

Development of an Attitude Determination and Control System (ADCS) for a 1U Cubesat

a project presented to
The Faculty of the Department of Aerospace Engineering
San José State University

in partial fulfillment of the requirements for the degree

Master of Science in Aerospace Engineering

by

Michael Hahn

December 2025

approved by

Dr. Yawo Ezunkpe
Faculty Advisor



**SAN JOSÉ STATE
UNIVERSITY**

© 2025

Michael X. Hahn

ALL RIGHTS RESERVED

Abstract

Development of an Attitude Determination and Control System (ADCS) for a 1U Cubesat

Michael X. Hahn

With the innovations that have occurred in the space industry over the past few decades, the cost to launch has gradually decreased...

With the innovations that have occurred in the space industry over the past few decades, the cost to launch has gradually decreased opening up the possibility of launching and deploying a spacecraft to just about anyone with the knowledge and even a small budget. This begs the question of whether the size of the spacecraft will end up determining the sharp decrease in launch costs or if eventually any spacecraft will be significantly cheaper to launch. Given that CubeSats and their simplicity have opened up the possibility to small startups, students and even hobbyists, the development of an attitude control system for a common 1U CubeSat must become commonplace for individuals. In the case of the size of spacecraft decreasing in size to 0.5U or lower, the implementation of reaction wheels may still be necessary depending on objective leading to smaller reaction wheels being a requirement.

As it stands, the attitude determination and control system will become increasingly important as CubeSats and other nanosatellites are implemented into more mainstream missions and objectives that involve various cameras and optical sensors. The B-dot controller will remain an important part of attitude determination as long as detumbling with low power consumption is necessary. The B-dot controller was deemed sufficient given the average fifteen hour detumbling at an average rate of angular velocity. The reaction wheel controller was capable of orienting the CubeSat in under five seconds given a sudden three axis perturbation. Overall, the ADCS was deemed sufficient in its phase of progress to handle realistic orbital perturbations and detumbling scenarios.

Acknowledgments

I would like to thank Dr. Ezunkpe first for his guidance and serving as my advisor throughout the entirety of this project. He always made himself available on such short notice despite our conflicting busy schedules.

I am grateful to my mentors and colleagues at NASA Ames who have assisted me academically and occupationally as I have strived to become a better engineer.

I would also like to thank my parents, family and friends for supporting me throughout this endeavor.

Lastly, I would like to thank my wife, Raquel, and my daughter, Eden, for their support and collectively serving as the best distraction I could hope for from my studies.

Contents

1	Introduction	1
1.1	Objective	1
1.2	Motivation	1
2	Literature Review	2
2.1	CubeSat Overview	2
2.2	Attitude Determination and Control System Components	3
2.3	Control Algorithms	6
3	System Requirements	11
3.1	Onboard Computer (OBC)	11
3.2	Magnetorquers	13
3.3	Reaction Wheels	13
3.4	Integrated GNC Subsystems	14
4	Legacy Missions	16
4.1	Ionospheric-Thermospheric Scanning Photometer for Ion-Neutral Studies (IT-SPINS)	16
4.2	Miniature X-ray Solar Spectrometer (MinXSS)	17
4.3	ADCS CubeSat Performance Limits	22
5	CubeSat Design and Layout	25
5.1	Structural Design	25
5.2	ADCS Unit Layout	29
6	Equations of Motion	32
6.1	General Derivations	32
6.2	Initial Deployment Detumbling	33
6.3	Reaction Wheel Controller	34
7	Controller Simulations	36
7.1	B Dot Controller	36
7.2	Reaction Wheel Stabilization Controller	40
7.3	Dual Attitude Control	46
8	Future Work	52
9	Conclusion	53
A	MATLAB Code	57
1	B dot Controller	57
2	Reaction Wheel Attitude Controller	62
2.1	One Axis Perturbation Reaction Wheel Controller (RWP015)	62
2.2	One Axis Perturbation Reaction Wheel Controller (RWP050)	65
2.3	LEO Simulation Reaction Wheel Controller	67

List of Figures

2.1	A poly picosatellite orbital deployer (P-POD) system [4]	3
2.2	High Performance Reaction Wheel Commonly Used by Kepler, Dawn and Several Other NASA Missions [8]	4
2.3	iMTQ magnetorquer board	6
2.4	Block diagrams for b-dot controller for (a) varying magnetic field and (b) average magnetic field	10
3.1	CubeADCS Core [23]	12
3.2	Arduino Nano Every [24]	12
3.3	EnduroSat solar panel with magnetorquer [25]	13
3.4	CubeWheel pyramid [23]	14
4.1	Annotated exploded view of IT-SPINS spacecraft and fully integrated IT-SPINS spacecraft[29]	16
4.2	MinXSS Prototype (left), MinXSS-1 (center), MinXSS-2 (right) [31]	17
4.3	Total momentum vs. time and altitude [31]	19
4.4	Reaction wheel speed vs. time and altitude [31]	19
4.5	Torque rod duty cycle vs. time and altitude [31]	20
4.6	Jitter histogram [31]	21
4.7	Star tracker sensor median background vs. time [32]	22
4.8	Absolute pointing error results [33]	24
5.1	Cubesat structure	26
5.2	Top wall bracket (interior)	27
5.3	Mass properties of the CubeSat structure (hardware included)	27
5.4	Pumpkin Space Systems CubeSat Kit Skeletonized Chassis Walls [35]	28
5.5	iADCS-100 CAD Drawing [37]	29
5.6	AAC Clyde Space Starbuck Nano (top view and specifications)	29
5.7	CubeSat subsystem stack	30
5.8	CubeSat as assembled (top solar panel suppressed)	31
7.1	Dipole moment over time	37
7.2	Amperage use over time	38
7.3	Change in magnetic field over time	39
7.4	Change in angular velocity over time	40
7.5	Euler angle change over time (RWP015)	41
7.6	Angular velocity of one axis perturbation over time (RWP015)	42
7.7	Angular momentum of one axis perturbation over time (RWP015)	43
7.8	Angular momentum of one axis perturbation over time (RWP050)	44
7.9	Angular momentum of one axis perturbation over time (RWP050)	45
7.10	Angular momentum of one axis perturbation over time (RWP050)	46
7.11	Attitude response of three axis tumbling (RWP050)	47
7.12	Rotation error of three axis tumbling (RWP050)	48
7.13	Body rates of three axis tumbling (RWP050)	49
7.14	Body rates and rotation error with high derivative gain	50
7.15	Reaction wheel momentum of three axis tumbling (RWP050)	51

List of Tables

2.1	Spacecraft mass classifications [2]	2
3.1	Currently available integrated systems [26]	15
4.1	Pointing accuracy performance summary [31]	18
4.2	Specifications of AX-RW1 reaction wheel system	23
5.1	CubeSat Standard Structure Dimensions	25
5.2	External and Internal Dimensions	28
6.1	Center of mass	32
6.2	Moments of Inertia	32

Symbol	Definition	Units
U	CubeSat Unit	
A	Area	mm ²
B	Local magnetic field	T
ϕ, θ, ψ	Euler angles	
H	Angular momentum	kg · m ² /s
J	Cost function	
M	Disturbance & motor torque	N · m
Q, R	Weighting matrices	
R	Torquer resistance	V
U	Voltage	V
U	Torque	N · m
Greek Symbols		
ϕ, θ, ψ	Euler angles	
ω	Angular velocity	rad/s
Acronyms		
ADCS	Attitude Determination & Control System	
GNC	Guidance, Navigation, & Control	
IMU	Inertial Measurement Unit	
LQR	Linear Quadratic Regulator	
PID	Proportional–Integral–Derivative controller	
P-POD	Poly–Picosatellite Orbital Deployer	
XACT	FleXible Attitude Control Technology	

1 Introduction

1.1 Objective

This project aims to develop an attitude determination and control system for a 1U CubeSat. The components utilized in this control system will include three reaction wheels covering each axis of rotation and two magnetorquers as supplemental control to the reaction wheels. The magnetorquers would also serve as a form of fault tolerance in the case of reaction wheel failure.

1.2 Motivation

Mission and launch costs have drastically declined with the introduction of newer technology and the entry of privatized companies. With these companies and advanced technology, previously stagnant concepts have advanced exponentially [1]. With these drastically reduced costs, come the possibility for many others to take part in launches. Startups dedicated to launching CubeSats and other nanosatellites into space for various reasons have cropped up in the past few decades. Even students and hobbyists now have the opportunity to participate in the launch of their own satellite.

As technology becomes even smaller and more efficient, new possibilities for spacecraft will emerge. Whereas before an entire team of people were needed to work on just one subsystem of a spacecraft for multiple years, a much smaller team can design, manufacture and launch a nanosatellite in under a year depending on objective. With these recent innovations, studying the systems that allow these CubeSats to stay oriented in space is important to understand present capabilities and how to improve upon them.

2 Literature Review

2.1 CubeSat Overview

CubeSats are small satellites that fall under the category of nanosatellites, meaning they are under 10 kg. CubeSats are classified by units or '1U', which means that the entire structure of a 1U CubeSat is (10 x 10 x 10) cm. In terms of weight requirements, CubeSats must be less than 2kg per 1U.

Table 2.1 – Spacecraft mass classifications [2]

Spacecraft Class Definition		Wet Mass, kg
Large Spacecraft		≥1000
Small Spacecraft	Medium Spacecraft	500-1000
	Mini Spacecraft	100-500 *
	Micro Spacecraft	10-100
	Nano Spacecraft	1-10
	Pico Spacecraft	0.01-1
	Femto Spacecraft	0.01-0.01 *

} Cubesats

* organization-dependent upper thresholds vary from 180 to 500 kg
 ** organization-dependent upper thresholds vary from 0.01 to 0.1kg

Additional units can be added to create more volume for the CubeSat depending on the mission objective, payload and the system components that must be included some of these missions can even call for a 12U chassis to contain all of the components.

The idea behind the CubeSat originated in 1999 and occurred due to collaboration between staff at California Polytechnic State University (Cal Poly), San Luis Obispo, and Stanford University. The initial goal of this collaboration was to create a new class of picosatellites allowing that abided by a standard that became known as the “CubeSat standard” which was simply a set of design specifications that the satellite would need to abide by in order to fit the deployment specifications. This standard covered general, mechanical, electrical and testing requirements and allowed for the CubeSat to be deemed deployable if it met said requirements [3].

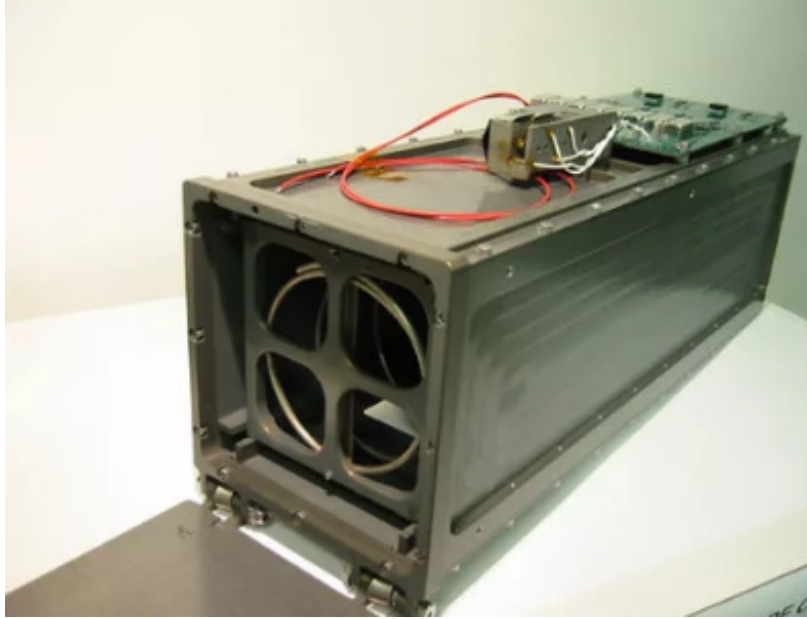


Figure 2.1 – A poly picosatellite orbital deployer (P-POD) system [4]

CubeSats have now been used for various purposes for the past couple decades and serve as an entry for budget friendly entities such as students, hobbyists and startups with little funding. With the costs of space launches steadily decreasing with the advancement of the space industry through many heavily funded startups, launching a relatively cheap satellite has become a reality to anyone with a little bit of money and an objective.

The standard became successful enough that NASA eventually decided to start designing their own CubeSat missions starting in 2008 after CubeSats were proven to be successful [5]. Eventually, a method of deployment was introduced for the CubeSat specifically allowing satellites to ride passenger with other launches driving down costs. This method became known as the Poly-Picosatellite Orbital Deployer (P-POD). The P-POD system is essentially just a rectangular box with a spring that allows for up to three CubeSats to be deployed into orbit.

2.2 Attitude Determination and Control System Components

Recent iterations in CubeSat mission design and development suggest that there are several ways to distribute an ADCS depending on components and mission objectives. Many CubeSats, regardless of units, include something akin to three reaction wheels and some other redundancy that allows for the spacecraft to correct itself in the case of any mishaps [6]. It is increasingly common to see the reaction wheels accompanied by the addition of magnetorquers used to assist in the initial process of detumbling the spacecraft in orbit [7].

2.2.1 Reaction Wheels

Many satellites from CubeSats to even larger satellites such as the International Space System use some form of reaction wheel as a means of reorientation without fuel and make up the most common form of attitude control for spacecraft [8]. The acceleration of mass in one direction creates an equal reaction in the opposite direction. Without any form of drag in the vacuum of space, this allows the affected mass to rotate in the opposite direction [5]. This is the primary

function of reaction wheels in a spacecraft. Given that this rotation can only occur on one axis, a spacecraft is required to have at least three reaction wheels in order to cover all three axes of rotation and allow the spacecraft to rotate in any conceivable direction. Any inclusion of additional reaction wheels is usually for the purpose of redundancy in the case of failure of a primary wheel.

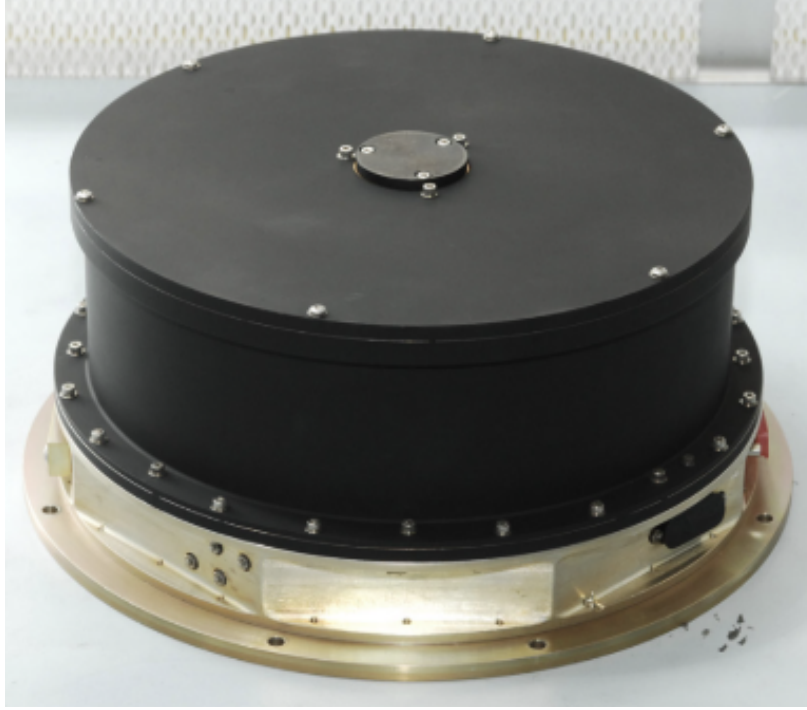


Figure 2.2 – High Performance Reaction Wheel Commonly Used by Kepler, Dawn and Several Other NASA Missions [8]

Because reaction wheels are used to create torque and momentum, there is an entirely separate set of these wheels called momentum wheels that generate no torque by running continuously at high speeds to keep the satellite oriented in a specific direction [8]. Because these wheels are running continuously at high speeds with slight variations for attitude adjustments, they generate little to no torque and instead produce a stiffness on the axis perpendicular to the wheel's rotation. Momentum wheels tend to be used in cases where the orientation of the satellite is important such as photography or some form of surveillance. It's also possible to combine the two in some form to increase the system's stability and maneuverability.

When it comes to mathematically modeling the governing equations of reaction wheels, it's useful to think of the components of angular velocity in a standard XYZ reference frame in terms of $(\omega_1, \omega_2, \omega_3)$. The attitude model is then modeled below using Euler angles (Ψ, Θ, Φ) that denote the satellite's position relative to Earth [9]:

$$\dot{\Psi} = (\omega_2 \sin \phi + \omega_3 \cos \phi) \sec \theta \quad (2.1)$$

$$\dot{\theta} = \omega_2 \cos \phi - \omega_3 \sin \phi \quad (2.2)$$

$$\dot{\Phi} = \omega_1 + \dot{\Psi} \sin \theta = \omega_1 + (\omega_2 \sin \phi + \omega_3 \cos \phi) \tan \theta \quad (2.3)$$

From there, equations for the satellite and actuators are developed from a simplistic relationship

between the two:

$$\vec{M} = \vec{H}'_G \quad (2.4)$$

$$\vec{T} = \vec{H}_W \quad (2.5)$$

HG is the angular momentum of both the satellite and its reaction wheels and HW is the angular momentum of the reaction wheels relative to the satellite. T is the torque of the motor that acts on the reaction wheels and M is the disturbance torque acting on the body of the satellite.

Simply put, by applying angular momentum, the equations for the actuators themselves can be written as:

$$I_a (\dot{\omega}_1 + \dot{\Omega}_1) = u_1 \quad (2.6)$$

$$I_a (\dot{\omega}_2 + \dot{\Omega}_2) = u_2 \quad (2.7)$$

$$I_a (\dot{\omega}_3 + \dot{\Omega}_3) = u_3 \quad (2.8)$$

The equations can be interpreted as the torque applied from the designated motor (1, 2, 3) of the reaction wheel [9].

Given that reaction wheels are the most common part of the ADCS, three reaction wheels will be included along three specified axes to cover rotation along any given direction. The wheels themselves will need to fit into the total 1000 m3 volume of a 1U CubeSat.

2.2.2 Magnetorquers

Magnetorquers are another form of actuator in the form of an electromagnet. They come in square or rod shapes and are a set of copper coil wire. Whereas some CubeSats incorporate the square shape to maximize the surface area enclosed by the coil, for smaller U CubeSats the rod serves as an alternative that uses less mass, volume and power. One of the potential disadvantages of using a rod which is practically a necessity for something like a 1U CubeSat is that even when the coil is not being powered, the magnetic field is still active although to a much lesser degree. However, this means that there is now another force acting on the spacecraft which will require the other components of the ADCS to compensate [5].

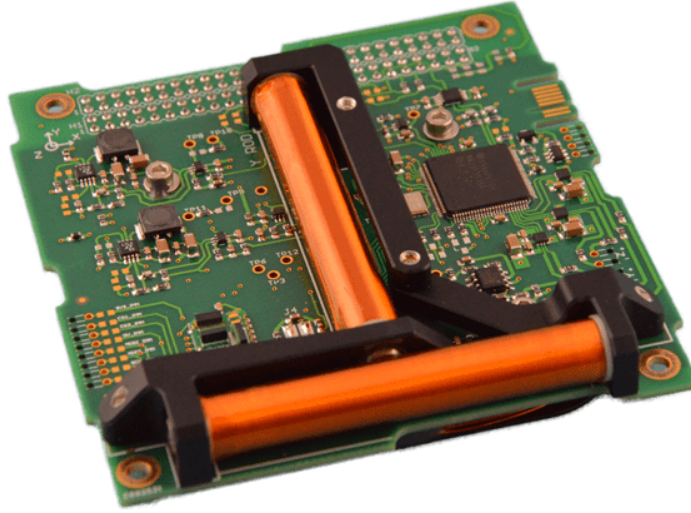


Figure 2.3 – iMTQ magnetorquer board

Magnetorquers can be mathematically modeled with the initial equation (9) which claims that torque has a relation to both the dipole moment $\vec{M}$ and local magnetic field $\vec{B}$ that is represented by

$$\vec{\tau} = \vec{M} \times \vec{B} \quad (2.9)$$

The dipole moment can be compared to a kinematic moment in that the load force is now the separation distance of the positive and negative charges multiplied by the magnitude of the charge and can be represented by [10].

$$\vec{M} = \mu_r A_e \frac{U}{R} \quad (2.10)$$

In this case A_e is the effective area of the torquer and r is the relative magnetic permeability of the magnetic core which is just the amount of magnetization a material produces in response to a magnetic field. R is the resistance of the torquers and U is the voltage.

Finally, the total power consumption of the torquer can be calculated using (11).

$$P = \frac{U^2}{R} = \left(\frac{|\vec{M}|}{A_e} \right)^2 \frac{R}{\mu_r} \quad (2.11)$$

2.3 Control Algorithms

A control system is simply an algorithm that allows for an input that continuously improves upon that input until it reaches a desired output [11]. There are two types of control systems that determine how the system itself will regulate. These are known as closed loop and open loop systems. The more simple system is known as the open loop system which runs through its algorithm based on a fixed input and doesn't consider the effect of the output. An example

of this system would be something like a washer or dryer. The system input is the soil level and temperature of the water used to wash clothing. The washer then runs through its cycle and finishes based on the input time and doesn't check whether the clothing was sufficiently washed.

The other form of control algorithm is the closed loop system which will be the focus of this project. This system takes an input and continuously processes it while providing feedback until a desired output is achieved [11]. In the case of this project, the CubeSat's orientation is the input and the reaction wheels and magnetorquers work to reorient the CubeSat to its desired position.

2.3.1 PID Controllers (Proportional-Integral-Derivative)

The PID controller is one of the most widely used systems in the present day. As technology has become more complex, the need for a better tuned system has arisen. The PID controller takes into account three different elements in order to correct a system for a desired output. Firstly, the P element stands for the proportional error which takes into account how much of a difference there is between the initial input and desired output and allows for it to correct itself. The Integral element is simply the integral of the error up until a certain point. Simply put, the integral element takes into account small errors up until a point and attempts to correct past errors. Finally, the D element takes into account the derivative of the error up until the specified time and attempts to correct for future errors [12]. The elements of the PID controller all work in tandem to correct for past, present and future errors. This is the first and will most likely serve as the simplest form of controller with the longest amount of input whereas the application of the following algorithms will serve as more complicated but should be a more efficient controller when combined.

The PID controller always includes what is called a transfer function denoted by $C(s)$ which takes into account three elements (K_p , T_i , T_D) otherwise known as the proportional gain, integral time and derivative time respectively.

$$C(s) = K_p \left\{ 1 + \frac{1}{T_i s} + T_D D(s) \right\} \quad (2.12)$$

The remaining variable $D(s)$ is known as the approximate derivative and is used as a placeholder for s as it is impossible to verify physically. Instead $D(s)$ is defined as an additional portion of the transfer function and defined by the following equation:

$$D(s) = \frac{1}{1 + \left(\frac{T_D}{\gamma}\right)s} \quad (2.13)$$

is defined as the derivative gain and as increases, the response of the approximate derivative gets closer to the pure derivative. Unfortunately the situation is more complicated as with an increase in derivative gain there comes an increased detection noise. This is why it's important to tune the controller through the use of increasing some variables and decreasing others or, in some cases, completely removing them [12].

2.3.2 Kalman Filter

Kalman Filtering is another algorithm that specifically targets noise and attempts to predict an outcome despite it [13]. Based on this, Kalman filtering can be described as an estimator for a somewhat predictable outcome. An example of this would be with autonomous driving where various types of Kalman filters are applied to test state estimation in nonlinear dynamic systems

[14]. To be more specific, the unscented Kalman filter has been introduced specifically as a method to combat against the uncertainty of a nonlinear system. The algorithm is designed to select a set of sigma points that represent the probability distribution of the state [15].

Starting with a linear dynamic system in state space form (14), the filter sets up a “predictor” using the set of equations (15)

$$x_k = Ax_{k-1} + Bu_k + w_k \quad (2.14)$$

$$y_k = Hx_k + v_k \quad (2.15)$$

In this case, x_k is the state vector at time k , u_k is the control input, y_k is the measurement, A , B , and H are system matrices, w_k and v_k are noise. The system is then iterated continuously changing the output and accounting for noise.

Predictor:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \quad (2.16)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (2.17)$$

The estimate is then updated when the predictor returns a new measurement:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (2.18)$$

$$\hat{x}_k = \hat{x}_k^- + K_k (y_k - H\hat{x}_k^-) \quad (2.19)$$

$$P_k = (I - K_k H) P_k^- \quad (2.20)$$

The way that this algorithm would apply to something such as the ADCS of a CubeSat would be to estimate the state vector. This would be done using measurements taken by the sensors of the system and combining that with a dynamic model of the spacecraft. By doing this, the filter working with the PID controller will further estimate the orientation of the CubeSat creating a smaller range of error so that the CubeSat will correct itself further and more efficiently than it would with just the PID.

2.3.3 LQR (Linear Quadratic Regulator)

Another option in the field of optimal control theory is that of the linear quadratic regulator. This controller is used to design an optimal state feedback controller for a linear dynamic system and is used in place of a PID. In the case of the PID, it is easier to create and implement relative to the LQR; the PID is much more difficult to tune as it requires tuning for each input. To put it simply, the LQR is a more adaptable system but requires more setup. It's a great tool for complex systems where a PID might otherwise struggle [16].

The LQR starts with a linear state space system:

$$\dot{x} = Ax + Bu \quad (2.21)$$

Then a cost function is used to weigh deviations and control effort:

$$J = \int_0^\infty (x^T Q x + u^T R u) dt \quad (2.22)$$

where Q and R are both matrices used for weighting. S , which is a matrix that represents the modified cost function, is found using (23):

$$A^T S + SA - SBR^{-1}B^T S + Q = 0 \quad (2.23)$$

Solving for S determines the future cost which is dependent on the state. Using S , the optimal feedback gain can now be determined:

$$K = R^{-1}B^T S \quad (2.24)$$

Determining K will finally allow the best course of action for control, and applying the feedback continuously allows the system to continue to correct itself to stability.

$$u^* = -Kx \quad (2.25)$$

In the case of this project, an LQR and a Kalman filter would work simultaneously to both develop a control system and a state estimator to assist the controller [17].

2.3.4 B-Dot Algorithm

When a satellite is initially launched into orbit, it requires a way to detumble and reorient itself. A CubeSat tends to do this through the use of a set of magnetorquers during its initial launch from a P-POD. This is done by imitating the viscous friction of the rotated satellite with regard to the vector of the geomagnetic field induction denoted by B [18].

In its simplest form, the B-dot controller is calculated by taking the derivative with respect to time of the magnetometer's measurement relative to the body frame of the satellite [19]. From there, the derivative of the magnetic field is multiplied by the gain thus producing the dipole moment of the magnetic controller which can be written as [20]:

$$\vec{m} = -k_{dot} \dot{\vec{b}} \quad (2.26)$$

Where k is the gain matrix defined by [38]:

$$k_{dot} = \begin{bmatrix} k_{dot,x} & 0 & 0 \\ 0 & k_{dot,y} & 0 \\ 0 & 0 & k_{dot,z} \end{bmatrix} \quad (2.27)$$

Finally, the b-dot controller in block diagram format can be viewed in Figure 3 below.

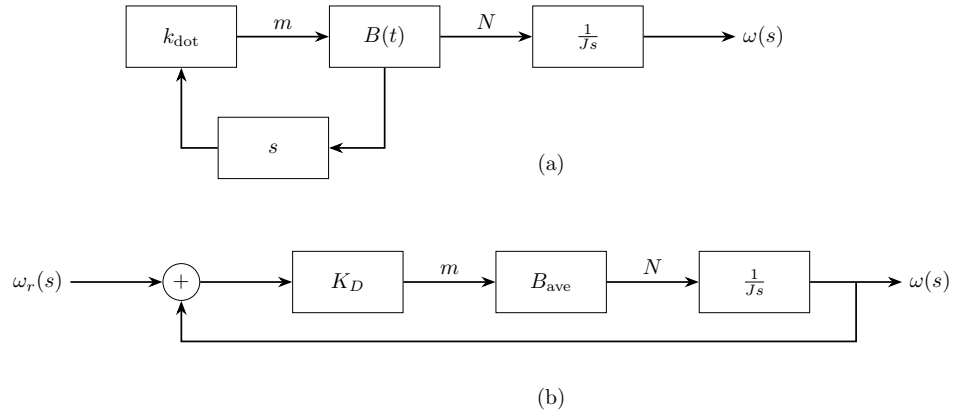


Figure 2.4 – Block diagrams for b-dot controller for (a) varying magnetic field and (b) average magnetic field

For the development of the attitude control system for a CubeSat, a B-dot controller is the most necessary and only controller needed for the magnetorquers. This algorithm will only be needed for the best case scenario of detumbling during its initial launch in order to detumble. The worst case scenario is that the magnetorquers would be needed for total flight time in the case of failure of one or more reaction wheels. Due to the scope of this project, every fraction of volume is taken up by each subsystem so that the system is able to run. Unfortunately, there is little to no room for redundancy which means an additional reaction wheel wasn't added to the configuration.

3 System Requirements

The purpose of this chapter is to outline the best hardware for each subsystem to outline the different layouts that the CubeSat can have depending on mission objectives. The standard requirements for a CubeSat in particular are that the 1U may not exceed the 10 x 10 x 10 cm volume and cannot exceed the 2 kg mass requirements. In the case of this project, the only requirement is that the spacecraft is able to correct itself using reaction wheels and magnetorquers. There are many startups or space related companies that have begun to release their own subsystem components such as ADCS bundles that include full spin stabilization through the use of reaction wheels, magnetorquers and various sensors. Components such as reaction wheels, magnetorquers and onboard computers will be the main topic of research for this chapter as many other components can be sourced cheaply and found practically anywhere. Magnetometers and other sensors also tend to be included in many of the onboard computers listed below.

3.1 Onboard Computer (OBC)

3.1.1 NanoMind A3200

Developed by GOMspace, this is one of the best options for a space-ready satellite as it was developed purely for smaller satellites such as a CubeSat. Relative to other options outlined later, this onboard computer unit is optimized for lower power consumption without taking away from its efficiency and performance. One of the first requirements that this component fits is mass and sizing requirements. At 96 mm × 90 mm × 12 mm, the NanoMind should be able to fit into one layer of the stack and weighs around 50 kg which only accounts for a small fraction of the 2 kg total allowed. As mentioned in terms of performance, the unit requires less than 0.5 W which is more than ideal for a 1U CubeSat as power requirements are also equally important since the satellite only has so much outer surface area for solar panels [21]. Most other competitors require 0.5 W or more to perform at the same rate of 400 MHz making this specific OBC one of the best options [22].

3.1.2 CubeComputer

Another option would be the CubeComputer created by CubeSpace, a company entirely centered around manufacturing system components for CubeSats [21]. This is by far the most convenient option for this project as it would require purchasing the CubeADCS package. This package includes the CubeComputer, ADCS software and reaction wheel pyramid already aligned for 3-axis stabilization [22].

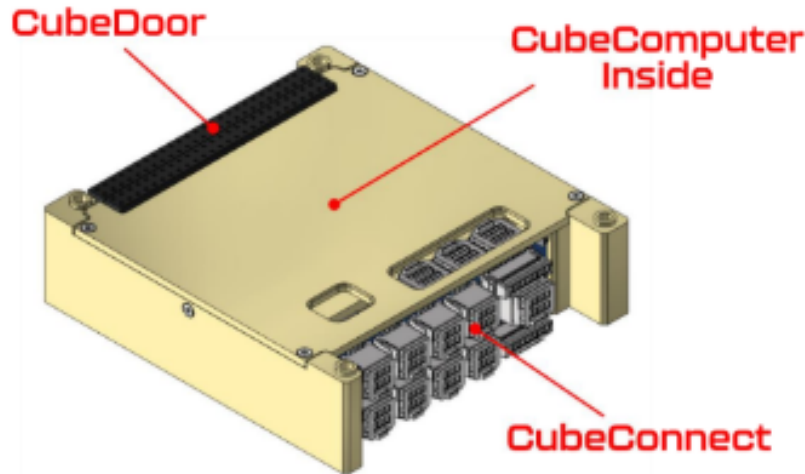


Figure 3.1 – CubeADCS Core [23]

Given that the size of the package is around 90 x 90 x 45 cm, this unit will leave about half of the stack for the batteries and magnetorquers along with any wires or cables needed for remaining miscellaneous hardware. The unit requires 2.5 W which is around half of the Nanomind but performs at around 80 MHz which is all that is required for the ADCS package to run. Along with radiation protection, this unit comes space ready and only requires sufficient space to perform efficiently.

3.1.3 Arduino Nano

When it comes to testing, radiation hardened components can be taken out of the equation which also allows for lower power usage and equally lower price. The Nano operates at the standard 5V that most lower power components tend to operate at and uses the Arduino IDE which is fairly user friendly and easy to pick up. The mass and sizing requirements are more than sufficient with a mass of 7 g and area of 18 x 45 mm respectively [24].



Figure 3.2 – Arduino Nano Every [24]

While a more processing intensive Arduino seems like a better option, the Nano Every model working with a ATmegaS128 microcontroller board will allow for more leftover volume in the space of the 1U but also shouldn't be necessary for a relatively straightforward control system. Likewise,

while a Raspberry Pi might seem like a great option as well with its computing power, the power requirements are unable to support such a unit. The surface area of the 1U can only hold so many solar panels and the power would be best allotted to the two aforementioned microcontrollers.

3.2 Magnetorquers

3.2.1 EnduroSat Magnetorquer Board

EnduroSat is a company that sells solar panels for CubeSats and offers a magnetorquer rod to be integrated into the panel itself. Given that panels are mounted to the exterior of the structure, this could serve as a sufficient option to free up space for the interior. With magnetorquers included, each solar panel comes in at 48 g meaning that the panels plus magnetorquers would only weigh around 0.3 kg total which still leaves plenty of space for the structure and payload [25].

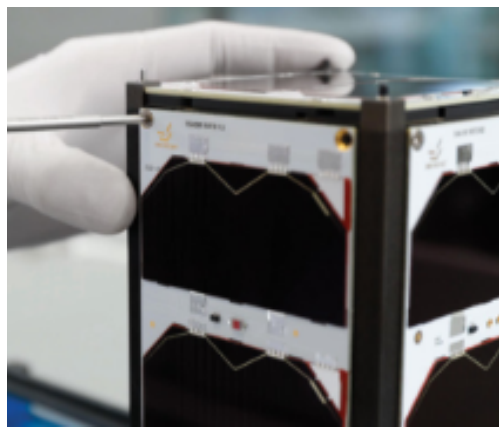


Figure 3.3 – EnduroSat solar panel with magnetorquer [25]

Another thing to note is that EnduroSat manufactures a separate XY panel. These are meant to cover the four sides that make up the x and y directions of the cube while the z axis is an entirely different option. It is here in the Z panel that the magnetorquers are included [25]. This allows the conditions of the project to be met with the minimum number of two magnetorquers are now included leaving the rest of the interior volume for the three reaction wheels, onboard computer and the remaining components.

3.3 Reaction Wheels

3.3.1 CubeWheel

Manufactured by the aforementioned CubeSpace, the CubeWheel is a reaction wheel that weighs 60 g and has a volume of 28 x 26 x 28 mm. Despite its size, the wheel can produce 0.23 mNm of saturation torque which is the maximum amount of torque it can produce at the wheel's maximum velocity. This would allow for the wheels to correct a 2k satellite's orientation within seconds thus serving as an adequate option with the inclusion of three total [23].



Figure 3.4 – CubeWheel pyramid [23]

CubeSpace also includes the option to have the reaction wheels mounted in three axis stabilized orientation. This can be done through the use of three or four wheels for the sake of redundancy. With the introduction of the pyramid as an option, the reaction wheels themselves would take up about 0.25 kg, leaving a large amount of mass for the remaining systems.

3.4 Integrated GNC Subsystems

As mentioned previously, with the costs of space flight drastically decreasing and the technology for modular spacecraft such as CubeSats becoming more popular, a niche market of integrated GNC units have been introduced as options.

According to NASA’s State of the Art Small Spacecraft Technology Report from 2023, the following options are available from various companies to purchase customizable Integrated GNC Systems. Many of these options are adjusted to the mass and sizing requirements of the objective. The CubeADCS unit covered previously is listed in Table 2 and was previously covered. Another viable option for a 1U CubeSat on this list would be AAC Clyde Space’s iADCS200 [26]. This is purely based on sizing requirements where the remaining units all exceed 100 mm in at least one dimension.

One of the ADCS integrated subsystem units commonly used with many smaller CubeSats is the fleXible Attitude Control Technology or “XACT” for short. The system includes a star tracker, IMU and magnetometers for attitude determination measurements and both magnetorquers and reaction wheels along with a processor. Each XACT model has an area of 10 cm x 10 cm and is made to fit into a stack along with an OBC and any other components. According to the data sheet for Blue Canyon’s ADCS units, the XACT-15 boasts a slew rate of up to 10 deg/sec which is sufficient for a 3U CubeSat according to their standards which would be in the range of viability for a 1U CubeSat as this particular model has a volume of 10 cm x 10 cm x 5 cm. It’s worth noting that the unit is advertised as having a length and width of 10 cm but in reality is actually 9.5 cm to account for the structure that the unit must fit into [27].

Table 3.1 – Currently available integrated systems [26]

Manufacturer	Model	Mass (kg)	Actuators	Sensors	Processor	Pointing Accuracy
Arcus ADC	–	0.715	3 reaction wheels, 3 magnetic torquers	1 star tracker, 3 gyros, 6 photodiodes, 3 magnetometers	Yes	0.1°
Berlin Space Technologies	IADCS-100	0.4	3 reaction wheels, 3 magnetic torquers	1 star tracker, 3 gyros, 1 magnetometer, 1 accelerometer	Yes	<< 1°
AAC Clyde Space	iADCS-200	0.47	3 reaction wheels, 3 magnetic torquers	1 star tracker, 1 IMU, optional high precision magnetometer and sun sensors	Yes	< 1°
AAC Clyde Space	iADCS-400	1.7	3 reaction wheels, 3 magnetorquers	1 star tracker, 1 IMU, optional high precision magnetometer and sun sensors	Yes	< 1°
Blue Canyon Technologies	XACT-15	0.885	3 reaction wheels, 3 magnetorquers	1 star tracker, 3-axis magnetometer	Yes	0.003/0.007°
Blue Canyon Technologies	XACT-50	1.23	3 reaction wheels, 3 magnetorquers	1 star tracker, 3-axis magnetometer	Yes	0.003/0.007°
Blue Canyon Technologies	XACT-100	1.813	3 reaction wheels, 3 magnetorquers	1 star tracker, 3-axis magnetometer	Yes	0.003/0.007°
Blue Canyon Technologies	Flexcore	config. dep.	3–4 reaction wheels, 3 magnetorquers	2 star trackers, 3-axis magnetometer	Yes	0.002°
CubeSpace Satellite Systems	CubeADCS 3-Axis Small	0.55	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer	Yes	< 1°
CubeSpace Satellite Systems	CubeADCS 3-Axis Small w/ ST	0.61	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer, 1 star tracker	Yes	< 0.1°
CubeSpace Satellite Systems	CubeADCS 3-Axis Medium	0.79	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer	Yes	< 1°
CubeSpace Satellite Systems	CubeADCS 3-Axis Medium w/ ST	0.84	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer, 1 star tracker	Yes	< 0.1°
CubeSpace Satellite Systems	CubeADCS 3-Axis Large	1.1	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer	Yes	< 1°
CubeSpace Satellite Systems	CubeADCS 3-Axis Large w/ ST	1.15	3 reaction wheels, 3 magnetorquers	10 coarse sun sensors, 2 fine sun/earth sensors, 1 magnetometer, 1 star tracker	Yes	< 0.1°
CubeSpace Satellite Systems	CubeADCS Momentum	Y- 0.3	1 momentum wheel, 3 magnetic torquers	10 coarse sun sensors, 1 magnetometer	Yes	< 5°

4 Legacy Missions

Many of the noteworthy 1U CubeSats include either no ADCS or just a system of magnetorquers that stabilize the satellite [28]. Given that many of these missions include a payload of any size, the inclusion of an ADCS takes up about half or more of a 1U leaving very little room for anything else. Instead, many of the legacy missions will include nanosatellites of a 2U or even 3U size to cover the entirety of an ADCS that also includes reaction wheels. Since there is no payload besides the necessary components to run an ADCS that includes magnetorquers and reaction wheels, the satellite should run sufficiently by itself. As technology continues to become more efficient with less mass required, 1U CubeSats have the potential to become more popular with additional payloads.

4.1 Ionospheric-Thermospheric Scanning Photometer for Ion-Neutral Studies (IT-SPINS)

The IT-SPINS satellite was a mission based around a 3U CubeSat whose purpose was to utilize a CubeSat Tiny Ionospheric Photometer (CTIP) to detect radiation emissions in the ionosphere.

Designed by Montana State University's Space Science and Engineering Lab (SSEL), IT-SPINS was designed all the way back in 2013 and wasn't finished until 2021. IT-SPINS featured a photometer stack that included sun sensors, a shade, and the imager instrument itself. Further down the stack sat the ADCS unit [29]. A custom made unit known as the MAI-400 included all necessary sensors, an OBC along with three electromagnets (magnetorquers) and three reaction wheels.

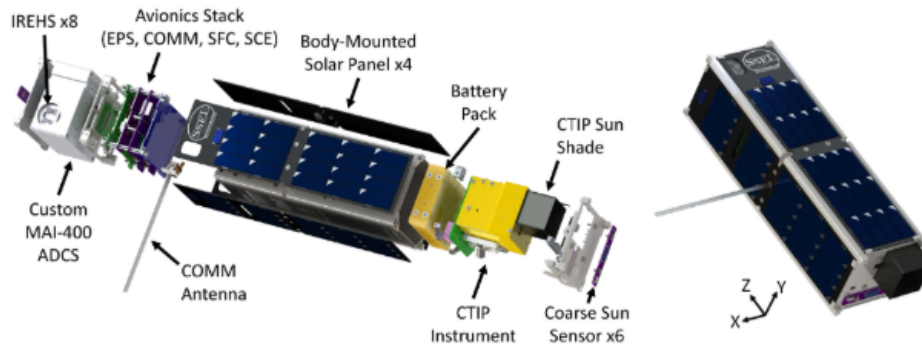


Figure 4.1 – Annotated exploded view of IT-SPINS spacecraft and fully integrated IT-SPINS spacecraft[29]

“Deployed in June 2021, IT-SPINS experienced intermittent communication with the SSEL ground station. The spacecraft was able to beacon properly, allowing SSEL researchers to monitor its health and general status. However, commanding IT-SPINS proved to be difficult. Of the many commands sent out by the SSEL ground station, IT-SPINS only showed a responsive action to a very small percentage. As a result, the spacecraft was never able to orient itself correctly and any science operations were put on hold indefinitely [29].”

While IT-SPINS itself was never able to effectively communicate with its ground station for a period of time, the ADCS unit included was completely functional save for some bugs that Adcole Maryland Aerospace has since removed. The satellite was found instead to have lower bandwidth

than initially thought which rendered it “deaf” to commands from ground station [29].

Further elaborating on IT-SPINS despite its shortcomings, the team developed and expanded upon legacy code from the Focused Investigations of Relativistic Electron Burst Intensity, Range, and Dynamics or FIREBIRD mission for short. This was the legacy mission that IT-SPINS was designed around which was also a CubeSat whose objective was to take images for research purposes. In the case of FIREBIRD, it was aimed at studying electron bursts in the Van Allen radiation belts. Both of these missions incorporated precise ADCS systems as the orientation of the CubeSat must remain constant in order to properly capture images.

One thing to note about the ADCS of the IT-SPINS is that when it came to the custom made MAI 400, one magnetometer was placed externally to the unit to prevent interference from “magnetically noisy” reaction wheels and magnetorquers [30]. Depending on the number of magnetometers included in the 1U for this particular project, it may be useful to place it a specified distance from the other ADCS components.

4.2 Miniature X-ray Solar Spectrometer (MinXSS)

The MinXSS was a series of satellites built by students at the University of Colorado Boulder. MinXSS-1 was a 3U CubeSat that was intended to take visual data on the solar x-ray spectrum.

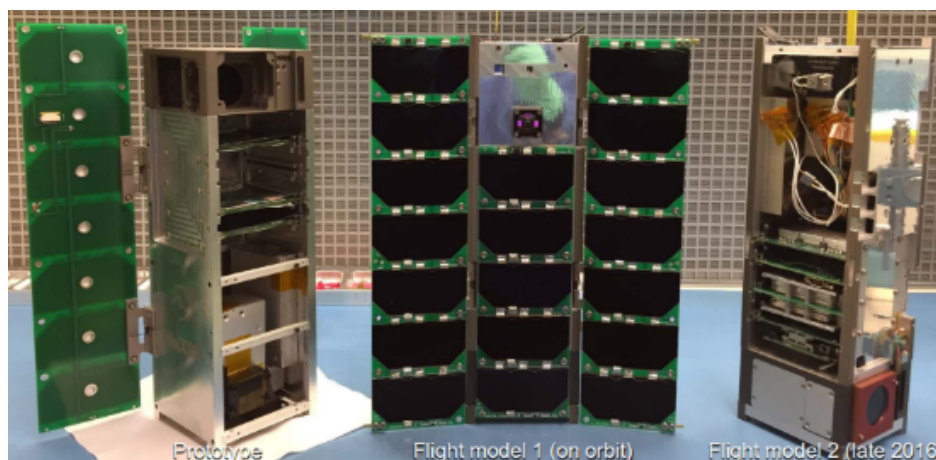


Figure 4.2 – MinXSS Prototype (left), MinXSS-1 (center), MinXSS-2 (right) [31]

Usually with imaging comes the need for precise stabilization and methods to lead to the reduction of jitter. This makes the MinXSS-1 mission a viable legacy mission as it included multiple sensors such as an IMU for the measurement of angular rates, sun sensors used to specify the location of the sun which was especially important for this mission, and a 3-axis magnetometer. For the actuator portion of the ADCS, MinXSS was equipped with 3 magnetorquer rods and a 3-axis reaction wheel system. It was actually equipped with an XACT unit covered in a previous chapter of this report. One of the great takeaways from this project was that it was heavily documented and contains data pertaining to all types of perturbations a satellite can be exposed to and how the team minimized it.

4.2.1 Pointing Accuracy

Pointing accuracy is simply the precision with which something like an imager can be directed to a specific target. The pointing accuracy for the XACT model used in the MinXSS states that

along the X and Z axes, accuracy of 0.009 degrees can be expected whereas the Z axis is 0.021 degrees. The Z axis is the bore sight of the imager; the X and Z axes are across the sight of the image [31]. This means that the Z axis moves along the image while the other two create a plane that the imager scope is normal to which is why the accuracy has to be more precise along those two axes.

Table 4.1 – Pointing accuracy performance summary [31]

Parameter	Requirement (3σ)	Specification (3σ)	Measured Performance (3σ)
Accuracy	X: 2.0 Y: 2.0 Z: 2.0	0.009° 0.021° 0.009°	0.0042° 0.0117° 0.0060°
Jitter	X: 0.3° (10 s ⁻¹) Y: 0.3° (10 s ⁻¹) Z: 0.3° (10 s ⁻¹)	---	0.0183° (10 s ⁻¹) 0.0073° (10 s ⁻¹) 0.0105° (10 s ⁻¹)
De-tumble	–	–	<145 s

Given that the MinXSS objective was to be within just 1 degree accuracy, the specifications given by Blue Canyon were multiple orders of magnitude above what was expected. Upon testing, it was found that the accuracy was even higher than anticipated pointing to the XACT ADCS unit as a successful option.

4.2.2 Momentum Dumping

Momentum dumping is the act of reducing or decaying the amount of angular momentum from reaction wheels to prevent saturation. Perturbations such as gravity, atmospheric drag or solar radiation can cause reaction wheels to spin very quickly to the point of reaching their maximum speed. If saturation occurs at this speed, it could cause loss of control of the spacecraft. This is one of the reasons that ADCS units tend to include magnetorquers as it is important that both reaction wheels and torquers work in tandem to fully stabilize the spacecraft.

During operation, the XACT takes in data at a rate of every five seconds. During that time, measurements such as the total momentum and altitude were logged. They were then plotted with respect to time to understand whether the satellite was capable of momentum dumping during its mission.

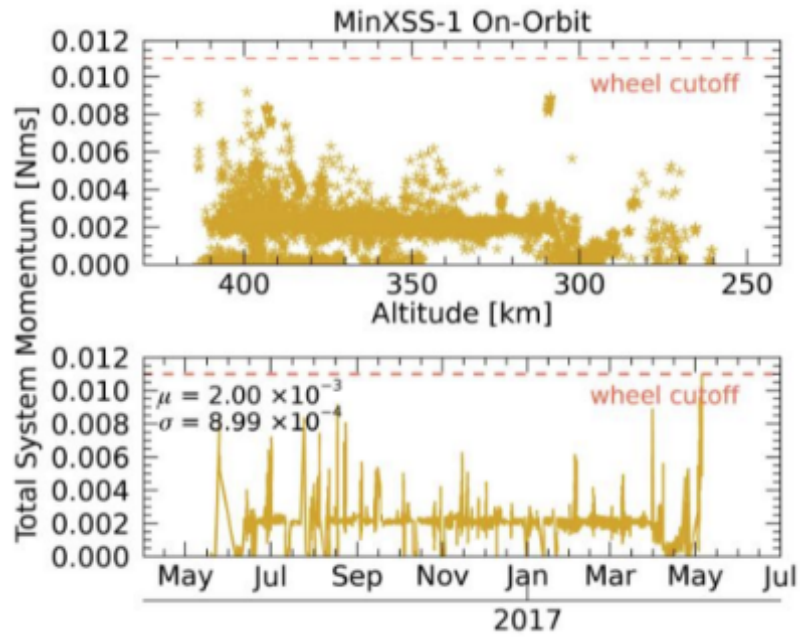


Figure 4.3 – Total momentum vs. time and altitude [31]

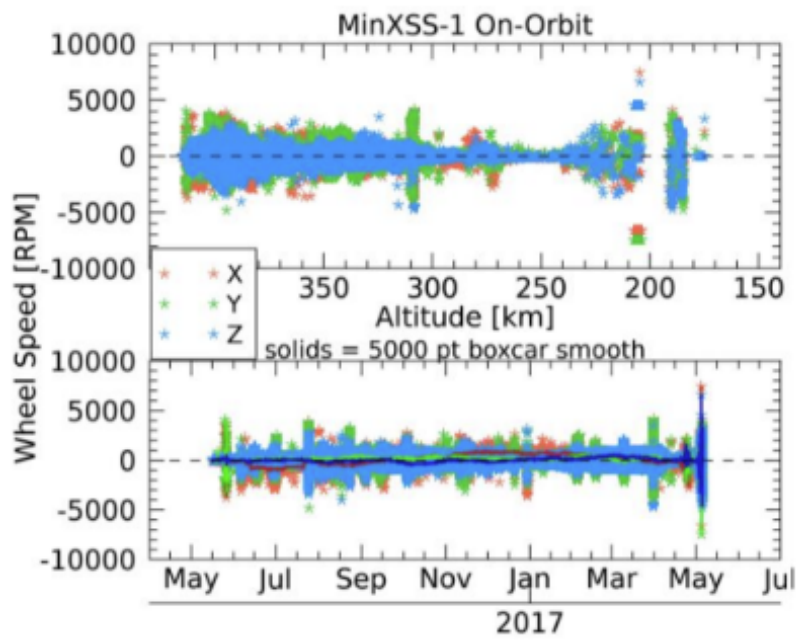


Figure 4.4 – Reaction wheel speed vs. time and altitude [31]

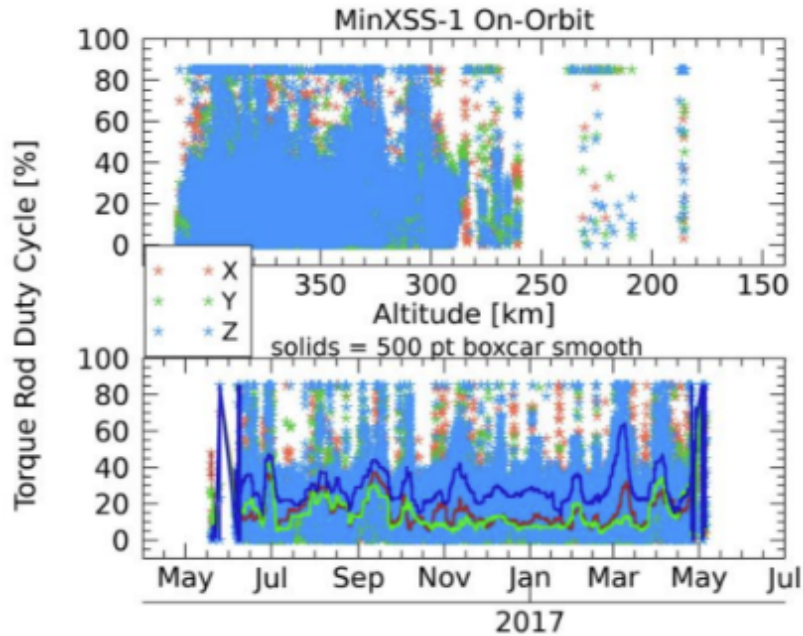


Figure 4.5 – Torque rod duty cycle vs. time and altitude [31]

During this time and altitude, it is evident that the ADCS was able to effectively offset the total system momentum up to the point of around 250 km. It was also reported that around the 185 km mark, atmospheric drag was too high for the system to counteract the additional torque [31]. This means that when selecting how low in altitude a satellite goes in LEO, it's important that atmospheric drag is taken into account to prevent the system from saturation.

4.2.3 Jitter

For this portion of the perturbation, jitter can be thought of as the tiny deviations in pointing accuracy that occur continuously. This is usually caused by the actuator components themselves simply running as they're supposed to. The team deemed that anything over 0.3 degrees over a measured 10 second cycle would cause too many issues with image focusing. When measured, the team reported that the maximum amount of jitter measured across all axes was more than acceptable [31].

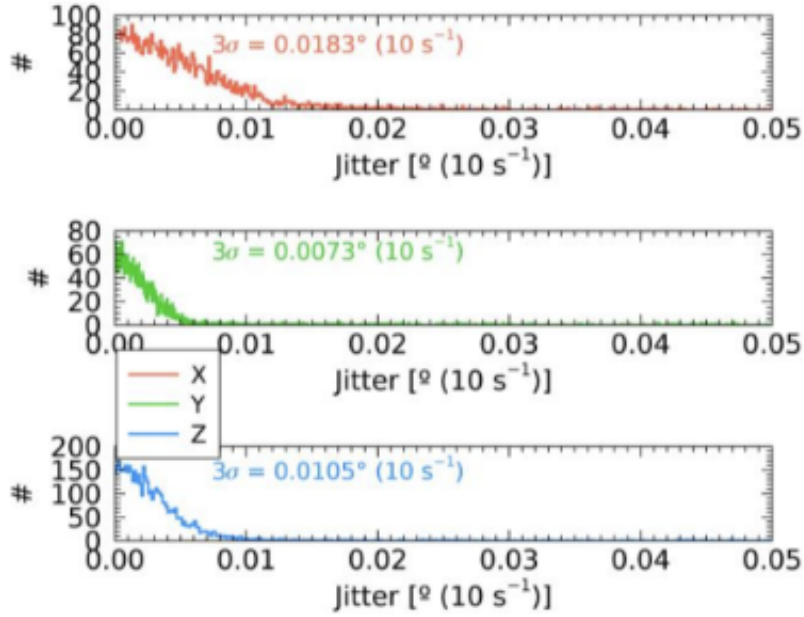


Figure 4.6 – Jitter histogram [31]

Many companies such as Blue Canyon that produce ADCS units are continuously testing for pointing accuracy and how precise the system can be in the face of natural perturbations caused by their own actuator components. Jitter is one of the more important perturbations to account for as it is always present whenever the system is running. It's important to weigh the cost of rotations per minute to vibrations caused to the system and how it affects accuracy. Unfortunately, this is why it's important for subsystem teams to share data with each other as doing so could point out multiple perturbations that other teams haven't considered. It's also important to remember that an ADCS unit that works for one satellite, may not work for another. Of course, working with manufacturers such as Blue Canyon and having access to specifications and previous mission data can prevent issues in the future when considering options [32].

4.2.4 Sensor Degradation

Sensor degradation is by no means a physical perturbation that affects the ADCS of a satellite but is incredibly important to the lifespan of a mission. It plays into the idea that radiation hardening is an important factor when selecting components or providing some type of protection.

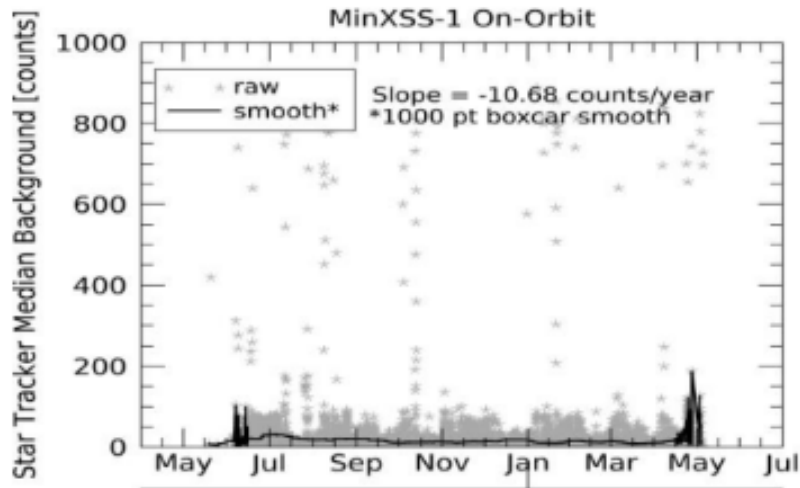


Figure 4.7 – Star tracker sensor median background vs. time [32]

In this case, it was observed by the team that both the sun sensor and star trackers had degraded over time leading to decreased performance. Luckily, the team reports that the degradation didn't lead to any meaningful decreases as the sensors still performed within the margin of error. The team speculates that this was due to exposure to light and radiation and while degradation did occur, the pointing accuracy was still about 0.0042° to 0.0117° , well above the 0.3 degree limit.

4.3 ADCS CubeSat Performance Limits

When it comes to expectations on how well a nanosatellite can perform based on budget, it's important to consider the capabilities of CubeSats in general at the present moment. Many CubeSats are designed by students or hobbyists with little to no budget. Despite the fact that launch costs are decreasing drastically over the past decade or so, it's still important to consider that suppliers are charging heavy prices for components that are "space ready" meaning they've undergone a multitude of tests such as radiation protection and vibrational testing. Reaction wheels also require incredible precision as many of these wheels will be working nonstop for a prolonged amount of time. It's important that they be nearly identical in mass to a small margin of error so as to not offset the system's capability to orient the satellite.

In a report titled 'Where is the Limit: The Analysis of CubeSat ADCS Performance', students at a university in the Netherlands study the performance of ADCS components in the short term, looking at how well pointing accuracy is upheld within a margin of error. They also observe a longer scenario and the components' reliability after about a year. Published in 2016, they cite that satellites such as the CanX-2 are able to achieve pointing accuracies of around 1 degree in the short term which has decreased over time with many integrated ADCS units suppliers boasting overall rates under 0.3 degrees [33].

Going more in depth on the short term performance, the team observed legacy missions of past CubeSats whose missions were anywhere from six months to a year. While the team also observed long term performance, it's important that a 1U ADCS is precise enough to last the entirety of a year long mission with little to no interference before considering more long term goals. For the short term performance model, the team put together a set of guidelines to determine how the mission would be successful. The simulation model guidelines are as follows:

- 1) The satellite is a 3U CubeSat with body-mounted solar panels only.
- 2) The distance between the Centre of Mass (CoM) and the Centre of Geometry (CoG) is within 2 cm based on CubeSat standard; this is to preserve power as the distance between these two constitute just how much more angular momentum is required for movement.
- 3) Only nadir pointing mode using reaction wheels, star tracker, gyros and GPS receiver is considered; this is the simplest mode of pointing as the object of interest in tracking is also the object the spacecraft is orbiting.
- 4) The reference orbit is 600 km Sun Synchronous Orbit (SSO) with 10:30 LTAN; the purpose of this is to maintain consistent lighting conditions for the spacecraft as the satellite is powered with mounted solar panels.

After choosing simulation guidelines, the team chose a specific model of reaction wheel to use in their simulation. They went with the AX-RW1 reaction wheel system. The specifications of the system are listed in Figure 17.

Table 4.2 – Specifications of AX-RW1 reaction wheel system

Item	Specification
Size	88×82×40 mm ³
Mass	413 grams
Momentum storage	5.67 mNms
Max torque	0.1 mNm “ 9000 RPM
Voltage	5V and 3.3V
Power consumption	1.5 W (peak)
Interface	I ² C

With the selection of the reaction wheel system, there were a set amount of modifications or improvements the team felt necessary to include in order to create a more efficient ADCS system. The modifications include:

- 1) Significantly increased torque and momentum capability by optimising the flywheel design.
- 2) Dramatically reduced imbalance due to the new manufacturing and integration process of the flywheel.
- 3) A change of supporting structure from non-rigid to fully rigid.
- 4) Improved wheel controller with much less control error.
- 5) A sealed wheel envelope to reduce the leakage of lubricant.

One important thing to note regarding this simulation was that the team decided not to rely on specification sheets for components but rather real mission data so as not to skew performance towards a best case scenario but rather a realistic set of results. In addition to modeling for the reaction wheels, modeling was done for components such as gyroscopes, star trackers, etc. Noise and disturbances were also taken into account in the form of the SGP4 model, a simplified mathematical model used to measure state vectors relative to an earth coordinate system when the spacecraft is subject to normal perturbations.

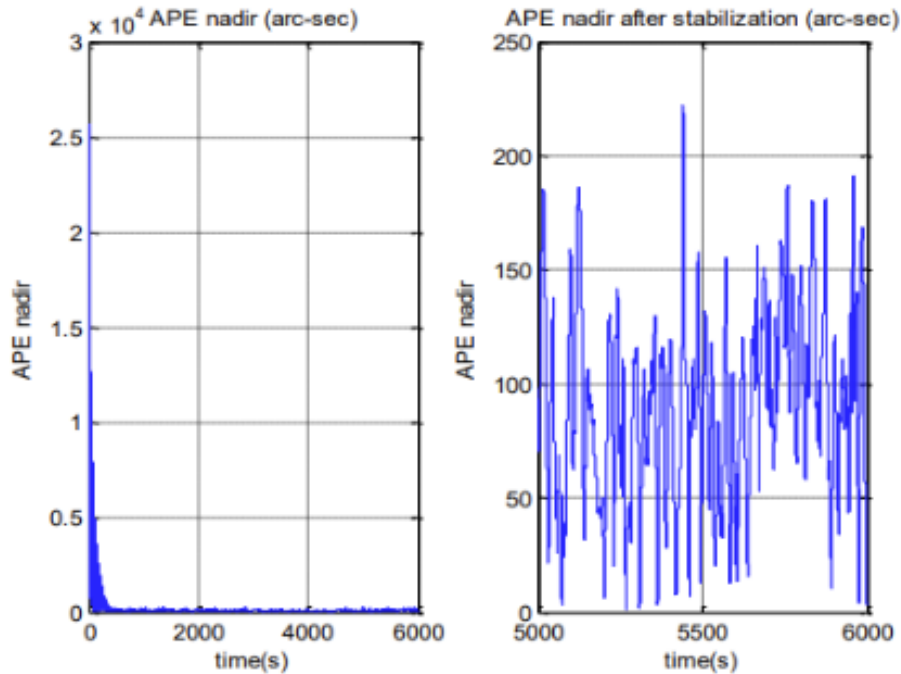


Figure 4.8 – Absolute pointing error results [33]

According to the team, “The attitude was determined using the outputs of the star tracker and gyro and seven states EKF for attitude determination. The attitude control was realised with system linearization and LQR. The simulation is implemented with an initial attitude of [5 5 5] deg and initial angular rate of [0.1 0.1 0.1] deg/s.” [33]. The resultant absolute pointing error was found to be 74.46 arcseconds. This result comes out to two hundredths of a degree which is considered sufficient for high precision pointing thus proving that in the short term, reaction wheel system technology was precise enough to execute a mission.

Recently, with the advent of integrated ADCS units, the inclusion of magnetorquers alongside reaction wheels tends to compensate for much of the noise produced by the wheels themselves. Looking at many of the CubeSats included in Where is the Limit, many include one or the other rather than both. The main issue that arises with these various builds is that the purpose of the CubeSat must be to image in some fashion in order to need high precision pointing otherwise magnetorquers would be sufficient enough to correct the orientation in a timely manner. Given that there are far more options for ADCS systems, the technology has been optimized to a higher degree allowing for more precise pointing. The question is whether the errors that arise will be able to support a more long term mission and whether all components will be able to utilize a smaller volume and the power generated through solar panels will be high enough given a much smaller surface area.

5 CubeSat Design and Layout

5.1 Structural Design

The 1U CubeSat was designed to meet both weight and dimension requirements while weighing the cost. Previously covered in section 1, the weight and dimension requirements can be viewed in the table below. Aluminum 6061 was selected as the structural material just due to weight and

Table 5.1 – CubeSat Standard Structure Dimensions

Type	Dimension (mm)	Average Structure Weight (kg)
1U	$100 \times 100 \times 113.5$	0.118
1.5U	$100 \times 100 \times 170.2$	0.142–0.25
2U	$100 \times 100 \times 227$	0.220
3U	$100 \times 100 \times 340.5$	0.352
6U	$100 \times 226.3 \times 366$	0.916–1.94
12U	$226.3 \times 226.3 \times 366$	1.84

manufacturability traits. Aluminum is typically used in many aerospace applications due to how light it is compared to many other materials. Al-7075 would have been a good option as well if cost and manufacturability weren't two of the main constraints for design. Aluminum 7075 has a much higher strength to weight ratio at 196 kN-m/kg compared to Aluminum 6061 at 115 kN-m/kg which is almost half that of Al 7075 [34]. For a more complex spacecraft such as a shuttle, it would make sense to manufacture some parts in 6061 and 7075 depending on the load or wear experienced by that specific portion or component.

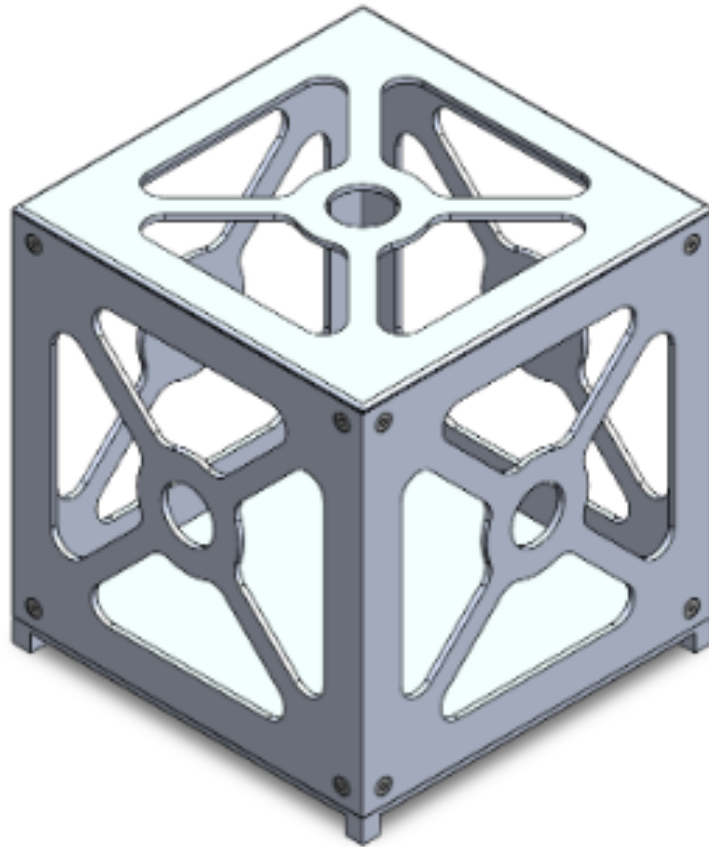


Figure 5.1 – Cubesat structure

The structure pictured in Figure 21 is a simple design of 1/16" Al-6061 sheet metal with a “skeletal” design for the walls to account for weight requirements. With a simplistic design using aluminum sheet metal, the design also features a top and bottom wall which include brackets that allow the walls to be screwed into place using threaded holes.

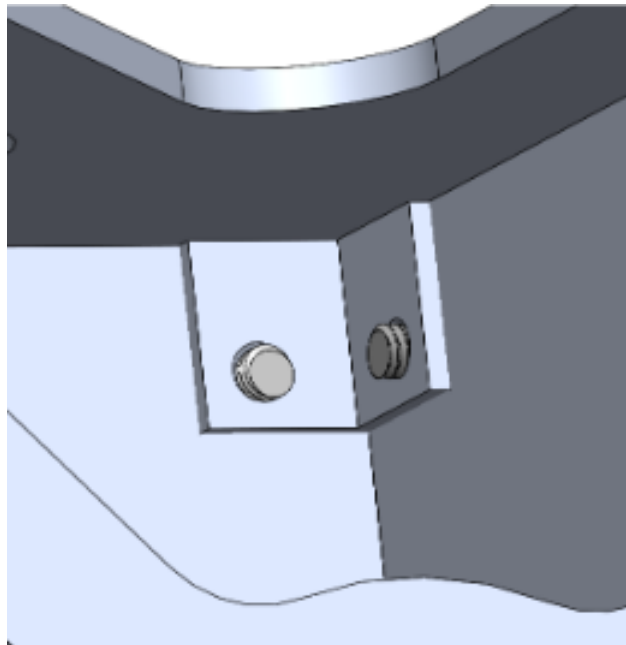


Figure 5.2 – Top wall bracket (interior)

At the top of the structure, a top plate features four countersunk holes for the M2 sized threaded holes at the top of the beams. Before the final step of tightening the screws, the threads should be taped to prevent galvanic corrosion between stainless steel and aluminum.

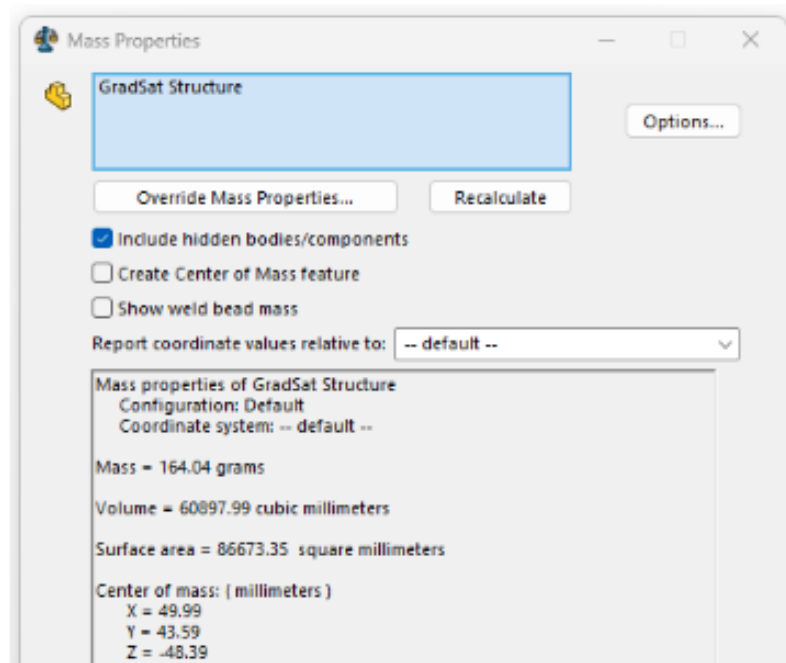


Figure 5.3 – Mass properties of the CubeSat structure (hardware included)

The structure weighs about 160 grams with hardware included which exceeds the average weight featured in Figure 20 by a little over 40 grams. This is due to the weight of the bottom

plate and the pattern of the walls. The bottom wall includes a solid 1/16” thick wall and a set of feet for the CubeSat to stand much like other structures. The top and bottom wall brackets were included into the design for the sake of simplicity when it comes to manufacturing as there are some chassis that are bent into shape from larger pieces of sheet metal such as the chassis offered by Pumpkin Space Systems who offer various components that make up the different subsystems of the spacecraft.

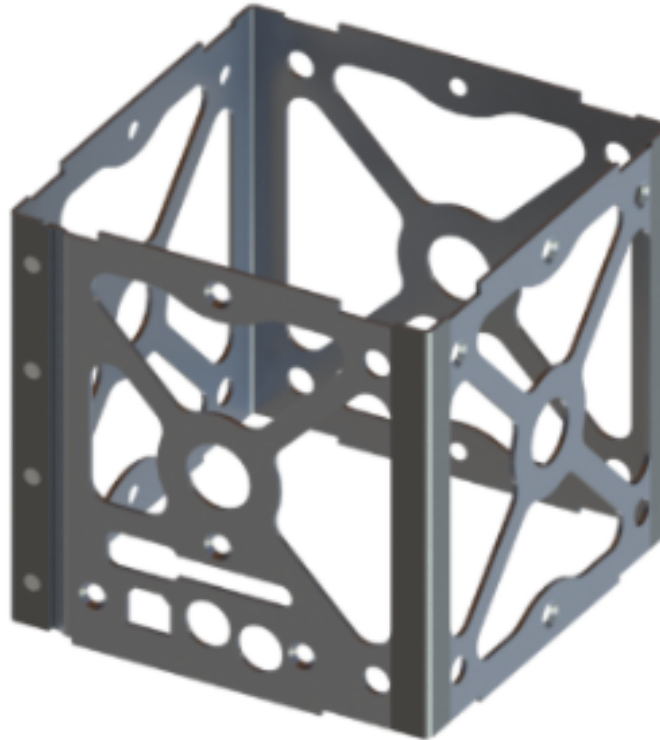


Figure 5.4 – Pumpkin Space Systems CubeSat Kit Skeletonized Chassis Walls [35]

Lastly, with the inclusion of the threaded bracket concept for the top and bottom walls, this leaves an internal plane length of 96.83 mm x 96.83 mm due to wall thickness of the sheet metal with the remaining dimensions available in the table below. This plane area is useful as many of the manufactured units such as the EPS, ADCS and Battery have some length or width sitting just under 95 or 96 mm in some cases.

Table 5.2 – External and Internal Dimensions

Dimension	External (mm)	Internal (mm)
Length	100	96.8
Width	100	96.8
Height	108	100

5.2 ADCS Unit Layout

AAC Clyde Space is a company headquartered in Sweden that “provides small satellite technologies and services that help governments, businesses and institutions access high-quality data from space. Covering satellite components, mission services and space-based data delivery, the company offers end-to-end solutions that turn space-based intelligence into real-world impact. [36]” One of the benefits for using AAC Clyde Space is that they partnered with Berlin Space Technologies to manufacture one of the few ADCS units that can fit into a 1U CubeSat while still possessing three reaction wheels along with three magnetorquers. The iADCS-100 is pictured in Figure 25 which points out the locations of the reaction wheels which are placed specifically so that they provide three axis stabilization.

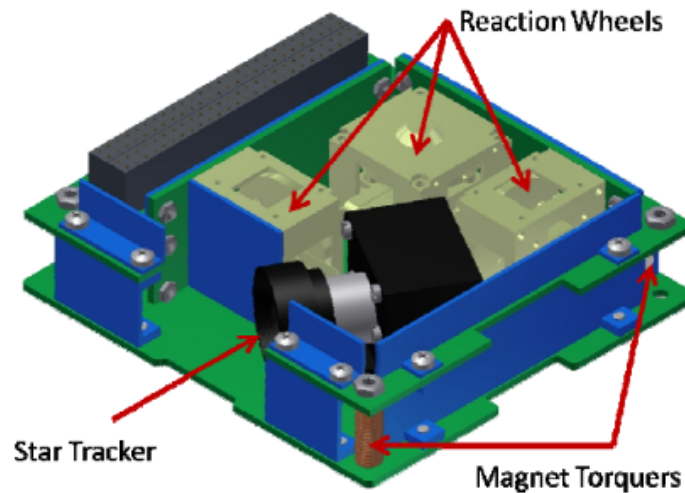


Figure 5.5 – iADCS-100 CAD Drawing [37]

Other components required in the hardware stack include the Optimus 30 CubeSat battery from AAC Clyde designed for smaller CubeSats ranging from 1U all the way up to 3U depending on configuration and payload.

	Product Information		
	Product	NANO-PICO	NANO
	Integrated Battery	Yes 10 or 20Wh	No
	Intended Use	1U CubeSat	2U and 3U CubeSat with body panels
	Mass (typical)	86 g	86 g
	Length	95.89 mm	95.89 mm
	Width	90.17 mm	90.17 mm
	Height*	16.2 mm	16.2 mm

Figure 5.6 – AAC Clyde Space Starbuck Nano (top view and specifications)

The result of all these components is pictured below in Figures 22 and 23 where the structure

can be viewed with the hardware stack and finally with solar panels. The solar panels can simply be fixed to the walls of the structure using M2.5 bolts. With the hardware stack, standoffs and the structure in addition to hardware, the CubeSat weighs 1.14 kg which allows for a remainder of 0.86 kg until the CubeSat must be converted to a 2U or higher. Leaving an additional 0.1 kg for various parts such as harnessing and solder, the CubeSat weighs 1.24 kg total. The mass restriction is a minor constraint as the CubeSat doesn't have a specific payload objective as it is only required to stabilize itself as its primary objective. With the efficiency of manufacturing smaller components becoming much more prevalent, users will be able to weigh the cost of keeping their spacecraft smaller while launch costs continue to become cheaper due to innovation in the industry.

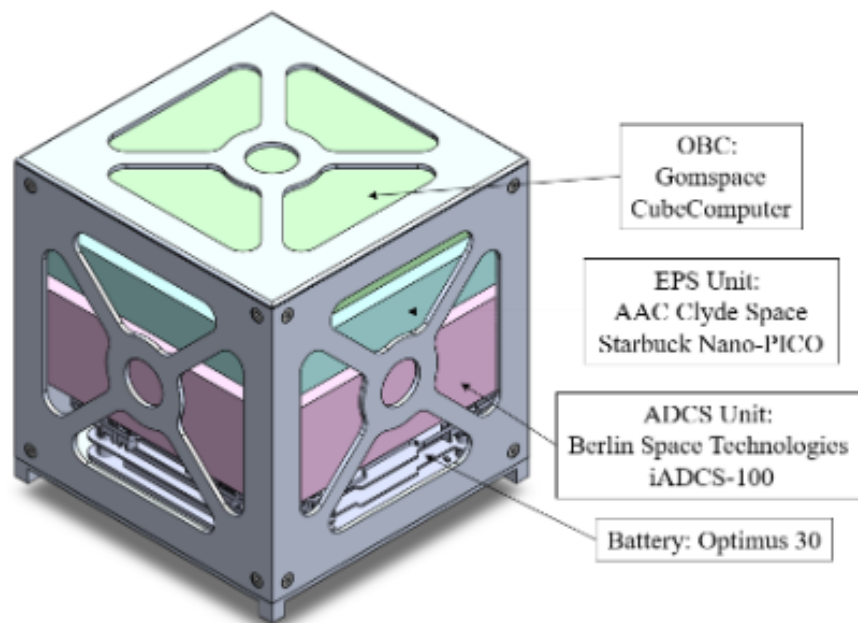


Figure 5.7 – CubeSat subsystem stack

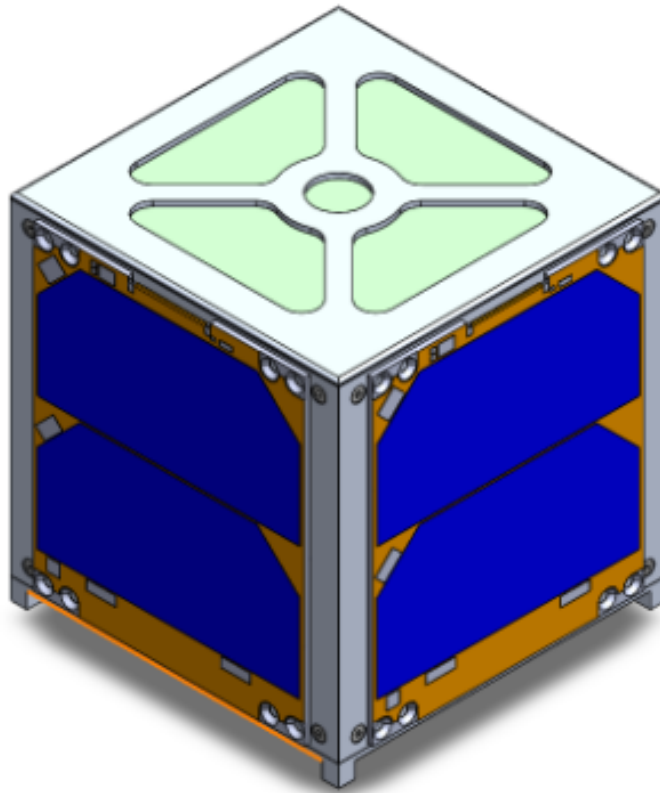


Figure 5.8 – CubeSat as assembled (top solar panel suppressed)

6 Equations of Motion

6.1 General Derivations

In order to find the equations of motion, it is necessary to outline the given variables first. As covered in section 5.2, the total mass of the spacecraft is estimated to be 1.238 kg and the volume was measured to be (10 x 10 x 10.82) cm. In addition to this, the center of mass calculated from Mass Properties in Solidworks was calculated in the table below.

Table 6.1 – Center of mass

Center of Mass (mm)	
X	3.32
Y	-8.18
Z	-3.32

Next, to derive the equations of motion when it comes to torque and angular velocity, the moments of inertia will be crucial as inputs for the controller:

Table 6.2 – Moments of Inertia

Moment of Inertia	
X	0.0022
Y	0.0022
Z	0.0021

Simply put, the equations of motion for a CubeSat in LEO can be written and modeled with just a few known variables. In the simplest terms, the orbit can be assumed to be two dimensional with the orbit taking place on a plane (XY in this case). According to Orbital Mechanics for Engineering Students, displacement in orbit can be expressed as follows [38]:

$$x(t) = a \cos(\omega t + \phi) \quad (6.1)$$

$$y(t) = a \sin(\omega t + \phi) \quad (6.2)$$

$$z(t) = 0 \quad (6.3)$$

With the simplification to a planar circular orbit, the displacement along the z axis can be ignored. By differentiating the equations once and then again, the velocity and acceleration can be modeled respectively by the following equations:

$$x'(t) = -a\omega \sin(\omega t + \phi) \quad (6.4)$$

$$y'(t) = a\omega \cos(\omega t + \phi) \quad (6.5)$$

$$x''(t) = -a\omega^2 \cos(\omega t + \phi) \quad (6.6)$$

$$y''(t) = -a\omega^2 \sin(\omega t + \phi) \quad (6.7)$$

These displacement equations come from the vector $\mathbf{r}$ which is used to denote total displacement or $x(t)$, $y(t)$ and $z(t)$. In terms of orbital mechanics, the two known equations to start with are

$$r = |\vec{r}| \quad (6.8)$$

$$\mu = GM \quad (6.9)$$

Where μ is typically used to denote the standard gravitational parameter. Knowing this, it can be applied to the simplification of the Equation of Gravitational Force:

$$\mathbf{F} = -G \frac{Mm}{r^2} \hat{\mathbf{r}} \quad (6.10)$$

$$\mathbf{F} = m\ddot{\mathbf{r}} \quad (6.11)$$

By applying Newton's Second Law from (35) and setting the force in (34), the equation then becomes (36):

$$m\ddot{\mathbf{r}} = -G \frac{Mm}{r^2} \hat{\mathbf{r}} \quad (6.12)$$

This allows for the mass of the satellite to cancel out:

$$\ddot{\mathbf{r}} = -G \frac{M}{r^2} \hat{\mathbf{r}} \quad (6.13)$$

Further simplifying using (32) defined earlier, the resulting equation is:

$$\ddot{\mathbf{r}} = -\mu \frac{\mathbf{r}}{r^3} \quad (6.14)$$

6.2 Initial Deployment Detumbling

Once a rocket reaches LEO, it is able to deploy its cargo in the P-Pod launcher. During this time when a CubeSat is deployed it will undergo a period of tumbling and spinning in different directions. It is imperative that the satellite reach a point where the rotational velocity is low enough that the reaction wheels are able to take over in order to continuously stabilize the satellite [39].

As covered in chapter 2, the dipole moment of the magnetorquers is necessary to determine how much power is required to continue detumbling. The first equation of importance is the base equation of the amount of torque produced by the dipole moment and magnetic field [40]:

$$\boldsymbol{\tau} = \mathbf{m} \times \mathbf{B} \quad (6.15)$$

From this, it is important to highlight the relationship between the angular velocity and the produced torque involved:

$$\boldsymbol{\omega}_i^T \boldsymbol{\tau} < 0 \quad (6.16)$$

To decrease the kinetic energy of the spacecraft, the control torque τ must be proportional to the angular velocity of the spacecraft. Instead, the magnetic moment is required to be perpendicular to $\omega \times \mathbf{B}$ in order for there torque to be present. After that, implement a scalar gain is necessary to tailor the necessary amount angular velocity relative to the magnetic field [40]:

$$\mathbf{m} = -k (\omega_i \times \mathbf{B}) \quad (6.17)$$

The understanding is that the change in the magnetic field over time is due primarily to the angular velocity leaving another relation:

$$\dot{\mathbf{B}} \approx (\omega_i \times \mathbf{B}) \quad (6.18)$$

Finally, simplifying the past equations leaves a simple controller equation to abide by:

$$\mathbf{m} = -k\dot{\mathbf{B}} \quad (6.19)$$

With this understanding, the B dot controller is much simpler than the average attitude controller given that it is only concerned with the change in magnetic field over time. Due to this, it can be labeled as a Derivative controller as opposed to the required PID for total attitude control.

6.3 Reaction Wheel Controller

In the case of reaction wheels, the controller is based entirely off the angular momentum of the spacecraft and the wheels ability to counteract that momentum [41]. First, by defining momentum as $\mathbf{H}$, the relationship between angular momentum and its differentiation with respect to time as well as the external torques applied to the system by the reaction wheels can be highlighted in equation 47 below where $\mathbf{H}$ and $\mathbf{T}$ are denoted by the angular momentum of the spacecraft and the torque produced by the wheels respectively [42].

$$\dot{\mathbf{H}} = \mathbf{T} - \omega \times \mathbf{H} \quad (47)$$

Next, defining the angular momentum by its definition as it relates to the moment of inertia multiplied by the angular velocity and $\mathbf{h}$ is defined as the angular momentum stored in the wheels.

$$\mathbf{H} = \mathbf{I}\omega + \mathbf{h} \quad (48)$$

By differentiating equation 48 and plugging it into the initial equation (47) and simplifying, the final equation is pictured as below:

$$\mathbf{I}\dot{\omega} = \mathbf{T} - \dot{\mathbf{h}} - \dot{\mathbf{I}}\omega - \omega \times \mathbf{H} \quad (49)$$

Finally, since the CubeSat is not experiencing a change in moment of inertia like some spacecraft do during a deployment phase such as with solar array deployment, the third term can be entirely disregarded.

$$\mathbf{I}\dot{\omega} = \mathbf{T} - \dot{\mathbf{h}} - \omega \times \mathbf{H} \quad (50)$$

Looking at the final equation, the relationship to the change in angular velocity of the spacecraft over time is directly proportional to the external torque experienced by the spacecraft, the change

in momentum with respect to time and the gyroscopic torque.

7 Controller Simulations

7.1 B Dot Controller

As mentioned in previous chapters, during deployment from the P-Pod into LEO, the CubeSat is subject to spinning and tumbling which requires damping and overall stabilization. This was done through the use of the included magnetorquers used in the iADCS-100. Unfortunately, the iADCS-100 does not have a specifications sheet publicly available that lists the model of magnetorquers used in the ADCS unit. Because of this, estimations made from the Small Spacecraft Technology Report [26] was required. Looking at a table of all publicly available magnetorquers, there are many options on the smaller end that list the dipole moment range for smaller CubeSats at $0.2 - 0.5 \frac{\text{A}\cdot\text{m}^2}{\text{A}}$. For this reason, in the controller script for the B dot algorithm controller, the input for the dipole moment of the magnetorquers was reasonably listed at $0.15 \frac{\text{A}\cdot\text{m}^2}{\text{A}}$. On a side note, the amperage was also set at 0.5 A per magnetorquer which is more than reasonable for the included iADCS-100 [37].

In regard to the LEO simulated inputs, the magnetic field of Earth when a satellite is at LEO was set at $50 \mu\text{T}$ [43]. The remaining values regarding orientation were arbitrary values set so the CubeSat could damp its angular velocity to a near zero value for the reaction wheels to take over. When it comes to attitude determination and control, this is the relatively simple but crucial step initially required for the attitude control to initiate at all. While the controller itself reads matrices, it is using the scalar values and positioning coordinates to read the level of rotation or "tumbling" taking place over time.

Instead, the quaternion commands located in the simulation are purely used for the simulation explaining the satellite's attitude over time. The controller itself only takes in information regarding the magnetic field and rotation.

Observing the first two figures, the total amperage and dipole moment listed over time seem to oscillate but still decrease over time and the CubeSat's angular velocity is dampened over time.

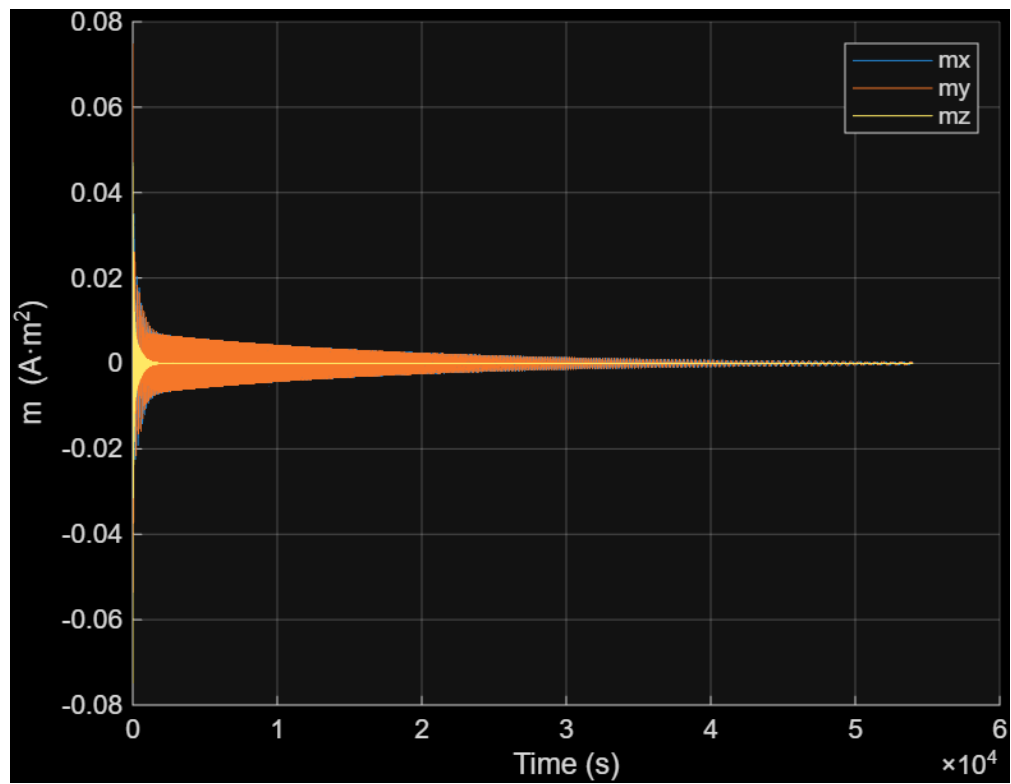


Figure 7.1 – Dipole moment over time

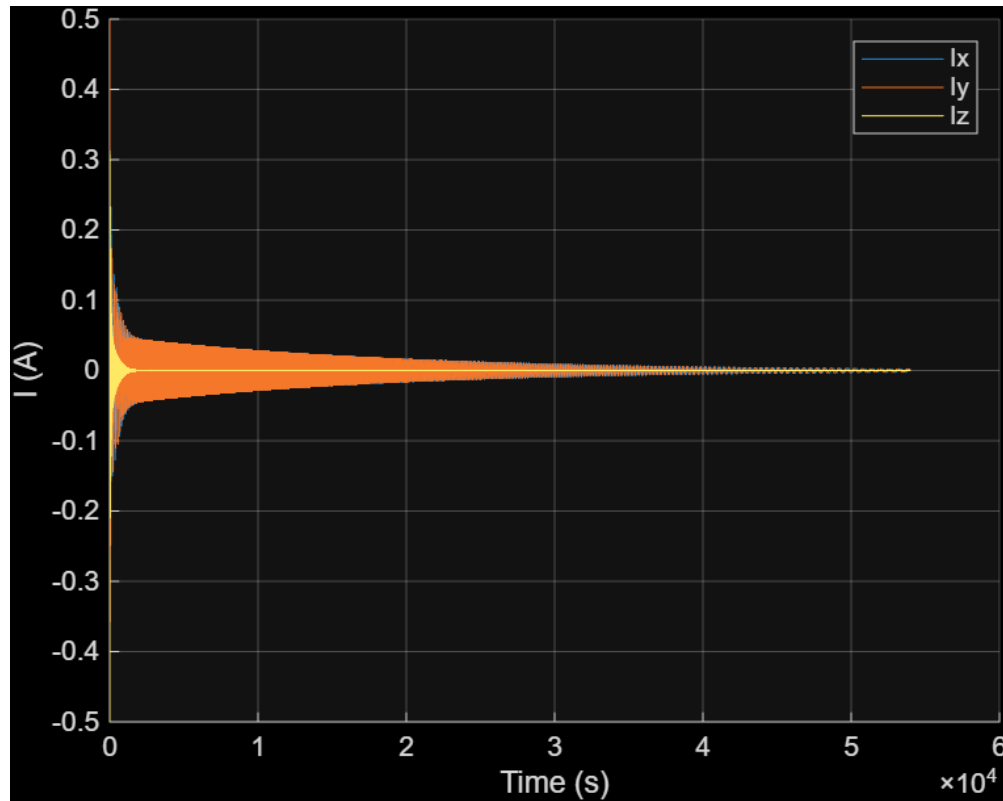


Figure 7.2 – Amperage use over time

Referring back to equation 45 in Chapter 6, it is observed that the relationship between the magnetic field over time is directly proportional to the cross product of the angular velocity and magnetic field itself. So, as the angular velocity observed in Figure 31 reaches a constant value, the change in magnetic field over time will go to zero signaling stability. Given that the damping value occurs from the start time all the way to the 15 hour mark, the CubeSat is sufficiently stable across the axes perpendicular to the direction of the magnetic field vector. Looking closer at that time frame, it can be observed that there are no oscillations in the change in angular velocity. This means that the torque produced could be increased all the way up unto the point that the torque produced causes some type of saturation or jitter. Fifteen hours is a more than realistic expectation however for stabilization to the point that the reaction wheels are able to take over.

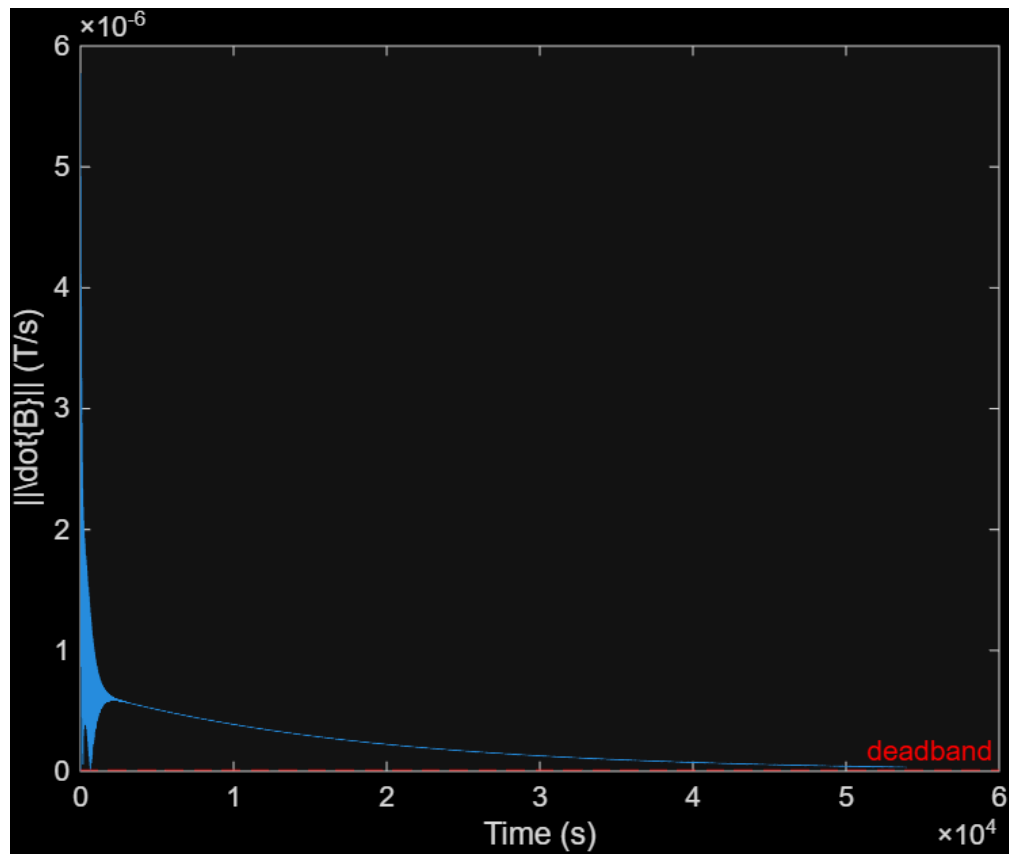


Figure 7.3 – Change in magnetic field over time

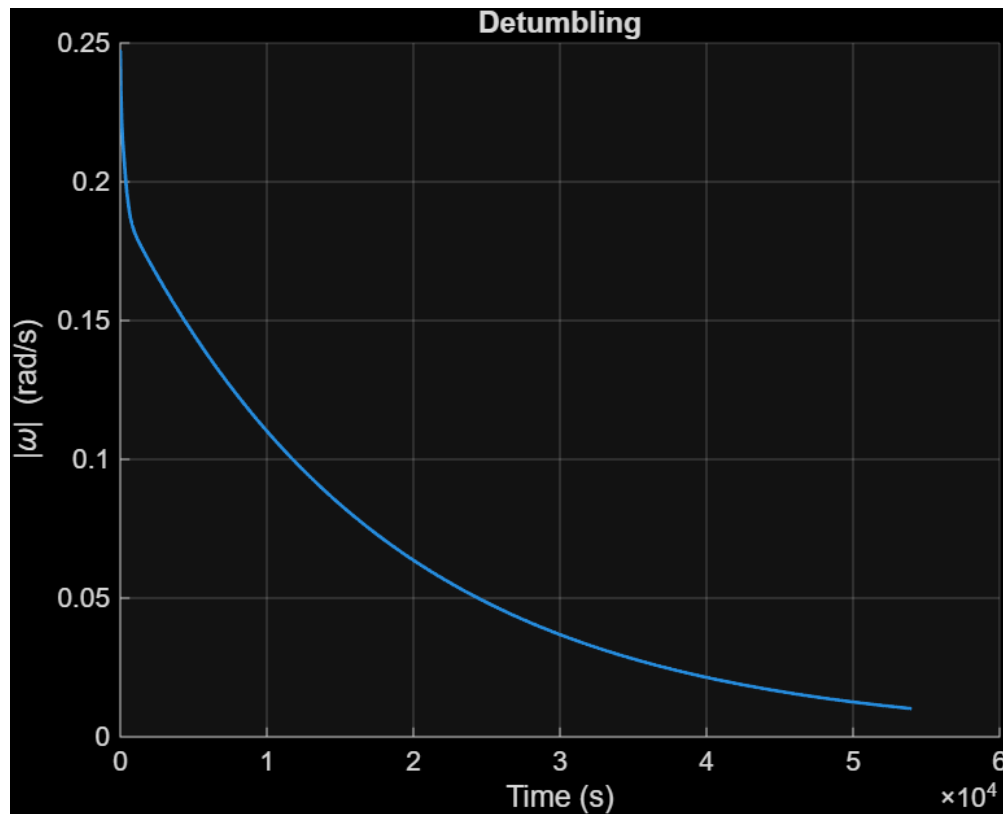


Figure 7.4 – Change in angular velocity over time

7.2 Reaction Wheel Stabilization Controller

7.2.1 One Axis Perturbation

Before introducing three axis stabilization, it was necessary to continue testing various reaction wheels based on peak torque and momentum capacity. Initially, the Blue Canyon RWP015 was selected as one of the smaller reaction wheels which seemed sufficient based on mass constraints. However, there were issues with wheel saturation especially in the beginning of the simulation with just a fifteen degree perturbation along the x axis. Instead, the wheel would saturate and was unable to stabilize given a half minute of time. Note that this perturbation is simplistic in that the spacecraft lacks any additional inertia experienced during detumbling. Instead, the spacecraft is simply being started fifteen degrees from its set position without any additional force being applied to its rotation.

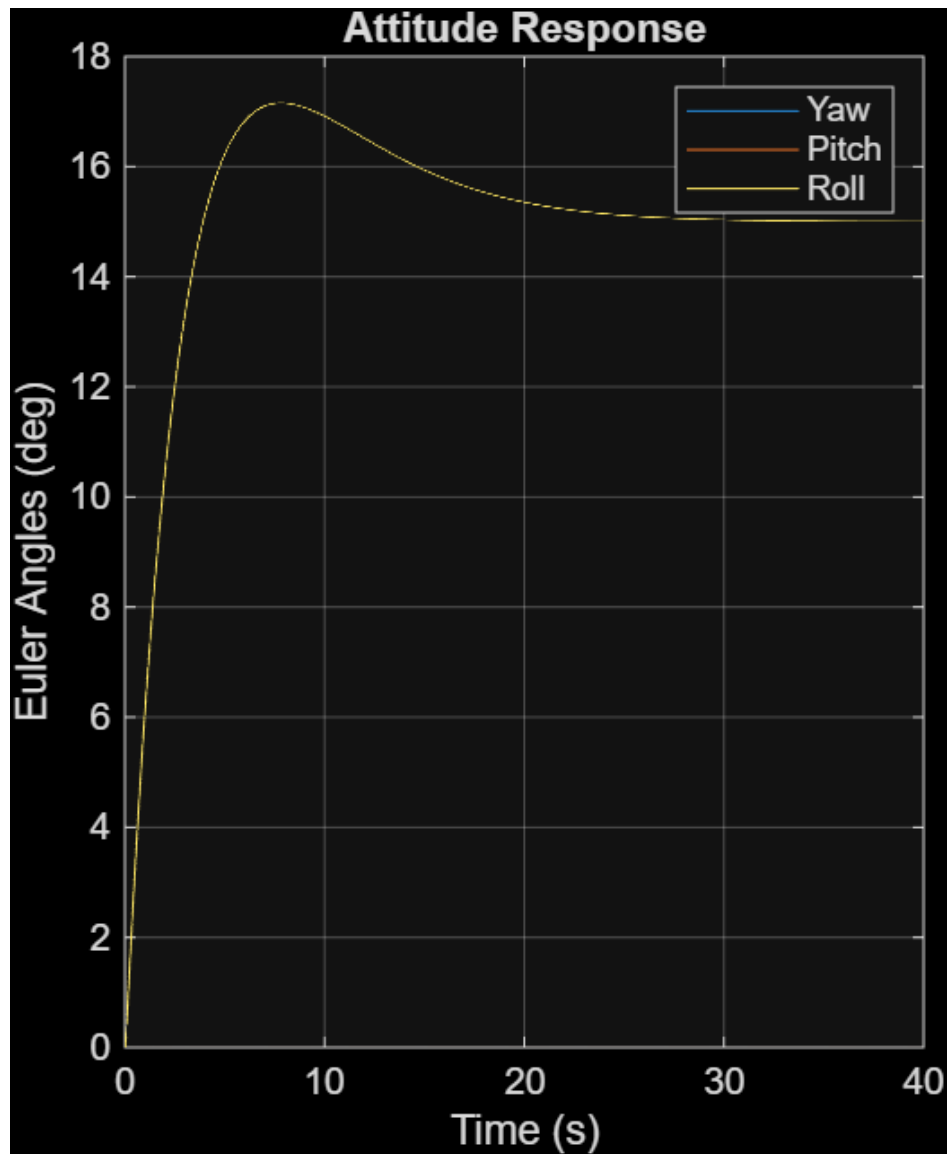


Figure 7.5 – Euler angle change over time (RWP015)

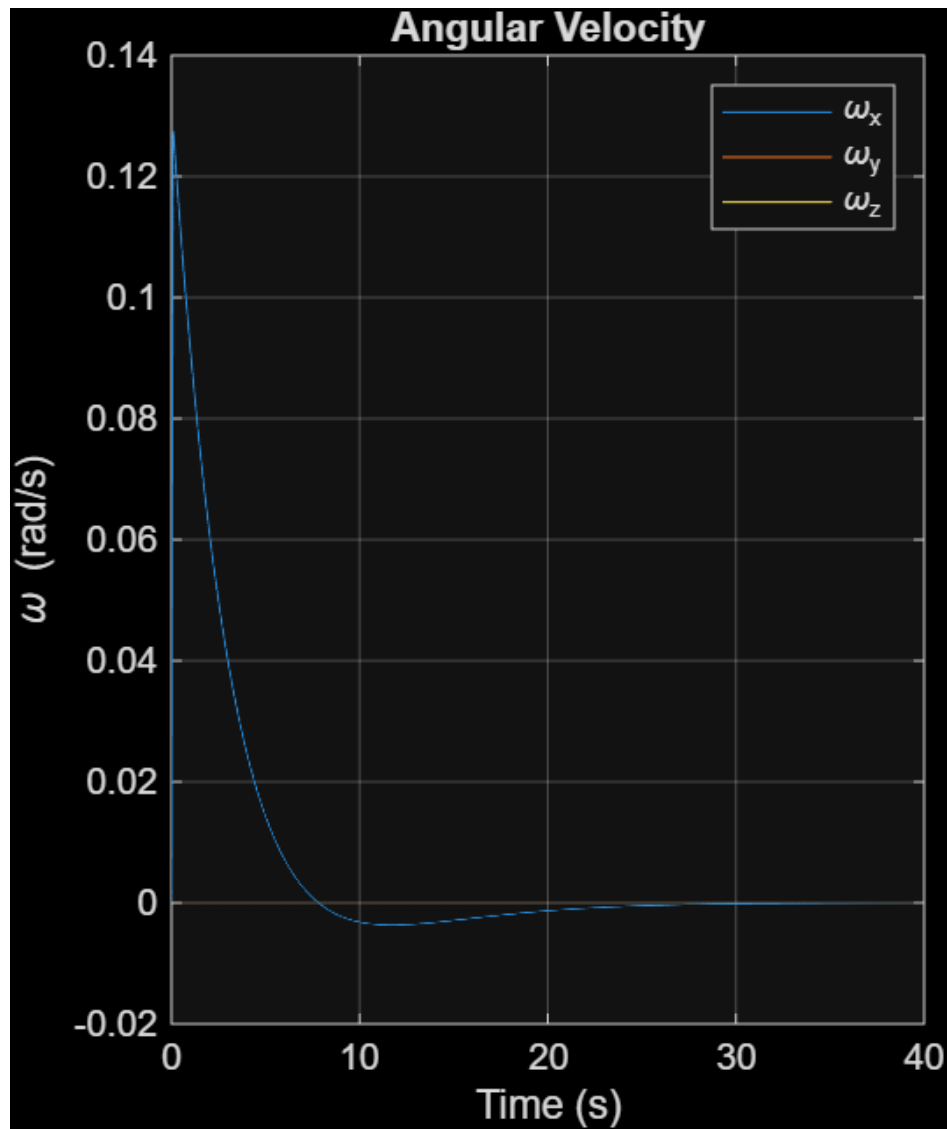


Figure 7.6 – Angular velocity of one axis perturbation over time (RWP015)

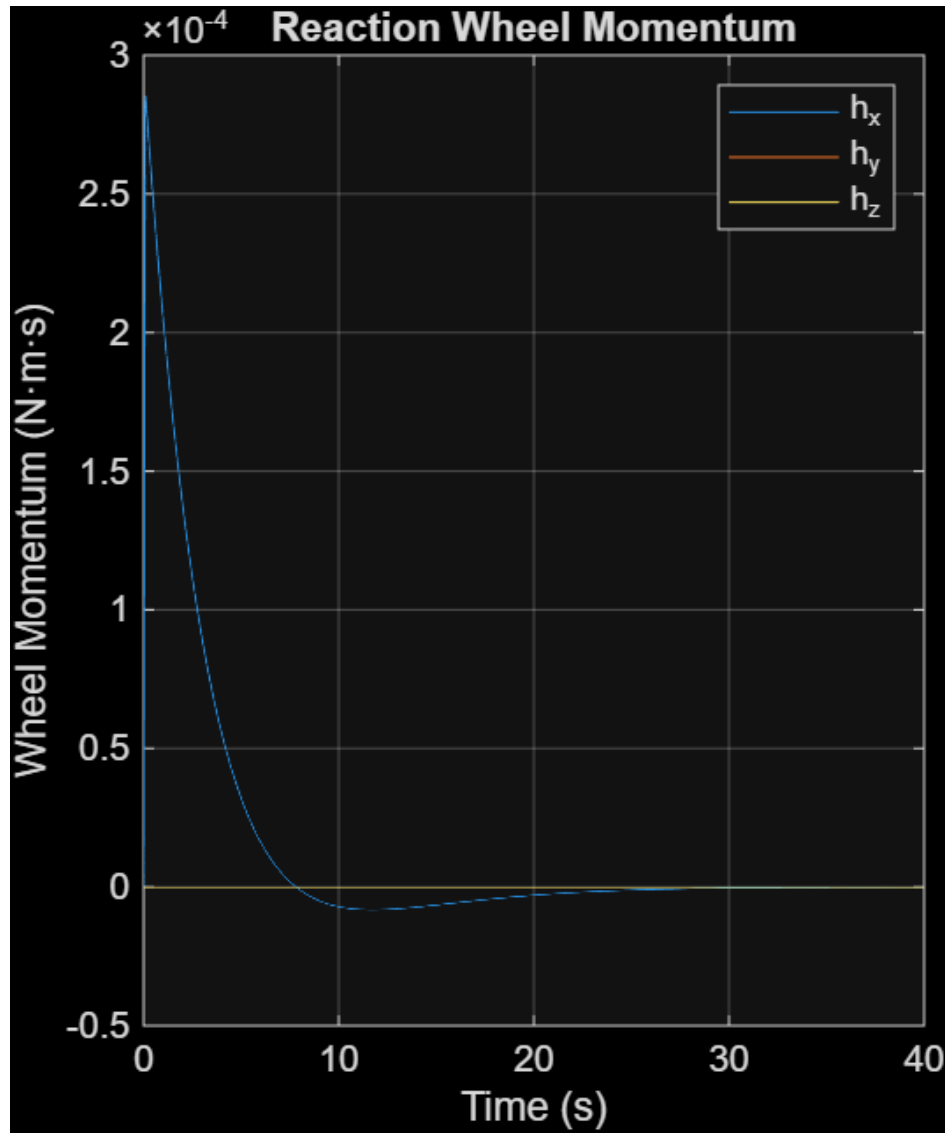


Figure 7.7 – Angular momentum of one axis perturbation over time (RWP015)

With this knowledge, another model of reaction wheels were selected. Instead, the RWP050, one size up relative to Blue Canyon's options were selected based on power constraints which remained the same. The RWP050 is double the size of the RWP015 but provides almost double the peak torque and an even larger amount of momentum capacity. Running the same simulation with different variables and giving the reaction wheels enough time to stabilize at under 5 minutes yields a smooth transition to the 15 degree change in orientation. According to the plots below, there is no saturation or jitter visible during the transition. While the time span for correction had the ability to be shortened, the proportional gain was kept at 0.015 to prevent any sudden shifts and saturation of the wheels.

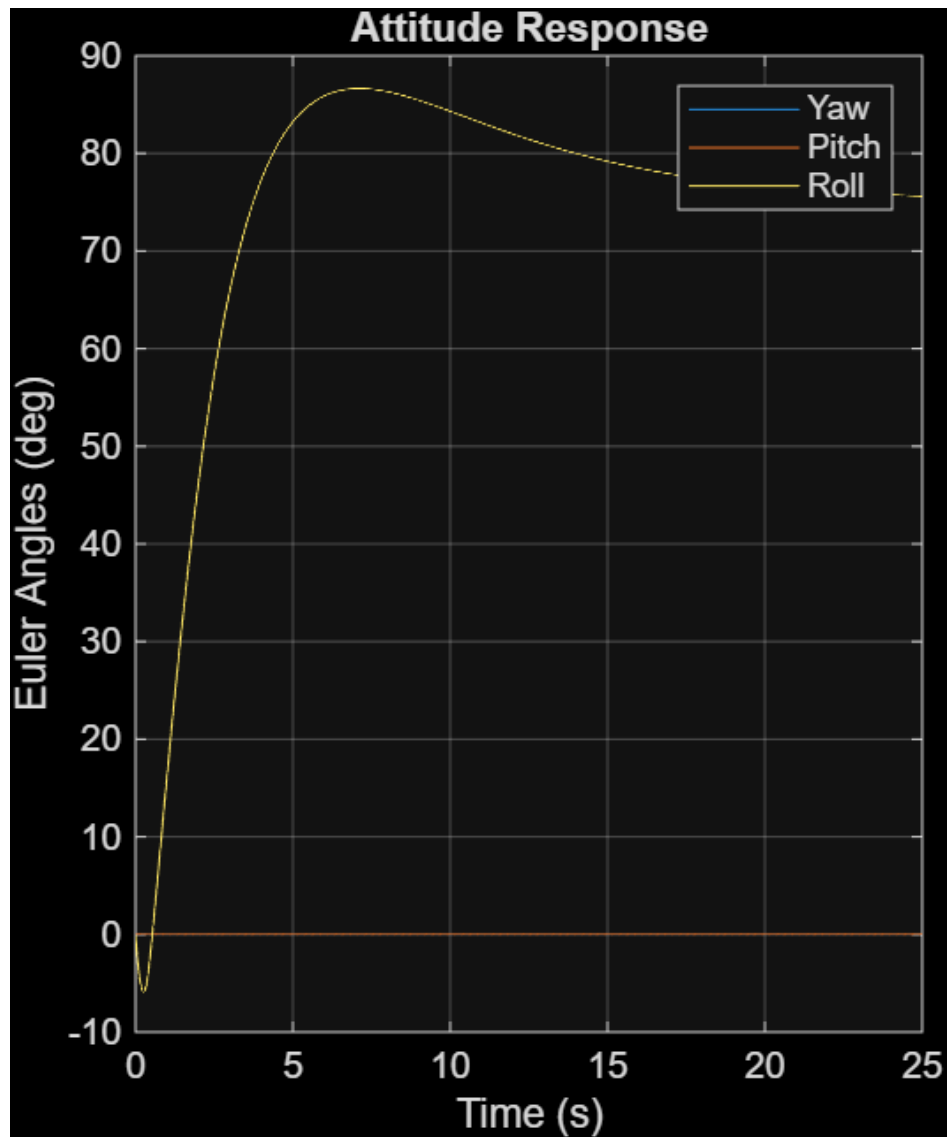


Figure 7.8 – Angular momentum of one axis perturbation over time (RWP050)

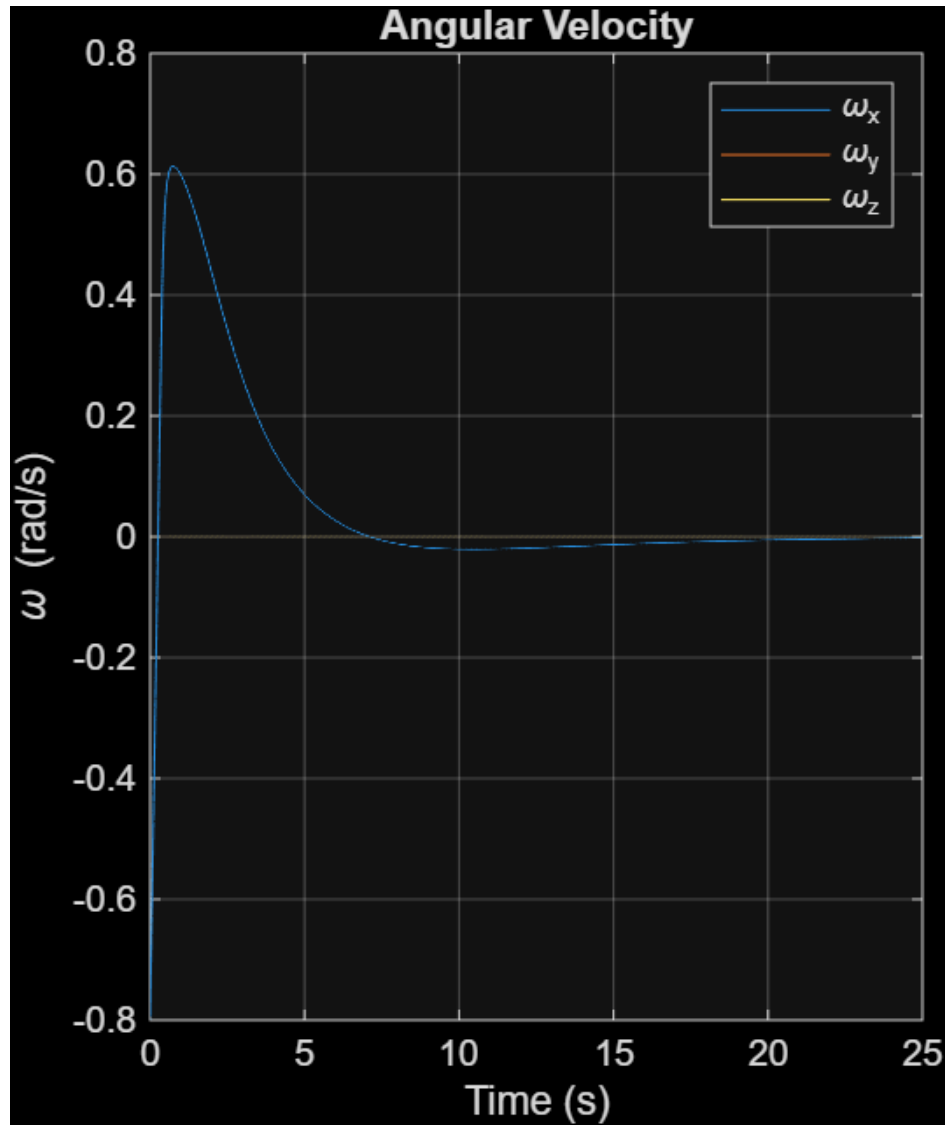


Figure 7.9 – Angular momentum of one axis perturbation over time (RWP050)

Comparing the angular velocity of the wheels between the two models, it's observable that the stabilization of the spacecraft can be achieved by the second model of reaction wheels in a lesser amount of time without saturation in the beginning even with an additional sixty degrees to reorient itself. In a little under five seconds, the spacecraft has corrected itself for the most part with the remaining work an order of magnitude less than the initial torque required as the angular velocity decreases at an exponential rate.

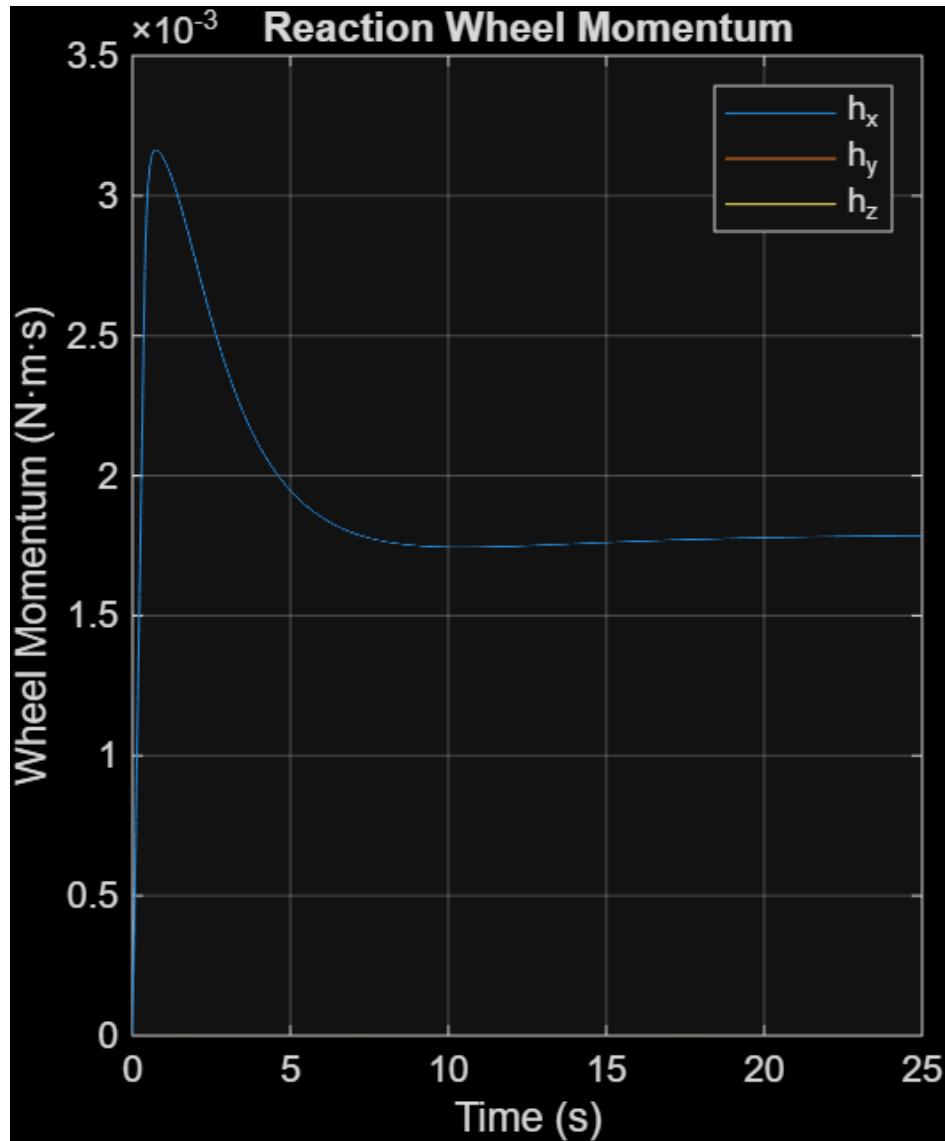


Figure 7.10 – Angular momentum of one axis perturbation over time (RWP050)

7.2.2 Reaction Wheels Acting Under Perturbation

7.3 Dual Attitude Control

After finalizing the controller with additional LEO parameter commands from MATLAB, the simulation was more realistic. In addition, the gains were adjusted with the final derivative gain being lowered to 0.025 whereas it had initially been 0.2 which required a much higher time span for stabilization. Even with an initial angular velocity in tumbling at 4.0 rad/s and a slew of 75 degrees along all axes, the CubeSat was able to correct itself within a 5.0 second time frame. This can be observed in Figure 38 below where the responses are pictured showing an initially high non zero value due to relatively high tumbling and angles to correct. Within just a few seconds, the angles collapse to a more correct level given the commands. After that, the angles slowly approach their desired alignment. With the proportional gain selected at 0.025 allows for quick torque towards

the desired attitude. There is no visible oscillation which suggests that the controller is critically or overly damped.

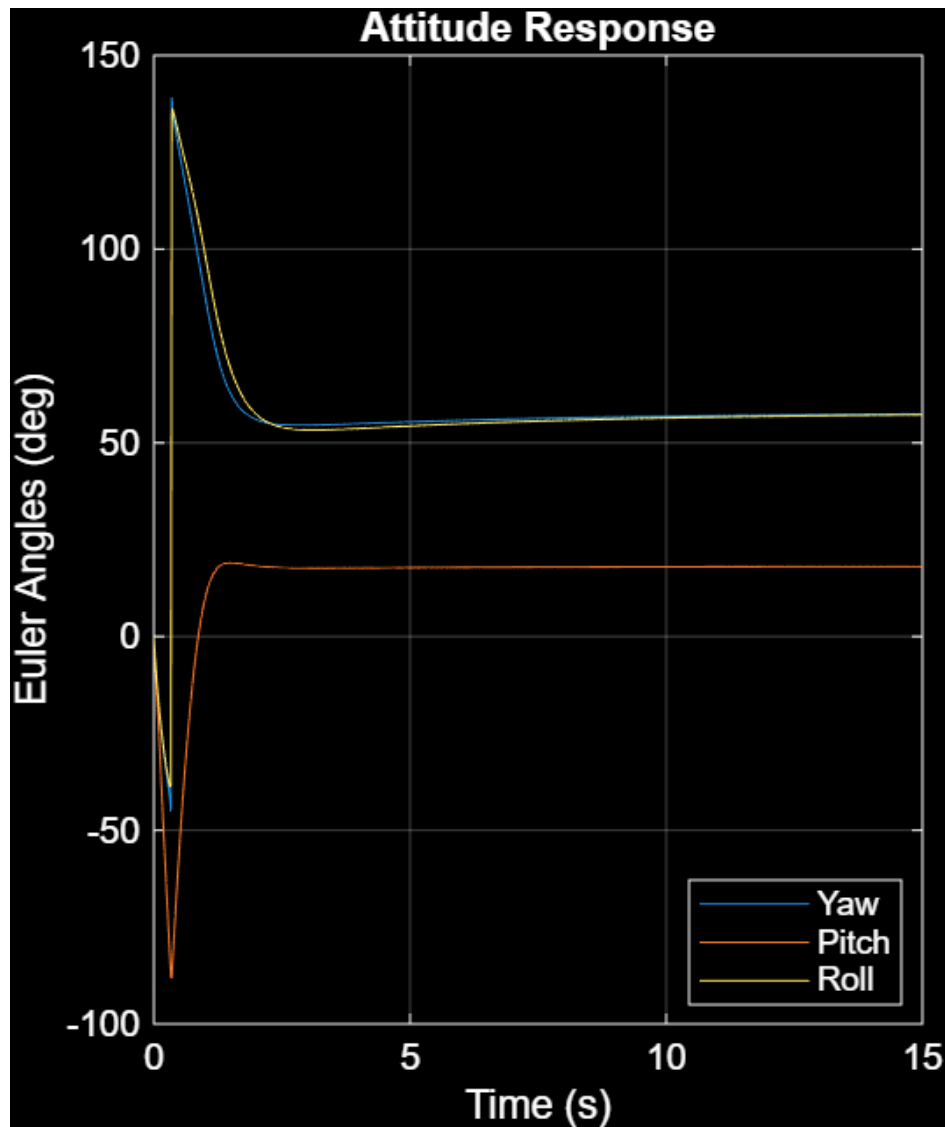


Figure 7.11 – Attitude response of three axis tumbling (RWP050)

Looking now at the rotation error, there is an exceptionally large peak towards 180 degrees which is a combination of the initially high attitude error or slew of 75 degrees and the relatively high body rates during tumbling. Again, in under 2.0 seconds, the error collapses under 10 degrees and gradually approaches zero. The gradual approach is due to the integral gain which slowly corrects the error under disturbance.

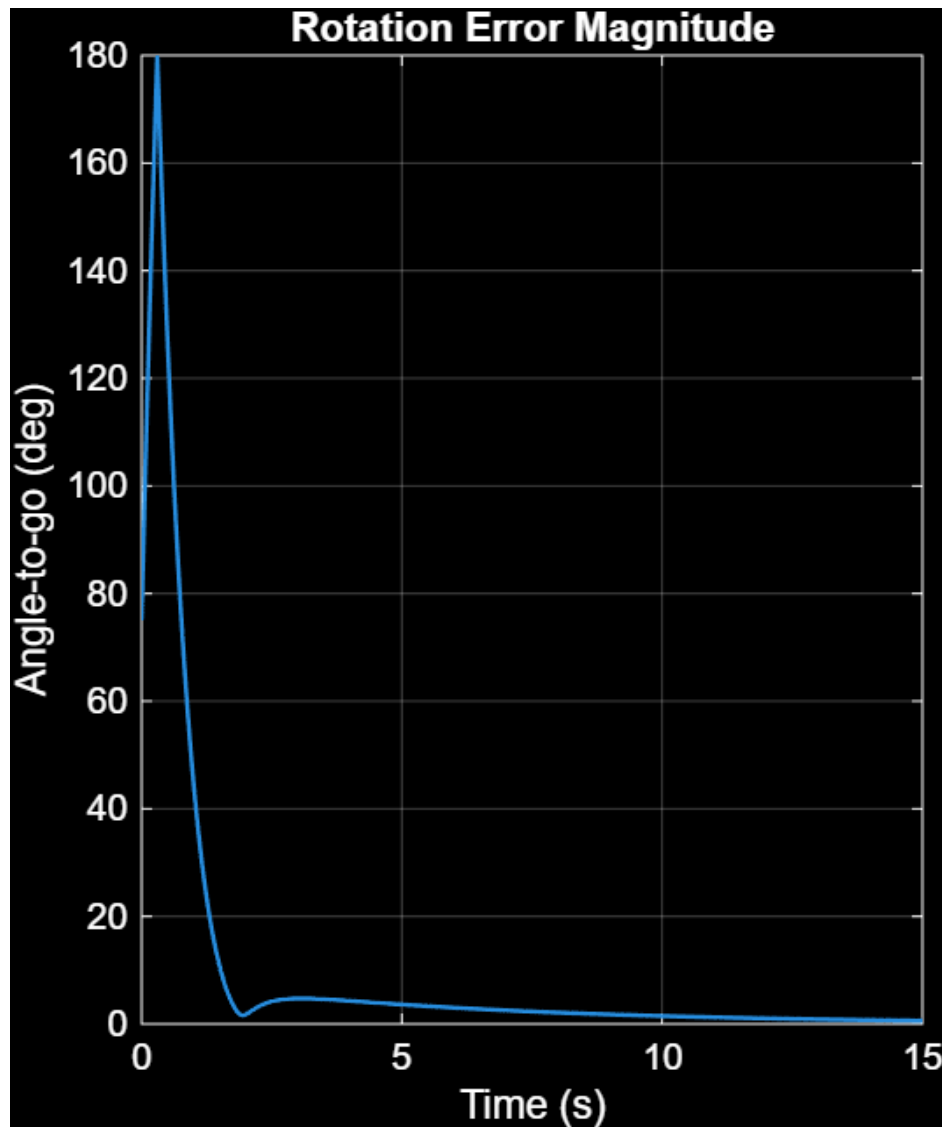


Figure 7.12 – Rotation error of three axis tumbling (RWP050)

Observing the body rates now, it is evident that all axes are experiencing a tumbling angular velocity of 4.0 rad/s. In under 2.0 seconds similar to the other graphs, the body rates have collapsed to zero and in a relatively smooth manner which suggests the derivative gain is correctly valued. In the case of high derivative gain that was initially selected, the results can be viewed in Figure 41 which shows both the body rate and rotation error. The main issue with having a gain set at 0.2 can be seen at the 10.0 second point where the error bounces after reaching zero. The reaction wheels sharply correct the orientation but do so with such a high amount of the torque, the wheels are unable to completely stop the inertial forces that result from it causing a bouncing effect back and forth. With a high enough gain, this could result in the wheels never fully having the capability to correct the spacecraft causing it to continuously correct the errors the wheels create which never produces full stabilization as a result.

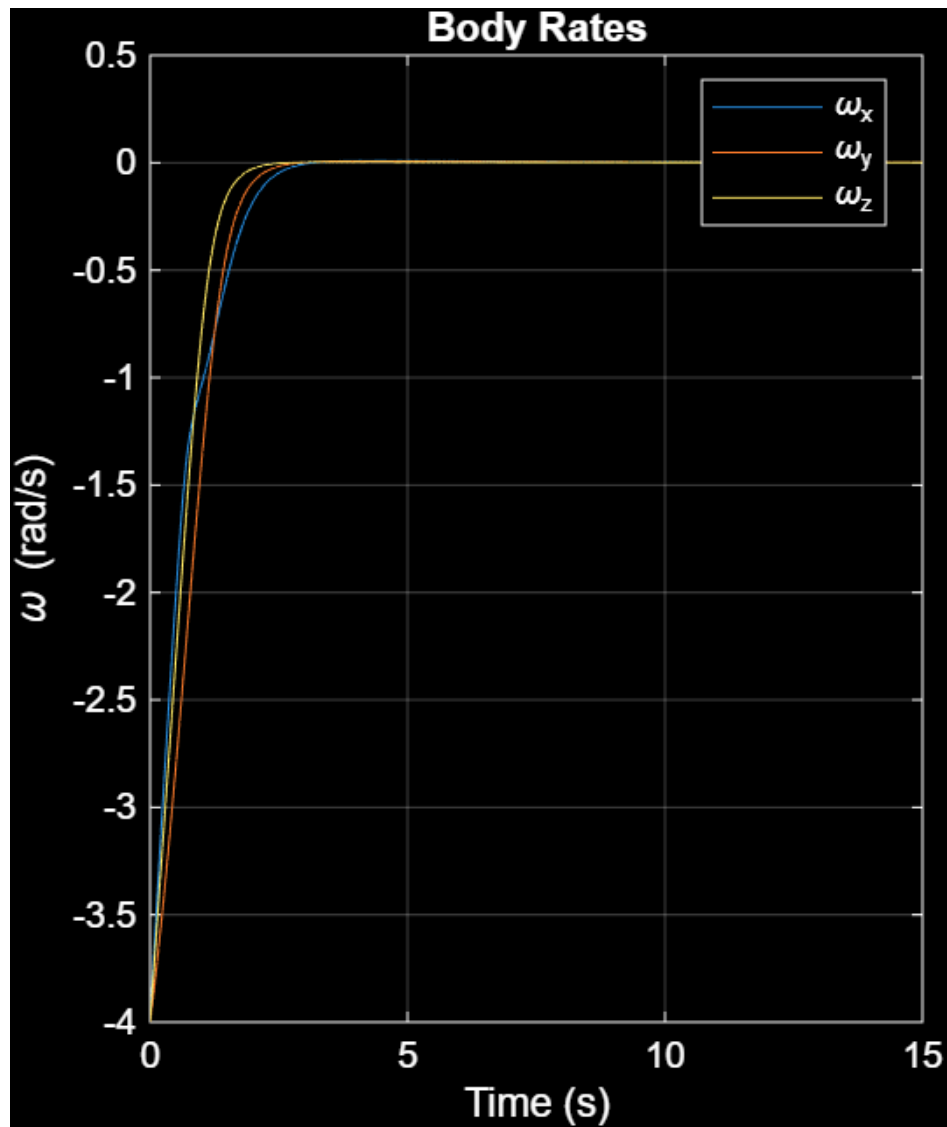


Figure 7.13 – Body rates of three axis tumbling (RWP050)

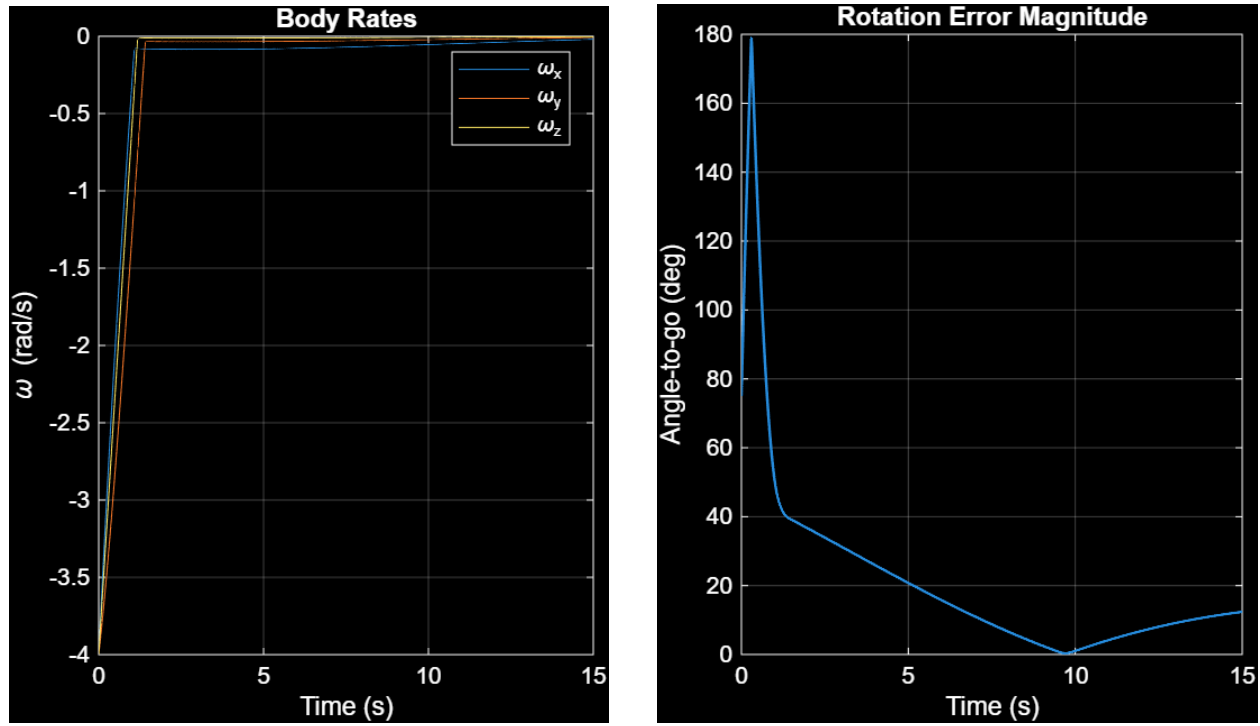


Figure 7.14 – Body rates and rotation error with high derivative gain

Finally, looking to Figure 42 at the momentum produced by the reaction wheels, the wheels never fully exceed their storage limit of 0.05 Nms. Once the spacecraft is corrected around the 2.0 second mark previously mentioned, the wheels reach a limit that is then continuously held in order to counteract the disturbances experienced at LEO. No saturation occurs according to the graph and the spacecraft is sufficiently stabilized.

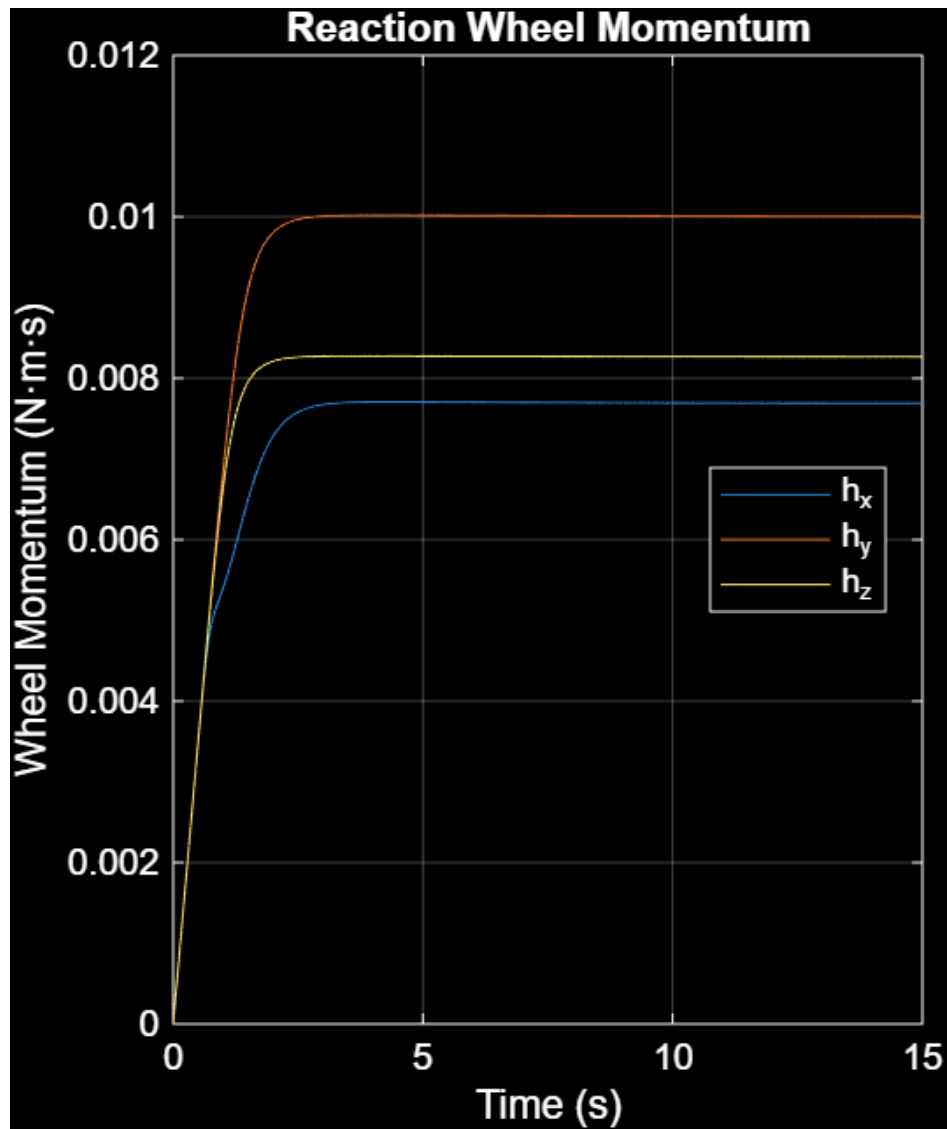


Figure 7.15 – Reaction wheel momentum of three axis tumbling (RWP050)

8 Future Work

The primary objective of this project initially was to develop an attitude control system in its entirety. Given the amount of funding and time required, the project was scaled back to just the research and development portion of the project. Even then, the control system is in its initial phase with the controller scripts being used to simulate the capabilities of a 1U CubeSat containing a set of magnetorquers and reaction wheels in an LEO environment under realistic perturbations.

The next step would then be to convert the scripts into separate functions and model them in Simulink for a more palatable view in understanding how the controller works. Next, the reaction wheels and magnetorquers would need to be combined so as to allow for more stable attitude control.

The final step would be to combine the remaining subsystems into realistic functions that allow the user to observe power usage along all subsystems and their components. These would then be added to the Simulink completing the controller and allowing for a more realistic representation of how the entire system works.

9 Conclusion

Comparing the initial selection of the RWP015 reaction wheels to the RWP050 model, it is evident that the slightly upscaled model is more suited to being used in the attitude control. The model is less susceptible to wheel saturation and jitter when required to correct itself over a reasonable time span. The RWP050 were able to sufficiently stabilize the spacecraft in under 5 seconds even with an initial perturbation of 4.0 rad/s to consider along with a slew of 75 degrees along all three axes from the desired orientation.

Likewise, the magnetorquers selected are able to correct the CubeSat during the detumbling phase with no assistance from the reaction wheels. Over an average time of fifteen hours, the magnetorquers serve as a sufficient method of detumbling and did not require an upscaled model.

Although the components in other subsystems weren't implemented into the controller, they were selected for compatibility with the reaction wheels and magnetorquers and the mass and sizing requirements required from a 1U CubeSat.

Bibliography

- [1] H. W. Jones. The recent large reduction in space launch cost. In *48th International Conference on Environmental Systems*, pages 1–10, July 2018.
- [2] S. M. Spremo, A. R. Crocker, and T. L. Panontin. Small spacecraft overview. Technical report, NASA Technical Reports Server (NTRS), May 2017. Accessed 3 Oct. 2025.
- [3] B. M. Bomani. CubeSat technology past and present: Current state-of-the-art survey. Technical report, NASA Technical Reports Server (NTRS), December 2021. Accessed 1 Oct. 2025.
- [4] P-POD CubeSat ejector. <https://www.planetary.org>, 2023. Accessed 10 Feb. 2025.
- [5] C. Cappelletti, S. Battistini, and B. K. Malphrus. *CubeSat handbook: From mission design to operations*. Academic Press, London, 2021.
- [6] D. S. Roldugin, S. S. Tkachev, S. A. Bogachev, and S. V. Kuzin. Comparing reaction wheels and magnetorquers performance in the CubeSat solar observatory stabilization. *Cosmic Research*, 62:444–453, October 2024.
- [7] S. A. Khan, Z. Tao, S. Fahad, M. Salman, M. Tahir, and A. Ali. Reliable attitude control integrating reaction wheels and embedded asymmetric magnetorquers for detumbling CubeSats. *Advances in Space Research*, 74:5906–5922, December 2024.
- [8] Fraser Cain. Spacecraft gyroscopes and reaction wheels: You can never have enough. <https://www.universetoday.com>, August 2021. Accessed 22 Mar. 2025.
- [9] S. S. Nudehi, U. Farooq, A. Alasty, and J. Issa. Satellite attitude control using three reaction wheels. In *2008 American Control Conference*, pages 4850–4855, June 2008.
- [10] N. Jovanovic, B. Riwanto, P. Niemela, M. R. Mughal, and J. Praks. Design of magnetorquer-based attitude control subsystem for Foresail-1 satellite. *IEEE Journal on Miniaturization for Air and Space Systems*, 2:220–235, December 2021.
- [11] K. Ogata. *Modern control engineering*. Prentice-Hall, Upper Saddle River, N.J., 2010.
- [12] M. Araki. PID control. In *Control systems, robotics and automation*, volume 2. Encyclopedia of Life Support Systems (EOLSS), 2010.
- [13] Toney Lacey. Chapter 11: Tutorial – the Kalman filter. In *Control system theory*. 2022.
- [14] W. Farag. Kalman-filter-based sensor fusion applied to road-objects detection and tracking for autonomous vehicles. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 235:1125–1138, December 2020.
- [15] E. A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. IEEE, 2000.

- [16] R. Tedrake. Chapter 8: Linear quadratic regulators. In *Underactuated robotics*. 2024.
- [17] M. T. Augustine. A note on linear quadratic regulator and Kalman filter. *arXiv preprint arXiv:2308.15798*, 2023.
- [18] A. C. Stickler and K. T. Alfried. Elementary magnetic attitude control system. *Journal of Spacecraft and Rockets*, 13:282–287, 1976.
- [19] P. A. Capo-Lugo and P. M. Bainum. *Orbital mechanics and formation flying: A digital control perspective*. PhD thesis, Cambridge, 2011.
- [20] P. A. Capo-Lugo, J. Rakoczy, and D. Sanders. The B-dot earth average magnetic field. *Acta Astronautica*, 95:92–100, 2014.
- [21] Versatile onboard computer for Cube, nano and microsat missions (NanoMind A3200). <https://gomspace.com/shop/subsystems/command-and-data-handling/nanomind-a3200.aspx>, 2021. Accessed 15 Mar. 2025.
- [22] ISIS onboard computer. <https://www.isispace.nl/product/on-board-computer/>, 2024. Accessed 14 Mar. 2025.
- [23] CubeADCS gen 2. <https://www.cubespace.co.za/uploads/a55e9627-9377-496b-800e-ecde36e3b21e.pdf>, 2024. Accessed 14 Mar. 2025.
- [24] Nano. <https://docs.arduino.cc/hardware/nano/>, 2025. Accessed 13 Mar. 2025.
- [25] 1u solar panel x/y. <https://www.endurosat.com/products/1u-solar-panel-x-y/>, 2025. Accessed 13 Mar. 2025.
- [26] NASA Small Spacecraft Systems Virtual Institute. 2023 state-of-the-art small spacecraft technology report. Technical report, 2023. Accessed 3 Oct. 2025; Page last updated 13 Dec. 2023.
- [27] Blue canyon technologies: Spacecraft buses, systems and solutions. <https://www.bluecanyontech.com/>, 2025. Accessed 17 Feb. 2025.
- [28] G. D. Krebs. CubeSat, 2025. Online since 1996.
- [29] IT-SPINS. <http://www.ssel.montana.edu/projects/itspins.html>, 2024. Accessed 8 May. 2025.
- [30] M. L. Handley. *Design, fabrication, and implementation of an embedded flight computer in support of the ionospheric–thermospheric scanning photometer for ion-neutral studies CubeSat mission*. PhD thesis, Thesis, 2017.
- [31] J. P. Mason, M. Baumgart, B. Rogler, C. Downs, M. Williams, T. N. Woods, S. Palo, P. C. Chamberlin, S. Solomon, A. Jones, et al. MinXSS-1 CubeSat on-orbit pointing and power performance: The first flight of the Blue Canyon Technologies XACT 3-axis attitude determination and control system. *arXiv preprint arXiv:1706.06967*, 2017.

- [32] C. Dennehy and O. S. Alvarez-Salazar. A survey of the spacecraft line-of-sight jitter problem. *AAS*, 19:1–14, 2019.
- [33] J. Guo and C. Han. Where is the limit: The analysis of CubeSat ADCS performance. https://www.researchgate.net/publication/311279495_WHERE_IS_THE_LIMIT_THE_ANALYSIS_OF_CUBESAT_ADCS_PERFORMANCE, May 2016. Accessed 9 Jun. 2025.
- [34] Aluminum 6061 vs 7075, 2023. Accessed 1 Oct. 2025.
- [35] CubeSat Kit™ chassis walls, 2025. Accessed 14 Apr. 2025.
- [36] Investor relations, 2025. Accessed 12 Apr. 2025.
- [37] T. Segert, S. Engelen, M. Buhl, and B. Monna. iADCS-100—an autonomous attitude determination and control subsystem based on reaction wheels and star tracker in 1/3u package. In *4S Symposium*, 2012.
- [38] H. D. Curtis. *Orbital mechanics for engineering students*. Butterworth–Heinemann, Amsterdam: Elsevier, 2014.
- [39] S. A. Khan, Y. Shiyouni, A. Ali, M. Tahir, S. Fahad, and S. Rao. CubeSat detumbling using only embedded asymmetric magnetorquers. *Advances in Space Research*, 71:2140–2154, March 2023.
- [40] I. Ofodile, H. Ehrpais, A. Slavinskis, and G. Anbarjafari. Stabilised LQR control and optimised spin rate control for nanosatellites. In *2019 9th International Conference on Recent Advances in Space Technologies (RAST)*, pages 715–722, June 2019.
- [41] M. Bagheri, M. Kabganian, and R. Nadehi. Stable design of attitude control for a spacecraft. *Advances in the Astronautical Sciences*, 145:777–788, 2012.
- [42] S. R. Starin and J. Eterno. Attitude determination and control systems. Preprint 20110007876, NASA Goddard Space Flight Center, Greenbelt, MD, United States, January 2011. Southwest Research Institute, TX, United States.
- [43] A. Neumann and N. Mühlich. Ground-based experiment for electric propulsion thruster plume—magnetic field interaction. *Aerospace*, 10:117, January 2023.

A. MATLAB Code

1 B dot Controller

The following MATLAB code was used to simulate the B dot controller and its detumbling.

Listing A.1 –: B dot Controller Script

```
function [m_cmd, I_cmd] = bdot_controller(B_body, Ts, k, alpha,
    deadband, m_max, Kmt, I_max)
% B-DOT detumbling controller (3-axis)

% declaration of variables that persist between function loops
persistent B_prev Bdot_prev isInit
if isempty(isInit)
    B_prev      = B_body;
    Bdot_prev   = zeros(3,1);
    isInit      = true;
end

% Time sample
Ts = max(Ts, eps);
alpha = min(max(alpha,0),1); % alpha is kept to 0 and 1

% derivative
Bdot = (B_body - B_prev) / Ts;

% First order infinite impulse response filter (low pass filter
    to reduce noise)
Bdot_f = alpha * Bdot + (1 - alpha) * Bdot_prev;

% range of input values within which a system does not respond or
    initiate
% any action.
if norm(Bdot_f) < deadband
    Bdot_f = zeros(3,1);
end

% B-dot control law
m_cmd = -k * Bdot_f; % commanded magnetic dipole proportional to
    the negative of field's rate of change.

% dipole saturation
m_cmd = max(min(m_cmd, m_max), -m_max); % limits the commanded
    dipole vector so each component does not exceed m_max.
```

```

% map to coil currents and apply limits
I_cmd = m_cmd ./ Kmt; % Converts magnetic dipole command to coil
    current command using the torque coil gain Kmt.
I_cmd = max(min(I_cmd, I_max), -I_max); % limits current to +/-
    I_max.
m_cmd = I_cmd .* Kmt; % actual dipole moment

% update persistents for next iteration
B_prev = B_body;
Bdot_prev = Bdot_f;
end

```

Listing A.2 –: B dot Controller Simulation Script

```

% run_bdot_sim_1u_15hr.m
clear; clc; close all; clear bdot_controller;

disp(which('bdot_controller','-all'))

%% Units
deg = pi/180; % radians to degrees

%% Satellite properties
J = diag([0.0020, 0.0020, 0.0025]); % 1U inertia (
    kg\cdot m^2)
omega0 = [10; -8; 6]*deg; % initial
    angular velocity (rad/s)
q0 = [1; 0; 0; 0]; % initial
    quaternion

%% Magnetic field
Bmag = 50e-6; % 50  $\mu$ T
Bdir0 = [0.6; -0.2; 0.7]; Bdir0 = Bdir0/norm(Bdir0); % initial
    position

%% Orbit
T_orbit = 90*60; w_orbit = 2*pi/T_orbit; % 90 min rotation of
    B_I

%% Simulation time
Ts = 0.1;
Tfinal = 15*3600; % 15 hours
N = floor(Tfinal/Ts); % number of steps
t = (0:N-1).' * Ts;

```

```

%% B-dot controller parameters
alpha      = 0.95;                % smoothing
deadband   = 1.0e-8;              % T/s

% Magnetorquer properties
Kmt  = [0.15; 0.15; 0.15];        % A$\cdot$m^2/A
Imax = [0.50; 0.50; 0.50];        % A
mmax = Kmt .* Imax;                % A*m^2

% Sanity check
assert(isequal(size(Kmt),[3,1]), 'Kmt must be 3x1 column');
assert(isequal(size(Imax),[3,1]), 'Imax must be 3x1 column');
assert(isequal(size(mmax),[3,1]), 'mmax must be 3x1 column');
assert(all(Kmt>0) && all(Imax>0) && all(mmax>0), 'Kmt/Imax/mmax
    must be > 0');

% Scaled controller gain for 15-hour detumble
k = 1.1e4;

%% Logs - stores values of angular velocity, currents, dipole,
    body frame and estimated B dot.
log_step = 1; % can increase to 10 or 100 to save memory
omega = zeros(N,3);
I_cmd = zeros(N,3);
m_cmd = zeros(N,3);
B_body = zeros(N,3);
Bdot_est = zeros(N,3);

%% Initial state
q = q0;
w = omega0;
B_b_prev = NaN(3,1); % special value used to represent undefined
    numeric results or missing values

%% Controller unit test (optional)
clear bdot_controller % clears controller to reset its
    persistents
B1 = [50e-6; 0; 0];
B2 = [50e-6; 1e-7; 0];
[m1,I1] = bdot_controller(B1, Ts, k, alpha, deadband, mmax, Kmt,
    Imax);
[m2,I2] = bdot_controller(B2, Ts, k, alpha, deadband, mmax, Kmt,
    Imax);
fprintf('Unit test: ||m2||=%.3e, ||I2||=%.3e\n', norm(m2), norm(

```

```

I2)); % magnitudes

%% Simulation loop
for kstep = 1:N
    % earth magnetic field in inertial frame
    ang = w_orbit * t(kstep);
    Rz = [cos(ang) -sin(ang) 0; sin(ang) cos(ang) 0; 0 0 1]; %
        Rotate the initial field direction about z
    B_I = Bmag * (Rz * Bdir0);

    % transform to body frame
    R_bi = dcmFromQuat(q); % quaternions to body coordinates
        using DCM
    B_b = R_bi.' * B_I;
    B_body(kstep,:) = B_b.';

    % diagnostic derivative & smoothing
    if kstep == 1
        Bdot_est(kstep,:) = [0 0 0];
    else
        Bdot_est_raw = (B_b - B_b_prev) / Ts;
        Bdot_est(kstep,:) = alpha * Bdot_est_raw(:).'+ (1 -
            alpha) * Bdot_est(kstep-1,:);
    end
    B_b_prev = B_b;

    % B-dot controller
    [m_k, I_k] = bdot_controller(B_b, Ts, k, alpha, deadband,
        mmax, Kmt, Imax);

    % Satellite dynamics
    tau_b = cross(m_k, B_b);
    wdot = J \ (tau_b - cross(w, J*w));
    w = w + wdot * Ts;

    dq = quatFromOmega(w, Ts);
    q = quatNormalize(quatMul(q, dq));

    % Log
    omega(kstep,:) = w.';
    I_cmd(kstep,:) = I_k(:).';
    m_cmd(kstep,:) = m_k(:).';
end

%% Diagnostics

```

```

Bdot_norm = vecnorm(Bdot_est,2,2);
fprintf('Bdot_est | min: %.3e  median: %.3e  max: %.3e  (T/s)\n',
    ...
        min(Bdot_norm), median(Bdot_norm), max(Bdot_norm));
fprintf('Deadband: %.3e  ->  fraction below deadband: %.1f%%\n',
    ...
        deadband, 100*mean(Bdot_norm < deadband));

%% Plots
figure('Name','|omega| vs time');
grid on;
hold on;
plot(t, vecnorm(omega,2,2),'LineWidth',1.3);
xlabel('Time (s)');
ylabel('|{\omega}| (rad/s)');
title('Detumbling');

figure('Name','Coil currents');
grid on;
hold on;
plot(t, I_cmd(:,1)); plot(t, I_cmd(:,2)); plot(t, I_cmd(:,3));
xlabel('Time (s)');
ylabel('I (A)');
legend('Ix','Iy','Iz');

figure('Name','Commanded dipole');
grid on;
hold on;
plot(t, m_cmd(:,1)); plot(t, m_cmd(:,2)); plot(t, m_cmd(:,3));
xlabel('Time (s)');
ylabel('m (A*m^2)');
legend('mx','my','mz');

figure('Name','||Bdot_est|| vs time');
grid on;
plot(t, Bdot_norm); hold on; yline(deadband,'r--','deadband');
xlabel('Time (s)');
ylabel('||\dot{B}|| (T/s)', 'Interpreter','latex');

%% Helper functions
function R = dcmFromQuat(q) % quaternions to body frame using
    direction cosine matrix
    w=q(1); x=q(2); y=q(3); z=q(4);
    R = [1-2*(y*y+z*z), 2*(x*y - z*w), 2*(x*z + y*w);
        2*(x*y + z*w), 1-2*(x*x+z*z), 2*(y*z - x*w);

```

```

        2*(x*z - y*w), 2*(y*z + x*w), 1-2*(x*x+y*y)];
end

function dq = quatFromOmega(w, dt) % rotational quaternion for
    angular velocity
    th = norm(w)*dt;
    if th < 1e-12, dq=[1;0;0;0]; return; end
    ax = w/norm(w); half = 0.5*th;
    dq = [cos(half); sin(half)*ax(:)];
end

function q = quatMul(q1,q2) % quaternion product
    w1=q1(1); x1=q1(2); y1=q1(3); z1=q1(4);
    w2=q2(1); x2=q2(2); y2=q2(3); z2=q2(4);
    q = [ w1*w2 - x1*x2 - y1*y2 - z1*z2;
          w1*x2 + x1*w2 + y1*z2 - z1*y2;
          w1*y2 - x1*z2 + y1*w2 + z1*x2;
          w1*z2 + x1*y2 - y1*x2 + z1*w2 ];
end

function q = quatNormalize(q), q = q./norm(q); end % quaternion
    unit norm to avoid drift accumulation.

```

2 Reaction Wheel Attitude Controller

2.1 One Axis Perturbation Reaction Wheel Controller (RWP015)

Listing A.3 –: One Axis Perturbation Reaction Wheel Controller (RWP015)

```

clear; clc; close all;

%% Satellite Parameters (1U CubeSat)
% Satellite mass and dimensions
m = 1.24; % kg
dims = [0.1 0.1 0.108]; % meters (length, width, height)

a = dims(1);
b = dims(2);
c = dims(3);

% Inertia matrix for rectangular prism
I_sat = diag([
    (1/12)*m*(b^2 + c^2); % Ixx
    (1/12)*m*(a^2 + c^2); % Iyy
    (1/12)*m*(a^2 + b^2) % Izz

```

```

]);

%% Reaction Wheel Parameters (RWP050)
tau_max = 0.004;
h_max = 0.015;

%% Desired 15 Degree Slew (about X axis)
theta = deg2rad(15);
axis = [1;0;0];
q_des = axang2quat([axis' theta]);

%% Control Gains (Geometric Controller)
kR = 0.15;      % attitude gain
kW = 0.30;      % rate damping
kI = 0.02;      % integral gain

%% Initial State
q0 = [1;0;0;0];
w0 = [0;0;0];
int0 = [0;0;0];
h0 = [0;0;0];
x0 = [q0; w0; int0; h0];

%% Simulate
tspan = [0 40];
opts = odeset('RelTol',1e-7,'AbsTol',1e-9,'MaxStep',0.02);

[t, x] = ode45(@(t,x) dyn(t,x,q_des,I_sat,tau_max,h_max,kR,kW,kI)
, tspan, x0, opts);

%% Extract
q = x(:,1:4).'; for k=1:length(t), q(:,k)=q(:,k)/norm(q(:,k));
end
w = x(:,5:7).';
h = x(:,11:13).';

R = quat2rotm(q. ');
for k=1:length(t), eul(k,:) = rad2deg(rotm2eul(R(:,:,k))); end

%% Plots
figure; plot(t,eul); grid on;
xlabel('Time (s)'); ylabel('Euler Angles (deg)');
legend('Yaw','Pitch','Roll'); title('Attitude Response');

figure; plot(t,w. '); grid on;

```

```

xlabel('Time (s)'); ylabel('\omega (rad/s)');
legend('\omega_x', '\omega_y', '\omega_z'); title('Angular Velocity
');

figure; plot(t,h.'); grid on;
xlabel('Time (s)'); ylabel('Wheel Momentum (N*m*s)');
legend('h_x', 'h_y', 'h_z'); title('Reaction Wheel Momentum');

%% ===== Dynamics =====

function dx = dyn(~,x,q_des,I,tau_max,h_max,kR,kW,kI)
q = x(1:4); q = q/norm(q);
w = x(5:7);
int = x(8:10);
h = x(11:13);

R = quat2rotm(q.');
Rdes = quat2rotm(q_des.');
Re = Rdes'*R;

eR = 0.5*vee(Re - Re.');
eW = w;

tau_cmd = -kR*eR - kW*eW - kI*int;
tau = max(min(tau_cmd,tau_max),-tau_max);

intdot = eR + 0.2*(tau - tau_cmd);

hdot = tau;
for i=1:3
    if abs(h(i)) >= h_max && sign(h(i))==sign(hdot(i)), hdot(i)
        =0; end
end

wdot = I \ (-cross(w,I*w) + tau);

Omega = [0 -w(1) -w(2) -w(3);
          w(1) 0 w(3) -w(2);
          w(2) -w(3) 0 w(1);
          w(3) w(2) -w(1) 0];
qdot = 0.5*Omega*q;

dx = [qdot; wdot; intdot; hdot];
end

```

```

function v = vee(M)
v = [M(3,2); M(1,3); M(2,1)];
end

function q = axang2quat(axang)
u = axang(1:3)/norm(axang(1:3)); th=axang(4);
q=[cos(th/2); u(:)*sin(th/2)];
end

```

2.2 One Axis Perturbation Reaction Wheel Controller (RWP050)

Listing A.4 –: One Axis Perturbation Reaction Wheel Controller (RWP050)

```

clear; clc; close all;

%% Satellite Parameters (1U CubeSat)
% Satellite mass and dimensions
m = 1.24; % kg
dims = [0.1 0.1 0.108]; % meters (length, width, height)

a = dims(1);
b = dims(2);
c = dims(3);

% Inertia matrix for rectangular prism
I_sat = diag([
    (1/12)*m*(b^2 + c^2); % Ixx
    (1/12)*m*(a^2 + c^2); % Iyy
    (1/12)*m*(a^2 + b^2) % Izz
]);

%% Reaction Wheel Parameters (RWP050)
tau_max = 0.007;
h_max = 0.05;

%% Desired 15 Degree Slew (about X axis)
theta = deg2rad(15);
axis = [1;0;0];
q_des = axang2quat([axis' theta]);

%% Control Gains (Geometric Controller)
kR = 0.015; % attitude gain
kW = 0.30; % rate damping
kI = 0.02; % integral gain

```

```

%% Initial State
q0 = [1;0;0;0];
w0 = [0;0;0];
int0 = [0;0;0];
h0 = [0;0;0];
x0 = [q0; w0; int0; h0];

%% Simulate
tspan = [0 240];
opts = odeset('RelTol',1e-7,'AbsTol',1e-9,'MaxStep',0.02);

[t, x] = ode45(@(t,x) dyn(t,x,q_des,I_sat,tau_max,h_max,kR,kW,kI)
, tspan, x0, opts);

%% Extract
q = x(:,1:4).'; for k=1:length(t), q(:,k)=q(:,k)/norm(q(:,k));
end
w = x(:,5:7).';
h = x(:,11:13).';

R = quat2rotm(q. ');
for k=1:length(t), eul(k,:) = rad2deg(rotm2eul(R(:, :, k))); end

%% Plots
figure; plot(t,eul); grid on;
xlabel('Time (s)'); ylabel('Euler Angles (deg)');
legend('Yaw','Pitch','Roll'); title('Attitude Response');

figure; plot(t,w. '); grid on;
xlabel('Time (s)'); ylabel('\omega (rad/s)');
legend('\omega_x','\omega_y','\omega_z'); title('Angular Velocity
');

figure; plot(t,h. '); grid on;
xlabel('Time (s)'); ylabel('Wheel Momentum (N*m*s)');
legend('h_x','h_y','h_z'); title('Reaction Wheel Momentum');

%% ===== Dynamics =====

function dx = dyn(~,x,q_des,I,tau_max,h_max,kR,kW,kI)
q = x(1:4); q = q/norm(q);
w = x(5:7);
int = x(8:10);
h = x(11:13);

```

```

R = quat2rotm(q. ');
Rdes = quat2rotm(q_des. ');
Re = Rdes'*R;

eR = 0.5*vee(Re - Re. ');
eW = w;

tau_cmd = -kR*eR - kW*eW - kI*int;
tau = max(min(tau_cmd,tau_max),-tau_max);

intdot = eR + 0.2*(tau - tau_cmd);

hdot = tau;
for i=1:3
    if abs(h(i)) >= h_max && sign(h(i))==sign(hdot(i)), hdot(i)
        =0; end
end

wdot = I \ (-cross(w,I*w) + tau);

Omega = [0 -w(1) -w(2) -w(3);
          w(1) 0 w(3) -w(2);
          w(2) -w(3) 0 w(1);
          w(3) w(2) -w(1) 0];
qdot = 0.5*Omega*q;

dx = [qdot; wdot; intdot; hdot];
end

function v = vee(M)
v = [M(3,2); M(1,3); M(2,1)];
end

function q = axang2quat(axang)
u = axang(1:3)/norm(axang(1:3)); th=axang(4);
q=[cos(th/2); u(:)*sin(th/2)];
end

```

2.3 LEO Simulation Reaction Wheel Controller

Listing A.5 –: Three Axis Perturbation and 75 Degree Angle Slew Reaction Wheel Controller (RWP050)

```

%% initial angular velocity of 0.8 rad/s correcting for a 75
   degree slew

```

```

% Reaction-wheel attitude control with LEO environment and
  perturbations

clear; clc; close all;

%% Satellite Parameters (1U CubeSat)
m = 1.24; % kg
dims = [0.1 0.1 0.108]; % m (X, Y, Z)
a = dims(1); b = dims(2); c = dims(3);

I_sat = diag([
    (1/12)*m*(b^2 + c^2); % Ixx
    (1/12)*m*(a^2 + c^2); % Iyy
    (1/12)*m*(a^2 + b^2) % Izz
]);

%% Reaction Wheel Parameters (RWP050-like)
tau_max = 0.007; % N*m (per-axis peak torque)
h_max = 0.05; % N*m*s (per-axis momentum storage)

%% LEO Parameters
params.alt_km = 500; % circular LEO estimation
params.incl_deg = 51.6; % degrees
params.Cd = 2.2; % drag coefficient
params.Cr = 1.3; % SRP coefficient
params.A_ref = 0.01; % m^2 (ref area)
params.rcp_aero = [0; 0; 0.01]; % m, CP offset for aero
params.rcp_sun = [0.01; 0; 0]; % m, CP offset for SRP
params.m_res = [0.03; -0.01; 0.02]; % A*m^2 residual dipole (
    for torque via B)
params.B0 = 3.12e-5; % Tesla (equatorial surface
    field)
params.Re = 6378.137e3; % m
params.mu = 3.986004418e14; % m^3/s^2
params.Psun = 4.5e-6; % N/m^2
params.rho0 = 5e-12; % kg/m^3 @ 500 km (very rough)
params.H = 60e3; % m, scale height

%% Desired Slew: 75 deg about [1 1 1] from identity
theta = deg2rad(75);
slew_axis = [1;1;1];
q_des = axang2quat([slew_axis.' theta]);

%% Controller Gains (geometric: -kR*eR - kW*w - kI*int)
kR = 0.02; % Proportional Gain

```

```

kW = 0.4;           % Derivative Gain
kI = 0.004;         % Integral Gain

%% Initial State
q0 = [1;0;0;0];      % identity quaternion
w0 = [0.8;0.8;0.8];  % initial detumbling rad/s on all
    axes
int0 = [0;0;0];
h0 = [0;0;0];
x0 = [q0; w0; int0; h0];

%% Simulation
tspan = [0 240];      % time span for
    test
opts = odeset('RelTol',1e-6,'AbsTol',1e-8); % quick & stable
    steps

tic;
[t, x] = ode45(@(t,x) dyn_rw_only(t,x,q_des,I_sat,tau_max,h_max,
    kR,kW,kI,params), tspan, x0, opts);
fprintf('ode45 wall time = %.2fs, N=%d, t_end=%.1f s\n', toc,
    numel(t), t(end));

%% Extract & Normalize
q = x(:,1:4).'; % 4xN
for k = 1:numel(t), q(:,k) = q(:,k)/norm(q(:,k)); end
w = x(:,5:7).'; % 3xN
h = x(:,11:13).';

%% Compute Euler (ZYX) and Angle-to-go
R = quat2rotm(q. ');
eul = zeros(numel(t),3);
phi = zeros(numel(t),1); % angle-to-go (rad)
for k = 1:numel(t)
    eul(k,:) = rad2deg(rotm2eul(R(:,:,k))); % [yaw pitch roll]
    qk = q(:,k).';
    qe_k = quatmultiply(q_des.', [qk(1) -qk(2:4)]).';
    phi(k) = 2*atan2(norm(qe_k(2:4)), abs(qe_k(1)));
end

%% Dynamics (RW controller only, with disturbances)
function dx = dyn_rw_only(t,x,q_des,I,tau_max,h_max,kR,kW,kI,
    params)
    % Unpack state
    q = x(1:4); q = q/norm(q);

```

```

w    = x(5:7);
int  = x(8:10);
h    = x(11:13);

% Attitude
R     = quat2rotm(q. '); % quaternions to inertial
Rdes  = quat2rotm(q_des. '); % quaternions to inertial

% Environment/disturbances (in body frame)
[r_eci, v_eci, sun_hat, B_eci, rho] = env_eci(t, params);
tau_dist = tau_disturb(R, I, r_eci, v_eci, sun_hat, B_eci,
    rho, params);

% Geometric error (Lee-style)
Re = Rdes' * R;
eR = 0.5 * vee(Re - Re. ');
eW = w;

% RW torque command
tau_cmd = -kR*eR - kW*eW - kI*int;

% Smooth torque clamp
tau_rw = tau_max * tanh(tau_cmd / tau_max);

% Momentum storage gate: block torque pushing deeper into
    saturation
for i = 1:3
    if abs(h(i)) >= h_max && sign(tau_rw(i)) == sign(h(i))
        tau_rw(i) = 0;
    end
end

% Total applied torque = RW + disturbances (no magnetorquer
    in this demo)
tau = tau_rw + tau_dist;

% Anti-windup (back-calculation on RW channel)
intdot = eR + 0.2*(tau_rw - tau_cmd);

% Wheel Momentum
hdot = tau_rw;

% Rigid-Body Dynamics
wdot = I \ (-cross(w, I*w) + tau);

```

```

% Quaternion Kinematics
Omega = [ 0      -w(1) -w(2) -w(3);
          w(1)   0      w(3) -w(2);
          w(2) -w(3)   0      w(1);
          w(3)  w(2) -w(1)   0   ];
qdot = 0.5 * Omega * q;

dx = [qdot; wdot; intdot; hdot];
end

%% Helpers
function [r_eci, v_eci, sun_hat, B_eci, rho] = env_eci(t, p)
% Circular Keplerian ECI state with simple inclination
a = (p.Re + p.alt_km*1e3);
n = sqrt(p.mu / a^3);
M = n*t;
ci = cosd(p.incl_deg); si = sind(p.incl_deg);

r_orb = a * [cos(M); sin(M); 0];
v_orb = a*n * [-sin(M); cos(M); 0];
Rinc = [1 0 0; 0 ci -si; 0 si ci];

r_eci = Rinc * r_orb;
v_eci = Rinc * v_orb;

% Crude sun direction (fixed in ECI)
sun_hat = [1;0;0];

% Simple dipole B-field aligned with +Z_ECI (non-zero, varies
    with r)
R = norm(r_eci);
mu0 = p.B0 * (p.Re^3);
ez = [0;0;1];
B_eci = (mu0/R^3) * (3*ez*(ez.'*r_eci)/R - r_eci/R);

% Exponential atmosphere
rho = p.rho0 * exp(-(R - (p.Re + p.alt_km*1e3))/p.H);
end

function tau = tau_disturb(R, I, r_eci, v_eci, sun_hat, B_eci,
rho, p)
% Body-frame unit vectors
rhat_b = R.' * (r_eci / norm(r_eci));
vhat_b = R.' * (v_eci / norm(v_eci));
shat_b = R.' * sun_hat;

```

```

% Gravity-gradient
taugrav = 3*p.mu/(norm(r_eci)^3) * cross(rhat_b, I*rhat_b);

% Aerodynamic
qdyn      = 0.5 * rho * norm(v_eci)^2;
Faero_b   = qdyn * p.Cd * p.A_ref * vhat_b;
tauA      = cross(p.rcp_aero, Faero_b);

% SRP (no eclipse modeling in this simple demo)
Fsun_b    = p.Psun * p.Cr * p.A_ref * shat_b;
tauS      = cross(p.rcp_sun, Fsun_b);

% Residual magnetic dipole torque
B_b       = R.' * B_eci;
tauM       = cross(p.m_res, B_b);

tau = taugrav + tauA + tauS + tauM;
end

function v = vee(M)
    v = [M(3,2); M(1,3); M(2,1)];
end

function q = axang2quat(axang)
    u = axang(1:3)/norm(axang(1:3)); th = axang(4);
    q = [cos(th/2); u(:)*sin(th/2)];
end

function out = quatmultiply(q,r)
    % q,r as [w x y z]
    w1=q(1); x1=q(2); y1=q(3); z1=q(4);
    w2=r(1); x2=r(2); y2=r(3); z2=r(4);
    out=[ w1*w2 - x1*x2 - y1*y2 - z1*z2,...
          w1*x2 + x1*w2 + y1*z2 - z1*y2,...
          w1*y2 - x1*z2 + y1*w2 + z1*x2,...
          w1*z2 + x1*y2 - y1*x2 + z1*w2 ];
end

%% Figures

set(0, 'DefaultFigureColor', 'k', ...
      'DefaultAxesColor', 'k', ...
      'DefaultAxesXColor', 'w', ...
      'DefaultAxesYColor', 'w', ...

```

```

        'DefaultAxesZColor', 'w', ...
        'DefaultAxesGridColor', 'w', ...
        'DefaultTextColor', 'w', ...
        'DefaultLegendTextColor', 'w', ...
        'DefaultLegendColor', 'k', ...
        'DefaultLegendEdgeColor', 'w');

% Attitude Response in Euler Angles
figure;
plot(t,eul); grid on;
xlabel('Time (s)');
ylabel('Euler Angles (deg)');
legend('Yaw','Pitch','Roll','Location','best');
title('Attitude Response');

% Angle to go
figure;
plot(t, rad2deg(phi),'LineWidth',1.2); grid on;
xlabel('Time (s)');
ylabel('Angle-to-go (deg)');
title('Rotation Error Magnitude');

% Angular Velocity of the spacecraft
figure;
plot(t,w.'); grid on;
xlabel('Time (s)');
ylabel('\omega (rad/s)');
legend('\omega_x','\omega_y','\omega_z','Location','best');
title('Body Rates');

% Momentum of the reaction wheels
figure;
plot(t,h.'); grid on;
xlabel('Time (s)');
ylabel('Wheel Momentum (N*m*s)');
legend('h_x','h_y','h_z','Location','best');
title('Reaction Wheel Momentum');

```