Design, Analysis, and Simulation of Attitude Controllers for the MicroNimbus Mission

Author:
Jian H. Li

Advisor:
Dr. Glenn Ligthsey

Space Systems Design Laboratory (SSDL)
Daniel Guggenheim School of Aerospace Engineering
Georgia Institute of Technology

April 20, 2018

This project is presented to the Academic Faculty by Jian H. Li under the advisement of Dr. Glenn Lightsey, in partial fulfillment of the requirements for the Degree Master of Sciences in Aerospace Engineering at Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology.

Contents

Design, Analysis, and Simulation of Attitude Controllers for the MicroNimbus Mission

Jian H. Li* and Glenn Lightsey†

Abstract

MicroNimbus is a small satellite mission currently under development at Georgia Tech Space System Design Lab. Its mission is to measure the vertical temperature profile of the atmosphere. To achieve mission success, the satellite needs to control its attitude in different modes throughout the mission, which has motivated a development of several attitude controllers. These controllers include a B-dot controller for detumble mode, a Nadir-Pointing controller for science mode, a Slew controller for ground station down-link mode, a Sun-Pointing controller for battery charging mode, and a User-Defined Pointing controller for calibration mode. These controllers are designed based their functionality, and the stability characteristics in the closed-loop system. This project includes adapting, designing, and implementing the controllers on a NASA GSFC open source software 42 for a full attitude control simulation. Additionally, the project report discusses the theory behind each of controllers and how it is implemented in the simulation tool. These controllers serve as a baseline design for eventual implementation as flight controllers on MicroNimbus and other similar CubeSat mission in the Space System Design Lab.

Nomenclature

ADC	=	Attitude Determination and Control
F_N	=	Inertia Frame
F_L	=	LVLH Frame
F_B	=	Spacecraft Body Frame
F_R	=	Commanded Frame
F_S	=	Sun Pointing Frame
F_G	=	Ground Station Pointing Frame
F_C	=	Axis Commanded Frame
$IGRF$	=	Inernational Geomagnetic Reference Field
$LVLH$	=	Local-Vertical, Local-Horizontal
ECI	=	Earth Centered Inertial
IMU	=	Inertial Measurement Unit
PD	=	Proportional-Derivative

*Graduate Student, Guggenheim School of Aerospace Engineering, Georgia Institute of Technology.
†Professor, Guggenheim School of Aerospace Engineering, Georgia Institute of Technology.

$\mathbf{T}_X$	=	Frame Rotational Matrix along X-axis
$\mathbf{T}_Y$	=	Frame Rotational Matrix along Y-axis
$\mathbf{T}_Z$	=	Frame Rotational Matrix along Z-axis
$\mathbf{C_{SR}}$	=	Frame Rotational Matrix from F_R to F_S
$\dot{\mathbf{C}}_{\mathbf{SR}}$	=	Time Rate Change of Frame Rotational Matrix from F_R to F_S
$\mathbf{C_{GR}}$	=	Frame Rotational Matrix from F_R to F_G
$\mathbf{K_p}$	=	Proportional Gain Matrix
$\mathbf{K_d}$	=	Derivative Gain Matrix
$\mathbf{q}$	=	Quaternion
$\mathbf{q}_{LR}$	=	Quaternion from F_R to F_L
$\mathbf{q}_{GR}$	=	Quaternion from F_R to F_G
$\mathbf{q}_{SR}$	=	Quaternion from F_R to F_S
$\mathbf{q}_{CR}$	=	Quaternion from F_R to F_C
$\boldsymbol{\omega}_{RL}$	=	Angular Rate of F_R to F_L
$\boldsymbol{\omega}_{RG}$	=	Angular Rate of F_R to F_G
$\boldsymbol{\omega}_{RS}$	=	Angular Rate of F_R to F_S
$\boldsymbol{\omega}_{RC}$	=	Angular Rate of F_R to F_C
$\boldsymbol{\delta q}$	=	Error Quaternion
$\boldsymbol{\omega}$	=	Angular Rate
$\boldsymbol{\tau_{cmd}}$	=	Commanded Torque
ISS	=	International Space Station
GPS	=	Global Positioning System
$MEKF$	=	Multiplicative Extended Kalman Filter
PMW	=	Pulse Width Modulation

1 Introduction

Most spacecraft missions have specific pointing requirements, which are driven by the operation modes of the mission. These pointing requirement can range from Earth pointing, inertial pointing, sun pointing, and other target pointing directions. Communications, broadcast, and Earth science related satellites are examples of Earth pointing satellite. They point their payload instruments nadir (towards the center of earth) for most portion of their missions. The Hubble Space Telescope[1] (HST) is an example of inertial pointing satellite. The HST points its main mirror to a commanded location in the sky for observations. The Solar & Heliospheric Observatory[1] (SOHO) is an example of a sun pointing spacecraft whose mission is to study the sun. All pf these pointing modes are achieved by the on-board attitude determination and control system.

MicroNimbus is an Earth remote sensing spacecraft, whose mission is to collect global, near real-time atmospheric temperature measurements. The spacecraft requires several attitude controllers to complete its mission. These include a Nadir-Pointing controller, a Sun-Pointing controller, an Inertial-Pointing controller, and a Slew controller. Each of these controllers are designed based on specific payload and subsystem requirements. The Nadir-Pointing controller is used to point the aperture of the payload instruments towards the surface of the earth. The Sun-Pointing controller aligns the satellite's solar panel directly

towards the sun for battery charging. The Inertial-Pointing controller allows operators to point the spacecraft in any inertial direction for sensor calibration or sensor health check out. The Slew controller is used to maximize the downlink time during a ground station pass.

A thesis by Margaret Tam, *An Attitude Determination and Control System for Small Satellites*[2], details Attitude Determination and Control System (ADCS) designs from previous CubeSat missions by Texas Spacecraft Lab (TSL). These missions include, ARMADILLO, BEVO-2, and RACE [1]. The thesis provides hardware ADCS configurations for each mission mentioned above as well as attitude controller designs and simulations. The thesis outlines controller designs which are not only driven by the respective mission requirements but also constrained by the capability of the configuration and capability of the ADCS hardware. MicroNimbus has a CubeSat design that is influenced by and improves upon previous TSL CubeSats.More specifically, it has a different ADCS hardware configuration and mission science requirements from its predecessors. Thus, a new set of attitude controllers must be designed tailored to its new hardware setup.

The purpose of this project is to develop functional attitude controllers for the MicroNimbus mission. This paper discusses theory behind each controller design, the performance of each controller, and how it is implemented in a simulation.

2 ADC Hardware Overview

The key factors that drive the controller design are the ADCS hardware capabilities. This section provides the specifications of each ADCS component. The hardware specifications are used as input parameters for the simulation. The ADCS is show in Figure 1 and Figure 2. The ADCS is divided into top module and bottom module. Reaction wheels, magnetic torquers, one of the sun sensor modules, and the IMU are located at the top module. Another sun sensor module and magnetormeters are located at the bottom module. Fig.3 shows the GPS receiver, which is located on a 3U side of the spacecraft.

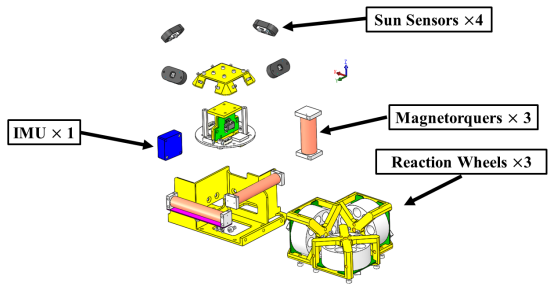

Figure 1: Top ADC Subsystem Module Exploded View

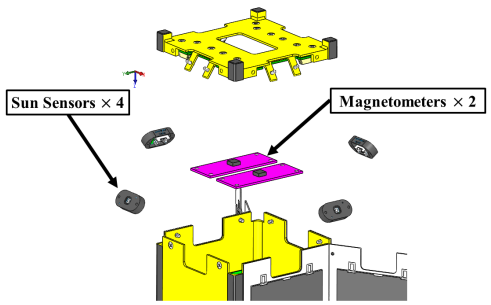

Figure 2: Bottom ADC Subsystem Module Exploded View

[1] http://sites.utexas.edu/tsl/past-missions/

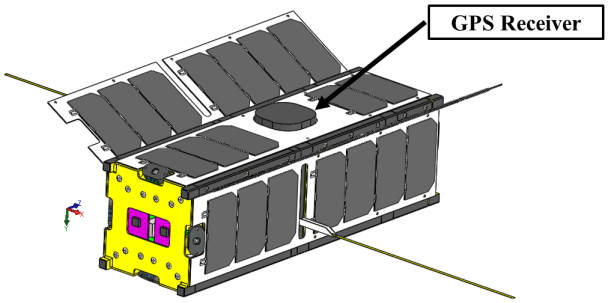

Figure 3: GPS Receiver

2.1 Sensors

This section provides an overview of the sensors used in MicroNimbus. The technical specifications and measurements of each type of sensors are also provided in individual section. The ADCS sensors include sun sensor, magnetometer, IMU and GPS receiver. For a given time, the sun sensor measures the sun vector in F_B, and the magnetometer measures the local magnetic field vector in F_B. The GPS receiver provides the location of the spacecraft in F_N. With an accurate magnetic field model and sun propagator model, one can predict the magnetic field vector and the sun vector based on the given location provided by the GPS receiver. Combine with the measured vectors and the predicted vectors, a MEKF can be used to calculate the spacecraft attitude with respect to the inertia frame.

2.1.1 Sun Sensors

The sun sensor measures the sun vector in body frame. The selected sun sensor is nanoSSOC-A60 that produced by SOLARMEMS. It is shown in Figure 4. It is a two orthogonal axes sun sensor with a field of view of $\pm 60^0$. Its accuracy is less than 0.5^0. The raw outputs of each sun sensor are analog voltage signals range between $0V$ and $3.3V$. Low-pass filters and Analog-to-Digital converters are implemented to filter and convert the output signals. There are 8 units of sun sensors on the spacecraft, which are divided into two modules, four units per module. The module is located on each end of the spacecraft. The sun sensor modules are shown in Figure 5. This configuration provides full-sky coverage for the spacecraft.

Figure 4: nanoSSOC-A60 Sun Sensor device[3]

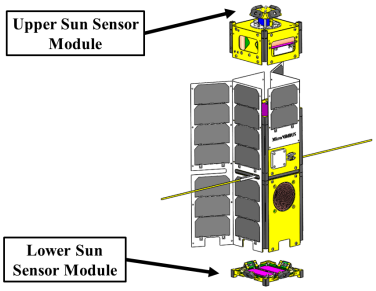

Figure 5: Sun Sensor Module Configuration

2.1.2 IMU

The M-G364PDCA is a small factor IMU with 6 degrees of freedom that produced by EPSON. Figure 6 [1]. shows an image of the IMU. The features of the IMU are listed in Table 1

Figure 6: EPSON M-G364PDCA IMU

The IMU consists two sensors, the triple gyroscopes and the accelerometer. The gyro sensor measures the speed or angular of rotation about each body axis of the spacecraft. Similarly, the accelerometer measures the gravity acceleration of each body axis. The IMU operates without any knowledge of the external inertial frame, F_N. The attitude determination is done using the IMU combined with external references such as sun sensors and magnetometers.

[1] https://global.epson.com/products$_a nd_d rivers/sensing_s ystem/imu/g364/$

Table 1: M-G364PDCA EPSON IMU Features[4]

Features	**Note**
Dimension	24×24×10 mm
Weight	10 grams
Triple Gyroscopes Dynamic Range	± 100°/s
Gyro Bias Instability	2.2°/h
Gyro Angular Random Walk	$0.09°/\sqrt{hr}$
Tri-Axis Accelerometer Dynamic Range	± 3 G
Accelerometer Bias Instability	0.05 mG
Digital Serial Interface	SPI/UART
Operating Temperature Range	−40°C to +85°C

2.1.3 Magnetometer

The magnetometer measures the strength and the direction of the local magnetic field. The measurements are recorded in F_B. Its measurement helps to determine the spacecraft attitude relative to the local magnetic field. The measurements combined with earth magnetic field model and orbit information provide spacecraft attitude relative to F_N.

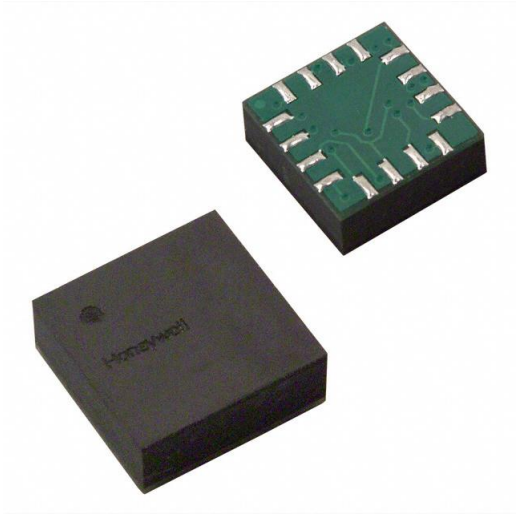

Figure 7: Honeywell Magnetic Sensor HMC 1053

Fig.7 shows the Honeywell magnetometer HMC1053. Two magnetometers are used in the ADCS.

Table 2: Honeywell HMC1053 Features[5]

Features	Note
Field Range	± 6 gauss
Sensitivity	$0.8 - 1.2$ mV/V/gauss
Resolution	120 μ gauss
Bandwidth	5 MHz

2.1.4 GPS Receiver

The GPS receiver used in MicroNimbus is the dual-frequency OEM615 unit produced by NovAtel. It provides precise position measurement for the ADCS. The OEM615 tracks all current GNSS constellation including GPS, GLONASS, Galileo, and QZSS[6]. Some technical specification are highlighted in Table3.

Table 3: NovAtel OEM615 Features[6]

Features	Note
Channel Configuration	120 Channels
Signal Tracking	GPS L1, L2
Position Accuracy (RMS)	L1:1.5m; L1/L2:1.2m
Measurement Precision (RMS)	L1:4cm
Maximum Data Rate	50Hz
Velocity Accuracy	0.03 m/s RMS
Velocity Limit	515 m/s

2.2 Actuators

There two types of actuators used in MicroNimbus, reaction wheels and magnetic torquers. Reaction wheels are used for fine attitude pointing. Magnetic torquers are used for detumble mode and reaction wheel angular momentum unloading.

2.2.1 Reaction Wheel

There are three reaction wheels in the spacecraft, one per each axis. The selected reaction wheel is the RW-0.01 by Sinclair. They are used as momentum wheel throughout the mission, which means that the reaction wheels operates at a nominal spin rate above zero to provide a nearly constant momentum on each axis. The reaction wheel specification is shown in Table4.

Table 4: Sinclair Picosatellite Reaction Wheels RW-0.01[7]

Features	**Note**
Nominal Momentum	0.01 Nms
Peak Momentum	0.018 Nms
Torque	± 1 mNm
Dimension	$50mm \times 50mm \times 30mm$
Telemetry	UART or I^2C

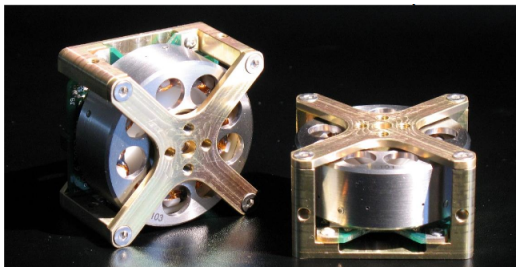

Figure 8: Sinclair Interplanetary Reaction Wheel

2.2.2 Magnetic Torquer

The magnetic torquers are used mainly for spacecraft detumble and reaction wheel momentum unloading. The magnetic torquers can compensate attitude drift by minor disturbance torques, but the actuation is slow. Thus, they are not used for fine attitude pointing. The spacecraft includes three magnetic torquers, one per each axis. The magnetic torquers are made in-house. The designed is driven by many factors including size, power consumption, magnetic moment dipole, and material. The desired magnetic moment dipole is $0.2Am^2$ for each torquer. However, the actual magnetic dipoles turned out to be much higher than expected, $0.8Am^2$ for X-axis and Y-axis and $0.36Am^2$ for Z-axis. Some design modifications are needed in order to achieve $0.2Am^2$ magnetic moment dipole. Fig.9 shows a magnetic torquers during the manufacturing progress.

Figure 9: Magnetic Torquer

3 Attitude Controllers

3.1 Attitude Controllers Overview

This section discusses the attitude controllers design for the MicroNimbus mission. There are several attitude controllers designed for the mission. Each controller governs a mission mode. These controllers are Nadir-Pointing controller, Axis-Command controller, Sun-Pointing Controller, Slew coontroller, and B-Dot controller.

A control flow diagram is shown in Fig.10. The logic transitions between each control mode are indicated in figure. Note that the De-Saturation module is not an attitude controller. It is a mechanism that helps to de-saturate the angular momentum of each reaction wheel. Additionally, an Axis-Command controller is not listed in this flow diagram.

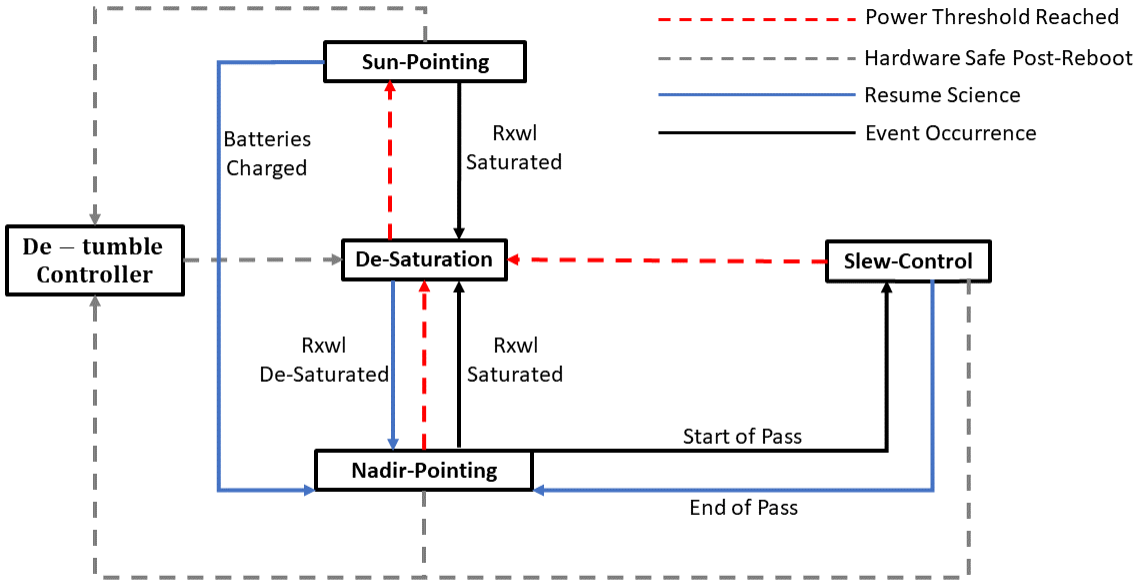

Figure 10: MicroNimbus Rotational Dynamics

3.2 Control Law & Stability Analysis

The control law that used for most of the attitude controls except the B-dot control is a quaternion-based nonlinear PD type tracking controller. The PD controller is shown in Eq.1

$$\boldsymbol{\tau}_{\mathbf{cmd}} = -\mathbf{K}_p \times \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3}} - \mathbf{K}_{\mathbf{d}} \times \boldsymbol{\omega} \tag{1}$$

Each component in Eq.1 is shown from Eq.2 to Eq.6,

$$\boldsymbol{\tau}_{\mathbf{cmd}} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} \tag{2}$$

$$\boldsymbol{K}_p = \begin{bmatrix} K_{p,x} & 0 & 0 \\ 0 & K_{p,y} & 0 \\ 0 & 0 & K_{p,z} \end{bmatrix} \tag{3}$$

$$\boldsymbol{K}_d = \begin{bmatrix} K_{d,x} & 0 & 0 \\ 0 & K_{d,y} & 0 \\ 0 & 0 & K_{d,z} \end{bmatrix} \tag{4}$$

$$\boldsymbol{\delta}\mathbf{q} = \begin{bmatrix} \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3}} \\ \delta q_4 \end{bmatrix} = \mathbf{q} \otimes \mathbf{q}_{\mathbf{cmd}}^{-\mathbf{1}} \tag{5}$$

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{6}$$

The PD controller is suggested by Crassidis and Markley in [1].

The closed-loop dynamics system with the PD controller implemented can be described by Eq.7, Eq.8, and Eq.9 shown below. For simplicity of the analysis, the dynamics used here assume perfect attitude knowledge and no perturbation forces.

$$\dot{\boldsymbol{\delta q}}_{1:3} = \frac{1}{2}[\boldsymbol{\delta q}_{1:3}\times]\boldsymbol{\omega} + \frac{1}{2}\delta q_4 \boldsymbol{\omega} \tag{7}$$

$$\dot{\delta q}_4 = -\frac{1}{2}\boldsymbol{\delta q}_{1:3}^T \boldsymbol{\omega} \tag{8}$$

$$\dot{\boldsymbol{\omega}} = -\mathbf{J}^{-1}([\boldsymbol{\omega}\times]\mathbf{J}\boldsymbol{\omega} + \mathbf{K}_p \times \boldsymbol{\delta q_{1:3}} + \mathbf{K_d} \times \boldsymbol{\omega}) \tag{9}$$

The closed-loop system is an error dynamics system with the only equilibrium point of $[\boldsymbol{\delta q}_{1:3}^T, \boldsymbol{\omega}^T] = \mathbf{0}$. The stability property of the equilibrium point of the closed-loop system can found using Lyapunov's direct method. Define a Lyapunov candidate function as follow,

$$V = \frac{1}{4}\boldsymbol{\omega}^T\mathbf{J}\boldsymbol{\omega} + \frac{1}{2}K_p\boldsymbol{\delta q}_{1:3}^T\boldsymbol{\delta q}_{1:3} + \frac{1}{2}K_p(1-\delta q_4)^2 \tag{10}$$

For simplicity of the Lyapunov analysis, K_p in Eq.10 and K_d in Eq.12 are taken as scalars. Note that the Lyapunov function satisfies the following conditions, **a**). V is energy-like which means that $V = 0$ only at the equilibrium point and $V > 0$ for all $\boldsymbol{\omega} \neq \mathbf{0}$ and $\boldsymbol{\delta q} \neq \mathbf{I}_q$. **b**). $\dot{V} \leq 0$. Condition **b** can be shown in the following,

$$\dot{V} = \frac{1}{2}\boldsymbol{\omega}^T\mathbf{J}\dot{\boldsymbol{\omega}} + K_p\boldsymbol{\delta q}_{1:3}^T\boldsymbol{\dot{\delta q}_{1:3}} - K_p(1-\delta q_4)\dot{\delta q}_4 \tag{11}$$

Substitute Eq. 7, Eq. 8, and Eq. 9 into Eq 11 gives

$$\dot{V} = -\frac{1}{2}(\boldsymbol{\omega}^T\boldsymbol{\delta q}_{1:3})[K_p + K_p\delta q_4 - K_p(1+\delta q_4)] - \frac{1}{2}K_d\boldsymbol{\omega}^T\boldsymbol{\omega} = -\frac{1}{2}K_d\boldsymbol{\omega}^T\boldsymbol{\omega} \tag{12}$$

Eq.12 shows that $\dot{V} \leq 0$ as long as $K_d > 0$. This result only shows the equilibrium point being simply stable. To further prove asymptotic stability of the equilibrium point, LaSalle Theorem is used as followed, let the set $\mathbf{V_0} = \{\boldsymbol{\delta q}, \boldsymbol{\omega}_e | \dot{V} = 0\}$. Find the largest invariant set, $\mathbf{M}$, in $\mathbf{V_0}$. According to Eq.12, the largest invariant set $\mathbf{M}$ can be found as followed,

$$\mathbf{V_0} = \{\boldsymbol{\delta q}, \boldsymbol{\omega}_e | \dot{V} = 0\} = \{\boldsymbol{\omega}(t) = \mathbf{0}\}, \forall t > 0. \tag{13}$$

$$\Rightarrow \dot{\boldsymbol{\omega}}(t) = \mathbf{0} \tag{14}$$

which according to Eq.9,

$$\dot{\boldsymbol{\omega}}(t) = \mathbf{0} \Rightarrow \boldsymbol{\delta q}_{1:3}(t) = \mathbf{0} \tag{15}$$

Note that the largest invariant set $\mathbf{M}$ in $\mathbf{V_0}$ is the equilibrium point itself. According to Corollary 4.1 by Khalil [8], the equilibrium point of the closed-loop system is asymptotically stable.

The next step is to determine the control gains, $\mathbf{K}_p$ and $\mathbf{K}_d$. There is no systematic to obtain the gains using the full nonlinear model directly. Hence, a simplified process is used here to obtain the control gain. Fig.11 shows the spacecraft rotation along its body y-axis with a commanded angle, θ_c.

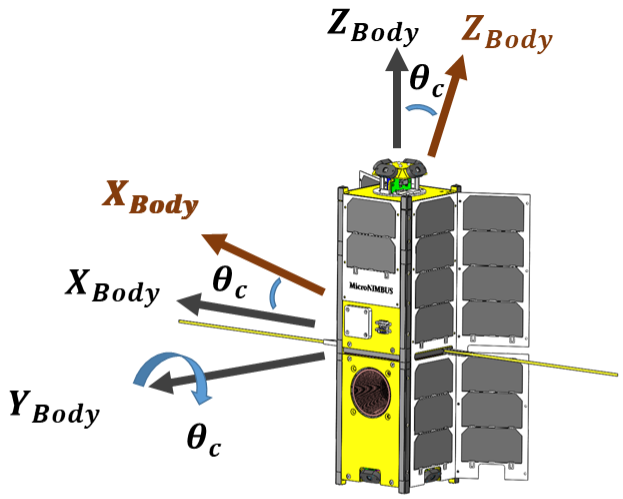

Figure 11: MicroNimbus Rotational Dynamics

The rotation dynamics can be describe by Eq.16 below,

$$\tau_{cmd} = J_y \ddot{\theta} \tag{16}$$

Now, let the τ_{cmd} be the PD control effort. It has the following form in Eq.17

$$\tau_{cmd} = K_p(\theta_{cmd} - \theta) + K_d(\dot{\theta}_{cmd} - \dot{\theta}) \tag{17}$$

The closed-loop system can be modeled as the system diagram shown in Fig.12,

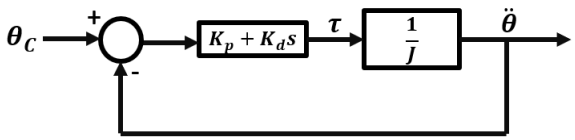

Figure 12: Control Diagram of Rotation Along y-axis

The closed-loop transfer function of the system is

$$G(s) = \frac{K_p + K_d s}{J_y s^2 + K_d s + k_p} \tag{18}$$

The denominator of Eq.18 is the characteristic equation of the system. It has a similar form as the characteristic equation of a second-order spring-damper system, which is shown in Eq.19.

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \tag{19}$$

$$s^2 + \frac{1}{J_y} K_d s + \frac{K_p}{J_y} = 0 \tag{20}$$

Hence, one can relate K_p and K_d gain to the damping ratio, ζ and natural frequency, ω_n of the system. Step response properties of the system such as rise time, maximum percent overshoot, and settling time can be adjusted by choices of ζ and ω_n. Eq.21 and Eq.22 show the calculation of K_p and K_d gain.

$$K_p = J_y \omega_n^2 \tag{21}$$

$$K_d = 2 J_y \zeta \omega_n \tag{22}$$

The same method can be applied to different body axes of the spacecraft which results in Eq.3 and Eq.4.

A step response analysis is performed using MATLAB. For the analysis, damping ratio is set to be 0.7 and the natural frequency is set to be 0.1. The PD compensator is shown in Eq.23.

$$C = 0.00059 \times \frac{1 + 12S}{1} \tag{23}$$

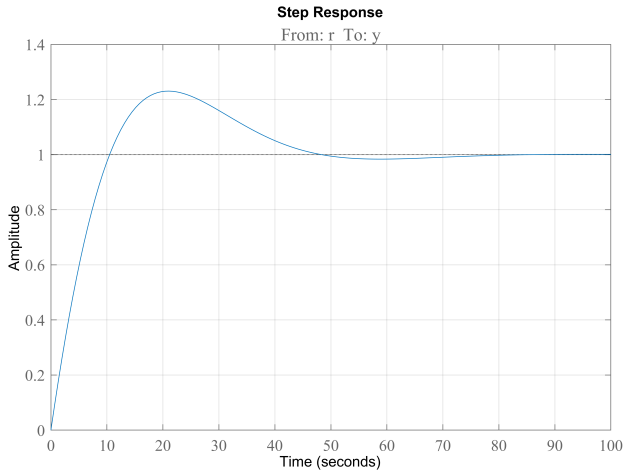

Figure 13: Step Response of the Closed-Loop System

Fig.13 shows a unit step response of the closed-loop system. The rise time is 8.05 seconds, and the settling time is 44.2 seconds. The maximum percent overshoot is 23.1%.

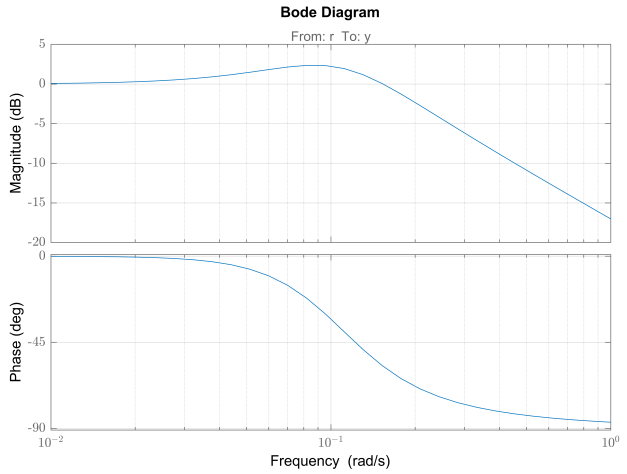

Figure 14: Bode Plot of the Closed-Loop System

Fig.14 shows the gain margin and phase margin of the system. The simulation setting yields a phase margin of 122^0. The gain margine is infinity since the system phase does not cross -180^0.

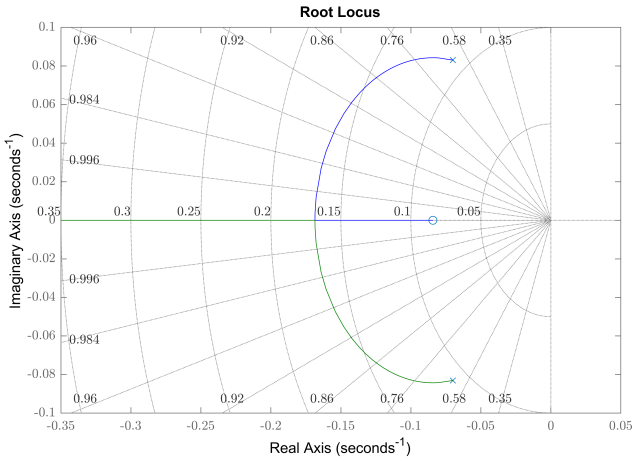

Figure 15: Root Locus of the Closed-Loop System

Fig.15 shows the root locus of the system. The closed-loop poles are stable, and the zero is in the right hand side of the plane. It implies that the system is minimum-phase.

3.3 Attitude Controllers

This section discusses each mission mode and its corresponding attitude controller. The procedure and method to construct specific pointing controller are heavily based on the requirements that set by the mission mode. Pointing controllers are constructed using a PD controller as the core control law. On the other hand, the B-dot controller uses a bang-bang control scheme.

Fig.16 shows isometric view of the MicroNimbus spacecraft. The body axes configuration is also shown in the figure.

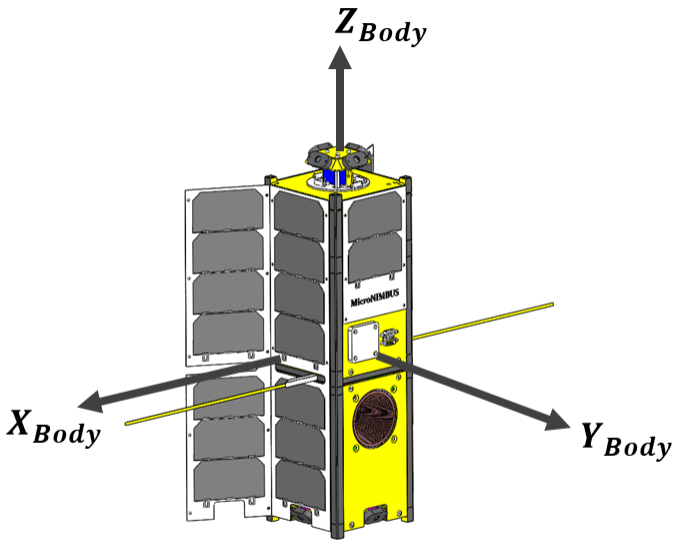

Figure 16: MicroNimbus Spacecraft

A series of Cartesian Frames express the dynamics of the spacecraft. F_B represents the spacecraft body frame in current time. F_R represents the command frame. The Commanded frame is the body frame at desired position at final time. F_L represents the LVLH frame. F_N represents the inertial frame which is the ECI frame.

At initial state, the spacecraft is at a random attitude. Its body attitude F_B with respect to F_N is found be on-board sensors measurements. The pointing controllers shall drive the spacecraft to some commanded attitudes, F_R. F_R is related to some reference frames. These reference frames are designed based on different pointing mode requirements. For example, during science mode, the reference frame is F_L. During sun charging mode, the reference frame is F_S which has an axis constantly pointing to the sun. Once F_R and its reference frame are established, the quaternion between these two frames is obtained and used as the control command input. Additionally, the angular rate between these two frames shall be zero to maintain spacecraft's constant attitude.

3.3.1 Nadir-Pointing Controller

The Nadir-Pointing controller is used for the science mode of the mission. The science mode requires the spacecraft to point its payload sensing axis directly down toward the earth surface (nadir). The science payload is a frequency-agile microwave radiometer that installed in the payload module[9]. Fig.16 shows the radiometer horn antenna right below the spacecraft body y-axis. The payload sensing axis is parallel to the spacecraft body y-axis. During the science mode, the Nadir-Pointing controller shall point the sensing axis nadir. Fig.17 shows the spacecraft attitude during the science mode.

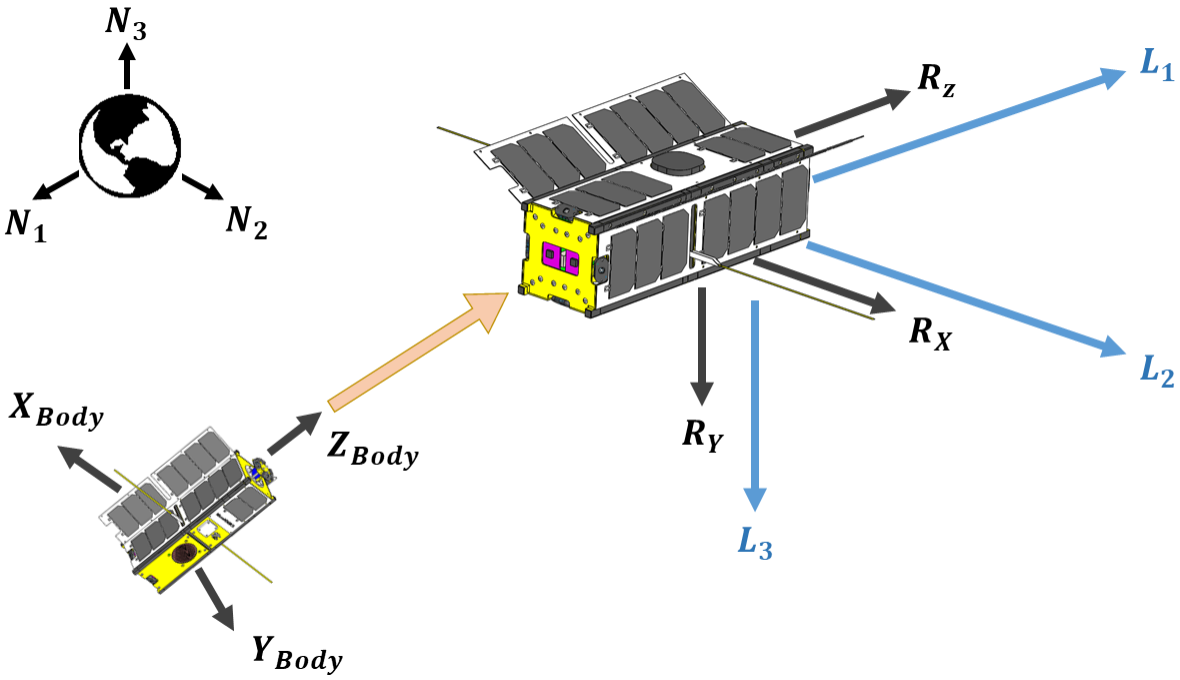

Figure 17: Nadir Pointing Scheme during Science Mode

In Fig.17, F_R represents the command frame. It is the spacecraft body frame at desired attitude with control action. The F_R align with F_L in a specific manner. R_Z axis aligns with L_1 axis, R_X axis aligns with L_2 axis, and R_Y axis aligns with L_3 axis. L_1 points toward local velocity vector, L_3 points local nadir, and L_2 points negative orbital angular momentum direction.

The advantage of using F_L as reference frame for F_R is that L_3 axis always points nadir during the flight. R_Y and L_3 axis alignment ensures sensing axis nadir pointing.

Eq.24, Eq.25, and Eq.26 show the frame transformation between F_L and F_R.

$$F_L = \mathbf{T}_X(-\frac{\pi}{2})\mathbf{T}_Y(-\frac{\pi}{2})F_R \tag{24}$$

$$\mathbf{T}_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & sin(\theta) \\ 0 & -sin(\theta) & cos(\theta) \end{bmatrix} \tag{25}$$

$$\mathbf{T}_y = \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{26}$$

Furthermore, the quaternion from F_R to F_L is,

$$\mathbf{q}_{LR,cmd} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} \tag{27}$$

$q_{LR,cmd}$ is the command quaternion for the Nadir-Pointing controller. It is also the step input for the dynamics system. When the controller reduces the quaternion error, $\delta\mathbf{q}_{LR}$ down to 0, the spacecraft will achieve the desired attitude shown in Fig.17. Eq.5 provides the calculation of $\delta\mathbf{q}_{LR}$.

After achieving the desired attitude, the spacecraft needs to maintain its attitude which means that the angular rate between F_R and F_L, $\boldsymbol{\omega}_{RL}$ need to be 0. Hence, $\boldsymbol{\omega}_{RL}$ is the derivative error term in the PD controller. Eq.28 shows the $\boldsymbol{\tau}_{cmd}$ for Nadir-Pointing controller.

$$\boldsymbol{\tau}_{\mathbf{cmd,Nadir-Points}} = -\mathbf{K}_p \times \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3LR}} - \mathbf{K}_\mathbf{d} \times \boldsymbol{\omega}_{RL} \tag{28}$$

3.3.2 Sun-Pointing Controller

During the sun charging mode, the spacecraft need to point its maximum solar panel covered body surface directly toward the sun. The maximum solar panel covered surface is directly opposite of the body y-axis. Fig.18 shows the spacecraft attitude during sun charging mode.

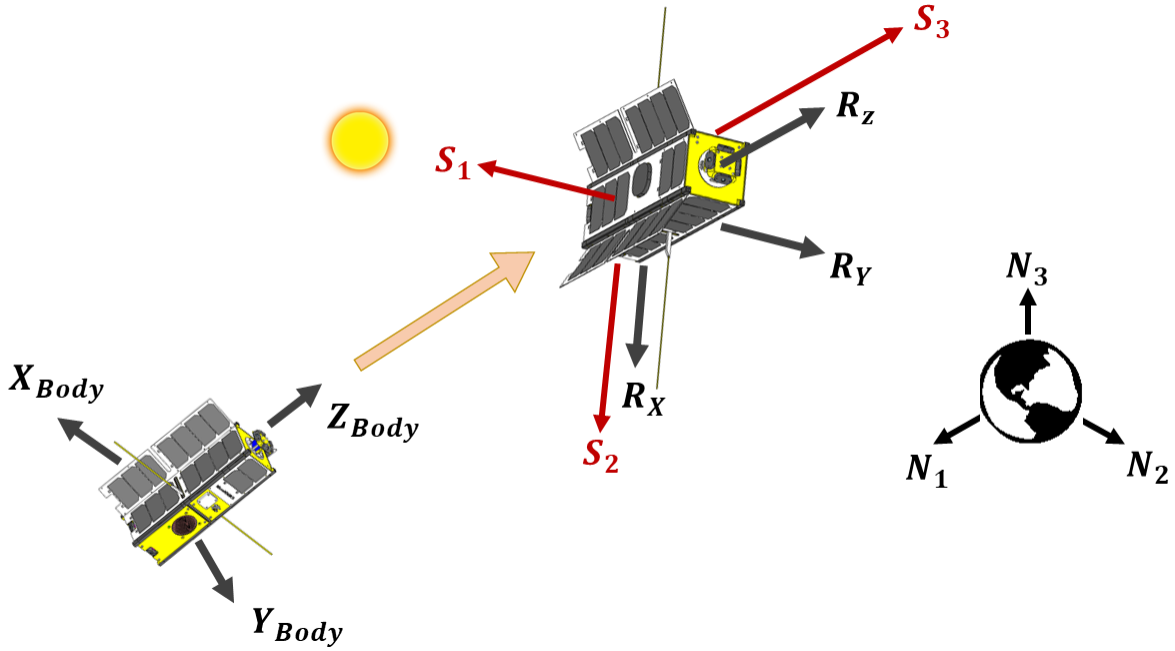

Figure 18: Sun Pointing Scheme during Charging Mode

F_S represents the sun pointing frame in space. S_1 axis points directly toward the sun. Direction of S_1 is provided by the sun vector measured by sun sensors. S_2 is an arbitrary axis in space that perpendicular to S_1. S_3 is the axis that complete the Cartesian frame. Components of F_S are expressed the in inertial frame, F_N. At desired attitude, the spacecraft shall align the $-R_Y$ body axis with S_1, and R_X with S_2.

The key to solve the problem is to obtain the frame transformation matrix between F_S and F_R. Fig.18 shows that at the desire state, $-R_Y$ aligns with S_1 and R_X aligns with S_2.

Essentially, R_X and S_2 can be viewed as the same vector but with different representations. The same idea applies to $-R_Y$ and S_1 as well. With the knowledge of these two axes with expressions in two frames, one can obtain the frame transformation matrix between these two frames by using the Triad method [1]. The quaternion between these two frames, q_{SR}, can be obtained from the matrix as well.

After achieving the desired attitude, the spacecraft need to maintain zero angular rate with respect to F_S which means $\boldsymbol{\omega}_{RS}$ should be zero. Eq.29 shows calculation to obtain $\boldsymbol{\omega}_{RS}$.

$$\dot{\mathbf{C}}_{\mathbf{SR}} = \mathbf{C}_{\mathbf{SR}}[\boldsymbol{\omega}_{RS}\times] \tag{29}$$

$\mathbf{C_{SR}}$ is the frame transformation matrix from F_R to F_S. $\dot{\mathbf{C}}_{\mathbf{SR}}$ is the time rate change of $\mathbf{C_{SR}}$ which is obtained through the Triad method at every time step. Eq.30 shows the $\dot{\mathbf{C}}_{\mathbf{SR}}$ calculation.

$$\dot{\mathbf{C}}_{\mathbf{SR}} = \frac{\mathbf{C}_{\mathbf{SR}}(t_k) - \mathbf{C}_{\mathbf{SR}}(t_{k-1})}{dt} \tag{30}$$

Eq.31 shows the $\boldsymbol{\tau}_{cmd}$ for Sun-Pointing controller.

$$\boldsymbol{\tau}_{\mathbf{cmd,Sun-Points}} = -\mathbf{K}_p \times \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3SR}} - \mathbf{K_d} \times \boldsymbol{\omega}_{\boldsymbol{RS}} \tag{31}$$

3.3.3 Slew Controller

A Slew controller is used to maximize the downlink time during ground pass. Fig.19 shows the slew control scheme during data downlink mode. The S-band patch antenna is located on the surface that perpendicular to R_Y axis. To maximize the data downlink time, spacecraft shall its R_Y axis directly toward the ground station once the spacecraft appears in the local horizon.

$\boldsymbol{\rho}$ represents the range vector from the ground station to the spacecraft. G_1 axis from F_G points directly opposite $\boldsymbol{\rho}$. G_2 is an arbitrary axis that perpendicular to G_1. G_3 is the third axis that complete the Cartesian frame. Similar to F_S, F_G is expressed in inertia frame, F_N.

Note that F_G rotates in inertial space as $\boldsymbol{\rho}$ changes when spacecraft passes over the ground station. The Slew controller shall maintain F_R's orientation constant with respect to F_G. The relation between F_R and F_G is shown in Fig.19.

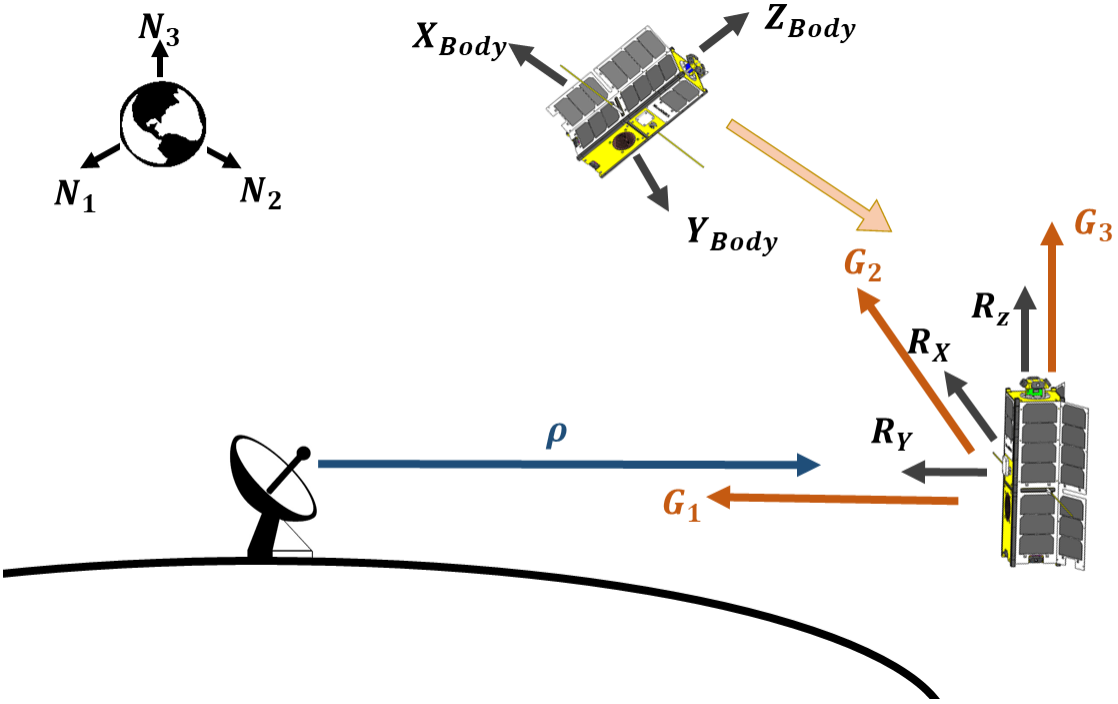

Figure 19: Slew Control Scheme during Ground Station Pass

Since G_2 and R_X point to the same direction during slew control, as well as G_1 and R_Y, one can use Triad method to obtain the frame transformation matrix, $\mathbf{C_{GR}}$, between F_G and F_R as well as $\mathbf{q_{GR}}$. $\mathbf{q_{GR}}$ is the command quaternion for the Slew controller. The angular rate, $\boldsymbol{\omega_{RG}}$, can be found from Eq.30 and Eq.29. Eq.32 shows the $\boldsymbol{\tau}_{cmd}$ for Slew controller.

$$\boldsymbol{\tau}_{\mathbf{cmd,Slew}} = -\mathbf{K}_p \times \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3GR}} - \mathbf{K_d} \times \boldsymbol{\omega}_{RG} \tag{32}$$

3.3.4 Axis-Command Controller

Axis-Command controller allows operators to command the spacecraft to any desired attitude. It is mainly used for sensor calibrations or health check. Fig.20 shows an arbitrary Axis-Command control scheme.

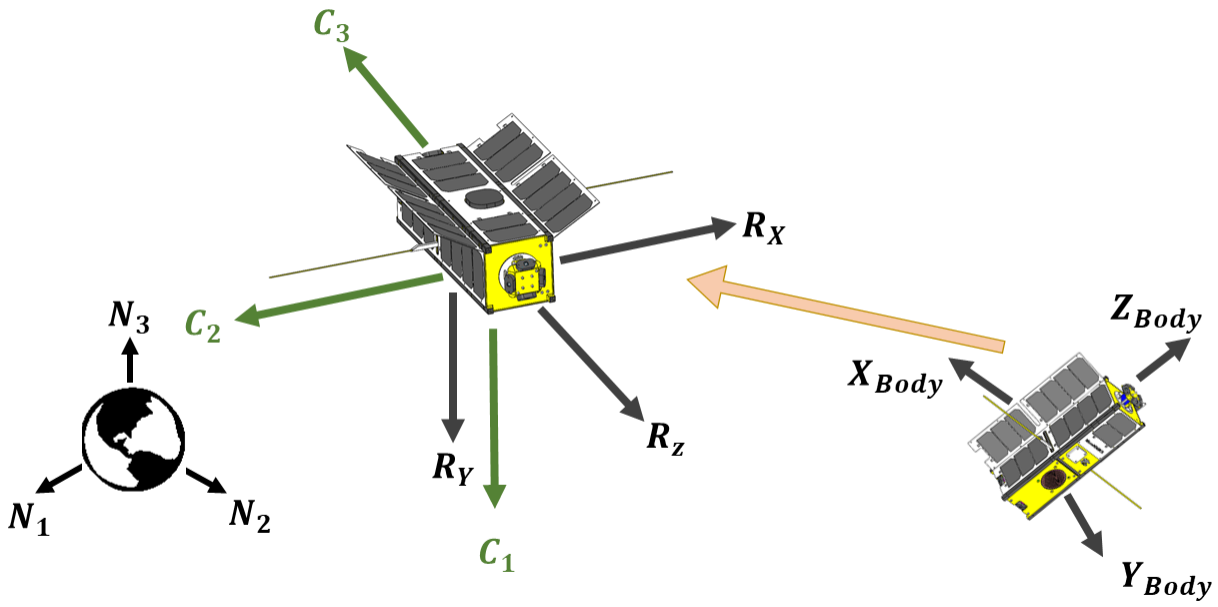

Figure 20: Axis-Command Control Scheme

F_C is the reference frame for F_R. C_1 and C_2 axes as well as F_R attitude with respect to F_C are defined by user command. The quaternion between F_C and F_R can be found by using Traid method. Eq.33 shows the $\boldsymbol{\tau}_{cmd}$ for control input. Additionally, the angular rate between these two frames, $\boldsymbol{\omega_{RC}}$, shall be zero.

$$\boldsymbol{\tau}_{\mathbf{cmd,Axis-Command}} = -\mathbf{K}_p \times \boldsymbol{\delta}\mathbf{q}_{\mathbf{1:3CR}} - \mathbf{K}_\mathbf{d} \times \boldsymbol{\omega}_{\boldsymbol{RC}} \tag{33}$$

3.3.5 B-Dot Controller

B-dot controller is a bang-bang controller that used for detumble mode. It uses three-axes magnetorquer to reduce the spacecraft random angular rate. The controller uses the torque generated by local magnetic filed vector and magnetorquer moment dipole to damp out spacecraft tumble motion.

The B-dot controller is easy to implement since it does not involve any frames quaternion calculations like the pointing controllers do. However, its actuation is slow. Its pointing accuracy is much less than that of the regular pointing controllers. In some orbits, spacecraft may not be fully controllable by B-dot control due to local magnetic field direction.

Eq.34 shows the mathematical model of the B-dot controller. i represents the body axes. $M_{max,i}$ is the maximum moment dipole can be generated by the i_{th} body axis.

$$M_i = \begin{cases} -M_{max,i} & if \dot{B}_i > 0 \\ M_{max,i} & if \dot{B}_i < 0 \end{cases}, i = x, y, z \tag{34}$$

$$\dot{\mathbf{B}} = \frac{\mathbf{B}(t_k) - \mathbf{B}(t_{k-1})}{dt} \tag{35}$$

The derivative of measured magnetic field, $\mathbf{B}$, is calculated in a discrete time manner.

4 Simulation & Results

The simulation is performed using software, 42, by NASA Goddard. To accurately depict the performance of each controller, the high fidelity orbit dynamics model includes two-body motion, aerodynamic forces and torques, spherical harmonic earth gravity gradient torque, solar pressure forces and torques, third body effect from moon. The simulation assumes perfect attitude knowledge.

The magnetic field model used for the simulation is the International Geomagnetic Reference Field (IGRF) with a degree of 8 and order of 8. Earth's gravity model has a degree of 4 and order of 4. Moon's gravity model has a degree of 2 and order of 0.

The simulation orbit has a periapsis of 401 km and a apoapsis of 408 km. Its eccentricity is 0.0002316. The orbit inclination is 51.6^0, and its right ascension of ascending node is 50^0. The orbit period is about 92 minutes.

Simulation time varies for each controller. Simulation time step is 0.2 second. The simulation date is set at April, 4^{th}, 2018. The data output is set at 1-second interval.

The mass moment of inertial, natural frequency, damping ratio, and control gains are listed in Table.5

Due to the lack of sensor model and actuator model in current simulation, the attitude knowledge is assumed to be perfect, and the command actuations are assumed to be prefect and instantaneous.

Table 5: Controller Gain Table

	Mass Moment of Inertia $[kg - m^2]$	$\boldsymbol{\zeta}$	$\boldsymbol{\omega}_n$ $[rad/sec]$	K_p	K_d
X	0.05071	0.7	0.1	0.0005071	0.007099
Y	0.04604	0.7	0.1	0.0004604	0.0064456
Z	0.02985	1	0.5	0.0074625	0.02985

4.1 B-dot Controller

The B-dot controller is used for detumble mode. The controller does not complete eliminate the spacecraft's angular rate. It can only reduce the spacecraft's body rotation to certain value. When the spacecraft body angular rate reduces down to a predefined threshold, the detumble mode is considered complete. Three initial angular rate conditions are used for simulation. Table.6 shows the initial conditions for the detumble simulation. An angular rate is preset on each body axis after deployment. The simulation duration is 16000 second.

Table 6: B-dot Controller Detumble Simulation Initial Conditions

	$\omega_{X,0}$ $[^0/sec]$	$\omega_{Y,0}$ $[^0/sec]$	$\omega_{Z,0}$ $[^0/sec]$
Case 1	-5	5	5
Case 2	5	5	5
Case 3	-4	5	7

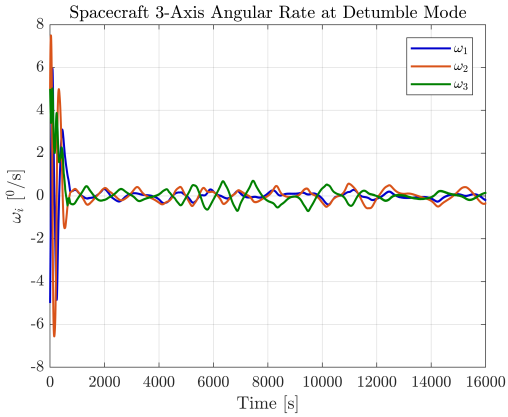

Figure 21: Detumble Simulation Case 1

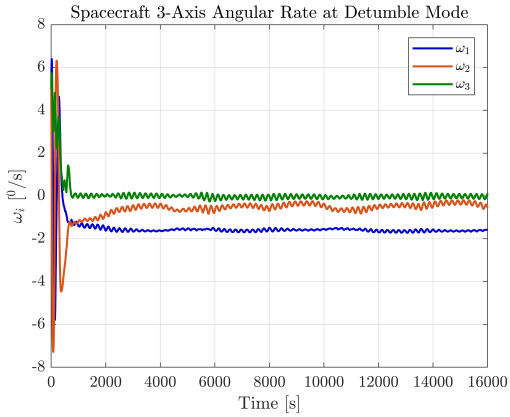

Figure 22: Detumble Simulation Case 2

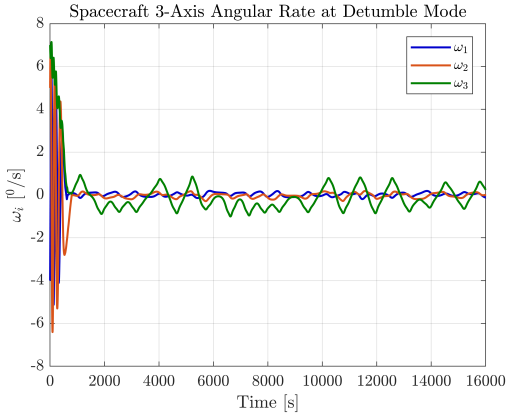

Figure 23: Detumble Simulation Case 3

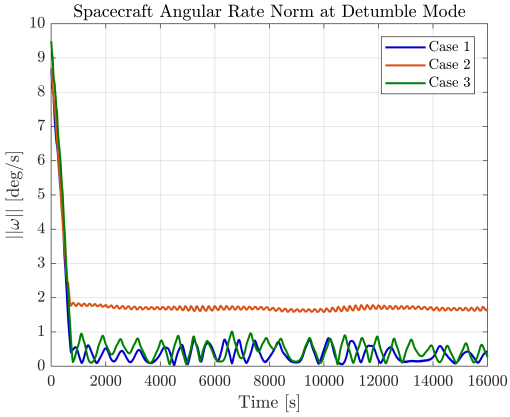

Figure 24: Angular Rate Magnitude

Fig.21 to Fig.24 show the spacecraft angular rates during detumble. Fig.21, Fig.22, and Fig.23 show the body axis angular rate history during detumble. Fig.24 shows the norms of total angular rate for all three cases. The figures show that the angular rate of each body axis reaches a steady state at around 3000 second with B-dot detumble control. Each simulation case show a different steady state result. These indicate that the detumble control performance is not only driven by the controller but also driven by initial conditions. Case 1 and Case 3 both show that the spacecraft reaches steady state after 2000 seconds, and

the biggest angular rate residuals remain in body z-axis. This mean that the body z-axis is parallel to the local magnetic field vector at steady state. No effective torque can be produced to slow down the rotation. The condition is shown in Fig.25. The red color axis is local magnetic field vector which is parallel to the body z-axis, $\hat{b}_3$. Case 2 shows a worst case. A large angular rate residual remains at body x-axis. The spacecraft is rotating around its body x-axis with a $-2^0/sec$ rate. Again, the x-axis is parallel to local magnetic field vector. No torque can be generated to slow down the rotation. Fig.24 shows the magnitude of angular rate, $||\boldsymbol{\omega}||$, for each detumble case. At steady state, the minimum $||\boldsymbol{\omega}||$ of case 2 is $1.6^0/sec$, and the minimum $||\boldsymbol{\omega}||$ for case 1 and case 3 are $0.078^0/sec$ and $0.117^0/sec$ respectively.

One of the reasons the bang-bang controller is selected for detumble is physical hardware communication constraint. If a PWM functionality is available for torque rod control, a proportional controller can be used. The proportional control requires to vary the magnetorquer moment dipole. According to Leomanni[10], a proportional controller can further reduce steady state angular rate.

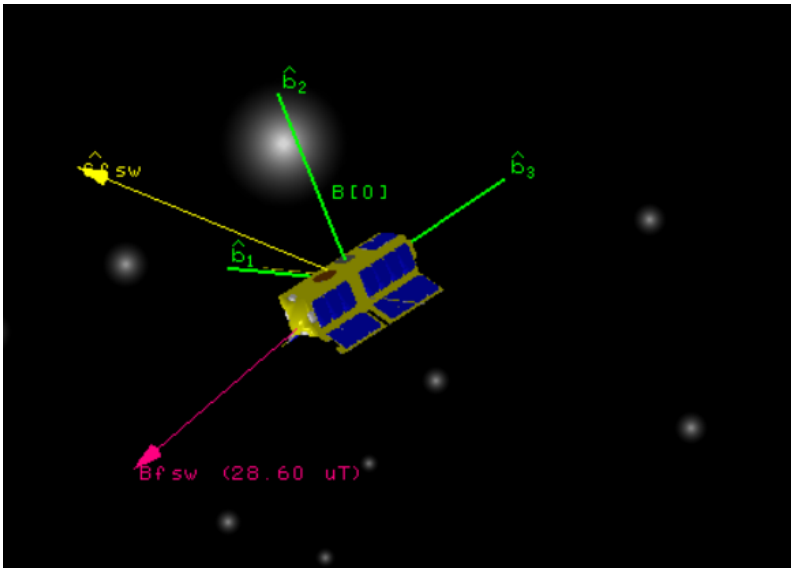

Figure 25: B-dot Controller Detumble

4.2 Nadir-Pointing Controller

The Nadir-Pointing controller is used for science mode. The quaternion command is given by Eq.27. Pointing axis error is defined as the angle between body y-axis, R_Y and the l_3 axis. Fig.17 shows the axes configuration. The simulation duration is 6000 seconds. The control command starts at 5 second. The simulation assumes perfect attitude knowledge during the whole orbit. Fig.26 shows the spacecraft at science mode during the simulation.

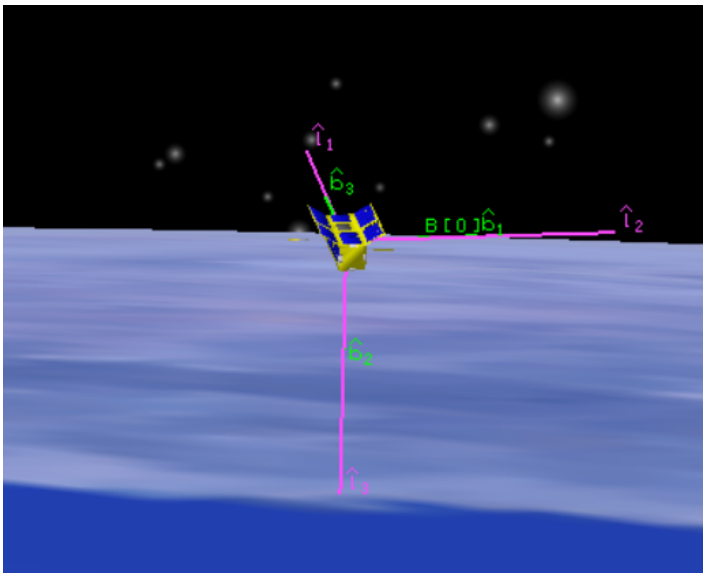

Figure 26: Nadir-Pointing Attitude in Simulation

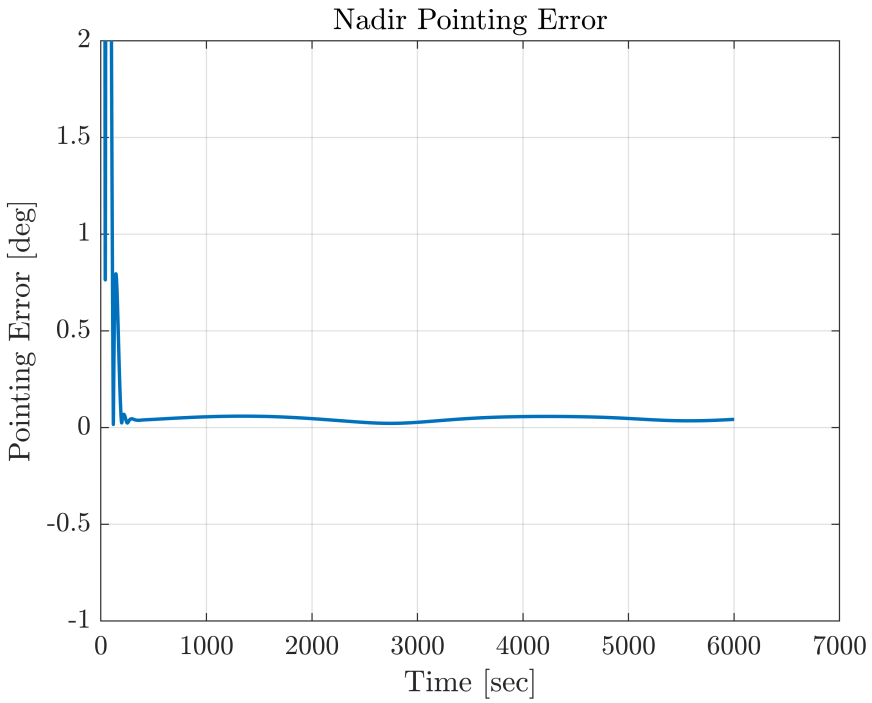

Figure 27: Nadir-Pointing Axis Error

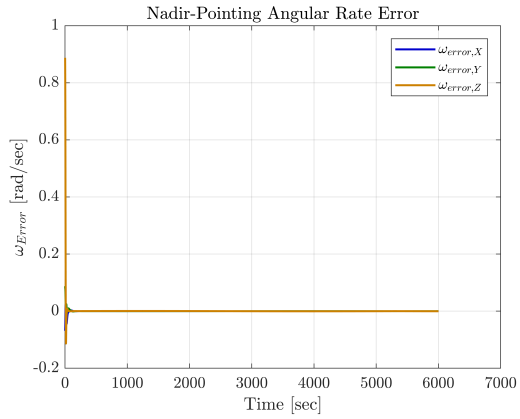

Figure 28: Nadir Pointing Angular Rate Error

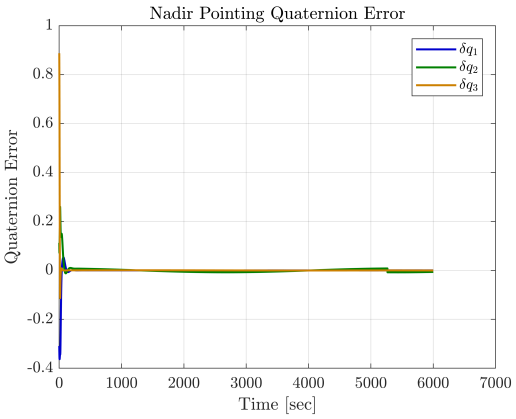

Figure 29: Nadir Pointing Quaternion Error

Fig.27 and Fig.29 show the pointing error and the quaternion error. The maximum pointing error is about 0.06^0 in steady state. The quaternion error of each axis is close to zero. Fig.28 shows the angular rate between F_R and F_L. The Nadir-Pointing controller preforms well in terms of pointing accuracy.

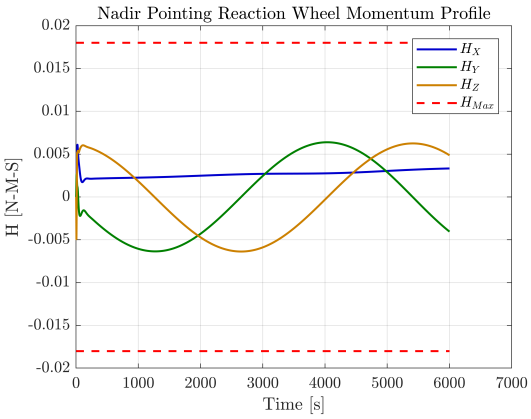

Figure 30: Nadir Pointing Reaction Wheel Angular Momentum

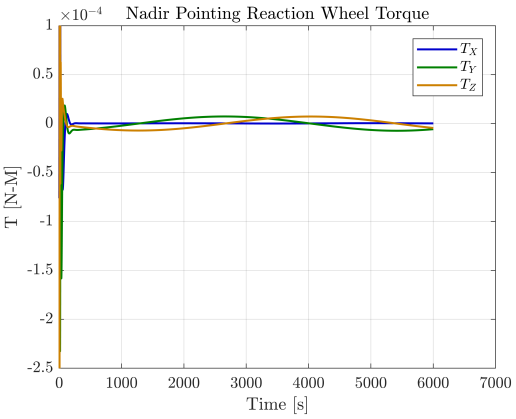

Figure 31: Nadir Pointing Reaction Wheel Angular Torque

Fig.30 and Fig.31 show the reaction wheels command torques and angular momenta. Fig.31 shows that the command torque for each wheel. Even the required torque is small, the angular momenta build up quickly. The current simulation uses 0 Nms nominal momentum for each reaction wheels. Technically, the reaction wheel should operates at 0.01 Nms nominal momentum. Reaction wheel saturation will occur with 0.01 Nms nominal momentum. The spacecraft loss control authority during wheel saturation. A desaturation mechanism is needed to unload the reaction wheels' momenta. The desaturation mechanism shall be implemented using the three-axes magnetorquers.

4.3 Sun-Pointing Controller

The Sun-Pointing controller is used for sun charging mode. Fig.18 shows that the controller helps the spacecraft to point its negative y-axis toward the sun vector to maximize charging area. Fig.32 and Fig.33 show the spacecraft at sun pointing attitude during the simulation. $-\hat{b}_2$ axis is parallel to sun vector, $\odot$.

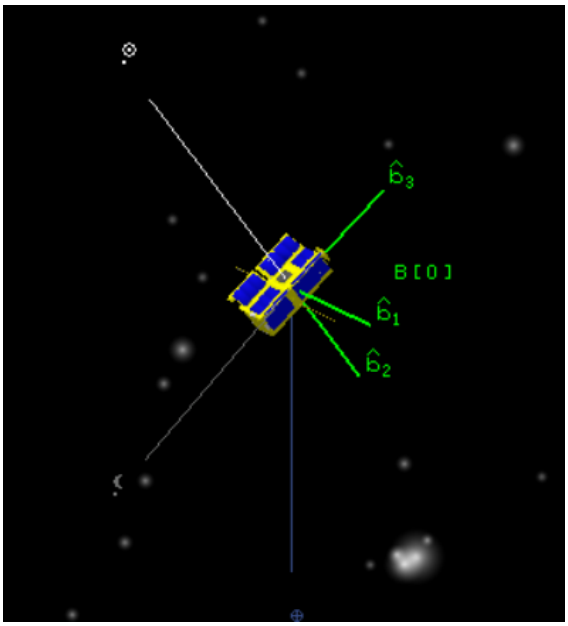

Figure 32: Sun Pointing Attitude Simulation in 42

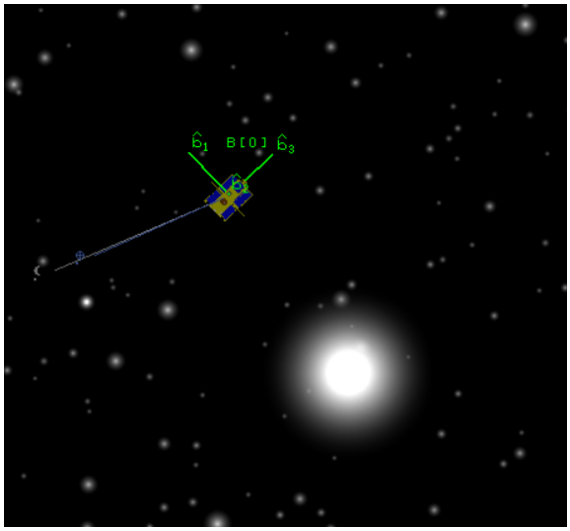

Figure 33: Sun Pointing Controller Simulation in 42

The simulation duration is 4000 seconds. The control command starts at 5 seconds. The simulation assumes perfect knowledge of spacecraft's attitude and sun vector. The pointing error is defined as the angle between body -y-axis, R_Y, and the sun vector, S_1.

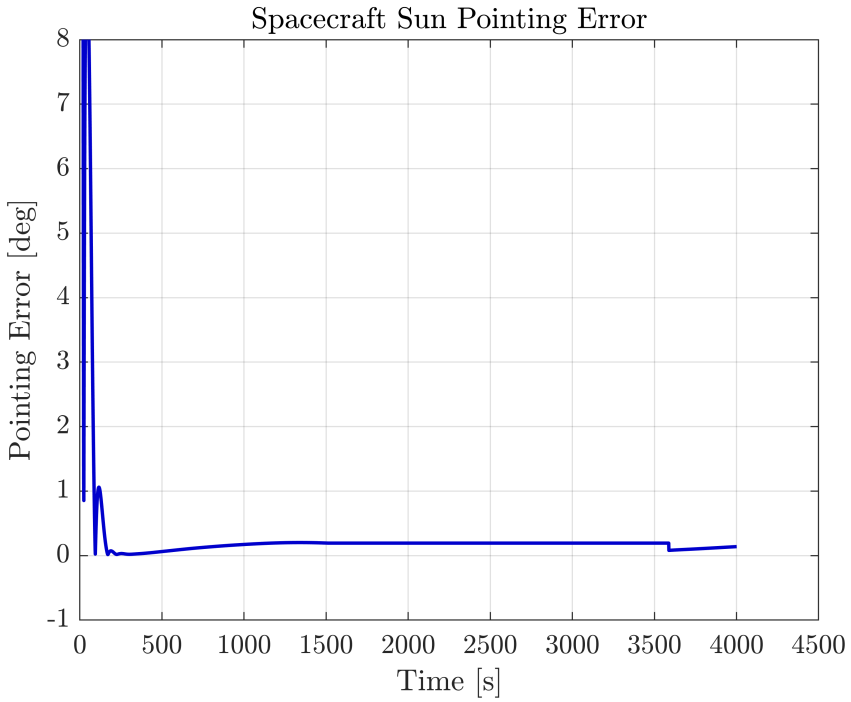

Figure 34: Sun-Pointing Axis Error

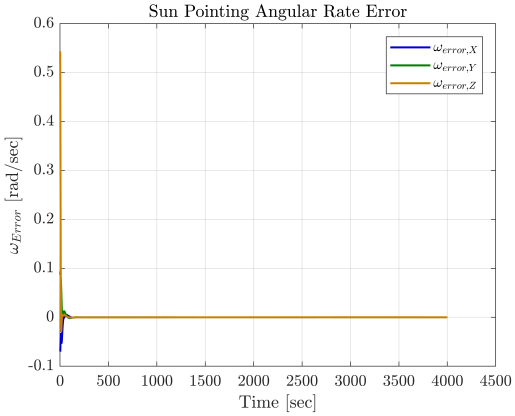

Figure 35: Sun Pointing Angular Rate Error

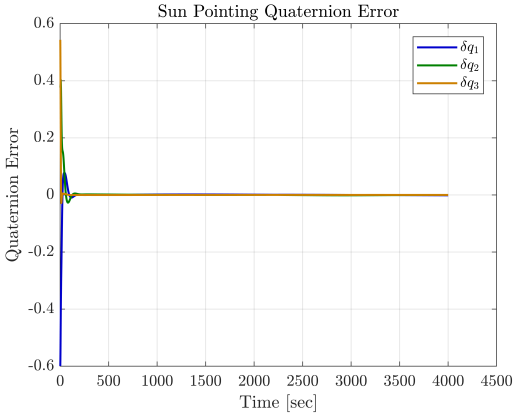

Figure 36: Sun Pointing Quaternion Error

Fig.34 shows the pointing error during sun charging mode. The maximum error is 0.19^0 at steady state. Fig.36 shows the quaternion errors. They are close to zero at steady state. The three-axes angular rates are also close to zero at steady state.

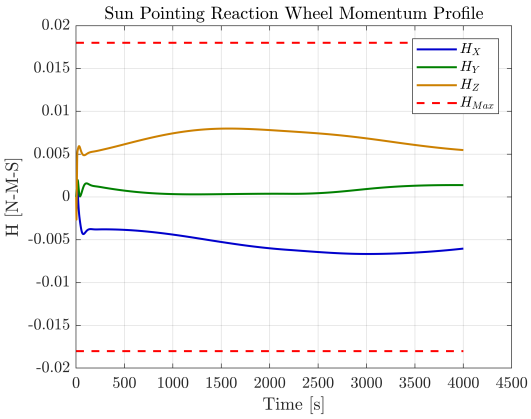

Figure 37: Sun Pointing Reaction Wheel Angular Momentum

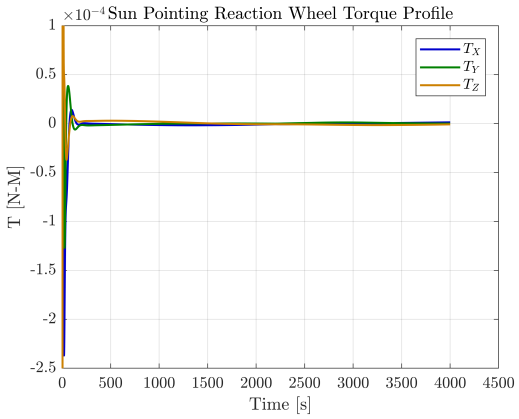

Figure 38: Sun Pointing Reaction Wheel Angular Torque

Fig.37 and Fog.38 show the angular momentum and torque of each reaction wheel. The nominal angular momentum of each reaction wheel is currently set as zero. Fig.37 shows that the angular momenta required for sun charging mode are less than those for science mode for the current designed F_S. However, the reaction wheel will saturate if the nominal angular momentum is set to be $0.01Nms$. A desaturation mechanism is highly recommended.

4.4 Slew Controller

Slew controller is used to maximize data downlink time during ground station passing. The simulation time is 1500 seconds. The orbit right ascension of ascending node is adjusted to 160^0 to provide a ground track that passes Georgia Tech ground station. The location of Georgia Tech ground station is 33.77^0N and 84.49^0W with 300 meters above sea level. These information is used to calculate the location vector with respect to F_N in 42. Fig.39 shows the ground track coverage of the slew control simulation. The ground track passes over Georgia Tech ground station. Fig.40 shows the spacecraft slew control when passing over the ground station. The F_b aligns with F_G instead of F_L. However, F_G is not showed in the figure.

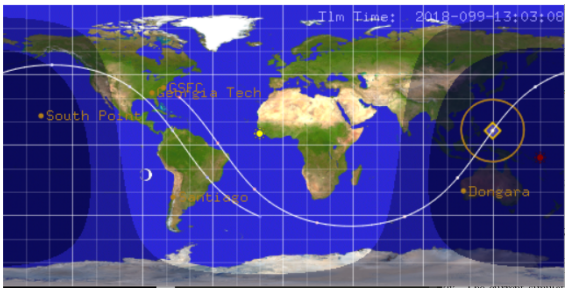

Figure 39: Slew Control Ground Track Simulation

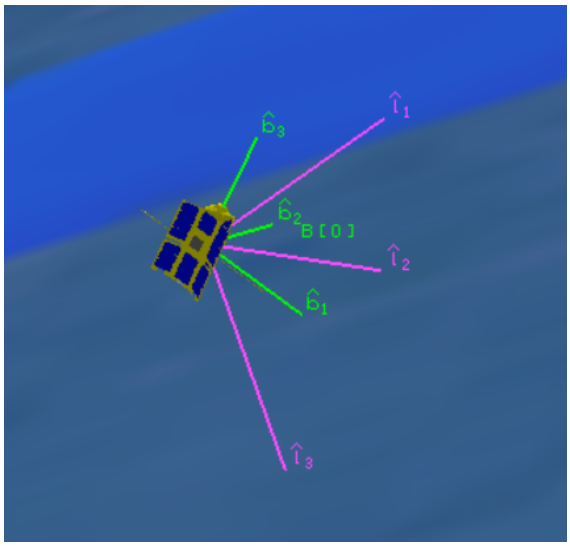

Figure 40: Slew Control Simulation

Fig.41 shows the pointing error when using the Slew controller. The error angle is defined as the angle between $-R_y$ axis and G_1 axis that shown in Fig.19. The maximum error is 0.28^0 at steady state. Fig.43 and Fig.42 show that the quaternion error and angular rate are relatively zero during slew control.

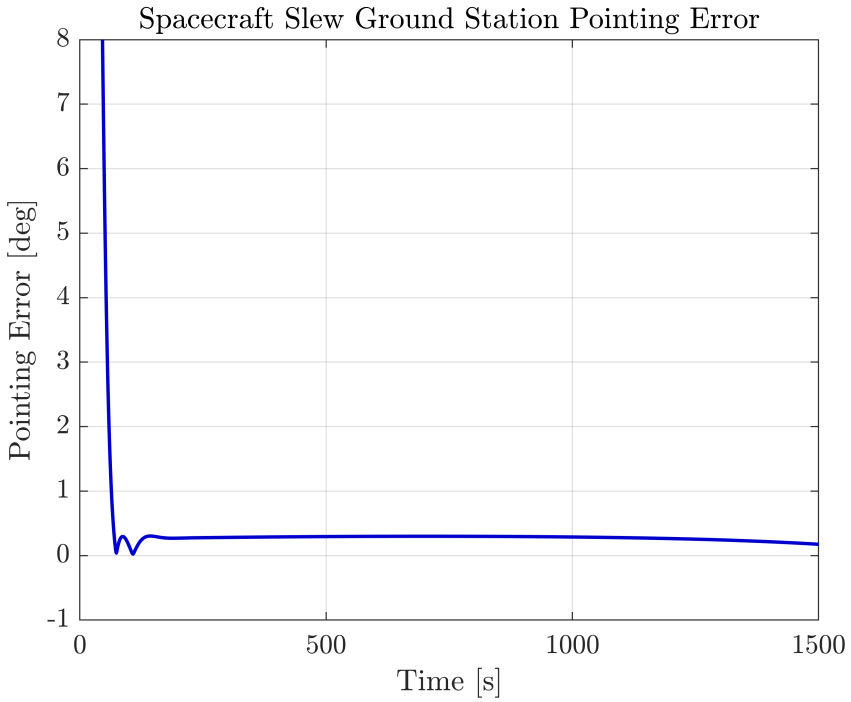

Figure 41: Slew Ground Station Pointing Axis Error

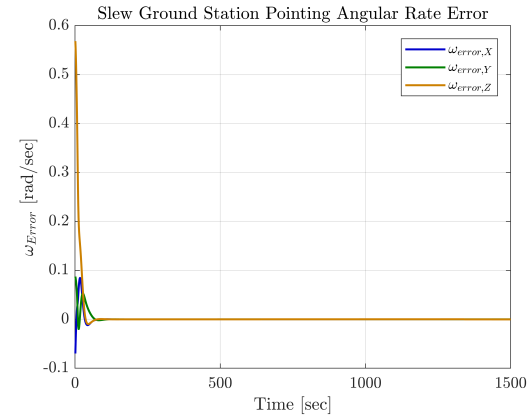

Figure 42: Slew Ground Station Pointing Angular Rate Error

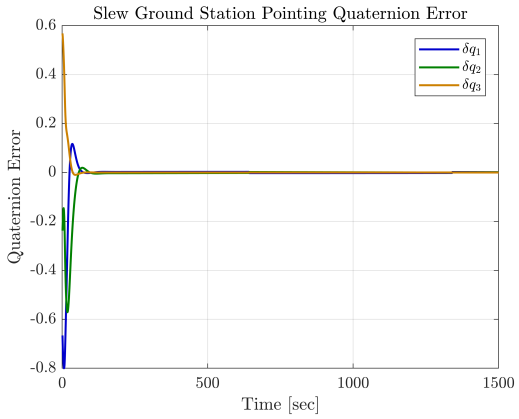

Figure 43: Slew Ground Station Pointing Quaternion Error

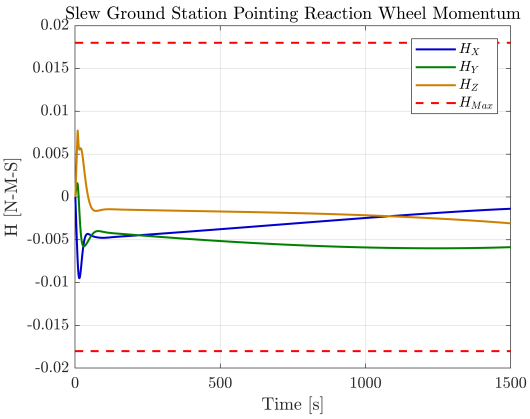

Figure 44: Slew Ground Station Pointing Reaction Wheel Angular Momentum

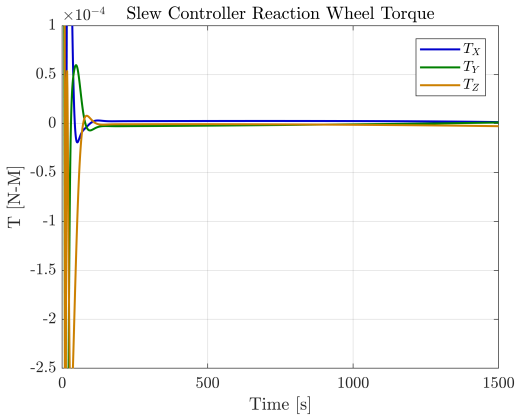

Figure 45: Slew Ground Station Pointing Reaction Wheel Angular Torque

Fig.44 and Fig.45 show the angular momenta and torques of the reaction wheel during slew. The angular momenta required for slew control is smaller than those required for Nadir-Pointing control.

4.5 Axis-Command Controller

The purpose of the Axis-Command controller is to give the ability to operators to orient the spacecraft to any desired attitude. It can be used for sensor calibration or spacecraft health check. The current design uses F_L for reference frame. Operators can control the spacecraft attitude with respect to LVLH frame. Fig.46 shows the Axis-Command control for the simulation. The simulation time is 6000 seconds. Pointing error is not provided since the current command attitude is arbitrary. It is up to users to define the pointing error based on the choice of primary vector.

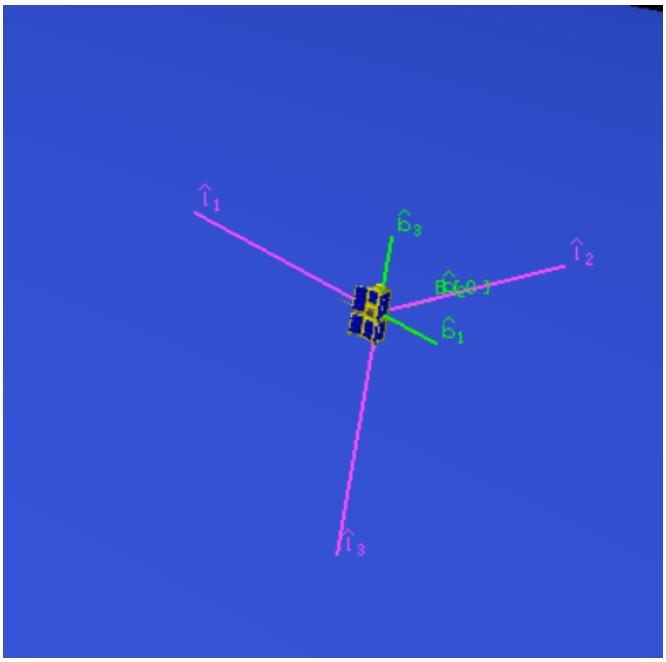

Figure 46: Axis-Command Pointing Sample

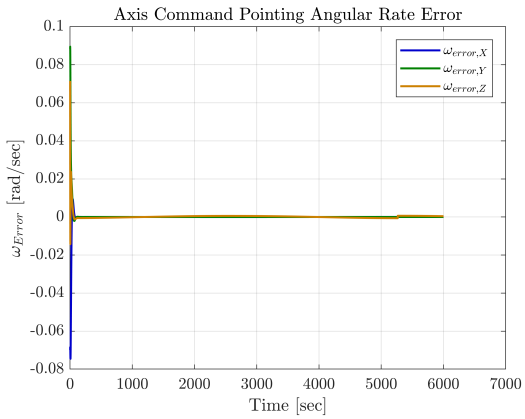

Figure 47: Axis Command Pointing Angular Rate Error

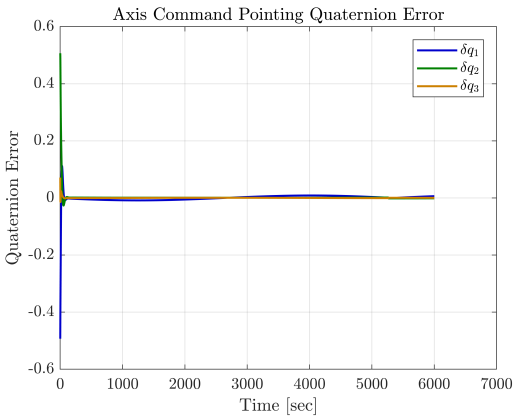

Figure 48: Axis Command Pointing Quaternion Error

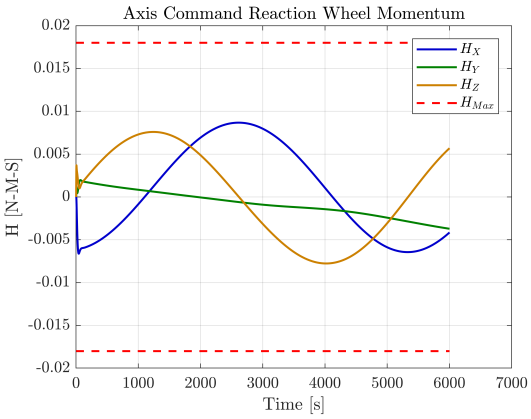

Figure 49: Axis Command Pointing Reaction Wheel Angular Momentum

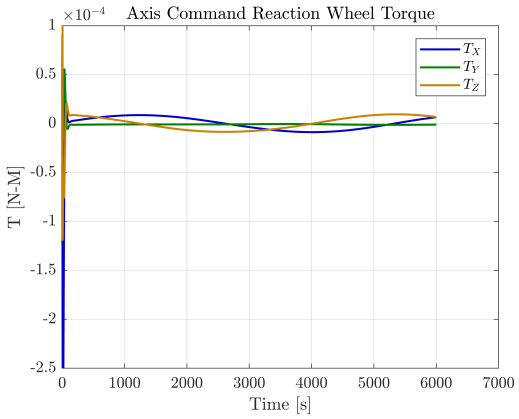

Figure 50: Axis Commadn Pointing Reaction Wheel Angular Torque

Fig.49 and Fig.50 show the angular momenta and commanded torques of the reaction wheels. They are similar to the results provided the Nadir-Pointing controller. This is expected since the Nadir-Pointing controller is a special case of the Axis-Command controller.

5 Pseudocode

This section details the algorithm and variables used for each attitude controller. The simulation version of the algorithm is written in *C*, and it is located on the mission server drive[1].

5.1 B-Dot Controller

Table 7: Variables for B-Dot Controller Variables

Variables	Descriptions
dt	Simulation time step size, scalar.
bvb	Local magnetic field vector measured in $\mathbf{F_b}$, 3×1 Vector
bvbold	Local magnetic field vector measured in $\mathbf{F_b}$ from previous step, 3×1 Vector
Bdot	Time rate change of local magnetic field vector, 3×1 Vector
Mmtbcmd	Commanded magnetic moment dipoles of the magnetorquers, 3×1 Vector
Mmtbmax	Maximum magnetic moment dipoles of the magnetorquers, 3×1 Vector

[1] Z:/16043048-MicroNimbus/Documents/ADC/sim/controller

Algorithm 1 B-dot Controller

procedure B-DOT CONTROLLER FOR DETUMBLE
    set **bvb**, **bvbold**, **Bdot**
    Compute **Bdot** for each body axis as follow:
    for $i = 1:3$ **do**
        Bdot[i] = (**bvb**[i] - **bvbold**[i]) / dt
    Compute control effort using bang-bang control as follow:
    for $i = 1:3$ **do**
        if Bdot[i] > 0 **then**
            Mmtbcmd[i] = -1 $\times$ **Mmtbmax**[i]
        else if Bdot[i] < 0 **then**
            Mmtbcmd[i] = 1 $\times$ **Mmtbmax**[i]
    Update **bvbold** for the next time step as follow:
    for $i = 1:3$ **do**
        bvbold[i] = **bvb**[i]

5.2 Nadir-Pointing Controller

Table 8: Variables for Nadir-Pointing Controller Variables

Variables	Descriptions
dt	Simulation time step size, scalar.
CLN	Directional cosine matrix from $\mathbf{F_N}$ to $\mathbf{F_L}$, 3×3 Matrix
qln	Quaternion from $\mathbf{F_N}$ to $\mathbf{F_L}$, 4×1 Vector
qrl	Quaternion from $\mathbf{F_R}$ to $\mathbf{F_L}$, 4×1 Vector
qrn	Quaternion for $\mathbf{F_N}$ to $\mathbf{F_L}$, 4×1 Vector
qbr	Quaternion for $\mathbf{F_R}$ to $\mathbf{F_B}$, 4×1 Vector
$\boldsymbol{\omega ln}$	Angular rate of $\mathbf{F_L}$ w.r.t $\mathbf{F_N}$, 3×1 Vector
$\boldsymbol{\omega rn}$	Angular rate of $\mathbf{F_R}$ w.r.t $\mathbf{F_L}$, 3×1 Vector
$\boldsymbol{\omega err}$	Angular rate error, 3×1 Vector
therr	Quaternion error in angular form , 3×1 Vector
Tcmd	Control torques, 3×1 Vector
Twhlcmd	Reaction wheel command torques, 3×1 Vector
Kp	Proportional control gains, 3×1 Vector
Kr	Derivative control gains, 3×1 Vector

Algorithm 2 Nadir-Pointing Controller

procedure NADIR-POINTING CONTROLLER
    set **qln**
    set command quaternion **qrl** = $[0.5, 0.5, 0.5, 0.5]$
    Compute **qln** from the **CLN**
    Compute command quaternion **qrn** using **qln** and **qrl**
    Compute command $\boldsymbol{\omega rn}$ using $\boldsymbol{\omega ln}$ and **qrn**
    Compute error quaternion **qbr** using **qrn** and **qbn**
    Convert **qbr** into angular form **therr**
    Compute angular rate error $\boldsymbol{\omega err}$ as follow:
    for $i = 1 : 3$ **do**
        $\boldsymbol{\omega err}[i]$ = $\boldsymbol{\omega bn}[i]$ - $\boldsymbol{\omega rn}[i]$
    Compute the command torque as follow:
    for $i = 1 : 3$ **do**
        $\mathbf{Tcmd}[i] = -\mathbf{Kp}[i] \times \mathbf{therr}[i] - \mathbf{Kr}[i] \times \boldsymbol{\omega err}[i]$
    Input command torque to reaction wheels as follow:
    for $i = 1 : 3$ **do**
        $\mathbf{Twhlcmd}[i] = -\mathbf{Tcmd}[i]$

5.3 Sun-Pointing Controller

Table 9: Variables for Sun-Pointing Controller Variables

Variables	Descriptions
dt	Simulation time step size, scalar.
CRN	Directional cosine matrix from $\mathbf{F_N}$ to $\mathbf{F_R}$, 3×3 Matrix
OldCRN	**CRN** from previous time step, 3×3 Matrix
Cdot	Time rate change of **CRN**, 3×3 Matrix
KN	An inertial vector in $\mathbf{F_N}$, 3×1 Vector
R2N	Normal vector to **svn** in $\mathbf{F_N}$, 3×1 Vector
svn	Sun vector in $\mathbf{F_N}$, 3×1 Vector
PriVR	Primary vector in $\mathbf{F_R}$, 3×1 Vector
SecVR	Secondary vector in $\mathbf{F_R}$, 3×1 Vector
PriVN	Primary vector in $\mathbf{F_N}$, 3×1 Vector
SecVN	Secondary vector in $\mathbf{F_N}$, 3×1 Vector
qbn	Quaternion from $\mathbf{F_N}$ to $\mathbf{F_B}$, 4×1 Vector
qrn	Quaternion for $\mathbf{F_N}$ to $\mathbf{F_L}$, 4×1 Vector
qbr	Quaternion for $\mathbf{F_R}$ to $\mathbf{F_B}$, 4×1 Vector
$\boldsymbol{\omega bn}$	Angular rate of $\mathbf{F_B}$ w.r.t $\mathbf{F_N}$, 3×1 Vector
$\boldsymbol{\omega rn}$	Angular rate of $\mathbf{F_R}$ w.r.t $\mathbf{F_L}$, 3×1 Vector
$\boldsymbol{\omega err}$	Angular rate error, 3×1 Vector
therr	Quaternion error in angular form , 3×1 Vector
Tcmd	Control torques, 3×1 Vector
Twhlcmd	Reaction wheel command torques, 3×1 Vector
Kp	Proportional control gains, 3×1 Vector
Kr	Derivative control gains, 3×1 Vector

Algorithm 3 Sun Pointing Controller

procedure SUN-POINTING CONTROLLER
    set **CRN**, **Cdot**, **R2N**
    set inertial vector **KN** $= [0, 0, 1]$
    set primary vector **PriVR** $= [0, -1, 0]$
    set secondary vector **SecVR** $= [0, 0, 1]$
    Compute **R2N** by cross product **KN** $\times$ **svn**
    Assign primary vector **PriVN** as follow:
    for 1=1:3 **do**
        PriVN$[i]$ = **svn**$[i]$
    Assign secondary vector **SecVN** as follow:
    for 1=1:3 **do**
        SecVN$[i]$ = **R2N**$[i]$
    Compute **CRN** by TRIAD Method using **PriVR**, **SecVR**,**PriVN**, **SecVN**
    Compute command quaternion **qrn** from **CRN**
    Compute **Cdot** as follow:
    for i=1:3 **do**
        for j=1:3 **do**
            Cdot$[i][j]$ = (**CRN**$[i][j]$ - **OldCRN**$[i][j]$) / dt
    Compute $\boldsymbol{\omega rn}$ using **Cdot** and **CRN**
    Update **OldCRN** as follow:
    for i=1:3 **do**
        for j=1:3 **do**
            OldCRN$[i][j]$ = **CRN**$[i][j]$
    Compute quaternion error **qbr** using **qbn** and **qrn**
    Convert **qbr** into angular form **therr**
    Compute angular rate error $\boldsymbol{\omega err}$ as follow:
    for $i = 1 : 3$ **do**
        $\boldsymbol{\omega err}[i]$ = $\boldsymbol{\omega bn}[i]$ - $\boldsymbol{\omega rn}[i]$
    Compute the command torque as follow:
    for $i = 1 : 3$ **do**
        Tcmd$[i] = -$**Kp**$[i] \times$ **therr**$[i] -$**Kr**$[i] \times \boldsymbol{\omega err}[i]$
    Input command torque to reaction wheels as follow:
    for $i = 1 : 3$ **do**
        Twhlcmd$[i] = -$**Tcmd**$[i]$

5.4 Slew Controller

Table 10: Variables for Slew Controller Variables

Variables	Descriptions
dt	Simulation time step size, scalar.
CRN	Directional cosine matrix from $\mathbf{F_N}$ to $\mathbf{F_R}$, 3×3 Matrix
CWN	Directional cosine matrix from $\mathbf{F_N}$ to $\mathbf{F_W}$, 3×3 Matrix
OldCRN	**CRN** from previous time step, 3×3 Matrix
Cdot	Time rate change of **CRN**, 3×3 Matrix
gsECEF	Ground station location in $\mathbf{F_W}$, 3×1 Vector
gsECI	Ground station location in $\mathbf{F_N}$, 3×1 Vector
Lat	Ground station latitude in radian, Scalar
Lon	Ground station longitude in radian, Scalar
Alt	Ground station sea level altitude in meter, Scalar
range	$\boldsymbol{\rho}$ in $\mathbf{F_N}$, 3×1 Vector
range_unit	$\boldsymbol{\rho}$ unit vector in $\mathbf{F_N}$, 3×1 Vector
PosN	Spacecraft location in $\mathbf{F_N}$, 3×1 Vector
KN	An inertial vector in $\mathbf{F_N}$, 3×1 Vector
R2N	Normal vector to **svn** in $\mathbf{F_N}$, 3×1 Vector
PriVR	Primary vector in $\mathbf{F_R}$, 3×1 Vector
SecVR	Secondary vector in $\mathbf{F_R}$, 3×1 Vector
PriVN	Primary vector in $\mathbf{F_N}$, 3×1 Vector
SecVN	Secondary vector in $\mathbf{F_N}$, 3×1 Vector
qbn	Quaternion from $\mathbf{F_N}$ to $\mathbf{F_B}$, 4×1 Vector
qrn	Quaternion for $\mathbf{F_N}$ to $\mathbf{F_L}$, 4×1 Vector
qbr	Quaternion for $\mathbf{F_R}$ to $\mathbf{F_B}$, 4×1 Vector
$\boldsymbol{\omega bn}$	Angular rate of $\mathbf{F_B}$ w.r.t $\mathbf{F_N}$, 3×1 Vector
$\boldsymbol{\omega rn}$	Angular rate of $\mathbf{F_R}$ w.r.t $\mathbf{F_L}$, 3×1 Vector
$\boldsymbol{\omega err}$	Angular rate error, 3×1 Vector
therr	Quaternion error in angular form , 3×1 Vector
Tcmd	Control torques, 3×1 Vector
Twhlcmd	Reaction wheel command torques, 3×1 Vector
Kp	Proportional control gains, 3×1 Vector
Kr	Derivative control gains, 3×1 Vector

Algorithm 4 Slew Controller

procedure SLEW CONTROLLER
    set **CRN**, **Cdot**, **OldCRN**
    set inertial vector **KN** $= [0, 0, 1]$
    set **R2N**, **range**, **range_unit**, **gsECEF**, **gsECI**
    set **Lat** $= 0.5895$, **Lon** $= -1.473$, **Alt** $= 300$
    set primary vector **PriVR** $= [0, -1, 0]$
    set secondary vector **SecVR** $= [0, 0, 1]$
    Compute **gsECEF** using function $WGS84ToECEF$(**Lat**, **Lon**, **Alt**)
    Compute **gsECI** using **CWN** and **gsECEF**
    Compute **range** as follow:
    for i=1:3 **do**
        range[i] = **PosN**[i] - **gsECI**[i]
    Calculate **range_unit**
    Compute **R2N** by cross product **KN** $\times$ **range_unit**
    Assign primary vector **PriVN** as follow:
    for 1=1:3 **do**
        PriVN[i] = **range_unit**[i]
    Assign secondary vector **SecVN** as follow:
    for 1=1:3 **do**
        SecVN[i] = **R2N**[i]
    Compute **CRN** by TRIAD Method using **PriVR**, **SecVR**,**PriVN**, **SecVN**
    Compute command quaternion **qrn** from **CRN**
    Compute **Cdot** as follow:
    for i=1:3 **do**
        for j=1:3 **do**
            Cdot[i][j] = (**CRN**[i][j] - **OldCRN**[i][j]) / dt
    Compute $\boldsymbol{\omega rn}$ using **Cdot** and **CRN**
    Update **OldCRN** as follow:
    for i=1:3 **do**
        for j=1:3 **do**
            OldCRN[i][j] = **CRN**[i][j]
    Compute quaternion error **qbr** using **qbn** and **qrn**
    Convert **qbr** into angular form **therr**
    Compute angular rate error $\boldsymbol{\omega err}$ as follow:
    for $i = 1 : 3$ **do**
        $\boldsymbol{\omega err}$[i] = $\boldsymbol{\omega bn}$[i] - $\boldsymbol{\omega rn}$[i]
    Compute the command torque as follow:
    for $i = 1 : 3$ **do**
        Tcmd[i] = $-$**Kp**[i] $\times$ **therr**[i] $-$**Kr**[i] $\times$ $\boldsymbol{\omega err}$[i]
    Input command torque to reaction wheels as follow:
    for $i = 1 : 3$ **do**
        Twhlcmd[i] = $-$**Tcmd**[i]

5.5 Axis-Command Controller

Table 11: Axis-Command Controller Variables

Variables	Descriptions
dt	Simulation time step size, scalar.
CRL	Directional cosine matrix from $\mathbf{F_L}$ to $\mathbf{F_R}$, 3×3 Matrix
CLN	Directional cosine matrix from $\mathbf{F_N}$ to $\mathbf{F_L}$, 3×3 Matrix
PriVR	Primary vector in $\mathbf{F_R}$, 3×1 Vector
SecVR	Secondary vector in $\mathbf{F_R}$, 3×1 Vector
PriVN	Primary vector in $\mathbf{F_N}$, 3×1 Vector
SecVN	Secondary vector in $\mathbf{F_N}$, 3×1 Vector
qbn	Quaternion from $\mathbf{F_N}$ to $\mathbf{F_B}$, 4×1 Vector
qrn	Quaternion for $\mathbf{F_N}$ to $\mathbf{F_R}$, 4×1 Vector
qln	Quaternion for $\mathbf{F_N}$ to $\mathbf{F_L}$, 4×1 Vector
qbr	Quaternion for $\mathbf{F_R}$ to $\mathbf{F_B}$, 4×1 Vector
$\boldsymbol{\omega ln}$	Angular rate of $\mathbf{F_L}$ w.r.t $\mathbf{F_N}$, 3×1 Vector
$\boldsymbol{\omega rn}$	Angular rate of $\mathbf{F_R}$ w.r.t $\mathbf{F_L}$, 3×1 Vector
$\boldsymbol{\omega err}$	Angular rate error, 3×1 Vector
therr	Quaternion error in angular form , 3×1 Vector
Tcmd	Control torques, 3×1 Vector
Twhlcmd	Reaction wheel command torques, 3×1 Vector
Kp	Proportional control gains, 3×1 Vector
Kr	Derivative control gains, 3×1 Vector

Algorithm 5 Axis Command Controller

procedure AXIS-COMMAND CONTROLLER
    set **CRL**, **qln**
    set primary vector **PriVR** = $[0, 1, 0]$
    set secondary vector **SecVR** = $[0, 0, 1]$
    set primary vector **PriVN** = $[0, 1, 0]$
    set secondary vector **SecVN** = $[0, 0, -1]$
    Compute **CRL** by TRIAD Method using **PriVR**, **SecVR**,**PriVN**, **SecVN**
    Compute command quaternion **qrl** from **CRL**
    Compute quaternion **qln** from **CLN**
    Compute command quaternion **qrn** using **qln** and **qrl**
    Compute command $\boldsymbol{\omega rn}$ using $\boldsymbol{\omega ln}$ and **qrn**
    Compute error quaternion **qbr** using **qrn** and **qbn**
    Convert **qbr** into angular form **therr**
    Compute angular rate error $\boldsymbol{\omega err}$ as follow:
    for $i = 1 : 3$ **do**
        $\boldsymbol{\omega err}[i]$ = $\boldsymbol{\omega bn}[i]$ - $\boldsymbol{\omega rn}[i]$
    Compute the command torque as follow:
    for $i = 1 : 3$ **do**
        $\mathbf{Tcmd}[i] = -\mathbf{Kp}[i] \times \mathbf{therr}[i] - \mathbf{Kr}[i] \times \boldsymbol{\omega err}[i]$
    Input command torque to reaction wheels as follow:
    for $i = 1 : 3$ **do**
        $\mathbf{Twhlcmd}[i] = -\mathbf{Tcmd}[i]$

6 Conclusion and Future Work

This paper presents the design of the attitude controllers for the Earth remote sensing CubeSat, MicroNimbus. These controllers are developed bases on the pointing requirement of the mission. The design methods and theories are explained in this paper. A stability analysis for these controllers is also included in this paper. The commanded attitudes are proved to be stable using the current controller. These controllers are implemented on a NASA GSFC software 42 for simulations. The simulation results show that the controllers are functional. These controllers serve as a baseline design for the flight version controllers on MicroNimbus and other similar CubeSat mission in the Space System Design Lab.

Future work includes sensors and actuators model implementation on the simulation, desaturation mode implementation for each pointing controllers, controller testing and optimization. The current attitude controller simulation assumes perfect attitude knowledge and perfect control actuation. Sensor noise and biases can be incorporated into the simulation which makes the simulation more realistic. Actuators model will also enhance the fidelity of the simulation. Reaction wheel momentum imbalance, torque and momentum relation, and hardware actuation time are largely influence the control actuation performance. These hardware characteristics should be included in the future simulation model. The most important task is to create a desaturation mechanism for each pointing controller. This mechanism can be done using the three-axes magnetic torquers. Simulation results show that during the nadir pointing mode and the axis-command pointing mode, the angular momenta of reaction wheels change dramatically. If the reaction wheels operate as momentum wheels at a nominal momentum 0.01 Nms, all three wheels will saturate quickly. Thus, the spacecraft losses controllability. Magnetic torquers should be used to unload the momenta throughout any attitude pointing mode. Furthermore, the attitude controllers are not optimized. For example, the F_R used in Sun-Pointing controller and Slew Controller is not unique. Users can optimize the F_R to reduce angular momenta of reaction wheels.

7 Acknowledgments

The authors would also like to thank Dr. Glenn Lightsey, Tanish Himani, Julian Brew, Shez Virani, Andrew Fear and Chris Carter for supporting this work in various ways.

References

[1] F Landis Markley and John L Crassidis. *Fundamentals of spacecraft attitude determination and control*, volume 33. Springer, 2014.

[2] Margaret Hoi Ting Tam et al. An attitude determination and control system for small satellites. *University of Texas Austin*, 2015.

[3] SOLARMEMS Technology. *nanoSSOC-A60 Technical Specification, Interfaces Operation*, 11 2015. Ver. 1.05.

[4] EPSON. *M-G364PDCA Data Sheet*, 12 2016. Rev.20161220.

[5] Honeywell. *1,2 and 3 Axis Magnetic Sensros HMC1051/HMC1052L/HMC1053*, 01 2010. Form 900308.

[6] NovAtel. *Compact, Dual-Frequency GNSS Receiver Delivers Robust RTK Functionalitys*, 07 2014. Version 7.

[7] Sinclair Interplanetary. *Picosatellite Reaction Wheels (RW-0.01)*, 05 2013. Revision 1.3.

[8] Hassan K Khalil. Noninear systems. *Prentice-Hall, New Jersey*, 2(5):5–1, 1996.

[9] Parker Francis. Development of the evolved common hardware bus (techbus). 2016.

[10] Mirko Leomanni. Comparison of control laws for magnetic detumbling.