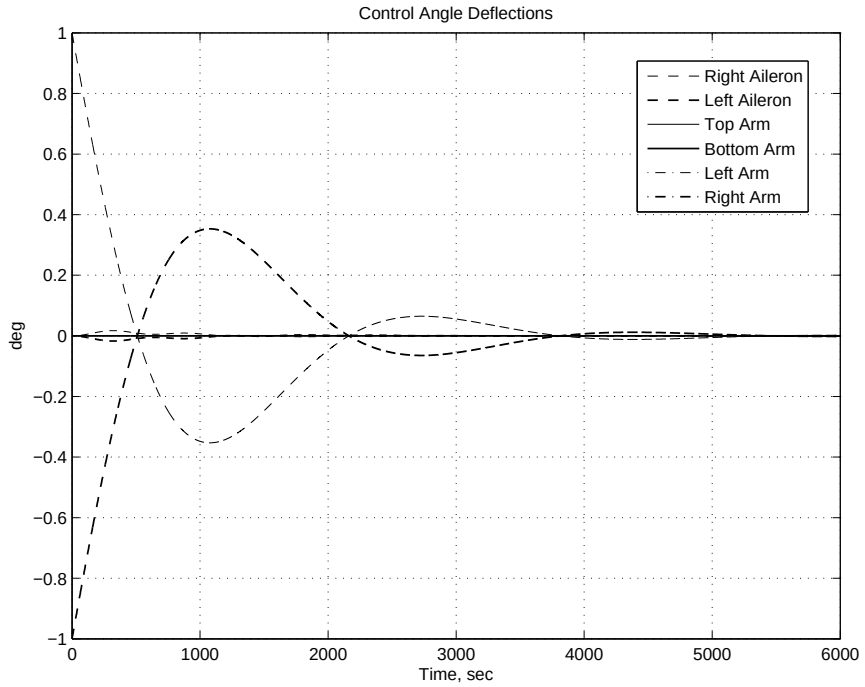


Figure E.20: Control angle deflections 300 km with roll offset 10°.

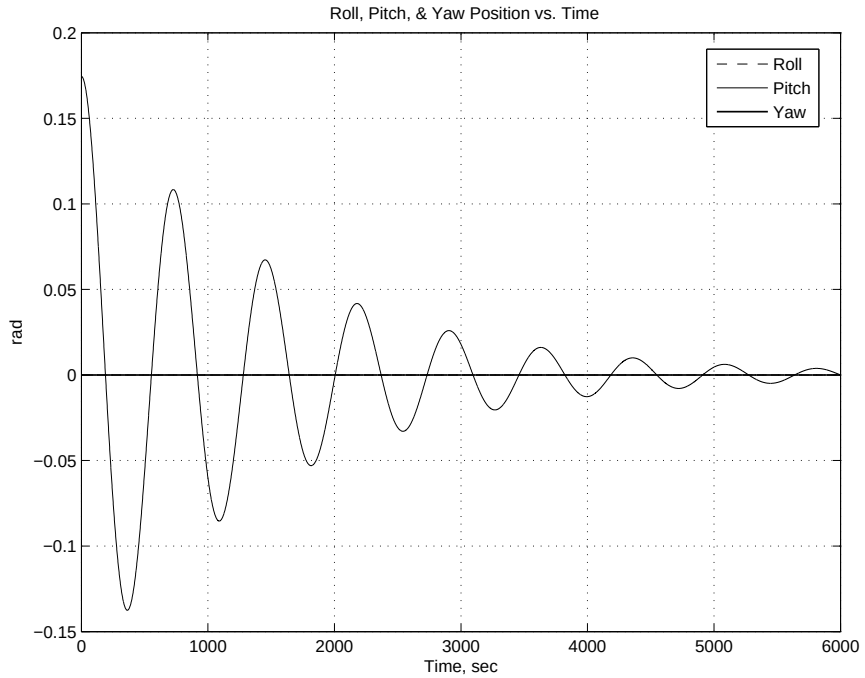


Figure E.21: Angular displacement at 300km with pitch offset 10°.

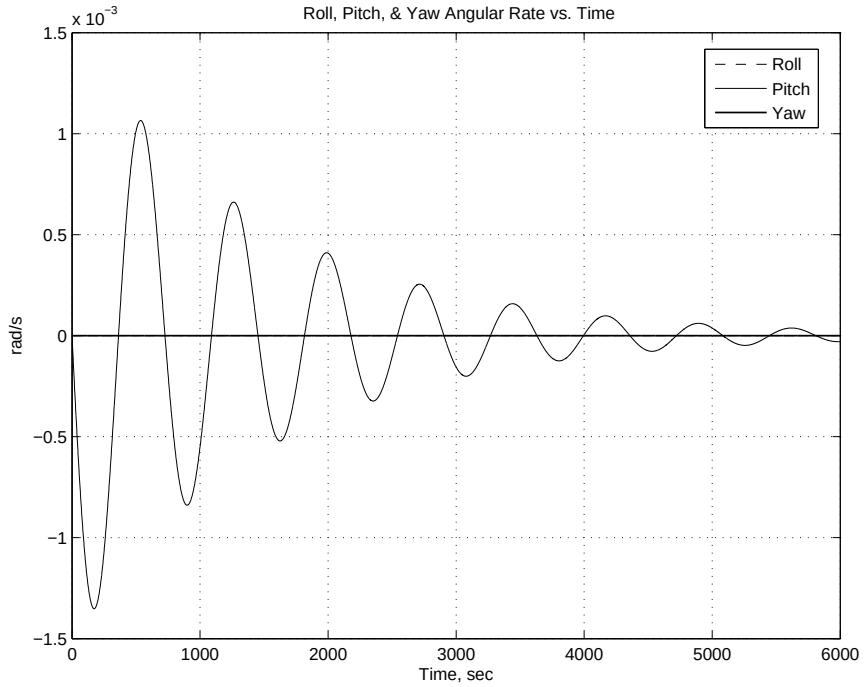


Figure E.22: Angular rate at 300km with pitch offset 10°.

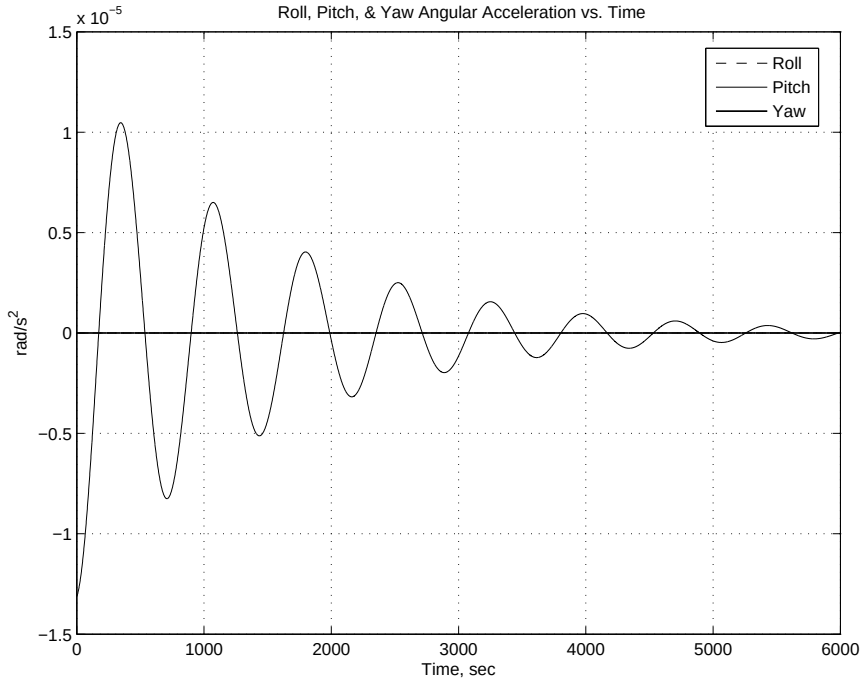


Figure E.23: Angular Acceleration at 300km with pitch offset 10°.

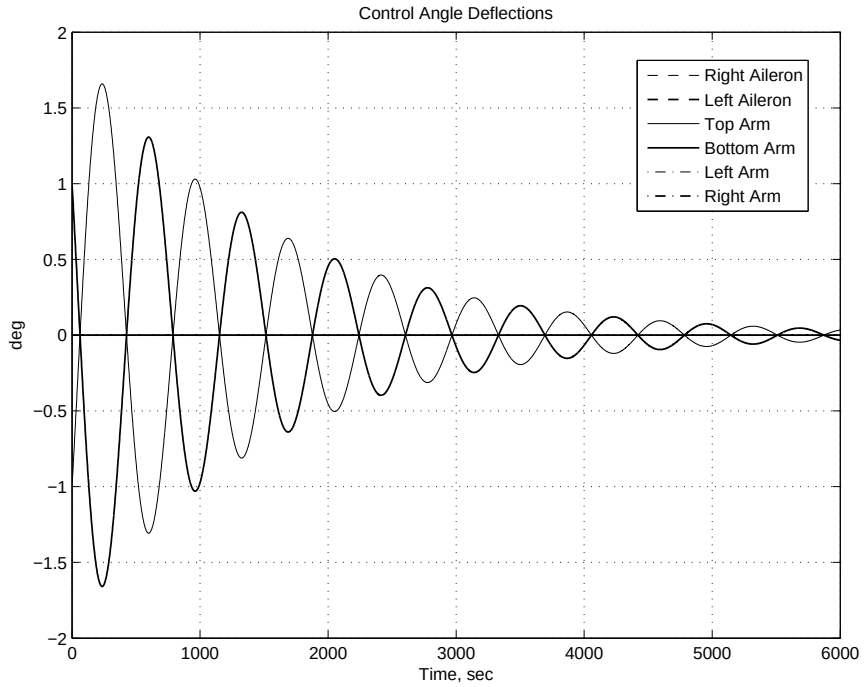


Figure E.24: Control angle deflections 300 km with pitch offset 10°.

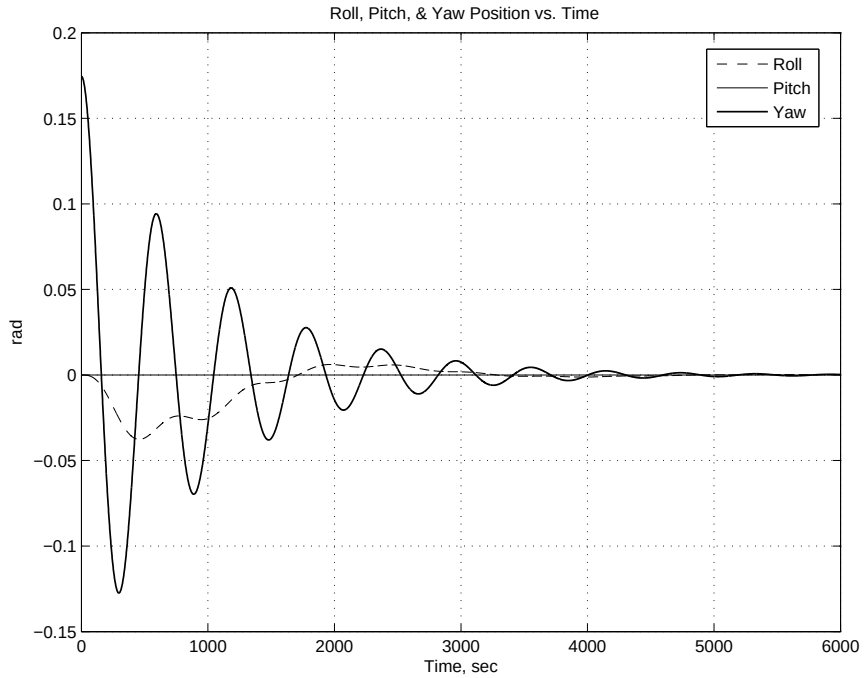


Figure E.25: Angular displacement at 300km with yaw offset 10°.

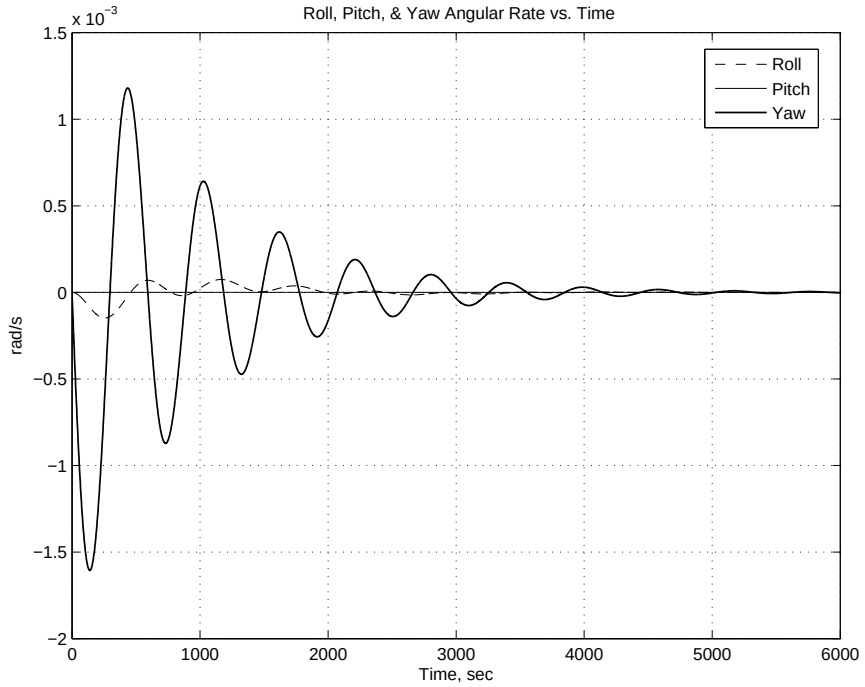


Figure E.26: Angular rate at 300km with yaw offset 10°.

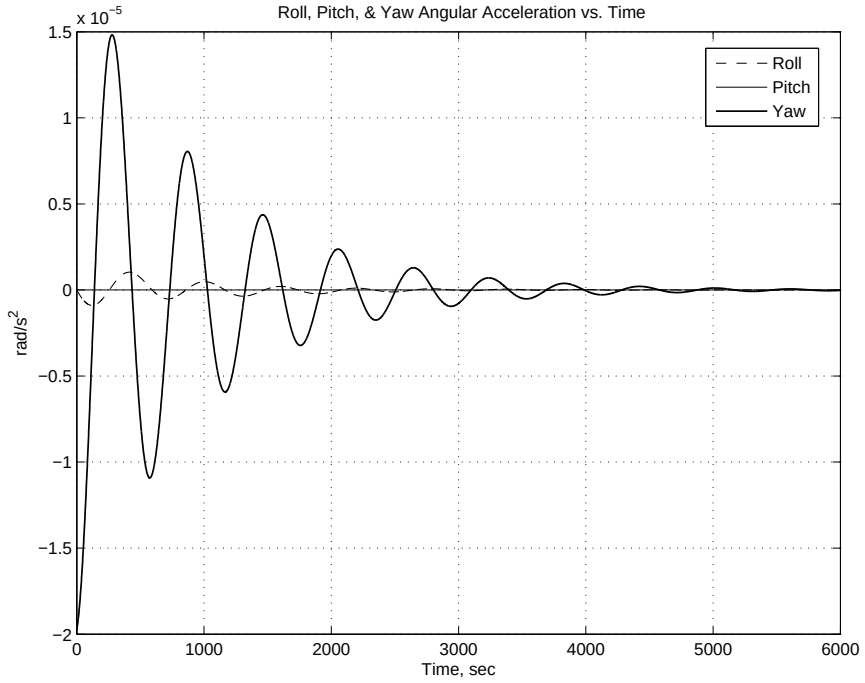


Figure E.27: Angular Acceleration at 300km with yaw offset 10°.

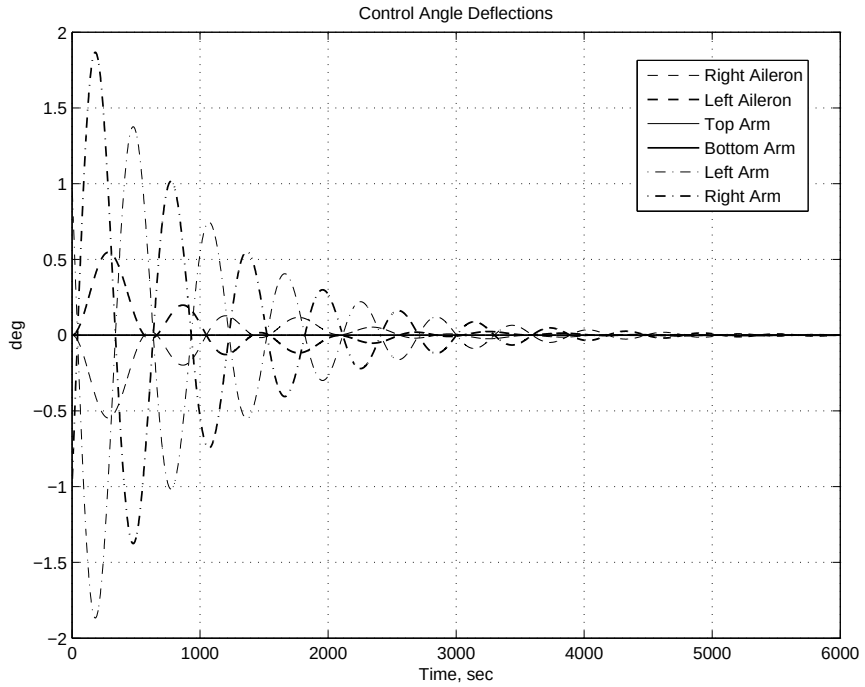


Figure E.28: Control angle deflections at 300km with yaw offset 10°.

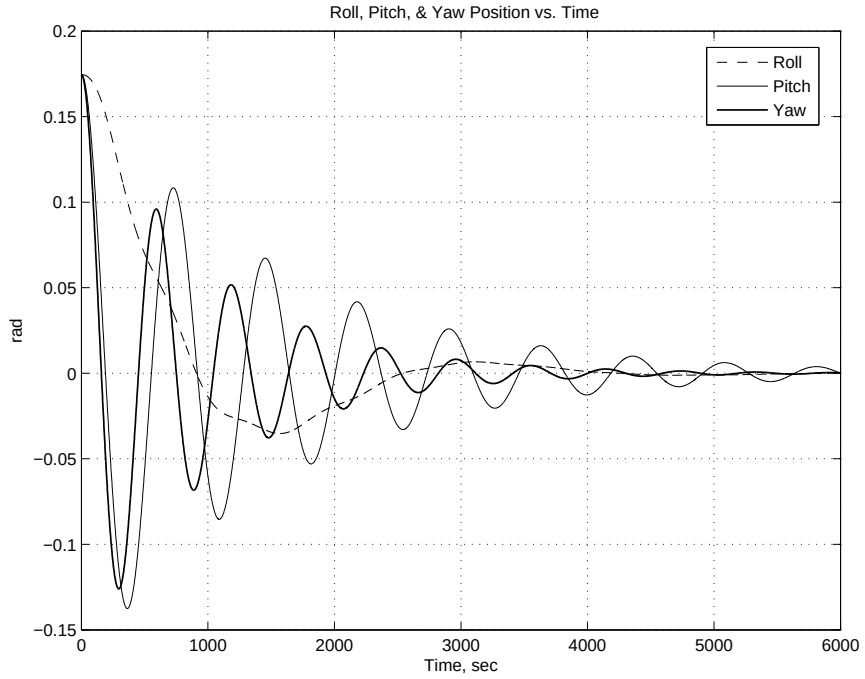


Figure E.29: Angular displacement at 300km with roll, pitch and yaw offset 10°.

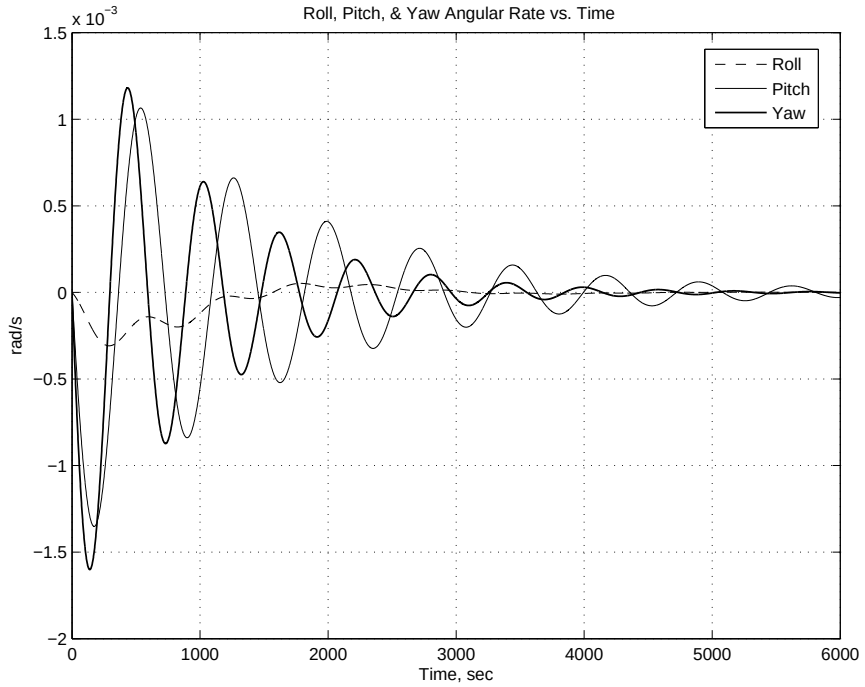


Figure E.30: Angular rate at 300km with roll, pitch and yaw offset 10°.

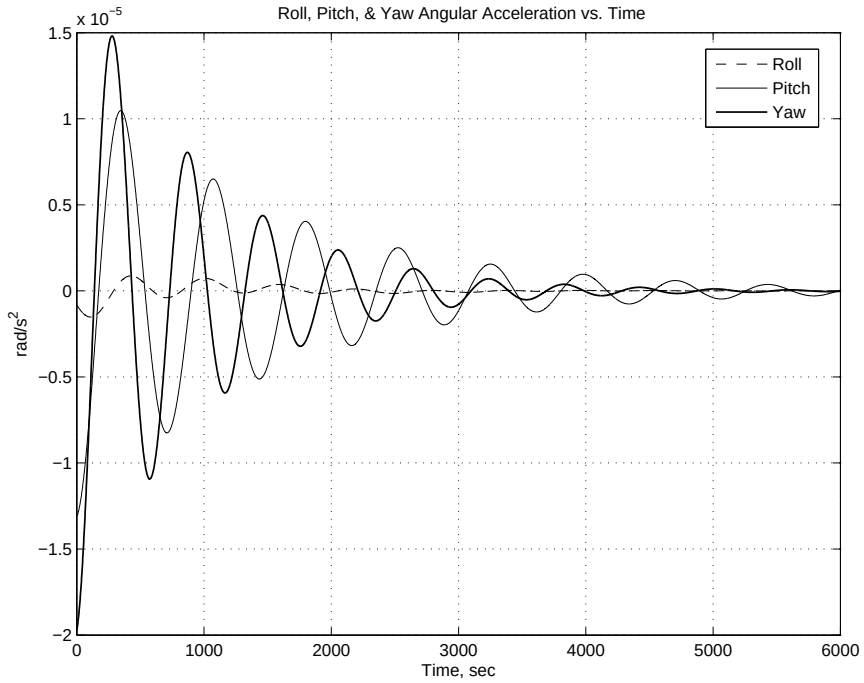


Figure E.31: Angular Acceleration at 300km with roll, pitch and yaw offset 10°.

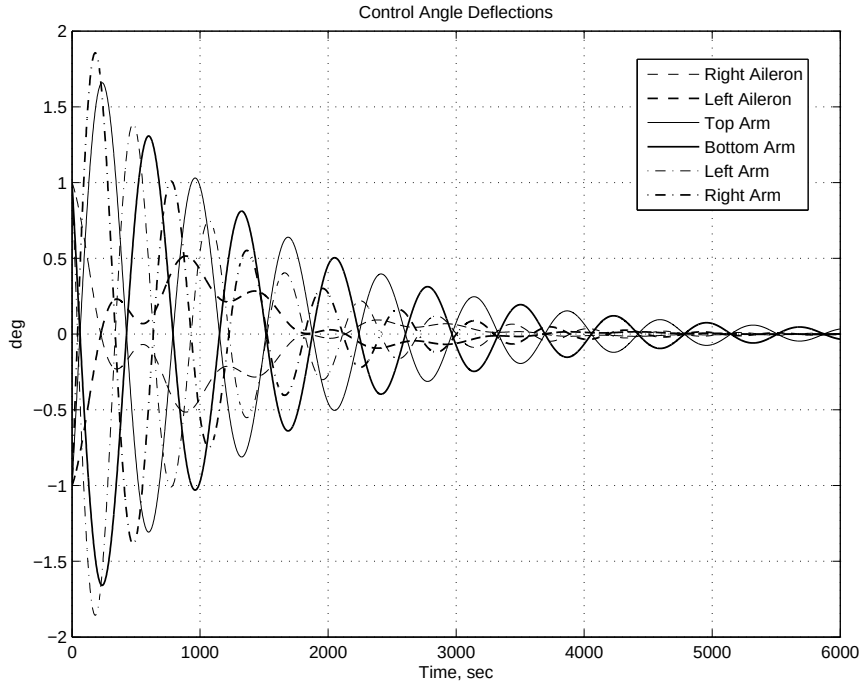


Figure E.32: Control angle deflections at 300km with roll, pitch and yaw offset 10°.

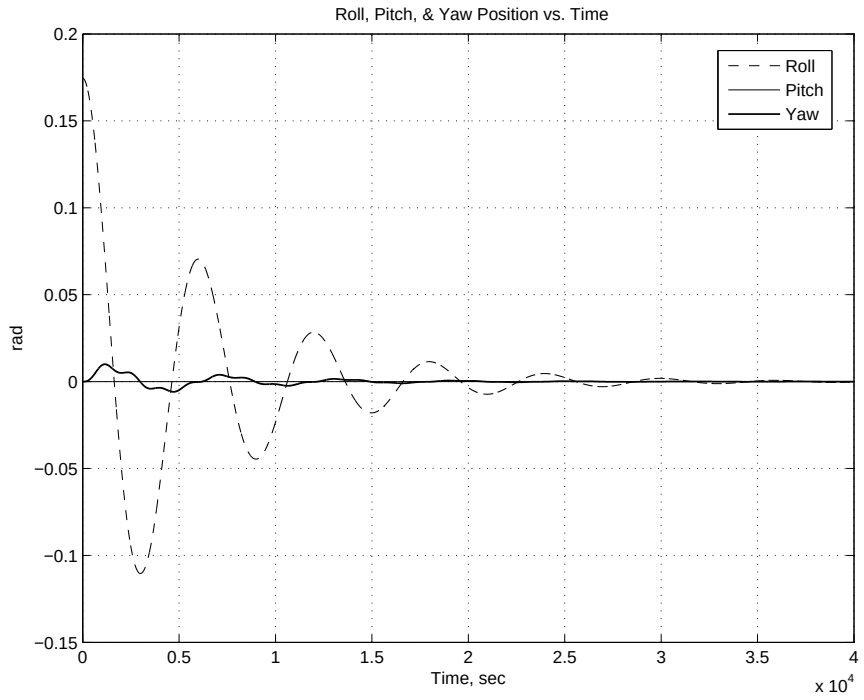


Figure E.33: Angular displacement at 400km with roll offset 10°.

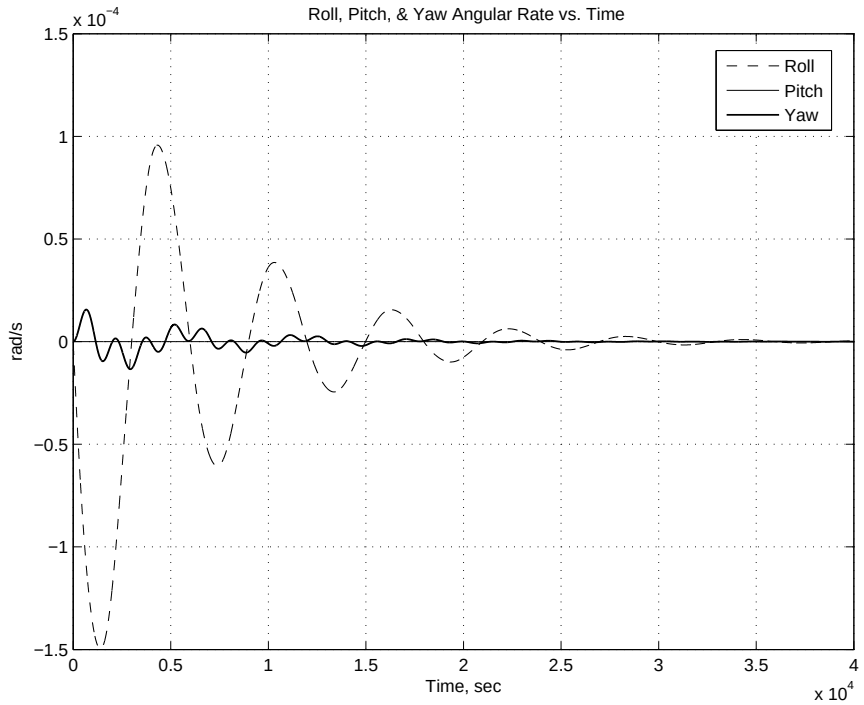


Figure E.34: Angular rate at 400km with roll offset 10°.

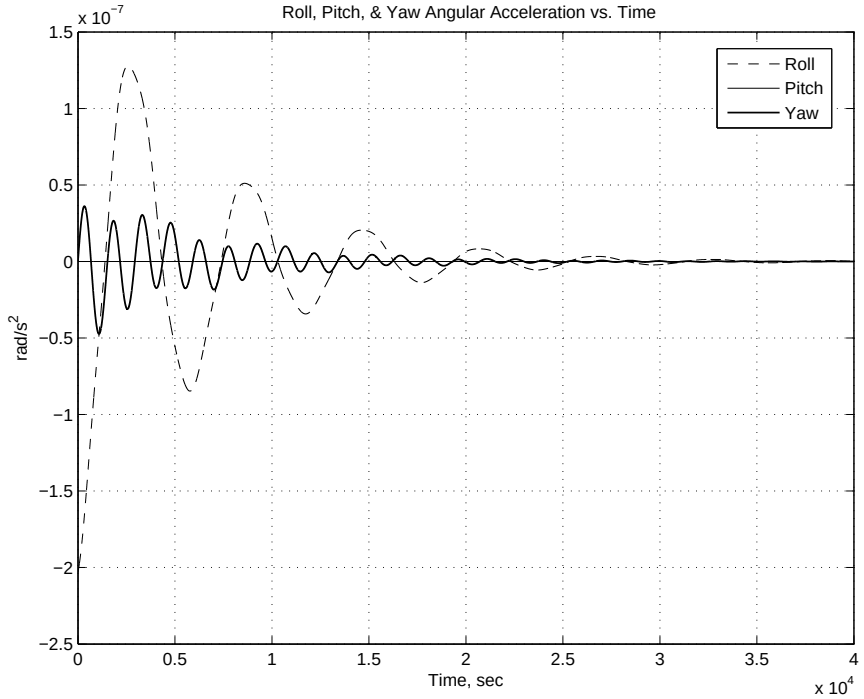


Figure E.35: Angular Acceleration at 400km with roll offset 10°.

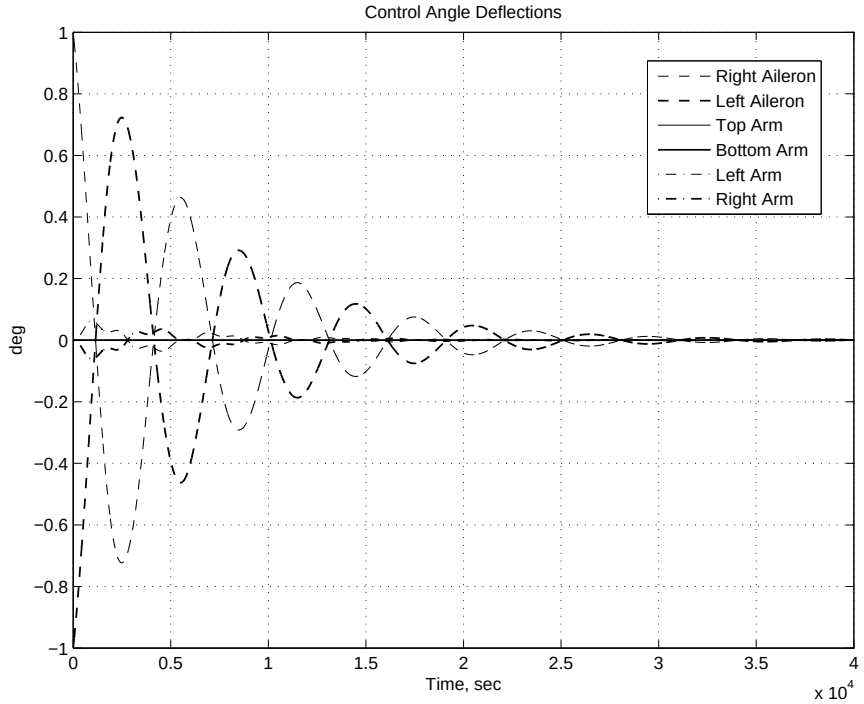


Figure E.36: Control angle deflections 400 km with roll offset 10°.

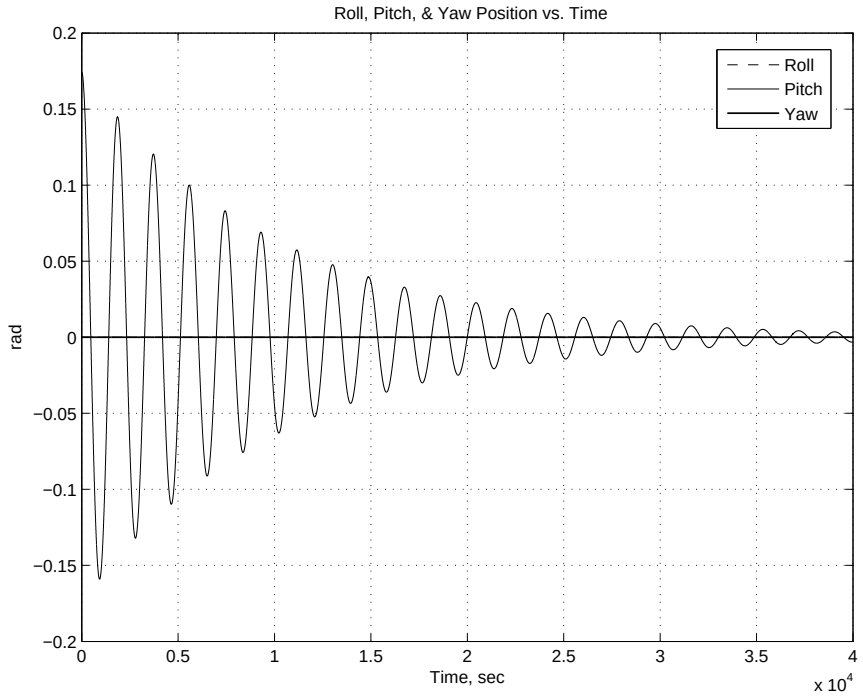


Figure E.37: Angular displacement at 400km with pitch offset 10°.

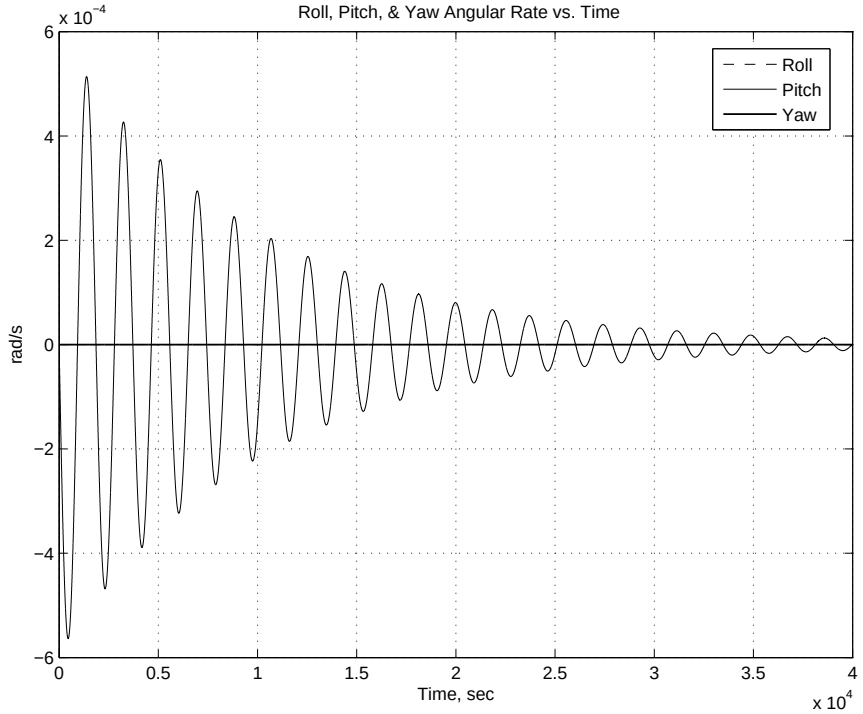


Figure E.38: Angular rate at 400km with pitch offset 10°.

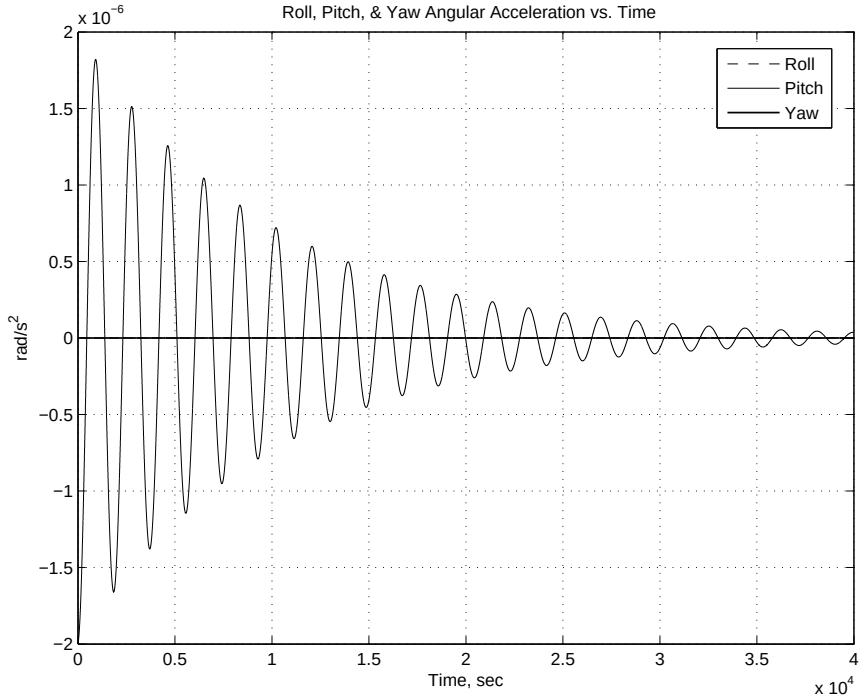


Figure E.39: Angular Acceleration at 400km with pitch offset 10°.

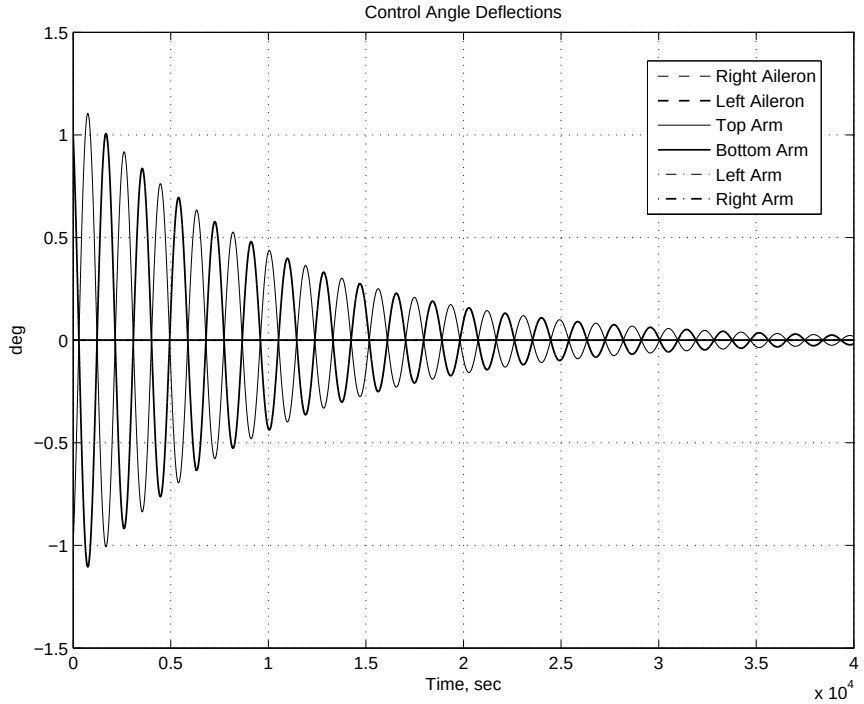


Figure E.40: Control angle deflections 400 km with pitch offset 10°.

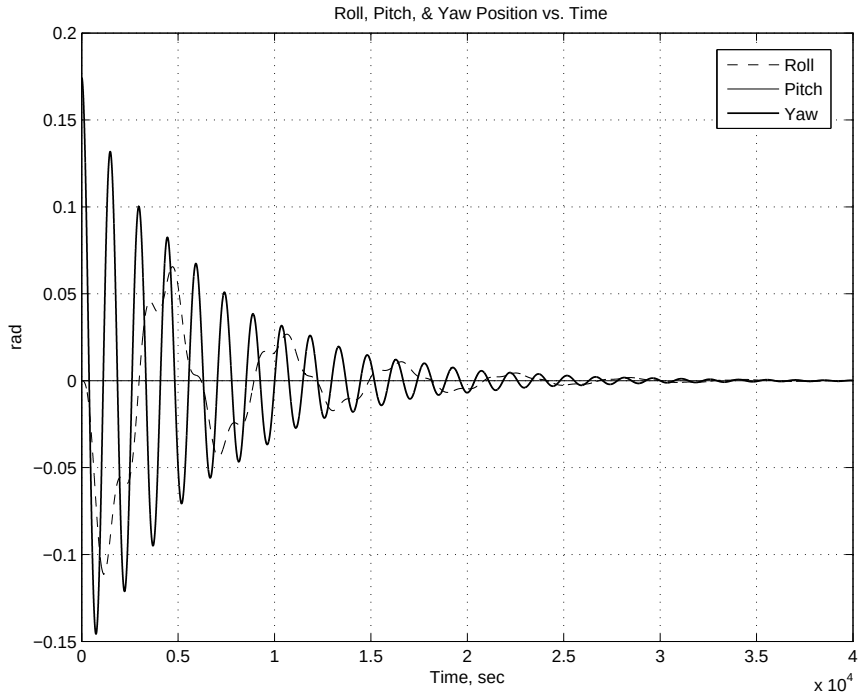


Figure E.41: Angular displacement at 400km with yaw offset 10°.

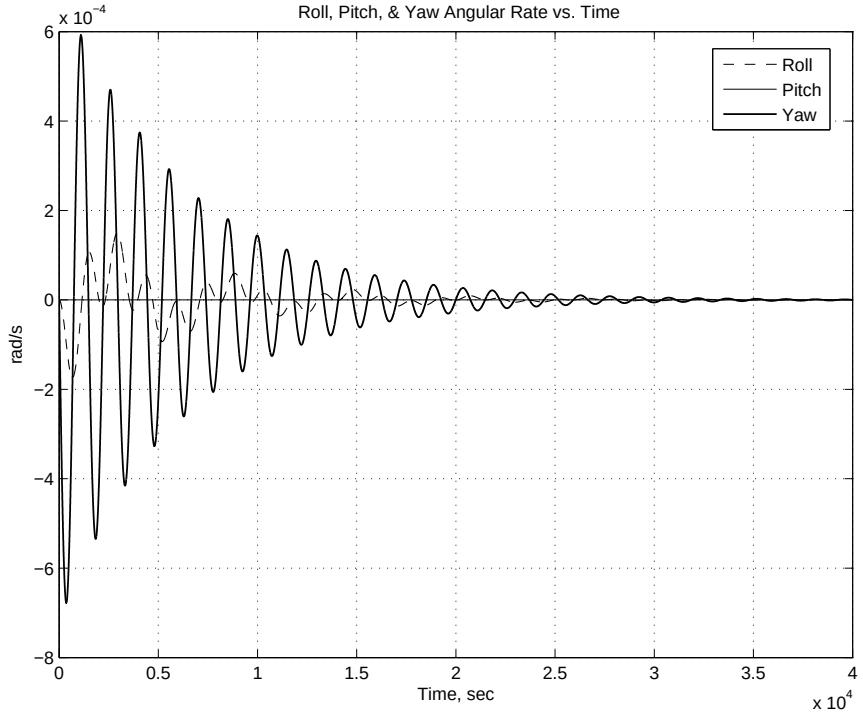


Figure E.42: Angular rate at 400km with yaw offset 10°.

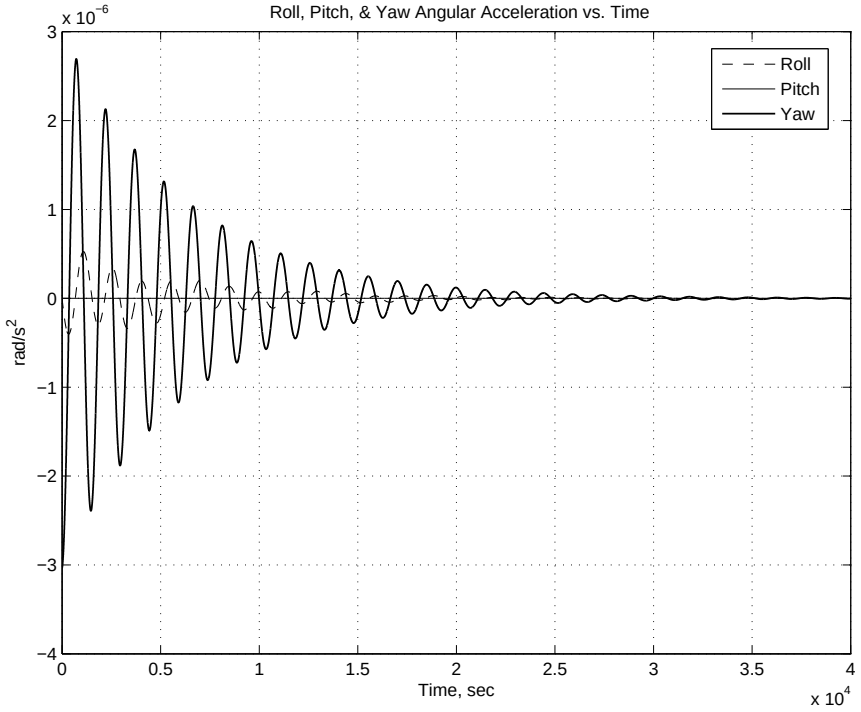


Figure E.43: Angular Acceleration at 400km with yaw offset 10°.

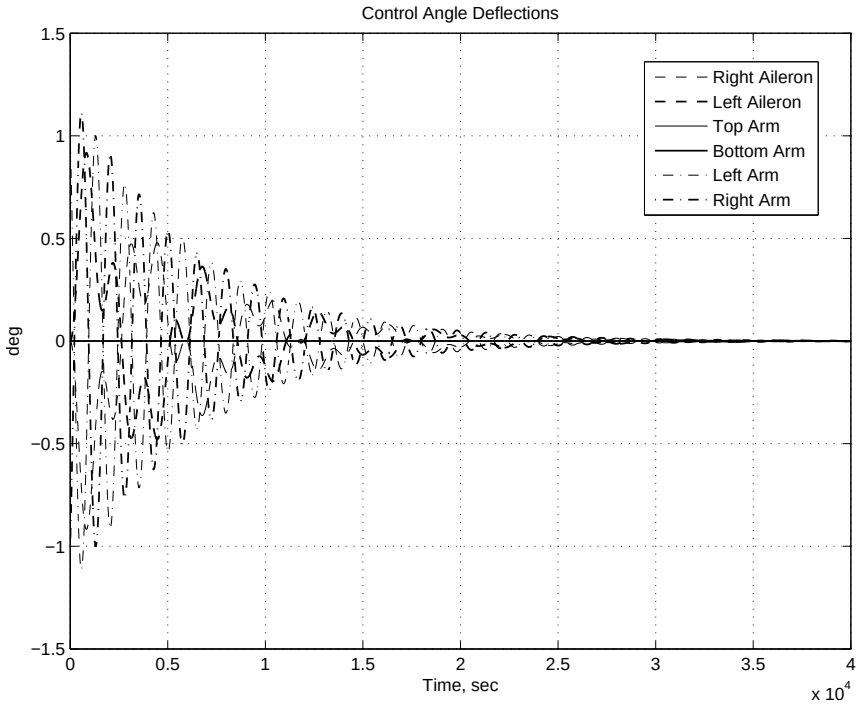


Figure E.44: Control angle deflections 400 km with yaw offset 10°.

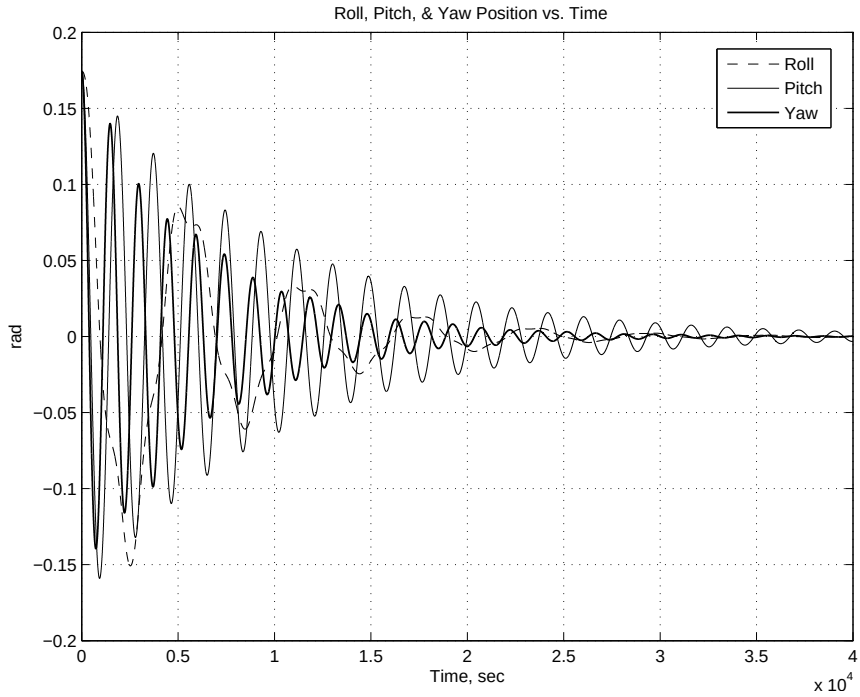


Figure E.45: Angular displacement at 400km with roll, pitch and yaw offset 10°.

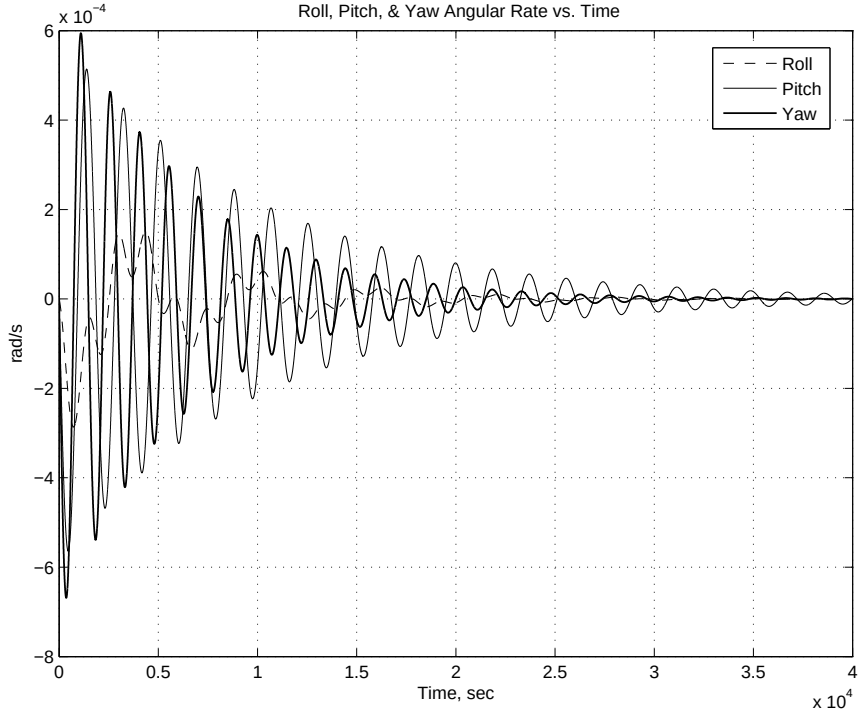


Figure E.46: Angular rate at 400km with roll, pitch and yaw offset 10°.

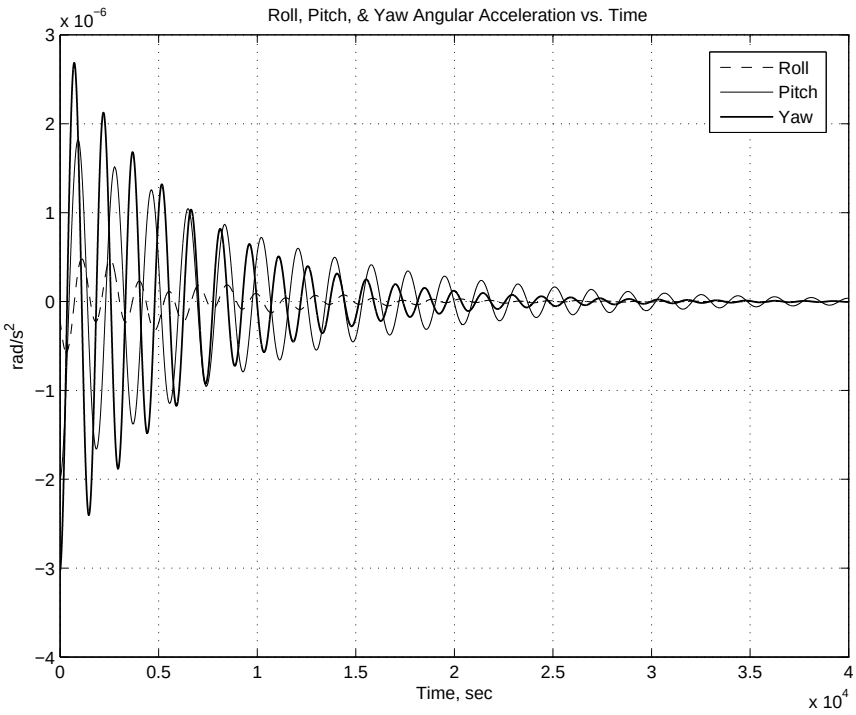


Figure E.47: Angular Acceleration at 400km with roll, pitch and yaw offset 10°.

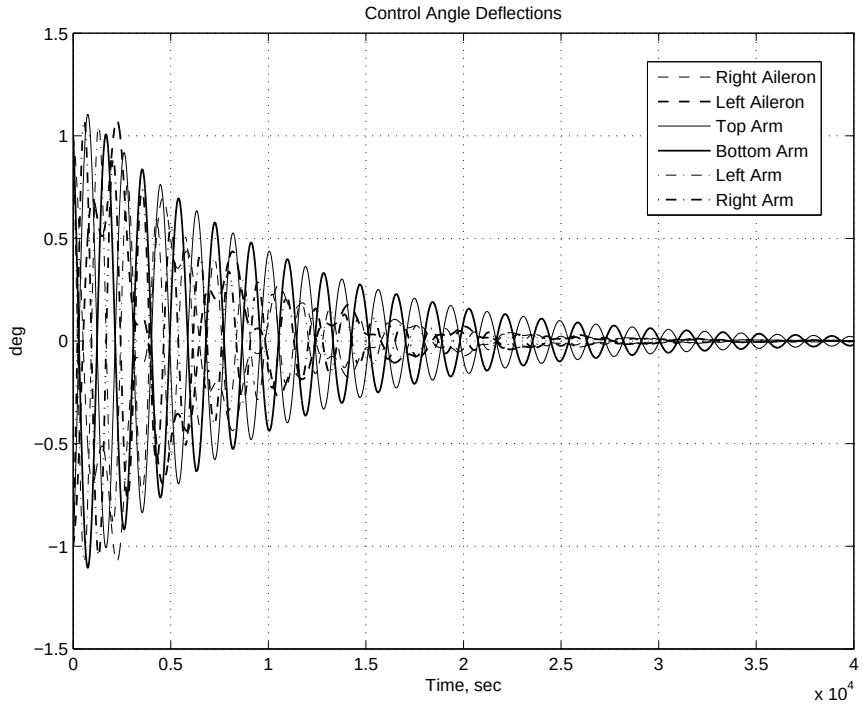


Figure E.48: Control angle deflections 400 km with roll, pitch and yaw offset 10°.

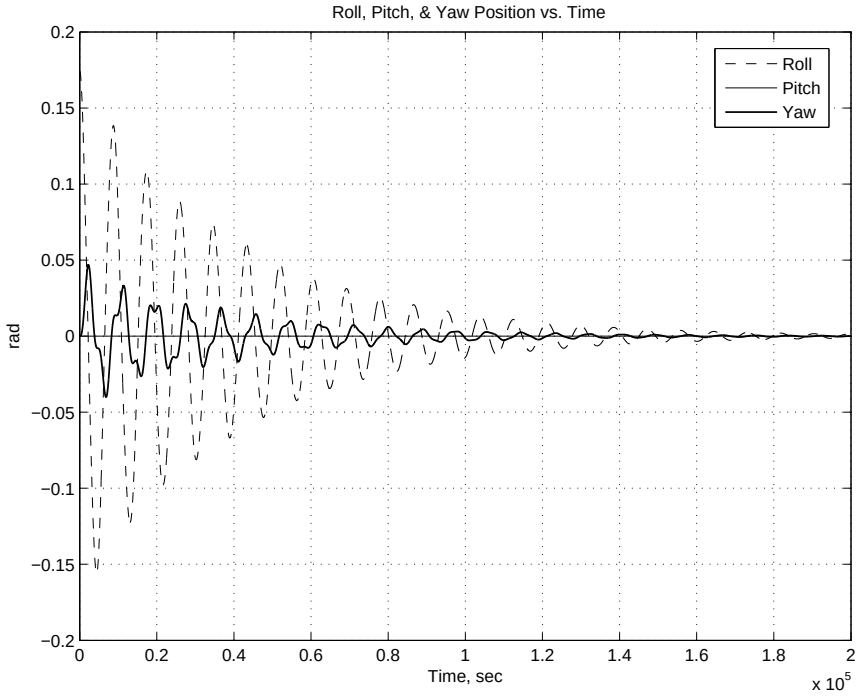


Figure E.49: Angular displacement at 500km with roll offset 10°.

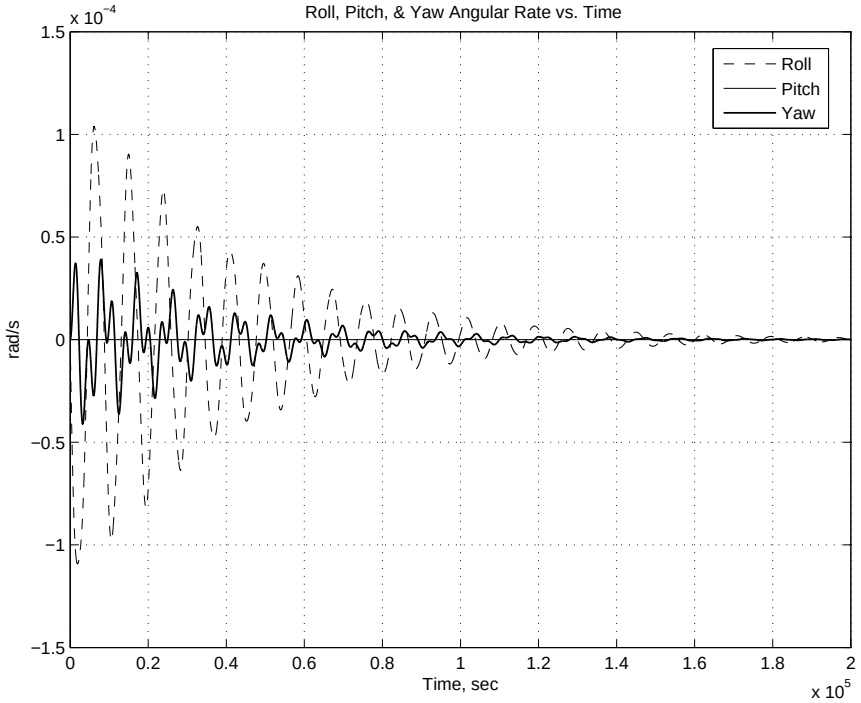


Figure E.50: Angular rate at 500km with roll offset 10°.

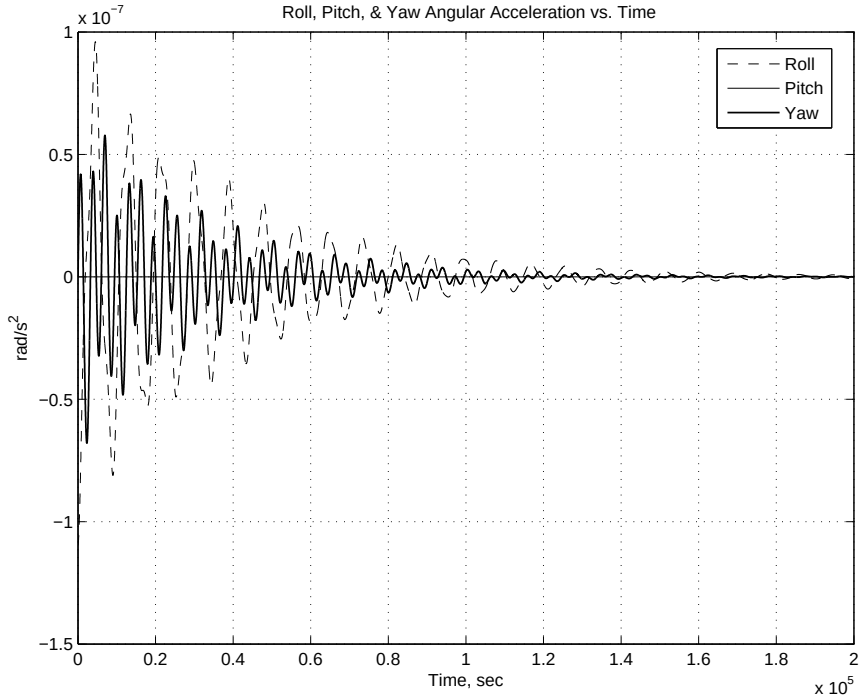


Figure E.51: Angular Acceleration at 500km with roll offset 10°.

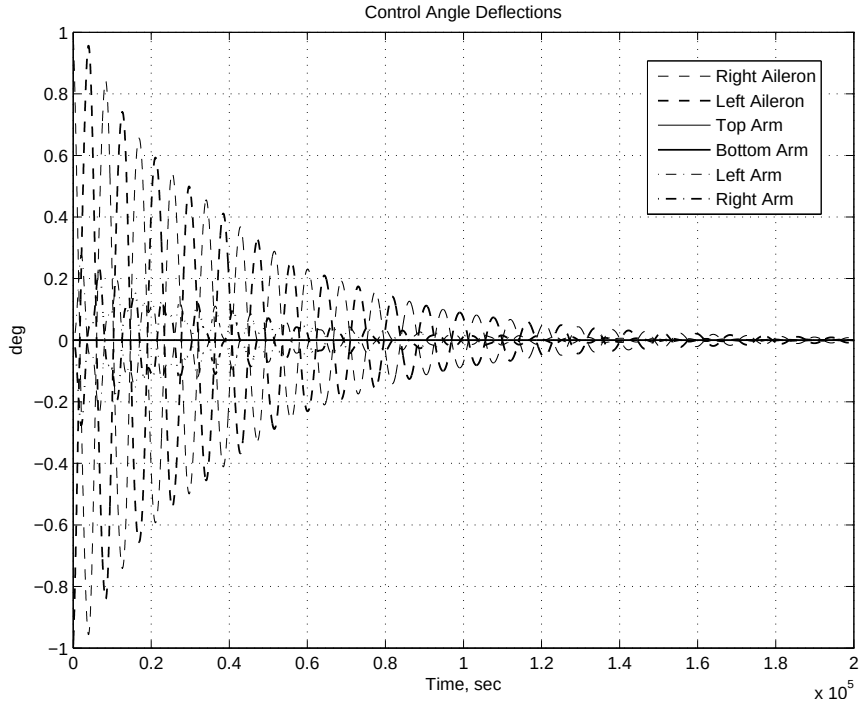


Figure E.52: Control angle deflections 500 km with roll offset 10°.

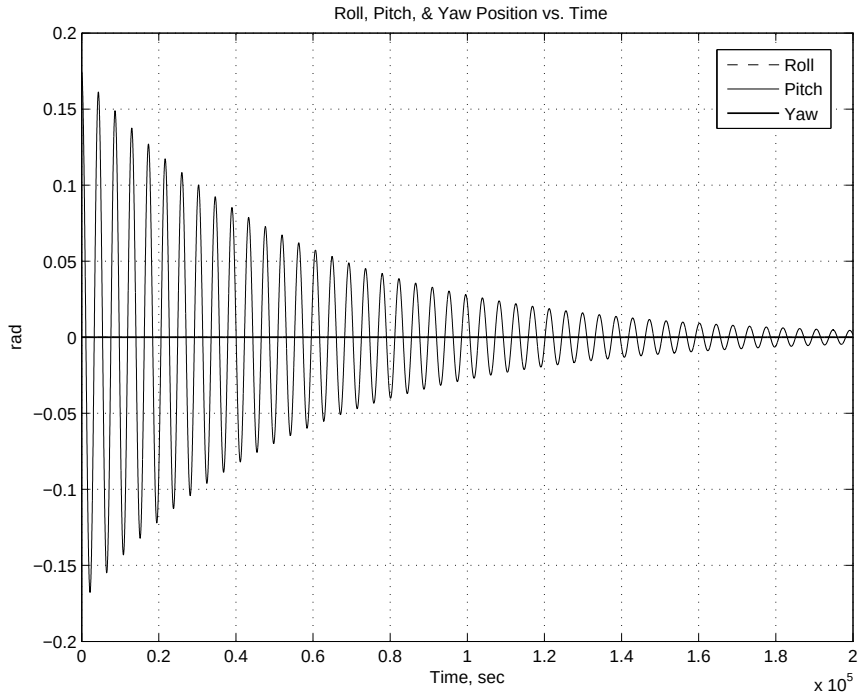


Figure E.53: Angular displacement at 500km with pitch offset 10°.

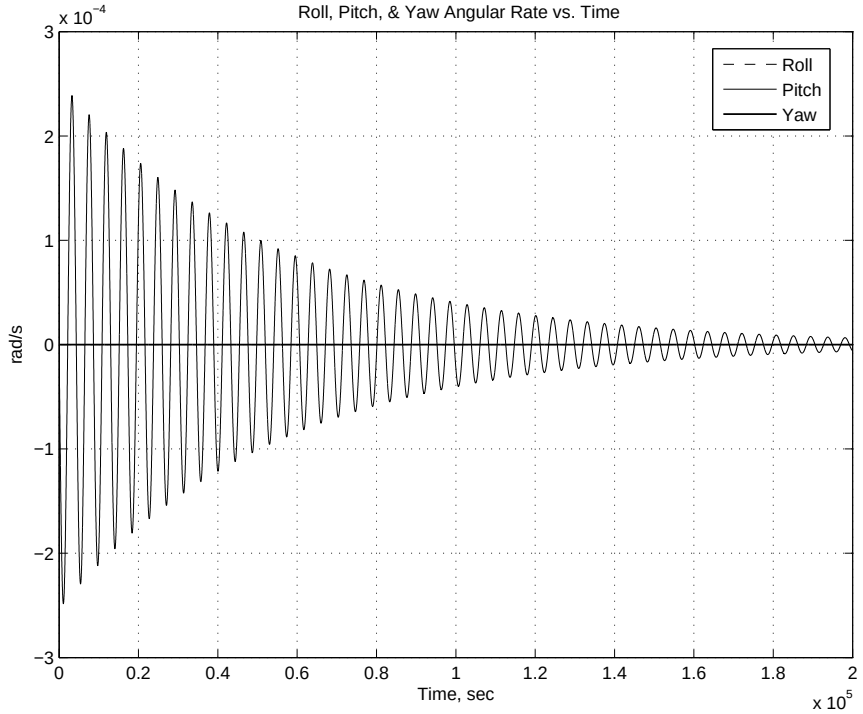


Figure E.54: Angular rate at 500km with pitch offset 10°.

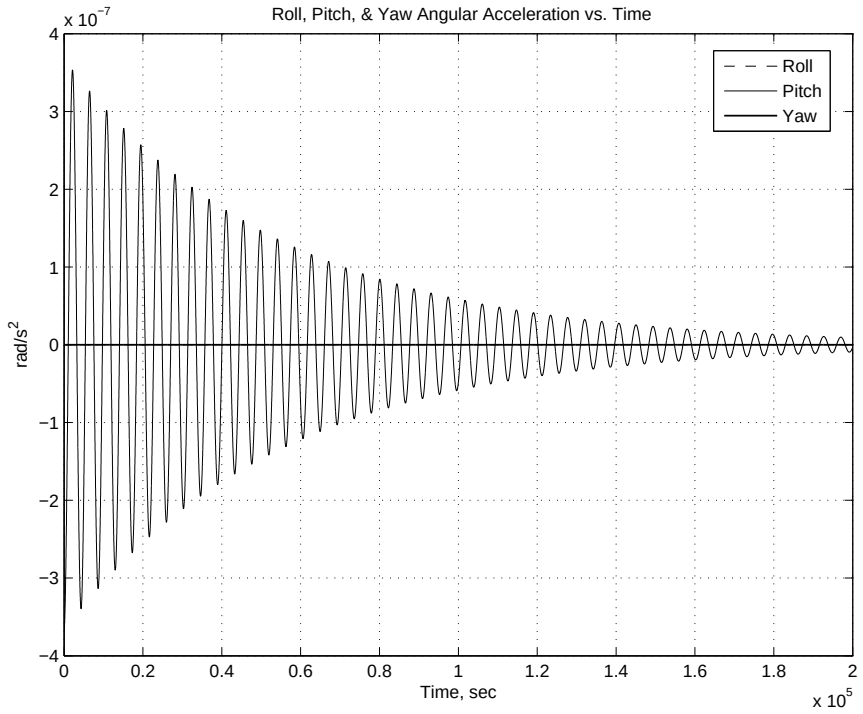


Figure E.55: Angular Acceleration at 500km with pitch offset 10°.

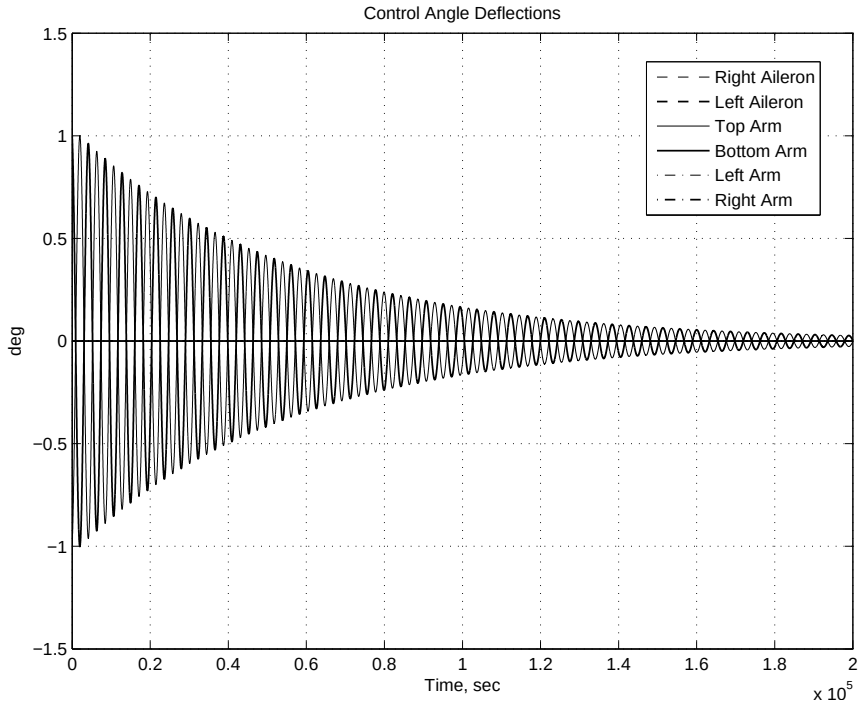


Figure E.56: Control angle deflections 500 km with pitch offset 10°.

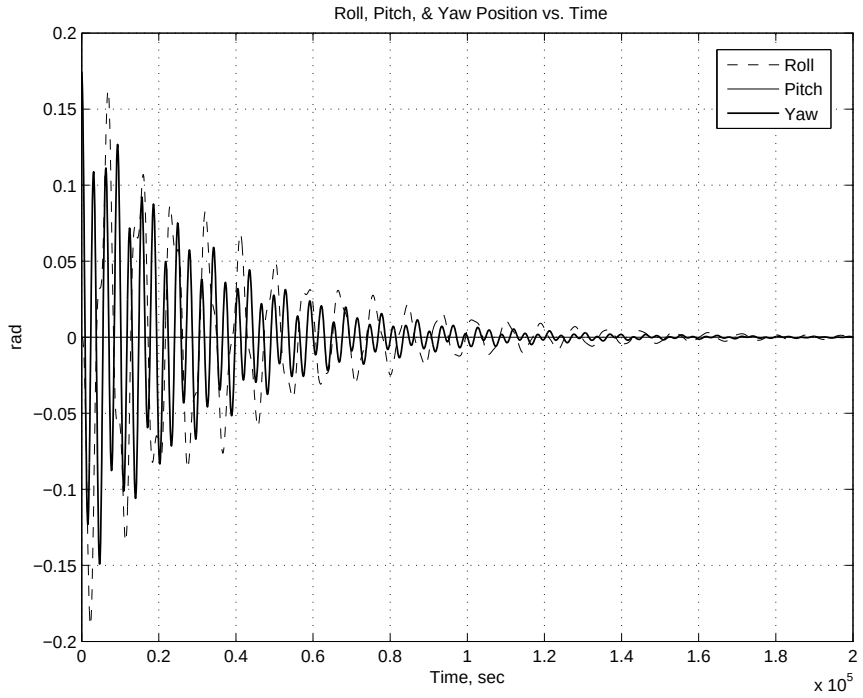


Figure E.57: Angular displacement at 500km with yaw offset 10°.

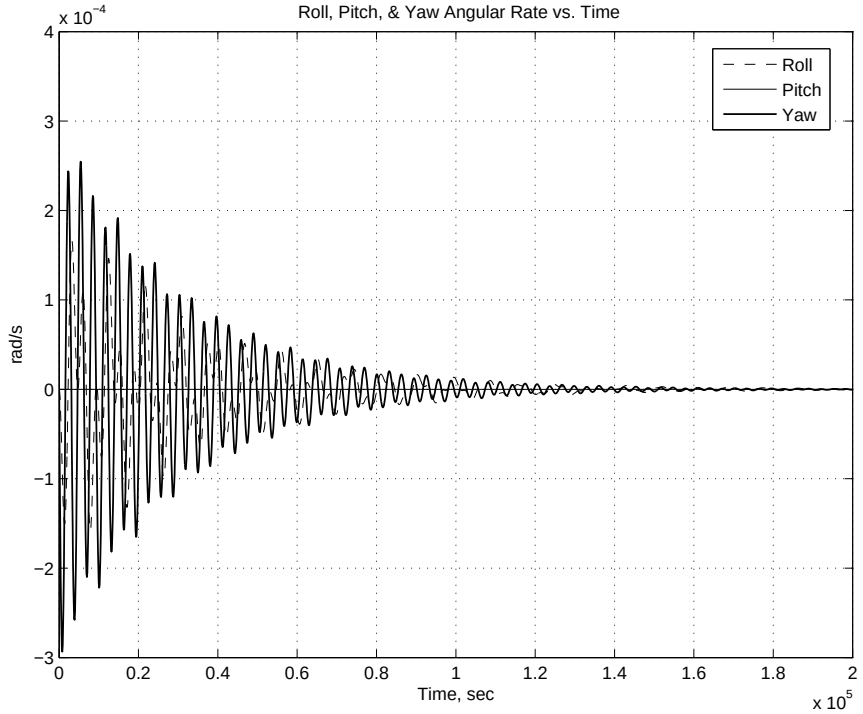


Figure E.58: Angular rate at 500km with yaw offset 10°.

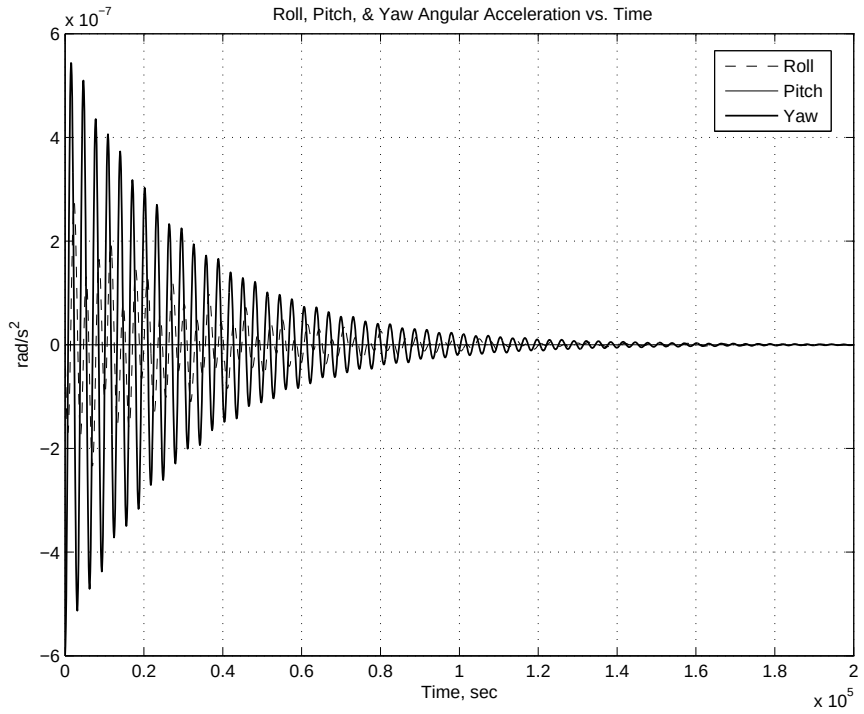


Figure E.59: Angular Acceleration at 500km with yaw offset 10°.

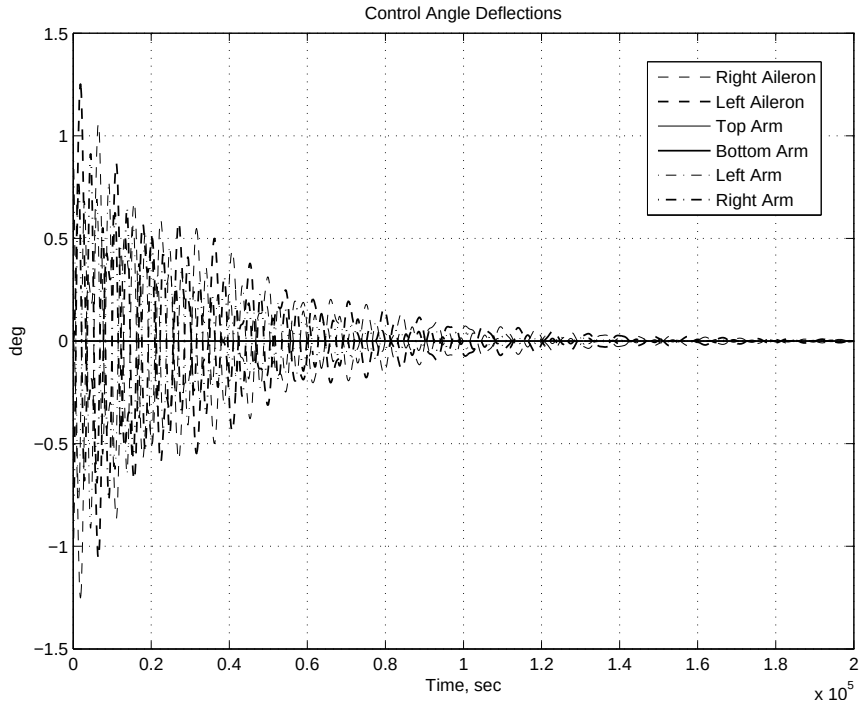


Figure E.60: Control angle deflections 500 km with yaw offset 10°.

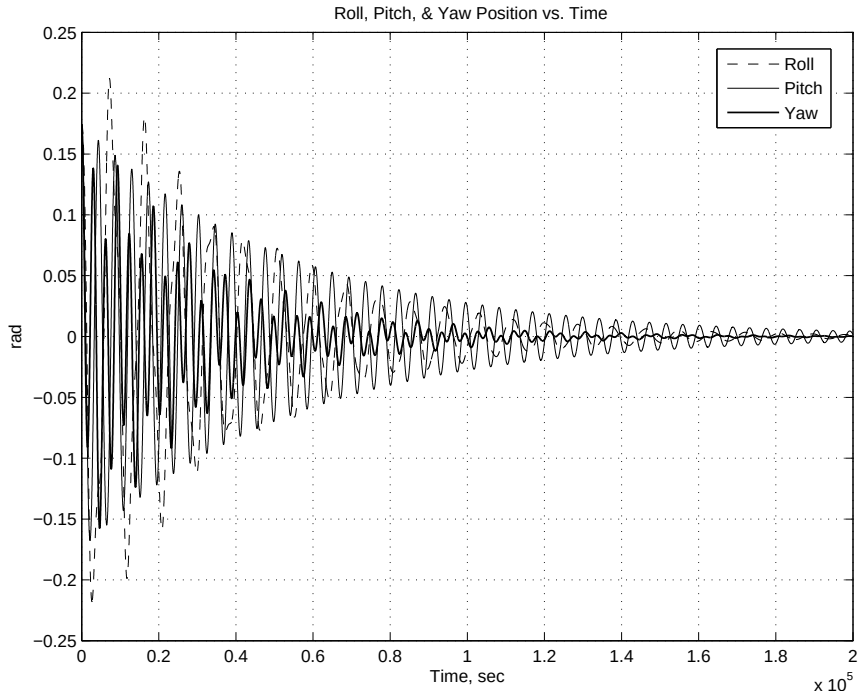


Figure E.61: Angular displacement at 500km with roll, pitch and yaw offset 10°.

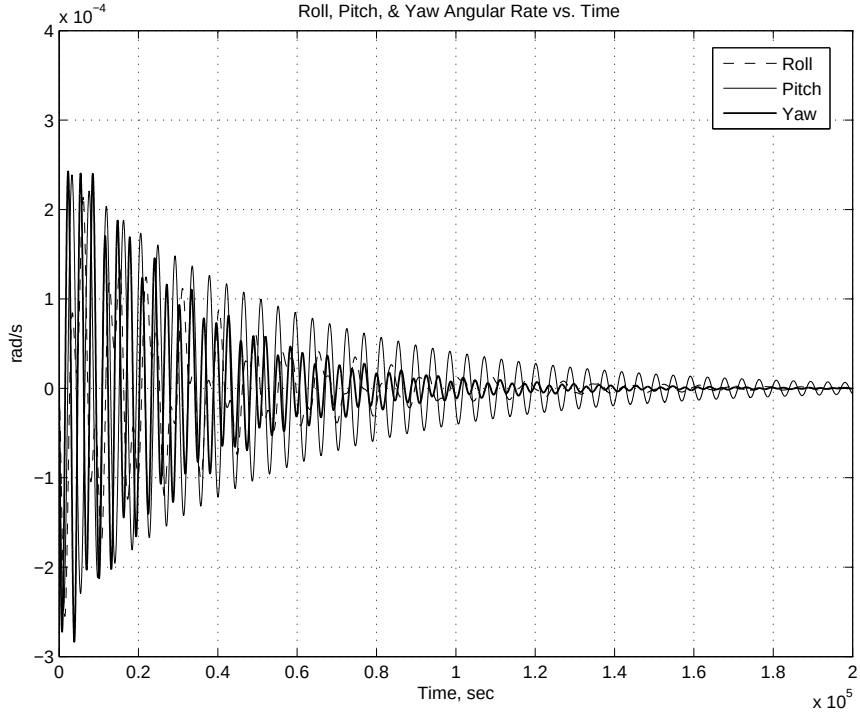


Figure E.62: Angular rate at 500km with roll, pitch and yaw offset 10°.

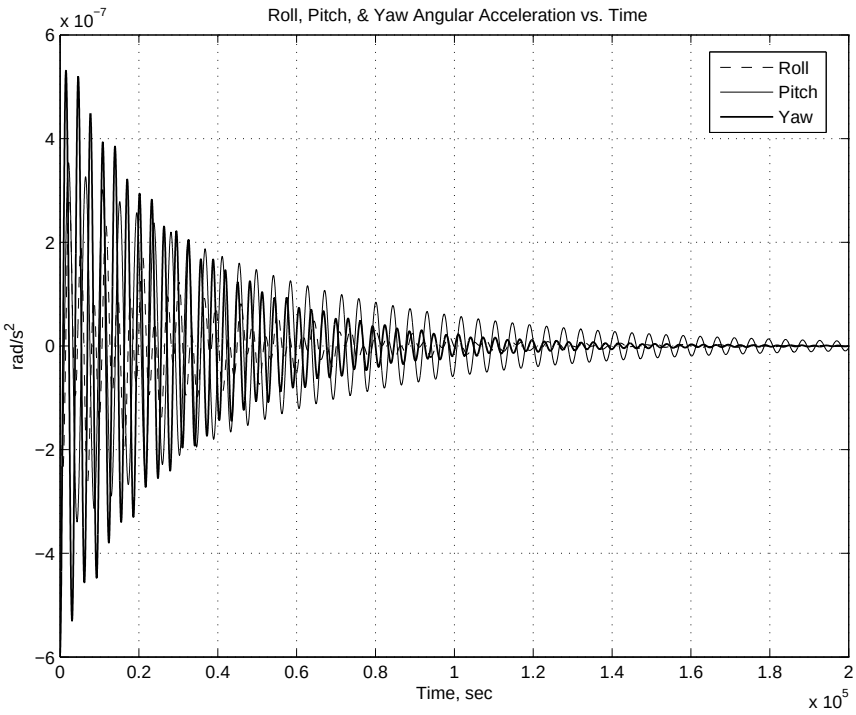


Figure E.63: Angular Acceleration at 500km with roll, pitch and yaw offset 10°.

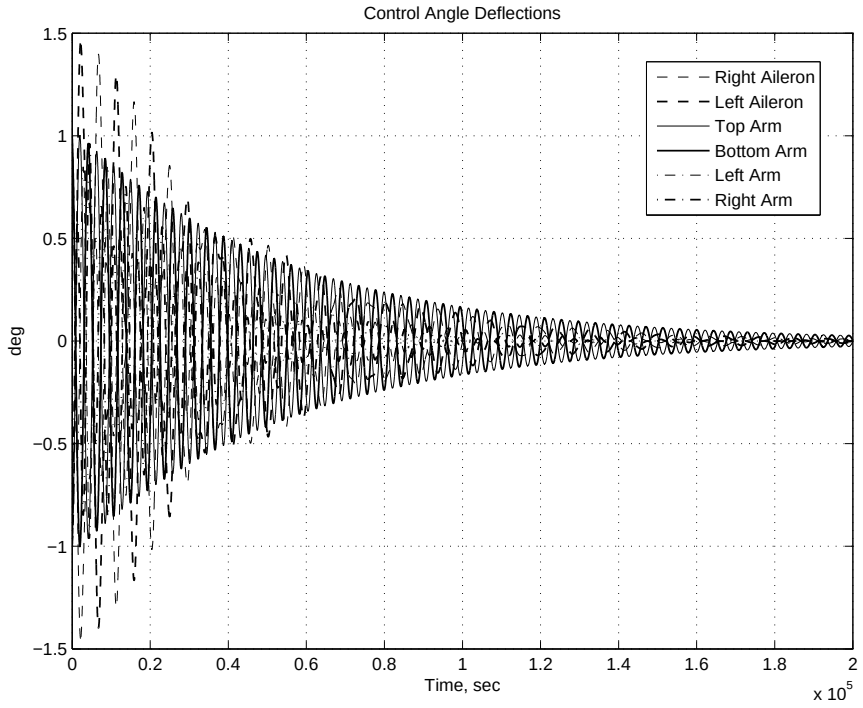


Figure E.64: Control angle deflections 500 km with roll, pitch and yaw offset 10°.

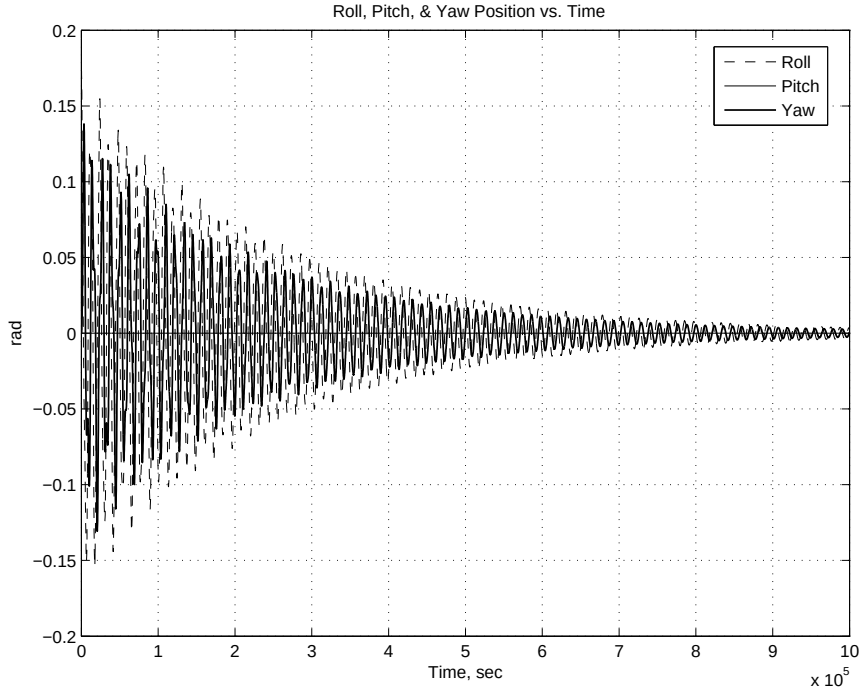


Figure E.65: Angular displacement at 600km with roll offset 10°.

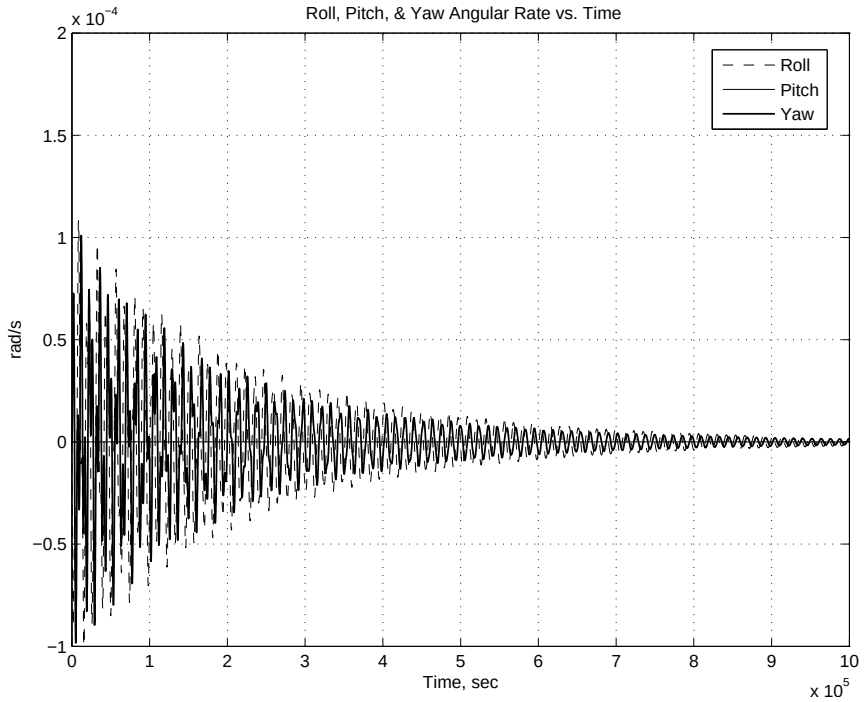


Figure E.66: Angular rate at 600km with roll offset 10°.

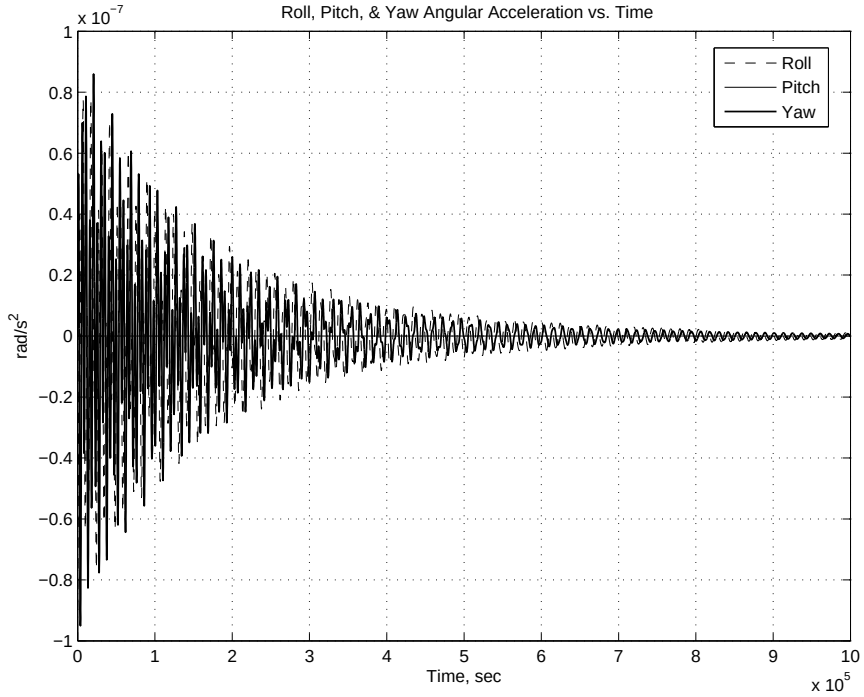


Figure E.67: Angular Acceleration at 600km with roll offset 10°.

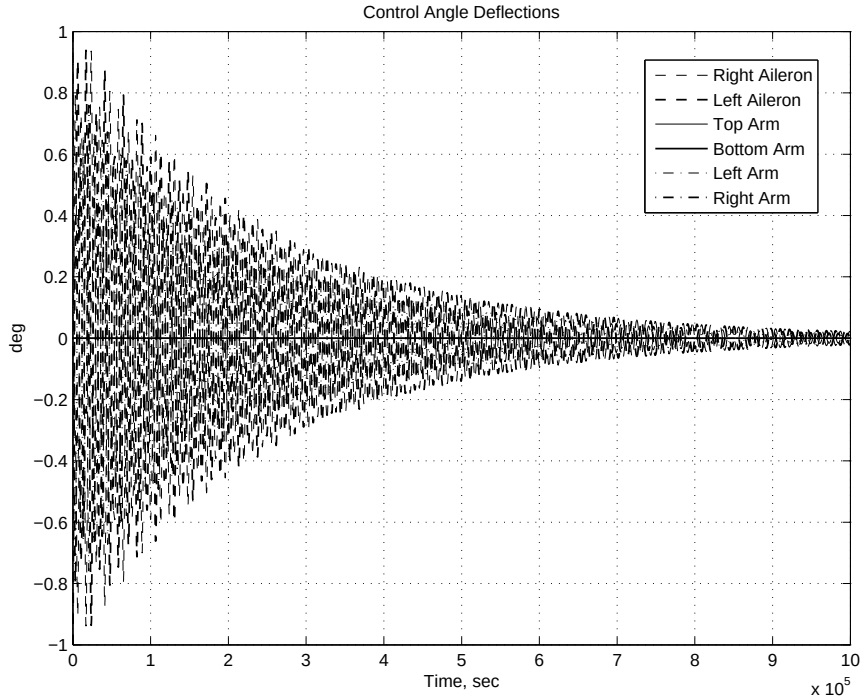


Figure E.68: Control angle deflections 600 km with roll offset 10°.

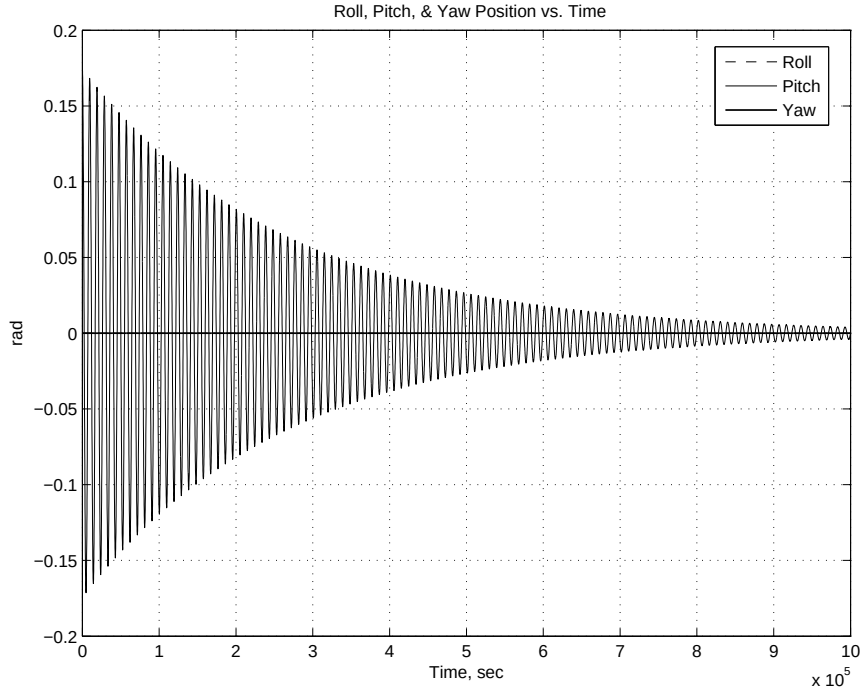


Figure E.69: Angular displacement at 600km with pitch offset 10°.

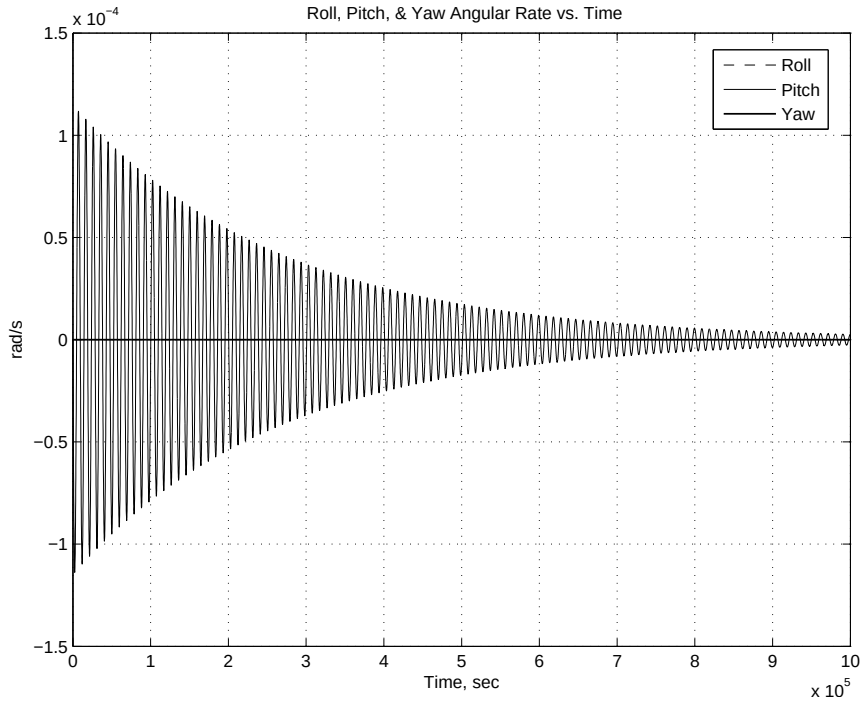


Figure E.70: Angular rate at 600km with pitch offset 10°.

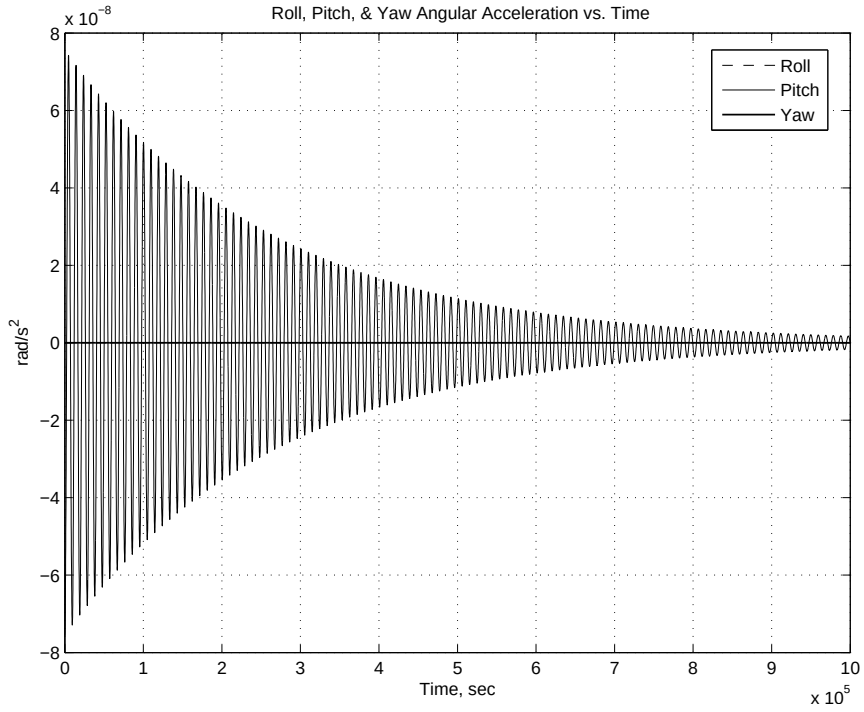


Figure E.71: Angular Acceleration at 600km with pitch offset 10°.

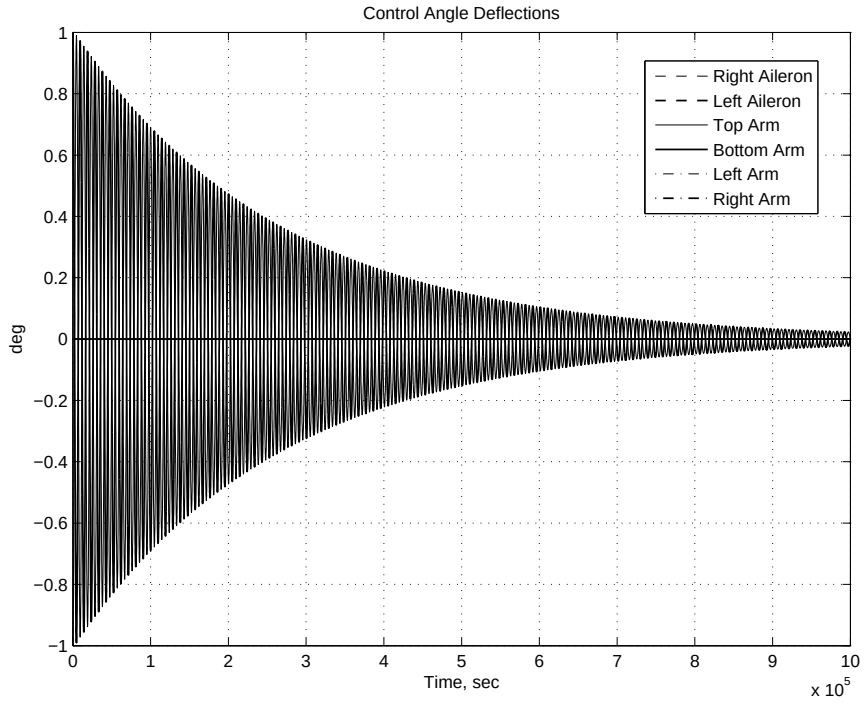


Figure E.72: Control angle deflections 600 km with pitch offset 10°.

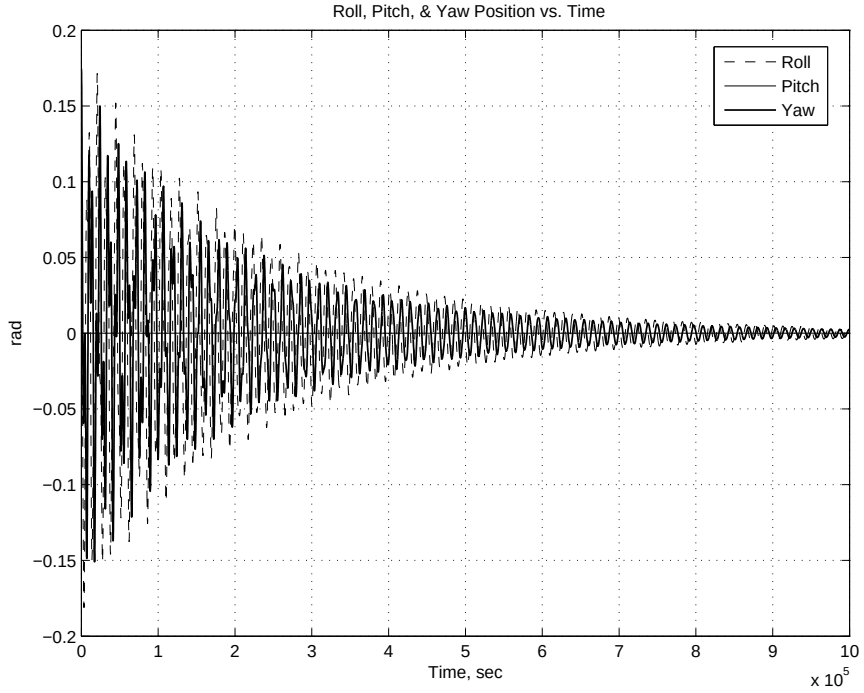


Figure E.73: Angular displacement at 600km with yaw offset 10°.

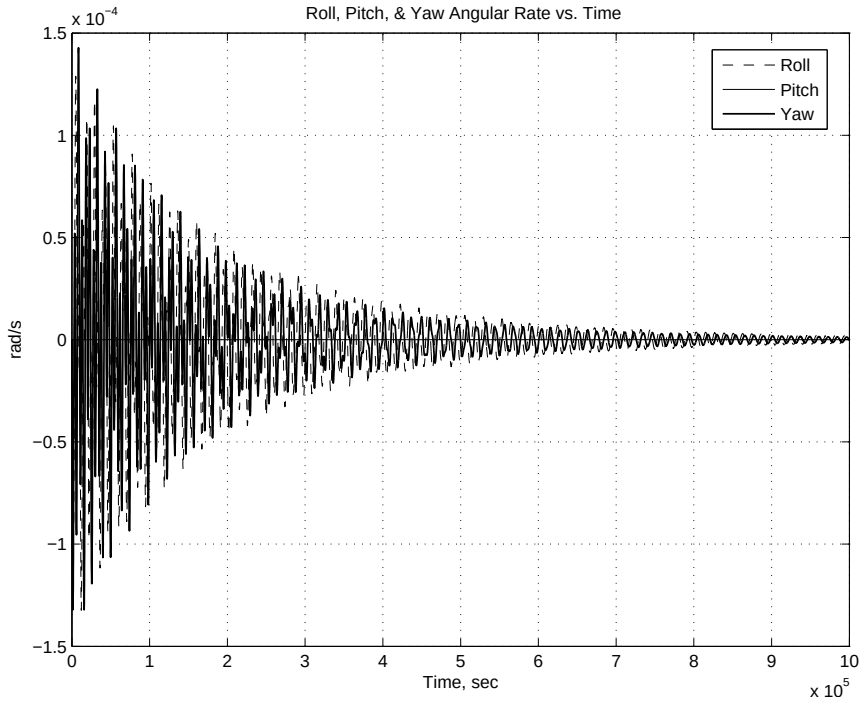


Figure E.74: Angular rate at 600km with yaw offset 10°.

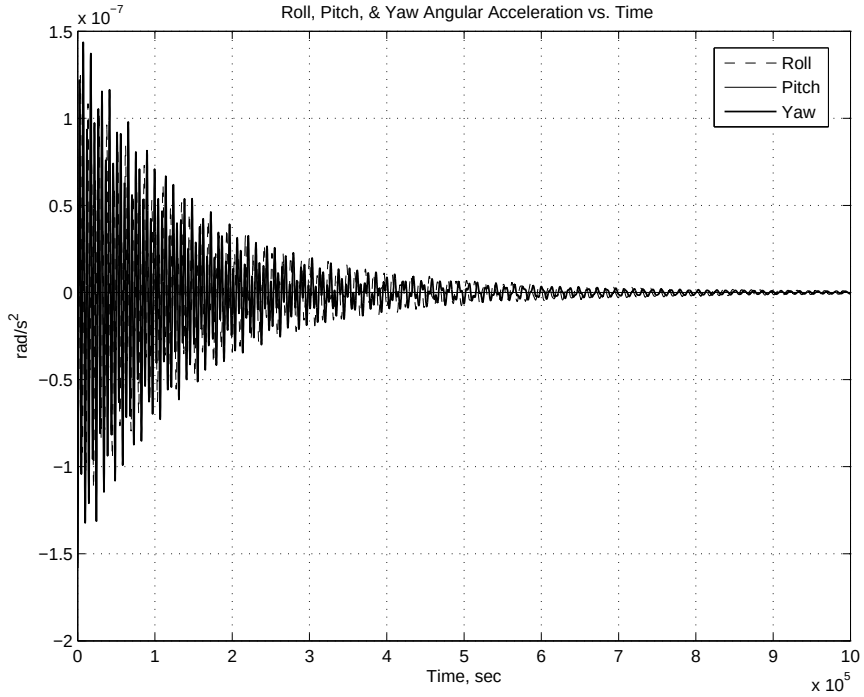


Figure E.75: Angular Acceleration at 600km with yaw offset 10°.

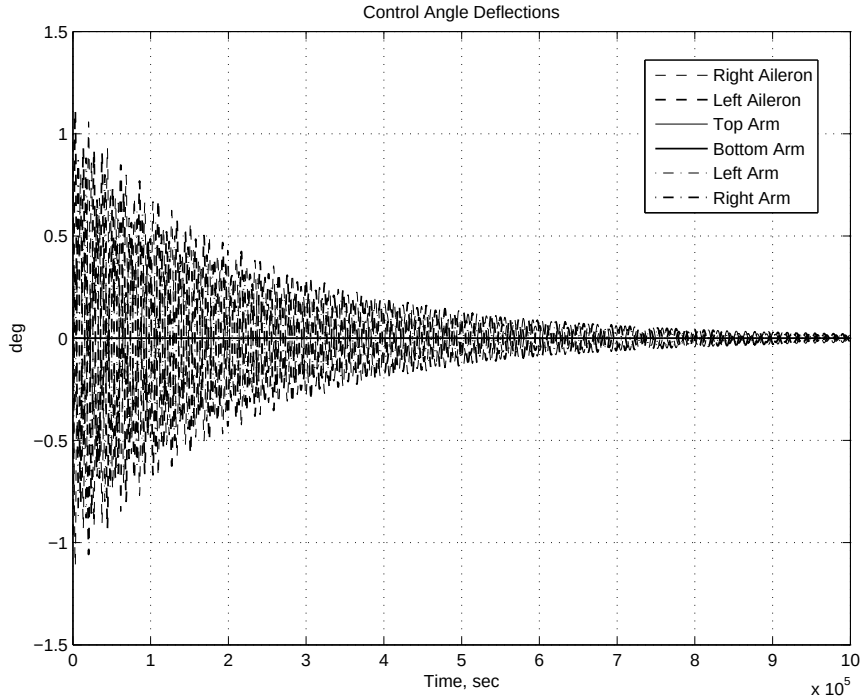


Figure E.76: Control angle deflections 600 km with yaw offset 10°.

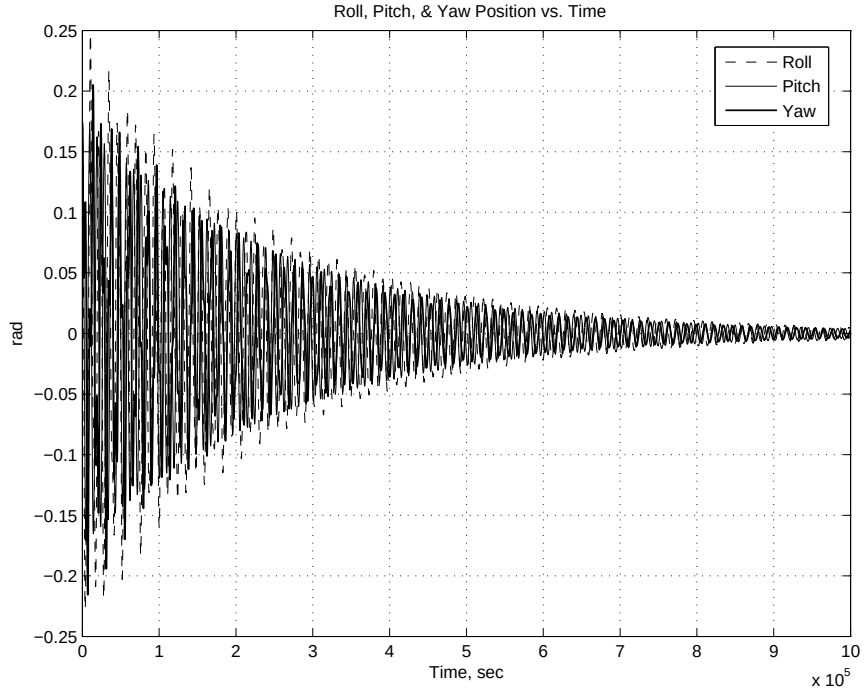


Figure E.77: Angular displacement at 600km with roll, pitch and yaw offset 10°.

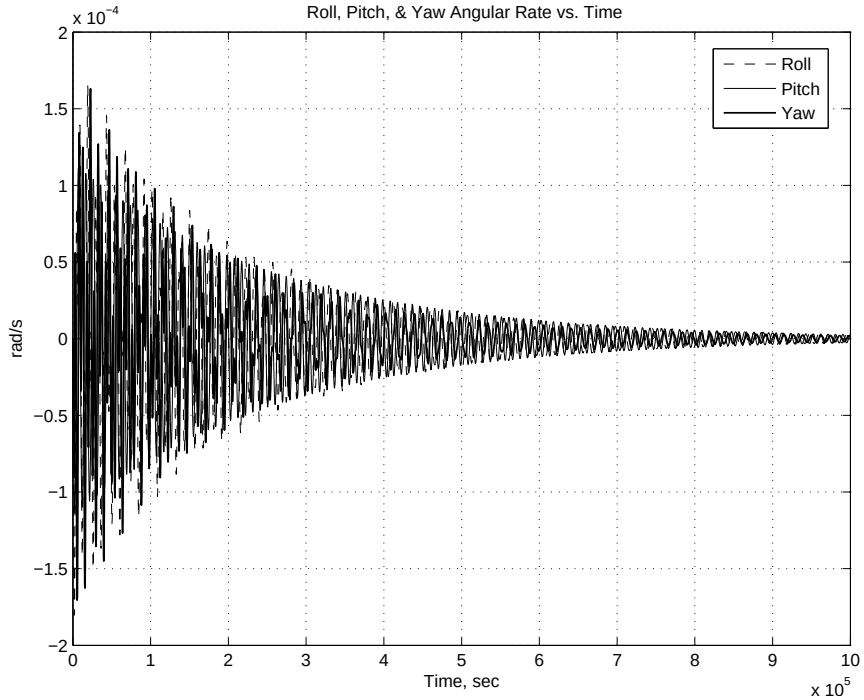


Figure E.78: Angular rate at 600km with roll, pitch and yaw offset 10°.

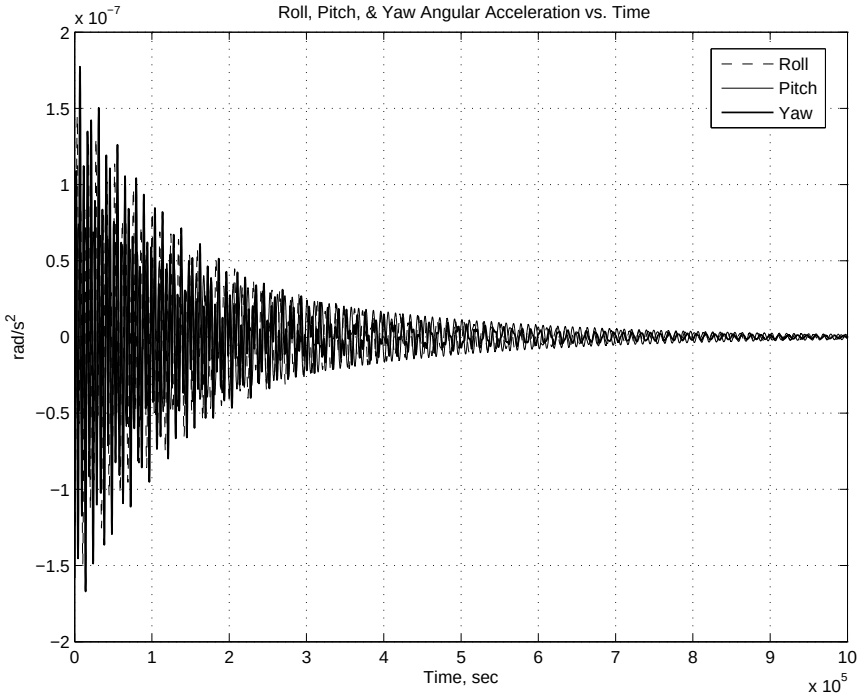


Figure E.79: Angular Acceleration at 600km with roll, pitch and yaw offset 10°.

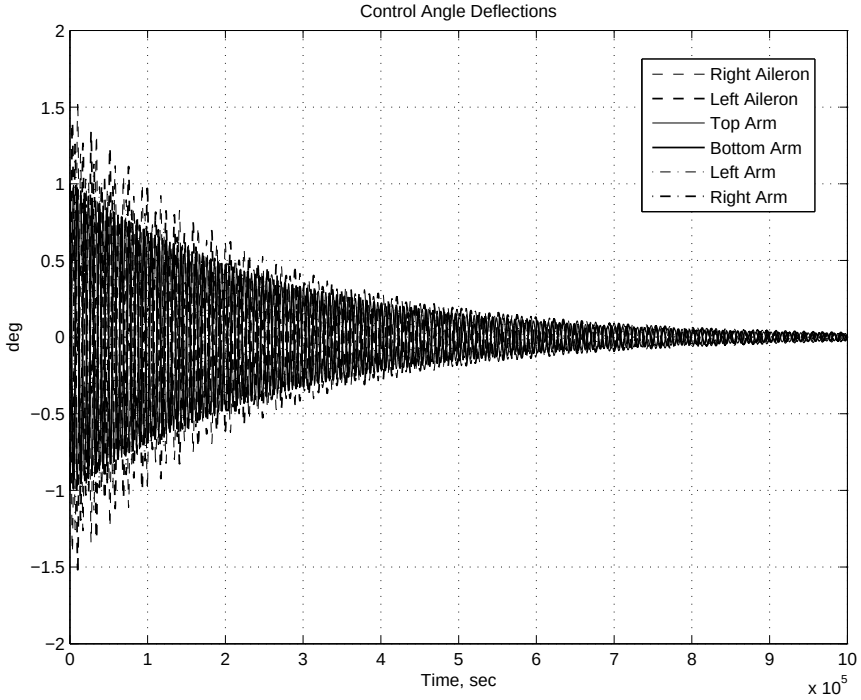


Figure E.80: Control angle deflections 600 km with roll, pitch and yaw offset 10°.

Appendix F. Original Design Nonlinear Drag Equations

F.1 External Torques from Atmospheric Drag (Nonlinear)

Satellites in LEO (LEO)(between 130 km to 600 km) experience an aerodynamic drag force which is given by:

$$F_{drag} = \frac{1}{2}\rho V^2 C_D A \tag{6.1}$$

Where F_{drag} =drag force (N), ρ =atmospheric density (kg/m^3), V =velocity (m/s), $C_D = 2$ drag coefficient and A =projected area (m^2). Altitudes above 125 km altitude are in the free molecular flow regime [1, page 318]. In the free molecular flow regime, particles are typically modeled either as specular or diffuse reflections. A specular reflection assumes that molecules are perfectly elastic where the tangential velocity is constant and the normal velocity is reversed. The diffuse model assumes the molecules are reflected in a diffuse manner and have no memory of previous velocities. Either model imparts a force normal to the surface. Spacecraft in LEO will always experience a small drag torque since it's not possible to locate the CG to the exact geometric center and therefore will need some way to control it's attitude. Assumptions that were made for the following drag torque equations are:

- The drag panels are only visible to the incoming wind only on the front/outward facing sides.
- The drag coefficient was chosen to be between $1.5 < C_D \leq 2.5$.
- Atmospheric density is constant and averaged over the orbit.(neglecting solar effects and atmospheric perturbations).

F.1.1 Drag Torque Equations. The basic design being modeled as a cube shaped spacecraft that has arms with 4 drag panels controlling pitch, roll and yaw. To pitch up or down, the top or bottom panel will be swung out into the air stream.

For yaw, the left and right panels are used. To roll left and right, all panels will be extended and rotated with respect to the arms in a "propeller" configuration. To determine the drag effects on both the cube and the panels, the velocity must be known and is given by [11, page 70]

$$V = \sqrt{\frac{\mu}{r_{orbit}}} \tag{6.2}$$

where the standard gravitational parameter $\mu = 398600\frac{km^3}{s^2}$ and the orbital radius $r_{orbit} = 6378.135km + Altitude$, and the velocity vector is in the $-a_1$ direction. The density must also be determined and based on the altitude range being modeled, the density ρ varies between 10^{-8} to $10^{-1}\frac{kg}{m^3}$ for LEO orbits.

F.1.2 Drag Effects from Spacecraft Body. The spacecraft body doesn't produce any torques unless the CG is not at the geometric center. In this model, the CG location can be moved off center to produce small torques for modelling purposes. First the roll, pitch, yaw rotation matrix is used to go from the A frame to the B frame as in equation 3.2. The angle of the incoming molecules and the inward normal directions n for the cube surfaces can be found by using the dot product of the two vectors and since we are dealing with unit vectors, $\cos(\alpha) = v \cdot n$. Since the inward normal vectors are aligned with the principal axes, and from equation 3.6 the dot products for each cube surface become:

$$\cos(\alpha_{Left}) = (-a_1 \cdot b_2) = -R^{BA}{}_{(2,1)} \tag{6.3a}$$

$$\cos(\alpha_{Right}) = (-a_1 \cdot -b_2) = R^{BA}{}_{(2,1)} \tag{6.3b}$$

$$\cos(\alpha_{Bot}) = (-a_1 \cdot -b_3) = R^{BA}{}_{(3,1)} \tag{6.3c}$$

$$\cos(\alpha_{Top}) = (-a_1 \cdot b_3) = -R^{BA}{}_{(3,1)} \tag{6.3d}$$

$$\cos(\alpha_{Front}) = (-a_1 \cdot -b_1) = R^{BA}{}_{(1,1)} \tag{6.3e}$$

$$\cos(\alpha_{Back}) = (-a_1 \cdot b_1) = -R^{BA}{}_{(1,1)} \tag{6.3f}$$

To find the drag force on each exposed surface, the projected area must be known and is found by $A_p \triangleq \iint H(\cos(\alpha))\cos(\alpha)dA$ [2, page 251]. where H is the Heaviside function and $\cos(\alpha)$ is from equations 6.3, a-f. The Heaviside function is used so that only the areas the see the incoming molecules are taken into account. Since the areas of the faces are the side of a cube, the integral becomes L^2, where L is the length of the side of the spacecraft body. The projected areas for each side become:

$$A_{Left} = L^2 H(\cos(\alpha_{Left}))\cos(\alpha_{Left}) \tag{6.4a}$$

$$A_{Right} = L^2 H(\cos(\alpha_{Right}))\cos(\alpha_{Right}) \tag{6.4b}$$

$$A_{Bot} = L^2 H(\cos(\alpha_{Bot}))\cos(\alpha_{Bot}) \tag{6.4c}$$

$$A_{Top} = L^2 H(\cos(\alpha_{Top}))\cos(\alpha_{Top}) \tag{6.4d}$$

$$A_{Front} = L^2 H(\cos(\alpha_{Front}))\cos(\alpha_{Front}) \tag{6.4e}$$

$$A_{Back} = L^2 H(\cos(\alpha_{Back}))\cos(\alpha_{Back}) \tag{6.4f}$$

By using equation 6.1 and substituting in equations 6.13, the drag forces for each face in the A frame become:

$$F_{Left_A} = \frac{1}{2} C_D A_{Left} \rho V^2 \hat{v} \tag{6.5a}$$

$$F_{Right_A} = \frac{1}{2} C_D A_{Right} \rho V^2 \hat{v} \tag{6.5b}$$

$$F_{Bot_A} = \frac{1}{2} C_D A_{Bot} \rho V^2 \hat{v} \tag{6.5c}$$

$$F_{Top_A} = \frac{1}{2} C_D A_{Top} \rho V^2 \hat{v} \tag{6.5d}$$

$$F_{Front_A} = \frac{1}{2} C_D A_{Front} \rho V^2 \hat{v} \tag{6.5e}$$

$$F_{Back_A} = \frac{1}{2} C_D A_{Back} \rho V^2 \hat{v} \tag{6.5f}$$

By using the rotation matrix 3.2 the force in the B frame can be calculated by substituting equations 6.5 and 3.2 into the following equations (note, we are only interested in the inward normal force).

$$F_{Left_B} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} R^{BA} F_{Left_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6.6a}$$

$$F_{Right_B} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} R^{BA} F_{Right_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6.6b}$$

$$F_{Bot_B} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} R^{BA} F_{Bot_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{6.6c}$$

$$F_{Top_B} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} R^{BA} F_{Top_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{6.6d}$$

$$F_{Front_B} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} R^{BA} F_{Front_A} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{6.6e}$$

$$F_{Back_B} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} R^{BA} F_{Back_A} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{6.6f}$$

Once the drag force is calculated in the B frame, the torques can be calculated by $T = r \mathrm{x} F$ [2, page 251] where r is the vector from the CG to the center of pressure

for each surface and F is the drag force in the B frame. For the cube surfaces, r becomes:

$$r_{Left} = \begin{bmatrix} 0 \\ -0.5L \\ 0 \end{bmatrix} - CG \tag{6.7a}$$

$$r_{Right} = \begin{bmatrix} 0 \\ 0.5L \\ 0 \end{bmatrix} - CG \tag{6.7b}$$

$$r_{Bot} = \begin{bmatrix} 0 \\ 0 \\ 0.5L \end{bmatrix} - CG \tag{6.7c}$$

$$r_{Top} = \begin{bmatrix} 0 \\ 0 \\ -0.5L \end{bmatrix} - CG \tag{6.7d}$$

$$r_{Front} = \begin{bmatrix} 0.5L \\ 0 \\ 0 \end{bmatrix} - CG \tag{6.7e}$$

$$r_{Back} = \begin{bmatrix} -0.5L \\ 0 \\ 0 \end{bmatrix} - CG \tag{6.7f}$$

Substituting in equations 6.6 and 6.17, the torque equations for each surface of the spacecraft body in the B frame become:

$$M_{Left} = r_{Left} \mathrm{x} F_{Left_B} \tag{6.8a}$$

$$M_{Right} = r_{Right} \mathrm{x} F_{Right_B} \tag{6.8b}$$

$$M_{Bot} = r_{Bot} \mathrm{x} F_{Bot_B} \tag{6.8c}$$

$$M_{Top} = r_{Top} \mathrm{x} F_{Top_B} \tag{6.8d}$$

$$M_{Front} = r_{Front} \mathrm{x} F_{Front_B} \tag{6.8e}$$

$$M_{Back} = r_{Back} \mathrm{x} F_{Back_B} \tag{6.8f}$$

And finally the total torques (equations 6.8 a-f) from the spacecraft body are added to get:

$$M_{SatBody} = M_{Left} + M_{Right} + M_{Bot} + M_{Top} + M_{Front} + M_{Back} \tag{6.9}$$

F.1.3 Drag Effects from Spacecraft Drag Panels. By going through a similar process, the torques from the drag panels can be found. The drag panels adds some complexity to the problem since there are two sets of rotations, one going from the A frame to the B frame (equation 3.2) and the other going from the B frame to the F frame which by using the roll, pitch, yaw rotation sequence results in:

$$R^{FB} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_1) & s(\phi_1) \\ 0 & -s(\phi_1) & c(\phi_1) \end{bmatrix} \begin{bmatrix} c(\phi_2) & 0 & -s(\phi_2) \\ 0 & 1 & 0 \\ s(\phi_2) & 0 & c(\phi_2) \end{bmatrix} \begin{bmatrix} c(\phi_3) & s(\phi_3) & 0 \\ -s(\phi_3) & c(\phi_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.10}$$

To get from the A frame to the F frame for each drag panel, $R^{FA} = R^{FB} R^{BA}$ where R^{BA} is from equation 3.2 and each panel has its specific rotation matrix and set of angles from the B frame to the F frame:

$$R^{FB}{}_{Top} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_{TPan}) & s(\phi_{TPan}) \\ 0 & -s(\phi_{TPan}) & c(\phi_{TPan}) \end{bmatrix} \begin{bmatrix} c(-\phi_{TArm}) & 0 & -s(-\phi_{TArm}) \\ 0 & 1 & 0 \\ s(-\phi_{TArm}) & 0 & c(-\phi_{TArm}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.11a}$$

$$R^{FB}{}_{Bot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_{BPan}) & s(\phi_{BPan}) \\ 0 & -s(\phi_{BPan}) & c(\phi_{BPan}) \end{bmatrix} \begin{bmatrix} c(\phi_{BArm}) & 0 & -s(\phi_{BArm}) \\ 0 & 1 & 0 \\ s(\phi_{BArm}) & 0 & c(\phi_{BArm}) \end{bmatrix} \begin{bmatrix} c(0) & s(0) & 0 \\ -s(0) & c(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.11b}$$

$$R^{FB}{}_{Left} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_{LPan}) & s(\phi_{LPan}) \\ 0 & -s(\phi_{LPan}) & c(\phi_{LPan}) \end{bmatrix} \begin{bmatrix} c(0) & 0 & -s(0) \\ 0 & 1 & 0 \\ s(0) & 0 & c(0) \end{bmatrix} \begin{bmatrix} c(\phi_{LArm}) & s(\phi_{LArm}) & 0 \\ -s(\phi_{LArm}) & c(\phi_{LArm}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.11c}$$

$$R^{FB}{}_{Right} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi_{RPan}) & s(\phi_{RPan}) \\ 0 & -s(\phi_{RPan}) & c(\phi_{RPan}) \end{bmatrix} \begin{bmatrix} c(0) & 0 & -s(0) \\ 0 & 1 & 0 \\ s(0) & 0 & c(0) \end{bmatrix} \begin{bmatrix} c(-\phi_{RArm}) & s(-\phi_{RArm}) & 0 \\ -s(-\phi_{RArm}) & c(-\phi_{RArm}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.11d}$$

where ϕ_{TPad}, ϕ_{BPad}, ϕ_{LPad}, and ϕ_{RPad}, are the top, bottom, left and right drag panel angles used for roll control and range from $\pm 45°$. ϕ_{TArm}, ϕ_{BArm}, ϕ_{LArm}, and ϕ_{RArm}, are the top, bottom, left and right arms that are connected to the drag panels. The angles between the b^1 direction and the arms range from $0 - 90°$. In equations 6.11a and 6.11d, the arm angles have a negative sign so that all angles will be positive so the controller will only have to input positive angles. One of the rotation angles in each equation are always zero, this is due to the fact that there are only two degrees of freedom for each panel.

The rest of the torque equations follow the same process as before (see equations 6.3), the angles between the incoming molecules and the inward normal directions are

$$\cos(\alpha_{Left}) = (-a_1 \cdot f_2) = -R^{FA}{}_{(2,1)} \tag{6.12a}$$

$$\cos(\alpha_{Right}) = (-a_1 \cdot -f_2) = R^{FA}{}_{(2,1)} \tag{6.12b}$$

$$\cos(\alpha_{Bot}) = (-a_1 \cdot -f_3) = R^{FA}{}_{(3,1)} \tag{6.12c}$$

$$\cos(\alpha_{Top}) = (-a_1 \cdot f_3) = -R^{FA}{}_{(3,1)} \tag{6.12d}$$

The projected areas from the panels become:

$$A_{Left} = L^2 H(\cos(\alpha_{Left}))\cos(\alpha_{Left}) \tag{6.13a}$$

$$A_{Right} = L^2 H(\cos(\alpha_{Right}))\cos(\alpha_{Right}) \tag{6.13b}$$

$$A_{Bot} = L^2 H(\cos(\alpha_{Bot}))\cos(\alpha_{Bot}) \tag{6.13c}$$

$$A_{Top} = L^2 H(\cos(\alpha_{Top}))\cos(\alpha_{Top}) \tag{6.13d}$$

where L is the side length of each square drag panel.

Calculate forces from the drag panels in the A frame:

$$F_{Left_A} = \frac{1}{2} C_D A_{Left} \rho V^2 \hat{v} \tag{6.14a}$$

$$F_{Right_A} = \frac{1}{2} C_D A_{Right} \rho V^2 \hat{v} \tag{6.14b}$$

$$F_{Bot_A} = \frac{1}{2} C_D A_{Bot} \rho V^2 \hat{v} \tag{6.14c}$$

$$F_{Top_A} = \frac{1}{2} C_D A_{Top} \rho V^2 \hat{v} \tag{6.14d}$$

Calculate force from drag panels in the F frame and pull out the inward normal component:

$$F_{Left_F} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} R^{FA}{}_{Left} F_{Left_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6.15a}$$

$$F_{Right_F} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} R^{FA}{}_{Right} F_{Right_A} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6.15b}$$

$$F_{Bot_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} R^{FA}{}_{Bot} F_{Bot_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{6.15c}$$

$$F_{Top_F} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} R^{FA}{}_{Top} F_{Top_A} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{6.15d}$$

Convert the inward normal component of each surface in the F frame to the B frame.

$$F_{Left_B} = (R^{FB}{}_{Left})^T F_{Left_F} \tag{6.16a}$$

$$F_{Right_B} = (R^{FB}{}_{Right})^T F_{Right_F} \tag{6.16b}$$

$$F_{Bot_B} = (R^{FB}{}_{Bot})^T F_{Bot_F} \tag{6.16c}$$

$$F_{Top_B} = (R^{FB}{}_{Top})^T F_{Top_F} \tag{6.16d}$$

Find r from the CG to the center of pressure for each panel:

$$r_{Left} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{LArm}) \\ -0.5L - (L_{Arm} + 0.5L_{panel})\sin(\phi_{LArm}) \\ 0 \end{bmatrix} - CG \tag{6.17a}$$

$$r_{Right} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{RArm}) \\ 0.5L + (L_{Arm} + 0.5L_{panel})\sin(\phi_{RArm}) \\ 0 \end{bmatrix} - CG \tag{6.17b}$$

$$r_{Bot} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{BArm}) \\ 0 \\ 0.5L + (L_{Arm} + 0.5L_{panel})\sin(\phi_{BArm}) \end{bmatrix} - CG \tag{6.17c}$$

$$r_{Top} = \begin{bmatrix} -0.5L - (L_{Arm} + 0.5L_{Panel})\cos(\phi_{TArm}) \\ 0 \\ -0.5L - (L_{Arm} + 0.5L_{panel})\sin(\phi_{TArm}) \end{bmatrix} - CG \tag{6.17d}$$

where L is the side length of the spacecraft body, L_{Arm} is the length of the control arm, L_{Panel} is the side length of the square panels, ϕ_{LArm}, ϕ_{RArm}, ϕ_{BArm} and ϕ_{TArm} are the angles of the arms from their retracted positions.

Finally, the torque equations for the drag panels with all the substitutions become:

$$M_{Left} = r_{Left} \mathrm{x} F_{Left_B} \tag{6.18a}$$

$$M_{Right} = r_{Right} \mathrm{x} F_{Right_B} \tag{6.18b}$$

$$M_{Bot} = r_{Bot} \mathrm{x} F_{Bot_B} \tag{6.18c}$$

$$M_{Top} = r_{Top} \mathrm{x} F_{Top_B} \tag{6.18d}$$

$$\tag{6.18e}$$

And finally the total torques (equations 6.8 a-f) from the spacecraft body are added to get:

$$M_{Panels} = M_{Left} + M_{Right} + M_{Bot} + M_{Top} \tag{6.19}$$

And the total torques are:

$$M_{Total} = M_{SatBody} + M_{Panels} \tag{6.20}$$

Bibliography

1. Frank J. Regan, Satya M. Anandakrishnan. *Dynamics of Atmospheric Entry*. American Institute of Aeronautics and Astronautics, Inc., 370 L'Enfant Promenade, SW, Washington, DC 20024-2518, first edition, 1993.

2. Hughes, Peter C. *Spacecraft Attitude Dynamics*. Wiley and Sons, New York, first edition, 1986.

3. Minzner, R.A. *The 1976 Standard Atmosphere Above 86-km Altitude*. Tech Report NASA SP-398, NASA, Goddard Spaceflight Center, Greenbelt, MD, 1976.

4. Moe, Kenneth. *Surface-Particle-Interaction Measurements Using Paddlewheel Satellites*. Tech Report AD 673 946, Douglas Aircraft Company, Inc., Santa Monica, CA, 1968.

5. Psiaki, Mark L. "Nanosatellite Attitude Stabilization Using Passive Aerodynamics and Active Magnetic Torquing". *Journal of Guidance Control and Dynamics*, 27(3):347–355, May-June 2004.

6. Ravindran, R. and P.C. Hughes. "Optimal Aerodynamic Attitude Stabilization of Near-Earth Satellites". *Journal of Spacecraft and Rockets*, 9(7):499–506, July 2004.

7. Renjith R. Kumar, Michael L. Heck, Daniel D. Mazanek. "Simulation and Shuttle Hitchhiker Validation of Passive Satellite Aerostabilization". *Journal of Spacecraft and Rockets*, 32(5):806–811, September-October 1995.

8. Renjith R. Kumar, Michael L. Heck, Daniel D. Mazanek. "Parametric and Classical Resonance in Passive Satellite Aerostabilization". *Journal of Spacecraft and Rockets*, 33(2):806–811, March-April 1996.

9. Sellers, Jerry Jon. *Understanding Space An Introduction to Astronautics*. McGraw Hill, Boston, MA, second edition, 2004.

10. Wie, Bong. *Space Vehicle Dynamics and Control*. American Institute of Aeronautics and Astronautics, Inc., 1801 Alexander Bell Dr, Reston, VA 20191, first edition, 1952.

11. Wiesel, William E. *Spaceflight Dynamics*. McGraw Hill Companies, Inc., New York, second edition, 1997.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
22–03–2007	Master's Thesis	Jun 2006 – Mar 2007

4. TITLE AND SUBTITLE

Satellite Attitude Control Using Atmospheric Drag

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Guettler, David B., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GA/ENY/07-M10

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Attitude control is a requirement for most satellites. Many schemes have been devised over the years including control moment gyros, reaction wheels, spin stabilization and gravity gradient stabilization. For low Earth orbits, the Earth's atmosphere can have an affect on a satellite's orbit and attitude. This research effort examines the feasibility of using the atmosphere to actively control a spacecraft's attitude using drag panels. Using the atmosphere to control spacecraft attitude has been researched in the past however very little research has been done using an active feedback control system to maintain spacecraft attitude. A linear computer model was created using a proportional controller. This model was used to evaluate the effectiveness of using drag panels for attitude control. Results from the simulation show that the spacecraft can recover from disturbance torques that may cause a change in attitude very effectively especially at low altitudes (200-300km). Settling time increases as altitude increases and varies from minutes to weeks.

15. SUBJECT TERMS

Satellite Drag; Attitude Control; Free Molecular Flow; Orbit Decay; Atmospheric Perturbations; Aerodynamic Torques; Satellite Attitude Simulator

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. William E. Wiesel, (ENY)
U	U	U	UU	177	19b. TELEPHONE NUMBER *(include area code)* (937) 255–6565, ext 4312