Design and Analysis of a CubeSat

A Major Qualifying Project

Submitted By:

Robaire Galliath

Oliver Hasson

Andrew Montero

Chistopher Renfro

David Resmini

Submitted To:

Professor Michael A. Demetriou

Submitted to the Faculty of the Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Aerospace Engineering.

August 2019 - March 2020

Contents

List of Tables

List of Figures

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras ac condimentum eros. Sed quis est eu ante ultrices rhoncus. Praesent a odio eget dolor hendrerit tincidunt. Proin ullamcorper lacus odio, nec dignissim enim hendrerit vitae. Quisque quis risus tellus. Sed sit amet lectus non massa accumsan malesuada. Aenean purus erat, fringilla id ante et, finibus vehicula ex. Sed a ex mi. Duis nec libero ex. Donec aliquam eu tortor quis sodales. Nulla vel urna vitae tellus accumsan bibendum ut quis purus. Sed at ipsum nibh. Fusce erat nulla, aliquet pretium tempor ut, facilisis non sem.

1 Introduction

The goal of this project is to design and conduct analysis of a CubeSat on an extreme low Earth orbit (eLEO) mission. The satellite will be carrying a mass spectrometer to conduct atmospheric observation. Following deployment from the ISS the satellite will enter a 250 - 600 kilometer orbit and maintain this orbit as long as possible. The overall project is composed of three separate MQP teams, each responsible for a different aspect of the satellite design and analysis. This portion of the overall project focuses on the attitude determination and control, orbital determination and control, and command subsystems of the satellite.

1.1 CubeSat Background

Cube satellites (CubeSats) are miniature satellites used for space research and technology development. They are a particular class of nano-satellite that was developed by the California Polytechnic State University and Stanford University in 1999 in order to promote the design, manufacture, and testing of satellite technology in low Earth orbit [1]. CubeSats are comprised of multiple 10 cm by 10 cm by 10 cm units, referred to as 'U's. Layouts can vary greatly, but the most common form factors are 1U and 3U [1]. In recent years, larger CubeSats have been developed to increase available space for mission payloads. Typically, CubeSats are deployed by a launch mechanism attached to the upper stage of a launch vehicle and offer an easy way to deploy CubeSats into Earth orbit. The CubeSat concept began as a plan to provide scientific and military laboratories a tool to grow their operations in space [2]. In recent years, the CubeSat has become a unique tool in the scientific community. NASA's CubeSat Launch Initiative (CSLI) provides opportunities to launch CubeSats aboard larger launch vehicles as secondary payloads [3]. CubeSats are currently in an era of rapid growth; it is estimated that the global CubeSat market was valued at $152 million in 2018, and is projected to rise to nearly $375 million in coming years [4].

1.2 Social, Economic, and Educational Considerations

The continued expansion and development of CubeSat technologies has yielded a variety of positive social, economic, and educational effects. The space industry has been heavily

impacted by the surge of CubeSat and related space technology start-ups in the last five years. From 2000 to 2014 space start-ups have received a combined sum of $1.1 billion in venture capital investments [4]. The number has only increased in recent years, with a total investment of $3.9 billion in 2017 alone [4]. These companies provide satellite components, development, or launch and integration services. As of 2018, 51% of CubeSats are developed by the private sector, demonstrating that CubeSats are not limited to large research institutions or governments [4]. The wide availability of CubeSat components and hardware has significantly reduced the cost and complexity of creating a flight capable system. Many universities and several high schools have launched CubeSats thanks to these advantages [2]. Low development and launch cost also provides an opportunity for cost effective flight testing of new and experimental technologies. In a notable example, the *Mars InSight* mission carried two CubeSats as a secondary payload in order to test new miniaturized deep space communication equipment [5]. The expansion and promotion of CubeSats has also garnered interest from students at all levels in the space sector. Space agencies such as NASA and ESA are able to use CubeSats to inspire students to pursue an education and career in STEM fields.

Unfortunately, the rapid expansion of CubeSats has had negative effects as well. The large increase in satellites in LEO complicates flight paths and increases the risk of unintended collisions. Unlike larger satellites, CubeSats are generally designed without extensive maneuvering capabilities making it difficult to avoid collisions. Their small size also makes them difficult to track, further increasing the risk posed to other satellites. ESA has already experienced a collision due to CubeSat debris. The Sentinel-1A satellite was struck by debris from a CubeSat. A study conducted by NASA using the LEO to GEO Environment Debris (LEGEND) model estimated significant increases in the number of collisions caused by CubeSats in LEO. The increase is space debris due to satellite collisions would make operations in LEO more difficult. As the accessibility of CubeSats increases a greater effort will be necessary to ensure their operation and control is safe and secure.

1.3 Project Constraints

There are four major constraints on this project: the orbit profile, the primary propulsion system, the scientific payload, and the satellite form factor. As mentioned in the introduction, the satellite must enter and maintain a 250 - 600 kilometer orbit as long as possible. This is to allow the scientific payload, the miniature Ion Neutral Mass Spectrometer (mini-INMS), to conduct atmospheric analysis in low Earth orbit. The mini-INMS is pictured in figure 1. A Busek electro-spray thruster has already been selected as the primary propulsion system for this mission profile and cannot be changed. It is expected that the satellite adhere to the CubeSat form factor, although the final size of the satellite is flexible.

The mini-INMS was developed at NASA's Goddard Space Flight Center. It is capable of detecting ions of densities between $10^3 - 10^8 \text{cm}^3$ and neutrals of densities between $10^4 - 10^9 \text{cm}^3$, with low energies between 0.1eV and 20eV. The apertures of the mini-INMS must be RAM facing to conduct measurements. The instrument occupies nearly 1.5U of space and has a mass of 560 grams [6].

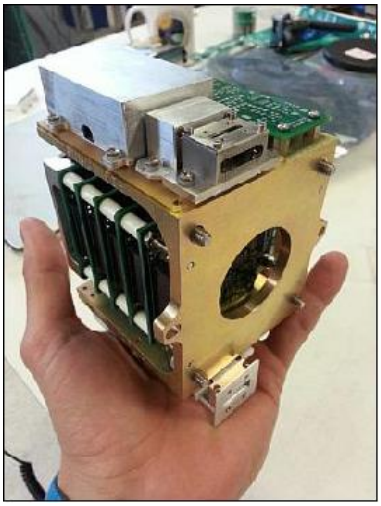

Figure 1: Miniature Ion Neutral Mass Spectrometer

2 Background

2.1 Attitude Determination and Control

The purpose of the attitude determination and control subsystem (ADCS) is to properly position and orient the spacecraft to meet the needs of the mission. The ADCS is responsible for three distinct operations throughout the mission: detumbling, initial attitude determination, and attitude maintenance. Successful operation in all three phases is vital to the overall success of the mission. This is accomplished by combining a variety of sensors and actuators in a closed-loop control system.

2.1.1 Requirements

Each phase of the mission has different requirements. In order to successfully detumble the satellite must correct for the angular spin imparted during deployment. It is expected that such an angular velocity would not have a magnitude greater than ten degrees per second about any axis. Once the satellite has reduced its angular velocity about two of its axis it is considered detumbled. It must then determine its orientation with respect toe the Earth inertial frame. From this point onwards the satellite must maintain its attitude within plus or minus five degrees. As the orbit dips lower into the atmosphere the effects of drag become significant. Should the angular orientation deviate further than this limit the torques induced by atmospheric drag risk overcoming the strength of the on board actuators, causing the spacecraft to enter an uncontrollable spin. The torque exerted on the spacecraft can be described as a function of atmospheric density ρ, cross sectional area A, velocity V_{rel}, drag

coefficient C_D, center of pressure c_p, and center of gravity c_g, as shown in equation 1.

$$\tau_d = \frac{1}{2}\rho A C_D V_{rel}^2 (c_p - c_g) \tag{1}$$

Velocity is a function of the orbital velocity and the pointing angle. As the velocity increases, it is evident that the torque experienced by the spacecraft increases as well. This relationship is shown in figure 2.

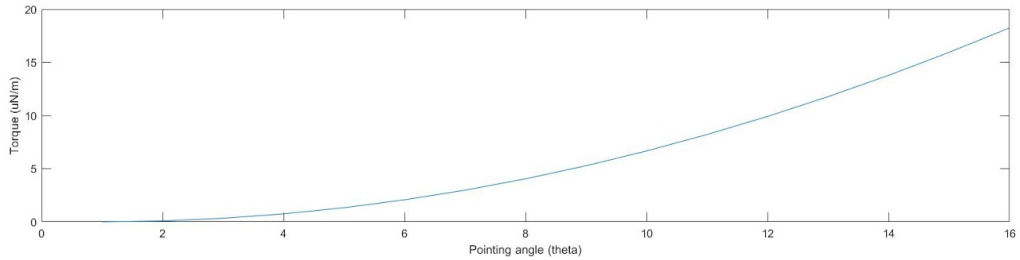

Figure 2: Pointing Angle vs. Torque

2.1.2 Actuator Selection

Actuators are needed to effect change on the satellites orientation in space. Two different types of actuators are used to accomplish this, magnetorquers and reaction wheels. Magnetorquers create a torque using an external magnetic field and a magnetic dipole moment. A unique property of the magnetorquer is that is has no moving parts, unlike a reaction wheel that relies on moving parts. Due to this unique characteristic, magnetorquers are less prone to malfunction, and along with being cheap, lightweight and simple, are great for CubeSats. Magnetorquers are commonly made from a metal rod wrapped in a copper wire. When a current is run through the coil a magnetic moment is created. The interaction of this magnetic moment, ν, with the Earth's magnetic field, B, induces a torque, τ, on the space craft as shown in equation 2.

$$\tau = \nu \times B \tag{2}$$

The magnetorquer chosen for this task is the NCTR-M002 Magnetorquer Rod, which is manufactured by *New Space*. The NCTR-M002 uses a magnetic alloy rod that produces an amplification effect over an air cored magnetorquer. The NCTR-M002 consumes 200mW of power from a 5V power supply while producing a magnetic moment greater than 0.2 Am^2. The residual moment left over from the magnetorquer rod is negligible at less than 0.001 Am^2. Specific performance characteristics are listed in table 1.

Figure 3: NCTR-M002 Magnetorquer Rod

Table 1: Magnetorquer Parameters

Parameter	Value
Magnetic Moment	0.2 Am^2
Linearity	$< \pm 5\%$
Residual Moment	< 0.005 Am^2
Dimensions	$70mm \times \varnothing 10mm$
Mass	< 30 g
Power	200 mW
Operating Temperature	$-20°C$ to $60°C$
Vibration	14 g(rms)

In addition to the magnetorquers reaction wheels will be included onboard the CubeSat. Reaction wheels, also refereed to as momentum wheels, are internal components that store rotational momentum, providing satellites with three axis control. By adjusting the momentum of a weighted wheel the reaction wheel induces an equal and opposite torque on the space craft, as the total momentum of the space craft must remain constant. Reaction wheels provide very precise pointing control without the fuel requirements of conventional attitude control thrusters. The reaction wheel chosen for this mission is the RWP050, pictured in figure 4. This wheel can create a maximum torque of 0.007 Nm and create a total moment of 0.050 Nms while consuming less that 1W at full power. Operating parameters are available in table 2.

Figure 4: RWP050 Reaction Wheels

Parameter	Value
Dimensions	$58 \times 58 \times 25$ mm
Voltage	10 - 14 V
Power	<1 W
Operating Temperature	$-20°C$ to $60°C$

2.1.3 Sensor Selection

The CubeSat uses a collection of sensors to accurately determine its attitude and position relative to the Earth. A Global Positioning System (GPS) receiver allows the satellite to determine its position above the Earth. This information is then used to predict the expected magnetic field and sun vectors using well proven models. The NGPS-01-422, manufactured by *New Space* is a great choice for the on-board GPS. It is relatively low powered and provides sufficiently accurate readings. The GPS module includes an external antenna, the NANT-PTCL1. Specifications for the GPS and antenna are available in tables 3 and 4 respectively.

Figure 5: GPS Module and Antenna

Table 3: GPS Parameters

Parameter	Value
Mass	<500 g
Power Consumption	1.5 W
Position	< 10 m
Velocity	$< 25 \frac{cm}{s}$
Operating Temperature	$-10°C$ to $50°C$
Dimensions	$155 \times 76 \times 34$ mm

Table 4: GPS Receiver Parameters

Parameter	Value
Mass	<80 g
Power Consumption	80 mW
Frequency	1575.42 MHz
Bandwidth	20 MHz
Operating Temperature	$-25°C$ to $55°C$
Dimensions	$54 \times 54 \times 14.1$ mm
Active Gain (RHC)	$>$ 16dBi
Noise Figure	$<$ 2 dB

A gyroscope is also included to measure the angular rotation rates of the spacecraft. The gyroscope chosen is the ADXRS453 manufactured by Analog Devices. This gyroscope consumes very little power and is very light weight.

Figure 6: ADXRS453

Table 5: Gyroscope Parameters

Parameter	Value
Mass	<56.7 g
Power Consumption	18.9 mW
Operating Temperature	$-40°C$ to $105°C$
Dimensions	$33 \times 33 \times 3$ mm

The LSM303AGR is a combination triple axis accelerometer and magnetometer. It is small, lightweight, low cost, and power efficient compared to other accelerometer and magnetometer options. Magnetometer readings allow the spacecraft to estimate its orientation based on the expected magnetic field given its position in orbit.

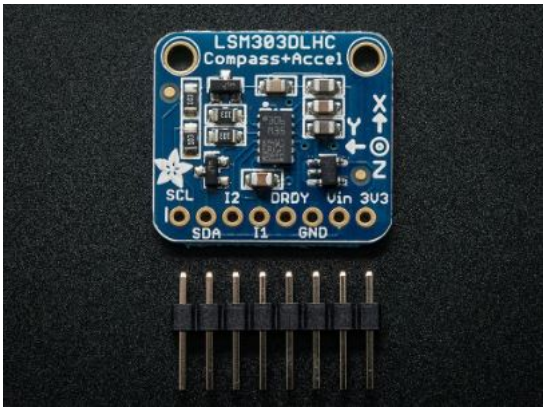

Figure 7: LSM303AGR Triple Axis Accelerometer and Magnetometer

Table 6: Accelerometer and Magnetometer Parameters

Parameter	Value
Mass	10 mg
Power Consumption	5 mW
Operating Temperature	$-40°C$ to $85°C$
Dimensions	$2 \times 2 \times 1$ mm
Linear Acceleration	$\pm 0.01\%$
Magnetic Sensitivity	$\pm 1\%$

Sun sensors are used to determine the position of the sun relative to the space craft. This provides a high accuracy method for determining the satellite's attitude. Two-axis, fine sun sensors measure the incident angle of the sun in two orthogonal axes. A total of five Nano-SSOC-A60 sun sensors are mounted to the satellite. Five sensors are used to provide near total coverage such that the sun is always in view of at least one sensor.

Figure 8: Nano-SSOC-A60 Sun Sensor

Table 7: Sun Sensor Parameters

Parameter	Value
Mass	4 g
Power Consumption	10 mW

2.1.4 Detumbling

All CubeSat deployment systems will induce some angular rotation to the CubeSat as it is deployed. As such, the CubeSat will immediately begin tumbling at the beginning of its mission, even before the satellite can activate any sort of attitude control system. Generally, the angular velocity induced by the deployer is unknown and would interfere with the operation of the space craft. It becomes necessary to detumble the satellite before it can begin its main mission. One of the most common methods used to detumble satellites is the B-Dot control algorithm. The B-Dot controller detumbles a spacecraft by commanding a magnetic dipole moment based on the rate of change of the Earth's magnetic field, hence the term 'B-Dot'. The torque produced by the interaction of the magnetorquers' magnetic field with the Earth's magnetic field, as described in equation 2. acts counter to the angular velocity of the spacecraft, slowly reducing its magnitude. If the spacecraft's angular velocity, $\vec{\omega}$, is known, as provided by gyroscopes, the command torque, $\vec{m}$, can be expressed in terms of the gain, k, and magnetic field as shown in 4. The normal of the Earth's magnetic field is expressed as $\vec{b}$ in 3.

$$\vec{b} = \frac{\vec{B}}{\|\vec{B}\|} \tag{3}$$

$$\vec{m} = \frac{k}{\|\vec{B}\|}\vec{\omega} \times \vec{b} \tag{4}$$

It is assumed that the change in the Earth's magnetic field due the change in orbital position over time occurs much more slowly than the change of the magnetic field in the spacecraft body frame due to the the tumbling motion. The B-Dot control law is expressed in 5.

$$\vec{L} = \vec{m} \times \vec{B} \tag{5}$$

To make it simpler to prove the stability of the control law, 5 can be rewritten as 6.

$$\vec{L} = -k(I_3 - \vec{b}\vec{b}^T)\vec{\omega} \tag{6}$$

To prove the sability of the control law, it has to be shown to reduce a positive Lyapunov function asymptotically to zero by making the derivative of the function less than or equal to zero for all cases. For this application the Lyapunov function in 7 is used; where J is the spacecraft moment of inertia matrix. This function is analogous to the rotational kinetic energy. This is useful as the goal of the detumbling control law is to reduce the total angular velocity, and thus reduce the rotational kinetic energy.

$$V = \frac{1}{2}\vec{\omega}^T J\vec{\omega} \tag{7}$$

Applying the B-Dot control law written in 7, its derivative is calculated as 8.

$$\dot{V} = -k\vec{\omega}^T(I_3 - \vec{b}\vec{b}^T)\vec{\omega} \tag{8}$$

The eigen values of $(I_3 - \vec{b}\vec{b}^T)$ are always 0, 1, and 1 implying that $(I_3 - \vec{b}\vec{b}^T)$ is positive semidefinite. Therefore, $\dot{V}$ is always less than or equal to zero. The only case where $\dot{V}$ is equal to zero and global asymptotic stability cannot be obtained is when $\vec{\omega}$ is parallel to $\vec{b}$, in which case the spacecraft is already not tumbling. This this case, the spacecraft would only be spinning around one axis and the mission can move into the attitude determination phase. If it was necessary to use 4 in a bang-bang application, it can be rewritten into 9, where N is the number of magnetorquers on the spacecraft, m_i^{max} is the maximum moment the $i-th$ magnetorquer can delivery, and u_i is the direction of the magnetic moment for the ith magnetorquer. This implementation is less computationally expensive than the other implementation, but less efficient in terms of power consumption.

$$\vec{L} = \sum_{i=1}^{N} -m_i^{max} sign(\vec{u_i} \cdot \dot{\vec{B}}) \tag{9}$$

2.1.5 Attitude Determination

There are two main types of methods used to determine the attitude of a spacecraft: recursive and deterministic. Recursive methods work by estimating the current attitude from a previous attitude measurement. To estimate the current attitude quaternion, q_n, the previous attitude quaternion, q_{n-1}, is integrated. An alternative to recursive methods are deterministic methods. Deterministic algorithms estimate the current attitude quaternion by comparing a set of sensor observations with their expected values. Previous MQP teams used the deterministic TRIAD method to determine attitude. The TRIAD algorithm is well understood, reliable, and computationally efficient, making it an ideal choice for simple attitude determination systems.

Harold Black's 1964 TRIAD algorithm was the first of its kind. It utilizes the sun and Earth magnetic field vectors to determine a spacecrafts orientation. To implement this algorithm four pieces of information must be known: the sun and magnetic field vectors in the spacecraft body-fixed frame, and the sun and magnetic field vectors in the Earth-fixed reference frame [7]. The sun and magnetic field vectors in the body-fixed frame are easily determined using the onboard magnetometer and sun sensors. Values for the sun and magnetic field vectors in the Earth-fixed frame can be determined using the position of the satellite in its orbit, provided by the GPS, and mathematical models. The World Magnetic Model (WMM) provides the Earth's magnetic field vector at any point in LEO given a satellite's longitude, latitude, and altitude [8]. The sun vector in the Earth-fixed frame can be calculated using the time of the year with a precision of 0.01° (36") for years between 1950 to 2050. The number of days, n, since Greenwich Noon, Terrestrial Time, on the 1st of January, 2000 can be computed given

the current Julian Date, JD, using 10.

$$n = JD - 2451545.0 \tag{10}$$

The mean longitude of the sun, L, and the mean anomaly of the sun, g can be determined by applying n to 11 and 12 respectively.

$$L = 280.46^\circ + 0.9856474^\circ n \tag{11}$$

$$g = 357.528^\circ + 0.9856003^\circ n \tag{12}$$

From these values the ecliptic longitude of the sun, λ, and the distance to the sun in astronomical units, R, can be calculated using 13 and 14.

$$\lambda = L + 1.915^\circ \sin(g) + 0.020^\circ \sin(2g) \tag{13}$$

$$R = 1.00014 - 0.01671 \cos(g) - 0.00014 \cos(2g) \tag{14}$$

It then becomes possible to determine the position of the sun in equatorial coordinates using 15 - 18.

$$\epsilon = 23.439^\circ - 0.0000004^\circ n \tag{15}$$

$$X = Rcos(\epsilon)cos(\lambda) \tag{16}$$

$$Y = Rcos(\epsilon)sin(\lambda) \tag{17}$$

$$Z = Rsin(\epsilon) \tag{18}$$

The TRIAD method allows for coordinates in the body fixed reference frame to be rotated into the Earth-fixed inertial reference frame. However, because vectors must be normalized for this method, it is possible that there could be a decent amount of sensor noise. Later research, specifically by Grace Wahba, poses a potential way of minimizing input sensor error, thus making the TRIAD method more accurate. Solutions to Wahba's problem include Davenport's q-method and QUEST. These methods provide and optimum quaternion for expressing the spacecrafts attitude.

The TRIAD method begins by defining two linearly independent body fixed vectors, b_1 and b_2, as well as their corresponding reference frame vectors, r_1 and r_2. The attitude matrix, A, is defined as a rotation from the body-fixed frame into the Earth-fixed frame in 19.

$$Ar_i = b_i \tag{19}$$

Ideally, A would be the same for both $i = 1$ and $i = 2$, however due to noise in each sensor input this may not be strictly true. For the TRIAD method to work, it is assumed that the transformation for $i = 1$ is significantly more accurate than its counter part. Two sets of orthonormal right-handed triads of vectors are defined, one for the reference frame, M_{ref}, and one for the body frame, M_{obs}.

$$M_{obs} = \langle r_1, \frac{r_1 \times r_2}{|r_1 \times r_2|}, r_1 \times \frac{r_1 \times r_2}{|r_1 \times r_2|} \rangle \tag{20}$$

$$M_{ref} = \langle b_1, \frac{b_1 \times b_2}{|b_1 \times b_2|}, b_1 \times \frac{b_1 \times b_2}{|b_1 \times b_2|} \rangle \tag{21}$$

A direct cosine matrix can be obtained by substituting M_{obs} and M_{ref} for b and r in 19. Thus, 19 can be rewritten as 22.

$$AM_{obs} = M_{ref} \tag{22}$$

Expanding the relation for each individual vector and simplifying the expression for the direct cosine matrix becomes 23.

$$A_{\text{DCM}} = b_1 r_1^T + (b_1 \times \frac{b_1 \times b_2}{|b_1 \times b_2|})(r_1 \times \frac{r_1 \times r_2}{|r_1 \times r_2|})^T + \frac{b_1 \times b_2}{|b_1 \times b_2|} \frac{r_1 \times r_2}{|r_1 \times r_2|}^T \tag{23}$$

This provides a mechanism for coordinates to be rotated from the body-fixed frame to the Earth-fixed and back. It is important to note however, that for cases where either the reference vectors or observed vectors are parallel or anti-parallel equation 23 becomes undefined.

2.1.5.1 Wahba's Problem

Mathematician Grace Wahba attempted to describe issues associated with using direction cosine matrices in attitude estimation and provide a way to build upon the TRIAD method. Improvements include, adding a way to weigh sensor measurements and allowing for more than 2 sets of measurements to be used. To understand how to improve upon the TRIAD method, one must understand the problem Wahba proposed. Wahba's problem serves to find an orthogonal matrix with a positive determinant that minimizes the loss function in 24

$$L(A) = \frac{1}{2} \sum_{i=1}^{N} a_i ||b_i - Ar_i||^2 \tag{24}$$

This essential finds the rotation matrix A that brings the first set of unit vectors $(b_1, b_2, \ldots, b_n)$ into the best least squares coincidence with the second set of vectors $(r_1, r_2, \ldots, r_n)$. Where b_i is a set of n unit vectors in the spacecraft body frame, r_i is the corresponding set of vectors in the reference frame, and a_i represents the non-negative weights of each sensor. These weights must be applied according to the accuracy of the sensors. This is necessary to relate Wahba's problem to Maximum Likelihood Estimation, a technique that uses sensor accuracy to help to more accurately model a system. Using the orthogonality of A, the unit norm of the unit vectors, and the cyclic invariance of the trace we can rewrite $L(A)$ in 25.

$$L(A) = \lambda_0 - tr(AB^T) \tag{25}$$

$$\lambda_0 = \sum_{i=1}^{N} a_i \quad \text{and} \quad B = \sum_{i=1}^{N} a_i b_i r_i^T$$

Solutions to Wahba's problem include Davenport's q method, Singular Value Decomposition, Quaternion Estimator (QUEST) and Estimators of the Optimal Quaternion (ESOQ), and start with the rewritten error estimation above. Through various real world applications, QUEST and ESOQ are deemed significantly faster than their robust counterparts.

2.1.5.2 Quaternions

For the orientation of the spacecraft to be determined, controlled, or otherwise used, there must be a way of expressing it. A common way of doing this in other applications is with a sequence of Euler angles. This information can be determined and operated on as a three element vector and is generally enough to uniquely determine the orientation. For some applications, especially spacecraft, three Euler angles cannot determine the orientation because of singularities in the determination when an angle is near 90 degrees. For this reason, spacecraft often use quaternions, which are four element vectors with definitions for additional operations. Quaternions are composed of a three element "vector part" and a single "scalar part," and because of the additional element, there are no singularities when expressing orientation with quaternions (Crassidis). Quaternions are also computationally easy to work with because most calculations can be done with quaternion operations instead of trigonometry. Just like most other attitude representations, quaternion attitudes can be converted to other systems, such as Euler angle representations, which are more intuitive to visualize. Because of these advantages, quaternion attitude representations are great for a cubesat mission, as the computations are less taxing for the OBC and the lack of singularities allows for diverse mission profiles.

2.1.5.3 Davenports Q Method

Mathematician Harold Davenport proposed a solution to Wahba's problem that maximizes the following gain function, which is a quantity from eq. (wahba's problem):

$$g = tr(AB^T)$$

He specified a rotation matrix A that can be expressed in terms of euler parameters:

$$A = (q_0^2 - \epsilon^T\epsilon) * [I_{3\times3}] + 2\epsilon\epsilon^T - 2q_0|\epsilon|$$

Where ϵ =(q1,q2,q3)and q0 is the scalar term of the quaternion

The Gain function g() can then be written in terms of the 4x4 K matrix:

$$g(\bar{q}) = q^T[K]q$$

$$K = \begin{bmatrix} \sigma & Z^T \\ Z & S - \sigma[I_{3\times3}] \end{bmatrix}$$

$$B = \sum_{i=1}^{N} a_i b_i r_i^T$$
$$|S| = [B] + [B]^T$$
$$\sigma = tr([B])$$
$$[Z] = [B_{23} - B_{32}\ B_{31} - B_{13}\ B_{12} - B_{21}]^T$$

In order to maximize the gain function, we must abide by the unit length constraint, which means we cannot set the values to infinity. Due to this, a lagrange multiplier is needed to yield a new gain factor g'.

$$g(q) = q^T[K]q - \lambda(q^Tq - 1)$$

Differentiating and setting the equation equal to zero, will find the extrema of the function .

$$0 = \frac{d}{dq}(g(q)) = 2[K]q - 2\lambda q$$

which can, alternatively, be expressed as:

$$[K]q = \lambda_{max}q$$

Therefore, the desired euler parameter vector is the eigenvector of the K matrix. In order to maximize the gain, we have to choose the largest eigenvalue. Through substitution of equation 3.9 into equation 3.7, and applying some linear algebra, it is easy to show that:

$$g(p) = \lambda$$

2.1.5.4 QUEST

The QUEST method (QUaternion ESTimator) builds off of davenport's Q method, and allows for more frequent attitude computations. Currently, it is the most widely used algorithm for solving Wahba's problem, and was first used in 1979 by the MAGSAT spacecraft. The algorithm begins by rewriting into two separate equations:

$$[(\lambda_{max} + trB)I_3 - S]\hat{q}_{1:3} = \hat{q}_4 z$$

$$(\lambda_{max} - trB)\hat{q}_4 - \hat{q}_{1:3} z = 0$$

From there, we can rewrite q using the classical adjoint representation, knowing that the adjoint divided by the determinant gives the inverse of a matrix.:

$$q_{1:3} = q_4((\lambda_{max} + trB)I_3 - S)^{-1} z = \frac{q_4(adj((\lambda_{max} + trB)I_3 - S)z)}{\det((\lambda_{max} + trB)I_3 - S)}$$

From there, we can use the Cayley-Hamilton theorem, which states that every square matrix over a real or complex field satisfies its own characteristic equation [cite], the previous equation can be rewritten in terms of the classical adjoint. This theorem holds for general quaternionic matrices, which is the case for this part of the QUEST algorithm and gives the optimal quaternion estimate as follows:

$$q = \alpha \begin{bmatrix} adj(\rho I_3 - S)z \\ \det(\rho I_3 - S) \end{bmatrix} \rho = \lambda_{max} + trB$$

The term is determined by normalizing the resultant q. This equation assumes we know the value of max, which is the solution to the characteristic equation of the K matrix. Finding the maximum eigenvalue of the characteristic equation is complicated for states using more than two observations. However, in the case of our CubeSat, we are only using two observations, which means the characteristic equation for K has a simple closed form solution. Several simplifications result from only having two observations, which ends up yielding the following equation for λ_{max}:

$$\lambda_{max} = a_1^2 + a_2^2 + 2a_1 a_2[(b_1 \odot b_2)(r_1 \odot r_2) + ||b_1 \otimes b_2||||r_1 \otimes r_2||]^{\frac{1}{2}}$$

2.2 Command and Data Handling

The Command and Data Handling (C&DH) subsystem is composed of many different components and systems whose purpose is to manage the entire spacecraft throughout its mission. This includes during the startup, deployment, detumble, and general mission phases. The key components of the C&DH system are the onboard computer, flight software, radio,

and data storage systems. The C&DH system is responsible for all aspects of the spacecraft's operation. It implements the control system, taking sensor data and issuing commands to the thrusters and ADC systems. It also collects, stores, and transmits sensor and payload data to ground stations. The individual component selection and overall architecture of the C&DH system is dependent on the overall needs of the mission and scientific payload.

2.2.1 Data Handling Requirements

The needs of the scientific payload drives the majority of the data handling requirements. Given the processing power of modern processors, little consideration need be given to the needs of the ADC system. The mission profile of the satellite is relatively simple, requiring only basic slew operations, well within the capabilities of the onboard computer. In comparison the payload will be producing large amounts of data that needs to be stored, possibly operated on, and eventually transmitted to ground stations. The payload produces data at 13.7 kbps. This data must be stored and eventually down-linked to a ground station. Based on the ground station coverage a total of 237.01 Mb of data may be downloaded each day. Thus there must be sufficient storage space to collect data over the course of several orbits.

2.2.2 Onboard Computer

We have selected the Clyde Space Kryten-M3 flight computer as our onboard computer (OBC). The Kryten-M3 is a flight proven OBC built around a Cortex-M3 processor core running at 50 MHz. Similar OBC's from Clyde Space were also chosen by previous MQPs. The OBC includes 8 MB of MRAM and 4 GB of bulk flash memory. Both the MRAM and bulk storage include automatic error detection and correction (EDAC). This is especially important for correcting potential errors introduced by the high radiation environment of orbit. The system also includes space for an external SD card, increasing the bulk storage capacity.

The Kryten-M3 is designed for use with FreeRTOS, a real-time operating system commonly used in high reliability embedded applications. FreeRTOS is highly configurable and capable of managing the entire spacecraft and scientific payload.

2.2.3 Radio Transceiver

The radio transceiver enables communication between the satellite and ground stations. Different radios operate in different bands and are capable of different data rates. Certain frequency bands are regulated by government agency and require specific approval to use. Depending on the data handling needs of the mission it may be necessary to use a radio capable of a higher data rate. As a result it may be necessary to consider the regulatory approval process needed to operate such radio systems.

Figure 9: Kryten-M3 Onboard Computer

3 Analysis

3.1 Detumbling

Below is the analysis for the detumbling simulation, created in Simulink, the first stage of attitude control. Without successfully detumbling the spacecraft after deployment, the mission would not be able to continue to the next phase. The team simulated the detumbling of the CubeSat using Simulink. This was done by simulating the on board magnetometers and applying B-dot control theory to actuate the magnetometers. An option to use the CubeSat's gyroscope was also simulated.

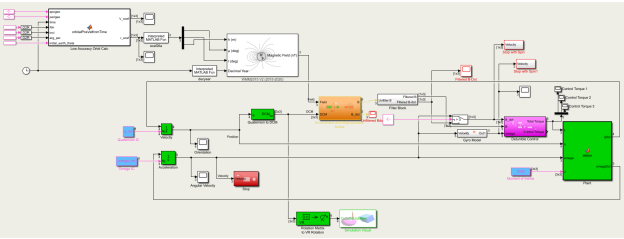

Figure 10: Total Detumbling Simulink Simulation

Overall, the detumbling of the satellite was accomplished by providing inputs into our custom Simulink function 'attdyn' or attitude dynamics. The attitude dynamics block required an input control torque, quaternion, angular velocity, and moment of inertia, in order to output the rate of change of the quaternion ($\dot{q}$) and the rate of change of the angular velocity ($\dot{\omega}$). Moment of inertia was a constant input into the system and was given by the geometry of our CubeSat.

The attitude dynamics block uses the inputted quaternion to create a quaternion product

matrix, denoted $\Xi(q)$, that speeds up the quaternion multiplication process.This is because $\Xi(q)$ is the same as $[q\odot]$, which is a common quaternion operator.

From there, expressions for $\dot{q}$ and $\dot{\omega}$ can be derived and outputted.

$$[L] = \begin{bmatrix} L(1) \\ L(2) \\ L(3) \end{bmatrix} \tag{26}$$

$$[q] = \begin{bmatrix} q(1) \\ q(2) \\ q(3) \\ q(4) \end{bmatrix} \tag{27}$$

$$[\omega] = \begin{bmatrix} \omega(1) \\ \omega(2) \\ \omega(3) \end{bmatrix} \tag{28}$$

$$[\Xi(q)] = \begin{bmatrix} q(4) & -q(3) & q(2) \\ q(3) & q(4) & -q(1) \\ -q(2) & q(1) & q(4) \\ -q(1) & -q(2) & -q(3) \end{bmatrix}$$

$$\dot{q} = \frac{1}{2}\Xi_q \omega$$

$$\dot{\omega} = \frac{J}{\omega \cdot J\omega}$$

The quaternion and angular velocity inputs were first given by an initialized value from the simulations initialization file. Then, the outputted $\dot{q}$ and $\dot{\omega}$ got fed back into the Simulink loop, and integrated using a built in continuous integration block in Simulink. The resulting quaternion and angular velocity (ω), then got passed back into the 'attdyn' block.

Inputting the control torques into the 'attdyn' block was a little more complicated and required the simulation of the magnetometers and the application of B dot control.

3.1.1 Simulating the Input Control Torques

The first step in finding the necessary control torques for our attitude dynamics block, was to simulate the magnetometers. This was done with our custom Simulink function 'Environment and Magnetometer Model.' In order to use this function, we needed to take the inertial magnetic field reading and convert it to the body fixed frame by multiplying it by the spacecraft's body fixed attitude.

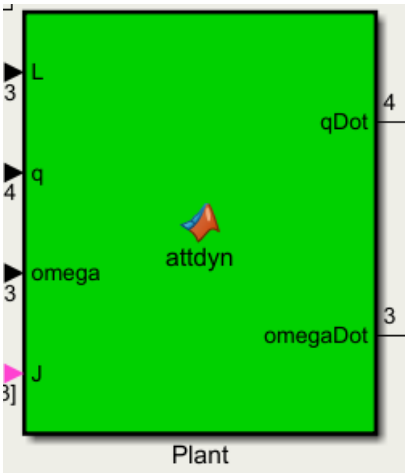

Figure 11: Attitude Dynamics Plant

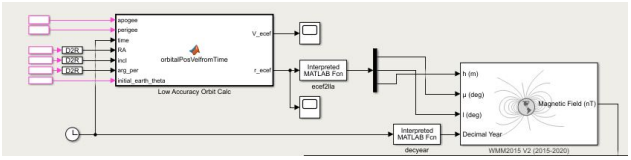

Figure 12: Simulation Path to Find Inertial Magnetic Field Vector

To find the inertial magnetic field vector, we simulated the following path shown below.

On the left, in the pink outlined boxes, are the constants that feed into the simple two-body time based orbit propagator. The constants fed into the orbit propagator block are the apogee, r_a, perigee, r_p, time, RA, inclination, i, argument of perigee, w, and the initial earth theta. The apogee, perigee and initial earth theta, θ_{i_earth} are unaffected constants. The RA, inclination and argument of perigee are entered as constants, in degrees, and translated into radians using a "Degrees to Radians" converter within Simulink. The "Degrees to Radians" converter follows the equation below.

$$Radians = \theta * \frac{\pi}{180}$$

The final input, time, t, was specified by the clock input block which outputs the current simulation time as the simulation was running.

These seven inputs were then sent through the low accuracy orbit propagation block to output V_{ecef} and r_{ecef}.

In order to find V_{ecef} and r_{ecef}, the following process was used. To start, the μ of Earth, the semi-major axis, a, period, p_t, the mean motion, in radians, n, and time since perigee, TSP.

$$\mu = 3.986e10^{14}$$

$$a = \frac{r_a + r_p}{2}$$

$$p_t = 2\pi * \sqrt{\frac{a^3}{\mu}}$$

$$n = \frac{2\pi}{p_t}$$

Time Since Perigee is equal to the remainder of $\frac{t}{p_t}$.

After this, the mean anomaly, M, the specific energy, and the eccentricity are found.

$$M = nTSP$$
$$\epsilon = \frac{-\mu}{2a}$$
$$e = \frac{r_a - r_p}{2a}$$

After this, we then need to compute the numerical approximation of the Inverse Kepler Equation. First, you need to find the eccentric anomaly, E. To do this, you initialize the values E_n and E_{nplus}, being 10 and 0 respectively. Then, we ran E_n and E_{nplus} through a while-loop that states as long as the absolute value of E_n and E_{nplus} is greater than 0.001, E_n and E_{nplus} equal each other, therefore, $E_{nplus} = E_n - (\frac{(E_n - \epsilon \sin(E_n) - M}{1 - \epsilon \cos(E_n)})$. Then, the output of the while-loop, E_{nplus} is equal to the eccentric anomaly, E. Using E, we then found the true anomaly, ν, the distance, d, the angular momentum, h, and the orbital parameter, p.

$$\nu = 2atan(\sqrt{\frac{1+e}{1-e}})tan(\frac{E}{2})$$
$$d = a(1 - e\cos(E))$$
$$h = \sqrt{\mu a(1 - e^2))}$$
$$p = \frac{h^2}{\mu}$$

From here, we are then able to compute the position components of the spacecraft.

$$x = d(\cos(RA)\cos(w + \nu) - \sin(RA)\sin(w + \nu\cos(i))$$
$$y = d(\sin(RA)\cos(w + \nu) - \cos(RA)\sin(w + \nu\cos(i))$$
$$z = d(\sin(i)\sin(w + \nu))$$

Using these, we find our position vector:

$$Position = [xyz]$$

From there, we are then able to find the velocity components.

$$\dot{x} = \frac{(x)(h)(e)}{(d)(p)}\sin(\nu) - \frac{h}{d}(\cos(RA)\sin(w+\nu) + \sin(RA)\cos(w+\nu)\cos(i))$$
$$\dot{y} = \frac{(y)(h)(e)}{(d)(p)}\sin(\nu) - \frac{h}{d}(\sin(RA)\sin(w+\nu) - \cos(RA)\cos(w+\nu)\cos(i))$$
$$\dot{z} = \frac{(z)(h)(e)}{(d)(p)}\sin(\nu) + \frac{h}{d}\sin(i)\cos(w+\nu)$$

From there, we can calculate the velocity vector:

$$Velocity = [\dot{x}\ \dot{y}\ \dot{z}]$$

Using ω_{earth}, which is equal to $7.29e^{-5}$ rad/s, we can find the true anomaly at time t, θ_t.

$$\theta_t = \omega_{earth}t + \theta_{i_{earth}}$$

Next, we defined a normal rotation matrix that includes θ_t.

$$[RM] = \left(\begin{bmatrix} cos(\theta_t) & sin(\theta_t) & 0 \\ -sin(\theta_t) & cos(\theta_t) & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)$$

Finally, to find the position of the spacecraft in ECEF, we transposed the product of the rotation matrix and the initial position vector defined above. Finding the velocity of the spacecraft in ECEF was a bit more complicated, but defined below.

$$[V_{ecef}] = RM\left(V\begin{bmatrix}\omega_{earthY} \\ \omega_{earthX} \\ 0\end{bmatrix}\right)$$

Where V is the velocity vector defined above.

After we found V_{ecef} and r_{ecef}, we then converted r_{ecef} from ECEF to LLA using a built in Simulink block in the Aerospace toolbox. From there, the r_LLA coordinates filter into a Demux block in Simulink. The Demux block split the vector signals into scalar values. This specific Demux block splits the input into three scalar outputs, height, h, in meters, latitude, μ, in degrees, and longitude, l, in degrees. These three scalar outputs then flow into the World Magnetic Model, WMM2015, that is also built into Simulink, along with the decimal year, which is created by a clock and a built in MatLab function that converts the input clock

time, that's equal to the time input in the orbital propagator, to decimal years. The World Magnetic Model implements the mathematical representation of the National Geospatial Intelligence Agency (NGA) World Magnetic Model, which is then used to find the Magnetic field vector, in nT, which is then fed directly into the Environmental and Magnetometer Model as one of the inputs.

To find the body fixed attitude, the quaternion within the detumbling loop was fed into a built in Simulink function that converted quaternions to a direct cosine matrix. Once we had both the inertial magnetic field reading, and the direct cosine matrix representation of the spacecraft's attitude, a simple matrix multiplication yielded our true magnetic field in body fixed coordinates.

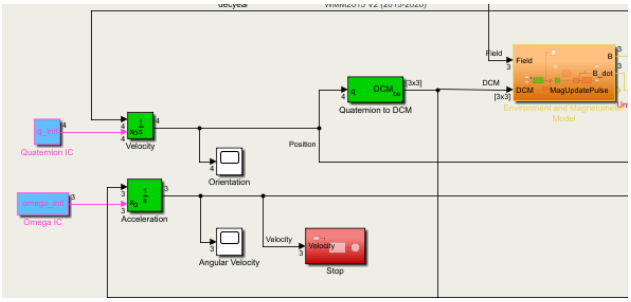

Figure 13: Integration of the q and omega vectors and Simulink q to DCM block

This output was then seeded with noise via a noise maker we created, to simulate how a real sensor would operate. The noise characteristics we inputted into the simulation are on par with those listed in the spec sheet for our magnetometer, and provided a random Gaussian distribution that was added to the true magnetic field vector. Next, the magnetic field with the addition of noise was passed through a Simulink Quantizer that discretizes our input to a specific interval. For our case, the quantization value was set to 200, which was an arbitrary value that we found worked well after some trial in error. The quantization block works by using a quantization algorithm with a round-to-nearest method to map signal values to quantized values at the output that are defined by the quantization interval. After the vector is quantized, it is passed through a built in Simulink function called 'Zero-Order Hold.' This block essentially sets the sample rate of our signal to any specific value. We set our value to $\frac{1}{220}$, or 220 Hz to be consistent with our sensors refresh rate. For comparison purposes, a 'Discrete Derivative' block was added in order to see the unfiltered rate of change of the magnetic field, or $\dot{B}$. This was to ensure that once we passed the magnetic field through our low pass filter, the filter was properly working.

The next step was to pass our outputted B vector from our 'Environment and Magnetometer Model' through a low pass filter to filter out the noise we previously added.

Simulink allows for users to specify passband edge frequency and stopband edge frequency in Hz. Those values were set to 90 and 120 Hz respectively. This was due to the sample rate of our magnetometer being 220 Hz. Any fast, small changes in magnetic field vector are most likely errors, so we want to remove any changes that aren't happening slowly.

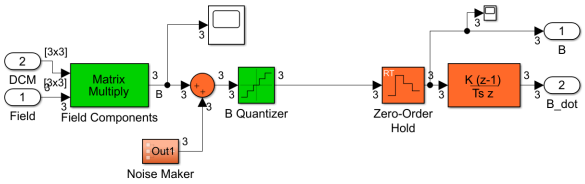

Figure 14: Contents of the Magnetometer Simulation Block

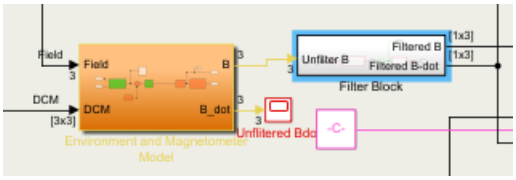

Figure 15: Magnetometer Simulation Block and Low Pass Filter

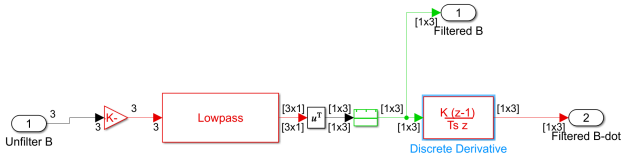

Figure 16: Contents of the Low Pass Filter Block

In theory, this also means that if we were spinning fast enough, it would filter out that too, and make the satellite spin faster. However, because we are using gyroscope measurements as a second reference, this situation would not occur. The derivative of the signal was then found using another 'discrete derivative' block, to give us the filtered rate of change of the magnetic field, or $\dot{B}$.

3.2 Reaction Wheel Sizing

The reaction wheels selected for the CubeSat had to be capable of producing enough torque to counteract the disturbance torques on the satellite. First, the gravitational torque was defined by:

$$T_g = \frac{3M}{2R^3}|I_x - I_y|sin(2\theta)$$

Where θ is equal to 5°, or 0.0872005 radians, M is the mass of the Earth, $3.9816x10^{14}\frac{m^3}{s^2}$ and R is the radius of the earth, $6.978x10^6m$. The next disturbance torque needed to be accounted for was the torque due to solar pressure. This was defined by:

$$T_{sp} = F(C_{ps} - C_g)$$

Where C_{ps} is the center of solar pressure and C_g is the center of gravity. The flux due to solar pressure, F is defined as:

$$F = \frac{F_s}{c}A_s(1+q)cos(l)$$

Where l is the angle of incidence to the Sun, F_s, is the solar flux constant , $1.367\frac{W}{m^2}$, c is the speed of light, $3x10^8\frac{m}{s}$, A_s is the surface area of the RAM facing surface, 0.02 m, and q is the coefficient of reflection, 0.6. Next We can then calculate the torque due to the magnetic field by the following equation:

$$T_m = DB$$

Where D is equal to the dipole moment of the vehicle and B is the Earth's magnetic field. The final disturbance torque to consider was the torque due to aerodynamic pressure. This was defined as:

$$T_a = 0.5[\rho C_d A_s V^2](C_{pa} - C_g)$$

Where V is the velocity of the vehicle and C_{pa} is the center of aerodynamic pressure. With all of the disturbance torques calculated and summed, we now had a known value for the

disturbance torques we needed to overcome. To find the torque for the reaction wheel sizing, T_{rw}, we evaluated the following expression:

$$T_{rw} = T_D(M_f)$$

Where M_f is the margin factor to help calculate the torque of the reaction wheel for the disturbance rejection and T_D is the reaction wheel torque for the worst case anticipated torque. The Reaction wheel torque, T_{rw} must be equal to the worst case anticipated disturbance torque plus some margin. Finally, the momentum storage can be calculated by:

$$h = T_D \frac{t}{4}(0.707)$$

Where t is the orbital period, in seconds and 0.707 is the rms average of a sinusoidal function.

3.3 Simulink Control

3.3.1 Detumble Control Subsystem

This is the subsystem that takes the filtered measurement readings and uses them to command control torques to detumble the spacecraft. There are two different control laws that can be chosen in this block by setting a variable in the initialization file depending on the desired control method.

3.3.1.1 Bang-Bang B-dot Control

The first control law block made for this model implements the control law described in 9 using an estimate of $\dot{\vec{B}}$ and a preset maximum for magnetorquer dipole moment. The block sums up the individual commanded dipole moments from each of the magnetorquers and the finds the resultant torque using 4. This resultant torque, $\vec{L}$, is then passed to the rest of the subsystem.

3.3.1.2 Proven-Stability B-dot Control

The second control law block, it is a direct improvement on the first control law because of the use of a direct measurement of the angular velocity. This block implements 6 to find a resultant torque given measurements from the gyroscopes and the magnetometers. Because this control law does not use $\dot{\vec{B}}$, which is found by taking discrete differences of a corrupted signal, it is more effective than the other controller. In addition, as was shown in an earlier section, this control law is has proven Lyapunov stability. The advantages of this control law far outweigh the downside of having to also use the gyroscopes for this law, so this is the control law that is used for our analysis. The option of using the other law is still available, although discouraged.

3.3.1.3 Control Activation Delay

In order to meet any potential guidelines for launching from the ISS, this block gives us the option of delaying the activation of the active controls until a set amount of time has passed. This also has the benefit of allowing our controls to ignore the initial transient signals that result from differences between initial conditions and initial estimates upon simulation start up. This block is very simple and does not apply a signal delay to the commanded torques when they are allowed to pass. In addition, it does nothing to any disturbance torques that may be added to the simulation regardless of simulation time or set delay.

3.4 STK

Systems Tool Kit, or STK, is a physics-based software packaged created by Analytical Graphics (AGI). STK allows engineers to perform analyses of ground, sea air and space objects in their respective environments. STK is used in government, commercial and defense applications around the world to determine the attitude, spatial relationships and time-dynamic positions of objects under many simultaneous constraints. STK's ability to simulate subsystems like ADC is what makes the software so valuable to the aerospace industry. Our team considered using STK to simulate the detumbling and attitude control of our 6U CubeSat. An example of the simulation can be seen below.

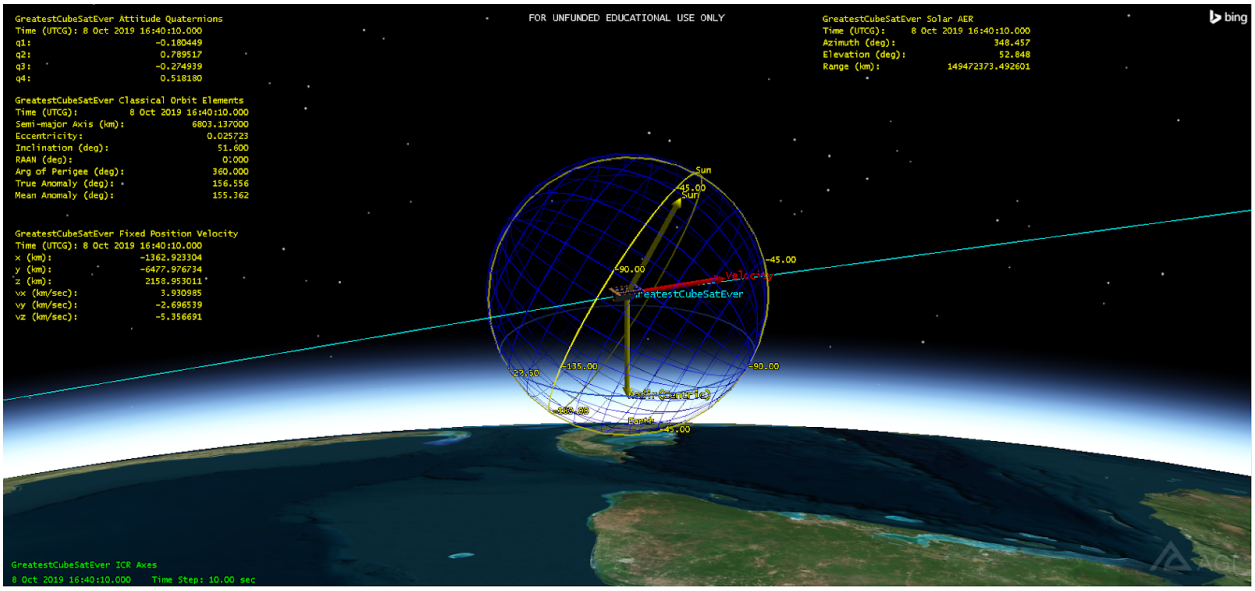

Figure 17: STK Satellite View

The figure above contains a snapshot of our CubeSat in its simulated elliptical orbit. STK has the ability to overlay particular orbital elements as vectors, such as sun vector and the nadir vector. Attitude determination in STK is done primarily with the use of quaternions. We originally intended on integrating directly between STK and MATLAB. MATLAB was to be used as our attitude determination and controller, whereas STK was simply to be used as a visual simulation, with the potential of utilizing it for sensor readings such as magnetic

field and the sun vector. Utilizing the work of Xiang Fang and Yunhai Geng of the Research Center of Satellite Technology of Harbin Institute of Technology (CITATION) Fang and Gent laid out detailed steps into the integration of MATLAB into a real-time running STK environment. MATLAB was to be used to introduce sensor noise, whether that be from MATLAB simulated sensors, or STK sensors with their data passed into MATLAB. We would then pass the noisy sensor data through an extended Kalman Filter (EKF) prior to being sent to the controller. This EKF-processed data could be compared to the full state vector to check on the functionality of our EKF. Attitude control commands would then be sent to the controller flow, and then pass the data back into the STK environment for visual simulation, completing the loop.

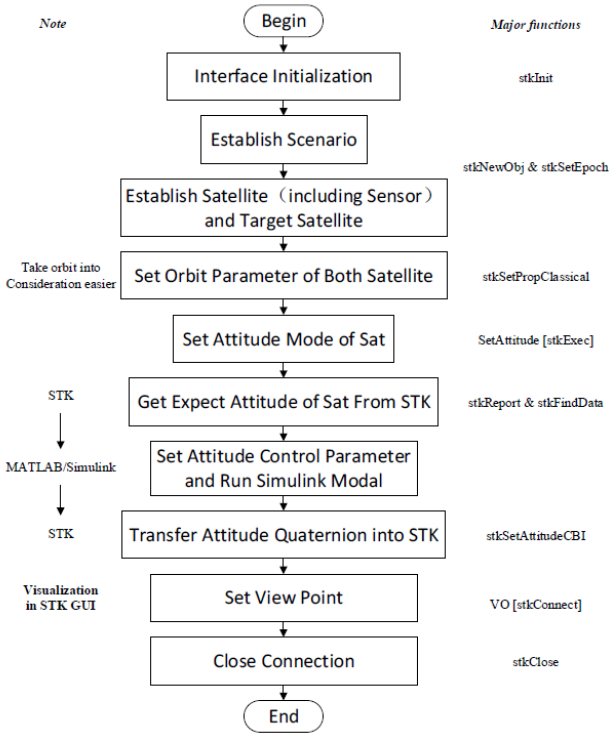

Figure 18: STK-Matlab Integration Workflow

Although without any code snippets or a codebase to be referenced, it is possible to integrate MATLAB into STK using the above figure and aforementioned paper as a guide. However, our focus changed from purely using MATLAB for our attitude and orbital simulation and control, to primarily utilizing Simulink into our control scheme.

Since changing our focus to Simulink, the MATLAB to STK integration has become less realistic of a goal, especially with the introduction of the Aerospace Blockset CubeSat Simulation Library, which included an orbital propagator, which simulates in 3-D the orbital and attitude elements of a CubeSat. We will utilize this CubeSat Simulation Library along with using a Level-2 S-Function block, which will connect our Simulink controller to a STK environment, similar to what we had planned with MATLAB. Similar to our approach outlined above, Simulink will do all of the spacecraft control, and STK will merely model

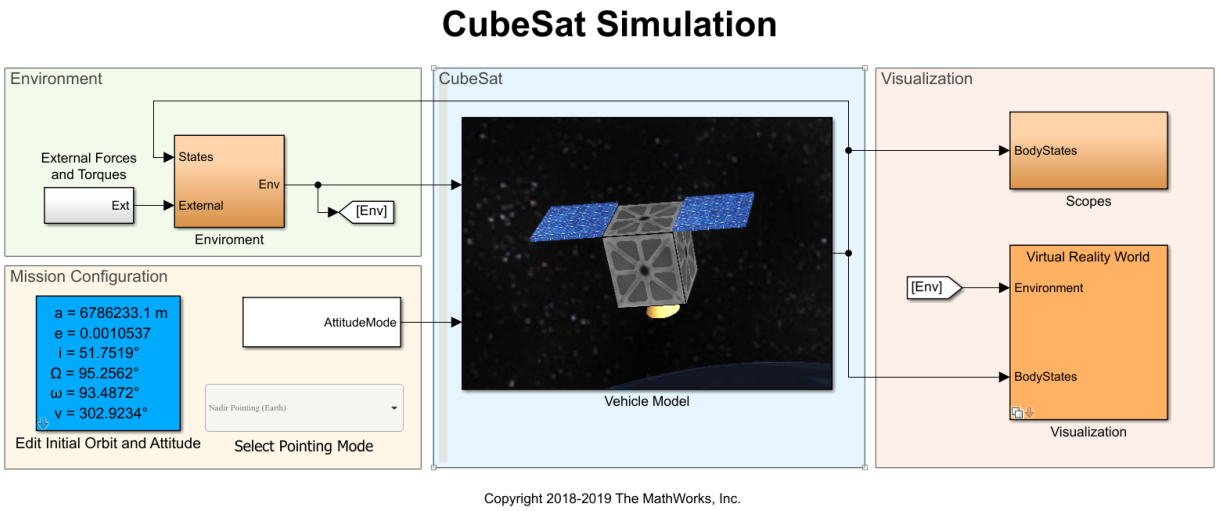

Figure 19: Matlab CubeSat Simulation Toolbox

and return whatever sensor data we request.

3.5 System Power Requirements

4 Results

References

[1] J. Foley, "The cubesat program." California Polytechnic Institute, San Luis Obispo, 1999.

[2] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, and R. Leonardi, "Towards the thousandth cubesat: A statistical overview," *International Journal of Aerospace Engineering*, vol. 2019, 2019.

[3] S. Jackson, Ed., "NASA's cubesat launch initiative," *NASA*. NASA, Jan-2019.

[4] "CubeSat market report," *Market and Markets*.

[5] "NASA's insight mars lander," *NASA*. NASA, Jun-2019.

[6] N. P. Paschalidis, "Mass spectrometers for cubesats." NASA Goddard Space Flight Center, 2017.

[7] H. D. Black, "A passive system for determining the attitude of a satellite," *AIAA Journal*, vol. 2, no. 7, pp. 1350–1351, 1964.

[8] "The world magnetic model," *World Magnetic Model*. NOAA.

Appendix