A Low-Cost Attitude Determination and Control System and Hardware-in-the-Loop Testbed for CubeSats

Benjamin Jensen

CMU-RI-TR-22-43
August 04, 2022

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Zachary Manchester, *chair*
Michael Kaess
Brian Jackson
Kevin Tracy

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Abstract

Since their initial development in the late 1990s, CubeSats have quickly grown popular due to their relatively low cost and short development period. However, CubeSat launches are prone to failure, with less than half of CubeSats completely fulfilling their mission objectives. To improve mission success, we present a low-cost attitude determination and control system (ADCS) that scales to 1U CubeSats and other small satellites. The ADCS is necessary to point satellite antennae and solar panels effectively and to orient on-board payloads to the target locations, all of which are crucial to a mission's success.

Modern methods for attitude determination and control rely on sophisticated actuators and sensors, such as reaction wheels and star trackers, that are often unable to be used on CubeSats due to weight, volume, and cost restrictions. Our proposed system relies on simple consumer-grade magnetometers, gyroscopes, and sun sensors to estimate the attitude of the satellite, along with a set of magnetic torque coils for actuation. Additionally, we provide a new method for calibrating on-board sensors that requires less storage space and allows for all parameters to be time-varying. By combining these low-cost sensors and actuators with sophisticated calibration, estimation, motion planning, and control software, we achieve full three-axis attitude determination and control. The system is also completely solid-state, with no moving parts or consumable propellant, greatly reducing the chance of hardware failure.

To further improve the development cycle, we have developed an open-source hardware-in-the-loop simulator to enable rapid testing of ADCS algorithms and other flight software. The result is a robust, open-source development suite for CubeSats that is low cost, easy to program, and reliable.

Acknowledgments

I owe many thanks to Professor Zachary Manchester, who provided keen insight, wise guidance, and regular support throughout my work on this thesis. His deep understanding and genuine enjoyment of research have inspired me to challenge myself to work both harder and smarter. I also would like to thank my colleagues in RExLab for their wonderful help, especially Kevin Tracy, who answered all my constant questions with aplomb.

I am grateful to my undergraduate advisors, Professors Willie Harrison and Michael Rice, for getting me started in research and letting me participate in numerous projects, and for putting up with me in the beginning when I required a lot of attention and produced few results.

Finally, I would like to thank my cousin, Brady Moon, who has provided enourmous guidance throughout my many years in school, and always inspires me with his enthusiasm for learning.

Funding

The work presented in this thesis was supported by NASA and the Breakthrough Foundation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Development of new technologies and more accurate models requires the ability to gather data, test assumptions, and validate current methodologies. Because this often requires multiple iterations, this development works best when the process is consistent, quick, and low-cost. Unfortunately, when it comes to space research, the development process is rarely consistent, quick, *or* low-cost. Gathering new data and evaluating new models often requires large satellites that are complicated and expensive and can take years to get launched into space for use, dramatically slowing down the development and testing processes.

CubeSats ("Cube Satellites") have been proposed to help address the problem of slow, expensive satellite design and deployment by providing a standardized platform from which to develop skills and test new space technologies. These satellites, a particular class of small satellites comprising $10 \times 10 \times 10 cm^3$ units, were initially designed to provide universities with affordable access to space and have done so quite successfully. Since their creation in 1999, there have been over 1,000 CubeSats launched from around the world [24]. Due to their small size, relatively inexpensive components, and standardized platform, CubeSats have enabled universities, companies, and individuals to develop space programs. Additionally, because of their light weight, CubeSats are often able to be deployed by the International Space Station, or as auxiliary payloads on other space missions, reducing the time it takes to get one into space.

Thanks to their low cost and fast launch procedure, CubeSats have quickly

become popular and demonstrated their merit. These satellites are used for many purposes, including weather investigation, space-based astronomy, ionospheric and meteorological measurements, and earth imaging. One of their biggest benefits is their unique offer of rapid iterations. This allows them to be deployed relatively quickly and affordably, allowing them to be used as a proof-of-concept or testbed for new technologies before use on much larger, more expensive systems. Additionally, they provide a platform for gathering data from a wide variety of sensors, enabling the validation and enhancement of relevant solar, ionospheric, and thermospheric models, while also enabling people from all realms to more effectively access space to gather data and test their own respective theories.

However, there are still some significant challenges with CubeSats. According to a study as part of the CubeSat Developers' Workshop in 2019, CubeSat missions have a significant fail rate, with only around $30 - 40\%$ completing a full mission and a large portion dead on arrival [24]. There are several contributing factors to this failure rate, including environmental wear, low safety margins, software and hardware errors, and poor communication systems. Perhaps the two greatest benefits of CubeSats—their accessibility and rapid development cycle—are also their Achilles heel, with many satellites being developed and launched by first-time builders in time frames that do not allow for sufficient testing.

1.1 Contributions

Previous work has been done into addressing some of these problems by developing a radiation-resistant hardware platform known as PyCubed [13]. In this thesis, we build on PyCubed to address some of the other common problems that lead to CubeSat mission failure through development of an attitude determination and control system (ADCS) that is scalable to CubeSats and other small satellites. The ADCS is responsible for many roles crucial to a successful mission, including orienting the antenna and solar panels for effective communication and power generation, and pointing on-board payloads to the desired locations.

Our ADCS relies on inexpensive, off-the-shelf sun sensors and an IMU for attitude determination, and includes functionality for calibrating these sensors in-orbit after launch. Attitude control is performed exclusively with magnetic torque coils; while this

adds some complication due to inherent underactuation, it also provides a lightweight and low-cost control system that has low power requirements. We rely on an existing method proposed by Gatherer [10] that provides full three-axis control using only magnetic torque coils, and the result is an entirely solid-state system that reduces the chance of hardware failure. Additionally, we provide a sophisticated simulator with a hardware-in-the-loop (HITL) testbed that can be used for developing new flight algorithms and on-hardware validation before launch. The result is an open-source, inexpensive development suite for CubeSats.

1.2 Related Works

Due to the high importance of a satellite's ADCS, many different systems have been designed that are incredibly effective and robust, and these systems have been consistently effective on a variety of missions. Unfortunately, these methods rely on highly optimized equipment, such as star trackers and reaction wheels, which are often too large and too costly for use on nano and pico-class satellites. Commercial systems developed specifically for CubeSats exist, but at a cost that precludes their use in projects developed by groups with tight cost constraints, such as university research groups or clubs. This has required universities interested in CubeSat development to either invest a large amount of money in each launched satellite, or develop their own systems, making it difficult to test and develop new satellites. Several such projects exist, but are often larger than 1U, use equipment that is expensive, or rely on systems with moving parts that are prone to breaking [5, 7, 9, 20].

A common setup used in HITL testbeds for small satellites involves building a special air-bearing table designed to create a low-torque environment [19], sometimes with the addition of a fixed light source to simulate the sun [9]. Our testbed consists of a box that a CubeSat can be placed into for testing, with an LED embedded in each panel that has an adjustable light level. Rather than relying on an air-bearing table to allow the satellite to rotate, we generate the body-frame sun vector and use the LEDs to illuminate the appropriate sides of the satellite in the appropriate proportions. Additionally, each panel has a mounted magnetometer to measure the magnetic field generated by pulsing the magnetic torquer coils. This box is placed inside of a Helmholtz coil to cancel out the geomagnetic field or replace it with a

desired magnetic field.

Although our HITL setup does not allow for testing of the satellite gyroscope, which can be done in other HITL systems, the system as a whole is smaller and simpler, relying on common materials and off-the-shelf hardware, and can be controlled with an Arduino.

Chapter 2

Background

2.1 Unit Quaternions

While there are many different parameterizations for representing the attitude of a rigid body, one of the most popular methods is a unit quaternion. This parameterization uses a scalar term and a three-element vector term to represent the orientation of an object, and is subject to a unit-norm constraint, so that

$$q = \begin{bmatrix} q_s & \boldsymbol{q}_v \end{bmatrix}, \qquad ||q|| = 1. \tag{2.1}$$

This parameterization was chosen to avoid issues caused by singularities inherent to three-parameter representations (e.g., Euler angles, axis-angle, etc.) and more complicated constraints required by rotation matrices.

Traditionally, composition of unit quaternions is handled using the Hamilton product; however, we choose to use the quaternion notation suggested by Jackson et al. [14], which converts quaternions to matrices for mathematical convenience. With this method, the composition of two unit quaternions q_1 and q_2 can be calculated as

$$q = L(q_1)q_2 = q_1 R(q_2) \tag{2.2}$$

where

$$L(q) = \begin{bmatrix} q_s & -\mathbf{q}_v^T \\ \mathbf{q}_v & q_s I + [\mathbf{q}_v]_\times \end{bmatrix}, \tag{2.3}$$

$$R(q) = \begin{bmatrix} q_s & -\mathbf{q}_v^T \\ \mathbf{q}_v & q_s I - [\mathbf{q}_v]^\times \end{bmatrix}, \tag{2.4}$$

and the $[\cdot]^\times$ notation represents the skew-symmetric cross-product matrix for a three-element vector. Using this notation, a vector v can be rotated as

$$v' = H^T L(q) R(q)^T H v, \tag{2.5}$$

where

$$H = \begin{bmatrix} 0 \\ I_3 \end{bmatrix} \tag{2.6}$$

converts a three-element vector into a quaternion with zero scalar. This notation can be simplified by defining

$$Q(q) = H^T L(q) R(q)^T H, \tag{2.7}$$

so that

$$v' = Q(q) v. \tag{2.8}$$

Finally, when working with Jacobians, a "conversion factor" is needed to convert between the four-parameter unit quaternion and a three-parameter vector space. This can be achieved using the attitude Jacobian

$$G(q) \in \mathbf{R}^{4\times 3} = \begin{bmatrix} -\mathbf{q}_v^T \\ q_s I_3 + [\mathbf{q}_v]^\times \end{bmatrix}. \tag{2.9}$$

2.2 Earth's Albedo

Many attitude determination systems rely on various types of sun sensors to estimate the sun vector, which generate current proportional to their illumination, and these are particularly common in low-cost designs. However, while the majority of the illumination *does* come from the sun, there is also a non-negligible amount that is reflected off the Earth's atmosphere onto the satellite. This reflected illumination affects the measurement step in Kalman filters and can lead to suboptimal performance. Different methods exist to solve this, including using specialized sun sensors with diode arrays or just treating it as noise; however, we include it in our system using

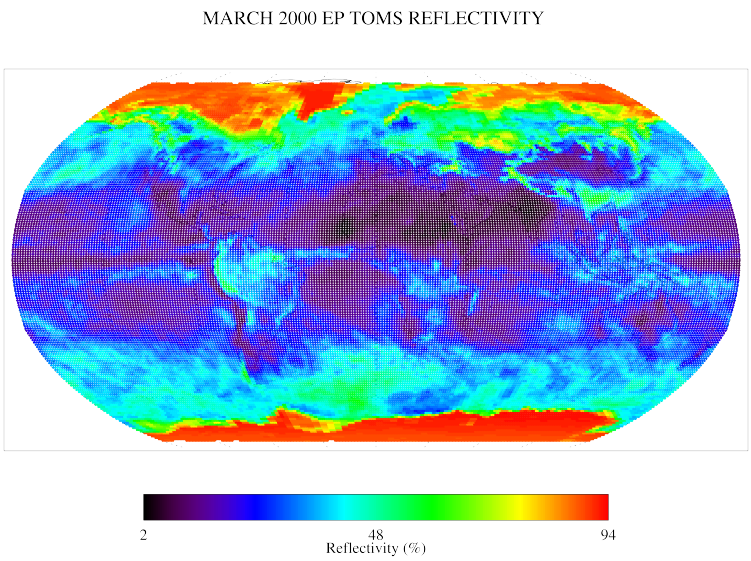

Figure 2.1: Averaged surface reflectivity for Earth as measured in the NASA TOMS mission [6].

the model suggested by Bhanderi [4]. This involves dividing the surface of the Earth into a series of cells, where the reflectivity of each cell is determined by averaging data gathered by the NASA TOMS mission [17] over several years. An example image of average reflectivity is shown in Fig. 2.1.

This reflectivity data is used for all cells on the Earth's surface that are in the field-of-view of both the sun and the satellite to estimate the amount of light that would be reflected by the Earth's atmosphere onto the satellite. This was added into the simulator using a Julia port [15] of a MATLAB toolbox created by the original author [3].

2.3 Sensor Calibration

Due to manufacturing and installation errors, sensors often need to be calibrated in order to maximize their use; this is particularly true for more affordable versions of off-the-shelf sensors. For our system, the onboard sensors include a magnetometer, the sun sensors, and the gyroscope. Although the onboard gyroscope experiences a

bias β_ω that must be accounted for, it is time-varying and is estimated as part of the state using the MEKF discussed later in Chapter 5. The calibration processes for the remaining sensors are described in their respective sections below.

2.3.1 Magnetometer Calibration

Magnetometers are popular sensors in attitude determination because they are lightweight, have low power requirements, and involve no moving parts. However, these sensors can be very noisy, particularly for lower-cost models, and this noise can result in poor-quality attitude estimation for the satellite. There are several factors that contribute to this error, including installation error and corruption from soft-iron and hard-iron metals. The effects of these factors are often grouped into three error categories: scale factors, non-orthogonality angles, and biases. If we assume these errors to be time-invariant, we can estimate them on-board the satellite and their effects can be corrected out of a magnetometer's measurement, increasing their accuracy.

The measured magnetic field in the body frame $\tilde{B}^B = [\tilde{B}_x^B, \tilde{B}_y^B, \tilde{B}_z^B]^T$ is commonly modeled as

$$\tilde{B}_x^B = aB_x + x_0 + \eta_x \tag{2.10}$$

$$\tilde{B}_y^B = b(B_x \sin\rho + B_y \cos\rho) + y_0 + \eta_y \tag{2.11}$$

$$\tilde{B}_z^B = c(B_x \sin\lambda + B_y \cos\lambda \sin\phi + B_z \cos\lambda \cos\phi) + z_0 + \eta_z, \tag{2.12}$$

where a, b, and c are the scale factors along the x, y, and z axes; x_0, y_0, and z_0 are the bias terms along each axis; ρ is the non-orthogonality angle between a y-axis orthogonal to x and the measured $\tilde{y}$; λ is the non-orthogonality angle between a z-axis orthogonal to x and the measured $\tilde{z}$; and ϕ is the non-orthogonality angle between a z-axis orthogonal to y and the measured $\tilde{z}$, as shown in Fig. 2.2 [8, 22]. Additional measurement noise along each axis is included as η_x, η_y, and η_z.

Because the magnetometers are used in attitude determination, it is helpful to have an attitude-independent calibration method. This is possible because the magnetometer errors affect the magnitude of a measurement, but a rotation between two frames does not; as such, techniques that rely only on the magnitude of a magnetometer measurement are effective without any need for information about

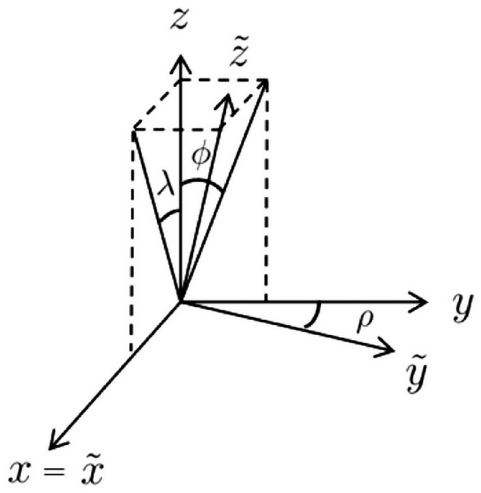

Figure 2.2: Visualization of the non-orthogonality angles for a sensor, as illustrated in [22]

attitude, as shown by Foster [8] and Springmann [22]. This method relies on batch estimation and nonlinear least squares minimization is performed using the Gauss-Newton algorithm. A cost function

$$J = \frac{1}{2}\left[\mathbf{B}_E^2 - f(\tilde{\mathbf{B}}^B, x)\right]^T \left[\mathbf{B}_E^2 - f(\tilde{\mathbf{B}}^B, x)\right], \tag{2.13}$$

is minimized, where $\mathbf{B}_E$ is a vector of the magnitudes of the expected magnetic field vectors at each time step, $\tilde{\mathbf{B}}^B$ is a vector of the measured magnetic field vectors in body frame, x is the current guess for the calibration parameters, and $f(\tilde{\mathbf{B}}^B, x)$ provides the magnitude of the measured vectors after correction by the calibration parameters in x. Note that, while independent of attitude, this method *does* require a time-varying magnetic field.

2.3.2 Sun Sensor Calibration

Sun sensors generate a current when illuminated that is proportional to the angle between the photodiode surface normal and the vector in the direction of the sun. This angle can be used to provide a component of the sun vector in the body frame of the satellite. When multiple sun sensors are used, the full sun vector can be estimated. However, effective estimation requires accurate surface normals for each

photodiode, as well as the scale factor for each photodiode so as to normalize the measured currents. There are many methods for calibrating sun sensors, including attitude-independent methods [8, 21]; however using a recursive, attitude-dependent method, such as that proposed by Springmann [23], allows for arbitrary numbers and configurations of photodiodes. Additionally, the attitude-dependent method allows for the inclusion of Earth's albedo, which can have a significant effect on the illumination of the diodes. This calibration is done on-orbit to account for changes that may occur during launch, and allows for periodic reestimation, as scale factors may degrade over time due to radiation.

Calibration is done by augmenting the state of a traditional MEKF with additional states to track each diode calibration value and surface normal. Because each surface normal is constrained to the surface of a unit sphere, they can be parameterized in terms of an elevation angle ϵ and azimuth angle α, so that the state to be estimated at each time step in the augmented MEKF is

$$x = \begin{bmatrix} q \\ \beta_\omega \\ \mathbf{C} \\ \boldsymbol{\alpha} \\ \boldsymbol{\epsilon} \end{bmatrix}, \tag{2.14}$$

where q represents the attitude (as a unit quaternion, in our case), β is the gyroscope bias, $\mathbf{C}$ is the vector of the scale factors for each diode, and $\boldsymbol{\alpha}$, $\boldsymbol{\epsilon}$ are the vectors of each diode's surface normal, represented as azimuth and elevation angles. At each step of the MEKF, the expected currents (predicted using Eq. 3.13) are compared to the measured currents (generated using Eq. 3.14). This information is used to iteratively update the estimate for each calibration parameter.

Chapter 3

Simulator

In order to aid in the development and testing of an ADCS, a high-fidelity simulator has been developed in Julia. This simulator is open-source and modular, allowing for different controllers and estimators to be swapped in and out as desired, while also accounting for many high-order factors that affect the dynamics of a spacecraft. The simulator is divided into a dynamics module, which is used to update the state of both the satellite and the environment, and a measurement module, which uses the current state to generate measurements for the satellite sensors.

3.1 Dynamics

3.1.1 Orbital Dynamics

For a given satellite, the orbit dynamics can be modelled using the position of a satellite r, its velocity v, and acceleration a.

There are a wide variety of factors that go into calculating the acceleration a, including affects due to the gravity of Earth a_g, the moon a_M, and the sun a_S, as well as drag a_d and solar radiation pressure a_{srp}. Each of the modeled factors is described below in its respective section, with the total acceleration being the resulting sum of each contributing term, so that

$$a = a_g + a_d + a_{srp} + a_M + a_S. \tag{3.1}$$

Note that in the simulation, a_{srp} and a_d both are attitude independent and assume a constant cross-sectional area A for simplicity.

Earth's Gravitational Field

Although Earth's gravitational field is often approximated as uniform for simplicity, in reality the non-uniformities in the Earth combine to form a much more complicated gravitational field. The largest factor for satellites in low-Earth orbits (LEO) are perturbations caused by the Earth's oblateness, known as J_2, but there are many more such perturbations that produce effects on a satellite's orbit. This results in orbital trajectories that cannot be captured using a simplified model using uniform gravity, such as sun-synchronous orbits. In order to account for this behavior in simulation, the gravitational field is approximated using spherical harmonics, where each successive term accounts for higher-order factors contributing to a_g, the acceleration due to gravity [18].

Drag

Because CubeSats and other nanosatellites are often flown in LEO, atmospheric drag can lead to significant trajectory differences, especially as the effect accumulates over multiple orbits. This drag can be estimated as

$$a_d = -\frac{1}{2} C_D \frac{A}{m} ||v_r|| v_r, \tag{3.2}$$

where C_D is the coefficient of drag, m is the satellite mass, A is the cross-sectional area of the satellite, v_r is the satellite velocity relative to the atmosphere, and ρ is the local atmospheric density, which is estimated using the Harris-Priester density model [18].

Solar Radiation Pressure

Direct solar radiation pressure (SRP) also contributes to the acceleration of a spacecraft, and can be approximated as

$$a_{srp} = -P_0 C_R \frac{A}{m} \frac{d}{||d||^3} AU^2, \tag{3.3}$$

where d is the vector from the satellite to the sun, C_R is the radiation pressure coefficient, A is the cross-sectional area of the satellite facing the sun, AU is the astronomical unit, and $P_0 \approx 4.56 \times 10^{-6} Nm^{-2}$ is the solar radiation pressure at 1 AU from the sun [18]. Note that we assume that a surface normal of the satellite points in the direction of the sun for simplicity.

Third-body Gravitational Fields

Finally, the effect of the gravitational fields of both the moon and the sun are modeled as

$$a_M = -\frac{GM_m r_m}{||r_m||^3}, \tag{3.4}$$

$$a_S = -\frac{GM_s r_s}{||r_s||^3}, \tag{3.5}$$

where GM_m and GM_s are the standard gravitational parameters for the moon and sun, respectively, and r_m and r_s are the distances between the satellite and the moon and sun.

3.1.2 Attitude Dynamics

The attitude dynamics of the satellite can be modeled using Euler's equation

$$\dot{\omega} = J^{-1}(\tau - \omega \times J\omega), \tag{3.6}$$

where τ is the sum of the applied torques, J is the inertia matrix of the satellite, and ω is the angular velocity. The three-parameter angular velocity ω can be mapped to the four-parameter unit quaternion q as [14]

$$\dot{q} = \frac{1}{2} L(q) H \omega. \tag{3.7}$$

Finally, the bias β of the gyroscope is updated as a random walk for later use in updating sensor measurements, with its dynamics expressed as

$$\dot{\beta}_\omega \sim N(0,\ \sigma). \tag{3.8}$$

3.1.3 Environment State

The resulting state vector for the simulator environment, including both the orbital and attitude dynamics, is then

$$x = \begin{bmatrix} r \in \mathbf{R}^3 \\ v \in \mathbf{R}^3 \\ q \in \mathbf{H} \\ \omega \in \mathbf{R}^3 \\ \beta_\omega \in \mathbf{R}^3 \end{bmatrix}. \tag{3.9}$$

3.2 Measurements

In addition to simulating the orbit of a satellite, the simulator also generates data for each of the three sensor types onboard the CubeSat, as well as for position.

3.2.1 Magnetometer

The position of the satellite is fed into the 13th generation International Geomagnetic Reference Field (IGRF13) [1] model to generate the magnetic field vector in the inertial frame B^I. The attitude of the satellite is then used to convert the magnetic field into the body frame, B^B. Because sensors are imperfect, the measured magnetic field vector is not the same as the true vector. This must be accounted for when generating measurements, so that the measured magnetic field vector in the body frame $\tilde{B}^B$ is

$$\tilde{B}^B = \eta_B T B^B + \beta_B, \tag{3.10}$$

where

$$T = \begin{bmatrix} a & 0 & 0 \\ b\sin\rho & b\cos\rho & 0 \\ c\sin\lambda & c\sin\phi\cos\lambda & c\cos\phi\cos\lambda \end{bmatrix} \tag{3.11}$$

converts a perfect magnetometer reading into an imperfect one in need of calibration and is formed using Eq. 2.10, 2.11, 2.12; β_B is the magnetometer bias; and η_B represents multiplicative Gaussian noise.

3.2.2 Gyroscope

The measured gyroscope vector $\tilde{\omega}$ is generated from the true gyroscope reading ω as

$$\tilde{\omega} = \omega + \beta_\omega + \eta_\omega, \tag{3.12}$$

with η_ω as white Gaussian noise.

3.2.3 Sun Sensors

The current generated from each photodiode is a function of the incoming light, the surface normal of the photodiode, and the scale factor for the photodiode. In addition to the light coming directly from the sun, there is also light coming from Earth's albedo; this additional light source depends on the location of the sun and the satellite, and its effect can represent up to $30\% - 40\%$ [23] of the experienced solar irradiance. To account for this, we model the effect of Earth's albedo after the method proposed by Bhanderi [4] using the method described in 2.2.

The total amount of current I produced by the jth diode can then be computed as

$$I_j = \hat{n}_j^T \hat{s}^B + \frac{E_{a,j}}{E_{AM0}}, \tag{3.13}$$

where $\hat{n}_j$ is the unit surface normal of the jth diode, $\hat{s}^B$ is the unit vector in the direction of the sun expressed in the body frame, E_{AM0} is the irradiance of sunlight at 1AU and with no loss due to atmosphere, and the Earth's albedo $E_{a,j}$ is a function of both satellite position and attitude [23]. To generate the measured current $\tilde{I}_j$, the true value is scaled by a scale factor C_j and noise is added, so that

$$\tilde{I}_j = C_j I_j + \eta_I \tag{3.14}$$

3.2.4 Position

The position of the satellite can either be determined using an on-board GPS or uploaded from a ground station. These are both treated as a type of measurement,

and as such are subject to noise, which we model as

$$\tilde{r} = r + \eta_r. \tag{3.15}$$

Chapter 4

Hardware

4.1 Satellite

For all testing and experiments, a 1U ($10 \times 10 \times 10 cm^3$) CubeSat was used, and is shown in Fig. 4.1. Each of the primary components is explained below.

4.1.1 PyCubed

The satellite hardware is built on PyCubed, an open-source complete avionics stack that has flight heritage on multiple CubeSat missions [13]. This avionics stack was designed for reliability and ease of use, providing functionality for all standard operating needs, including command and control (C&DH), energy harvesting and power management (EPS), telecommunication (TT&C), data collection and storage, and payload interfacing, as well as fail-safe deployment mechanisms [12]. Relevant hardware includes a power system (including solar panels, a power monitor, and a regulator), a BMX160 IMU, a 120MHz ARM M4F microcontroller, and a LoRa 1W 433MHz radio, as well as a mount for a GPS. A labeled image is shown in Fig. 4.2, and more information can be found in the related publication [13] or on the PyCubed website [12].

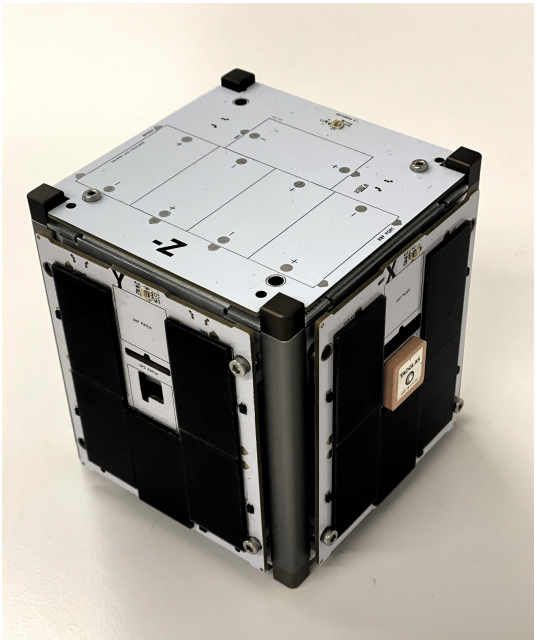

Figure 4.1: 1U CubeSat used for all testing and experiments in this document.

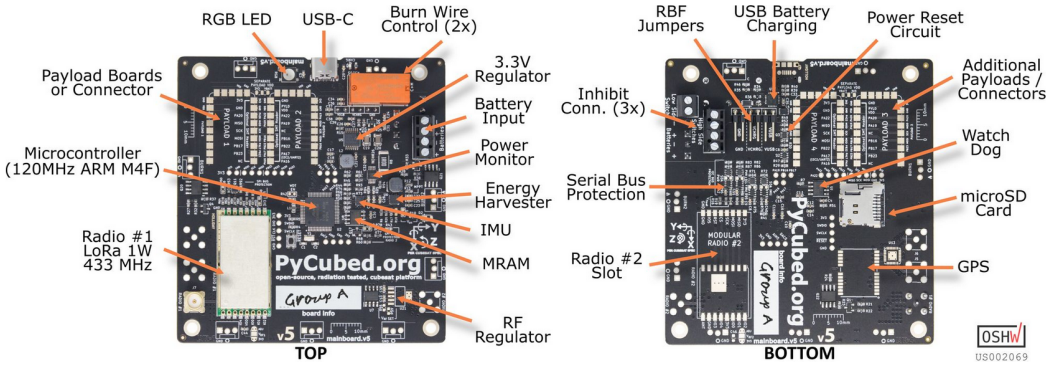

Figure 4.2: Labeled image of the PyCubed board, courtesy of the PyCubed website [12].

4.1.2 Sensors

The primary CubeSat sensors used in the attitude determination system are the gyroscope, magnetometer, and sun sensors. The gyroscope and magnetometer are both included in the on-board BMX160 IMU, which is only 2.5x3x0.95mm in size and only draws around 1.5mA of current when in high performance mode. This sensor also provides functionality for I^2C communication at up to 1MHz. There are six TSL2560 sun sensors on the satellite, with one on each panel. These sun sensors generate a current when illuminated that is proportional to the amount of light, with a maximum range of -1mA to 20mA of current; this current is passed to an ADC, resulting in a digital output that experiences very low noise.

4.1.3 Actuators

The CubeSat relies on magnetic torque coils for actuation, with one coil embedded on each of the outer panels of the spacecraft. When powered, the current running through these coils generates a magnetic field that attempts to align with an external field (in our case, the Earth's), providing torque to rotate the satellite into a desired attitude. Note that while many CubeSats utilize magnetic torque rods and/or coils in their attitude control system, they are often used in conjunction with another actuator, such as a reaction wheel. Exclusive reliance on magnetic torque coils limits the maximum amount of torque that can be produced at a given moment and results in an underactuated system, complicating the control problem. However, the resulting control system is extremely light weight and only requires low amounts of electricity to function, eliminating all moving parts and consumable fuel, resulting in a completely solid-state satellite that is less prone to breaking.

4.2 Hardware-in-the-Loop Testbed

Simulators help with conceptual and algorithmic development, but to truly validate the satellite hardware and detect errors before launch, a HITL testbed is necessary. We propose a unique testbed box that the 1U CubeSat can be placed into during testing. Each panel of the box contains a FLR-50T04-HW7 LED and a triple-axis

LIS3MDL magnetometer, which allows full control over the illumination of the satellite and enables us to measure the magnetic field generated by pulsing the magnetic torque coils. The magnetometers are connected to an Arduino Uno—which acts as the interface between the HITL testbed and the laptop—via a TCA9548A I^2C Multiplexer, which allows for easy communication. This box is placed inside of a Helmholtz coil to cancel out the geomagnetic field or replace it with a desired magnetic field.

This system allows for the testing of various sensors in isolation (e.g., running the LEDs to validate that the sun-vector estimation system works), or for the integration of the satellite hardware into a simulator (e.g., feeding the measured magnetic fields created by the magnetic torquer coils back into the simulator dynamics), allowing for the detection of faulty hardware or algorithms before launch.

Although our HITL setup does not allow for testing of the satellite gyroscope, which can be done in other HITL systems, the system as a whole is smaller and simpler. The box itself is made out of laser-cut wood (which can easily be replaced with acrylic or other on-hand materials) and the mounts are all made inside of a 3D printer. Additionally, the system is controlled with an Arduino and uses only off-the-shelf LEDs and magnetometers. The result is a cheap-to-make and easy-to-assemble test station that can be built in-house and requires only common materials that are easy to acquire.

4.2.1 Helmholtz Coils

In order to have control over the magnetic field that the testbed is experiencing, a Helmholtz coil was constructed. This device consists of two electromagnets that are aligned along the same axis, separated by a distance equal to their radii, and is used to generate a fairly consistent magnetic field in the region between the two rings. For our scenario, this provides the ability to generate a magnetic field opposite to that of the Earth's, effectively creating a magnetically neutral zone for testing.

The force of the magnetic field generated by a single loop on a particle x meters away is

$$B(x) = \frac{\mu_0 I R^2}{2(R^2 + x^2)^{3/2}}, \tag{4.1}$$

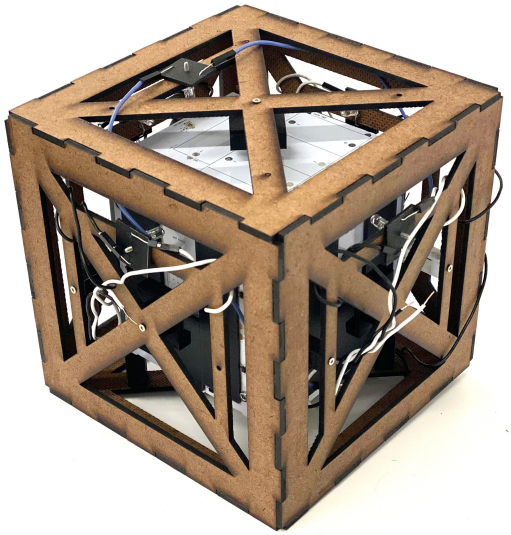

Figure 4.3: Assembled hardware-in-the-loop test box with 1U CubeSat inside.

where I is the current running through the wire, $\mu_0 = 4\pi \times 10^{-7} Tm/A$ is the permeability of free space, and R is the radius of the loop. By using *two* rings spaced Rm apart, two magnetic fields are created that combine to create a consistent magnetic field in the intersecting region, as shown in Fig. 4.4.

By placing a particle at the midpoint between the two rings so that $x = R/2$, the magnetic field becomes

$$B = \left(\frac{4}{5}\right)^{3/2} \frac{\mu_0 n I}{R}, \tag{4.2}$$

where n is the number of loops *per coil*.

System Requirements

Our Helmholtz coil was designed with three primary considerations in mind. First, in order to be able to create a magnetic field vector in any desired direction, the Helmholtz coil needs to be able to generate a field at least double that of the Earth's (in order to make an equally strong field in the opposite direction). Second, to prevent overheating and risk of damage, the total current must be kept relatively low. Finally, the testbed must be able to actually fit between the rings, limiting how small the setup can be. The target values for the device are shown in Table 4.1.

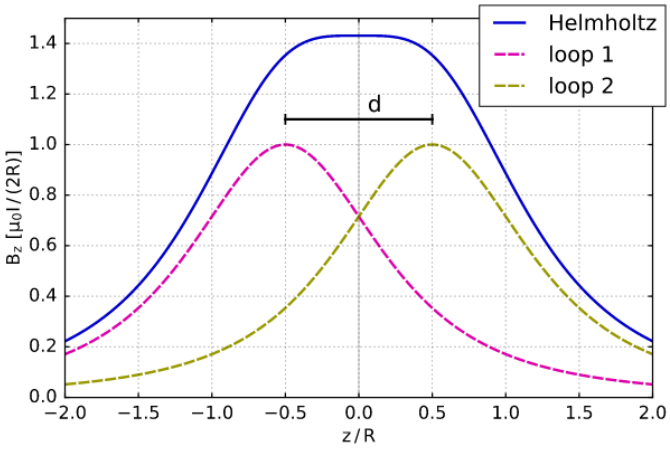

Figure 4.4: Plot of the magnetic field along the axis of a Helmholtz coil, as created by [11].

Mag Field Str.	Spacing	Current
$125\mu T$	$16cm$	$1mA$

Table 4.1: Minimum requirements for the Helmholtz coil.

Design and Construction

In order to maintain a low cost and easy assembly, the entirety of the Helmholtz coil structure was designed to be cut out of wood using a laser cutter. A radius of $R = 20$cm was chosen to provide enough room for the testbed to fit between, and $n = 300$ loops were wrapped using copper magnetic wiring. The system is shown in Fig. 4.5.

Figure 4.5: Helmholtz coil with the hardware-in-the-loop testbed placed in the center of the generated magnetic field.

Chapter 5

Flight Software

The ADCS software that will be flown on the CubeSat consists of a series of different steps. The satellite is first detumbled until the magnitude of the angular velocity is beneath some specified threshold. Next, the magnetometer is calibrated, and then the sun sensors. Once calibration has been done (which involves some estimation of the gyroscope bias), the satellite proceeds through another detumbling routine, this time with a lower threshold for magnitude of the angular velocity. Once this has been done, the satellite is ready to be pointed, and uses a standard MEKF for attitude estimation, using the calibration parameters determined previously.

The state machine illustrating state transitions and conditions is shown in Fig. 5.1, with additional information included below.

Detumbling

When a satellite is deployed, it often has a high angular velocity. This can cause problems communicating with a ground station, which prevents crucial mission updates from being received, and mission data from being transmitted. Additionally, the state estimation in the MEKF has some reliance on small angle approximations that do not hold for high spin rates.

However, the on-board gyroscope experiences a bias that makes it difficult to completely slow down the tumbling of the satellite, at least until the bias can be estimated. The proposed ADCS relies on two separate detumbling steps. The first

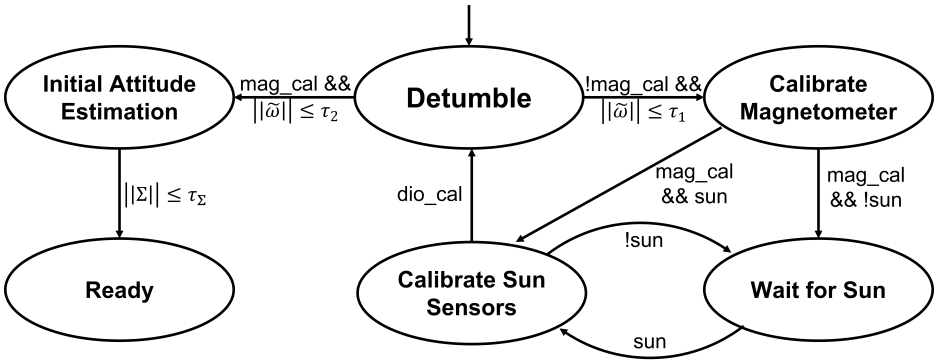

Figure 5.1: State machine illustrating sequence of phases for the ADCS. Transitions are determined using a series of flags, as well as the measured angular velocity $\tilde{\omega}$, the detumbling thresholds τ_1 and τ_2, the state covariance Σ, and the covariance threshold τ_Σ.

is a rough detumble that merely slows the angular velocity to a manageable rate, allowing for sensor calibration and radio communication. Once an estimate of the gyroscope bias is generated, it is subtracted out of the gyroscope measurement and the system is detumbled again.

Many methods exist for detumbling a satellite, include passive methods that slowly reduce spin rates over multiple orbits. We use the so-called "B-Cross" controller proposed by Avanzini [2], which uses the unit magnetic field vector expressed in the body frame $\hat{B}^B$ and the angular velocity ω of the CubeSat. These values are used to calculate a torque perpendicular to $\hat{B}^B$, so that

$$\tau = -\kappa \left(I_3 - \hat{B}^B (\hat{B}^B)^T \right) \omega, \tag{5.1}$$

where κ is the control gain and τ is the generated torque. This control law is linear and Lyapunov stable, and allows for detumbling from arbitrary initial angular velocities, provide the estimates of $\hat{B}^B$ and ω are close enough to the true values.

Magnetometer Calibration

Next, the satellite sensors are calibrated, as described in Chapter 2. First, the magnetometer is calibrated using batch estimation. The initial guess for the magnetometer parameters is provided using linear least squares, using magnetometer measured ($\tilde{B}^B$)

and predicted ($\bar{B}^B$) values gathered over two orbits, downsampled to one sample per minute to reduce space consumption. The equations are set up in typical linear fashion $Av = b$, where

$$v = \begin{bmatrix} a \\ b \\ c \\ \rho \\ \lambda \\ \phi \\ x_0 \\ y_0 \\ z_0 \end{bmatrix} \tag{5.2}$$

is the set of magnetometer calibration parameters to be determined. The vector v is formed by stacking the measured magnetometer readings, i.e.,

$$b = \begin{bmatrix} \tilde{B}^B_{x,1} \\ \tilde{B}^B_{y,1} \\ \tilde{B}^B_{z,1} \\ \vdots \\ \tilde{B}^B_{x,n} \\ \tilde{B}^B_{y,n} \\ \tilde{B}^B_{z,n} \end{bmatrix} \in \mathbf{R}^{3n}, \tag{5.3}$$

and the matrix A is formed using the predicted magnetometer readings:

$$A = \begin{bmatrix} \bar{B}^B_{x,1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \bar{B}^B_{x,1} & 0 & \bar{B}^B_{y,1} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \bar{B}^B_{x,1} & 0 & \bar{B}^B_{y,1} & \bar{B}^B_{z,1} & 0 & 0 & 1 \\ & & & \cdots\cdots\cdots\cdots\cdots\cdots & & & & & \\ \bar{B}^B_{x,n} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \bar{B}^B_{x,n} & 0 & \bar{B}^B_{y,n} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \bar{B}^B_{x,n} & 0 & \bar{B}^B_{y,n} & \bar{B}^B_{z,n} & 0 & 0 & 1 \end{bmatrix} \in \mathbf{R}^{3n\times 9}, \tag{5.4}$$

in accordance with magnetometer measurement Eqs. 2.10, 2.11, and 2.12 in Chapter 2. This initial guess for the parameters is then used in a Gauss-Newton root-finding algorithm to account for non-linearity in the system and refine the guess by minimizing the cost function in Eq. 2.13.

Sun Sensor Calibration

The sun sensors are calibrated using the method explained in Chapter 2, which involves augmenting the state of the MEKF with additional parameters. It is worth noting that parameterizing the surface normal of the sun sensors using azimuth α and elevation ϵ angles can be problematic, as the azimuth angle is undefined for $\epsilon = \pm\frac{\pi}{2}$, which correspond to the $\pm Z$ axes for our satellite. To avoid this problem, we simply rotate the reference frame about the Y axis by 45° so that no diodes have surface normals along that axis.

Multiplicative Extended Kalman Filter

Kalman filters are a type of recursive estimator that allow for state estimation in uncertain environments, combining state dynamics and measurement functions to calculate the best estimate for the current state. Although Kalman filters are designed for linear systems, they can be extended to work for our non-linear attitude determination problem with a few adjustments: the first involves linearizing the dynamics and measurement functions about the state estimate at each step, and the second is accounting for the multiplicative nature of attitude updates (as opposed to the additive updates used in a normal Kalman filter). The resulting filter is known as a multiplicative extended Kalman filter, and relies on two primary assumptions: one, that the dynamics and measurement functions are differentiable at every time step, and two, that the uncertainty is centered on the state estimate [16]. Note that these assumptions do not have to strictly hold for the MEKF to work well; this is fortunate, as the diode measurements for our system are not smooth when transitioning from illumination to darkness.

The satellite state consists of the satellite attitude and gyroscope bias, as well as

the calibration values for the sun sensors when they are being calibrated, so that

$$x = \begin{bmatrix} q \in \mathbf{H} \\ \beta_\omega \in \mathbf{R}^3 \\ C \in \mathbf{R}^N \\ \alpha \in \mathbf{R}^N \\ \epsilon \in \mathbf{R}^N \end{bmatrix} \tag{5.5}$$

for N sun sensors. One additional consideration to be made involves the choice of parameterization for the attitude. As previously stated, our system uses unit quaternions to represent satellite attitude due to their lack of singularities, low storage space, and easy-to-enforce constraints; however, this four-parameter representation for a three degree-of-freedom system creates problems when working with Jacobian matrices, resulting in rank-deficient matrices. To account for this, we rely on axis-angle representations of attitude ψ during each step, which represents attitude as a rotation of θ degrees around axis r and only requires three parameters. Axis-angle representations have singularities at 180°, but we do not expect to rotate that much between successive time steps, so that is not a concern. They are also ill-defined at 0° rotations, but that can easily be accounted for in code. The attitude Jacobian matrix (see Eq. 2.9) is used to convert between from a Jacobian using axis-angle vectors to one with quaternions.

Dynamics Function

The dynamics function updates the satellite attitude using the previous attitude estimate and the angular velocity. The gyroscope bias is assumed to be constant, and adjustments to the bias estimate are made later during the correction steps. First, the current estimate for the gyroscope bias is subtracted out of the measured angular velocity to estimate the true angular velocity

$$\omega_k = \tilde{\omega_k} - \beta_{\omega,k}, \tag{5.6}$$

which is then converted into an axis-angle representation and then into a quaternion to update the state, so that

$$r = \frac{\omega_k}{||\omega_k||} \tag{5.7}$$

$$\theta = ||\omega_k||\Delta t \tag{5.8}$$

$$q_k^- = q_{k-1}^+ \odot \begin{bmatrix} \cos\frac{\theta}{2} \\ r\sin\frac{\theta}{2} \end{bmatrix}, \tag{5.9}$$

where q_{k-1}^+ is the estimated attitude at the previous time step, the $\odot$ symbol represents quaternion multiplication, and q_k^- is the predicted attitude for the next time step *before* measurements have been accounted for. When calibrating the sun sensors, the calibration values are also assumed to be constant and remain the same. The dynamics function is then

$$x_k^- = f(x_{k-1}^+, \tilde{\omega_k}) = \begin{bmatrix} q_{k-1}^+ \odot \begin{bmatrix} \cos\frac{\theta}{2} \\ r\sin\frac{\theta}{2} \end{bmatrix} \\ \beta_{\omega,k-1} \\ C_{k-1} \\ \alpha_{k-1} \\ \epsilon_{k-1} \end{bmatrix}. \tag{5.10}$$

The Jacobian of the dynamics function is

$$F = \frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial\psi}{\partial\psi} & \frac{\partial\psi}{\partial\beta_\omega} \\ \frac{\partial\beta_\omega}{\partial\psi} & \frac{\partial\beta_\omega}{\partial\beta_\omega} \end{bmatrix}, \tag{5.11}$$

remembering that we are using a three-parameter attitude representation ψ when taking a Jacobian to avoid a rank-deficient matrix. For small angles, this Jacobian is approximately equal to

$$F_k = \begin{bmatrix} e^{-\left([\tilde{\omega}_k - \beta_{\omega,k}]^\times \Delta t\right)} & -\Delta t \\ 0_3 & I_3 \end{bmatrix}, \tag{5.12}$$

When calibrating sun sensors, $\frac{\partial C}{\partial C}$, $\frac{\partial\alpha}{\partial\alpha}$, and $\frac{\partial\epsilon}{\partial\epsilon}$ are all identity matrices, and the remaining terms are zero. Note the dependence on the small angle approximation,

which breaks down at high angular velocities or small time steps.

Measurement Functions

Our system has two measurement functions: one for the magnetometer and one for the sun sensors, denoted as h_m and h_s, respectively. The magnetometer measurement function uses the satellite's position r and the current time t to predict what the magnetic field vector in inertial frame B^I should be, utilizing the International Geomagnetic Reference Field 13 (IGRF13) model, and then uses the estimate of the predicted satellite attitude q_k^- to predict the magnetic field vector in body frame, which is later used to correct the predicted attitude:

$$\bar{B^B} = h_m(q_k^-, t, r) = Q(q_k^-)B^I(t, r). \tag{5.13}$$

Becuase the gyroscope bias does not affect this measurement function (nor do the sun sensor calibration parameters, when those are being estimated), the Jacobian of this measurement function is simply

$$H_m = \begin{bmatrix} [B^B]^\times & 0 \end{bmatrix}. \tag{5.14}$$

The sun sensor measurement is a bit more complicated: the sun vector in inertial frame is generated using the current time, and the position of the satellite is used to determine if the satellite is being eclipsed by the Earth (which would result in no expected current). In the event that the satellite is illuminated, the predicted attitude is used to calculate the sun vector in body frame, which is used with the estimated surface normals of each sun sensor to predict the sun sensor readings. The satellite and sun positions are also used to determine which portions of the Earth's surface are in their shared field-of-view, which is used to approximate the affect of Earth's albedo. The resulting function is

$$\bar{I}_j = h_{s,j}(q_k^-, t, r) = C_k^- \hat{n}_j^T Q(q_k^-)\hat{s}^N + C_k^- \frac{E_{a,j}(q_k^-, t, r)}{E_{AM0}} \tag{5.15}$$

where

$$\hat{n}_j = \begin{bmatrix} \cos\epsilon_{k,j}^- \cos\alpha_{k,j}^- \\ \cos\epsilon_{k,j}^- \sin\alpha_{k,j}^- \\ \sin\epsilon_{k,j}^- \end{bmatrix} \tag{5.16}$$

is the unit surface normal of the jth diode, $\hat{s}^B = Q(q_k^-)\hat{s}^N$ is the unit sun vector in body frame, $E_{AM0} = 1366.9$ is the irradiance of sunlight at 1AU, and $E_{a,j}$ is the affect of the Earth's albedo on the jth diode [23].

The Jacobian H_s is as follows. Note that the terms corresponding to the calibration parameters are derived by Springmann [23], but are included here for completeness.

$$\frac{\partial \bar{I}_j}{\partial \psi} = C_j \hat{n}_j^T [\hat{s}^B]^\times, \tag{5.17}$$

$$\frac{\partial \bar{I}_j}{\partial \beta} = 0_{1\times 3}, \tag{5.18}$$

$$\frac{\partial \bar{I}_j}{\partial C_j} = \hat{n}_j^T \hat{s^B}, \tag{5.19}$$

$$\frac{\partial \bar{I}_j}{\partial \alpha_j} = C_j \begin{bmatrix} -\cos\epsilon_j \sin\alpha_j & \cos\epsilon_j \cos\alpha_j & 0 \end{bmatrix} \hat{s}^B, \tag{5.20}$$

$$\frac{\partial \bar{I}_j}{\partial \epsilon_j} = C_j \begin{bmatrix} -\sin\epsilon_j \cos\alpha_j & -\sin\epsilon_j \sin\alpha_j & \cos\epsilon_j \end{bmatrix} \hat{s}^B. \tag{5.21}$$

When not calibrating the sun sensors, Eqs. 5.19, 5.20, and 5.21 are not used.

The matrix overall measurement Jacobian is then

$$H = \begin{bmatrix} H_m \\ H_s. \end{bmatrix} \tag{5.22}$$

Numerical Stability

To further improve the performance of our filter, we rely on a few techniques for improving numerical stability. A common problem for the covariance matrices used in an MEKF is that that magnitude of some values are many orders larger than others, so that some numbers get really large as others get relatively small. This can lead to dangerous round-off errors at each step when the covariance is corrected using

the measurement Jacobian and Kalman gain. However, this effect can be reduced by using an alternative form for the covariance correction step, known as Joseph form:

$$P_k^+ = (I - KH)P_k^-(I - KH)^T + KRK^T, \tag{5.23}$$

where P is the coviariance matrix, I is the identity matrix, K is the Kalman gain, H is the Jacobian of the measurement function evaluated at the current state estimate, and R is the measurement noise. This takes a bit more computation, but results in the addition of two positive semi-definite matrices, precluding an indefinite matrix result.

The second technique we rely on is using square-root filtering. This method reduces the effects of round-off error by storing the square root of a covariance matrix, rather than the actual covariance matrix. This requires a smaller range, allowing for shorter word lengths, and helps reduce the conditioning of a matrix; because of the symmetric nature of covariance matrices, only one side is needed. There are multiple ways to take the square root of a matrix, but for our filter we rely on the upper Cholesky factor C, so that

$$P = C^T C. \tag{5.24}$$

We can then utilize QR decomposition to further improve the filter, using the fact that

$$\sqrt{A + B} = qr_r(\sqrt{A}, \sqrt{B}), \tag{5.25}$$

where A and B are matrices and $qr_r(\cdot, \cdot)$ returns the R matrix from QR decomposition. This allows us to use the matrix square root C in all steps of the standard MEKF algorithm, without needing to convert back to the full covariance matrix P [25].

This also allows us to work with the square roots of our process noise Q and measurement noise R matrices, once again using Cholesky factorization, so that

$$\Gamma_Q = \sqrt{Q} \tag{5.26}$$

and

$$\Gamma_R = \sqrt{R}. \tag{5.27}$$

Algorithm 1 Procedure for a square-root multiplicative extended Kalman filter (MEKF), using Cholesky factorization, Joseph form, and QR decomposition to update the covariance matrix with higher numerical precision. Note the special update method in line 15 that results in the multiplicative nature of the filter.

1: **procedure** SQUARE-ROOT MEKF
2: C_1^+, x_1^+
3: **for** t = 2 to N **do**
4: (1) Prediction
5: $x_k^- = f(x_{k-1}^+, \tilde{\omega}_{k-1})$
6: $C_k^- = qr_r(C_{k-1}^+ F(x_{k-1}^+)^T, \Gamma_Q)$
7: (2) Innovation
8: $z_k = [h_m(q_k^-, t, r); h_s(q_k^-, t, r)]$
9: $G_k = qr_r(C_k^- H_k^T, \Gamma_R)$
10: $K_k = \left[G^{-1}(G^{-T}H_k)(A_k^-)^T A_k^-\right]^T$
11: (3) Update
12: $\Delta x = K_k z_k$
13: $\theta = ||\Delta x_\psi||$
14: $r = \frac{\Delta x_\psi}{\theta}$
15: $$x_k^+ = \begin{bmatrix} q_k^- \odot \left[\cos(\theta/2) \quad r\sin(\theta/2)\right]^T \\ \beta_k^- + \Delta x_\beta \\ C_k^- + \Delta x_C \\ \alpha_k^- + \Delta x_\alpha \\ \epsilon_k^- + \Delta x_\epsilon \end{bmatrix}$$
16: $C_k^+ = qr_r(A_k^-(I - K_k H_k)^T, \Gamma_R K_k^T)$
17: **end for**
18: **return** x_N^+, C_N^+
19: **end procedure**

Using our square-root notation, the covariance update in Eq. 5.23 becomes [25]

$$P_k^+ = qr_r\left(P_k^-(I - KH)', \Gamma_R L^T\right). \tag{5.28}$$

Algorithm

The complete algorithm for the square-root multiplicative extended Kalman filter is shown in Algorithm 1.

Scale Factor σ_C	**Azimuth Angle** σ_α	**Elevation Angle** σ_ϵ
0.15	4°	4°

Table 5.1: Initial error in estimates for sun sensor calibration parameters.

Scale Factor σ_s	**Non-Orthogonality Angle** σ_ζ	**Bias** σ_{β_B}
0.2	4°	$1\mu T$

Table 5.2: Initial error in estimates for magnetometer calibration parameters.

5.1 Experiments and Results

Simulations

The performance of the attitude determination system—as well as detumbling using the magnetic torque coils—was evaluated in simulation using Monte Carlo testing. Initial states for the satellite were randomly generated using the noise values shown in Table 5.3. The values for W and V used in the simulations were

$$W = \begin{bmatrix} (5\times10^{-4})I_3 & 0 & 0 & 0 & 0 \\ 0 & (5\times10^{-5})I_3 & 0 & 0 & 0 \\ 0 & 0 & (1\times10^{-5})I_N & 0 & 0 \\ 0 & 0 & 0 & (0.01°)I_N & 0 \\ 0 & 0 & 0 & 0 & (0.01°)I_N \end{bmatrix}^2 \tag{5.29}$$

and

$$V = \begin{bmatrix} (8°)I_3 & 0_3 \\ 0_3 & (0.01)I_3 \end{bmatrix}^2. \tag{5.30}$$

The initial error in sun sensor calibration parameters is shown in Table 5.1 and the error in magnetometer parameters in Table 5.2, where $s = [a, b, c]^T$ is the vector of scale factors along each axis, $\zeta = [\rho, \lambda, \phi]^T$ is the vector of non-orthogonality angles, and $\beta_B = [x_0, y_0, z_0]^T$ is the vector of biases along each axis of the magnetometer.

Twenty samples were averaged to evaluate the performance after 3 orbits, with

Magnetometer	Gyroscope	Gyroscope Bias	Position	Sun Sensor
$0.3\mu T$	$2.73 \times 10^{-4} \frac{rad}{s}$	$1.45 \times 10^{-5} \frac{rad}{s}$	$20km$	0.01

Table 5.3: Noise values used for the simulations, based off of the data from sensor specification sheets.

	Initial Error μ (σ)	Final Error μ (σ)	Improvement μ
Magnetic Field Vector	8.04° (3.24°)	0.26° (0.09°)	7.78°
Sun Vector	7.33° (1.8°)	2.85° (0.77°)	4.48°
Attitude	8.97° (4.23)	1.07° (0.52°)	7.9°

Table 5.4: Performance of the sensor calibration and attitude determination system, evaluated as the mean error over 30 simulated runs, each lasting for 3 orbits. The standard deviation is also provided. Note that the attitude error metric is the norm of the Cayley map.

the results shown in Table 5.4.

Example plots are shown for one simulated run, including a plot of the angular velocity during the detumbling phase (Fig. 5.2), a histogram of error in the magnetic field vector estimation before and after calibration (Fig. 5.3), the iterative guess on the calibration values for one sun sensor (Fig. 5.4), and the attitude error and gyroscope bias estimation (Fig. 5.5).

Hardware-in-the-Loop

In addition to testing performed in simulation, a hardware-in-the-loop test was also run to estimate the sun sensor calibration values on the CubeSat hardware. Our 1U CubeSat was placed inside our HITL testbed and hooked up to the simulator, which was used to generate unit sun vectors in the body frame. Diode measurements were gathered and stored, along with the true sun vector values for comparison. This data was used to estimate the sun sensors' scale factors, azimuth, and elevation angles, and then this updated information was used to estimate the sun vector for each of the saved time steps, along with the results for the uncalibrated system for comparison. The result is shown in Fig. 5.6.

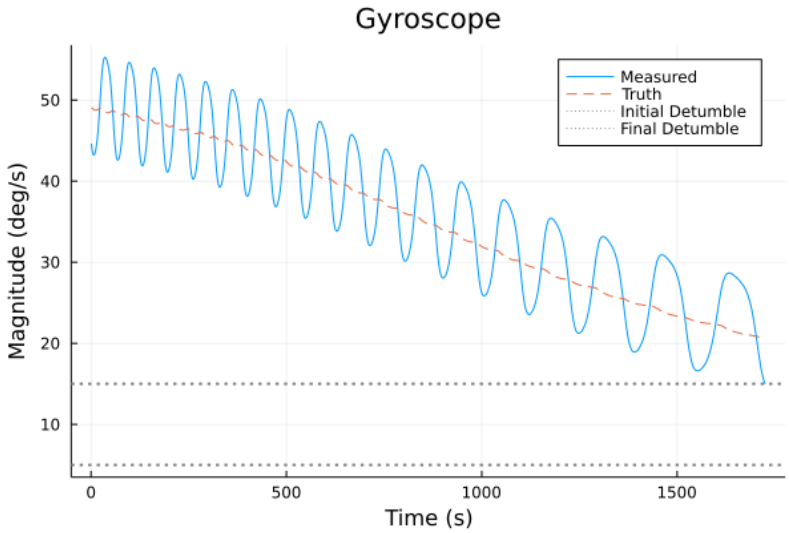

Figure 5.2: Example of simulated detumbling process.

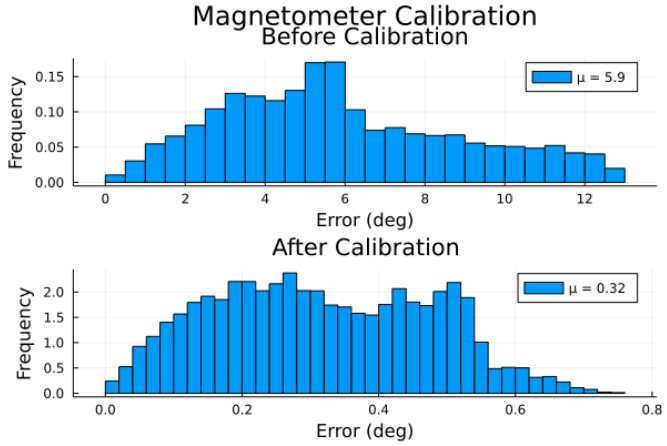

Figure 5.3: Histogram of the error in the magnetic field vector estimate in simulation before and after magnetometer calibration, in degrees.

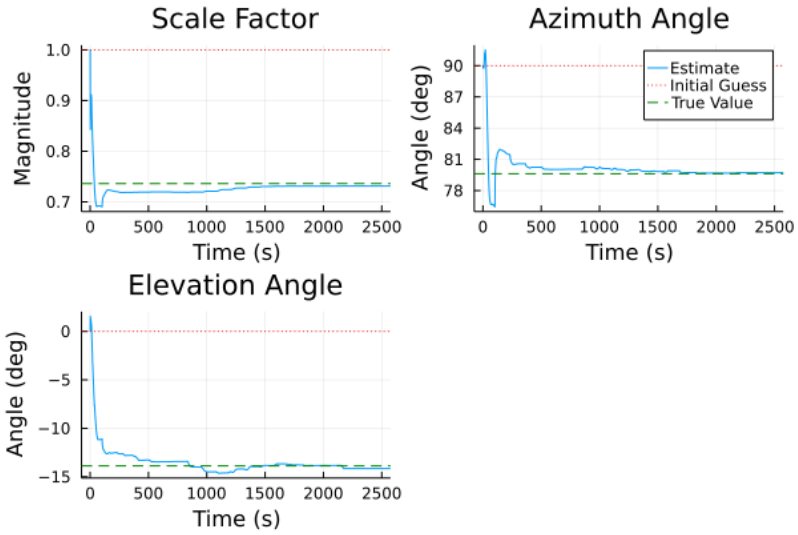

Figure 5.4: Calibration of a single sun sensor for a simulated orbit, including the initial guess, the true value, and the best estimate at each time step. Note that the time is measured since the start of the diode calibration phase.

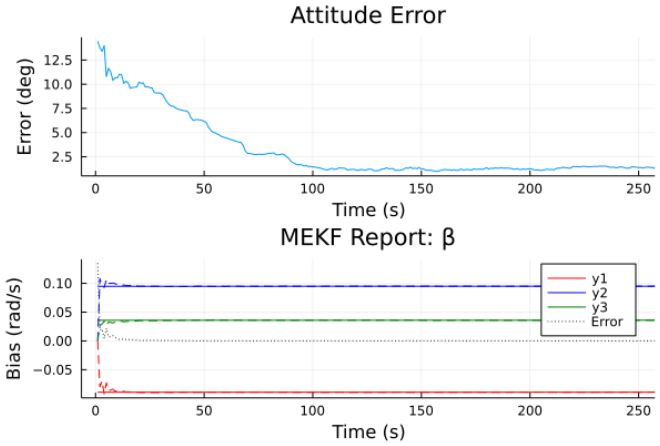

Figure 5.5: Error in estimated satellite attitude and gyroscope bias over time during a simulated orbit using our MEKF, where the attitude error is represented as the norm of the Cayley map. Note that the time is measured since the start of the diode calibration, which is attitude-dependent and includes attitude and gyroscope bias estimation.

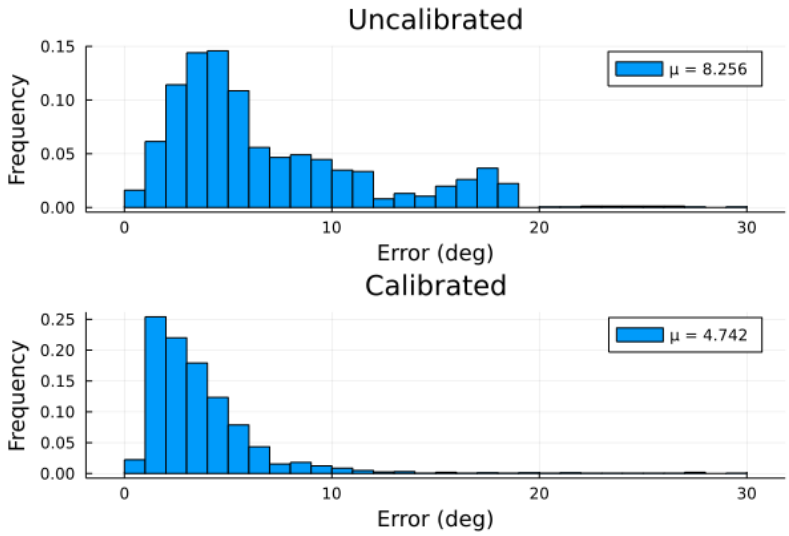

Figure 5.6: Error in sun vector estimation before and after calibration using the hardware-in-the-loop testbed. The mean error for the uncalibrated system was 8.3°, while the calibrated system had an error of only 4.7°.

5.2 Simultaneous Magnetometer and Sun Sensor Calibration

Thus far, we have used two separate methods for calibrating the magnetometer and the sun sensors: for magnetometers, we used an attitude-independent method relying on batch estimation, and for the sun sensors we augmented the state of our MEKF to use an attitude-*dependent* method for calibrating. As shown, this works well and provides good estimates for calibration parameters in simulation. However, batch estimation requires a large amount of data to be accumulated, which takes time and may not be possible due to limited computation and storage on satellites. This is particularly true for picosatellites or satellites relying on well-tested (old) computational equipment. Additionally, it relies on the assumption that the calibration parameters are not time-varying, which may not always be true, particularly for the magnetometer bias. We propose an alternative method that can be used to simultaneously calibrate the magnetometer and sun sensors by further augmenting the state in our MEKF to include the calibration parameters for both the sun sensors *and* the magnetometers. This allows for recursive calibration using

filtering and allows for time-varying parameters, at the cost of potentially lower accuracy in more extreme conditions.

Formulation

The augmented state of the MEKF becomes

$$x = \begin{bmatrix} q \in \mathbf{H} \\ \beta_\omega \in \mathbf{R}^3 \\ C \in \mathbf{R}^N \\ \alpha \in \mathbf{R}^N \\ \epsilon \in \mathbf{R}^N \\ s \in \mathbf{R}^3 \\ \zeta \in \mathbf{R}^3 \\ \beta_B \in \mathbf{R}^3 \end{bmatrix}, \tag{5.31}$$

where the magnetometer scale factors, non-orthogonality angles, and magnetometer biases have been appended to the end of the state used previously.

Dynamics

The dynamics function in Eq. 5.10 is almost the same and leaves the magnetometer calibration values untouched at each time step, so that

$$x_k^- = f(x_{k-1}^+, \tilde{\omega}) = \begin{bmatrix} q_{k-1}^+ \odot \begin{bmatrix} \cos\frac{\theta}{2} & r\sin\frac{\theta}{2} \end{bmatrix}^T \\ \beta_{\omega,k-1} \\ C_{k-1} \\ \alpha_{k-1} \\ \epsilon_{k-1} \\ s_{k-1} \\ \zeta_{k-1} \\ \beta_{B,k-1} \end{bmatrix}. \tag{5.32}$$

This results in the same dynamics Jacobian as before in Eq. 5.11, with all the new diagonal terms as identity and all the cross terms as zero:

$$F_k = \begin{bmatrix} e^{-\left([\tilde{\omega}_k - \beta_k]^\times \Delta t\right)} & -\Delta t & 0 \\ 0 & I_3 & 0 \\ 0 & 0 & I_{3N+9} \end{bmatrix}, \tag{5.33}$$

Measurements

The augmented state does not affect the sun sensor measurement function h_s, nor does it affect its Jacobian H_s other than to add additional zeros for each new parameter. However, the measurement Jacobian H_m becomes different; using Eq. 2.10, 2.11, and 2.12, the equations for the new parameters are

$$\frac{\partial B^B}{\partial s} = \begin{bmatrix} B_x^B & 0 & 0 \\ 0 & B_x^B \sin\rho + B_y^B \cos\rho & 0 \\ 0 & 0 & B_x^B \sin\lambda + B_y^B \cos\lambda \sin\phi + B_z^B \cos\lambda \cos\phi \end{bmatrix}, \tag{5.34}$$

$$\frac{\partial B^B}{\partial \zeta} = \begin{bmatrix} 0 & b(B_x^B \cos\rho - B_y^B \sin\rho) & 0 \\ 0 & 0 & c(B_x^B \cos\lambda - B_y^B \sin\lambda \sin\phi - B_z^B \sin\lambda \cos\phi) \\ 0 & 0 & c(B_y^B \cos\lambda \cos\phi - B_z^B \cos\lambda \sin\phi) \end{bmatrix}^T \tag{5.35}$$

$$\frac{\partial B^B}{\partial \beta_B} = I_3. \tag{5.36}$$

Note that Eq. 5.35 has been transposed in order to fit on the page.

Experiments

The simultaneous magnetometer and sun sensor algorithm was tested in simulation using the method explained above. Plots showing the estimated magnetometer scale factors s, non-orthogonality angles ζ, and biases β_B are shown in Figs. 5.7, 5.8, and 5.9, respectively.

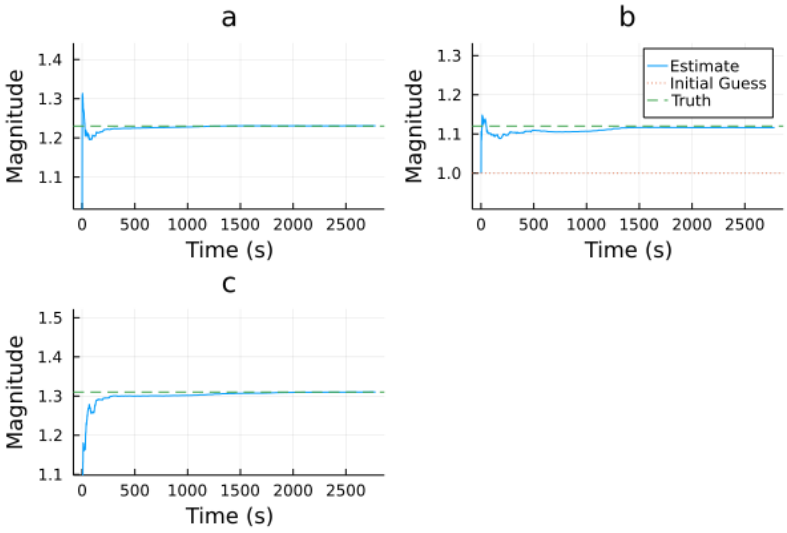

Figure 5.7: Example estimates of magnetometer scale factors $s = [a, b, c]^T$ versus time using an MEKF, as well as the true value and the initial guess.

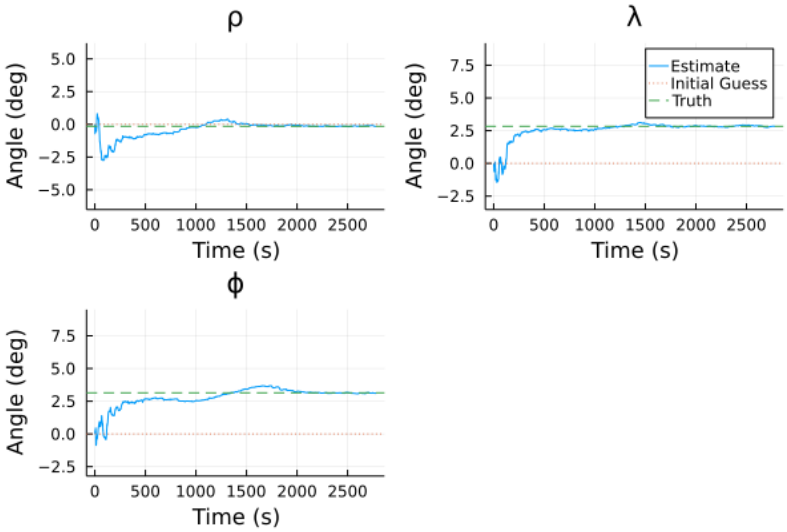

Figure 5.8: Example estimates of magnetometer scale factors $\zeta = [\rho, \lambda, \phi]$ versus time using an MEKF, as well as the true value and the initial guess.

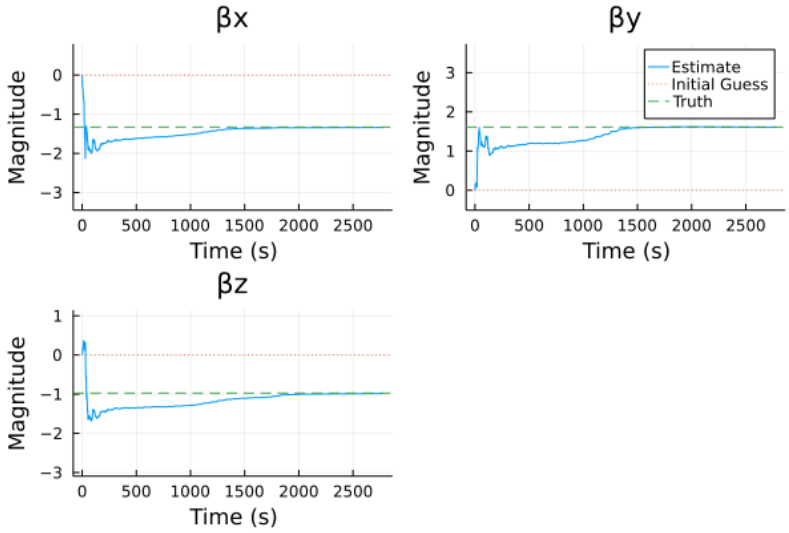

Figure 5.9: Example estimates of magnetometer scale factors $\beta_B = [\beta_{B,x}, \beta_{B,y}\beta_{B,z}]^T$ versus time using an MEKF, as well as the true value and the initial guess.

Chapter 6

Conclusion

In this work we develop a low-cost attitude determination and control system that is scalable to small satellites and uses sensors and actuators with low weight, cost, and volume requirements. This system calibrates on-board sensors, utilizes on a square-root multiplicative extended Kalman filter for attitude determination, and requires only magnetic torque coils for control, resulting in a completely solid-state satellite, with no moving parts or consumable fuel.

In order to further aid in the development of new algorithms for CubeSats, we also provide an open-source simulator that accounts for numerous factors that affect a satellite's orbit, as well as a hardware-in-the-loop testbed that allows for parts of the satellite hardware to be tested throughout the development cycle.

Future Work

The current hardware-in-the-loop testbed is inexpensive, easy to make, and quick to assemble; however, it has several limitations. Small satellites come in a variety of sizes (e.g., PocketCubes, 3U CubeSats, etc...); while the current design for the testbed could easily be modified to fit any of these types, a new version would have to be made for every new size of satellite, which is not ideal. The testbed design could be modified to allow for one structure that works for a variety of different sizes of satellites.

Additionally, while the current version of the Helmholtz coil is capable of generating

a magnetic field opposite to the Earths, creating a zone that is approximately magnetically neutral, it is only able to generate a magnetic vector in *one* direction. The hardware-in-the-loop testbed would be more effective if, like the sun vector, the magnetic field vector could be adjusted to any arbitrary direction at an instant. This would allow for simulation of the magnetic field over time as the satellite "orbits" the Earth. This is possible through construction of a three-axis Helmholtz coil with controllable currents.

Perhaps most importantly, the performance of the attitude determination and control system needs to be evaluated on an actual CubeSat in orbit. We intend to launch a 1U CubeSat in the near future that will run our system. The data will then be evaluated and compared to the results of our simulator and tests.

Bibliography

[1] P. Alken, E. Thébault, C.D. Beggan, and et al. International geomagnetic reference field: the thirteenth generation. *Earth Planets Space*, 74, 2021. doi: 10.1186/s40623-020-01288-x. 3.2.1

[2] Giulio Avanzini and Fabrizio Giulietti. Magnetic detumbling of a rigid spacecraft. *Journal of Guidance, Control, and Dynamics*, 35(4):1326–1334, 2012. doi: 10.2514/1.53074. URL https://doi.org/10.2514/1.53074. 5

[3] Dan Bhanderi. Earth albedo toolbox. https://www.mathworks.com/matlabcentral/fileexchange/11226-earth-albedo-toolbox, 2008. 2.2

[4] Dan D. V. Bhanderi. *Spacecraft Attitude Determination with Earth Albedo Corrected SunSensor Measurements*. PhD thesis, Aalborg University, Fredrik Bajers Vej 7, DK-9220 Aalborg Ø, Denmark, 2005. URL https://www.bhanderi.dk/research/publications/bhanderi_phd_thesis.pdf. 2.2, 3.2.3

[5] James W. Cutler, Aaron Ridley, and Andrew Nicholas. Cubesat investigating atmospheric density response to extreme driving (cadre). In *Proceedings of the 25th Annual AIAA/USU Conference on Small Satellites*, 2011. 1.2

[6] TOMS Science Team Goddard Earth Sciences Data and Information Services Center. Toms earth probe uv reflectivity, 1996. URL https://disc.gsfc.nasa.gov/datacollection/TOMSEPL3mref_008.html. (document), 2.1

[7] Nathanael England, James Cutler, and Srinagesh Sharma. Tandem beacon experiment - tbex. In *CubeSat Workshop, Michigan Exploration Lab*, 2018. 1.2

[8] C. C. Foster and G. H. Elkaim. Extension of a two-step calibration methodology to include nonorthogonal sensor axes. *IEEE Transactions on Aerospace and Electronic Systems*, 44(3):1070–1078, 2008. doi: 10.1109/TAES.2008.4655364. 2.3.1, 2.3.1, 2.3.2

[9] F.Reichel, P.Bangert, S.Busch, K.Ravandoor, and K.Schilling. The attitude determination and control system of the picosatellite uwe-3. *IFAC Symposium on Automatic Control in Aerospace*, 46(19):271–276, 2013. doi: 10.3182/20130902-5-DE-2040.00088. 1.2

[10] Andrew Gatherer and Zac Manchester. Magnetorquer-only attitude control of small satellites using trajectory optimization. In *Proceedings of AAS/AIAA Astrodynamics Specialist Conference*, August 2019. 1.1

[11] Geek3. https://commons.wikimedia.org/wiki/File:Mplwp_Helmholtz_coil_field.svg, December 2019. (document), 4.4

[12] Max Holliday. Pycubed, 2022. URL https://pycubed.org/. (document), 4.1.1, 4.2

[13] Max Holliday, Andrea Ramirez, Connor Settle, Tane Tatum, Debbie Senesky, and Zac Manchester. Pycubed: An open-source, radiation-tested cubesat platform programmable entirely in python. In *AIAA/USU Conference on Small Satellites (SmallSat)*, 2019. 1.1, 4.1.1

[14] Brian E. Jackson, Kevin Tracy, and Zachary Manchester. Planning with attitude. *IEEE Robotics and Automation Letters*, 6(3):5658–5664, 2021. doi: 10.1109/LRA.2021.3052431. 2.1, 3.1.2

[15] Benjamin Jensen. Earth albedo. https://github.com/RoboticExplorationLab/EarthAlbedo.jl, 2021. 2.2

[16] Tucker McClure. How kalman filters work, 2022. URL https://anuncommonlab.com/articles/how-kalman-filters-work/. 5

[17] R. McPeters, P. K. Bhartia, A. J. Krueger, , and J. R. Herman. Earth probe total ozone mapping spectrometer (toms) data products user's guide. *Technical Report No. 19987-206895, National Aeronautics and Space Administration*, 1998. 2.2

[18] O. Montenbruck and E. Gill. *Satellite Orbits: Models, Methods and Applications*. Springer, 2012. 3.1.1, 3.1.1, 3.1.1

[19] J. Prado, G. Bisiacchi, L. Reyes, E. Vicente, F. Contreras, M. Mesinas, , and A. Juárez. Three-axis air-bearing based platform for small satellite attitude determination and control simulation. *Journal of Applied Research and Technology*, 3(3):222–237, 2005. 1.2

[20] Darren Rowen and Rick Dolphus. 3-axis attitude determination and control of the aerocube-4 cubesats. In *Proceedings of the Small Satellite Conference, CubeSat Developers' Workshop*, 2013. 1.2

[21] M.D. Shuster and D.S. Pitone. Batch estimation of spacecraft sensor alignments ii. absolute alignment estimation. *Journal of Astronautical Sciences*, 39(4): 547–571, 1991. 2.3.2

[22] John C. Springmann and James W. Cutler. Attitude-independent magnetometer calibration with time-varying bias. *Journal of Guidance Control and Dynamics*, 2012. URL https://arc.aiaa.org/doi/pdf/10.2514/1.56726. (document),

2.3.1, 2.2, 2.3.1

[23] John C. Springmann and James W. Cutler. On-orbit calibration of photodiodes for attitude determination. *Journal of Guidance Control and Dynamics*, 37:1808–1823, 2014. URL https://deepblue.lib.umich.edu/bitstream/handle/2027.42/140645/1.g000175.pdf?sequence=1. 2.3.2, 3.2.3, 3.2.3, 5

[24] Michael Swartwout. Cubesat mission success: Are we getting better? In *CubeSat Developers' Workshop*, 2019. 1

[25] Kevin S. Tracy. A square-root kalman filter using only qr decompositions. *ArXiv*, 2022. URL arxiv.org/abs/2208.06452. 5, 5