CENTIMETER-SCALE SPACECRAFT: DESIGN, FABRICATION, AND DEPLOYMENT

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Zachary Robert Manchester

August 2015

CENTIMETER-SCALE SPACECRAFT: DESIGN, FABRICATION, AND DEPLOYMENT

Zachary Robert Manchester, Ph.D.

Cornell University 2015

The rise of the consumer electronics industry has led to a profusion of ever smaller, cheaper, and more capable devices, from sensors to computers to radios. Many of these components are suitable for use in spacecraft. Their availability has led to the ongoing revolution in small satellites, most visibly exemplified by the CubeSat standard. This study seeks to push these trends toward their ultimate extreme: a satellite-on-a-chip.

The ability to mass produce small, cheap, essentially disposable spacecraft has many implications for space exploration, Earth and atmospheric science, and education. New missions to characterize planetary surfaces and atmospheres, asteroids, and Earth's ionosphere could be planned using thousands of tiny spacecraft equipped with a variety of sensors. Such mission architectures would allow thousands of data points to be captured simultaneously across vast distances, offering unprecedented spatial resolution. Additionally, the cost of launching a satellite will be within reach for high schools, student groups, and even individual hobbyists, making space accessible to the general public in new ways.

This dissertation documents the development of the Sprite centimeter-scale spacecraft, along with the associated hardware and software necessary for its deployment in low-Earth orbit and communication with ground stations. In addition, new solutions to several general spacecraft dynamics, control, and estimation problems with relevance to small and low-cost spacecraft are presented. These include

control laws for flat-spin recovery and spin inversion, an algorithm for on-orbit inertia estimation, and variational integrators for spacecraft attitude dynamics that offer improved performance over traditional Runge-Kutta schemes in spacecraft guidance, navigation, and control applications.

BIOGRAPHICAL SKETCH

Zac Manchester was born in Chicago but spent most of his childhood in Asia, returning to the U.S. in high school. Growing up, he had a long-running fascination with aviation and spaceflight. After experimenting with remote control aircraft, model rocketry, and flying lessons, Zac decided to attend engineering school at Cornell.

Zac earned his bachelor's degree in engineering physics in 2009. During his undergraduate years he rowed varsity crew for a year, played bass in the Cornell jazz ensembles, developed an autopilot for a UAV, and also worked part time for aerospace software maker Analytical Graphics, Inc (AGI). It was at AGI that Zac first became acquainted with the subject of astrodynamics and decided that he would pursue a Ph.D. in aerospace engineering.

Shortly after beginning graduate school at Cornell in 2010, Zac founded the KickSat project as an outgrowth of his Ph.D. research. The project has the goal of dramatically reducing the cost of access to space and has included participants from all over the world. KickSat was financed through the crowd-funding website Kickstarter and awarded a launch through NASA's Educational Launch of Nanosatellites (ELaNa) Program.

From 2012 to 2014, Zac worked at NASA Ames Research Center. In addition to his own research, he developed attitude determination and control algorithms for use on several NASA spacecraft. He also worked in the Ames SpaceShop experimenting with rapid prototyping equipment (3D printers, laser cutters, and desktop CNC machines) to build spacecraft components.

Zac looks forward to continuing his research in small spacecraft and spacecraft dynamics and control as a postdoc and is pursuing an academic career in which he can continue to advance space exploration and access to space.

To the people of Earth, especially Becky.

ACKNOWLEDGEMENTS

I owe thanks to many people for helping and guiding me along the way to this dissertation. First and foremost, my family has been a constant source of support and encouragement throughout my education. If not for my mother, I would almost certainly never have made it to Cornell in the first place as an eager and somewhat-cocky eighteen-year-old. My wife Becky has stuck with me through long hours, many late nights in the lab, and multiple cross-country moves, and my three siblings and my wife's family have also been there for me on numerous occasions during my time in graduate school.

I have had the privilege of meeting and working with many inspiring people at Cornell. My adviser, Mason Peck, has listened to too many of my crazy ideas to count and always seems to have something insightful to add. The other members of my special committee, Alex Vladimirsky, Brandon Hencey, and Mark Campbell, have all offered ideas, technical advice, and career guidance at various points. In addition to my committee, I'd also like to thank Liran Gazit and Rob McCurdy, who both played critical roles in some of the early work on the Sprite, Hadas Kress-Gazit for her academic career advice, and Marcia Sawyer and Pati Wojcik for keeping things running on a daily basis.

Much of the work documented in the following chapters was done during the nearly two years I spent at NASA Ames Research Center. Many people there were supportive of my research and were generous with their time and expertise, including Andy Filo, Matt Reyes, Pete Worden, John Hines, Harry Partridge, Elwood Agasid, and Chad Frost. Lastly, I must extend a huge thanks to the entire PhoneSat team, especially Oriol Tintore-Gazulla, Jasper Wolfe, Cedric Priscal, Alberto Guillen-Salas, and Ken Oyadomari, who provided the avionics for KickSat.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation

The rapid miniaturization of commercial-off-the-shelf (COTS) electronics, driven in recent years by the emergence of smart phones, has made many of the components used in spacecraft available in very small, low-cost, low-power packages. This trend has inspired the "ChipSat" concept[1]–[3], the idea of building chip-scale satellites with the same devices and processes used in the consumer electronics industry. The ability to mass produce such devices, along with their small size, leads to the realistic near-term possibility of sub-thousand-dollar-per-satellite missions for scientific, educational, and hobbyist use.

By dramatically reducing the cost and complexity of building and launching spacecraft, ChipSats could help expand access to space. In the near future, it will be possible for a high school science class, amateur radio club, or motivated hobbyist to choose sensors, assemble a ChipSat, configure a ground station, and fly their own satellite mission. A new class of science missions could also be built around large ensembles or "swarms" of ChipSats equipped with a range of electromagnetic, micro-electro-mechanical (MEMS), and nanofluidic sensors. Such swarms could perform, for example, large-scale in-situ surveys of planetary atmospheres[4] or determine the composition of asteroids[5], allowing thousands of data points to be collected simultaneously over large spatial volumes. Earth's ionosphere, in particular, could be studied in ways that are impractical or impossible with current sounding rocket and multi-satellite missions[6], [7].

In addition to providing a new and unique set of capabilities, building missions around large numbers of ChipSats will also provide a high degree of robustness. Traditional spacecraft are typically built with redundant subsystems and high-reliability components to achieve robustness, often leading to increased cost, complexity, and spacecraft mass. In contrast, a ChipSat mission could achieve a similar or even greater probability of success by simply deploying sufficiently many spacecraft. Due to their individual simplicity and mass-producibility, the marginal cost of producing additional ChipSats is very low. Even with relatively high rates of individual failure, a mission utilizing hundreds or thousands of ChipSats would have a very high likelihood of overall success.

1.2 Contributions

This dissertation describes a complete architecture for low-cost missions based on centimeter-scale spacecraft. In addition, solutions to several general spacecraft dynamics and control problems with relevance to small satellites are developed. The major contributions are:

1. The design and implementation of a complete spacecraft on a printed circuit board.
2. A long-range low-power communication system for centimeter-scale spacecraft scalable to hundreds of satellites.
3. A CubeSat-based deployment system for centimeter-scale spacecraft.
4. A novel algorithm for recursive estimation of spacecraft inertia based on semidefinite programming.

5. Feedback control laws capable of both flat-spin recovery and spin inversion of spinning spacecraft.
6. The development of variational integrators for spacecraft attitude dynamics with quaternion state variables.

1.3 Outline

Each chapter in this dissertation corresponds roughly to a journal publication. Chapter 2 describes the design, fabrication, and testing of the Sprite spacecraft, a 3.5-by-3.5 centimeter printed circuit board satellite. It also details the communication architecture developed to allow many Sprites to simultaneously communicate with a single ground station from low-Earth orbit.

Chapter 3 discusses the KickSat project – an effort to launch and deploy over one hundred Sprite spacecraft in low-Earth orbit from a CubeSat "mothership." The first KickSat mission, KickSat-1, was launched in April 2014. The design and testing of the KickSat-1 spacecraft are discussed, as well as the outcomes of the mission.

Chapter 4 introduces a recursive algorithm for on-orbit estimation of spacecraft inertia. Its development was motivated by the need for inertia knowledge in the control system on the KickSat spacecraft along with the relatively high cost of traditional methods for accurate ground-based inertia measurements. The algorithm makes use of semidefinite programming to guarantee that a physically valid inertia matrix is always returned.

Chapter 5 develops a family of nonlinear control laws capable of reorienting

a tumbling spacecraft so that it spins about its minor axis of inertia in a desired direction. This problem was also motivated by the KickSat mission, and is a generalization of the classic flat-spin recovery problem. The controllers are derived from a Lyapunov function and shown to be almost-globally asymptotically stabilizing.

Chapter 6 develops variational integrators for spacecraft attitude dynamics parameterized with quaternions. These integrators offer many advantages over Runge-Kutta schemes, including energy and momentum conservation. They are also very well suited for use in spacecraft guidance, navigation, and control algorithms, such as attitude determination filters.

CHAPTER 2

THE SPRITE SPACECRAFT

This chapter describes the Sprite ChipSat, along with its associated deployment, communication, and ground station systems. It begins with a survey of existing work in section 2.1. Section 2.2 then describes the hardware design of both the Sprite and its deployer, which is compatible with the CubeSat specification. Section 2.3 describes the code division multiple access (CDMA) communication architecture employed by the Sprite. Finally, section 2.5 summarizes the results of an on-orbit test of prototype Sprite hardware.

2.1 Existing Work

Previous studies have advocated the mass production of silicon-wafer-based spacecraft[1] and satellite-on-chip devices[2], [3]. There have also been several investigations of the dynamics of centimeter-scale spacecraft and the potential for creating "swarms" or constellations of such spacecraft: Atchison and Peck studied the dynamics of very small spacecraft subject to a variety of environmental perturbation forces[2], [4], [8], and Colombo and McInnes designed orbits that balance solar pressure and atmospheric drag effects to produce long-lived ChipSat swarms[9].

The fist laboratory prototype of a small planar spacecraft constructed using COTS components was built by Barnhart et. al.[3], [10]. Their "PCBSat," with dimensions of ten-by-ten-by-two centimeters and a mass of 200 grams, included solar cells, a battery, a microcontroller, a radio, a GPS receiver, and a variety of sensors. A serious deficiency in their design, however, is that the radio and communication protocols used provide a maximum range of 1.3 kilometers, mak-

ing communication feasible only with other (very) nearby spacecraft, and direct communication with ground stations impossible.

2.2 Hardware Design

The hardware design of the Sprite, rather than being driven from the "top down" by mission requirements, as is typical of most spacecraft, is driven by a "bottom up" desire to build the smallest-possible functional spacecraft using standard commercial components and manufacturing technologies. The Sprite makes use of modern, low-cost, low-power integrated circuits to create a general purpose "spacecraft bus" for chip-scale sensors. It includes Spectrolab TASC solar cells, a Texas Instruments CC430 microcontroller and radio system-on-chip (SoC), a Honeywell HMC5883L three-axis magnetometer, and an InvenSense ITG-3200 three-axis MEMS gyro, as well as associated passive components, on a printed circuit board measuring 3.5 by 3.5 centimeters with a mass of four grams (figure 2.1).

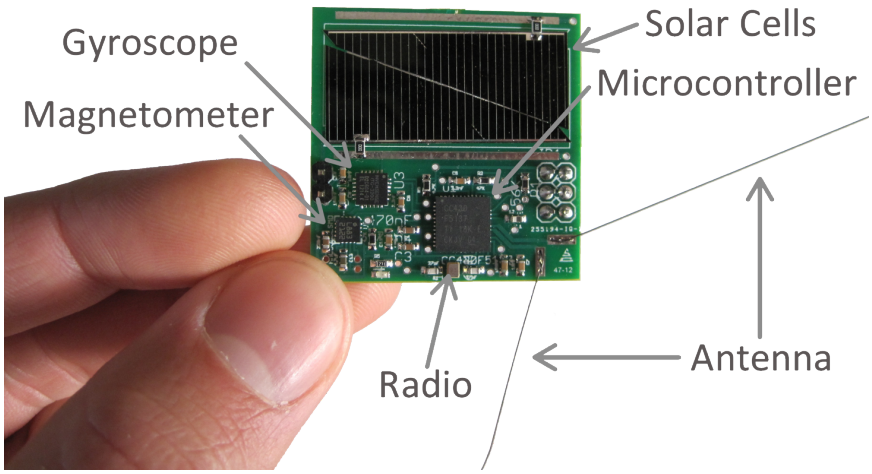


Figure 2.1: The Sprite Spacecraft

The overall dimensions of the Sprite are determined primarily by the peak

power required by the radio while transmitting. Figure 2.2 shows the expected temperature range of a Sprite in low-Earth orbit. The very cold temperatures

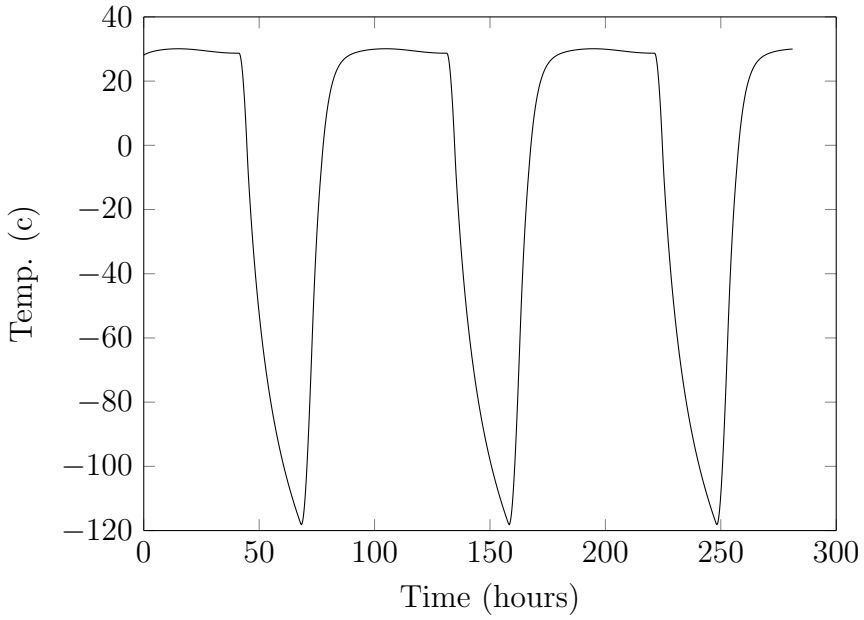


Figure 2.2: Thermal simulation of a Sprite in low-Earth Orbit

encountered during eclipse cause modern batteries based on lithium-ion or lithium-polymer chemistries to fail permanently. As a result, all power must be delivered directly by solar cells, leaving Sprites completely unpowered while above the night side of the Earth.

The TASC solar cells used on the Sprite are a triple-junction gallium-arsenide design with an open-circuit voltage of 2.2V and a nominal conversion efficiency of 27%[11]. All of the integrated circuits chosen for the Sprite are capable of operating at a supply voltage of 2.2V or less, eliminating the need for additional power conditioning or regulation circuitry. The maximum current required by the radio during transmission is approximately 35 mA. A pair of TASC solar cells delivers 60 mA of current when aligned directly with the sun vector, offering considerable margin to account for attitude variations.

The CC430 SoC provides the core computing and communication capabilities of the Sprite. It combines a 16-bit MSP430 microcontroller, which is clocked at 8 Mhz and provides 4 kB of RAM and 32 kB of flash memory, with a very flexible CC1101 UHF transceiver capable of an output power of 10 mW and raw bit rates up to 500 kbps. The 70 cm amateur band was chosen as a compromise between antenna size and transmitter efficiency. The 2.4 GHz S-band was also considered, which would allow a smaller antenna. However, the efficiency of available COTS transmitter ICs is an order of magnitude worse at 2.4 GHz than at 437 MHz. Both the MSP430 and CC1101 have flight heritage on previous CubeSat missions[12], [13].

The Sprite's antenna is a half-wave V-dipole, chosen for its isotropic gain pattern, easy tuning, and 50 ohm characteristic impedance, which eliminates the need for a matching network or balun circuit. The antenna is made of superelastic nitinol, a nickel-titanium alloy commonly referred to as "shape-memory alloy" or "memory metal," which can undergo very large strain and still return to its original shape. Nitinol was chosen so that the antenna could be tightly coiled for launch and return to its intended geometry after deployment.

2.3 Closing The Link

Perhaps the most difficult engineering challenge associated with the Sprite is closing the communication link from low-Earth orbit to ground stations. The Sprite's transmitter is limited to 10 mW of output power. A lack of closed-loop attitude control means a low-gain antenna with an omnidirectional gain pattern is required. Additionally, due to licensing constraints, the Sprites must efficiently share limited

RF bandwidth. Closing link over several hundred kilometers with all of these constraints is a formidable challenge, but thanks to recent advances in software-defined radio technology, it is possible with relatively inexpensive hardware.

As motivation, the signal-to-noise ratio (S/N) is first derived for a nominal set of orbit and receiver parameters. With a transmitter power $P_t = 10$ dBm, a transmitter antenna gain of $G_t = 0$ dB, receiver antenna gain $G_r = 10$ dB, polarization loss of $L_p = 3$ dB, atmospheric loss of $L_a = 2$ dB, range $R = 650$ km, and wavelength $\lambda = 70$ cm, the received power $P_r \approx -127$ dBm is given by the Friis equation[14]:

$$P_r = P_t + G_t + G_r - L_p - L_a + 20 \log_{10} \left(\frac{\lambda}{4\pi R} \right) \tag{2.1}$$

The noise power in the receiver $P_n \approx -122$ dBm is given by,

$$P_n = F_n + 10 \log_{10} (K_B T B) \tag{2.2}$$

where $F_n = 4$ dB is the receiver's noise figure, K_B is Boltzmann's constant, $T = 290$ K is the noise temperature in degrees Kelvin, and $B = 64$ kHz is the receiver bandwidth[14]. Subtracting the noise power from the signal power gives a signal-to-noise ratio of $S/N = -10$ dB. Table 2.1 summarizes the link budget parameters for a Sprite in a nominal low-Earth orbit.

It is worth pausing here to consider the implications of the previous result: A negative S/N means, in a power sense, that there is more noise than signal. If one were to connect an oscilloscope between the antenna and receiver, the observed signal would be nearly indistinguishable from white noise. In a more typical situation, a design engineer might use a more powerful transmitter or higher gain antenna, but in the case of the Sprite, neither of those solutions are possible.

Table 2.1: Link budget summary

Altitude	350 km
Minimum Elevation Angle	30°
Range (R)	652.5 km
Frequency	437 MHz
Maximum Doppler Shift	9.7 kHz
Transmitter Power (P_t)	10.0 dBm
Transmitter Antenna Gain (G_t)	0.0 dB
Free-Space Loss (L_{fs})	-141.6 dB
Atmospheric Attenuation (L_a)	-2.0 dB
Receiver Antenna Gain (G_r)	10.0 dB
Polarization Loss (L_p)	-3.0 dB
Receiver Bandwidth (B)	64 kHz
Receiver Noise Temperature (T_n)	290 K
Receiver Noise Figure (F_n)	4.0 dB
Receiver Noise Power (P_n)	-121.9 dBm
Received Signal Power (P_r)	-126.6 dBm
Code Gain (G_c)	27.1 dB
S/N	-10.3 dB
E_b/N_0	16.8 dB

2.3.1 Communication Background

The Sprite communication system makes use of two techniques, matched filtering and forward error correction (FEC), which have long histories in space and terrestrial communication, navigation, and radar systems, but have not yet seen widespread use within the small satellite community. This section will provide a brief overview of these techniques for an aerospace engineering audience with limited signal processing experience. More efficient utilization of communication bandwidth with techniques like these would enable far greater data throughput from small satellite missions.

To overcome low S/N, matched filtering, which is the optimal linear filter for

maximizing signal-to-noise ratio[15], is used. The basic idea is to substitute each data bit with a long specially-chosen string of bits known as a pseudo-random number (PRN) code that is agreed upon by both the transmitter and receiver *a priori*. In this context, the bits making up the PRN code are commonly referred to as "chips" to differentiate them from the data bits. Rather than attempting to lock onto the carrier and demodulate the chips individually, the receiver instead looks for the entire PRN code by cross-correlating the incoming signal against the known code at each time step.

Cross-correlation in the discrete-time setting can be interpreted as a "sliding inner product," as shown in equation (2.3), where x is the cross-correlation, p is the sampled PRN code vector of length N, and s is a vector of signal samples. At each time step k, the signal vector is shifted one sample, with the oldest element being removed from one end and a new sample being added at the opposite end. A new inner product is then calculated. If the PRN is present, the correlation magnitude will be large, even in the presence of substantial noise, while if no code is present, the correlation will be small.

$$x_k = p^\dagger \begin{bmatrix} s_{k-N} \\ \vdots \\ s_k \end{bmatrix} \tag{2.3}$$

Matched filtering essentially allows the energy in the entire PRN code to be summed, providing a "code gain" G_c equal to the length of the PRN. The trade-off, however, is that the data rate is also lower by the same factor. PRN codes with a length of 511 chips are used on the Sprite, providing a code gain $G_c = 10 \cdot \log_{10}(511)$ of about 27 dB for a very robust link margin. When adding G_c to the signal-to-noise ratio, the resulting dimensionless quantity is known as E_b/N_0 (pronounced "energy-per-bit over noise-power-spectral-density"). E_b/N_0 is the standard figure

of merit used to characterize digital communication links.

Aside from adding code gain, matched filtering also makes it possible for multiple users to share a communication channel through what is known as code division multiple access (CDMA). By assigning each Sprite a different PRN, the receiver can "tune" to a particular spacecraft's signal by correlating against its unique code. This allows all the Sprites on a particular mission to share the same allocated frequency, simplifying licensing and eliminating the need for clock synchronization that would otherwise be required for the Sprites to alternate transmitting on the same frequency. CDMA has been used for many years in the cellular telephone industry[15] and the global positioning system[16], and has proven, in practice, to be the most efficient channel access method when a large number of users must be accommodated[15].

To successfully implement CDMA, the family of PRN codes assigned to the group of Sprites must be carefully chosen to minimize interference. They should be as orthogonal as possible in the sense that their cross-correlations should be close to zero. Unfortunately, it is mathematically impossible to generate code families with perfect zero cross-correlations for all time offsets[17]. Many code families exist, however, with low and bounded cross-correlations, notably the Gold codes[18] used in GPS[16]. A family of Gold codes of length 511 bits has been selected for use with Sprites, offering good cross-correlation performance and allowing up to 511 unique codes to be assigned to individual spacecraft.

Due to the finite cross-correlations between Gold codes, large numbers of Sprites cannot transmit simultaneously without causing unacceptable levels of interference. To overcome this problem, the Sprites operate their radios on a duty cycle with randomized sleep and wake times. Because excessive interference may still occur

occasionally, forward error correction is applied to further improve robustness. FEC is widely used in modern digital communication systems because it allows a receiver to correct errors in a message without having to request a re-transmission. The idea is to pad the message with extra bits, known as parity bits, based on a mathematical rule. For the Sprite, a linear block code is used where the parity bits are generated by simple matrix multiplication.

To encode an 8-bit message byte m, the Sprite encoder treats the byte as an 8-dimensional binary vector and multiplies it by a 16-by-8-bit matrix G, known as the generator matrix of the code, producing a 16-bit code word c:

$$c = mG \tag{2.4}$$

All arithmetic operations in equation (2.4) are performed modulo-2, with multiplication equivalent to the Boolean *and* operation and addition equivalent to the *exclusive-or* operation. Note that it is conventional to use row vectors in coding theory[19].

The generator matrix used on the Sprite is

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

In the notation commonly used in coding theory, this is a (16,8,5) block code, where 16 is the code word length, 8 is the message length, and 5 is the Hamming distance[19], which determines the ability of the code to correct errors. With a Hamming distance of 5, up to 2 bit flips or 5 bit erasures can be corrected[19].

There are several ways a receiver can decode a block code like the one presented here. In the Sprite receiver, a simple brute-force soft decoder is used, where the received code word is compared to all 256 possible code words. An inner product is calculated with each one, and the decoded byte is taken to be the best match. While this decoder is optimal in the sense that it produces the maximum-likelihood message byte, its computational complexity scales exponentially with the message length and it quickly becomes intractable for longer messages. Several algebraic decoding methods exist[15], [19] which are, in general, sub-optimal, but are much more computationally efficient.

2.3.2 Signal Design

The Sprite signal is designed to be easily transmitted by commercially available single-chip industrial, scientific, and medical (ISM) band radios like the CC1101 from Texas Instruments[20], and to be demodulated by software-defined radio (SDR) receivers. This combination allows for low cost and flexibility in both the transmitter and receiver. Additionally, the protocol is designed to accommodate multiple Sprites on the same channel, handle large frequency offsets due to Doppler shift and oscillator drift, and reliably close the communication link at a range of 650 km.

The Sprite protocol operates at the level of 8-bit bytes. A data byte is first encoded using the generator matrix in equation (2.5), producing a 16-bit codeword containing the original data byte along with a parity byte. A preamble and postamble are then added to the resulting codeword. Figure 2.3 illustrates this packet structure. The preamble and postamble are 7-bit Barker codes, which have optimal autocorrelation properties[21] and aid synchronization in the receiver.

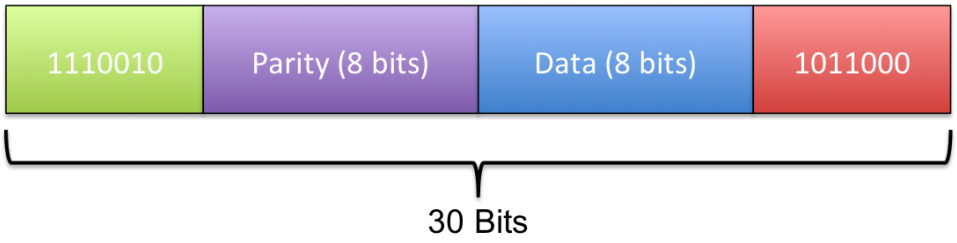


Figure 2.3: Sprite packet structure

Every Sprite is assigned a pair of 511-bit Gold codes, one corresponding to a zero bit and the other corresponding to a one bit. Due to hardware constraints, an extra zero is appended to each Gold code to make its length exactly 64 bytes. After a packet has been encoded, each bit in the packet is replaced with the corresponding Gold code. The packet is then transmitted using some form of phase-coherent digital modulation. In the case of the CC1101 radio on the current Sprite, minimum-shift keying (MSK) modulation is used, but other modulation schemes, such as binary-phased-shift keying (BPSK) and quadrature-phase-shift keying (QPSK) could also be employed.

The maximum Doppler shift of a Sprite signal as seen by a ground station is approximately 10 kHz. Additionally, the crystal oscillator on the Sprite has a nominal frequency stability of 2 ppm, corresponding to roughly 1 kHz at the Sprite's operating frequency of 437 Mhz. As a result, a worst-case frequency deviation of 11 kHz must be handled. The Sprite's chipping rate of 64 kHz was chosen to accommodate this deviation while simultaneously balancing computational requirements in the receiver.

The power spectral density of a Sprite signal is shown in figure 2.4. The central lobe, which accounts for approximately 99% of the total signal power, occupies 64 kHz of bandwidth. With a frequency deviation of 11 kHz, a decrease of approximately 2 dB in received signal power occurs. While a lower chipping rate would

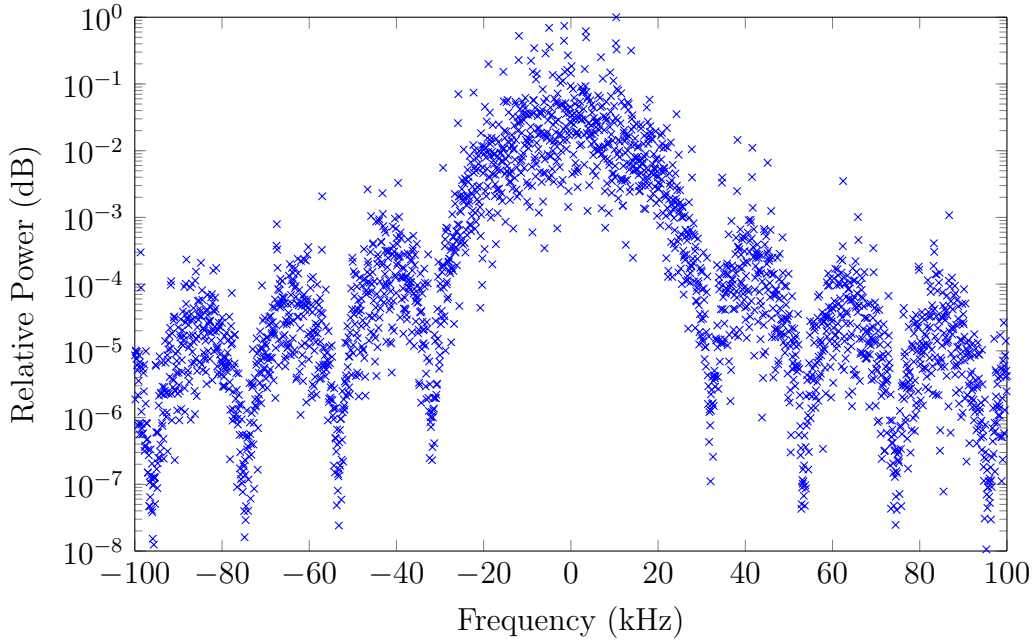


Figure 2.4: Power spectral density of a Sprite signal

require less bandwidth and less computational throughput to demodulate, it would lead to a greater loss of signal power in the presence of frequency offsets, and an unacceptable degradation of link margin.

2.3.3 Receiver Implementation

Because of the signal processing requirements inherent in the Sprite communication protocol, a software-defined radio architecture was chosen for the receiver. In an SDR, a radio front end is connected directly to an analog-digital converter, which then feeds the resulting digitized baseband signal into a computer. All demodulation and decoding then happens in software. SDR architectures are very flexible, allowing changes to the signal structure, modulation, and decoding process to be implemented purely at the software level without any hardware changes.

A reference design for a Sprite receiver has been developed using a hand-held Yagi antenna, a low-noise amplifier (LNA), a low-cost software-defined radio receiver (commonly known as a DVB-T or RTL receiver), and a laptop computer running the GNU Radio software[22]. All required hardware components are commercially available and are shown in figure 2.5.

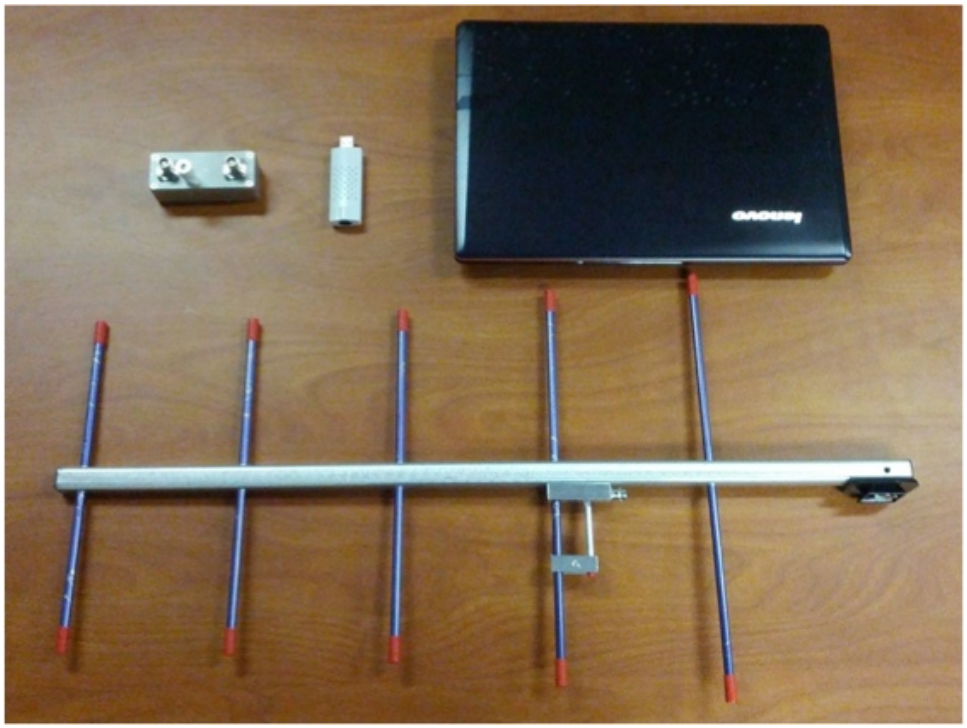


Figure 2.5: Sprite ground station hardware

Figure 2.6 shows the signal flow diagram of the receiver in GNU Radio. Each signal processing block is written in either C++ or Python, with the GNU Radio libraries providing low-level hardware drivers, scheduling, and synchronization.

The first block in the upper left, labeled "RTL-SDR Source," is the driver for the USB radio receiver, which provides a tuner, gain stage, and analog to digital conversion. Next, the signal is low-pass filtered and re-sampled at the chipping rate of 64 kHz. The correlator block then simultaneously performs a frequency

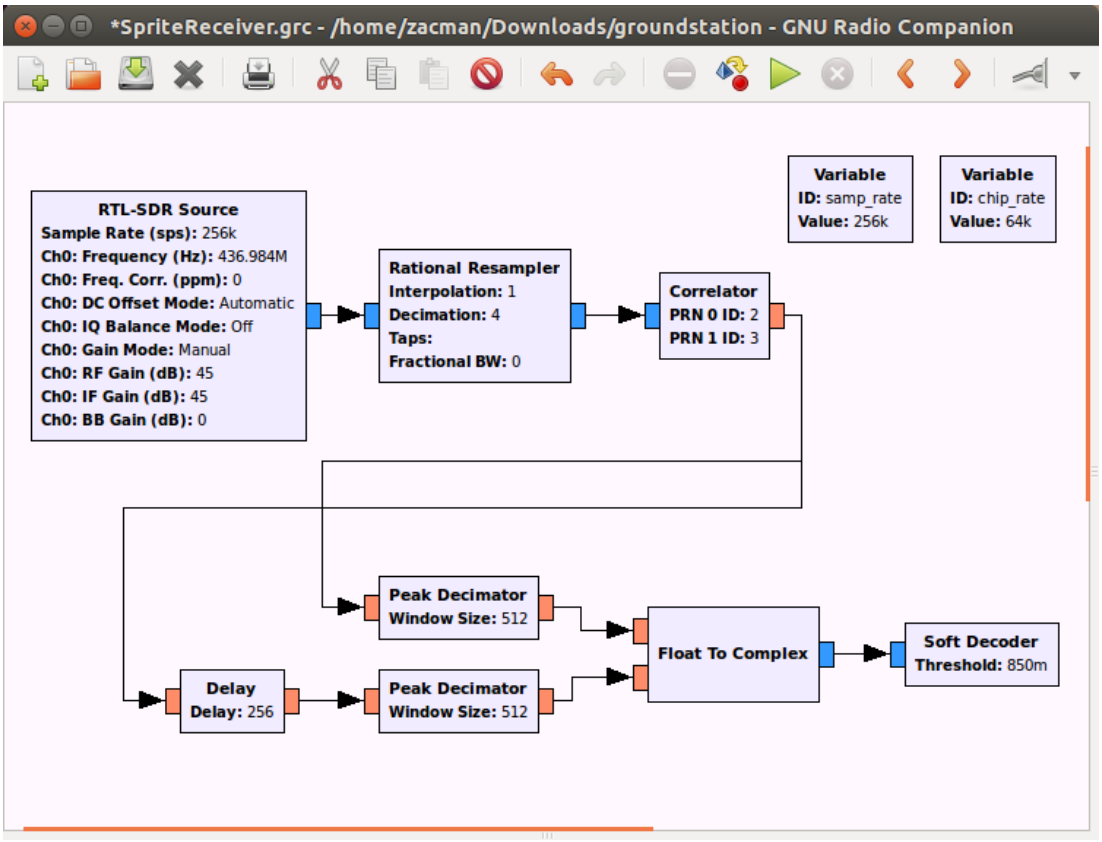


Figure 2.6: GNU Radio signal flow diagram

search and cross-correlations against a pair of Gold codes. The frequency search operation is carried out efficiently using fast Fourier transforms[23].

At each time step, the correlator block outputs the largest-magnitude correlation coefficient, with those corresponding to the zero-bit PRN code negated. The "Peak Decimator" block then breaks the data stream into windows of 512 samples, returning only the largest magnitude sample in each window. The resulting data stream is at a sample rate of 125 Hz. A pair of these blocks operates in parallel, with windows offset by half a PRN period, to ensure that two bits are not conflated by being in the same 512 sample window due to slight clock offsets between the transmitter and receiver. Finally, the "Soft Decoder" block performs the soft-decision linear block code decoding operation described in section 2.3.1.

The Sprite software receiver can run in real time on recent PC hardware. It can also run in a batch mode where RF data is recorded during a pass and fed through the receiver later, when the signals of each Sprite can be extracted individually. Both scenarios have been successfully tested in the laboratory and outdoors.

2.4 Communication Testing

Both laboratory hardware-in-the-loop and outdoor "ridge tests" have been performed to validate the Sprite communication system. The laboratory tests serve to validate decoding algorithms and test individual hardware components in a controlled environment. The outdoor tests, on the other hand, attempt to test the end-to-end communication system in as realistic a setting as possible before flight.

Hardware-in-the-loop tests have been performed using a combination of MATLAB and GNU Radio software to synthesize Sprite signals over a wide range of signal-to-noise ratios and Doppler shifts. Signals are then transmitted by an Ettus Research Universal Software Radio Peripheral (USRP), which performs digital to analog conversion, frequency conversion, and amplification[24]. The analog signal from the USRP is then fed through an attenuator and into the receiver. Reliable communication has been achieved under simulated link conditions equivalent to those in table 2.1.

Several outdoor link tests have been performed with Sprite flight units placed in sunlit locations at the University of California's Lick Observatory, atop Mount Hamilton in San Jose County, California. The Sprites were suspended several wavelengths above the ground to avoid ground effects and were entirely solar powered during the tests. The receiver described in section 2.3.3 was positioned at NASA

Ames Research Center, in Mountain View, California. The line-of-sight distance between Sprites and receiver was 37 kilometers. An additional 23 decibels of attenuation was added between the receiving antenna and low-noise amplifier, producing an equivalent effective range of approximately 523 kilometers. Figure 2.7 shows an oscilloscope trace taken from the output of a correlator during an outdoor ridge test, with the spike indicating detection of a PRN code clearly visible.

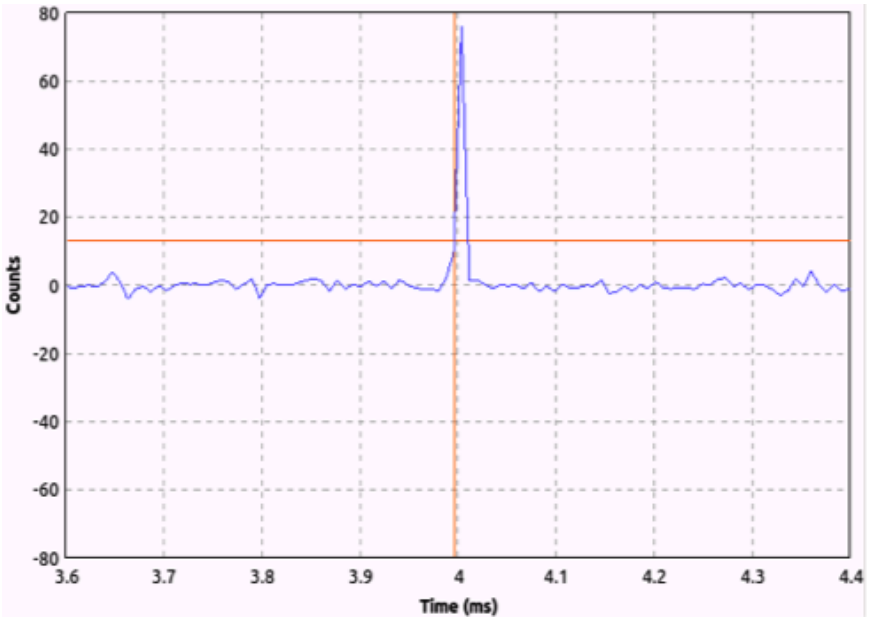


Figure 2.7: Receiver correlator output during outdoor ridge test

2.5 On-Orbit Testing

Three Sprite prototypes were mounted to the outside of the International Space Station (ISS) during the STS-134 Space Shuttle mission in May 2011 as part of the 8th Materials International Space Station (MISSE-8) experiment[25]. While the Sprites were launched in fully functional condition and were presumably operating while on orbit, the location at which they were mounted on the ISS prevented

any signals from reaching Earth. Figure 2.8 shows a photograph taken during the spacewalk in which the MISSE-8 experiment was installed on the ISS. The three Sprites are visible in the lower left.

Figure 2.8: Sprite prototypes mounted on the MISSE-8 experiment

The MISSE-8 experiment, including the Sprite prototypes, was returned to Earth in May 2014 on the Space-X CRS-3 mission after spending three years exposed to the space environment. Post-flight functional testing indicates that the solar cells, microcontrollers, and radios of all three Sprites remain in working condition. While functional testing alone cannot quantify radiation damage to the individual semiconductor devices, this early positive result indicates that the COTS components used on the Sprite can survive in the space environment for extended periods of time.

2.6 Conclusions

The Sprite represents a significant advancement over previous femtosatellites. It is smaller and less massive than any existing spacecraft by at least one order of magnitude. In addition, the low-power, long-range communication solution developed for the Sprite not only solves a key outstanding problem with previous femtosatellite designs, but also has potential uses in other contexts, such as in fail-safe or backup modes on larger spacecraft.

CHAPTER 3

KICKSAT

The KickSat project was founded in 2011 with the goal of advancing the core technologies needed to enable low-cost ChipSat missions. In addition to the Sprite ChipSat, this includes a CubeSat-based deployment system and a software-defined radio ground station. The project's first orbital demonstration mission, KickSat-1, launched as a secondary payload on the SpaceX CRS-3 mission in April 2014.

This chapter includes a description of the KickSat spacecraft in section 3.1, followed by a discussion of the mission profile for KickSat-1 in section 3.2. Next, section 3.3 describes the procedure used to measure the inertia of KickSat, followed by a discussion of deployment, thermal vacuum, and vibration testing in section 3.4. Finally, section 3.5 discusses the outcomes of the KickSat-1 mission.

3.1 The KickSat Spacecraft

The KickSat spacecraft, shown in figure 3.1, is a three-unit ("3U") CubeSat[26]. The bottom third of the spacecraft comprises the bus, which provides power, communication, command and data handling, and attitude determination and control functions. The upper two thirds contains a Sprite deployer. KickSat's avionics are based on the flight-proven PhoneSat 2.0 CubeSat developed at NASA Ames Research Center[27], and are built entirely from commercial-off-the-shelf (COTS) components.

The Sprite deployer is designed for simplicity and robustness while minimizing attitude disturbances on the spacecraft during deployment. The deployer houses 104 Sprites stacked in four columns in a two-by-two arrangement. Each Sprite is

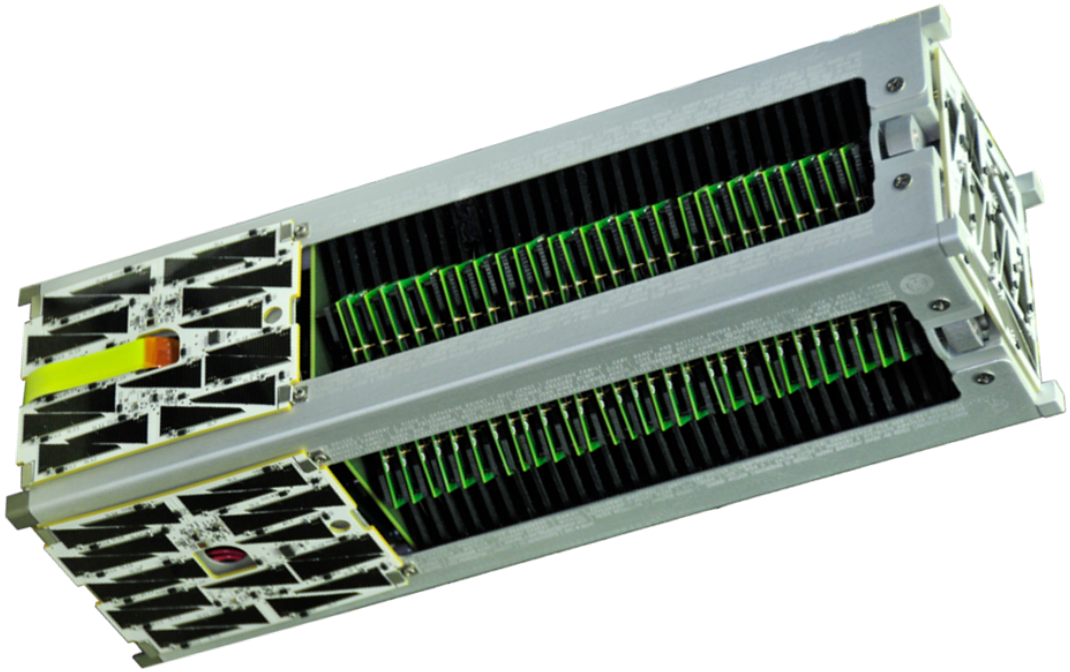


Figure 3.1: The KickSat Spacecraft

housed in an individual slot and constrained by a carbon-fiber rod that runs the length of the column and passes through a hole in the corner of every Sprite, as shown in figure 3.2. The coiled nitinol wire antennas on the Sprites double as deployment springs to push the Sprites out of their slots.

All four carbon fiber rods are attached to a single plate at the top of the deployer that is actuated by a compressed spring and held in place by a locking mechanism. Deployment is triggered by a nichrome burn wire, which unlocks the mechanism, allowing the spring to pull the four rods out. The Sprites' antennas then push them from the deployer housing, as shown in figure 3.3.

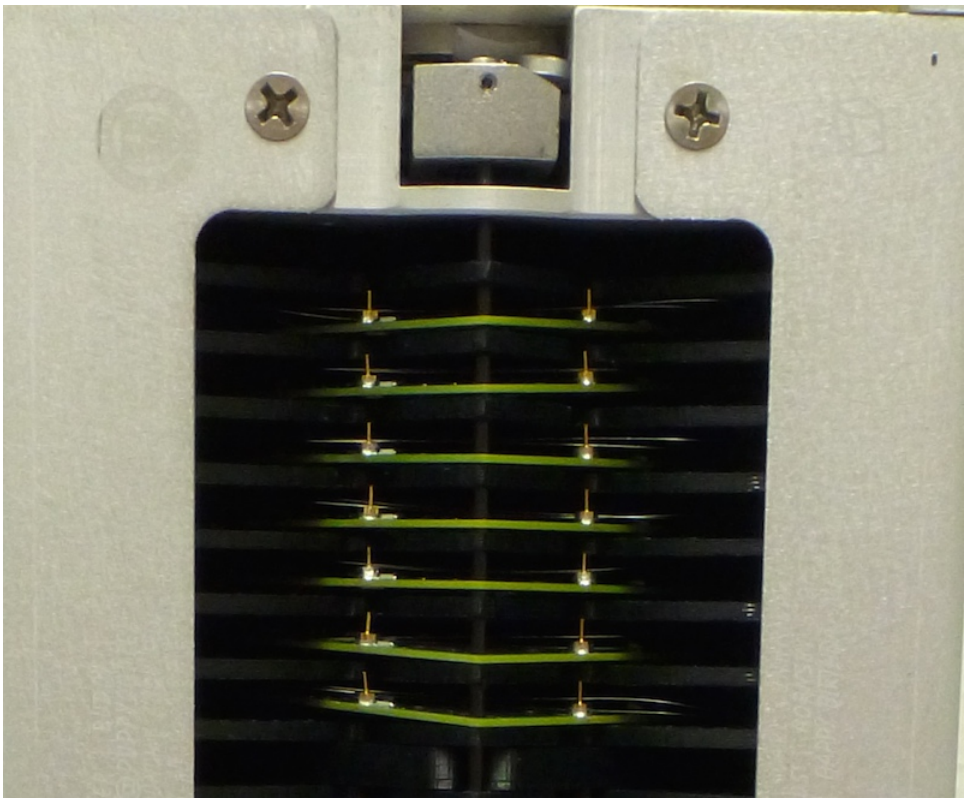


Figure 3.2: Sprites in KickSat deployer

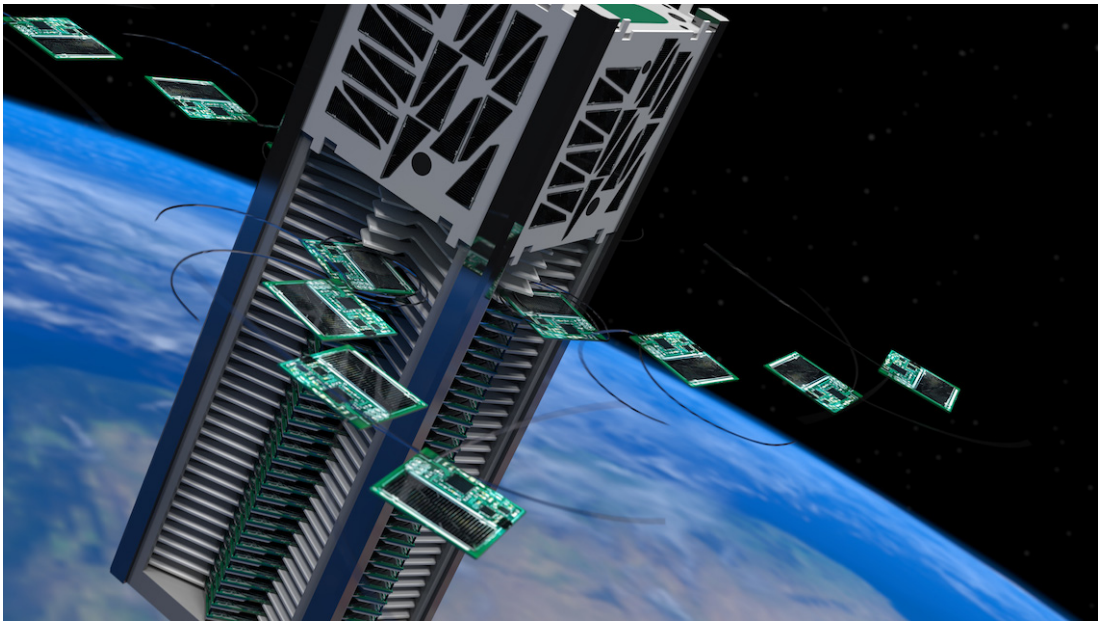


Figure 3.3: Sprite deployment

3.2 Mission Profile

KickSat has been awarded a launch through NASAs Educational Launch of Nanosatellites (ELaNa) program, which places university-built CubeSats as secondary payloads on NASA missions. The majority of ELaNa launch opportunities are to a roughly circular orbit with an altitude of approximately 400 km and an inclination of 51.5°. This orbit has two useful properties. First, the high inclination puts the Sprites in view of almost every populated area of the Earth. Second, the low altitude limits the orbital lifetime of the Sprites to a few days, mitigating orbital debris concerns.

Upon separation from the launch vehicle, KickSat will power up and start a countdown timer. After 30 minutes, a UHF radio antenna will be deployed from the bus. Then, after 45 minutes, the bus radio will begin transmitting a beacon signal. During the first few passes, ground station operators will establish communication and perform checkouts of the spacecraft. Over the next three to four days, the attitude control system in the bus will be used to align KickSats minor axis of inertia (long axis) with the sun vector and spin the spacecraft up to 10-15 RPM, ensuring attitude stability during the deployment sequence.

Once a stable sun-pointing attitude has been established, all systems have been checked out, and KickSat is in view of a ground station, a signal from the ground will trigger a nichrome burn wire to unlock the deployer. The deployer's spring mechanism will then release the Sprites as free-flying spacecraft. The deployment sequence is intended to release the Sprites in a sun-pointing major-axis spin, which is dynamically stable[28]. While there are no strict pointing requirements, the goal of this spin stabilization is to keep the Sprites' solar panels pointed at the sun for the duration of the mission and to minimize nutation, which would otherwise

introduce attitude-dependent fluctuations in power.

Due to their extremely low ballistic coefficient, the Sprites are expected to remain in orbit for only a few days before reentering and burning up in the atmosphere, alleviating debris concerns. Figure 3.4 shows altitude vs. time plots for bounding maximum and minimum, as well as average, atmospheric density taken from the MSIS atmospheric model[14]. In line with other CubeSat missions, the

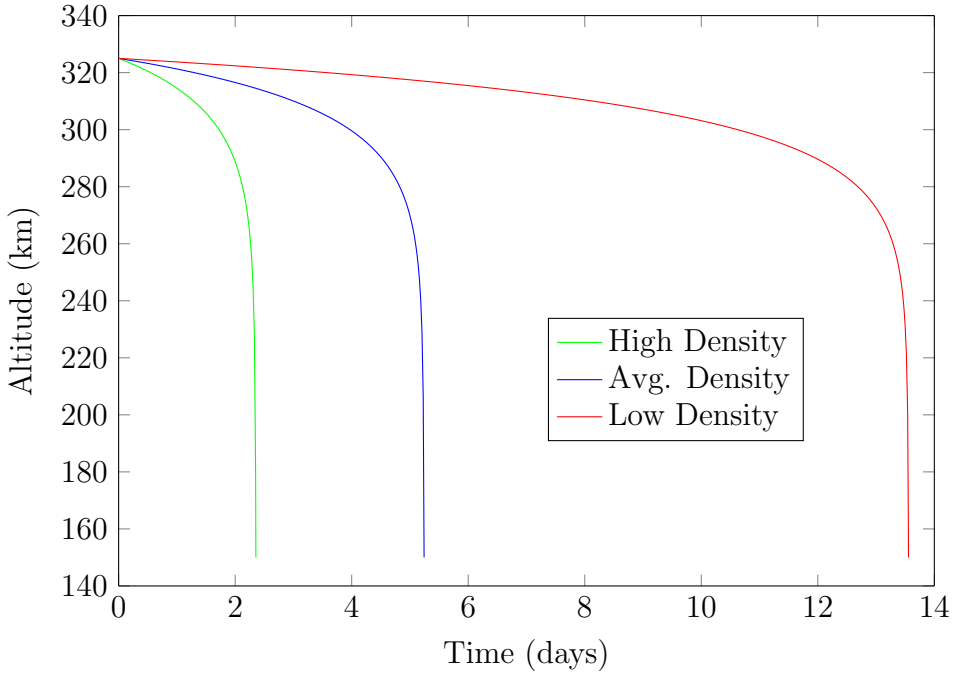


Figure 3.4: Altitude vs. time for a Sprite in low-Earth orbit

KickSat bus is expected to remain in orbit for a few months before it too reenters. It may serve as a test bed for further communication and attitude control experiments during that time.

3.3 Inertia Measurement

The attitude determination and control requirements of KickSat necessitate knowledge of the spacecraft's inertia tensor. Traditional methods (e.g. the "spin balance method") used in the aerospace industry to measure mass properties are accurate but require expensive specialized equipment[29]. As an alternative, a system known as a bifilar pendulum[30], [31] was used to measure moments of inertia about KickSat's body x, y, and z axes. This section provides a derivation of the bifilar pendulum's equation of motion and a description of the method's implementation.

3.3.1 The Bifilar Pendulum

A bifilar pendulum consists of a rigid body with mass m and inertia I suspended from a support structure by pair of parallel strings or cables of length L with horizontal separation $2r$. Figure 3.5 depicts a such a device and and defines the coordinates used in the following derivation.

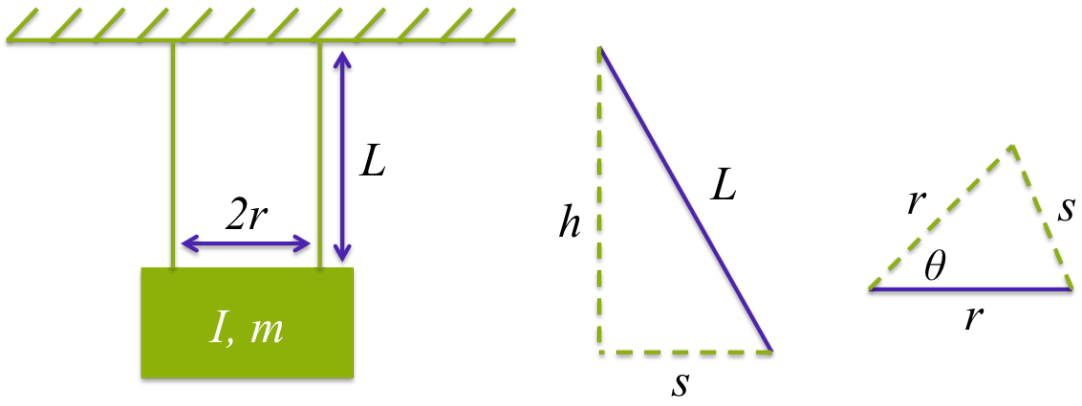


Figure 3.5: Bifilar pendulum schematic

The dynamics of the bifilar pendulum will be parameterized in terms of θ, the

angular displacement about the vertical axis. Some trigonometry reveals that the height of the body h can be written as

$$h = \sqrt{L^2 - 2r^2(1 - \cos(\theta))} \tag{3.1}$$

The Lagrangian of the system is then,

$$\mathcal{L} = \frac{1}{2} I\dot{\theta}^2 + mg\sqrt{L^2 - 2r^2(1 - \cos(\theta))} \tag{3.2}$$

where g is the local gravitational acceleration. Substituting equation 3.2 into the Euler-Lagrange equation[32],

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\dot{\theta}}\right) - \frac{\partial \mathcal{L}}{\partial \theta} = 0 \tag{3.3}$$

then gives the nonlinear equation of motion

$$I\ddot{\theta} + \frac{mgr^2 \sin(\theta)}{\sqrt{L^2 - 2r^2(1 - \cos(\theta))}} = 0 \tag{3.4}$$

To estimate the moment of inertia about the vertical axis I, a small-angle approximation is made and equation 3.4 is linearized, producing the following simple harmonic oscillator equation:

$$I\ddot{\theta} + \frac{mgr^2}{L}\,\theta = 0 \tag{3.5}$$

The frequency of oscillation ω can then be related to the inertia as follows,

$$\omega^2 = \frac{mgr^2}{IL} \implies I = \frac{mgr^2}{\omega^2 L} = \frac{mgr^2T^2}{4\pi^2 L} \tag{3.6}$$

where T is the period of oscillation. A first-order error analysis gives the following approximation for the uncertainty in I given the uncertainties in the measured system parameters:

$$\sigma_I = \frac{mgr^2T^2}{4\pi^2 L}\sqrt{\frac{\sigma_m^2}{m^2} + \frac{\sigma_g^2}{g^2} + \frac{\sigma_r^2}{4r^2} + \frac{\sigma_L^2}{L^2} + \frac{\sigma_T^2}{4T^2}} \tag{3.7}$$

3.3.2 Experimental Setup and Results

A bifilar pendulum was constructed by suspending the KickSat spacecraft from the laboratory ceiling using two lengths of nylon monofilament, chosen for its light weight and high tensile strength. Small-amplitude oscillations were excited by applying a torque to the spacecraft. Measurements of the oscillation period T were averaged over ten periods. Figure 3.6 shows the apparatus during a measurement of the z-axis moment of inertia. Tests were repeated for the x-axis and y-axis.

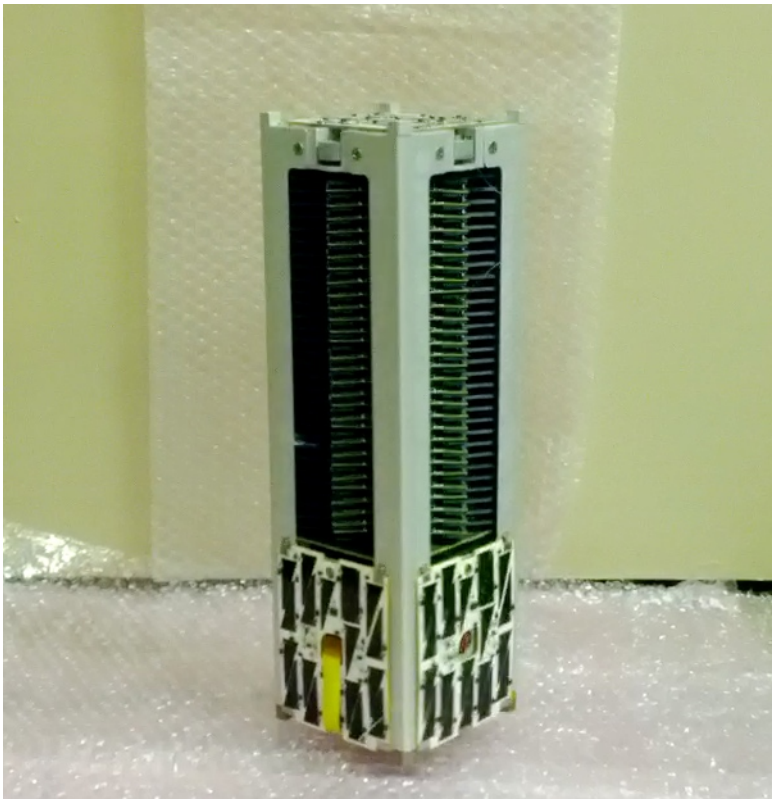


Figure 3.6: KickSat z-axis inertia measurement

Table 3.1 lists the measurements recorded for each moment of inertia, along with corresponding measurement uncertainties. Relative uncertainties near 1% were achieved for all three axes. However, these bifilar pendulum experiments fail to capture information about the products of inertia or, equivalently, the true

Table 3.1: Inertia measurement data

Axis	L (mm)	r (mm)	m (kg)	g (mm/s^2)	T (s)	I (kg·cm^2)
x	1945±5	27.0±.5	2.68±.01	9807±10	11.60±.05	387 ± 3.9
y	2000±5	33.5±.5	2.68±.01	9807±10	9.90±.05	366 ± 3.3
z	1750±5	51.0±.5	2.68±.01	9807±10	2.25±.05	$49.6 \pm .65$

orientation of the principle axes of inertia. As a result, significant uncertainty remains in the off-diagonal elements of the inertia tensor, necessitating the on-line inertia estimation algorithm developed in chapter 4.

3.4 Pre-flight Testing

KickSat was subjected to a variety tests, both during engineering development and to meet NASA requirements[33], [34]. This section documents deployment, vibration, and thermal vacuum testing. The full contents of all test and verification reports submitted to the NASA Launch Services Program are reproduced in appendix E.

3.4.1 Deployment Testing

Numerous deployment tests were performed to tune and verify the performance of the Sprite deployment system. Before each test, the KickSat spacecraft was loaded with Sprites and placed on a pedestal one meter above the laboratory floor, which was covered with antistatic padding material to cushion the impact of the Sprites. Deployment was triggered by a radio command and high-speed video was recorded at 240 frames per second during each test for subsequent analysis. Figure

3.7 shows a series of video frames captured during a deployment test.

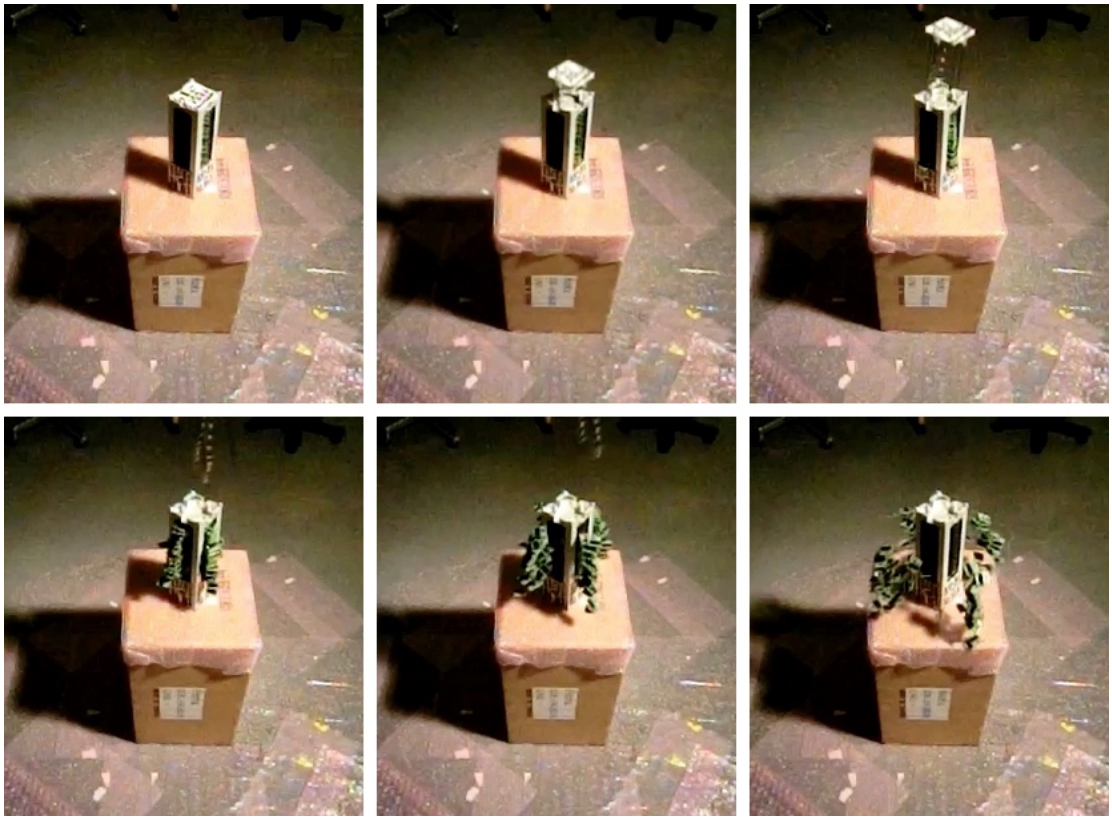


Figure 3.7: KickSat deployment test

3.4.2 Vibration Testing

KickSat underwent vibration testing from August 26-28, 2013 at Space Systems Loral in Palo Alto, California. The testing process was performed in accordance with NASA CubeSat requirements[33]. Vibration frequency and amplitude profiles for the Falcon 9 launch vehicle were provided by SpaceX.

All tests were performed on a Ling shaker table. KickSat was mounted in a TestPOD, a mechanical simulator for the Poly Picosatellite Orbital Deployer (P-POD) used to deploy CubeSats from a launch vehicle[26]. A one inch thick

aluminum adapter plate was fabricated to bolt the TestPOD to the standard two-inch-by-two-inch bolt pattern on the shaker table. A mechanical drawing of this plate is included in appendix E. Three three-axis accelerometers were used in all tests: one attached to the outside corner of the TestPOD, one attached to KickSat's structure, and a control accelerometer attached to the adapter plate. The complete test setup is shown in figure 3.8.

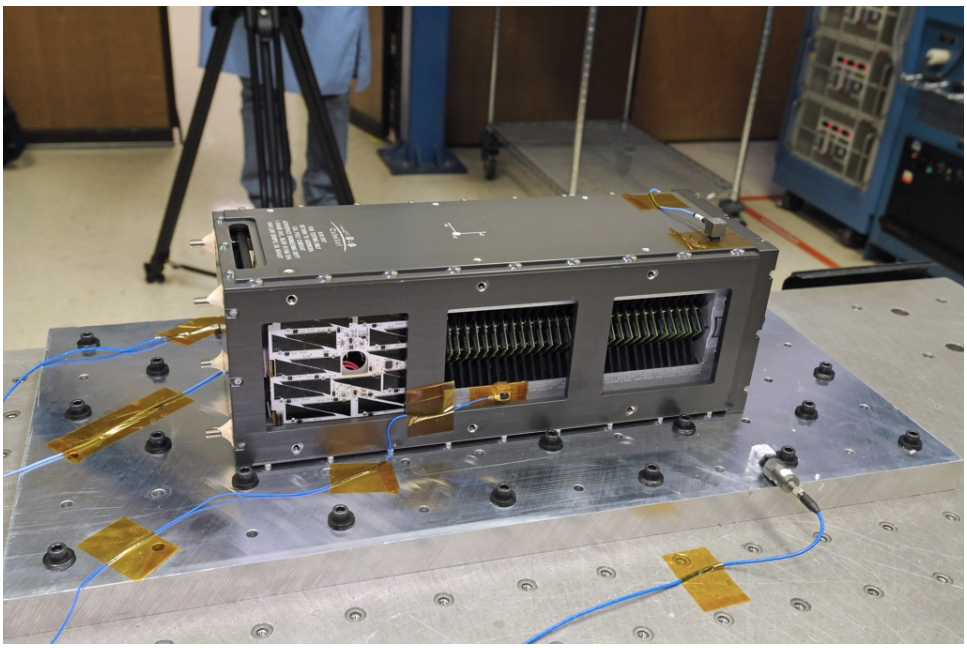


Figure 3.8: Vibration test setup

For each of the spacecraft's x, y, and z axes, a sine sweep was first performed to measure the frequency response of the spacecraft's structure. The CubeSat was then subjected to random vibration, followed by a second sine sweep to ensure that the structural modes of the spacecraft had not changed, which would indicate damage. Plots of all test data are provided in appendix E. KickSat passed all vibration tests with no signs of damage.

3.4.3 Thermal Vacuum Testing

KickSat underwent thermal vacuum testing from August 21-22, 2013 at NASA Ames Research Center's Engineering Evaluation Laboratory. The test requirements imposed by NASA's Launch Services Program specified maintaining a pressure less than 10^{-4} Torr and a nominal temperature of 60°C for a minimum of six hours. Additionally, the temperature ramp rate was not to exceed 5°C per minute.

KickSat was fitted with two thermocouples; one each on the $+x$ and $-x$ exterior faces of the spacecraft. The thermocouples were attached to the anodized aluminum surface of the CubeSat structure before the test as shown in Figure 3.9. Figure 3.10 shows the temperature and pressure data recorded during the test. The temperature readings of the two thermocouples agreed to within 2°C throughout the test.

In total, a minimum temperature of 60°C and maximum pressure of 3.7×10^{-4} Torr were maintained for six hours and 50 minutes. Significant outgassing was observed as the chamber temperature increased, beginning approximately seven hours into the test. It was determined that this outgassing was due to sodium hydroxide residue left on 3D printed plastic components as part of the manufacturing process.

3.5 KickSat-1 Mission Outcomes

The KickSat-1 spacecraft launched on the SpaceX CRS-3 mission on April 18, 2014. After successful separation from the launch vehicle and antenna deployment, telemetry was received by Cornell's ground station during KickSat's second orbit.

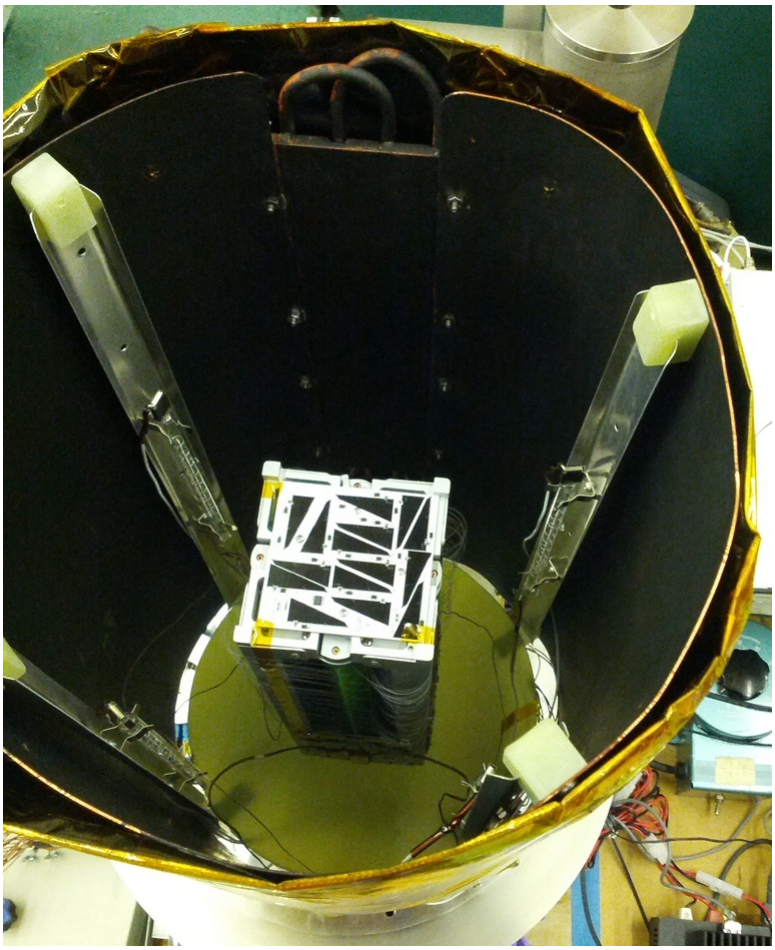


Figure 3.9: Thermal vacuum test setup

Over the next few weeks, a group of dozens of amateurs from around the world collaborated over the internet to receive, decode, and analyze flight data; make reentry predictions; and share a wealth of technical information.

Sprite deployment had originally been planned for three to four days after launch. However, due to launch delays and scheduling conflicts with other spacecraft bound for the ISS, a 16-day delay between launch and Sprite deployment was imposed by NASA. This delay was implemented prior to launch in the form of a software countdown timer. Unfortunately, after operating normally for 14 days, KickSat experienced an anomaly in which the bus momentarily lost power and the

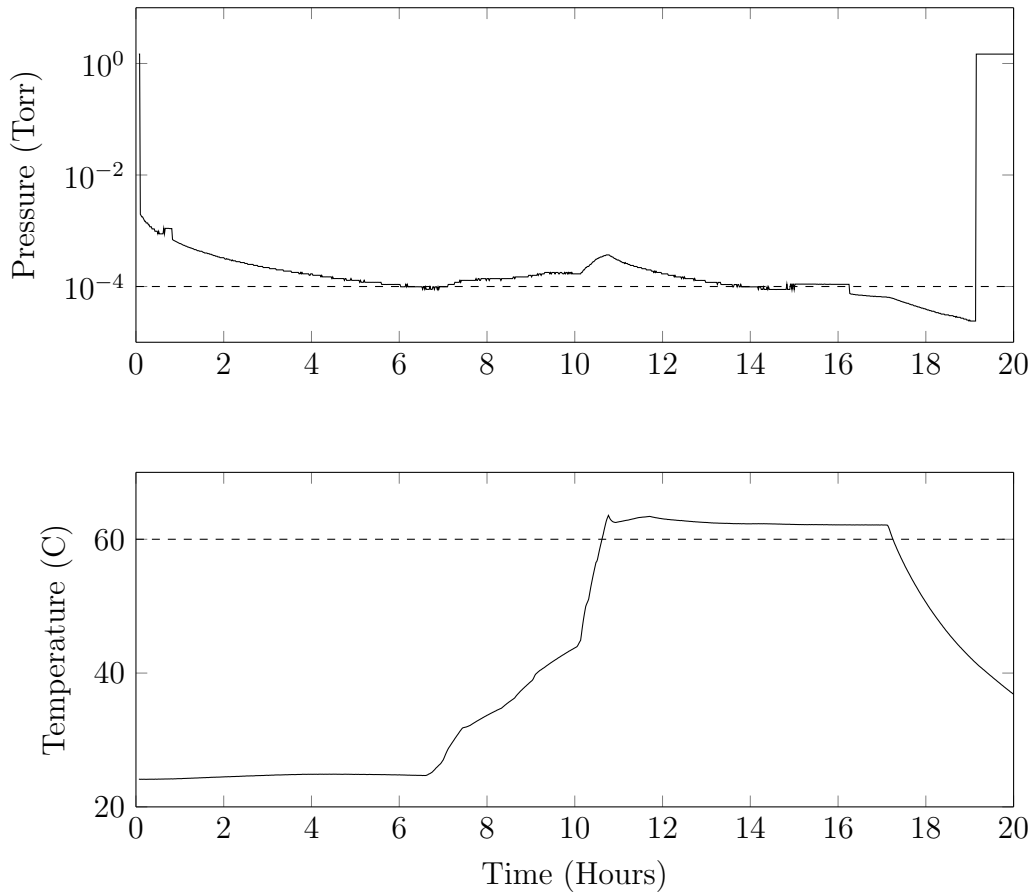


Figure 3.10: Temperature and pressure during thermal vacuum test

avionics were reset. As a result, the 16 day deployment countdown was restarted.

Several attempts were made to uplink commands to the spacecraft. Ultimately, however, they were unsuccessful due to power and other technical constraints inherent in the spacecraft bus design. On May 13, after spending 25 days in orbit, KickSat reentered Earth's atmosphere without deploying its payload of Sprites.

3.6 Conclusions

KickSat represents the beginning of a new paradigm for low-cost space missions. The small size, mass producibility, low cost, and short development cycles possible with ChipSats can enable new science missions, as well as expand access to space. While the KickSat-1 mission was not a complete success, it directly involved non-specialists from across the globe in spaceflight in new ways. To ensure that the technologies developed as part of KickSat are available to as broad an audience as possible, design files and code have been made freely available under open-source licenses. In the near future, it will be possible for students, hobbyists, and scientists alike to put together ChipSat-based missions in a matter of weeks or days, and at costs one to two orders of magnitude less than current small satellite missions.

CHAPTER 4

RECURSIVE INERTIA ESTIMATION WITH SEMIDEFINITE PROGRAMMING

Knowledge of a spacecraft's inertia tensor is vital to the performance of guidance, navigation, and control algorithms. While estimates of the inertia can be calculated before launch by tabulating the masses and locations of spacecraft components, the accuracy of such estimates is limited. Additionally, a spacecraft's inertia may change in unpredictable ways throughout a mission due to fuel use, deployment failures, damage, or a variety of other reasons. As a result, the ability to estimate a spacecraft's inertia on orbit can offer improved pointing performance and robustness to modeling uncertainty and component failures.

This chapter proceeds with a review of existing methods for inertia estimation in section 4.1. Semidefinite programming is introduced in section 4.2, along with the notation conventions used throughout the chapter. Section 4.3 then develops a discrete-time equation of motion for a gyrostat spacecraft. Section 4.4 formulates batch inertia estimation as a semidefinite program. Section 4.5 uses the batch estimator as the basis for a recursive inertia estimation algorithm. Finally, numerical simulations involving both spinning and three-axis stabilized spacecraft are presented in section 4.6 to demonstrate the performance of the recursive algorithm.

4.1 Existing Work

Several algorithms for estimating a spacecraft's inertia parameters from telemetry data have been proposed. Bergmann, Walker, and Levy developed a filter for estimating inertia that ignores the nonlinear terms in the rigid body equations of mo-

tion[35]. Ahmed, Coppola, and Bernstein proposed an adaptive attitude-tracking controller that recovers the elements of the inertia matrix under certain excitation conditions[36]. Psiaki presented a batch least-squares algorithm for estimating inertia as well as actuator alignment and scaling parameters[37]. Tanygin and Williams developed a least-squares estimator based on kinetic energy and integration of the work done by actuator inputs[38]. Norman, Peck, and O'Shaughnessy proposed a recursive least-squares algorithm for estimating inertia and actuator alignment parameters based on angular momentum[39]. The algorithm developed here is most closely related to that of Keim, Behcet, and Shields, who formulated batch least-squares inertia estimation as a semidefinite program to enforce positive-definiteness and constrain the elements of inertia with upper and lower bounds[40].

All of the existing inertia estimation methods cited suffer from at least one of the following drawbacks: First, most do not take into account the physical constraints on the components of the inertia tensor, in particular, positive-definiteness and the "triangle inequality"[41]. Without accounting for these constraints, estimation algorithms can return unphysical results and perform poorly when data is sparse or very noisy. Second, many algorithms require derivatives or integrals of measured spacecraft state variables as input, necessitating prefiltering that introduces additional complexity and numerical error. This study seeks to address both of these issues with a recursive estimator suitable for real-time implementation.

4.2 Background

This section provides brief reviews of semidefinite programming and the algebra of quaternions. More thorough treatments of both topics are available in the books of Boyd and Vandenberghe[42] and Altmann[43], respectively.

4.2.1 Semidefinite Programming

A semidefinite program (SDP) is an optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^\intercal \boldsymbol{x} \\ \text{subject to} \quad & F(\boldsymbol{x}) \geq 0 \end{aligned} \tag{4.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{c}$ are vectors in $\mathbb{R}^n$ and F is an $m \times m$ symmetric matrix defined as follows:

$$F(\boldsymbol{x}) = F_0 + \sum_{i=1}^{n} x_i F_i \tag{4.2}$$

The inequality constraint $F(\boldsymbol{x}) \geq 0$ means that $F(x)$ must be positive semidefinite. That is, $\boldsymbol{z}^\intercal F(x) \boldsymbol{z} \geq 0$ for all vectors $\boldsymbol{z}$ in $\mathbb{R}^m$. Constraints of this type are known as linear matrix inequalities (LMIs)[44]. SDPs are convex. As a result, fast numerical algorithms with guaranteed convergence are available for solving them[42].

Many standard optimization problems, including linear and quadratic programs, can be put into SDP form. In particular, the linear least-squares problem

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \|H\boldsymbol{x} - \boldsymbol{y}\|_2^2 \tag{4.3}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{y} \in \mathbb{R}^m$, and $H \in \mathbb{R}^{m \times n}$, can be recast as an SDP by introducing a scalar slack variable s into the vector of optimization variables:

$$\boldsymbol{x}' = \begin{bmatrix} \boldsymbol{x} \\ s \end{bmatrix} \tag{4.4}$$

Using the fact that the Schur complement of a positive semidefinite matrix must also be positive semidefinite, the least-squares problem can be embedded into the following LMI,

$$F_{LS}(\boldsymbol{x}') = \begin{bmatrix} s & (H\boldsymbol{x} - \boldsymbol{y})^\mathsf{T} \\ (H\boldsymbol{x} - \boldsymbol{y}) & I \end{bmatrix} \geq 0 \tag{4.5}$$

which is equivalent to the scalar inequality

$$s - (H\boldsymbol{x} - \boldsymbol{y})^\mathsf{T}(H\boldsymbol{x} - \boldsymbol{y}) \geq 0 \tag{4.6}$$

The least-squares problem in standard SDP form is then

$$\begin{aligned} &\text{minimize} \quad \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ s \end{bmatrix} \\ &\text{subject to} \quad F_{LS}(\boldsymbol{x}') \geq 0 \end{aligned} \tag{4.7}$$

Once an optimization problem is posed as an SDP, it can be modified by imposing a variety of additional LMI constraints. Multiple constraints can be enforced by forming a block-diagonal concatenation of their LMI representations. In this way, linear and quadratic problems subject to a large class of convex constraints can be solved efficiently.

4.2.2 Quaternion Algebra

Quaternions form an algebra with a non-commutative binary product operation. It is often convenient to think of them as four-dimensional objects composed of a three-dimensional vector part $\boldsymbol{v}$ and a scalar part s.

$$q = \begin{bmatrix} \boldsymbol{v} \\ s \end{bmatrix} \tag{4.8}$$

This representation allows the quaternion product to be written in terms of scalar and vector products:

$$q_1 q_2 = \begin{bmatrix} \boldsymbol{v_1} \times \boldsymbol{v_2} + s_1 \boldsymbol{v_2} + s_2 \boldsymbol{v_1} \\ s_1 s_2 - \boldsymbol{v_1} \cdot \boldsymbol{v_2} \end{bmatrix} \tag{4.9}$$

Note that $q_1 q_2 \neq q_2 q_1$. Throughout the chapter, quaternion products are indicated by juxtaposition, while scalar and vector products are indicated in the usual way, with the $\cdot$ and $\times$ symbols, respectively.

Rotations can be conveniently represented by unit-length quaternions. If $\boldsymbol{r}$ is a unit vector in $\mathbb{R}^3$ representing the axis of rotation and θ is the angle of rotation, then the quaternion representing the rotation is as follows:

$$q = \begin{bmatrix} \boldsymbol{r} \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} \tag{4.10}$$

Both q and $-q$ correspond to the same rotation, making the unit quaternions a "double cover" of the group of rotations.

The conjugate of a quaternion is denoted with a superscript $\dagger$ and represents the rotation about the same axis $\boldsymbol{r}$ by $-\theta$.

$$q^\dagger = \begin{bmatrix} -\boldsymbol{v} \\ s \end{bmatrix} \tag{4.11}$$

Two rotations can be composed by multiplying their quaternion representations. A quaternion q_3 representing a rotation q_1 followed by a rotation q_2 is simply $q_3 = q_2 q_1$. The rotation of a three-dimensional vector $\boldsymbol{x}$ by a unit quaternion q is

$$\hat{\boldsymbol{x}}' = q \hat{\boldsymbol{x}} q^\dagger \tag{4.12}$$

where $\hat{\boldsymbol{x}}$ indicates the formation of a quaternion with zero scalar part from the vector $\boldsymbol{x}$:

$$\hat{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x} \\ 0 \end{bmatrix} \tag{4.13}$$

Finally, the subsequent analysis requires some kinematic identities relating quaternion derivatives to vector quantities more familiar in rigid body dynamics. First, the time derivative of a body's attitude quaternion is related to its angular velocity in the following way:

$$\hat{\boldsymbol{\omega}} = 2q^{\dagger}\dot{q} \tag{4.14}$$

Second, the quaternion generalized force corresponding to a torque on the body is[45], [46]

$$\mathcal{F} = 2q\hat{\boldsymbol{\tau}} \tag{4.15}$$

Schaub and Junkins provide a thorough discussion of rigid body dynamics using quaternions[47].

4.3 Discrete Gyrostats Mechanics

A gyrostat is a system of coupled rigid bodies whose relative motions do not change the total inertia tensor of the system. This abstraction serves as a practical mathematical model for a spacecraft with reaction wheels. The fundamental differential equation governing the motion of a gyrostat is,

$$J \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (J \cdot \boldsymbol{\omega} + \boldsymbol{\rho}) + \dot{\boldsymbol{\rho}} = \boldsymbol{\tau} \tag{4.16}$$

where J is the symmetric positive-definite inertia tensor of the gyrostat, $\boldsymbol{\omega}$ is the body angular velocity, $\boldsymbol{\rho}$ is the total angular momentum stored in the rotors, and $\boldsymbol{\tau}$ is the external torque applied to the gyrostat[28]. In addition to being symmetric and positive definite, the elements of J are also constrained by the

triangle inequality[41]:

$$\begin{aligned} J_{11} + J_{22} &\geq J_{33} \\ J_{11} + J_{33} &\geq J_{22} \\ J_{22} + J_{33} &\geq J_{11} \end{aligned} \tag{4.17}$$

In principle, the inertia can be estimated using equation (4.16) if time histories of the variables $\boldsymbol{\omega}$, $\dot{\boldsymbol{\omega}}$, $\boldsymbol{\rho}$, and $\dot{\boldsymbol{\rho}}$ are available. In practice, however, the angular acceleration $\dot{\boldsymbol{\omega}}$ is not measured directly and must be obtained from noisy measurements of $\boldsymbol{\omega}$. Many solutions to this problem have been proposed, including various finite difference and filtering schemes[40], [48] and integrating both sides of equation (4.16) with respect to time[37]–[39], but all add additional complexity and suffer to varying degrees from numerical error and noise amplification. Here, a new approach is taken in which discrete variational mechanics is used to derive a discrete-time version of equation (4.16) that does not contain $\dot{\boldsymbol{\omega}}$.

Discrete mechanics is a mathematical framework for rigorously deriving discrete-time algebraic equations of motion for mechanical systems[49]. The essential idea is to approximate the derivatives and integrals encountered in classical Lagrangian or Hamiltonian mechanics with finite differences and quadrature rules. Discretizations derived from variational principles offer many advantages over more traditional schemes like Runge-Kutta methods, including momentum and energy conservation[49]. The remainder of this section gives a brief derivation of the discrete-time gyrostat equation originally presented by the authors in [50].

4.3.1 Torque-Free Motion

The Lagrangian for a gyrostat is,

$$\mathcal{L} = \frac{1}{2}\boldsymbol{\omega}_B \cdot J_B \cdot \boldsymbol{\omega}_B + \sum_{r=1}^{N_r} \frac{1}{2}(\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) \cdot J_r \cdot (\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) \tag{4.18}$$

where J_B is the carrier body's inertia tensor (including rotor masses), $\boldsymbol{\omega}_B$ is the carrier body's angular velocity, the J_r are the rotor inertia tensors, the $\boldsymbol{\omega}_r$ are the rotor angular velocities relative to the carrier body, and the $\boldsymbol{x}_r$ are the rotor positions relative to the carrier body's center of mass. Using equation (4.18), Hamilton's principle states that the equation of motion for the gyrostat, in the absence of external torques, can be found by setting the variation of the action equal to zero[32]:

$$\delta\mathcal{S} = \delta \int_{t_0}^{t_f} \mathcal{L}\,\mathrm{d}t = 0 \tag{4.19}$$

The transition to discrete mechanics begins by breaking the action integral into short segments of length h, with $t_k = t_0 + kh$:

$$\delta\mathcal{S} = \delta \int_{t_0}^{t_f} \mathcal{L}\,\mathrm{d}t = \delta \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} \mathcal{L}\,\mathrm{d}t = 0 \tag{4.20}$$

The integral over a single time step on the right hand side of equation (4.20) is then approximated using a quadrature rule. First, the body's angular velocity is approximated by a finite difference of quaternions:

$$\hat{\boldsymbol{\omega}}_k = 2\,q_k^\dagger \dot{q}_k \approx 2\,q_k^\dagger \left(\frac{q_{k+1} - q_k}{h}\right) = 2\left(\frac{f_k - 1}{h}\right) \tag{4.21}$$

The rectangle rule is then applied to arrive at the following discrete Lagrangian[50],

$$\mathcal{L}_d = \frac{2}{h}\left[f_k \cdot \hat{J}_B \cdot f_k + \sum_{r=1}^{N_r} (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \cdot \hat{J}_r \cdot (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k})\right] \tag{4.22}$$

where $\hat{J}_B$ and $\hat{J}_r$ are augmented 4×4 equivalents of J_B and J_r:

$$\hat{J} = \begin{bmatrix} J_{11} & J_{12} & J_{13} & 0 \\ J_{21} & J_{22} & J_{23} & 0 \\ J_{31} & J_{32} & J_{33} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.23}$$

Substituting the discrete Lagrangian back into equation (4.20) results in the discrete action sum:

$$\mathcal{S}_d = \sum_{k=0}^{N-1} \frac{2}{h} \left[f_k \cdot \hat{J}_B \cdot f_k + \sum_{r=1}^{N_r} (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \cdot \hat{J}_r \cdot (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \right] \tag{4.24}$$

Hamilton's principle can then be applied by setting the variational derivative of equation (4.24) equal to zero using a constrained variation of f_k which respects the quaternion unit-norm constraint[50],

$$^{\epsilon}f_k = f_k + \epsilon(f_k\hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) \tag{4.25}$$

resulting in

$$\sum_{k=0}^{N-1} \left[f_k \cdot \hat{J}_B \cdot (f_k\hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) + \sum_{r=1}^{N_r} (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \cdot \hat{J}_r \cdot (f_k\hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) \right] = 0 \tag{4.26}$$

To ensure that the variations $\boldsymbol{\eta}$ in equation (4.26) have the same time index, a "discrete integration by parts" is performed, which amounts to some simple index manipulation:

$$\sum_{k=1}^{N-1} \left[f_{k-1} \cdot \hat{J}_B \cdot (f_{k-1}\hat{\boldsymbol{\eta}}_k - \hat{\boldsymbol{\eta}}_k f_k) + \sum_{r=1}^{N_r} (f_{k-1} + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k-1}) \cdot \hat{J}_r \cdot (f_{k-1}\hat{\boldsymbol{\eta}}_k - \hat{\boldsymbol{\eta}}_k f_k) \right] = 0 \tag{4.27}$$

Recognizing that equation (4.27) must hold for all variations $\boldsymbol{\eta}_k$, making the substitution,

$$f_k = \begin{bmatrix} \boldsymbol{\phi}_k \\ \sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k} \end{bmatrix} \tag{4.28}$$

and performing some algebra, results in the discrete-time gyrostat equation,

$$\begin{aligned} \sqrt{1 - \boldsymbol{\phi}_{k+1} \cdot \boldsymbol{\phi}_{k+1}}\,(J \cdot \boldsymbol{\phi}_{k+1} + \frac{h}{2}\boldsymbol{\rho}_{k+1}) + \boldsymbol{\phi}_{k+1} \times (J \cdot \boldsymbol{\phi}_{k+1} + \frac{h}{2}\boldsymbol{\rho}_{k+1}) \\ = \sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k}\,(J \cdot \boldsymbol{\phi}_k + \frac{h}{2}\boldsymbol{\rho}_k) - \boldsymbol{\phi}_k \times (J \cdot \boldsymbol{\phi}_k + \frac{h}{2}\boldsymbol{\rho}_k) \end{aligned} \tag{4.29}$$

where $\boldsymbol{\rho}$ is the total momentum stored in the rotors and J is the gyrostat inertia:

$$J = J_B + \sum_{r=1}^{N_r} J_r \tag{4.30}$$

4.3.2 External Torques

External torques can be added to equation (4.29) using the integral form of the Lagrange-D'Alembert principle[51],

$$\delta \int_{t_0}^{t_f} \mathcal{L}\, \mathrm{d}t + \int_{t_0}^{t_f} \mathcal{F} \cdot \delta q\, \mathrm{d}t = 0 \tag{4.31}$$

where the second term is the integral of the virtual work done by a generalized force $\mathcal{F}$. The discrete form of equation (4.31) is

$$\delta \sum_{k=0}^{N} \mathcal{L}_d + \sum_{k=0}^{N} \mathcal{F}_d^- \cdot \delta q_k + \mathcal{F}_d^+ \cdot \delta q_{k+1} = 0 \tag{4.32}$$

$\mathcal{F}_d^-$ and $\mathcal{F}_d^+$, known as discrete generalized forces, are defined as follows[49]:

$$\mathcal{F}_d^- = \int_{t_k}^{t_{k+1}} \frac{1}{2} \mathcal{F}(q, \dot{q}) \cdot \frac{\partial q(t)}{\partial q_k}\, \mathrm{d}t \tag{4.33}$$

$$\mathcal{F}_d^+ = \int_{t_k}^{t_{k+1}} \frac{1}{2} \mathcal{F}(q, \dot{q}) \cdot \frac{\partial q(t)}{\partial q_{k+1}}\, \mathrm{d}t \tag{4.34}$$

Inserting the expression for the quaternion generalized force given in equation(4.15) and applying the rectangle rule results in the following discrete quaternion generalized forces:

$$\mathcal{F}_d^- \approx h q_k \hat{\boldsymbol{\tau}}_k \tag{4.35}$$

$$\mathcal{F}_d^+ \approx h q_{k+1} \hat{\boldsymbol{\tau}}_{k+1} \tag{4.36}$$

Substituting these terms into equation (4.32) and performing some additional algebra reveals the forced discrete gyrostat equation:

$$\sqrt{1-\boldsymbol{\phi}_{k+1}\cdot\boldsymbol{\phi}_{k+1}}\,(J\cdot\boldsymbol{\phi}_{k+1}+\frac{h}{2}\boldsymbol{\rho}_{k+1})+\boldsymbol{\phi}_{k+1}\times(J\cdot\boldsymbol{\phi}_{k+1}+\frac{h}{2}\boldsymbol{\rho}_{k+1})+\frac{h^2}{2}\boldsymbol{\tau}_{k+1}$$
$$=\sqrt{1-\boldsymbol{\phi}_{k}\cdot\boldsymbol{\phi}_{k}}\,(J\cdot\boldsymbol{\phi}_{k}+\frac{h}{2}\boldsymbol{\rho}_{k})-\boldsymbol{\phi}_{k}\times(J\cdot\boldsymbol{\phi}_{k}+\frac{h}{2}\boldsymbol{\rho}_{k}) \quad (4.37)$$

While it may seem that $\boldsymbol{\phi}_k$ and $\boldsymbol{\phi}_{k+1}$ in equation (4.37) must be calculated by finite differences of quaternions, thus conferring no real advantage over equation (4.16) with respect to noise amplification, this is not the case. Most attitude determination filters used on board spacecraft, including the multiplicative extended Kalman filter (MEKF)[52], [53], directly estimate $\boldsymbol{\phi}_k$ or some other closely related three-parameter relative rotation at each time step. Equation (4.37), therefore, allows information readily available in existing spacecraft attitude determination systems to be used directly as input to the inertia estimation algorithm.

4.4 Batch Inertia Estimation

In this section, estimation of the components of the inertia tensor from a time history of attitude observations, rotor momenta, and external torques is formulated as a constrained batch least-squares problem. The resulting problem is then transformed into an SDP which can be efficiently solved.

While equation (4.37) is nonlinear in $\boldsymbol{\phi}$, it is linear in J. To make this explicit, the matrix $G(\boldsymbol{\phi})$ and the vector $\boldsymbol{j}$ are defined as follows:

$$G(\boldsymbol{\phi})=\begin{bmatrix}\phi_1 & 0 & 0 & \phi_2 & \phi_3 & 0\\ 0 & \phi_2 & 0 & \phi_1 & 0 & \phi_3\\ 0 & 0 & \phi_3 & 0 & \phi_1 & \phi_2\end{bmatrix} \quad (4.38)$$

$$\boldsymbol{j} = \begin{bmatrix} J_{11} \\ J_{22} \\ J_{33} \\ J_{12} \\ J_{13} \\ J_{23} \end{bmatrix} \tag{4.39}$$

In terms of G and $\boldsymbol{j}$, the matrix vector product $J\boldsymbol{\phi}$ becomes $G(\boldsymbol{\phi})\boldsymbol{j}$, allowing equation (4.37) to be rewritten as,

$$N(\boldsymbol{\phi}_{k+1})\left(G(\boldsymbol{\phi}_{k+1})\boldsymbol{j} + \frac{h}{2}\boldsymbol{\rho}_{k+1}\right) + \frac{h^2}{2}\boldsymbol{\tau}_{k+1} = M(\boldsymbol{\phi}_k)\left(G(\boldsymbol{\phi}_k)\boldsymbol{j} + \frac{h}{2}\boldsymbol{\rho}_k\right) \tag{4.40}$$

where the matrices $M(\boldsymbol{\phi})$ and $N(\boldsymbol{\phi})$ are defined as follows:

$$M(\boldsymbol{\phi}) = \sqrt{1 - \boldsymbol{\phi}\cdot\boldsymbol{\phi}}\, I - S(\boldsymbol{\phi}) \tag{4.41}$$

$$N(\boldsymbol{\phi}) = \sqrt{1 - \boldsymbol{\phi}\cdot\boldsymbol{\phi}}\, I + S(\boldsymbol{\phi}) \tag{4.42}$$

After collecting terms, equation (4.40) can be written as

$$H_k\boldsymbol{j} = \boldsymbol{y}_k \tag{4.43}$$

where the 3×6 matrix H_k and the 3×1 vector $\boldsymbol{y}_k$ are defined as follows:

$$H_k = N(\boldsymbol{\phi}_{k+1})G(\boldsymbol{\phi}_{k+1}) - M(\boldsymbol{\phi}_k)G(\boldsymbol{\phi}_k) \tag{4.44}$$

$$\boldsymbol{y}_k = M(\boldsymbol{\phi}_k)\frac{h}{2}\boldsymbol{\rho}_k - N(\boldsymbol{\phi}_{k+1})\frac{h}{2}\boldsymbol{\rho}_{k+1} - \frac{h^2}{2}\boldsymbol{\tau}_{k+1} \tag{4.45}$$

Given a set of attitude measurements collected over time $\boldsymbol{\phi}_1 \ldots\ \boldsymbol{\phi}_N$, along with corresponding time histories of reaction wheel momenta $\boldsymbol{\rho}_1 \ldots\ \boldsymbol{\rho}_N$ and external

torques $\boldsymbol{\tau}_1 \dots \boldsymbol{\tau}_N$, the batch inertia estimation problem can be stated as,

$$\begin{aligned} \underset{\boldsymbol{j}}{\text{minimize}} \quad & \|H_{1:N}\boldsymbol{j} - \boldsymbol{y}_{1:N}\|_2^2 \\ \text{subject to} \quad & \begin{cases} J > 0 \\ J_{11} + J_{22} - J_{33} \geq 0 \\ J_{11} + J_{33} - J_{22} \geq 0 \\ J_{22} + J_{33} - J_{11} \geq 0 \end{cases} \end{aligned} \tag{4.46}$$

where $H_{1:N}$ and $\boldsymbol{y}_{1:N}$ are concatenations of the H_k and $\boldsymbol{y}_k$ matrices corresponding to each measurement:

$$H_{1:N} = \begin{bmatrix} H_1 \\ \vdots \\ H_N \end{bmatrix} \qquad \boldsymbol{y}_{1:N} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_N \end{bmatrix} \tag{4.47}$$

The constrained least-squares problem (4.46) can then be transformed into an SDP using the results presented in section 4.2:

$$\begin{aligned} \text{minimize} \quad & \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{j} \\ s \end{bmatrix} \\ \text{subject to} \quad & \begin{cases} J - \epsilon I \geq 0 \\ J_{11} + J_{22} - J_{33} \geq 0 \\ J_{11} + J_{33} - J_{22} \geq 0 \\ J_{22} + J_{33} - J_{11} \geq 0 \\ \begin{bmatrix} s & (H_{1:N}\boldsymbol{j} - \boldsymbol{y}_{1:N})^\intercal \\ (H_{1:N}\boldsymbol{j} - \boldsymbol{y}_{1:N}) & I \end{bmatrix} \geq 0 \end{cases} \end{aligned} \tag{4.48}$$

While the five LMI constraints in (4.48) have been written separately due to space constraints, they are concatenated to form a single block-diagonal LMI in practice. The ϵI term in the first constraint, where ϵ is a small positive constant, ensures that J is strictly positive-definite rather than positive-semidefinite. This

constant should be chosen slightly larger than the error tolerance of the SDP solver being used.

In addition to enforcing positive-definiteness and the triangle inequality, a variety of other constraints can be included in the SDP formulation. This is especially useful for incorporating *a priori* knowledge into the estimator. For example, if bounds can be placed on the elements of J from modeling or ground-based testing, they can be easily accounted for by adding additional diagonal elements to the LMI in (4.48).

For J to be uniquely determined, a few conditions on the observations must be met. First, $\boldsymbol{\phi}_k$ must be changing in time. Otherwise the corresponding H_k matrices will be zero. Physically, this means that the spacecraft cannot be stationary or in a spin about a principle axis with no nutation. Second, some non-zero internal rotor momentum or known external torque must be applied to the spacecraft. Otherwise, the vectors $\boldsymbol{y}_k$ will be zero. In that case, the least-squares problem reduces to finding a vector in the null space of H, which is only determined up to an arbitrary scale factor.

4.5 Recursive Inertia Estimation

While the batch estimation scheme (4.48) is sufficient for offline analysis of flight data, a recursive formulation that allows estimates to be updated as new data becomes available is preferred in real-time applications. This section develops a recursive version of (4.48) that enables efficient updating.

The key to the recursive algorithm is the QR decomposition. A given matrix A

can always be factored into the product of an orthogonal matrix Q and an upper-triangular matrix R[54], [55]. If A is rectangular with more rows m than columns n, the lower $m-n$ rows of R are entirely filled with zeros:

$$A = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \tag{4.49}$$

Only the first n columns of Q and first n rows of R, denoted Q_1 and R_1 above, are actually needed to reconstruct A. Standard linear algebra routines are available for calculating Q_1 and R_1 given an input matrix A, known as a "thin" or "economy" QR decomposition[54], [55]. In the rest of this section it is assumed that such a routine is available and the "thin" $m \times n$ Q_1 and $n \times n$ R_1 matrices will simply be denoted Q and R.

By applying the QR decomposition to $H_{1:N}$, the least-squares problem (4.46) can be rewritten as

$$\begin{aligned} \underset{\boldsymbol{j}}{\text{minimize}} \quad & \|R_{1:N}\boldsymbol{j} - \boldsymbol{z}_{1:N}\|_2^2 \\ \text{subject to} \quad & \begin{cases} J > 0 \\ J_{11} + J_{22} - J_{33} \geq 0 \\ J_{11} + J_{33} - J_{22} \geq 0 \\ J_{22} + J_{33} - J_{11} \geq 0 \end{cases} \end{aligned} \tag{4.50}$$

where $\boldsymbol{z}_{1:N} = Q_{1:N}^{\intercal}\boldsymbol{y}_{1:N}$. If a new set of observations H_{N+1} and $\boldsymbol{y}_{N+1}$ becomes available, R and $\boldsymbol{z}$ can be easily updated:

$$[Q_{1:N+1},\, R_{1:N+1}] = \text{qr}\left(\begin{bmatrix} R_{1:N} \\ H_{N+1} \end{bmatrix}\right) \tag{4.51}$$

$$\boldsymbol{z}_{1:N+1} = Q_{1:N+1}\begin{bmatrix} \boldsymbol{z}_{1:N} \\ \boldsymbol{y}_{N+1} \end{bmatrix} \tag{4.52}$$

Once $R_{1:N+1}$ and $\boldsymbol{z}_{1:N+1}$ have been calculated, the constrained minimization (4.50) can be re-solved using the SDP formulation of section 4.4. Algorithm 4.1 summarizes the implementation of the recursive estimator.

Algorithm 4.1: Recursive Inertia Estimator

$H_1 \leftarrow$ Calculate using equation (4.44)

$\boldsymbol{y}_1 \leftarrow$ Calculate using equation (4.45)

$[Q_1, R_1] \leftarrow \text{qr}(H_1)$

$\boldsymbol{z}_1 \leftarrow Q_1^\intercal \boldsymbol{y}_1$

for $k = 2 : N$ **do**

    $H_k \leftarrow$ Calculate using equation (4.44)

    $\boldsymbol{y}_k \leftarrow$ Calculate using equation (4.45)

    $[Q_{1:k},\, R_{1:k}] \leftarrow \text{qr}\left(\begin{bmatrix} R_{1:k-1} \\ H_k \end{bmatrix}\right)$

    $\boldsymbol{z}_{1:k} \leftarrow Q_{1:k}^\intercal \begin{bmatrix} \boldsymbol{z}_{1:k-1} \\ \boldsymbol{y}_k \end{bmatrix}$

    $J_k \leftarrow$ Solve constrained least-squares problem (4.50) using SDP

end for

4.6 Numerical Examples

This section presents two test cases that demonstrate the performance of the recursive inertia estimation algorithm. First, a three-axis stabilized spacecraft is simulated performing a series of short slew maneuvers. Second, a spin-stabilized spacecraft is simulated spinning about its major axis of inertia with some nutation. The spinning case is used to illustrate the benefits of incorporating additional constraints into the estimator by bounding one of the moments of inertia to within 10% of a nominal value. Comparisons are made to the momentum-based recursive inertia estimation algorithm of Norman, Peck, and O'Shaughnessy[39].

All simulations use MATLAB's ODE45 solver to integrate equation (4.16) with white noise disturbance torques applied to the spacecraft. Simulated measurements

are also corrupted with white noise equivalent to that produced by a low-cost single-chip commercial gyroscope[56]. The true inertia in all cases is

$$J_{true} = \begin{bmatrix} 1.0035 & 0.0368 & -0.0678 \\ 0.0368 & 2.0047 & -0.0755 \\ -0.0678 & -0.0755 & 2.9918 \end{bmatrix} \tag{4.53}$$

The MATLAB-based SeDuMi SDP solver[57] is used in these examples. A wide variety of other SDP solvers, both free and commercial, are available[58].

In the first test case, a series of short slew maneuvers which could be executed on a three-axis stabilized spacecraft are simulated. No *a priori* information is used to initialize the estimation algorithms. Figure 4.1 shows the commanded rotor momentum components in body-fixed axes while figure 4.2 shows the simulated body-fixed angular velocity components for the slew maneuvers.

Figure 4.3 shows the relative Frobenius norm error in the estimated inertia for both algorithms. Figures 4.4 and 4.5 show the normalized errors in the estimated moments and products of inertia, respectively, over the course of the simulation. The SDP-based estimator yields substantially more accurate estimates of the inertia components throughout the simulation.

The second test case demonstrates the advantages of including additional constraints in the SDP-based estimator. A spacecraft is simulated spinning about its major axis of inertia with some nutation. The only control input is a short pulse applied to a single reaction wheel 30 seconds into the simulation. Figure 4.6 shows the simulated body-fixed angular velocity components, while figure 4.7 shows the rotor momentum components.

The J_{33} component of the inertia matrix is constrained to be within 10% of a nominal value of 3. Such an assumption is reasonable if, for example, a CAD

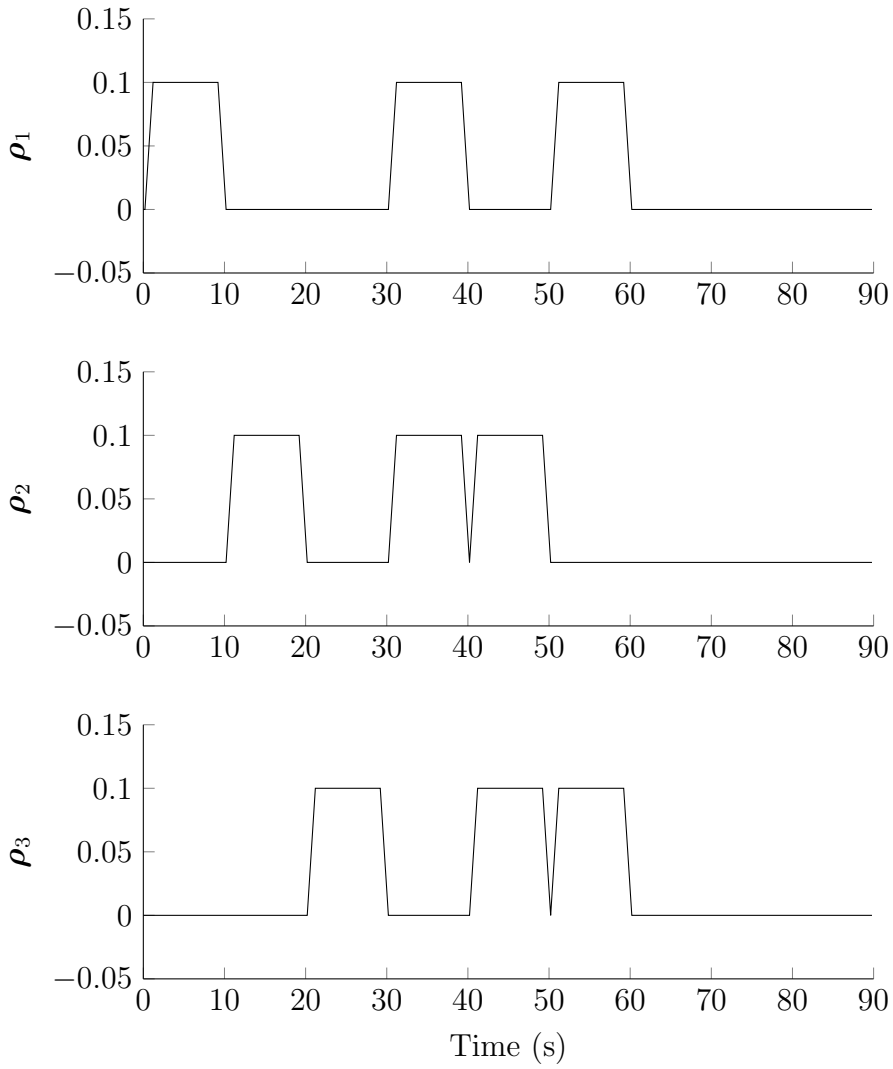


Figure 4.1: Rotor momentum components during slew maneuvers

model or ground-based experiment is used to calculate inertia components before launch. Figure 4.8 shows the relative Frobenius norm error in the estimated inertia for both estimators. Normalized errors in the estimated moments of inertia are plotted in figure 4.9, while the normalized errors in the products of inertia are plotted in figure 4.10.

The SDP-based estimator achieves lower error in all inertia components throughout the simulation. In spite of the fact that *a priori* information is only supplied

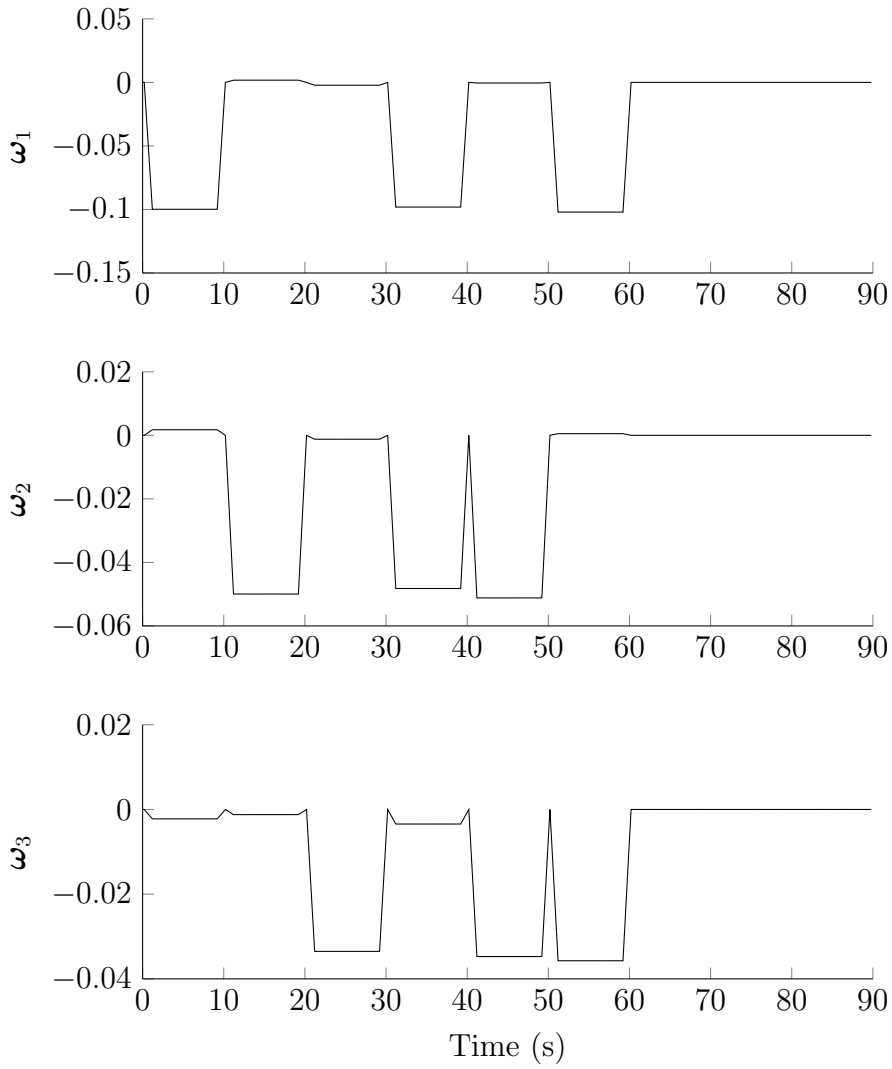


Figure 4.2: Body angular velocity components during slew maneuvers

for a single component of the inertia, the SDP-based estimator is able to achieve relative errors of 10% or better on all inertia components during the first half of the simulation. Physically valid inertia estimates that can be used in other guidance, navigation, and control algorithms on board the spacecraft are always returned. The momentum-based estimator, on the other hand, provides no useful information until a control torque is applied.

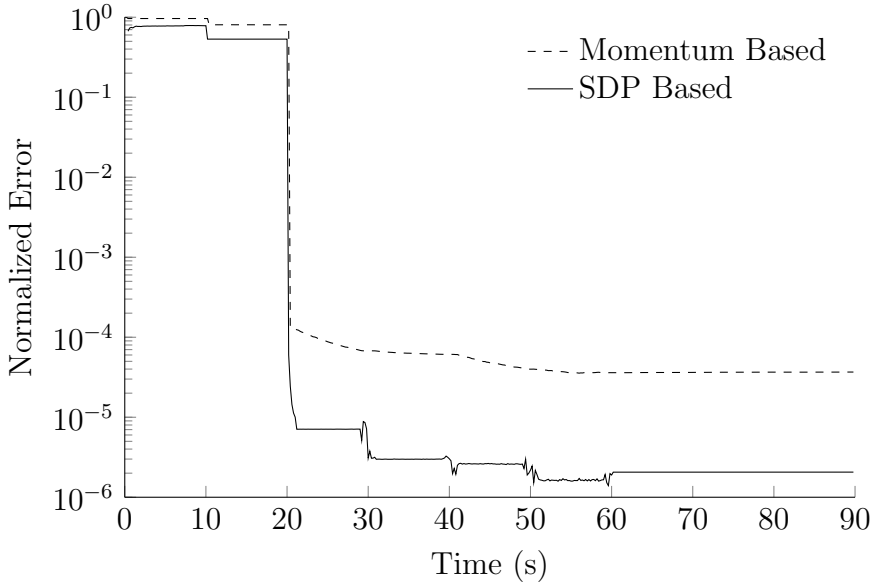


Figure 4.3: Normalized inertia error during slew maneuvers

4.7 Conclusions

This study presents a recursive algorithm for spacecraft inertia estimation using semidefinite programming. A discrete mechanics formulation of the spacecraft dynamics allows data generated by standard attitude determination algorithms to be directly used as input to the estimator. By formulating the estimation problem as an SDP, the positive-definiteness and triangle-inequality constraints on the elements of the inertia tensor can be enforced, ensuring physically valid results. *A priori* knowledge can also be incorporated into the estimator in the form of additional LMI constraints, enhancing convergence and leading to more accurate estimates when limited data is available. The availability of fast interior-point SDP solvers makes the algorithm suited for real-time implementation.

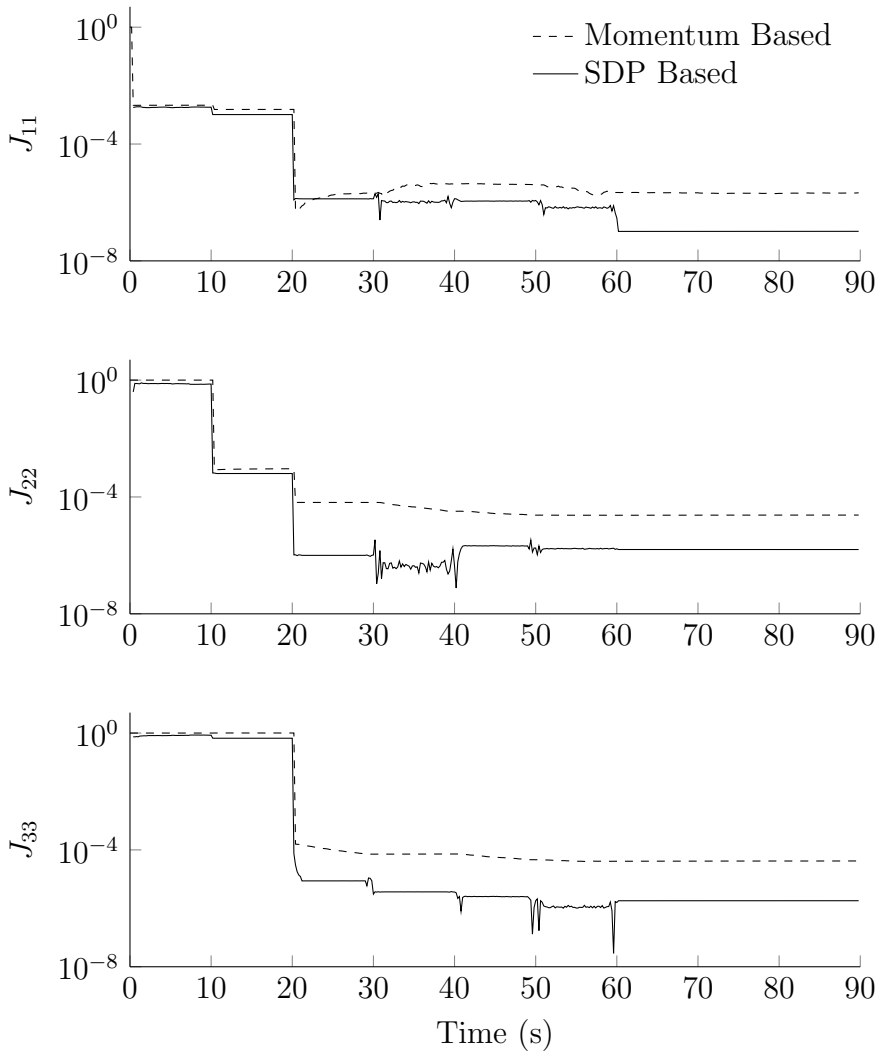


Figure 4.4: Normalized moment of inertia errors during slew maneuvers

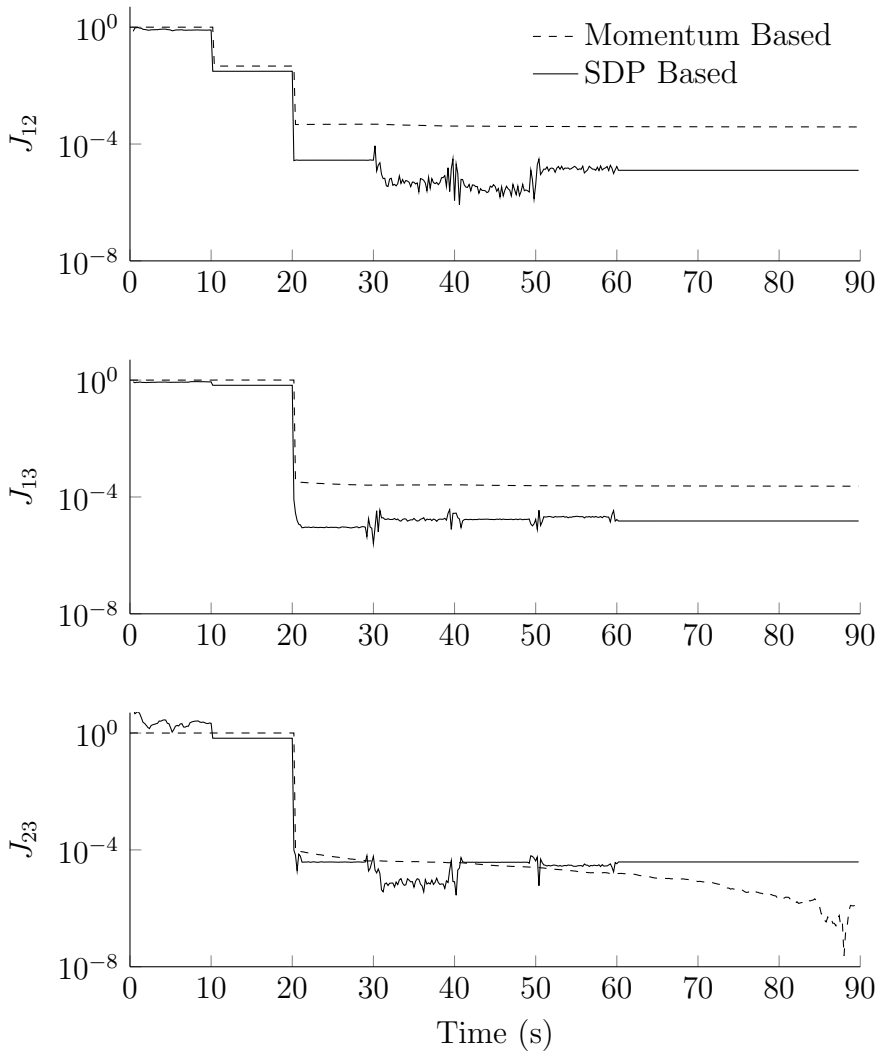


Figure 4.5: Normalized product of inertia errors during slew maneuvers

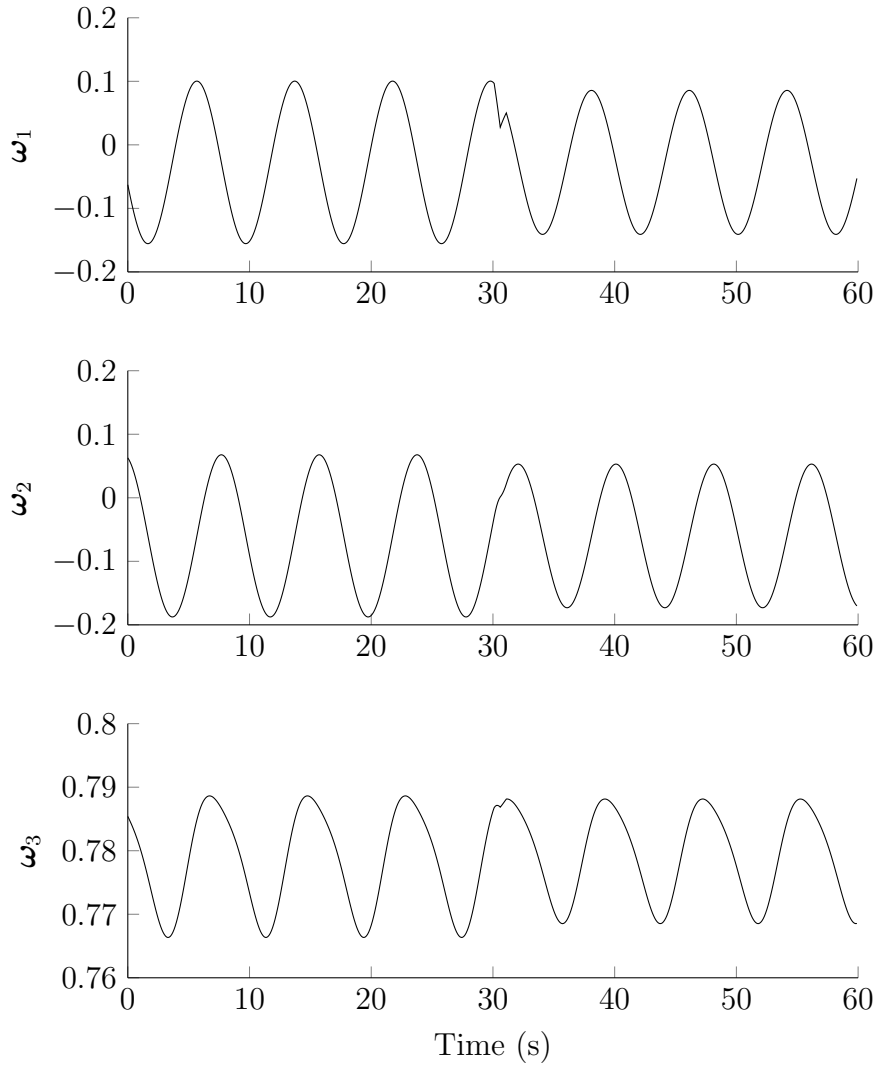


Figure 4.6: Body angular velocity components for spinning case

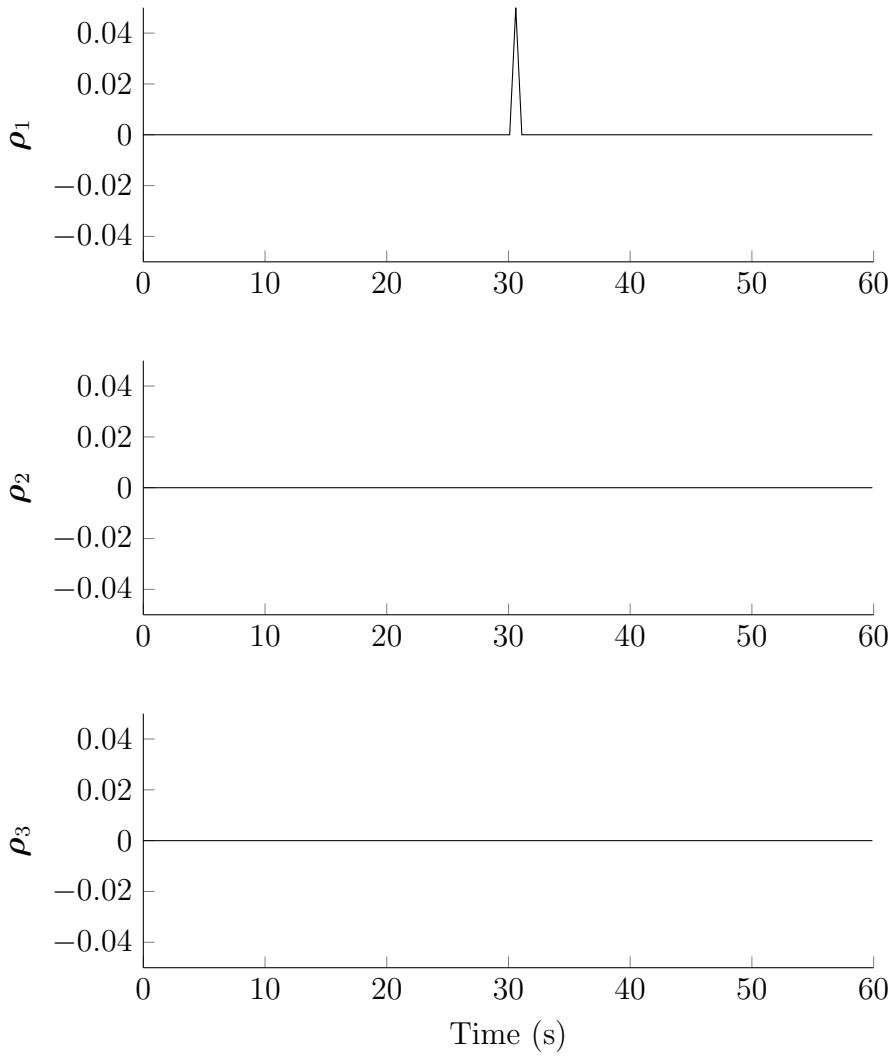


Figure 4.7: Rotor momentum components for spinning case

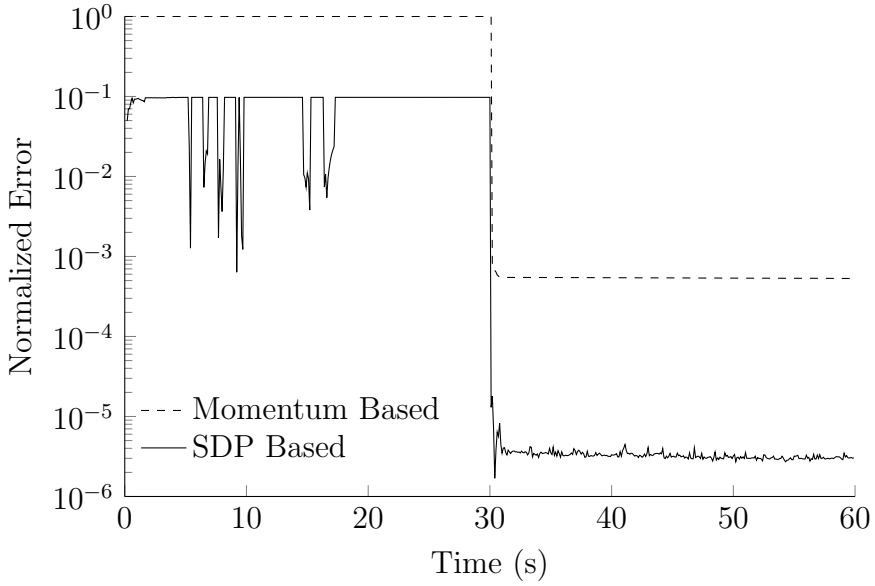


Figure 4.8: Total normalized error for spinning case

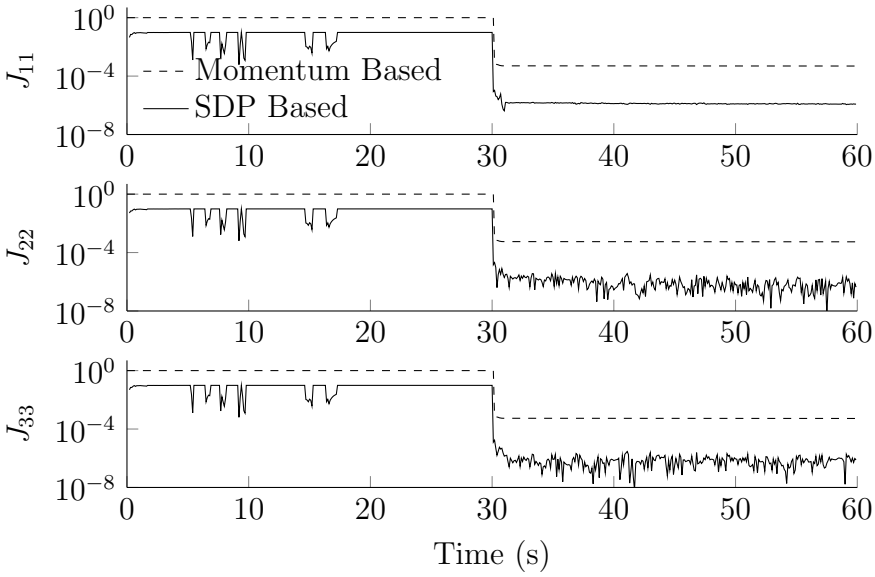


Figure 4.9: Normalized moment of inertia errors for spinning case

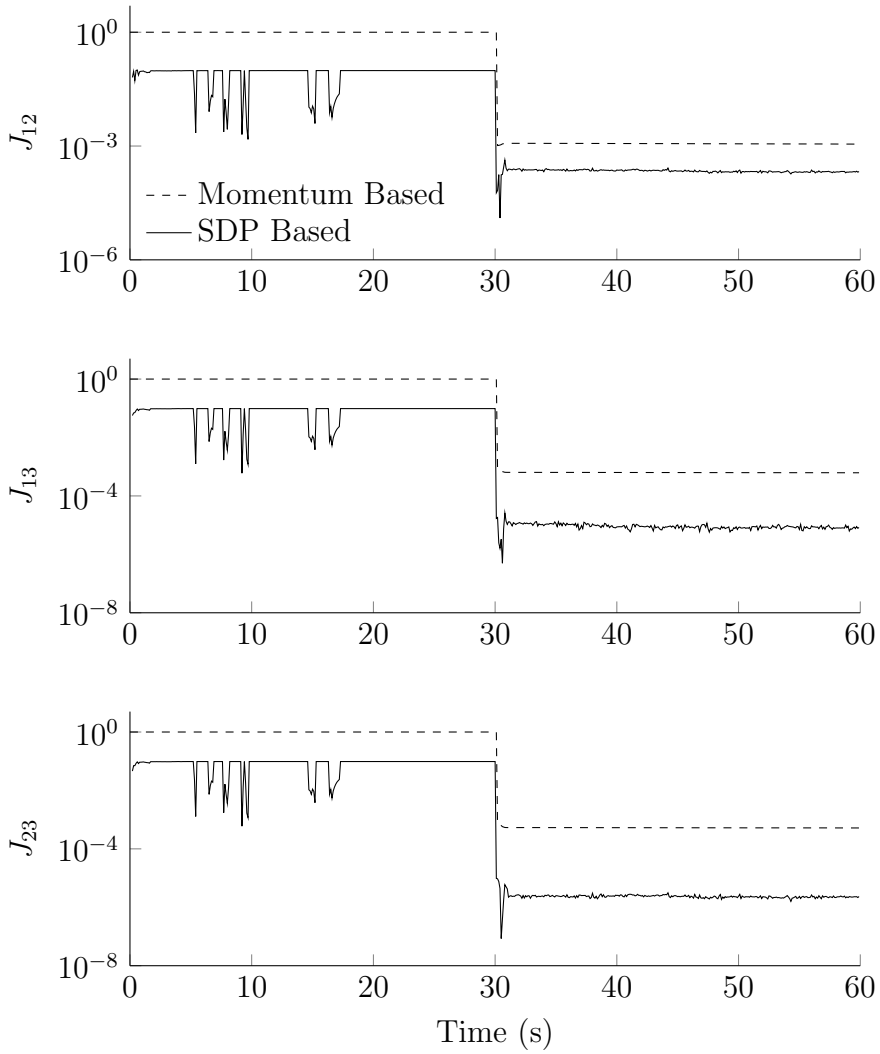


Figure 4.10: Normalized product of inertia errors for spinning case

CHAPTER 5

LYAPUNOV-BASED CONTROL FOR FLAT-SPIN RECOVERY AND SPIN INVERSION OF SPIN-STABILIZED SPACECRAFT

Flat-spin recovery is a classic problem in spacecraft dynamics and control with a rich history dating back to the very first man-made satellites[59]. Fundamentally, the problem has its roots in the geometry of rockets and the fairings that house satellites during launch, which tend to be long and narrow. As a result, most spacecraft have prolate mass distributions, at least in their launch configurations.

In the presence of energy dissipation due, for example, to fluid slosh or flexible structural elements, a spacecraft spinning about its long axis, or minor axis of inertia, will tend to transition into a spin about its major axis, commonly known as a flat spin. Maintaining a minor-axis spin, therefore, requires active control. While spin-stabilized satellites have largely been supplanted by three-axis stabilized designs, many missions still call for spacecraft to spin about their minor axes temporarily, especially during critical phases like deployments and orbit insertion maneuvers[60]. Additionally, low-cost small satellites are frequently designed as minor-axis spinners, both to provide stabilization and to meet other mission requirements[61]–[63].

In this chapter, a family of nonlinear feedback control laws capable of driving a spinning spacecraft from any initial state to a desired minor-axis spin using reaction wheels is developed. The controllers provide almost-global asymptotic stability and can explicitly accommodate actuator limits. Additionally, the feedback laws have a simple and explicit mathematical form and are easy to tune and implement.

The chapter proceeds as follows: Section 5.1 reviews existing methods for flat-spin recovery and spin-axis inversion. Section 5.2 then provides a review of relevant spacecraft dynamics as well as Lyapunov stability theory. Next, sections 5.3 and 5.4 introduce and analyze several candidate Lyapunov functions. Section 5.5 analyzes the global stability properties of controllers based on a particular choice of Lyapunov function. Section 5.6 then discusses the implementation of a family of control laws which are almost-globally asymptotically stabilizing. The control laws are then demonstrated in numerical examples in section 5.7. Finally, conclusions are outlined in section 5.8.

5.1 Existing Work

Many authors have analyzed variations of the flat-spin recovery problem over the past several decades. In its most basic form, the problem entails controlling a spacecraft in such a way that it transitions from a major-axis spin to a minor-axis spin. Early methods focused on utilizing thrusters as actuators[60], [64], however these have the disadvantage of consuming limited propellant resources. In the 1970s, Gebman and Mingori used perturbation methods to analyze control techniques for flat-spin recovery of dual-spin spacecraft in which actuation is provided by a motor applying internal torques parallel to the desired spin axis[65]. While these methods do not consume fuel, they have the disadvantage of leaving the spacecraft with residual nutation that must be damped out by some other means.

More recently, Lawrence and Holden have developed Lyapunov-based control laws for a spacecraft with a single reaction wheel mounted transverse to the desired spin axis[66]. While their method offers some asymptotic stability guarantees, their controller suffers from one potentially major drawback: It cannot determine

the sign of the final body-frame angular momentum vector. Depending on initial conditions, the spacecraft could find itself rotated 180° from the desired orientation.

The problem of effecting a 180° inversion of a spacecraft's spin axis has often been treated separately from flat-spin recovery. Rahn and Barba have proposed a method for "spin polarity control" using thrusters[67]. Beachey and Uicker describe a method for inverting the spin of an axisymmetric spinning spacecraft using a single reaction wheel[68]. Uicker has also proposed a method for inverting a spacecraft's spin axis by manipulating its mass distribution[69].

To the author's best knowledge, the first unified treatment of flat-spin recovery and spin inversion, in the form of a single controller capable of uniquely determining a spacecraft's final spin axis, was given by Myung and Bang[70]. They make use of a control technique known as predictive control[71] to derive a nonlinear feedback law for a spacecraft with a single reaction wheel. While they demonstrate numerically that their controller converges to the desired spin axis in many cases, they do not give strong stability guarantees. Additionally, their controller does not explicitly account for actuator limits and must be carefully tuned based on system parameters.

5.2 Background

This section provides brief reviews of gyrostat dynamics and Lyapunov stability. It also serves to introduce the notation and terminology used throughout the rest of the chapter. Thorough treatments of Lyapunov stability are given by Khalil[72] and Slotine and Li[73], while a thorough treatment of gyrostat dynamics is given by Hughes[28].

5.2.1 Gyrostat Dynamics

A gyrostat is a system of coupled rigid bodies whose relative motions do not change the total inertia tensor of the system. This abstraction serves as a practical mathematical model for a spacecraft with reaction wheels. The fundamental differential equation governing the motion of a gyrostat is[28],

$$\boldsymbol{I} \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\boldsymbol{I} \cdot \boldsymbol{\omega} + \boldsymbol{\rho}) + \dot{\boldsymbol{\rho}} = \boldsymbol{\tau} \tag{5.1}$$

where $\boldsymbol{I}$ is the symmetric positive-definite inertia tensor of the gyrostat, $\boldsymbol{\omega}$ is the body angular velocity, $\boldsymbol{\rho}$ is the total angular momentum stored in the rotors, and $\boldsymbol{\tau}$ is the external torque applied to the gyrostat. Assuming external torques can be neglected, the total system angular momentum,

$$\boldsymbol{h} = \boldsymbol{I} \cdot \boldsymbol{\omega} + \boldsymbol{\rho} \tag{5.2}$$

is conserved, and the gyrostat equation can be rewritten as,

$$\dot{\boldsymbol{h}} = \boldsymbol{h} \times \boldsymbol{J} \cdot (\boldsymbol{h} - \boldsymbol{\rho}) \tag{5.3}$$

where $\boldsymbol{J} = \boldsymbol{I}^{-1}$.

The dynamics of equation (5.3) have some properties that will be useful in the controller development to follow. First, because $\|\boldsymbol{h}\|$ is conserved, $\boldsymbol{h}$ is confined to the surface of a sphere, which will be referred to as the *momentum sphere*. Second, for a fixed value of $\boldsymbol{\rho}$, the solutions to equation (5.3) are periodic. In fact, $\boldsymbol{h}(t)$ can be expressed in closed form in terms of Jacobi elliptic functions[74], [75]. The details of these solutions will not be of concern in this chapter, only their existence and periodicity. Lastly, in the case where $\boldsymbol{\rho} = 0$, the uncontrolled system has, in general, six equilibria located along the eigenvectors of $\boldsymbol{J}$. The equilibrium points corresponding to the largest and smallest eigenvalues of $\boldsymbol{J}$ (commonly referred to as

the "minor" and "major" axes, respectively) are stable, while those corresponding to the intermediate eigenvalue are unstable[28].

Figure 5.1 depicts a momentum sphere with several trajectories marked in blue and the equilibria marked in green. Each of the four stable equilibria are

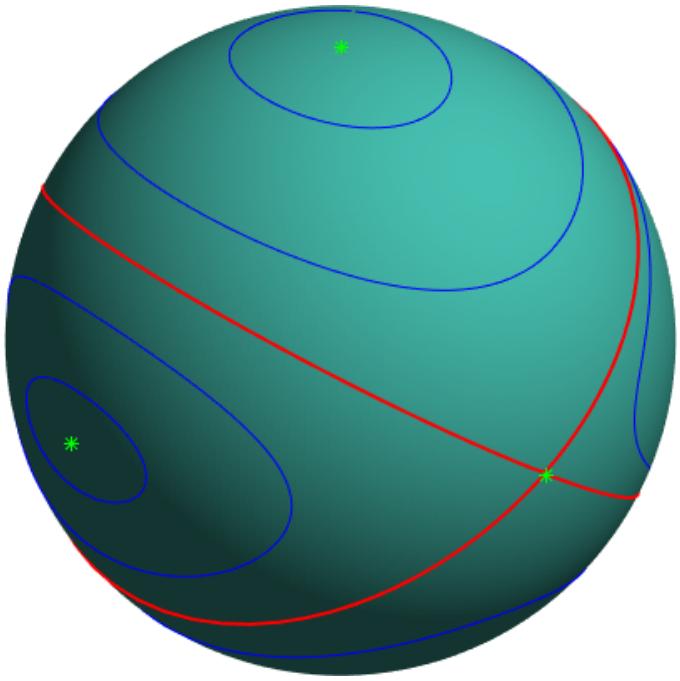


Figure 5.1: Momentum sphere with equilibria and example trajectories

surrounded by stable periodic orbits. The trajectories marked in red in figure 5.1 are known as *separatrices*[28]. They connect the two unstable equilibria and separate trajectories orbiting the other four equilibria.

5.2.2 Lyapunov Stability

Lyapunov stability theory provides a means for determining the stability of equilibrium points of nonlinear dynamical systems. The method provides a generalization of the concept of energy dissipation in damped mechanical systems. The necessary

ingredients are a system of the form,

$$\dot{x} = f(x) \tag{5.4}$$

where $x \in \mathbb{R}^n$, an equilibrium point x^* such that,

$$f(x^*) = 0 \tag{5.5}$$

and a scalar function $V(x)$ with the property,

$$V(x) \geq 0 \tag{5.6}$$

where $V(x) = 0$ only at the point $x = x^*$. If the derivative of $V(x)$ along trajectories of the system,

$$\dot{V}(x) = \frac{\partial V}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} = \nabla V(x) \cdot f(x) \tag{5.7}$$

is negative-semidefinite, the system is said to be stable *in the sense of Lyapunov*. If the stricter condition $\dot{V}(x) < 0$ everywhere except at $x = x^*$, the system is *asymptotically stable*. The main deficiency of Lyapunov stability theory is that there are no general methods for finding or constructing a function $V(x)$, which is known as a Lyapunov function.

For a dynamical system with control inputs,

$$\dot{x} = f(x, u) \tag{5.8}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$, a Lyapunov function can be used to find stabilizing control laws. In the controlled case, the stability condition becomes:

$$\dot{V}(x, u) = \nabla V(x) \cdot f(x, u) \leq 0 \tag{5.9}$$

A controller can then be found by, for example, solving the following optimization problem,

$$\begin{aligned} \underset{\boldsymbol{u}}{\text{argmin}} \quad & \nabla V(x) \cdot f(x, u) \\ \text{subject to} \quad & u \in \mathcal{U} \end{aligned} \tag{5.10}$$

where $\mathcal{U}$ is the set of possible control inputs. If u can be chosen such that $\dot{V}(x,u) < 0$ for every $x \neq x^*$, the closed loop system will be asymptotically stable.

5.3 Candidate Lyapunov Functions

As motivation, two candidate Lyapunov functions are considered. The first is a quadratic function based on kinetic energy,

$$V_Q = \frac{1}{2}\boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}_d - \frac{1}{2}\boldsymbol{h} \cdot \boldsymbol{J} \cdot \boldsymbol{h} \tag{5.11}$$

while the second is a linear function:

$$V_L = \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}_d - \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h} \tag{5.12}$$

Since $\boldsymbol{h}_d$ corresponds to a spin state with maximum kinetic energy, it is easy to show that both V_Q and V_L are non-negative everywhere on the momentum sphere, satisfying condition (5.6).

Looking first at V_Q and taking its derivative along trajectories of the system yields:

$$\dot{V}_Q = \frac{\partial V_Q}{\partial h}\frac{\mathrm{d}h}{\mathrm{d}t} = \boldsymbol{h} \cdot \boldsymbol{J} \cdot \boldsymbol{h} \times \boldsymbol{J} \cdot \boldsymbol{\rho} \tag{5.13}$$

As expected, when $\boldsymbol{\rho} = 0$, V_Q does not vary. With an appropriate choice of $\boldsymbol{\rho}$, the condition $\dot{V}_Q \leq 0$ can always be satisfied, indicating that the closed loop system will be stable. It must be noted, however, that V_Q is symmetric and possesses a pair of global minima located at $\boldsymbol{h} = \boldsymbol{h}_d$ and $\boldsymbol{h} = -\boldsymbol{h}_d$. As a result, the system can only be guaranteed to converge to one or the other of these equilibrium states, with the result depending on initial conditions. Figure 5.2 depicts a heat map of the function V_Q on the momentum sphere, with $\boldsymbol{h}_d$ marked in green.

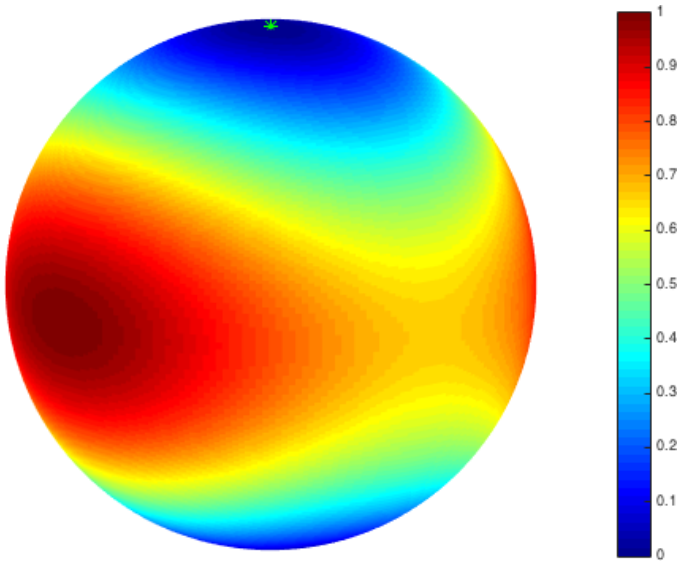


Figure 5.2: Heat map of candidate Lyapunov function V_Q

Unlike V_Q, V_L has the advantage of possessing a single global minimum on the momentum sphere at $\boldsymbol{h} = \boldsymbol{h}_d$, as can be seen in figure 5.3. Therefore, if a control law can be found which drives V_L to zero, the system is guaranteed to converge to $\boldsymbol{h}_d$. Unfortunately, V_L also suffers from several disadvantages. Evaluating its derivative with respect to time results in:

$$\dot{V}_L = \frac{\partial V_L}{\partial h}\frac{\mathrm{d}h}{\mathrm{d}t} = -\boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h} \times \boldsymbol{J} \cdot (\boldsymbol{h} - \boldsymbol{\rho}) \tag{5.14}$$

Note that, unlike what was encountered with V_Q, when $\boldsymbol{\rho} = 0$, $\dot{V}_L \neq 0$, meaning that V_L varies along uncontrolled system trajectories. Additionally, in the presence of actuator limits, where the maximum-possible rotor momentum $\|\boldsymbol{\rho}\|$ might be much smaller than $\|\boldsymbol{h}\|$, there may be situations in which the condition $\dot{V}_L \leq 0$ cannot be satisfied.

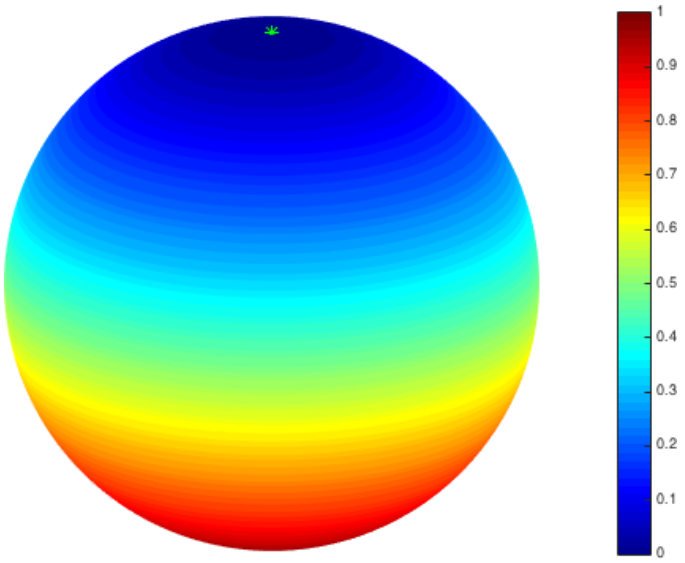


Figure 5.3: Heat map of candidate Lyapunov function V_L

5.4 Periodic Averaging

A variation on equation (5.12) will now be considered that mitigates some of the problems encountered in the previous section. A new candidate Lyapunov function V_L' is obtained by averaging V_L over periodic orbits of the uncontrolled system, where T is the nutation period:

$$V_L' = \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}_d - \frac{1}{T} \int_{t_0}^{t_0+T} \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}(t) \, \mathrm{d}t \tag{5.15}$$

It is worth noting that V_L' is a function only of the current state $\boldsymbol{h}$. While it does not have a convenient closed-form expression like V_L or V_Q, it could be, for example, precomputed over the entire momentum sphere and tabulated in a look-up table.

By construction, V_L' takes on the same value at every point along a given periodic orbit. Therefore, it does not vary along trajectories of the uncontrolled system. Also, like V_L, V_L' has a single global minimum at $\boldsymbol{h} = \boldsymbol{h}_d$, as shown in figure 5.4. Unfortunately, V_L' still suffers from one important deficiency: as can be understood based on simple symmetry considerations and seen in figure 5.4, V_L' is

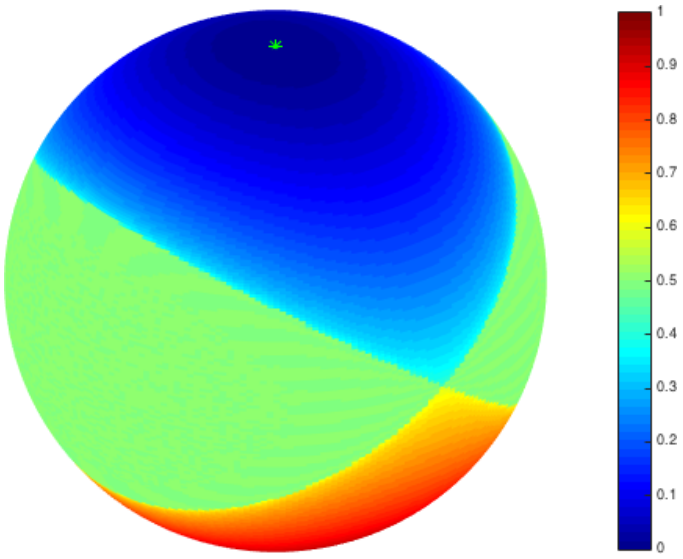


Figure 5.4: Heat map of candidate Lyapunov function V_L'

constant over the entirety of the regions surrounding the major-axis equilibria. As a result, $\dot{V}_L' = 0$, and the system will not converge to $\boldsymbol{h}_d$ from initial conditions in these regions.

A Lyapunov function will now be formed from the sum of V_Q and V_L', combining the behavior of the former in regions near the major-axis equilibria with the behavior of the latter in regions near the minor-axis equilibria:

$$V = \frac{3}{2}\,\boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}_d - \frac{1}{T}\int_{t_0}^{t_0+T} \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h}(t) + \frac{1}{2}\,\boldsymbol{h}(t) \cdot \boldsymbol{J} \cdot \boldsymbol{h}(t)\ \mathrm{d}t \tag{5.16}$$

As with equation (5.15), the integral in equation (5.16) is understood to be taken over periodic orbits of the uncontrolled system. Figure 5.5 shows a heat map of V on the momentum sphere with $\boldsymbol{h}_d$ marked in green.

An expression for $\dot{V}$ will now be sought. To aid in this process, the integral in equation (5.16) will be discretized. First, sampled versions of $\boldsymbol{h}$ and $\boldsymbol{\rho}$ are defined,

$$\boldsymbol{h}_k = \boldsymbol{h}(t_0 + k\delta_t) \tag{5.17}$$

$$\boldsymbol{\rho}_k = \boldsymbol{\rho}(t_0 + k\delta_t) \tag{5.18}$$

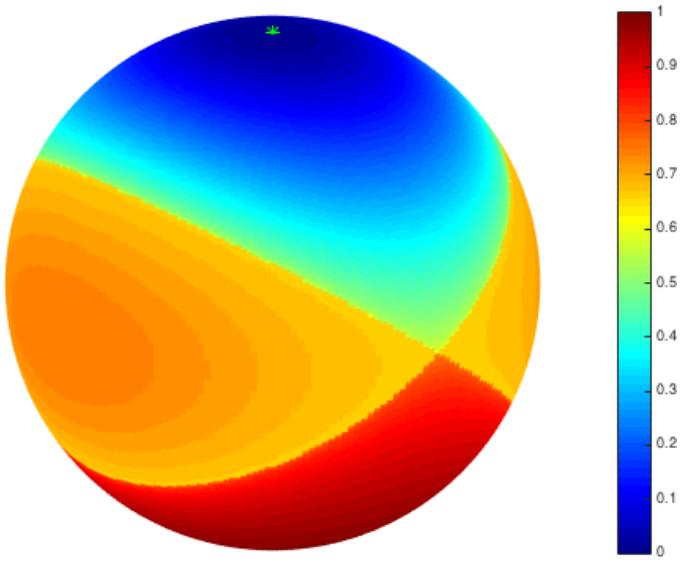


Figure 5.5: Heat map of Lyapunov function V

where $\delta_t = T/N$ and N is the number of samples. Next, the following matrices, which consist of uniform samples of the state and control inputs, are defined:

$$x_k = \begin{bmatrix} h_k \\ \vdots \\ h_{k+N-1} \end{bmatrix} \qquad u_k = \begin{bmatrix} \rho_k \\ \vdots \\ \rho_{k+N-1} \end{bmatrix} \tag{5.19}$$

In terms of x_k and u_k, the integral in equation (5.16) can be approximated as,

$$\int_{t_0}^{t_0+T} \boldsymbol{h}_d \cdot \boldsymbol{J} \cdot \boldsymbol{h} + \frac{1}{2}\, \boldsymbol{h} \cdot \boldsymbol{J} \cdot \boldsymbol{h} \; \mathrm{d}t \approx \frac{1}{N}\, x_d^\intercal\, \bar{J}\, x_k + \frac{1}{2N}\, x_k^\intercal\, \bar{J}\, x_k \tag{5.20}$$

where x_d is,

$$x_d = \begin{bmatrix} h_d \\ \vdots \\ h_d \end{bmatrix} \tag{5.21}$$

and $\bar{J}$ is the following block-diagonal matrix:

$$\bar{J} = \begin{bmatrix} J & 0 & \cdots & 0 \\ 0 & J & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J \end{bmatrix} \tag{5.22}$$

The next step in determining $\dot{V}$ is to calculate the following finite difference:

$$\Delta V = V(x_{k+1}) - V(x_k) = \frac{1}{N}\, x_d^\intercal\, \bar{J}\, x_k + \frac{1}{2N}\, x_k^\intercal\, \bar{J}\, x_k - \frac{1}{N}\, x_d^\intercal\, \bar{J}\, x_{k+1} - \frac{1}{2N}\, x_{k+1}^\intercal\, \bar{J}\, x_{k+1} \tag{5.23}$$

Thanks to periodicity, the system dynamics along an orbit can be approximated as,

$$x_{k+1} = Ax_k + B_k u_k \tag{5.24}$$

where A is a cyclic permutation matrix,

$$A = \begin{bmatrix} 0 & \mathbf{1}_{3\times3} & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{1}_{3\times3} & \vdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \mathbf{1}_{3\times3} \\ \mathbf{1}_{3\times3} & 0 & \cdots & 0 & 0 \end{bmatrix} \tag{5.25}$$

and B_k is the following block-diagonal matrix.

$$B_k = -\delta_t \begin{bmatrix} h_k^\times J & 0 & \cdots & 0 \\ 0 & h_{k+1}^\times J & 0 & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_{k+N-1}^\times J \end{bmatrix} \tag{5.26}$$

The expression $h^\times$ used in equation (5.26) denotes the formation of a 3×3 skew-symmetric cross-product matrix from the elements of h:

$$h^\times = \begin{bmatrix} 0 & -h_3 & h_2 \\ h_3 & 0 & -h_1 \\ -h_2 & h_1 & 0 \end{bmatrix} \tag{5.27}$$

Substituting equation (5.24) into equation (5.23) yields the following expression for ΔV:

$$\begin{aligned} \Delta V = {} & \frac{1}{N}\, x_d^\intercal\, \bar{J}\, x_k - \frac{1}{N}\, x_d^\intercal\, \bar{J}\, A\, x_k - \frac{1}{N}\, x_d^\intercal\, \bar{J}\, B_k\, u_k \\ & + \frac{1}{2N}\, x_k^\intercal\, \bar{J}\, x_k - \frac{1}{2N}\, x_k^\intercal\, A^\intercal\, \bar{J}\, A\, x_k - \frac{1}{N}\, x_k^\intercal\, A^\intercal\, \bar{J}\, B_k\, u_k - \frac{1}{2N}\, u_k^\intercal\, B_k^\intercal\, \bar{J}\, B_k\, u_k \end{aligned} \tag{5.28}$$

Because A is simply a permutation matrix, several terms in equation (5.28) cancel each other, resulting in:

$$\Delta V = -\frac{1}{N}\, x_d^\intercal\, \bar{J}\, B_k\, u_k - \frac{1}{N}\, x_k^\intercal\, A^\intercal\, \bar{J}\, B_k\, u_k - \frac{1}{2N}\, u_k^\intercal\, B_k^\intercal\, \bar{J}\, B_k\, u_k \tag{5.29}$$

Finally, $\dot{V}$ is recovered by taking a limit:

$$\dot{V} = \lim_{N\to\infty} \frac{\Delta V}{\delta_t} = \frac{1}{T}\int_{t_0}^{t_0+T} (\boldsymbol{h} + \boldsymbol{h}_d)\cdot \boldsymbol{J}\cdot \boldsymbol{h} \times \boldsymbol{J}\cdot \boldsymbol{\rho}\ \mathrm{d}t \tag{5.30}$$

5.5 Almost-Global Asymptotic Stability

The expression inside the integral in equation (5.30) is linear in the rotor momentum $\boldsymbol{\rho}$. To make this more explicit, the following vector is defined,

$$\boldsymbol{b}(\boldsymbol{h}) = -\boldsymbol{J}\cdot \boldsymbol{h} \times \boldsymbol{J}\cdot (\boldsymbol{h} + \boldsymbol{h}_d) \tag{5.31}$$

such that:

$$\boldsymbol{b}\cdot\boldsymbol{\rho} = (\boldsymbol{h} + \boldsymbol{h}_d)\cdot \boldsymbol{J}\cdot \boldsymbol{h} \times \boldsymbol{J}\cdot \boldsymbol{\rho} \tag{5.32}$$

In terms of $\boldsymbol{b}$, the problem of choosing a control input such that $\dot{V} \leq 0$ reduces to choosing $\boldsymbol{\rho}$ such that $\boldsymbol{b}\cdot\boldsymbol{\rho} \leq 0$.

Looking again at equation (5.30), it is clear that the condition for asymptotic stability, $\dot{V} < 0$, can be met as long as $\boldsymbol{b}\cdot\boldsymbol{\rho} < 0$ can be achieved at some point along every periodic orbit of the uncontrolled system. For the fully actuated case where the spacecraft has at least three reaction wheels, this is possible everywhere on the momentum sphere except at the single point $\boldsymbol{h} = -\boldsymbol{h}_d$, which is simultaneously an equilibrium point of the system and a point at which $\boldsymbol{b} = 0$. This failure is to be expected as, in general, constant (time-invariant) feedback laws cannot achieve global asymptotic stability for systems with rotational degrees of freedom[76]. In

practice, however, this does not present a problem, since $-\boldsymbol{h}_d$ is an unstable equilibrium of the closed-loop system, and perturbations will ensure that the system does not remain there.

5.6 Controller Implementation

If sufficient reaction wheel torque is available, wheel dynamics can effectively be ignored and direct control over $\boldsymbol{\rho}$ can be assumed. In this case, equation (5.31) can be applied directly to derive very simple controllers. One possible feedback law is,

$$\boldsymbol{\rho} = -k\boldsymbol{b} \tag{5.33}$$

where k is a scalar gain.

The optimal choice of $\boldsymbol{\rho}$, in the sense of decreasing V the fastest, can be found by minimizing $\boldsymbol{b} \cdot \boldsymbol{\rho}$, as in equation (5.10). In the particular case of a spacecraft with three reaction wheels, each aligned with an axis of the body coordinate frame, such a control law assumes the following form in body coordinates,

$$\rho = -\rho_{\max} \operatorname{sign}(b) \tag{5.34}$$

where $\rho_{\max}$ is the maximum wheel momentum and $\operatorname{sign}(b)$ denotes the element-wise signum function:

$$\operatorname{sign}(x) = \begin{cases} 1 & : x > 0 \\ 0 & : x = 0 \\ -1 & : x < 0 \end{cases} \tag{5.35}$$

In cases where reaction wheel torque is limited and the assumption of direct control over $\boldsymbol{\rho}$ cannot be made, a control law that specifies $\dot{\boldsymbol{\rho}}$ and respects torque limits is needed. One such controller can be derived by first defining a smoothed

version of the controller (5.34), where the signum function is replaced with a hyperbolic tangent:

$$\rho = -\rho_{\max} \tanh(\alpha b) \tag{5.36}$$

The scalar parameter α in equation (5.36) can be tuned to set the maximum allowable torque. Equation (5.36) is then differentiated with respect to time, which yields,

$$\dot{\rho} = \frac{\partial \rho}{\partial b}\frac{\partial b}{\partial h}\frac{\mathrm{d}h}{\mathrm{d}t} = -\rho_{\max}\alpha \,\mathrm{diag}(\mathrm{sech}^2(\alpha b))\, J\left[(J(h+h_d))^{\times} - h^{\times}J\right] h^{\times} J(h-\rho) \tag{5.37}$$

where $\mathrm{diag}(x)$ indicates the formation of a diagonal matrix from the elements of x:

$$\mathrm{diag}\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) = \begin{bmatrix} x_1 & 0 & 0 \\ 0 & x_2 & 0 \\ 0 & 0 & x_3 \end{bmatrix} \tag{5.38}$$

While equation (5.37) can, in principle provide explicit wheel torque commands, the stability results of the previous section are specified as conditions on the wheel momenta. Therefore, to ensure stability, an actual implementation should track the wheel momenta specified by equation (5.36) using, for example, PID motor controllers on each wheel.

5.7 Examples

This section presents two simulations that illustrate the performance of the control laws developed in section 5.6 in the presence of model uncertainty and actuator limits. Both simulations are performed using MATLAB's ODE45 variable-step Runge-Kutta solver[77] with default error tolerances. To demonstrate the robustness of the controllers to model uncertainty, the following nominal inertia is used

in the controller,

$$I = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.39}$$

while the true inertia used to simulate the dynamics differs by 5% in the the moments of inertia and 5° in the orientation of the principal axes:

$$I_{\text{true}} = \begin{bmatrix} 2.0961 & -0.0455 & 0.0299 \\ -0.0455 & 1.4271 & 0.0167 \\ 0.0299 & 0.0167 & 1.0517 \end{bmatrix} \tag{5.40}$$

Additionally, white noise is added to the simulated gyro measurements fed to the controller.

The first simulation illustrates the performance of the controller (5.36) in a flat-spin recovery maneuver on a spacecraft with three reaction wheels. The initial angular momentum is,

$$h_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{5.41}$$

and the desired final angular momentum is set to,

$$h_d = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{5.42}$$

which corresponds to an angular velocity of roughly 10 RPM about the minor axis. The maximum reaction wheel momentum is $\rho_{\max} = 0.01$ N·m·s, and the maximum reaction wheel torque is $\dot{\rho}_{\max} = 0.1$ N·m. The parameter α is set to a value of 60, which ensures that commanded torques are within the actuator saturation limits.

Figure 5.6 shows the closed-loop trajectory of the system, while figures 5.7–5.9 show the system momentum, reaction wheel momentum, and reaction wheel torque components, respectively. Rapid convergence to the desired minor axis-spin

is achieved. The residual error in the angular momentum components and non-zero steady-state reaction wheel momenta are due to the error in the inertia used in the controller.

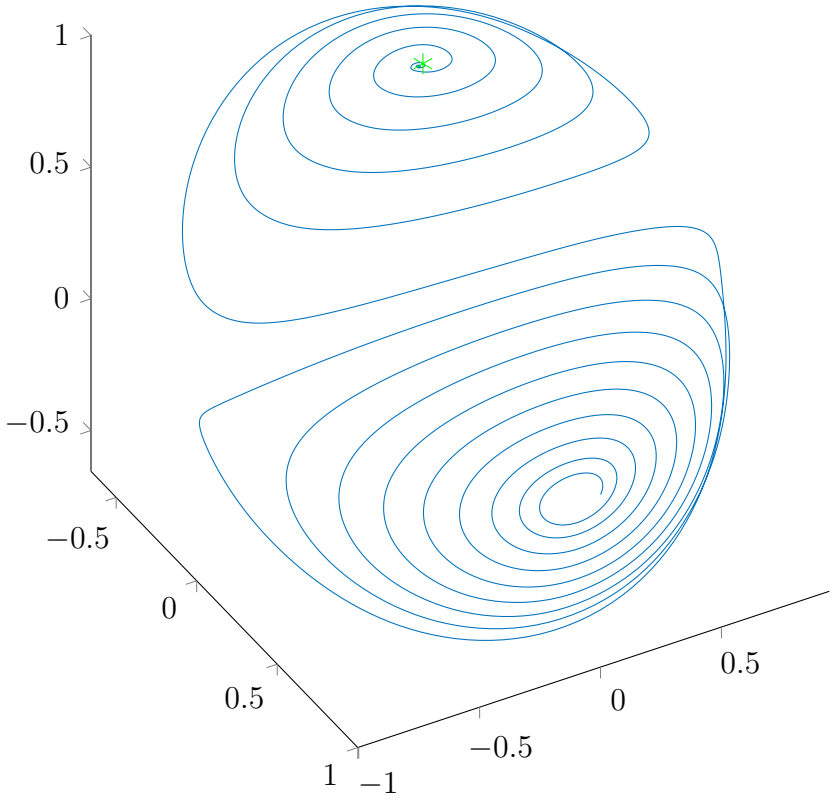


Figure 5.6: Flat-spin recovery trajectory

The second test case uses the same system and controller parameters but a different set of initial conditions to illustrate the control law's ability to perform spin inversion maneuvers. The initial angular momentum is,

$$h_0 = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \tag{5.43}$$

and the desired final angular momentum is again set to:

$$h_d = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{5.44}$$

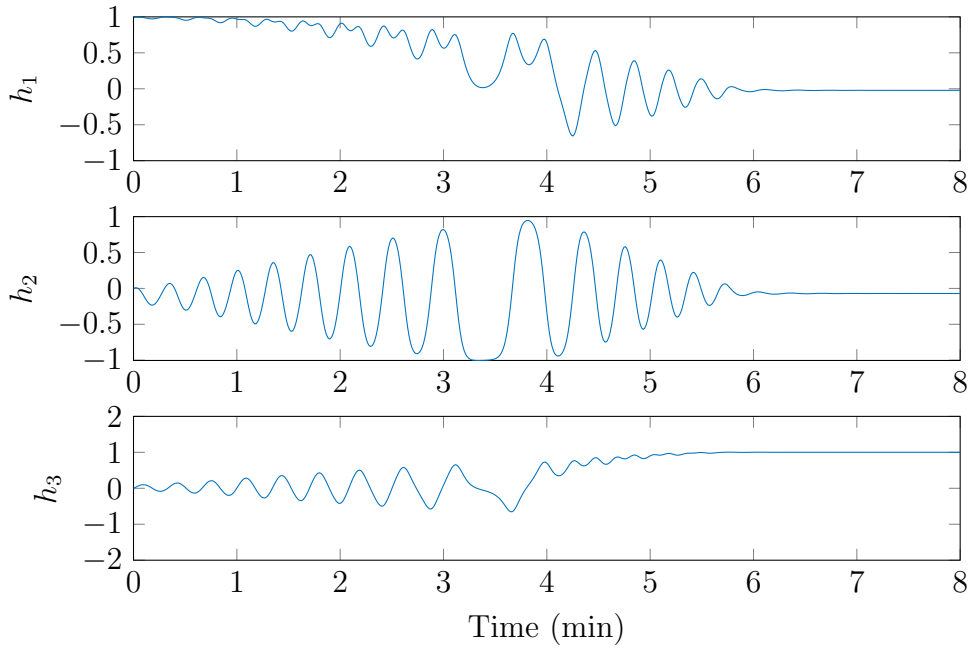


Figure 5.7: Flat-spin recovery momentum components

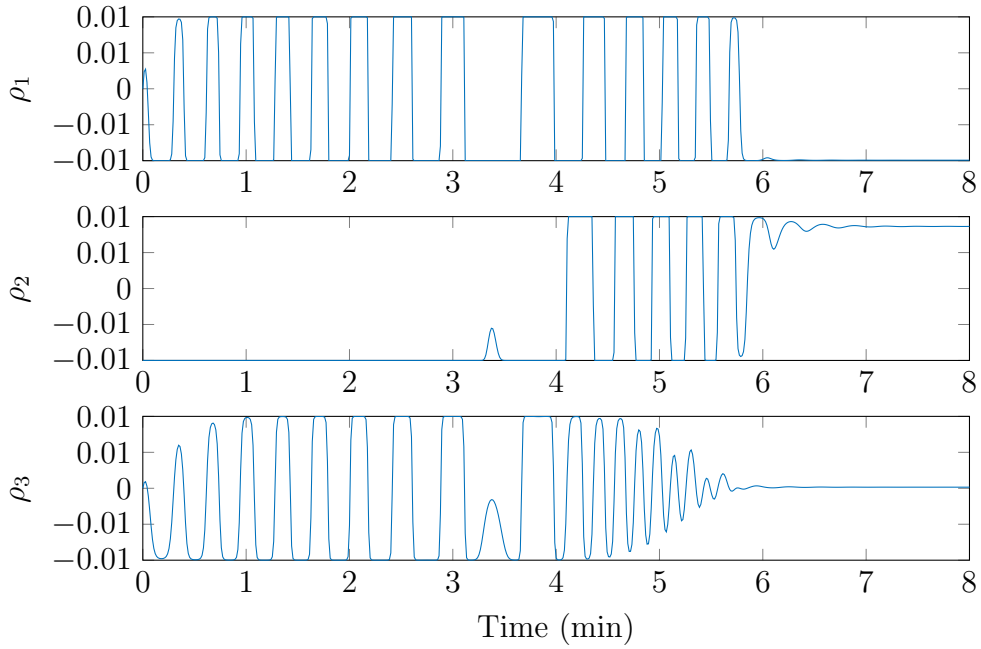


Figure 5.8: Flat-spin recovery reaction wheel momenta

Figure 5.10 shows the closed-loop trajectory of the system, while figures 5.11–5.13 show the system momentum, reaction wheel momentum, and reaction wheel

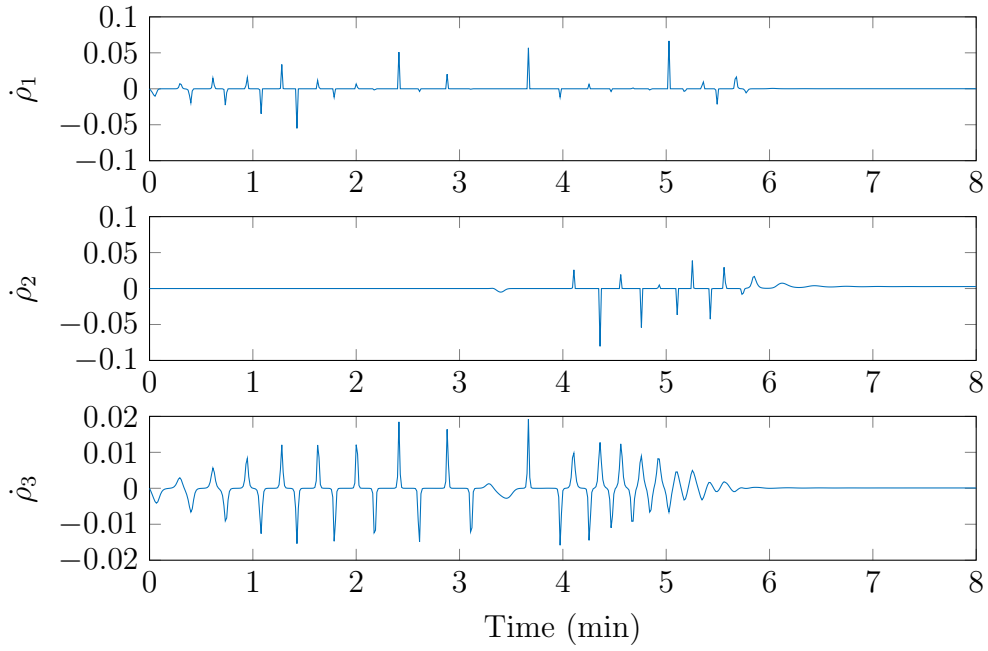


Figure 5.9: Flat-spin recovery reaction wheel torques

torque components, respectively. Once again, the controller demonstrates rapid convergence. This example also illustrates the behavior of the closed-loop system around the point $-\boldsymbol{h}_d$. As discussed in section 5.5, noise and modeling errors ensure that the system is perturbed away from this unstable equilibrium.

5.8 Conclusions

The control laws developed in this study provide a unified solution for both flat-spin recovery and spin inversion of spin-stabilized spacecraft. They are shown to be almost-globally asymptotically stabilizing using a Lyapunov function argument, and their performance is also demonstrated in the presence of model error and actuator limits through numerical simulation. Finally, the controllers have a simple mathematical form and are easy to tune, making them practical for implementation

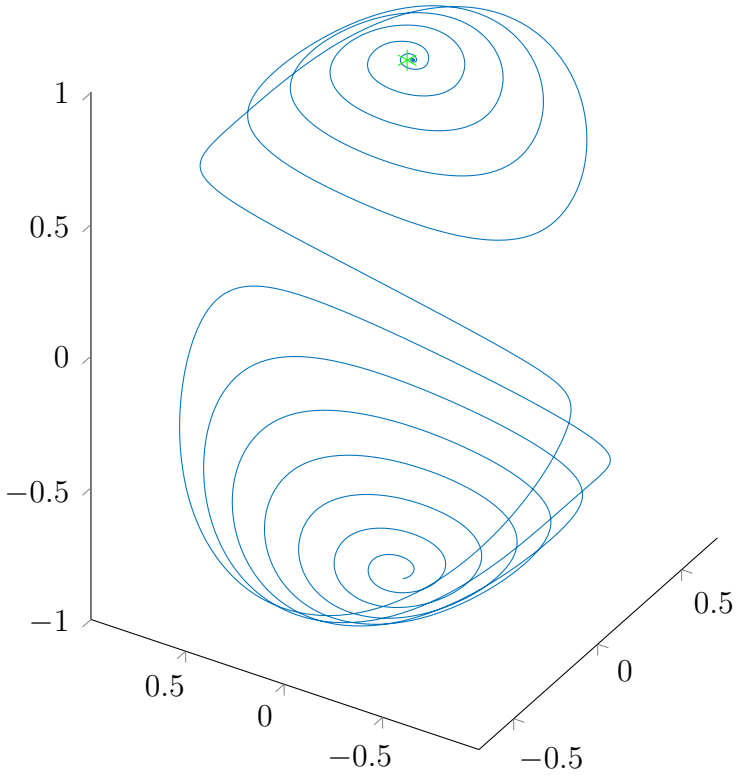


Figure 5.10: Spin inversion trajectory

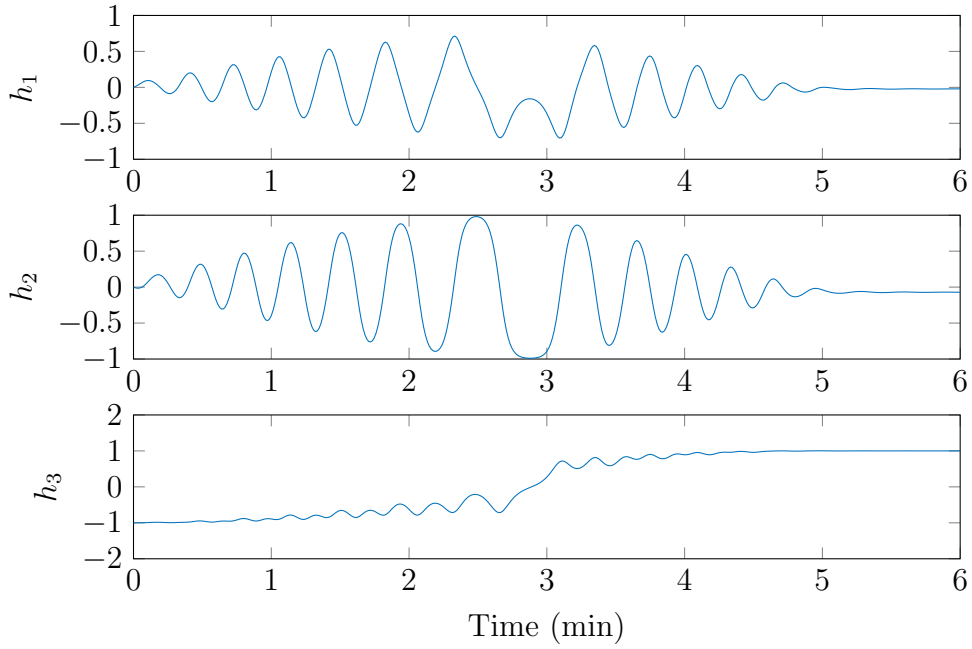


Figure 5.11: Spin inversion momentum components

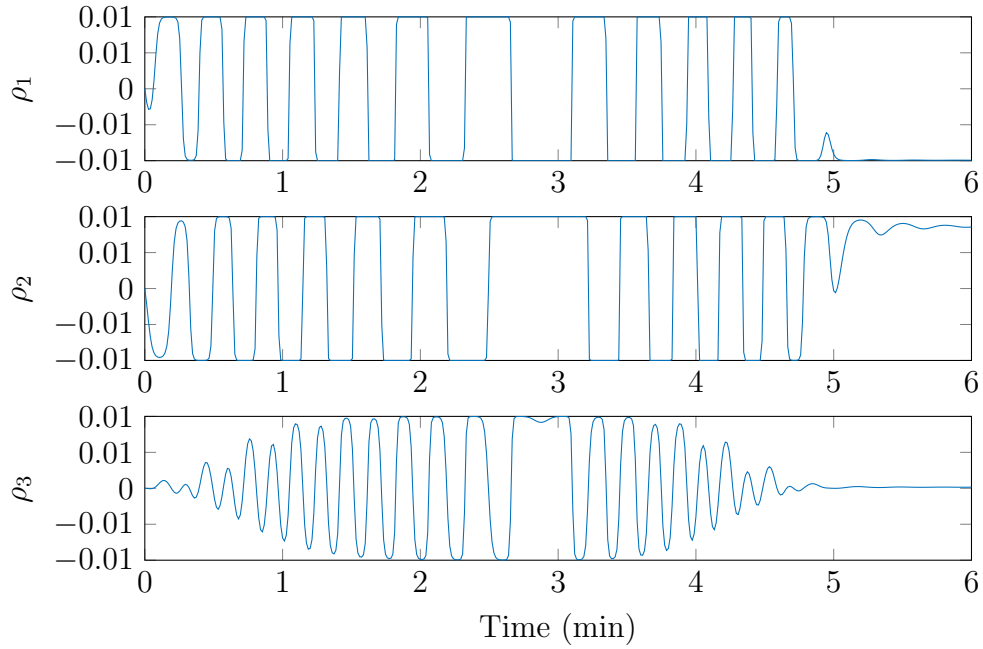


Figure 5.12: Spin inversion reaction wheel momenta

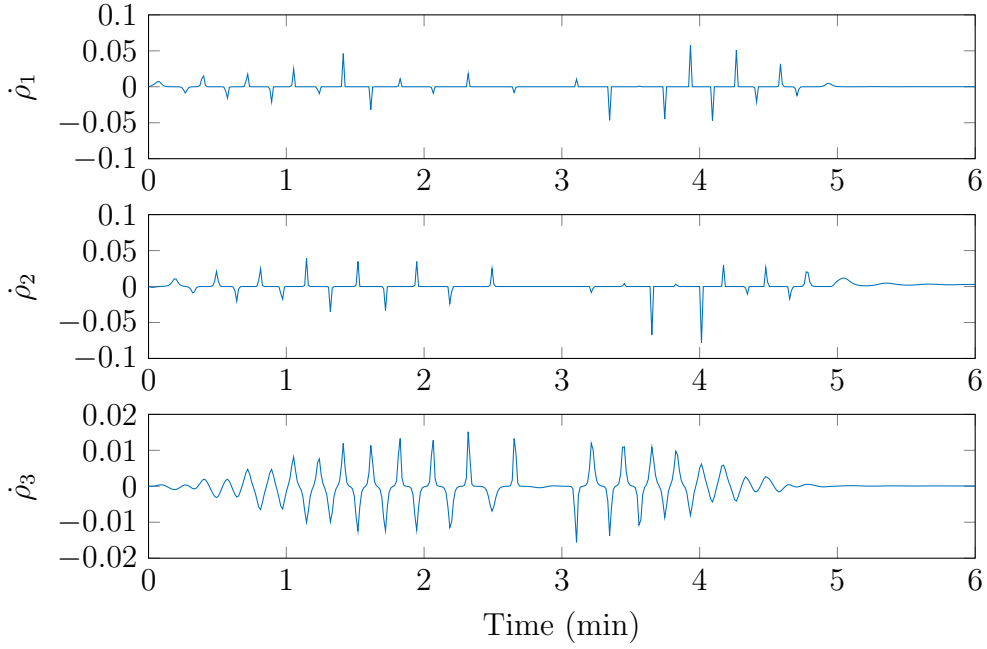


Figure 5.13: Spin inversion reaction wheel torques

in flight software.

CHAPTER 6

QUATERNION VARIATIONAL INTEGRATORS FOR SPACECRAFT DYNAMICS

Variational integrators have many advantages over Runge-Kutta methods and other traditional algorithms for numerically integrating equations of motion for mechanical systems. Rather than deriving differential equations of motion for a given system in continuous time, then discretizing them, variational approaches begin by discretizing the Lagrangian and the action integral for the system. The tools of variational mechanics are then used to derive discrete-time equations of motion. Integrators derived in this way retain many of the properties of the continuous system, such as momentum and energy conservation[49]. These integrators are also computationally efficient and stable, even for relatively large fixed time steps, making them well suited for use in real-time estimation and control applications.

The chapter proceeds as follows: Section 6.1 gives a brief survey of existing and related methods for numerically integrating the equations of motion for rigid body dynamics. Section 6.2 then gives a brief review of quaternions and outlines the notation used throughout the chapter. Section 6.3 derives the classical Euler equation of rigid body dynamics in continuous time using Hamilton's principle. Next, section 6.4 presents this derivation in discrete time, leading to the variational integrator presented in section 6.5. Sections 6.6–6.8 incorporate several extensions to the basic rigid body integrator, including reaction wheels, external torques, and internal damping. Finally, in section 6.9 several numerical examples are presented, including an extended Kalman filter for attitude determination.

6.1 Existing Work

Examples of integration methods that respect motion integrals of mechanical systems, often called geometric or symplectic integrators, have been known for decades. The classic example is the Verlet or Störmer-Verlet method[23], [78]. These early methods were often devised in ad hoc ways that do not generalize well to arbitrary mechanical systems or higher orders of accuracy. More recently, systematic methods for deriving symplectic integrators using discrete-time versions of ideas from variational mechanics have been introduced[49]. These methods are straightforward applications of Lagrangian and Hamiltonian dynamics, and they can generate integration schemes of any desired order of accuracy.

Discrete variational mechanics has previously been applied to problems in rigid body dynamics and the optimal control of rigid bodies using rotation matrices to parameterize attitude[79]–[81]. Momentum-preserving integrators have also been derived by other (non-variational) means for rigid body dynamics using quaternions to parameterize attitude[82]–[85]. The primary contribution of this chapter is the application of discrete variational mechanics to spacecraft attitude dynamics using quaternion state variables. The emphasis on quaternions over other attitude parameterizations here is due to both the compact and elegant derivations they enable and their prevalence in the implementation of spacecraft guidance, navigation, and control algorithms. Specifically, they lend themselves to straightforward feedback control and estimation schemes of practical relevance in flight software.

6.2 Background

Attitude dynamics and rotations are parameterized with quaternions throughout this chapter. A brief review of their properties is presented in this section. A more thorough treatment is given by Altmann[43].

Quaternions form an algebra with a non-commutative binary product operation. It is often convenient to think of them as four-dimensional objects composed of a three-dimensional vector part $\boldsymbol{v}$ and a scalar part s.

$$q = \begin{bmatrix} \boldsymbol{v} \\ s \end{bmatrix} \tag{6.1}$$

This representation allows the quaternion product to be written in terms of scalar and vector products:

$$q_1 q_2 = \begin{bmatrix} \boldsymbol{v_1} \times \boldsymbol{v_2} + s_1 \boldsymbol{v_2} + s_2 \boldsymbol{v_1} \\ s_1 s_2 - \boldsymbol{v_1} \cdot \boldsymbol{v_2} \end{bmatrix} \tag{6.2}$$

Note that $q_1 q_2 \neq q_2 q_1$. Throughout the chapter, quaternion products are indicated by juxtaposition, while scalar and vector products are indicated in the usual way, with the $\cdot$ and $\times$ symbols, respectively.

Rotations can be conveniently represented by unit-length quaternions. If $\boldsymbol{r}$ is a unit vector in $\mathbb{R}^3$ representing the axis of rotation and θ is the angle of rotation, then the quaternion representing the rotation is as follows:

$$q = \begin{bmatrix} \boldsymbol{r} \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} \tag{6.3}$$

Both q and $-q$ correspond to the same rotation, making the unit quaternions a "double cover" of the group of rotations.

The conjugate of a quaternion is denoted with a superscript $\dagger$ and represents

the rotation about the same axis $\boldsymbol{r}$ by $-\theta$.

$$q^\dagger = \begin{bmatrix} -\boldsymbol{v} \\ s \end{bmatrix} \tag{6.4}$$

Two rotations can be composed by multiplying their quaternion representations. A quaternion q_3 representing a rotation q_1 followed by a rotation q_2 is simply $q_3 = q_2 q_1$. The rotation of a three-dimensional vector $\boldsymbol{x}$ by a unit quaternion q is

$$\hat{\boldsymbol{x}}' = q\hat{\boldsymbol{x}}q^\dagger \tag{6.5}$$

where $\hat{\boldsymbol{x}}$ indicates the formation of a quaternion with zero scalar part from the vector $\boldsymbol{x}$:

$$\hat{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x} \\ 0 \end{bmatrix} \tag{6.6}$$

Analytic functions can be defined for quaternion arguments in much the same way as for complex numbers and matrices. In particular, the quaternion exponential has a simple closed-form expression in terms of the quaternion's scalar and vector parts:

$$e^q = \sum_{n=0}^{\infty} \frac{q^n}{n!} = e^s \begin{bmatrix} \frac{\boldsymbol{v}}{|\boldsymbol{v}|} \sin(|\boldsymbol{v}|) \\ \cos(|\boldsymbol{v}|) \end{bmatrix} \tag{6.7}$$

The formula for a rotation quaternion in equation (6.3) can be compactly written in terms of the exponential:

$$e^{\hat{\boldsymbol{r}}\theta/2} = \begin{bmatrix} \boldsymbol{r} \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} \tag{6.8}$$

Finally, in addition to the purely algebraic properties of quaternions outlined so far, the subsequent analysis requires some kinematic identities relating quaternion derivatives to vector quantities more familiar in rigid body dynamics. First, the time derivative of a body's attitude quaternion is related to its angular velocity in the following way:

$$\hat{\boldsymbol{\omega}} = 2q^\dagger \dot{q} \tag{6.9}$$

Second, the quaternion generalized force corresponding to a torque on the body is[45], [46]

$$\boldsymbol{F} = 2q\hat{\boldsymbol{\tau}} \tag{6.10}$$

Schaub and Junkins provide a thorough discussion of rigid body dynamics using quaternions[47].

6.3 Euler's Equation from Hamilton's Principle

This section presents a detailed derivation of the classical Euler equation using Hamilton's principle. While the results are not new, the techniques used provide the foundation for the development of the variational integrators in the following sections. A more in-depth treatment of variational mechanics on Lie groups, including the rotation group SO(3), is given by Holm[86].

The derivation begins with the Lagrangian for a free rigid body which, in the absence of a potential, is simply its kinetic energy:

$$L = \frac{1}{2}\boldsymbol{\omega} \cdot I \cdot \boldsymbol{\omega} = \frac{1}{2}\hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}} \tag{6.11}$$

J is the following augmented inertia matrix:

$$J = \begin{bmatrix} I_{11} & I_{12} & I_{13} & 0 \\ I_{21} & I_{22} & I_{23} & 0 \\ I_{31} & I_{32} & I_{33} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.12}$$

Following the standard approach in variational mechanics[32], [87], an action integral is constructed and its variational derivative is taken.

$$\delta S = \delta \int_{t_0}^{t_f} \frac{1}{2}\hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}} \, \mathrm{d}t = 0 \tag{6.13}$$

At this point one must be careful to take variations of $\boldsymbol{\omega}$ in such a way that the quaternion unit-norm constraint is maintained. There are several ways of explicitly enforcing the constraint in the action integral[46], [88]. An alternative is to incorporate the constraint into the variation[79], [80], [86]. From the fact that the exponential of a quaternion having zero scalar part is always a unit quaternion, a varied unit quaternion can be defined as

$$^{\epsilon}q = qe^{\epsilon\hat{\boldsymbol{\eta}}} \tag{6.14}$$

where a left superscript ϵ is used to denote a varied quantity. Next, $^{\epsilon}q$ is differentiated with respect to time.

$$^{\epsilon}\dot{q} = \dot{q}e^{\epsilon\hat{\boldsymbol{\eta}}} + \epsilon q e^{\epsilon\hat{\boldsymbol{\eta}}}\dot{\hat{\boldsymbol{\eta}}} \tag{6.15}$$

Equations (6.14) and (6.15) can be substituted into the identity in equation (6.9) to obtain the desired variation of $\boldsymbol{\omega}$, keeping in mind that only terms linear in ϵ need to be retained.

$$^{\epsilon}\hat{\boldsymbol{\omega}} = 2\,^{\epsilon}q^{\dagger}\,^{\epsilon}\dot{q} = e^{-\epsilon\hat{\boldsymbol{\eta}}}\,\hat{\boldsymbol{\omega}}\,e^{\epsilon\hat{\boldsymbol{\eta}}} + 2\epsilon\dot{\hat{\boldsymbol{\eta}}} \approx \hat{\boldsymbol{\omega}} + \epsilon(\hat{\boldsymbol{\omega}}\hat{\boldsymbol{\eta}} - \hat{\boldsymbol{\eta}}\hat{\boldsymbol{\omega}} + 2\epsilon\dot{\hat{\boldsymbol{\eta}}}) \tag{6.16}$$

Using equation (6.16), the variational derivative of the action integral in equation (6.13) is

$$\delta S = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0}\int_{t_0}^{t_f}\frac{1}{2}{}^{\epsilon}\hat{\boldsymbol{\omega}}\cdot J\cdot{}^{\epsilon}\hat{\boldsymbol{\omega}}\,\mathrm{d}t = \int_{t_0}^{t_f}\hat{\boldsymbol{\omega}}\cdot J\cdot(2\hat{\boldsymbol{\omega}}\hat{\boldsymbol{\eta}} + 2\dot{\hat{\boldsymbol{\eta}}})\,\mathrm{d}t = 0 \tag{6.17}$$

Following the usual procedure, integration by parts is used to eliminate $\dot{\hat{\boldsymbol{\eta}}}$, noting that variations must be zero at the endpoints of the integration interval.

$$\delta S = \int_{t_0}^{t_f} 2\hat{\boldsymbol{\omega}}\cdot J\cdot\hat{\boldsymbol{\omega}}\hat{\boldsymbol{\eta}} - 2\dot{\hat{\boldsymbol{\omega}}}\cdot J\cdot\hat{\boldsymbol{\eta}}\,\mathrm{d}t = 0 \tag{6.18}$$

Since all of the quaternions in equation (6.18) have scalar parts equal to zero, it can be converted to vector form:

$$\delta S = \int_{t_0}^{t_f}\boldsymbol{\omega}\cdot I\cdot(\boldsymbol{\omega}\times\boldsymbol{\eta}) - \dot{\boldsymbol{\omega}}\cdot I\cdot\boldsymbol{\eta}\,\mathrm{d}t = 0 \tag{6.19}$$

Using the fact that cyclically permuting the factors in a scalar triple product does not change its value, equation (6.19) can be rewritten as

$$\delta S = \int_{t_0}^{t_f} \boldsymbol{\eta} \cdot ((I \cdot \boldsymbol{\omega}) \times \boldsymbol{\omega}) - \dot{\boldsymbol{\omega}} \cdot I \cdot \boldsymbol{\eta} \, \mathrm{d}t = 0 \tag{6.20}$$

Finally, recognizing that equality must hold for all perturbations η results in Euler's equation.

$$I \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times I \cdot \boldsymbol{\omega} = 0 \tag{6.21}$$

6.4 A Discrete-Time Euler's Equation

This section derives an algebraic equation that is a discrete-time analogue of Euler's equation. The ideas used, collectively known as discrete mechanics, are presented in detail by Marsden and West[49]. The derivation here roughly follows that of Lee, Leok, and McClamroch[79] but uses quaternions where they have used rotation matrices.

The point of departure from classical mechanics is the action integral in equation (6.13). It is first broken into finite short segments of length h, with $t_k = t_0 + kh$.

$$S = \int_{t_0}^{t_f} \frac{1}{2} \hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}} \, \mathrm{d}t = \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} \frac{1}{2} \, \hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}} \, \mathrm{d}t \tag{6.22}$$

The integral of the Lagrangian over a single time step h inside the summation on the right hand side of equation (6.22) is known as the exact discrete Lagrangian[49].

$$L_d^E = \int_{t_k}^{t_{k+1}} \frac{1}{2} \, \hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}} \, \mathrm{d}t \tag{6.23}$$

The next step is to approximate L_d^E using a quadrature rule. Any quadrature rule for approximating integrals can be used for this purpose, with higher-order

rules generally leading to higher-order variational integrators[49]. The resulting approximation is known as the discrete Lagrangian of the system. In general, different quadrature rules lead to different discrete Lagrangians. For simplicity and clarity, the rectangle rule is used here. First, a finite difference approximation of $\boldsymbol{\omega}$ is defined.

$$\hat{\boldsymbol{\omega}}_k = 2\, q_k^\dagger \dot{q}_k \approx 2\, q_k^\dagger \left(\frac{q_{k+1} - q_k}{h} \right) = 2 \left(\frac{f_k - 1}{h} \right) \tag{6.24}$$

The quaternion rotation from q_k to q_{k+1} is denoted by $f_k = q_k^\dagger q_{k+1}$. Substituting the approximation for $\hat{\boldsymbol{\omega}}_k$ into equation (6.23), applying the rectangle rule, and simplifying leads to the following discrete Lagrangian:

$$L_d = \frac{2}{h}\, f_k \cdot J \cdot f_k \tag{6.25}$$

Using equation (6.25), the discrete action sum for the system can be formed:

$$S_d = \sum_{k=0}^{N-1} L_d = \sum_{k=0}^{N-1} \frac{2}{h}\, f_k \cdot J \cdot f_k \tag{6.26}$$

Equation (6.26) approximates equation (6.22) and serves the same role in discrete mechanics as the action integral does in traditional variational mechanics[49]. Analogously to the continuous case, Hamilton's Principle is applied to the action sum. First, a varied f_k that obeys the unit quaternion constraint is needed.

$${}^\epsilon f_k = {}^\epsilon q_k^\dagger {}^\epsilon q_{k+1} = e^{-\epsilon \hat{\boldsymbol{\eta}}} f_k e^{\epsilon \hat{\boldsymbol{\eta}}} \approx f_k + \epsilon (f_k \hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) \tag{6.27}$$

Using ${}^\epsilon f_k$, the variation of the action sum is set equal to zero.

$$\delta S_d = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \sum_{k=0}^{N-1} \frac{2}{h}\, {}^\epsilon f_k \cdot J \cdot {}^\epsilon f_k = \sum_{k=0}^{N-1} \frac{4}{h}\, f_k \cdot J \cdot (f_k \hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) = 0 \tag{6.28}$$

The next step is to eliminate $\boldsymbol{\eta}_{k+1}$ from the right hand side of equation (6.28) by performing the discrete equivalent of integration by parts, which amounts to some

simple index manipulation.

$$\delta S_d = f_{N-1} \cdot J \cdot f_{N-1} \hat{\boldsymbol{\eta}}_N - f_0 \cdot J \cdot f_0 \hat{\boldsymbol{\eta}}_0 + \sum_{k=1}^{N-1} \frac{4}{h} f_{k-1} \cdot J \cdot (f_{k-1} \hat{\boldsymbol{\eta}}_k - \hat{\boldsymbol{\eta}}_k f_k) = 0 \quad (6.29)$$

Using the fact that variations at the endpoints must be zero, just as in the continuous case, the first two terms in equation (6.29) can be eliminated.

$$\delta S_d = \sum_{k=1}^{N-1} \frac{4}{h} f_{k-1} \cdot J \cdot (f_{k-1} \hat{\boldsymbol{\eta}}_k - \hat{\boldsymbol{\eta}}_k f_k) = 0 \quad (6.30)$$

At this point equation (6.30), which implicitly includes unit-norm constraints on the quaternions, is converted to an unconstrained vector equation by parameterizing f_k in the following way:

$$f_k = \begin{bmatrix} \boldsymbol{\phi}_k \\ \sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k} \end{bmatrix} \quad (6.31)$$

This parameterization is only valid for $|\boldsymbol{\phi}_k| < 1$. Therefore, h must be chosen small enough to ensure that the incremental rotations between adjacent time steps are less than 180°. A number of other 3-parameter attitude representations could be used instead (modified Rodriguez parameters, for example), however, equation (6.31) is a natural choice that leads to simple and elegant expressions.

In terms of $\boldsymbol{\phi}_k$, equation (6.30) is

$$\sum_{k=1}^{N-1} \Bigg(\sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k}\, \boldsymbol{\eta}_k \cdot I \cdot \boldsymbol{\phi}_k - \boldsymbol{\phi}_k \cdot I \cdot (\boldsymbol{\phi}_k \times \boldsymbol{\eta}_k) \\ - \sqrt{1 - \boldsymbol{\phi}_{k-1} \cdot \boldsymbol{\phi}_{k-1}}\, \boldsymbol{\eta}_k \cdot I \cdot \boldsymbol{\phi}_{k-1} - \boldsymbol{\phi}_{k-1} \cdot I \cdot (\boldsymbol{\phi}_{k-1} \times \boldsymbol{\eta}_k) \Bigg) = 0 \quad (6.32)$$

Recognizing that equation (6.32) must be true for all $\boldsymbol{\eta}_k$ and performing some simple vector algebra reveals an algebraic equation relating $\boldsymbol{\phi}_k$, the incremental rotation from the last time step to the current time step, to $\boldsymbol{\phi}_{k+1}$, the incremental rotation from the current time step to the next time step.

$$\sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k}\, I \cdot \boldsymbol{\phi}_k - \boldsymbol{\phi}_k \times I \cdot \boldsymbol{\phi}_k = \sqrt{1 - \boldsymbol{\phi}_{k+1} \cdot \boldsymbol{\phi}_{k+1}}\, I \cdot \boldsymbol{\phi}_{k+1} + \boldsymbol{\phi}_{k+1} \times I \cdot \boldsymbol{\phi}_{k+1} \quad (6.33)$$

As a brief aside, equation (6.33) bears some resemblance to the classical Euler's equation. Taking its limit as h goes to zero does, in fact, recover equation (6.21). This result confirms that the discrete-time equation converges to the true differential equation for small time steps and establishes consistency with the continuous theory.

6.5 A Variational Integrator for the Free Rigid Body

This section uses equation (6.33) as the starting point for the development of a variational integrator for the unforced rigid body. The additional ingredients needed are a way to initialize the integrator given an attitude q_0 and angular velocity $\boldsymbol{\omega}_0$, a way to update the attitude q_{k+1} and angular velocity $\boldsymbol{\omega}_{k+1}$ after solving for $\boldsymbol{\phi}_{k+1}$, and a way to solve equation (6.33) for $\boldsymbol{\phi}_{k+1}$ given $\boldsymbol{\phi}_k$.

While it might seem simple enough to approximate $\boldsymbol{\phi}_0$ in any number of ways given $\boldsymbol{\omega}_0$, ad hoc approaches do not maintain the variational integrator's conservation properties. The discrete Legendre transform[49] gives a consistent way to convert between $\boldsymbol{\phi}_k$ and $\boldsymbol{\omega}_k$. Similar to the classical Legendre transform[32], it maps from $\boldsymbol{\phi}_k$ (which is effectively the discrete-time velocity variable) to $\boldsymbol{p}_k$, the momentum at time k, which can then be multiplied by I^{-1} to recover $\boldsymbol{\omega}_k$. Unlike the continuous version, there are actually two discrete Legendre transforms for a given time step[49]:

$$\boldsymbol{p}_k^- = -\frac{\partial L_d(q_k, q_{k+1})}{\partial q_k} \cdot \delta q_k \tag{6.34}$$

$$\boldsymbol{p}_k^+ = \frac{\partial L_d(q_k, q_{k+1})}{\partial q_{k+1}} \cdot \delta q_{k+1} \tag{6.35}$$

Applying these transformations to the discrete Lagrangian in equation (6.25) re-

veals that $\boldsymbol{p}_k^-$ and $\boldsymbol{p}_k^+$ correspond to the left and right sides of equation (6.33).

$$\boldsymbol{p}_k^- = \frac{2}{h}\sqrt{1-\boldsymbol{\phi}_k\cdot\boldsymbol{\phi}_k}\, I\cdot\boldsymbol{\phi}_k - \boldsymbol{\phi}_k \times I\cdot\boldsymbol{\phi}_k \tag{6.36}$$

$$\boldsymbol{p}_k^+ = \frac{2}{h}\sqrt{1-\boldsymbol{\phi}_{k+1}\cdot\boldsymbol{\phi}_{k+1}}\, I\cdot\boldsymbol{\phi}_{k+1} + \boldsymbol{\phi}_{k+1} \times I\cdot\boldsymbol{\phi}_{k+1} \tag{6.37}$$

This result leads to several key conclusions. First, equations (6.36) and (6.37) provide a new interpretation of the discrete-time equation of motion as a momentum balance between adjacent time steps. Second, equation (6.36) can be used to initialize the integrator by solving for $\boldsymbol{\phi}_0$ given I and $\boldsymbol{\omega}_0$. Lastly, $\boldsymbol{p}_k$, and hence $\boldsymbol{\omega}_k$, can be calculated at any point during the integration using either equation (6.36) or (6.37).

The final missing piece of the integrator is a method for solving equation (6.33), which is both implicit and nonlinear. Newton's method, which amounts to solving successive linear approximations of the equation until a desired level of accuracy is achieved[23], [89], provides an efficient solution in this case. The necessary linear approximation is the Jacobian matrix of equation (6.37)

$$\frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{\phi}_{k+1}} = \frac{2}{h}\left(\sqrt{1-\boldsymbol{\phi}_{k+1}^\intercal\boldsymbol{\phi}_{k+1}}\, I - \frac{I\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_{k+1}^\intercal}{\sqrt{1-\boldsymbol{\phi}_{k+1}^\intercal\boldsymbol{\phi}_{k+1}}} + \text{skew}(\boldsymbol{\phi}_{k+1})I - \text{skew}(I\boldsymbol{\phi}_{k+1})\right) \tag{6.38}$$

where skew($\boldsymbol{\phi}$) indicates the skew-symmetric matrix-multiplication equivalent of the cross product opperation.

$$\text{skew}\left(\begin{bmatrix}\boldsymbol{\phi}_1\\ \boldsymbol{\phi}_2\\ \boldsymbol{\phi}_3\end{bmatrix}\right) = \begin{bmatrix}0 & -\boldsymbol{\phi}_3 & \boldsymbol{\phi}_2\\ \boldsymbol{\phi}_3 & 0 & -\boldsymbol{\phi}_1\\ -\boldsymbol{\phi}_2 & \boldsymbol{\phi}_1 & 0\end{bmatrix} \tag{6.39}$$

Three or four Newton iterations are sufficient to reach machine precision in all of the examples presented in section 6.9 using standard 64 bit floating-point arithmetic. Once $\boldsymbol{\phi}_{k+1}$ is computed, the attitude can be updated by simple quaternion multiplication with the previous attitude $q_{k+1} = q_k f_k$.

In summary, given an inertia I and initial conditions q_0 and $\boldsymbol{\omega}_0$, the integrator is initialized by computing the momentum $\boldsymbol{p}_0 = I \cdot \boldsymbol{\omega}_0$ and an initial guess for $\boldsymbol{\phi}_0$. A reasonable guess is $\boldsymbol{\phi}_0 \approx \frac{h}{2}\boldsymbol{\omega}_0$. The true value of $\boldsymbol{\phi}_0$ is then calculated to machine precision using Newton's method with equations (6.37) and (6.38). From $\boldsymbol{\phi}_0$, $\boldsymbol{p}_1$ is calculated using equation (6.36), followed by $\boldsymbol{\omega}_1$ and q_1. The process is then repeated as desired.

Algorithm 6.1: Rigid Body Variational Integrator

function FREERIGIDBODY($I, q_0, \boldsymbol{\omega}_0, h, N$)
  $\boldsymbol{p} \leftarrow I \cdot \boldsymbol{\omega}_0$
  $\boldsymbol{\phi} \leftarrow \frac{h}{2}\boldsymbol{\omega}_0$ ▷ Initial Guess
  for $k = 1 : N$ **do**
    repeat ▷ Newton's Method
      $\boldsymbol{e} \leftarrow \sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}\, I \cdot \boldsymbol{\phi} + \boldsymbol{\phi} \times I \cdot \boldsymbol{\phi} - \frac{h}{2}\boldsymbol{p}$
      $J \leftarrow \sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}\, I - \frac{I\boldsymbol{\phi}\boldsymbol{\phi}^\intercal}{\sqrt{1-\boldsymbol{\phi}\cdot\boldsymbol{\phi}}} + \text{skew}(\boldsymbol{\phi})I - \text{skew}(I\boldsymbol{\phi})$
      $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - J^{-1}\boldsymbol{e}$
    until $|\boldsymbol{e}| <$ tolerance
    $f \leftarrow [\boldsymbol{\phi}^\intercal \sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}]^\intercal$
    $q_{k+1} \leftarrow q_k f$
    $p \leftarrow \frac{2}{h}(\sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}\, I \cdot \boldsymbol{\phi} - \boldsymbol{\phi} \times I \cdot \boldsymbol{\phi})$
    $\boldsymbol{\omega}_{k+1} \leftarrow I^{-1}\boldsymbol{p}$
  end for
  return $[q, \boldsymbol{\omega}]$
end function

6.6 Gyrostats

A gyrostat is a system of coupled rigid bodies whose relative motions do not change the total inertia tensor of the system[28]. A practical example is a rigid body with internal rotors or momentum actuators that can spin relative to the carrier body, such as a spacecraft with reaction wheels. Using the variational framework developed in the previous sections, both the classical equations of motion and a variational integrator are straightforward to derive.

The Lagrangian for a gyrostat system is

$$L = \frac{1}{2}\boldsymbol{\omega}_B \cdot I_B \cdot \boldsymbol{\omega}_B + \sum_{r=1}^{N} \frac{1}{2}(\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) \cdot I_r \cdot (\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) + \frac{1}{2} m_r (\boldsymbol{\omega}_B \times \boldsymbol{x}_r)^2 \quad (6.40)$$

where I_B and $\boldsymbol{\omega}_B$ are the carrier body's inertia tensor and angular velocity, the I_r are the rotor inertia tensors, the $\boldsymbol{\omega}_r$ are the rotor angular velocities relative to the carrier body, and the $\boldsymbol{x}_r$ are the rotor positions relative to the carrier body's center of mass. The Lagrangian can be simplified by introducing a modified body inertia I'_B which includes the rotor masses:

$$I'_B = I_B - \sum_{r=0}^{N} m_r \operatorname{skew}(\boldsymbol{x}_r)^2 \quad (6.41)$$

Substituting I'_B into equation (6.40) eliminates the last term, giving the simpler expression

$$L = \frac{1}{2}\boldsymbol{\omega}_B \cdot I'_B \cdot \boldsymbol{\omega}_B + \sum_{r=1}^{N} \frac{1}{2}(\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) \cdot I_r \cdot (\boldsymbol{\omega}_B + \boldsymbol{\omega}_r) \quad (6.42)$$

The rotor angular velocities $\boldsymbol{\omega}_r$ are treated as exogenous inputs to the system that can be set arbitrarily (e.g. by a controller), so variations need only be taken with respect to $\boldsymbol{\omega}_B$. This fact makes the derivation for the gyrostat nearly identical to the free rigid body. The only difference is that a few extra terms involving I_r

and $\boldsymbol{\omega}_r$ are carried through. Using ${}^{\epsilon}\boldsymbol{\omega}_B$ to vary the action and following the rest of the steps in section 6.3 results in the following differential equation:

$$I'_B \cdot \dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times I'_B \cdot \boldsymbol{\omega}_B + \sum_{r=1}^{N} I_r \cdot \dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times I_r \cdot \boldsymbol{\omega}_B + I_r \cdot \dot{\boldsymbol{\omega}}_r + \boldsymbol{\omega}_B \times I_r \cdot \boldsymbol{\omega}_r = 0 \quad (6.43)$$

Two new definitions help simplify equation (6.43). First, the gyrostat inertia I_G is

$$I_G = I'_B + \sum_{r=0}^{N} I_r = I_B + \sum_{r=0}^{N} I_r - m_r \,\mathrm{skew}(\boldsymbol{x}_r)^2 \quad (6.44)$$

Second, $\boldsymbol{\rho}$ is the total angular momentum stored in all the rotors.

$$\boldsymbol{\rho} = \sum_{r=1}^{N_r} I_r \cdot \boldsymbol{\omega}_r \quad (6.45)$$

Substituting I_G and $\boldsymbol{\rho}$ into equation (6.43) results in the classical equation of motion for the gyrostat[28]:

$$I_G \cdot \dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times (I_G \cdot \boldsymbol{\omega}_B + \boldsymbol{\rho}) + \dot{\boldsymbol{\rho}} = 0 \quad (6.46)$$

The steps involved in deriving the discrete-time equivalent of equation (6.46) are now briefly highlighted. If $\boldsymbol{\omega}_B$ is approximated as in equation (6.24) and the rectangle rule is used, the discrete Lagrangian for the gyrostat is

$$L_d = \frac{2}{h}\left(f_k \cdot J'_B \cdot f_k + \sum_{r=1}^{N} (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \cdot J_r \cdot (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \right) \quad (6.47)$$

where J'_B and J_r are the augmented 4×4 equivalents of I'_B and I_r. The discrete action sum S_d can then be formed as in equation (6.26) and its variation taken using ${}^{\epsilon}f_k$ from equation (6.27).

$$\delta \mathcal{S}_d = \sum_{k=0}^{N-1} \left(f_k \cdot J_B \cdot (f_k \hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) + \sum_{r=1}^{N} (f_k + \frac{h}{2}\hat{\boldsymbol{\omega}}_{r,k}) \cdot J_r \cdot (f_k \hat{\boldsymbol{\eta}}_{k+1} - \hat{\boldsymbol{\eta}}_k f_k) \right) = 0 \quad (6.48)$$

Following the rest of the steps in section 6.4 and substituting in I_G and $\boldsymbol{\rho}$ results in the discrete-time gyrostat equation:

$$\begin{aligned}&\sqrt{1-\boldsymbol{\phi}_k\cdot\boldsymbol{\phi}_k}\,(I_G\cdot\boldsymbol{\phi}_k+\frac{h}{2}\boldsymbol{\rho}_k)-\boldsymbol{\phi}_k\times(I_G\cdot\boldsymbol{\phi}_k+\frac{h}{2}\boldsymbol{\rho}_k)\\&=\sqrt{1-\boldsymbol{\phi}_{k+1}\cdot\boldsymbol{\phi}_{k+1}}\,(I_G\cdot\boldsymbol{\phi}_{k+1}+\frac{h}{2}\boldsymbol{\rho}_{k+1})+\boldsymbol{\phi}_{k+1}\times(I_G\cdot\boldsymbol{\phi}_{k+1}+\frac{h}{2}\boldsymbol{\rho}_{k+1})\end{aligned}\quad(6.49)$$

Equation (6.49) can be directly substituted into the variational integrator developed in section 6.5. The only other change necessary is to the Jacobian in the Newton iteration:

$$\begin{aligned}\frac{\partial\boldsymbol{p}^+}{\partial\boldsymbol{\phi}_{k+1}}=\frac{2}{h}\Bigg(&\sqrt{1-\boldsymbol{\phi}^\intercal\boldsymbol{\phi}}\,I-\frac{I\boldsymbol{\phi}\boldsymbol{\phi}^\intercal}{\sqrt{1-\boldsymbol{\phi}^\intercal\boldsymbol{\phi}}}+\operatorname{skew}(\boldsymbol{\phi})I\\&-\operatorname{skew}(I\boldsymbol{\phi})-\frac{h}{2}\frac{\boldsymbol{\rho}\boldsymbol{\phi}^\intercal}{\sqrt{1-\boldsymbol{\phi}^\intercal\boldsymbol{\phi}}}-\frac{h}{2}\operatorname{skew}(\boldsymbol{\rho})\Bigg)\end{aligned}\quad(6.50)$$

6.7 External Torques

External torques can be incorporated into the variational framework using the Lagrange-D'Alembert principle (often known simply as D'Alembert's principle)[32], [87]. In particular, its integral form[49], [51]

$$\delta\int_{t_0}^{t_f}L\,\mathrm{d}t+\int_{t_0}^{t_f}\boldsymbol{F}\cdot\delta q\,\mathrm{d}t=0\quad(6.51)$$

is most readily applied here, where the term on the left is simply the variation of the action δS and the term on the right is the integral of the virtual work done by a generalized force $\boldsymbol{F}$.

To apply the Lagrange-D'Alembert principle to a rigid body, the expression for the quaternion generalized force given in equation (6.10), as well as the variational

derivative of the attitude quaternion δq, are substituted into the second term of equation (6.51).

$$\int_{t_0}^{t_f} \boldsymbol{F} \cdot \delta q \, \mathrm{d}t = \int_{t_0}^{t_f} 2q\hat{\boldsymbol{\tau}} \cdot (q\hat{\boldsymbol{\eta}}) \, \mathrm{d}t \tag{6.52}$$

A little algebra reveals that $(q\hat{\boldsymbol{\tau}}) \cdot (q\hat{\boldsymbol{\eta}}) = \boldsymbol{\tau} \cdot \boldsymbol{\eta}$, further simplifying the expression. Combining this result with the action term from equation (6.18) leads to the following equation:

$$\int_{t_0}^{t_f} 2\hat{\boldsymbol{\omega}} \cdot J \cdot \hat{\boldsymbol{\omega}}\hat{\boldsymbol{\eta}} - 2\dot{\hat{\boldsymbol{\omega}}} \cdot J \cdot \hat{\boldsymbol{\eta}} + 2\hat{\boldsymbol{\tau}} \cdot \hat{\boldsymbol{\eta}} \, \mathrm{d}t = 0 \tag{6.53}$$

It is then straightforward to work through the rest of the steps in section 6.3 to arrive at the forced Euler equation:

$$I \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times I \cdot \boldsymbol{\omega} = \boldsymbol{\tau} \tag{6.54}$$

Incorporating forcing into the discrete variational framework is a bit more subtle than in the continuous case. The discrete version of the Lagrange-D'Alembert principle is

$$\delta \sum_{k=0}^{N} L_d + \sum_{k=0}^{N} \boldsymbol{F}_d^- \cdot \delta q_k + \boldsymbol{F}_d^+ \cdot \delta q_{k+1} = 0 \tag{6.55}$$

where $\boldsymbol{F}_d^-$ and $\boldsymbol{F}_d^+$ are known as discrete generalized forces[49]. Similar to what is encountered with the discrete Legendre transform, there are two discrete generalized forces corresponding to the beginning and end of a time step:

$$\boldsymbol{F}_d^- = \int_{t_k}^{t_{k+1}} \frac{1}{2} \boldsymbol{F}(q, \dot{q}) \cdot \frac{\partial q(t)}{\partial q_k} \, \mathrm{d}t \tag{6.56}$$

$$\boldsymbol{F}_d^+ = \int_{t_k}^{t_{k+1}} \frac{1}{2} \boldsymbol{F}(q, \dot{q}) \cdot \frac{\partial q(t)}{\partial q_{k+1}} \, \mathrm{d}t \tag{6.57}$$

As with the discrete Lagrangian, the integrals in the definition of the discrete generalized forces are approximated by quadrature. Here the rectangle rule is used,

though more accurate quadrature rules can lead to more accurate integrators at the expense of increased computational burden.

$$\boldsymbol{F}_d^- \approx hq_k\hat{\boldsymbol{\tau}}_k \tag{6.58}$$

$$\boldsymbol{F}_d^+ \approx hq_{k+1}\hat{\boldsymbol{\tau}}_{k+1} \tag{6.59}$$

Substituting the approximations for $\boldsymbol{F}_d^-$ and $\boldsymbol{F}_d^+$, as well as the discrete Lagrangian for the rigid body from equation (6.25), into equation (6.55) results in

$$\delta\sum_{k=0}^{N}\frac{2}{h}f_k\cdot J\cdot f_k+\sum_{k=0}^{N}hq_k\hat{\boldsymbol{\tau}}_k\cdot\delta q_k+hq_{k+1}\hat{\boldsymbol{\tau}}_{k+1}\cdot\delta q_{k+1}=0 \tag{6.60}$$

Carrying out the variational derivatives leads to the following:

$$\sum_{k=0}^{N}\frac{4}{h}f_k\cdot J\cdot(f_k\boldsymbol{\eta}_{k+1}-\boldsymbol{\eta}_k f_k)+h\hat{\boldsymbol{\tau}}_k\cdot\hat{\boldsymbol{\eta}}_k+h\hat{\boldsymbol{\tau}}_{k+1}\cdot\hat{\boldsymbol{\eta}}_{k+1}=0 \tag{6.61}$$

Eliminating the $\boldsymbol{\eta}_{k+1}$ terms again requires a "discrete integration by parts." Manipulating indices results in

$$\sum_{k=1}^{N}\frac{4}{h}f_{k-1}\cdot J\cdot(f_{k-1}\boldsymbol{\eta}_k-\boldsymbol{\eta}_k f_k)+2h\hat{\boldsymbol{\tau}}_k\cdot\hat{\boldsymbol{\eta}}_k=0 \tag{6.62}$$

Retracing the remaining steps in section 6.4 yields the discrete-time equation of motion for the forced rigid body:

$$\begin{aligned}\sqrt{1-\boldsymbol{\phi}_k\cdot\boldsymbol{\phi}_k}\,I\cdot\boldsymbol{\phi}_k-\boldsymbol{\phi}_k\times I\cdot\boldsymbol{\phi}_k&\\ =\sqrt{1-\boldsymbol{\phi}_{k+1}\cdot\boldsymbol{\phi}_{k+1}}\,I\cdot\boldsymbol{\phi}_{k+1}+\boldsymbol{\phi}_{k+1}\times I\cdot\boldsymbol{\phi}_{k+1}+\frac{h^2}{2}\boldsymbol{\tau}_{k+1}&\end{aligned} \tag{6.63}$$

This result can also be readily applied to the forced gyrostat by adding the same torque term onto the right hand side of equation (6.49).

6.8 A Gyrostat Spacecraft With Damping

The tools developed up to this point enable the construction of a variational integrator for a gyrostat with an internal energy-dissipating mechanism. The mecha-

nism considered here is known as a Kane damper and consists of a spherical mass immersed in a viscous fluid inside a spherical cavity in the spacecraft body[90]. The torque exerted on the spacecraft by the damper is

$$\boldsymbol{\tau} = C(\boldsymbol{\omega}_D - \boldsymbol{\omega}_B) \tag{6.64}$$

where C is a damping constant, $\boldsymbol{\omega}_D$ is the damper angular velocity, and $\boldsymbol{\omega}_B$ is the body angular velocity.

The basic approach taken here is to treat the body and damper as separate rigid bodies coupled through the viscous damping force. Equation (6.64) is approximated by finite differences in the usual way, giving

$$\boldsymbol{\tau}_k \approx \frac{2}{h} C(\boldsymbol{\gamma}_k - \boldsymbol{\phi}_k) \tag{6.65}$$

where $\boldsymbol{\phi}_k$ is the incremental body rotation defined in equation (6.31) and $\boldsymbol{\gamma}_k$ is the analogous incremental rotation for the damper. Substituting this approximation into equation (6.63) yields a set of coupled equations for the gyrostat-damper system:

$$\begin{aligned}\sqrt{1 - \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k}\,(I_G \cdot \boldsymbol{\phi}_k + \frac{h}{2}\boldsymbol{\rho}_k) - \boldsymbol{\phi}_k \times (I_G \cdot \boldsymbol{\phi}_k + \frac{h}{2}\boldsymbol{\rho}_k) \\ = \sqrt{1 - \boldsymbol{\phi}_{k+1} \cdot \boldsymbol{\phi}_{k+1}}\,(I_G \cdot \boldsymbol{\phi}_{k+1} + \frac{h}{2}\boldsymbol{\rho}_{k+1}) + \boldsymbol{\phi}_{k+1} \times (I_G \cdot \boldsymbol{\phi}_{k+1} + \frac{h}{2}\boldsymbol{\rho}_{k+1}) \\ + hC(\boldsymbol{\gamma}_{k+1} - \boldsymbol{\phi}_{k+1})\end{aligned} \tag{6.66}$$

$$\sqrt{1 - \boldsymbol{\gamma}_k \cdot \boldsymbol{\gamma}_k}\, I_D = \sqrt{1 - \boldsymbol{\gamma}_{k+1} \cdot \boldsymbol{\gamma}_{k+1}}\, I_D \cdot \boldsymbol{\gamma}_{k+1} - hC(\boldsymbol{\gamma}_{k+1} - \boldsymbol{\phi}_{k+1}) \tag{6.67}$$

These equations take advantage of the fact that the damper is spherical, and thus has an inertia tensor that is a scalar multiple of the identity, to eliminate the cross product term in equation (6.67).

Equations (6.66) and (6.67) must be solved simultaneously for $\boldsymbol{\phi}_{k+1}$ and $\boldsymbol{\gamma}_{k+1}$. Once again, Newton's method is used, this time with both equations combined to

form a single six-dimensional system. The necessary 6×6 Jacobian matrix is easily derived in terms of equations (6.38) and (6.50). In addition to the steps outlined in section 6.5, a subtlety arises in the implementation of this integrator in that the components of the damper's angular-momentum vector must be rotated by f_k at the end of each time step to keep them aligned with the spacecraft body frame.

6.9 Numerical Examples

This section presents some numerical examples to demonstrate the performance of the variational integrators derived in sections 6.4-6.8. Comparisons are made to the 2nd order fixed-step midpoint rule and MATLAB's ODE45 and ODE15s variable-step Runge-Kutta solvers with default error tolerances[77]. The computational cost of the midpoint rule roughly equals that of the variational integrators. In all simulations, the following inertia matrix is used:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{6.68}$$

6.9.1 Free Rigid Body

The first test compares the energy and momentum behavior of the integrator in section 6.5 with the midpoint rule and ODE45 by simulating a free rigid body with an initial angular velocity $\boldsymbol{\omega}_0 = [\pi/4,\, -\pi/5,\, \pi/6]^\intercal$ radians per second. The time steps for the midpoint rule and the variational integrator are chosen to be $h = .2$ seconds to make the run time for both roughly equal to that of ODE45. Since this system is conservative, both the inertial angular momentum vector components and the total energy should remain constant throughout the simulation.

Algorithm 6.2: Damped Gyrostat Integrator

function DAMPEDGYROSTAT($I, q_0, \boldsymbol{\omega}_0, I_D, C, \boldsymbol{\omega}_{D0}, \boldsymbol{\rho}, h, N$)
  $\boldsymbol{p} \leftarrow I \cdot \boldsymbol{\omega}_0$
  $\boldsymbol{p}_D \leftarrow I_D \cdot \boldsymbol{\omega}_{D0}$
  $\boldsymbol{\phi} \leftarrow \frac{h}{2}\boldsymbol{\omega}_0$ ▷ Initial Guess
  $\boldsymbol{\gamma} \leftarrow \frac{h}{2}\boldsymbol{\omega}_{D0}$ ▷ Initial Guess
  for $k = 1 : N$ **do**
    repeat ▷ Newton's Method
      $\boldsymbol{e} \leftarrow$ Calculate using equations (6.66) and (6.67)
      $J \leftarrow$ Jacobian based on equations (6.38) and (6.50)
      $[\boldsymbol{\phi}^\intercal \boldsymbol{\gamma}^\intercal]^\intercal \leftarrow [\boldsymbol{\phi}^\intercal \boldsymbol{\gamma}^\intercal]^\intercal - J^{-1}\boldsymbol{e}$
    until $|\boldsymbol{e}| <$ tolerance
    $f \leftarrow [\boldsymbol{\phi}^\intercal \sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}]^\intercal$
    $q_{k+1} \leftarrow q_k f$
    $\boldsymbol{p} \leftarrow \frac{2}{h}(\sqrt{1 - \boldsymbol{\phi} \cdot \boldsymbol{\phi}}\, I \cdot \boldsymbol{\phi} - \boldsymbol{\phi} \times I \cdot \boldsymbol{\phi})$
    $\boldsymbol{\omega}_{k+1} \leftarrow I^{-1}\boldsymbol{p}$
    $\boldsymbol{p}_D \leftarrow \frac{2}{h}(\sqrt{1 - \boldsymbol{\gamma} \cdot \boldsymbol{\gamma}}\, I_D \cdot \boldsymbol{\gamma})$
    $\boldsymbol{p}_D \leftarrow f^\dagger \boldsymbol{p}_D f$
    $\boldsymbol{\omega}_{D\,k+1} \leftarrow I_D^{-1}\boldsymbol{p}_D$
  end for
  return $[q, \boldsymbol{\omega}]$
end function

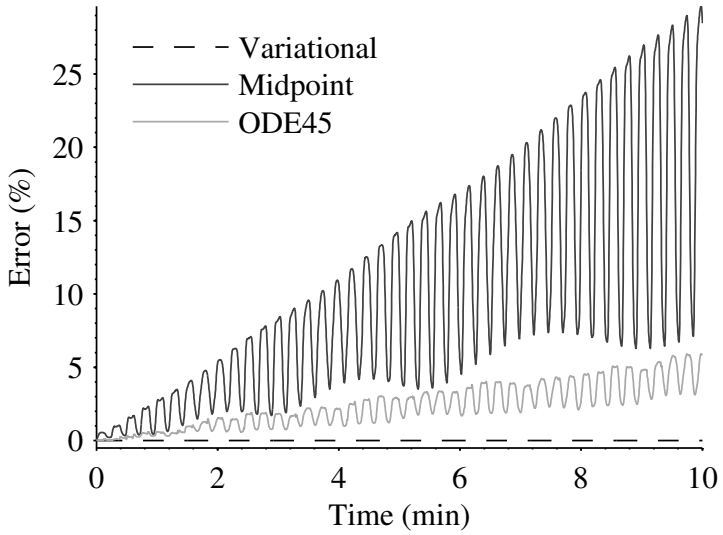


Figure 6.1: Momentum error for a free rigid body

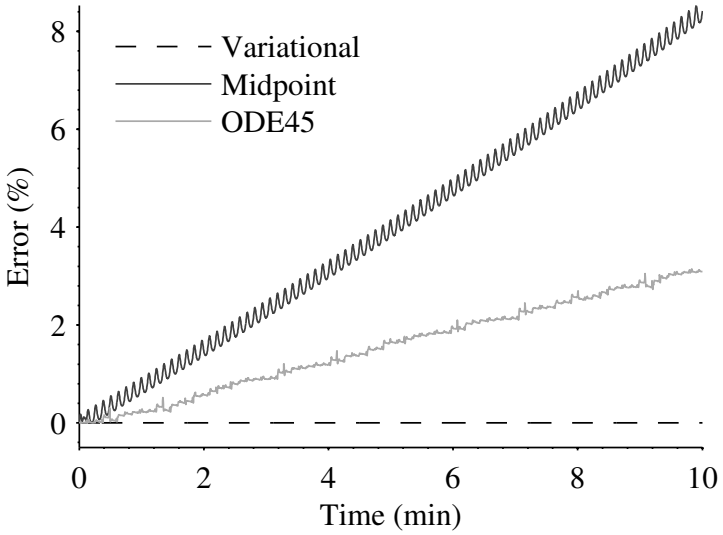


Figure 6.2: Energy error for a free rigid body

Figure 6.9.1 shows the normalized momentum error magnitude for all three integrators, and figure 6.2 shows the normalized energy error. Figure 6.3 shows these errors for the variational integrator after continuing the simulation for one million time steps. Together, all three demonstrate the excellent conservation properties

and long-term stability of the variational integrator. The gradual accumulation of error shown in figure 6.3 is due to numerical round-off, and is an unavoidable consequence of using finite-precision floating-point arithmetic.

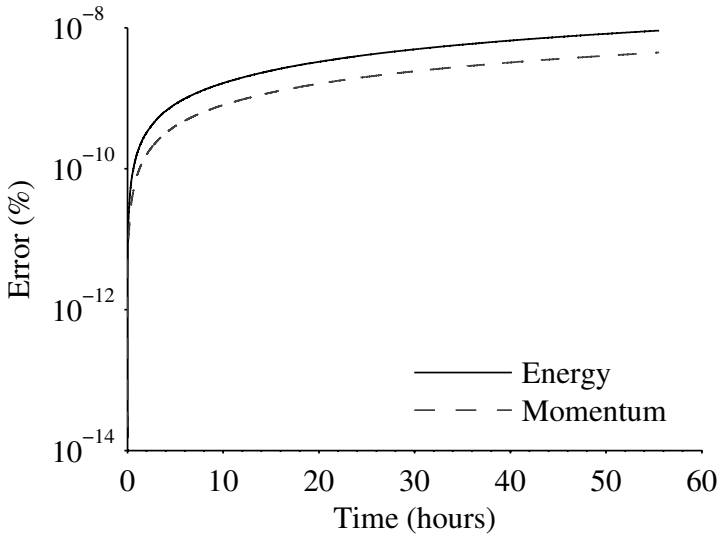


Figure 6.3: Long-term error for a free rigid body

While it is possible to achieve better momentum and energy conservation with traditional integrators by using smaller time steps and higher-order methods, doing so can become prohibitively computationally expensive for long integration times. The energy and momentum behavior of the variational integrator produces qualitatively realistic simulation results in cases where traditional integrators can produce unphysical behavior.

6.9.2 Damped Rigid Body

The second test incorporates the spherical damper of section 6.8 into the rigid body simulation. The damper inertia is set to $I_D = .2$ and the damping constant C is varied from 0.1 to 100. The midpoint rule is not shown because it quickly diverges

as C is increased and requires extremely small step sizes to avoid divergence.

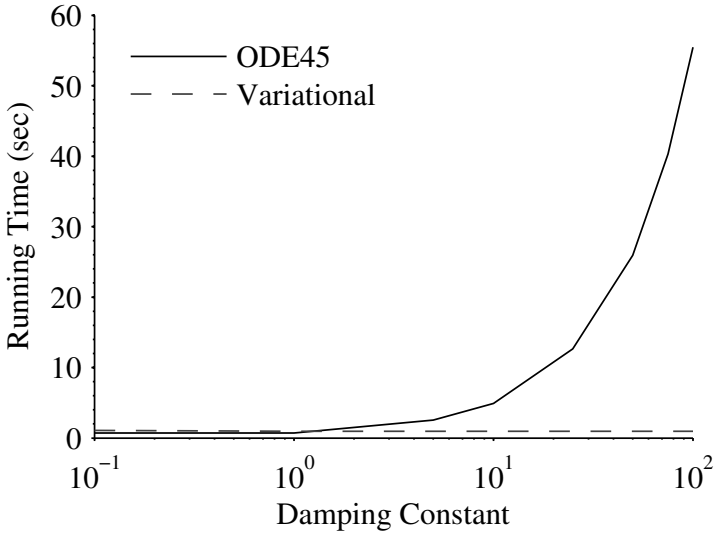


Figure 6.4: Integrator running time

Figure 6.4 shows the running time of ODE45 with default error tolerances and the variational integrator with a step size of $h = .3$ seconds, which is chosen so that the running times are roughly equal for small values of C. As the damping constant increases, the magnitude of the forces between the body and the damper increase and ODE45 must shorten its time steps to maintain accuracy and avoid diverging. The variational integrator, on the other hand, remains stable with a relatively large fixed step size.

Figure 6.5 shows the total energy of the system over the course of a simulation with $C = 100$. The variational integrator shows essentially the same energy damping behavior as ODE45 on this dissipative system while running over 50 times faster. Figure 6.6 shows the same simulation again, but with MATLAB's ODE15s solver, which is intended for solving stiff systems, substituted for ODE45. ODE15s runs in roughly the same time as the variational integrator on this problem but produces obviously incorrect and unrealistic energy behavior, with the total energy

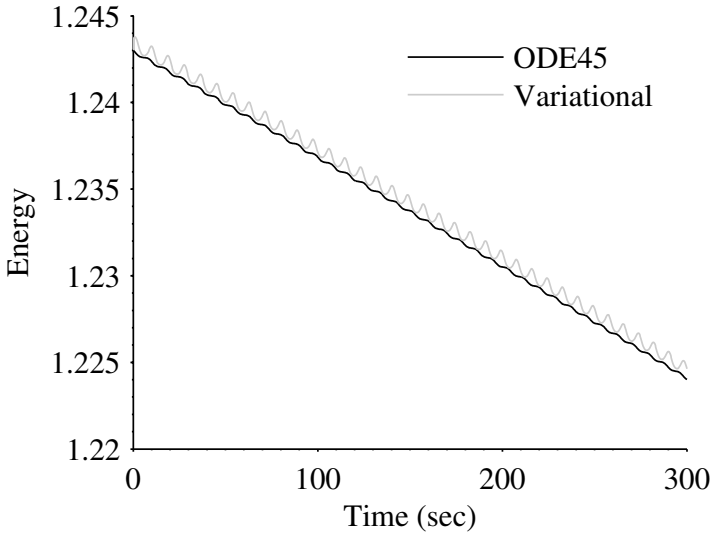


Figure 6.5: Energy for rigid body with damper

increasing over time.

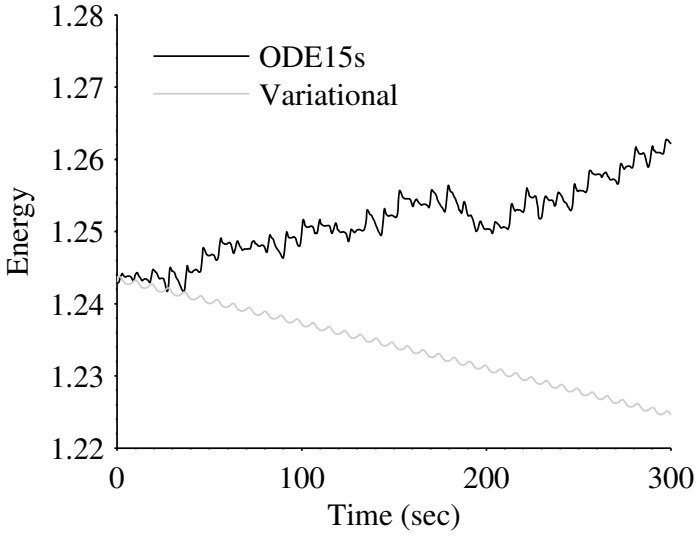


Figure 6.6: Energy for rigid body with damper

6.9.3 Extended Kalman Filter

The final test case demonstrates the advantages of variational integrators in a real-time estimation application. A spacecraft attitude determination problem is simulated where a multiplicative extended Kalman filter (MEKF)[52], [53] is used to estimate the attitude quaternion from noisy measurements of two inertial reference vectors. This situation is typical on CubeSats, for example, where magnetometer and sun vector measurements are commonly used for attitude determination.

A simulated truth model is constructed by integrating the rigid body equations of motion with initial conditions $q_0 = [0,\ 0,\ 0,\ 1]^\intercal$ and $\boldsymbol{\omega}_0 = [4,\ -5,\ 6]^\intercal$ degrees per second using ODE45 in MATLAB. Simulated vector measurements are then generated and Gaussian noise is added. Attitudes for filter initialization are computed from the first pair of noisy measurement vectors using the TRIAD algorithm[91].

Figure 6.7 compares a standard MEKF to one using a variational integrator to perform its state prediction step. The two filters have nearly identical perfor-

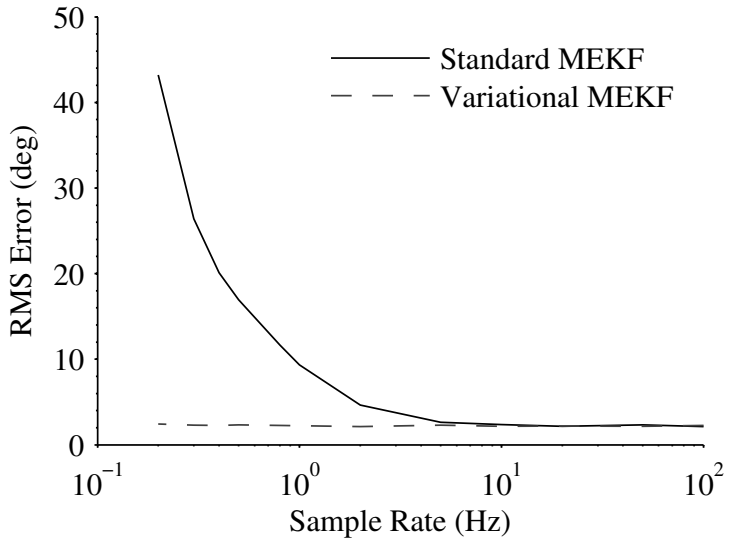


Figure 6.7: Multiplicative extended Kalman filter RMS attitude error

mance at high sample rates but show very different behavior as the sample rate decreases. The underlying reason for this performance difference is the quality of the linearizations that the variational integrator yields. Equation (6.38) and the corresponding Jacobian of equation (6.36) lead to the true linearization of the map from $\boldsymbol{\phi}_k$ to $\boldsymbol{\phi}_{k+1}$:

$$\frac{\partial \boldsymbol{\phi}_{k+1}}{\partial \boldsymbol{\phi}_k} = \left(\frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{\phi}_{k+1}} \right)^{-1} \frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{\phi}_k} \tag{6.69}$$

This linearization is completely independent of the step size taken. As a result, filters built around variational integrators are highly insensitive to sample rate and can maintain good performance and convergence at much lower rates than standard extended Kalman filters.

6.10 Conclusions

The integrators developed in this study offer physically realistic momentum and energy behavior while having modest computational costs. They consistently outperform Runge-Kutta schemes in a variety of tests on both conservative and non-conservative systems. High-quality linearizations can also be computed as part of the integration process, making the algorithms well suited for use in real-time estimation and control applications like attitude filtering. Lastly, the methods introduced are general and can be used to develop variational integrators for a wide range of applications in spacecraft dynamics.

APPENDIX A

KICKSAT CROWD FUNDING

Much of the funding that has made KickSat possible was raised through the crowd-funding website Kickstarter. Figure A.1 shows a screenshot of the KickSat web page on Kickstarter as it appeared in early 2012, shortly after fundraising was completed.

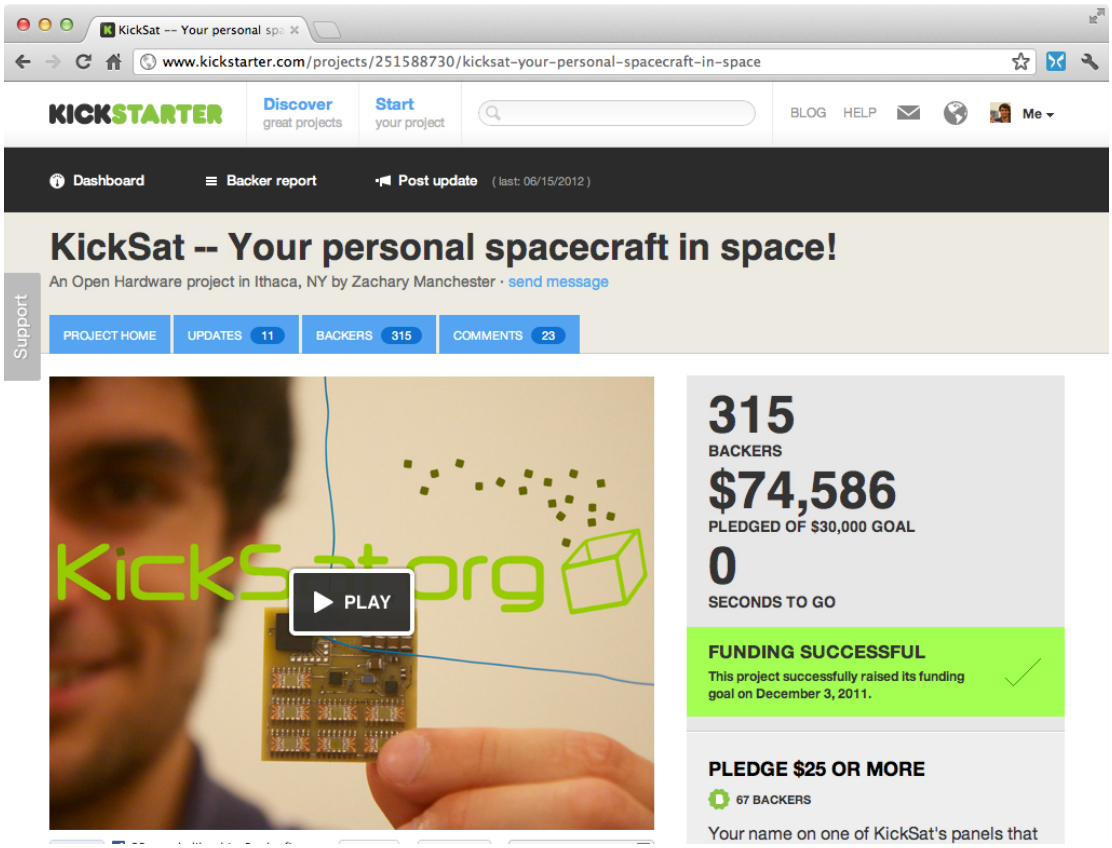


Figure A.1: KickSat Kickstarter web page

Kickstarter allows individuals to raise money from "backers" for well-defined projects[92], [93]. A project proposal is first submitted for review by Kickstarter and, if accepted, is hosted on Kickstarter's website for up to 60 days. During that time, individuals can donate to the project in various amounts set by the project's

creator, with the option of receiving "rewards." In the case of KickSat, 315 individuals pledged in amounts ranging from $25 to $10,000 in exchange for rewards including prototype Sprite hardware, passes to attend the launch of KickSat at Kennedy Space Center, and having their names engraved on KickSat's structure. In total, nearly $75,000 was raised between October and December 2011, as shown in figure A.2.

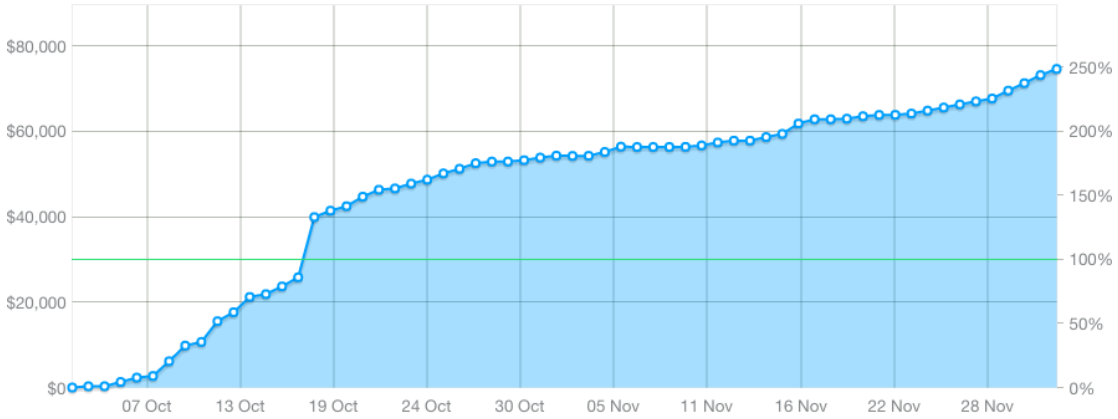
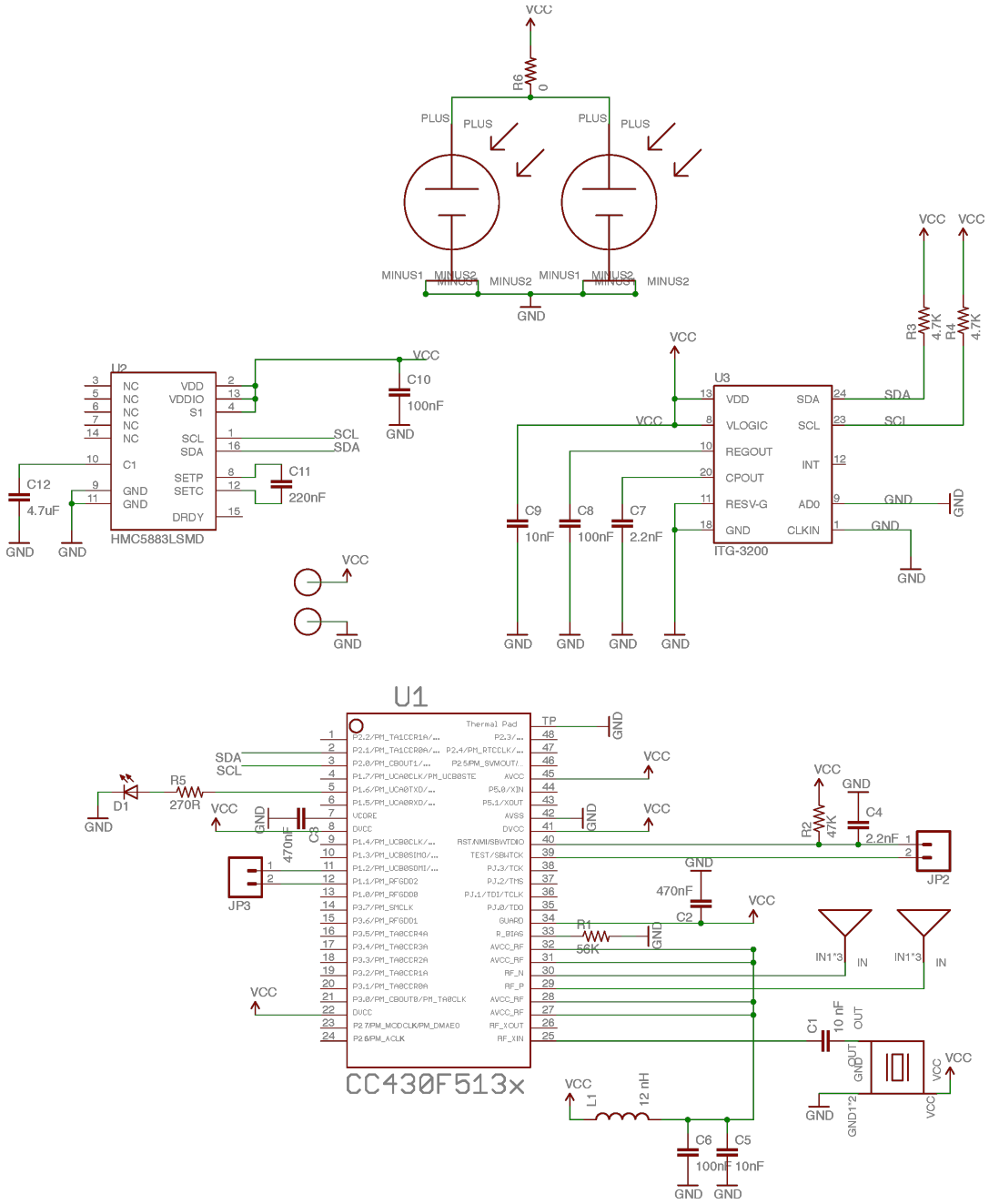


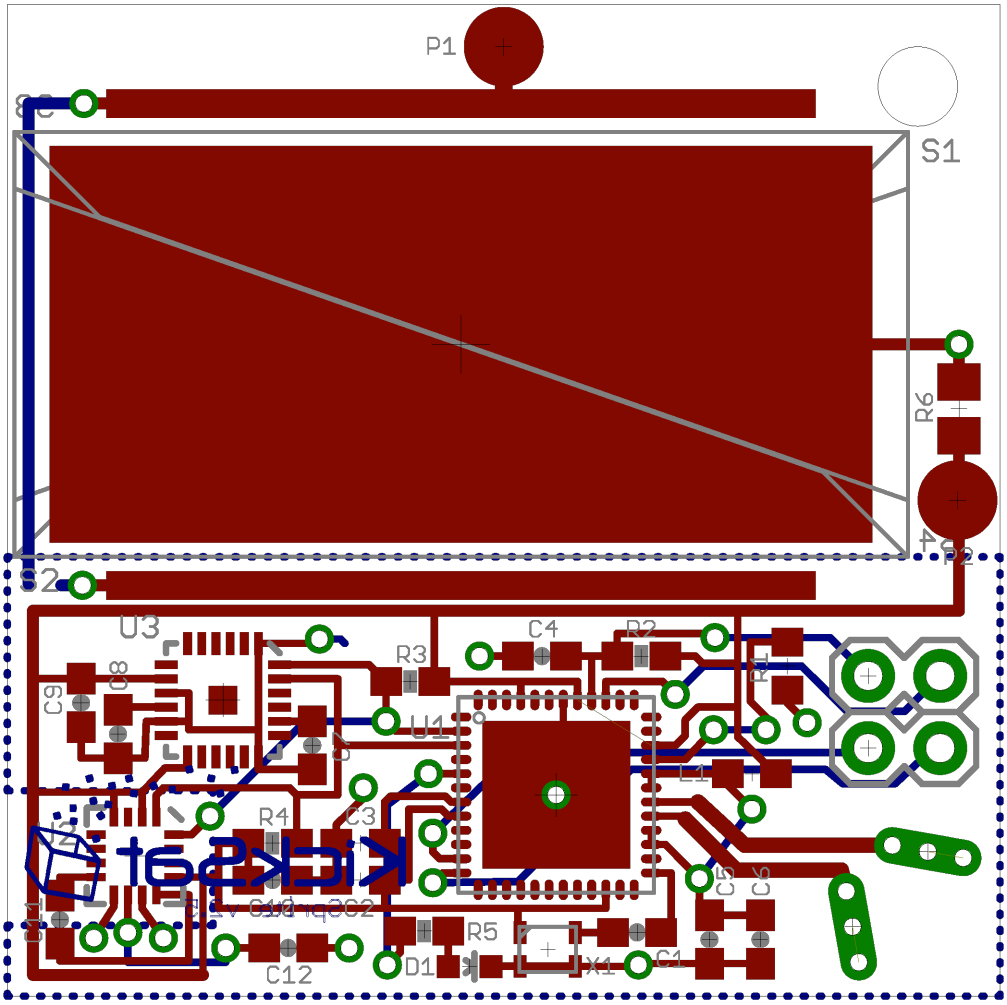
Figure A.2: Kickstarter fundraising progress

In a particularly interesting foreshadowing of the possibilities for low-cost amateur space missions, 26 individuals pledged $1,000 in exchange for a development kit to write flight code for their own Sprite. Of those, nine actually delivered finished software in time for the launch of KickSat-1. One experiment designed by members of the British Interplanetary Society attempted to use the RAM in the microcontroller on the Sprite as a radiation detector by writing pseudo-random data and reading it back repeatedly, checking for bit flips caused by high-energy particles. Most others relied on measurements from the onboard gyroscope, magnetometer, and temperature sensors.

APPENDIX B

SPRITE HARDWARE SCHEMATICS




P1
S1
R6
U3
C4
R2
R3
C9
U1
C3
R4
R5
C12
D1
X1
C1

APPENDIX C

SPRITE BILL OF MATERIALS

Part	Value	Package	Description	Part Number
C1, C5, C9	10 nF	0603	Capacitor	399-1091-1-ND
C2, C3	470nF	0603	Capacitor	399-3114-1-ND
C4, C7	2.2nF	0603	Capacitor	399-1085-1-ND
C6, C8, C10	100nF	0603	Capacitor	399-5089-1-ND
C11	220nF	0603	Capacitor	399-1102-1-ND
C12	4.7uF	0603	Capacitor	399-3482-1-ND
L1	12 nH	0603	Inductor	587-1546-1-ND
D1	Green	0603	LED	475-2709-1-ND
R1	56K	0603	Resistor	P56.0KHCT-ND
R2	47K	0603	Resistor	P47.0KHCT-ND
R3, R4	4.7K	0603	Resistor	P4.70KHCT-ND
R5	270R	0603	Resistor	P270GCT-ND
R6, R7, R8, R9, R10	0R	0805	Resistor	P0.0ACT-ND
U1	CC430F5137	RGZ48	MCU/Radio	296-27420-1-ND
U2	HMC5883LSMD	16LPCC	Magnetometer	342-1082-1-ND
U3	ITG-3200	QFN-24	Gyro	37T8091
X1	TCXO_7Z	TXC_7Z	Crystal Oscillator	887-1622-1-ND
P1, P2	SPRITE_PIN	SPRITE_PIN	Test Pin	ED1088CT-ND
S1, S2, S3, S4	TASC	TASC	Solar Cell	N/A

APPENDIX D

SPRITE RECEIVER SOURCE CODE

```
/* correlator_cf_impl.cc */
/*
 * Copyright 2014 Zac Manchester.
 *
 * This is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 3, or (at your option)
 * any later version.
 *
 * This software is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this software; see the file COPYING.  If not, write to
 * the Free Software Foundation, Inc., 51 Franklin Street,
 * Boston, MA 02110-1301, USA.
 */

#ifdef HAVE_CONFIG_H
#include "config.h"
#endif

#include <gnuradio/io_signature.h>
#include "correlator_cf_impl.h"
#include <gnuradio/gr_complex.h>
#include <gnuradio/fft/fft.h>
#include <complex>
#include <cmath>

using namespace std;

namespace gr {
  namespace sprite {

    correlator_cf::sptr
    correlator_cf::make(int prn_id0, int prn_id1)
    {
      return gnuradio::get_initial_sptr
        (new correlator_cf_impl(prn_id0, prn_id1));
    }

    /*
     * The private constructor
     */
    correlator_cf_impl::correlator_cf_impl(int prn_id0, int prn_id1)
      : gr::sync_block("correlator_cf",
              gr::io_signature::make(1, 1, sizeof(gr_complex)),
              gr::io_signature::make(1, 1, sizeof(float)))
    {
```

```
52      set_history(SPRITE_PRN_LENGTH);
53
54      generate_prns(prn_id0, prn_id1);
55
56      cc430_modulator(m_prn0, m_template0);
57      cc430_modulator(m_prn1, m_template1);
58      for (int k = 0; k < SPRITE_PRN_LENGTH; k++)
59      {
60        m_template0[k] = conj(m_template0[k]);
61        m_template1[k] = conj(m_template1[k]);
62      }
63
64      m_fft0 = new fft::fft_complex(SPRITE_PRN_LENGTH, true, 1);
65      m_fft_buffer_in0 = m_fft0->get_inbuf();
66      m_fft_buffer_out0 = m_fft0->get_outbuf();
67
68      m_fft1 = new fft::fft_complex(SPRITE_PRN_LENGTH, true, 1);
69      m_fft_buffer_in1 = m_fft1->get_inbuf();
70      m_fft_buffer_out1 = m_fft1->get_outbuf();
71    }
72
73    /*
74     * Our virtual destructor.
75     */
76    correlator_cf_impl::~correlator_cf_impl()
77    {
78    }
79
80
81    void correlator_cf_impl::cc430_modulator(int* prnBits, gr_complex* baseBand)
82    {
83      float* diffs = m_buffer_real1;
84      float* iBB = m_buffer_real2;
85      float* qBB = m_buffer_real3;
86
87      //Differentially encode with +/-1 values
88      diffs[0] = -2*prnBits[0] + 1;
89      for (int k = 1; k < SPRITE_PRN_LENGTH; k++)
90      {
91        char diff = prnBits[k]-prnBits[k-1];
92        if(diff == 0)
93        {
94          diffs[k] = 1;
95        }
96        else
97        {
98          diffs[k] = -1;
99        }
100     }
101
102     //Initialize with offset between I and Q
103     iBB[0] = 1;
104     qBB[0] = diffs[0];
105     qBB[1] = diffs[0];
106
```

```
107       for(int k = 1; k < SPRITE_PRN_LENGTH-2; k+=2)
108       {
109         iBB[k] = diffs[k]*iBB[k-1];
110         iBB[k+1] = iBB[k];
111       }
112       iBB[SPRITE_PRN_LENGTH-1] =
      diffs[SPRITE_PRN_LENGTH-1]*iBB[SPRITE_PRN_LENGTH-2];
113
114       for(int k = 2; k < SPRITE_PRN_LENGTH; k+=2)
115       {
116         qBB[k] = diffs[k]*qBB[k-1];
117         qBB[k+1] = qBB[k];
118       }
119
120       for(int k = 0; k < SPRITE_PRN_LENGTH; k++)
121       {
122         baseBand[k] = iBB[k]*cos(M_PI/2*k) + 1i*qBB[k]*sin(M_PI/2*k);
123       }
124     }
125
126     void correlator_cf_impl::generate_prns(int prn_id0, int prn_id1)
127     {
128       if(prn_id0 == -2)
129       {
130         //Deep copy M-sequence
131         for (int k = 0; k < M_SEQUENCE_LENGTH; k++)
132         {
133           m_prn0[k] = mseq1[k];
134         }
135       }
136       else if(prn_id0 == -1)
137       {
138         //Deep copy M-sequence
139         for (int k = 0; k < M_SEQUENCE_LENGTH; k++)
140         {
141           m_prn0[k] = mseq2[k];
142         }
143       }
144       else //if(prn_id >= 0 && prn_id < M_SEQUENCE_LENGTH)
145       {
146         //Generate Gold Codes by xor'ing 2 M-sequences in different phases
147         for (int k = 0; k < M_SEQUENCE_LENGTH-prn_id0; k++)
148         {
149           m_prn0[k] = mseq1[k] ^ mseq2[k+prn_id0];
150         }
151         for (int k = M_SEQUENCE_LENGTH-prn_id0; k < M_SEQUENCE_LENGTH; k++)
152         {
153           m_prn0[k] = mseq1[k] ^ mseq2[k-M_SEQUENCE_LENGTH+prn_id0];
154         }
155       }
156
157       m_prn0[SPRITE_PRN_LENGTH-1] = 0; //To pad out the last byte, add a zero to
      the end
158
159       if(prn_id1 == -2)
```

```
160       {
161         //Deep copy M-sequence
162         for (int k = 0; k < M_SEQUENCE_LENGTH; k++)
163         {
164           m_prn1[k] = mseq1[k];
165         }
166       }
167       else if(prn_id1 == -1)
168       {
169         //Deep copy M-sequence
170         for (int k = 0; k < M_SEQUENCE_LENGTH; k++)
171         {
172           m_prn1[k] = mseq2[k];
173         }
174       }
175       else //if(prn_id >= 0 && prn_id < M_SEQUENCE_LENGTH)
176       {
177         //Generate Gold Codes by xor'ing 2 M-sequences in different phases
178         for (int k = 0; k < M_SEQUENCE_LENGTH-prn_id1; k++)
179         {
180           m_prn1[k] = mseq1[k] ^ mseq2[k+prn_id1];
181         }
182         for (int k = M_SEQUENCE_LENGTH-prn_id1; k < M_SEQUENCE_LENGTH; k++)
183         {
184           m_prn1[k] = mseq1[k] ^ mseq2[k-M_SEQUENCE_LENGTH+prn_id1];
185         }
186       }
187
188       m_prn1[SPRITE_PRN_LENGTH-1] = 0; //To pad out the last byte, add a zero to
      the end
189
190     }
191
192     int
193     correlator_cf_impl::work(int noutput_items,
194         gr_vector_const_void_star &input_items,
195         gr_vector_void_star &output_items)
196     {
197         const gr_complex *in = (const gr_complex *) input_items[0];
198         float *out = (float *) output_items[0];
199
200         // Do <+signal processing+>
201         for(int k = 0; k < noutput_items; ++k) {
202
203           //Pointwise multiply by baseband template and copy to fft input
204           for (int j = 0; j < SPRITE_PRN_LENGTH; ++j)
205           {
206             m_fft_buffer_in0[j] = m_template0[j]*in[j+k];
207             m_fft_buffer_in1[j] = m_template1[j]*in[j+k];
208           }
209
210           //Take FFT
211           m_fft0->execute();
212           m_fft1->execute();
213
```

```
214           //Find largest value in FFT
215           float mag0 = real(m_fft_buffer_out0[0]*conj(m_fft_buffer_out0[0]));
216           float max0 = mag0;
217           float index0 = 0;
218           for (int j = 1; j < SPRITE_PRN_LENGTH; ++j)
219           {
220             mag0 = real(m_fft_buffer_out0[j]*conj(m_fft_buffer_out0[j]));
221             if (mag0 > max0)
222             {
223               max0 = mag0;
224               index0 = j;
225             }
226           }
227           float mag1 = real(m_fft_buffer_out1[0]*conj(m_fft_buffer_out1[0]));
228           float max1 = mag1;
229           float index1 = 0;
230           for (int j = 1; j < SPRITE_PRN_LENGTH; ++j)
231           {
232             mag1 = real(m_fft_buffer_out1[j]*conj(m_fft_buffer_out1[j]));
233             if (mag1 > max1)
234             {
235               max1 = mag1;
236               index1 = j;
237             }
238           }
239
240           out[k] = max1 >= max0 ? sqrt(max1) : -sqrt(max0);
241         }
242
243         // Tell runtime system how many output items we produced.
244         return noutput_items;
245       }
246
247   } /* namespace sprite */
248 } /* namespace gr */
```

```
/* peak_decimator_ff_impl.cc */
/*
 * Copyright 2014 Zac Manchester.
 *
 * This is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 3, or (at your option)
 * any later version.
 *
 * This software is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this software; see the file COPYING.  If not, write to
 * the Free Software Foundation, Inc., 51 Franklin Street,
 * Boston, MA 02110-1301, USA.
 */

#ifdef HAVE_CONFIG_H
#include "config.h"
#endif

#include <gnuradio/io_signature.h>
#include "peak_decimator_ff_impl.h"

namespace gr {
  namespace sprite {

    peak_decimator_ff::sptr
    peak_decimator_ff::make(int window_size)
    {
      return gnuradio::get_initial_sptr
        (new peak_decimator_ff_impl(window_size));
    }

    /*
     * The private constructor
     */
    peak_decimator_ff_impl::peak_decimator_ff_impl(int window_size)
      : gr::sync_decimator("peak_decimator_ff",
              gr::io_signature::make(1, 1, sizeof(float)),
              gr::io_signature::make(1, 1, sizeof(float)), window_size)
    {
  m_window = window_size;
    }

    /*
     * Our virtual destructor.
     */
    peak_decimator_ff_impl::~peak_decimator_ff_impl()
    {
    }

```

```
56      int
57      peak_decimator_ff_impl::work(int noutput_items,
58          gr_vector_const_void_star &input_items,
59          gr_vector_void_star &output_items)
60      {
61          const float *in = (const float *) input_items[0];
62          float *out = (float *) output_items[0];
63
64          for(int k = 0; k < noutput_items; ++k)
65          {
66            m_min = 0;
67            m_max = 0;
68
69            for(int j = m_window*k; j < m_window*(k+1); ++j)
70            {
71              if(in[j] > m_max)
72              {
73                m_max = in[j];
74              }
75              else if(in[j] < m_min)
76              {
77                m_min = in[j];
78              }
79            }
80
81            out[k] = m_max > -m_min ? m_max : m_min;
82          }
83
84          // Tell runtime system how many output items we produced.
85          return noutput_items;
86      }
87
88    } /* namespace sprite */
89  } /* namespace gr */
```

```
#!/usr/bin/env python
# soft_decoder.py
#
# Copyright 2015 Zac Manchester.
#
# This is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 3, or (at your option)
# any later version.
#
# This software is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this software; see the file COPYING.  If not, write to
# the Free Software Foundation, Inc., 51 Franklin Street,
# Boston, MA 02110-1301, USA.
#

from __future__ import print_function
from math import *
from numpy import *
from gnuradio import gr

class soft_decoder_c(gr.sync_block):
    """
    docstring for block soft_decoder_c
    """
    def __init__(self, threshold):
        gr.sync_block.__init__(self, name="soft_decoder_c", in_sig=[complex64],
    out_sig=[])

        self.set_history(30)
        self._detection_threshold = threshold

        self._preamble = array([1, 1, 1, -1, -1, 1, -1], dtype=float32)
        self._postamble = array([1, -1, 1, 1, -1, -1, -1], dtype=float32)
        self._template = hstack([self._preamble, zeros(16, dtype=float32),
    self._postamble])/sqrt(14)

        self._C = array([
            [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1],
            [1, 1, -1, 1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, 1],
            [-1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1],
            [1, -1, 1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
            [1, 1, 1, 1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1],
            [-1, -1, 1, -1, -1, 1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1],
            [1, -1, -1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, 1, 1, -1],
            [-1, 1, -1, 1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, 1],
            [-1, -1, 1, 1, -1, -1, 1, 1, -1, -1, -1, -1, 1, -1, -1, -1],
            [1, 1, 1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1, 1],
            [-1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, 1, -1, 1, -1],
            [1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1, 1, 1],
```

```
[1,  1, −1, −1, −1, −1, −1,  1, −1, −1, −1, −1,  1,  1, −1, −1],
[−1, −1, −1,  1, −1,  1,  1, −1, −1, −1, −1, −1,  1,  1, −1,  1],
[1, −1,  1,  1,  1, −1, −1,  1, −1, −1, −1, −1,  1,  1,  1, −1],
[−1,  1,  1, −1,  1,  1,  1, −1, −1, −1, −1, −1,  1,  1,  1,  1],
[−1,  1,  1, −1, −1,  1,  1,  1, −1, −1, −1,  1, −1, −1, −1, −1],
[1, −1,  1,  1, −1, −1, −1, −1, −1, −1, −1, −1,  1, −1, −1, −1,  1],
[−1, −1, −1,  1,  1,  1,  1,  1, −1, −1, −1,  1, −1, −1,  1, −1],
[1,  1, −1, −1,  1, −1, −1, −1, −1, −1, −1, −1,  1, −1, −1,  1,  1],
[1, −1, −1,  1, −1,  1, −1,  1, −1, −1, −1,  1, −1,  1, −1, −1],
[−1,  1, −1, −1, −1, −1,  1, −1, −1, −1, −1,  1, −1,  1, −1,  1],
[1,  1,  1, −1,  1,  1, −1,  1, −1, −1, −1,  1, −1,  1,  1, −1],
[−1, −1,  1,  1,  1, −1,  1, −1, −1, −1, −1,  1, −1,  1,  1,  1],
[−1,  1, −1,  1, −1,  1, −1, −1, −1, −1, −1,  1,  1, −1, −1, −1],
[1, −1, −1, −1, −1, −1,  1,  1, −1, −1, −1,  1,  1, −1, −1,  1],
[−1, −1,  1, −1,  1,  1, −1, −1, −1, −1, −1,  1,  1, −1,  1, −1],
[1,  1,  1,  1,  1, −1,  1,  1, −1, −1, −1,  1,  1, −1,  1,  1],
[1, −1,  1, −1, −1,  1,  1, −1, −1, −1, −1,  1,  1,  1, −1, −1],
[−1,  1,  1,  1, −1, −1, −1,  1, −1, −1, −1,  1,  1,  1, −1,  1],
[1,  1, −1,  1,  1,  1,  1, −1, −1, −1, −1,  1,  1,  1,  1, −1],
[−1, −1, −1, −1,  1, −1, −1,  1, −1, −1, −1,  1,  1,  1,  1,  1],
[1,  1, −1, −1,  1,  1, −1,  1, −1, −1,  1, −1, −1, −1, −1, −1],
[−1, −1, −1,  1,  1, −1,  1, −1, −1, −1,  1, −1, −1, −1, −1,  1],
[1, −1,  1,  1, −1,  1, −1,  1, −1, −1,  1, −1, −1, −1,  1, −1],
[−1,  1,  1, −1, −1, −1,  1, −1, −1, −1,  1, −1, −1, −1,  1,  1],
[−1, −1,  1,  1,  1,  1,  1,  1, −1, −1,  1, −1, −1,  1, −1, −1],
[1,  1,  1, −1,  1, −1, −1, −1, −1, −1,  1, −1, −1,  1, −1,  1],
[−1,  1, −1, −1, −1,  1,  1,  1, −1, −1,  1, −1, −1,  1,  1, −1],
[1, −1, −1,  1, −1, −1, −1, −1, −1, −1,  1, −1, −1,  1,  1,  1],
[1,  1,  1,  1,  1,  1,  1, −1, −1, −1,  1, −1,  1, −1, −1, −1],
[−1, −1,  1, −1,  1, −1, −1,  1, −1, −1,  1, −1,  1, −1, −1,  1],
[1, −1, −1, −1, −1,  1,  1, −1, −1, −1,  1, −1,  1, −1,  1, −1],
[−1,  1, −1,  1, −1, −1, −1,  1, −1, −1,  1, −1,  1, −1,  1,  1],
[−1, −1, −1, −1,  1,  1, −1, −1, −1, −1,  1, −1,  1,  1, −1, −1],
[1,  1, −1,  1,  1, −1,  1,  1, −1, −1,  1, −1,  1,  1, −1,  1],
[−1,  1,  1,  1, −1,  1, −1, −1, −1, −1,  1, −1,  1,  1,  1, −1],
[1, −1,  1, −1, −1, −1,  1,  1, −1, −1,  1, −1,  1,  1,  1,  1],
[1, −1,  1, −1,  1, −1,  1, −1, −1, −1,  1,  1, −1, −1, −1, −1],
[−1,  1,  1,  1,  1,  1, −1,  1, −1, −1,  1,  1, −1, −1, −1,  1],
[1,  1, −1,  1, −1, −1,  1, −1, −1, −1,  1,  1, −1, −1,  1, −1],
[−1, −1, −1, −1, −1,  1, −1,  1, −1, −1,  1,  1, −1, −1,  1,  1],
[−1,  1, −1,  1,  1, −1, −1, −1, −1, −1,  1,  1, −1,  1, −1, −1],
[1, −1, −1, −1,  1,  1,  1,  1, −1, −1,  1,  1, −1,  1, −1,  1],
[−1, −1,  1, −1, −1, −1, −1, −1, −1, −1,  1,  1, −1,  1,  1, −1],
[1,  1,  1,  1, −1,  1,  1,  1, −1, −1,  1,  1, −1,  1,  1,  1],
[1, −1, −1,  1,  1, −1, −1,  1, −1, −1,  1,  1,  1, −1, −1, −1],
[−1,  1, −1, −1,  1,  1,  1, −1, −1, −1,  1,  1,  1, −1, −1,  1],
[1, −1,  1,  1,  1,  1, −1, −1, −1, −1,  1,  1,  1,  1, −1,  1],
[−1, −1, −1,  1, −1, −1,  1,  1, −1, −1,  1,  1,  1,  1,  1, −1],
[1,  1, −1, −1, −1,  1, −1, −1, −1, −1,  1,  1,  1,  1,  1,  1],
[−1,  1, −1, −1,  1,  1,  1, −1, −1,  1, −1, −1, −1, −1, −1, −1],
[1, −1, −1,  1,  1, −1, −1,  1, −1,  1, −1, −1, −1, −1, −1,  1],
[−1, −1,  1,  1, −1,  1,  1, −1, −1,  1, −1, −1, −1, −1,  1, −1],
```

```
109  [1, 1, 1, −1, −1, −1, −1, 1, −1, 1, −1, −1, −1, −1, 1, 1],
110  [1, −1, 1, 1, 1, 1, −1, −1, −1, 1, −1, −1, −1, 1, −1, −1],
111  [−1, 1, 1, −1, 1, −1, 1, 1, −1, 1, −1, −1, −1, 1, −1, 1],
112  [1, 1, −1, −1, −1, 1, −1, −1, −1, 1, −1, −1, −1, 1, 1, −1],
113  [−1, −1, −1, 1, −1, −1, 1, 1, −1, 1, −1, −1, −1, 1, 1, 1],
114  [−1, 1, 1, 1, 1, 1, −1, 1, −1, 1, −1, −1, 1, −1, −1, −1],
115  [1, −1, 1, −1, 1, −1, 1, −1, −1, 1, −1, −1, 1, −1, −1, 1],
116  [−1, −1, −1, −1, −1, 1, −1, 1, −1, 1, −1, −1, 1, −1, 1, −1],
117  [1, 1, −1, 1, −1, −1, 1, −1, −1, 1, −1, −1, 1, −1, 1, 1],
118  [1, −1, −1, −1, 1, 1, 1, 1, −1, 1, −1, −1, 1, 1, −1, −1],
119  [−1, 1, −1, 1, 1, −1, −1, −1, −1, 1, −1, −1, 1, 1, −1, 1],
120  [1, 1, 1, 1, −1, 1, 1, 1, −1, 1, −1, −1, 1, 1, 1, −1],
121  [−1, −1, 1, −1, −1, −1, −1, −1, −1, 1, −1, −1, 1, 1, 1, 1],
122  [−1, −1, 1, −1, 1, −1, −1, 1, −1, 1, −1, 1, −1, −1, −1, −1],
123  [1, 1, 1, 1, 1, 1, 1, −1, −1, 1, −1, 1, −1, −1, −1, 1],
124  [−1, 1, −1, 1, −1, −1, −1, 1, −1, 1, −1, 1, −1, −1, 1, −1],
125  [1, −1, −1, −1, −1, 1, 1, −1, −1, 1, −1, 1, −1, −1, 1, 1],
126  [1, 1, −1, 1, 1, −1, 1, 1, −1, 1, −1, 1, −1, 1, −1, −1],
127  [−1, −1, −1, −1, 1, 1, −1, −1, −1, 1, −1, 1, −1, 1, −1, 1],
128  [1, −1, 1, −1, −1, −1, 1, 1, −1, 1, −1, 1, −1, 1, 1, −1],
129  [−1, 1, 1, 1, −1, 1, −1, −1, −1, 1, −1, 1, −1, 1, 1, 1],
130  [−1, −1, −1, 1, 1, −1, 1, −1, −1, 1, −1, 1, 1, −1, −1, −1],
131  [1, 1, −1, −1, 1, 1, −1, 1, −1, 1, −1, 1, 1, −1, −1, 1],
132  [−1, 1, 1, −1, −1, −1, 1, −1, −1, 1, −1, 1, 1, −1, 1, −1],
133  [1, −1, 1, 1, −1, 1, −1, 1, −1, 1, −1, 1, 1, −1, 1, 1],
134  [1, 1, 1, −1, 1, −1, −1, −1, −1, 1, −1, 1, 1, 1, −1, −1],
135  [−1, −1, 1, 1, 1, 1, 1, 1, −1, 1, −1, 1, 1, 1, −1, 1],
136  [1, −1, −1, 1, −1, −1, −1, −1, −1, 1, −1, 1, 1, 1, 1, −1],
137  [−1, 1, −1, −1, −1, 1, 1, 1, −1, 1, −1, 1, 1, 1, 1, 1],
138  [1, −1, −1, −1, −1, −1, 1, 1, −1, 1, 1, −1, −1, −1, −1, −1],
139  [−1, 1, −1, 1, −1, 1, −1, −1, −1, 1, 1, −1, −1, −1, −1, 1],
140  [1, 1, 1, 1, 1, −1, 1, 1, −1, 1, 1, −1, −1, −1, 1, −1],
141  [−1, −1, 1, −1, 1, 1, −1, −1, −1, 1, 1, −1, −1, −1, 1, 1],
142  [−1, 1, 1, 1, −1, −1, −1, 1, −1, 1, 1, −1, −1, 1, −1, −1],
143  [1, −1, 1, −1, −1, 1, 1, −1, −1, 1, 1, −1, −1, 1, −1, 1],
144  [−1, −1, −1, −1, 1, −1, −1, 1, −1, 1, 1, −1, −1, 1, 1, −1],
145  [1, 1, −1, 1, 1, 1, 1, −1, −1, 1, 1, −1, −1, 1, 1, 1],
146  [1, −1, 1, 1, −1, −1, −1, −1, −1, 1, 1, −1, 1, −1, −1, −1],
147  [−1, 1, 1, −1, −1, 1, 1, 1, −1, 1, 1, −1, 1, −1, −1, 1],
148  [1, 1, −1, −1, 1, −1, −1, −1, −1, 1, 1, −1, 1, −1, 1, −1],
149  [−1, −1, −1, 1, 1, 1, 1, 1, −1, 1, 1, −1, 1, −1, 1, 1],
150  [−1, 1, −1, −1, −1, −1, 1, −1, −1, 1, 1, −1, 1, 1, −1, −1],
151  [1, −1, −1, 1, −1, 1, −1, 1, −1, 1, 1, −1, 1, 1, −1, 1],
152  [−1, −1, 1, 1, 1, −1, 1, −1, −1, 1, 1, −1, 1, 1, 1, −1],
153  [1, 1, 1, −1, 1, 1, −1, 1, −1, 1, 1, −1, 1, 1, 1, 1],
154  [1, 1, 1, −1, −1, 1, −1, −1, −1, 1, 1, 1, −1, −1, −1, −1],
155  [−1, −1, 1, 1, −1, −1, 1, 1, −1, 1, 1, 1, −1, −1, −1, 1],
156  [1, −1, −1, 1, 1, 1, −1, −1, −1, 1, 1, 1, −1, −1, 1, −1],
157  [−1, 1, −1, −1, 1, −1, 1, 1, −1, 1, 1, 1, −1, −1, 1, 1],
158  [−1, −1, −1, 1, −1, 1, 1, −1, −1, 1, 1, 1, −1, 1, −1, −1],
159  [1, 1, −1, −1, −1, −1, −1, 1, −1, 1, 1, 1, −1, 1, −1, 1],
160  [−1, 1, 1, −1, 1, 1, 1, −1, −1, 1, 1, 1, −1, 1, 1, −1],
161  [1, −1, 1, 1, 1, −1, −1, 1, −1, 1, 1, 1, −1, 1, 1, 1],
162  [1, 1, −1, 1, −1, 1, 1, 1, −1, 1, 1, 1, 1, −1, −1, −1],
163  [−1, −1, −1, −1, −1, −1, −1, −1, −1, 1, 1, 1, 1, −1, −1, 1],
```

```
[1, −1, 1, −1, 1, 1, 1, 1, −1, 1, 1, 1, 1, −1, 1, −1],
[−1, 1, 1, 1, 1, −1, −1, −1, −1, 1, 1, 1, 1, −1, 1, 1],
[−1, −1, 1, −1, −1, 1, −1, 1, −1, 1, 1, 1, 1, 1, −1, −1],
[1, 1, 1, 1, −1, −1, 1, −1, −1, 1, 1, 1, 1, 1, −1, 1],
[−1, 1, −1, 1, 1, 1, −1, 1, −1, 1, 1, 1, 1, 1, 1, −1],
[1, −1, −1, −1, 1, −1, 1, −1, −1, 1, 1, 1, 1, 1, 1, 1],
[1, −1, −1, 1, 1, 1, 1, −1, 1, −1, −1, −1, −1, −1, −1, −1],
[−1, 1, −1, −1, 1, −1, −1, 1, 1, −1, −1, −1, −1, −1, −1, 1],
[1, 1, 1, −1, −1, 1, 1, −1, 1, −1, −1, −1, −1, −1, 1, −1],
[−1, −1, 1, 1, −1, −1, −1, 1, 1, −1, −1, −1, −1, −1, 1, 1],
[−1, 1, 1, −1, 1, 1, −1, −1, 1, −1, −1, −1, −1, 1, −1, −1],
[1, −1, 1, 1, 1, −1, 1, 1, 1, −1, −1, −1, −1, 1, −1, 1],
[−1, −1, −1, 1, −1, 1, −1, −1, 1, −1, −1, −1, −1, 1, 1, −1],
[1, 1, −1, −1, −1, −1, 1, 1, 1, −1, −1, −1, −1, 1, 1, 1],
[1, −1, 1, −1, 1, 1, −1, 1, 1, −1, −1, −1, 1, −1, −1, −1],
[−1, −1, −1, −1, −1, −1, 1, −1, 1, −1, −1, −1, 1, −1, 1, 1],
[−1, 1, −1, 1, 1, 1, 1, 1, 1, −1, −1, −1, 1, 1, −1, −1],
[1, −1, −1, −1, 1, −1, −1, −1, 1, −1, −1, −1, 1, 1, −1, 1],
[−1, −1, 1, −1, −1, 1, 1, 1, 1, −1, −1, −1, 1, 1, 1, −1],
[1, 1, 1, 1, −1, −1, −1, −1, 1, −1, −1, −1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, −1, −1, 1, 1, −1, −1, 1, −1, −1, −1, −1],
[−1, −1, 1, −1, 1, 1, 1, −1, 1, −1, −1, 1, −1, −1, −1, 1],
[1, −1, −1, −1, −1, −1, −1, 1, 1, −1, −1, 1, −1, −1, 1, −1],
[−1, 1, −1, 1, −1, 1, 1, −1, 1, −1, −1, 1, −1, −1, 1, 1],
[−1, −1, −1, −1, 1, −1, 1, 1, 1, −1, −1, 1, −1, 1, −1, −1],
[1, 1, −1, 1, 1, 1, −1, −1, 1, −1, −1, 1, −1, 1, −1, 1],
[−1, 1, 1, 1, −1, −1, 1, 1, 1, −1, −1, 1, −1, 1, 1, −1],
[1, −1, 1, −1, −1, 1, −1, −1, 1, −1, −1, 1, −1, 1, 1, 1],
[1, 1, −1, −1, 1, −1, 1, −1, 1, −1, −1, 1, 1, −1, −1, −1],
[−1, −1, −1, 1, 1, 1, −1, 1, 1, −1, −1, 1, 1, −1, 1, 1],
[1, −1, 1, 1, −1, −1, 1, −1, 1, −1, −1, 1, 1, −1, 1, −1],
[−1, 1, 1, −1, −1, 1, −1, 1, 1, −1, −1, 1, 1, −1, 1, 1],
[−1, −1, 1, 1, 1, −1, −1, −1, 1, −1, −1, 1, 1, 1, −1, −1],
[1, 1, 1, −1, 1, 1, 1, 1, 1, −1, −1, 1, 1, 1, −1, 1],
[−1, 1, −1, −1, −1, −1, −1, −1, 1, −1, −1, 1, 1, 1, 1, −1],
[1, −1, −1, 1, −1, 1, 1, 1, 1, −1, −1, 1, 1, 1, 1, 1],
[−1, 1, −1, 1, −1, −1, 1, 1, 1, −1, 1, −1, −1, −1, −1, −1],
[1, −1, −1, −1, −1, 1, −1, −1, 1, −1, 1, −1, −1, −1, −1, 1],
[−1, −1, 1, −1, 1, −1, 1, 1, 1, −1, 1, −1, −1, −1, 1, −1],
[1, 1, 1, 1, 1, 1, −1, −1, 1, −1, 1, −1, −1, −1, 1, 1],
[1, −1, 1, −1, −1, −1, −1, 1, 1, −1, 1, −1, −1, 1, −1, −1],
[−1, 1, 1, 1, −1, 1, 1, −1, 1, −1, 1, −1, −1, 1, −1, 1],
[1, 1, −1, 1, 1, −1, −1, 1, 1, −1, 1, −1, −1, 1, 1, −1],
[−1, −1, −1, −1, 1, 1, 1, −1, 1, −1, 1, −1, −1, −1, 1, 1],
[−1, 1, 1, −1, −1, −1, −1, −1, 1, −1, 1, −1, 1, −1, −1, −1],
[1, −1, −1, 1, −1, 1, 1, 1, 1, −1, −1, 1, 1, 1, 1, 1],
[−1, −1, 1, −1, 1, −1, 1, 1, 1, −1, 1, −1, −1, −1, 1, −1],
[1, 1, 1, 1, 1, 1, −1, −1, 1, −1, 1, −1, −1, −1, 1, 1],
[1, −1, 1, −1, −1, −1, −1, 1, 1, −1, 1, −1, −1, 1, −1, −1],
[−1, 1, 1, 1, −1, 1, 1, −1, 1, −1, 1, −1, −1, −1, 1, 1],
[1, 1, −1, 1, 1, −1, −1, 1, 1, −1, 1, −1, −1, 1, 1, −1],
[−1, −1, −1, −1, 1, 1, 1, −1, 1, −1, 1, −1, −1, 1, 1, 1],
[−1, 1, 1, −1, −1, −1, −1, −1, 1, −1, 1, −1, 1, −1, −1, −1],
[1, −1, 1, 1, −1, 1, 1, 1, 1, −1, 1, −1, −1, 1, −1, 1],
[−1, −1, −1, 1, 1, −1, −1, −1, 1, −1, 1, −1, 1, −1, 1, −1],
[1, 1, −1, −1, 1, 1, 1, 1, 1, −1, 1, −1, 1, −1, 1, 1],
[1, −1, −1, 1, −1, −1, 1, −1, 1, −1, 1, −1, 1, 1, −1, −1],
[−1, 1, −1, −1, −1, 1, −1, 1, 1, −1, 1, −1, 1, 1, −1, 1],
[1, 1, 1, −1, 1, −1, 1, −1, 1, −1, 1, −1, 1, 1, 1, −1],
[−1, −1, 1, 1, 1, 1, −1, 1, 1, −1, 1, −1, 1, 1, 1, 1],
[−1, −1, 1, 1, −1, 1, −1, −1, 1, −1, 1, 1, −1, −1, −1, −1],
```

```
[1, 1, 1, −1, −1, −1, 1, 1, 1, −1, 1, 1, −1, −1, −1, 1],
[−1, 1, −1, −1, 1, 1, −1, −1, 1, −1, 1, 1, −1, −1, 1, −1],
[1, −1, −1, 1, 1, −1, 1, 1, 1, −1, 1, 1, −1, −1, 1, 1],
[1, 1, −1, −1, −1, 1, 1, −1, 1, −1, 1, 1, −1, 1, −1, −1],
[−1, −1, −1, 1, −1, −1, −1, 1, 1, −1, 1, 1, −1, 1, −1, 1],
[1, −1, 1, 1, 1, 1, 1, −1, 1, −1, 1, 1, −1, 1, 1, −1],
[−1, 1, 1, −1, 1, −1, −1, 1, 1, −1, 1, 1, −1, 1, 1, 1],
[−1, −1, −1, −1, −1, 1, 1, 1, 1, −1, 1, 1, 1, −1, −1, −1],
[1, 1, −1, 1, −1, −1, −1, −1, 1, −1, 1, 1, 1, −1, −1, 1],
[−1, 1, 1, 1, 1, 1, 1, 1, 1, −1, 1, 1, 1, −1, 1, −1],
[1, −1, 1, −1, 1, −1, −1, −1, 1, −1, 1, 1, 1, −1, 1, 1],
[1, 1, 1, 1, −1, 1, −1, 1, 1, −1, 1, 1, 1, 1, −1, −1],
[−1, −1, 1, −1, −1, −1, 1, −1, 1, −1, 1, 1, 1, 1, −1, 1],
[1, −1, −1, −1, 1, 1, −1, 1, 1, −1, 1, 1, 1, 1, 1, −1],
[−1, 1, −1, 1, 1, −1, 1, −1, 1, −1, 1, 1, 1, 1, 1, 1],
[1, 1, −1, 1, −1, −1, −1, −1, 1, 1, −1, −1, −1, −1, −1, −1],
[−1, −1, −1, −1, −1, 1, 1, 1, 1, 1, −1, −1, −1, −1, −1, 1],
[1, −1, 1, −1, 1, −1, −1, −1, 1, 1, −1, −1, −1, −1, 1, −1],
[−1, 1, 1, 1, 1, 1, 1, 1, 1, 1, −1, −1, −1, −1, 1, 1],
[−1, −1, 1, −1, −1, −1, 1, −1, 1, 1, −1, −1, −1, 1, −1, −1],
[1, 1, 1, 1, −1, 1, −1, 1, 1, 1, −1, −1, −1, 1, −1, 1],
[−1, 1, −1, 1, 1, −1, 1, −1, 1, 1, −1, −1, −1, 1, 1, −1],
[1, −1, −1, −1, 1, 1, −1, 1, 1, 1, −1, −1, −1, 1, 1, 1],
[1, 1, 1, −1, −1, −1, 1, 1, 1, 1, −1, −1, 1, −1, −1, −1],
[−1, −1, 1, 1, −1, 1, −1, −1, 1, 1, −1, −1, 1, −1, −1, 1],
[1, −1, −1, 1, 1, −1, 1, 1, 1, 1, −1, −1, 1, −1, 1, −1],
[−1, 1, −1, −1, 1, 1, −1, −1, 1, 1, −1, −1, 1, −1, 1, 1],
[−1, −1, −1, 1, −1, −1, −1, 1, 1, 1, −1, −1, 1, 1, −1, −1],
[1, 1, −1, −1, −1, 1, 1, −1, 1, 1, −1, −1, 1, 1, −1, 1],
[−1, 1, 1, −1, 1, −1, −1, 1, 1, 1, −1, −1, 1, 1, 1, −1],
[1, −1, 1, 1, 1, 1, 1, −1, 1, 1, −1, −1, 1, 1, 1, 1],
[1, −1, 1, 1, −1, 1, 1, 1, 1, 1, −1, 1, −1, −1, −1, −1],
[−1, 1, 1, −1, −1, −1, −1, −1, 1, 1, −1, 1, −1, −1, −1, 1],
[1, 1, −1, −1, 1, 1, 1, 1, 1, 1, −1, 1, −1, −1, 1, −1],
[−1, −1, −1, 1, 1, −1, −1, −1, 1, 1, −1, 1, −1, −1, 1, 1],
[−1, 1, −1, −1, −1, 1, −1, 1, 1, 1, −1, 1, −1, 1, −1, −1],
[1, −1, −1, 1, −1, −1, 1, −1, 1, 1, −1, 1, −1, 1, −1, 1],
[−1, −1, 1, 1, 1, 1, −1, 1, 1, 1, −1, 1, −1, 1, 1, −1],
[1, 1, 1, −1, 1, −1, 1, −1, 1, 1, −1, 1, −1, 1, 1, 1],
[1, −1, −1, −1, −1, 1, −1, −1, 1, 1, −1, 1, 1, −1, −1, −1],
[−1, 1, −1, 1, −1, −1, 1, 1, 1, 1, −1, 1, 1, −1, −1, 1],
[1, 1, 1, 1, 1, 1, −1, −1, 1, 1, −1, 1, 1, −1, 1, −1],
[−1, −1, 1, −1, 1, −1, −1, 1, 1, 1, 1, −1, 1, 1, −1, 1],
[−1, 1, 1, 1, −1, 1, 1, −1, 1, 1, −1, 1, 1, 1, −1, −1],
[1, −1, 1, −1, −1, −1, −1, 1, 1, 1, −1, 1, 1, 1, −1, 1],
[−1, −1, −1, −1, 1, 1, 1, −1, 1, 1, −1, 1, 1, 1, 1, −1],
[1, 1, −1, 1, 1, −1, −1, 1, 1, 1, −1, 1, 1, 1, 1, 1],
[−1, −1, −1, 1, 1, 1, −1, 1, 1, 1, 1, −1, −1, −1, −1, −1],
[1, 1, −1, −1, 1, −1, 1, −1, 1, 1, 1, −1, −1, −1, −1, 1],
[−1, 1, 1, −1, −1, 1, −1, 1, 1, 1, 1, −1, −1, −1, 1, −1],
[1, −1, 1, 1, −1, −1, 1, −1, 1, 1, 1, −1, −1, −1, 1, 1],
[1, 1, 1, −1, 1, 1, 1, 1, 1, 1, 1, −1, −1, 1, −1, −1],
[−1, −1, 1, 1, 1, −1, −1, −1, 1, 1, 1, −1, −1, 1, −1, 1],
[1, −1, −1, 1, −1, 1, 1, 1, 1, 1, 1, −1, −1, 1, 1, −1],
[−1, 1, −1, −1, −1, −1, −1, −1, 1, 1, −1, −1, 1, 1, 1, 1],
```

```
274             [-1, -1, 1, -1, 1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1, -1],
275             [1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1, -1, 1, -1, -1, 1],
276             [-1, 1, -1, 1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, 1, -1],
277             [1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, 1, -1, 1, 1],
278             [1, 1, -1, 1, 1, 1, -1, -1, 1, 1, 1, -1, 1, 1, -1, -1],
279             [-1, -1, -1, -1, 1, -1, 1, 1, 1, 1, 1, -1, 1, 1, -1, 1],
280             [1, -1, 1, -1, -1, 1, -1, -1, 1, 1, 1, -1, 1, 1, 1, -1],
281             [-1, 1, 1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1],
282             [-1, 1, 1, 1, 1, -1, 1, -1, 1, 1, 1, 1, -1, -1, -1, -1],
283             [1, -1, 1, -1, 1, 1, -1, 1, 1, 1, 1, 1, -1, -1, -1, 1],
284             [-1, -1, -1, -1, -1, -1, 1, -1, 1, 1, 1, 1, -1, -1, 1, -1],
285             [1, 1, -1, 1, -1, 1, -1, 1, 1, 1, 1, 1, -1, -1, 1, 1],
286             [1, -1, -1, -1, 1, -1, -1, -1, 1, 1, 1, 1, -1, 1, -1, -1],
287             [-1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, -1, 1],
288             [1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, 1, 1, -1],
289             [-1, -1, 1, -1, -1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1],
290             [-1, 1, -1, -1, 1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1],
291             [1, -1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, -1, 1],
292             [-1, -1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, 1, -1],
293             [1, 1, 1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, -1, 1, 1],
294             [1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1],
295             [-1, 1, 1, -1, 1, 1, -1, -1, 1, 1, 1, 1, 1, 1, -1, 1],
296             [1, 1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1],
297             [-1, -1, -1, 1, -1, 1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1]
298             ], dtype=float32)/sqrt(16)
299
300     def work(self, input_items, output_items):
301         in0 = input_items[0]
302
303         k = 0
304         max_index = len(in0)-29
305         while k < max_index:
306
307             cor1 = dot(real(in0[k:k+30]),
    self._template)/sqrt(dot(real(in0[k:k+7]),real(in0[k:k+7]))+dot(real(in0[k+23:k+30]),real(in0[
308             cor2 = dot(imag(in0[k:k+30]),
    self._template)/sqrt(dot(imag(in0[k:k+7]),imag(in0[k:k+7]))+dot(imag(in0[k+23:k+30]),imag(in0[
309
310
311             if cor1 > self._detection_threshold and cor1 >= cor2:
312                 codeword = real(in0[k+7:k+23])
313                 cor3 = dot(self._C,codeword)/sqrt(dot(codeword,codeword))
314                 if max(cor3) > self._detection_threshold:
315                     k += 30
316                     print(chr(argmax(cor3)), end='')
317                 else:
318                     k += 1
319
320
321             elif cor2 > self._detection_threshold:
322                 codeword = imag(in0[k+7:k+23])
323                 cor3 = dot(self._C,codeword)/sqrt(dot(codeword,codeword))
324                 if max(cor3) > self._detection_threshold:
325                     k += 30
326                     print(chr(argmax(cor3)), end='')
```

```
327                 else:
328                     k += 1
329 
330             else:
331                 k += 1
332 
333         if max_index < 1:
334             return 0
335         elif k > max_index:
336             return k
337         else:
338             return max_index
```

APPENDIX E

KICKSAT VERIFICATION REPORTS

ELaNa V CubeSat Verification Report

Mass Properties

Revision:	4
Date:	9/3/2013
Authors:	Zachary Manchester Email: zrm3@cornell.edu Phone: 607-279-1358
Requirements Verified:	3.2.3, 3.2.4

Coordinate System:

In accordance with requirement 3.2.3, the figure 1 indicates the reference coordinate system used by KickSat. The RBF pin is not shown. Figure 1 does not show the remove before flight (RBF) pin, which is located on the –X face of the spacecraft.

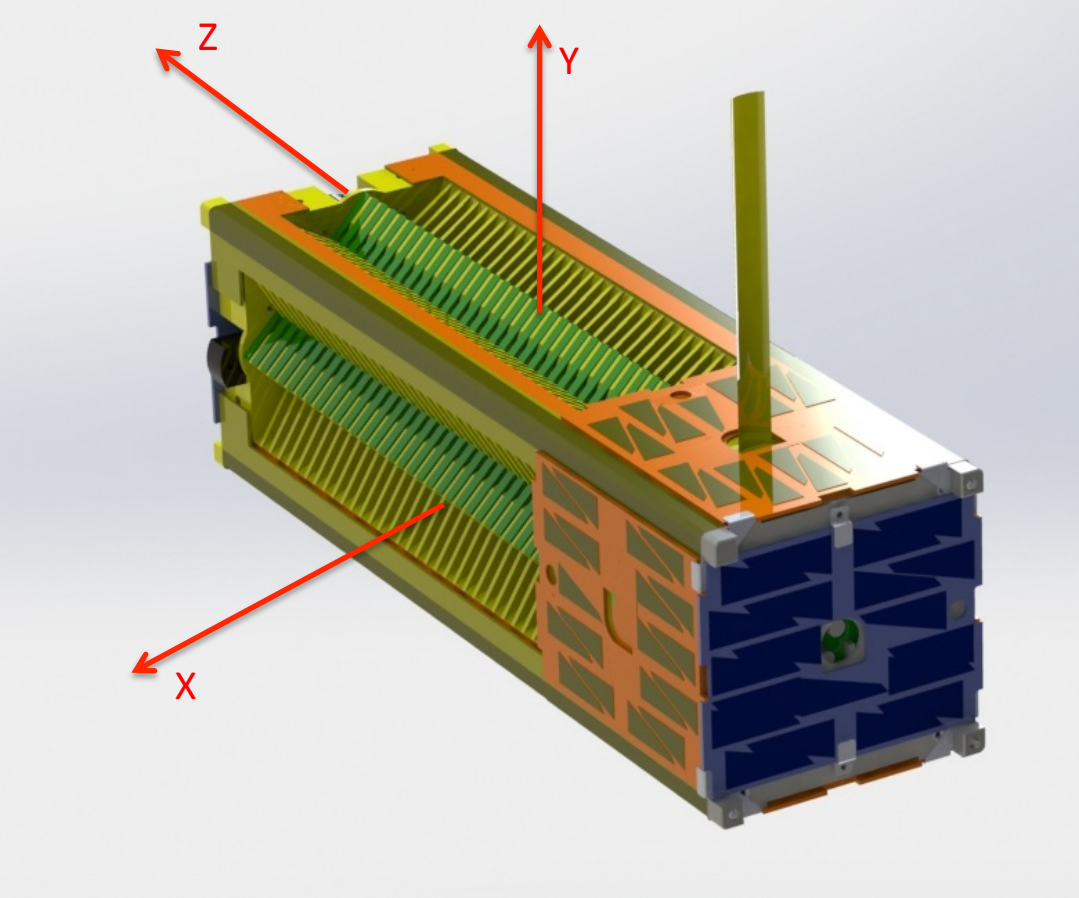


Figure 1: *KickSat Reference Coordinate System*

Mass Properties:

KickSat's mass was determined by weighing the actual flight unit. The center of mass was determined by balancing the flight unit on a straight edge. The moments of inertia were measured using a bifilar pendulum apparatus. The products of inertia were calculated by analyzing a simplified model of the mass distribution of the spacecraft to calculate the principal axes of inertia, after which a full inertia matrix consistent with both the measured products of inertia and calculated principal axes was calculated. All values are referenced to the geometric center of the spacecraft and the coordinate system used is that depicted in figure 1. All measurements were made without the RBF pin and with all deployables in their stowed (launch) configurations.

Mass: 2.68 kg

Center of Mass: X = -1.77 mm
Y = 0.29 mm
Z = -14.8 mm

Moments of Inertia: $I_{xx} = 3.87 \times 10^4$ kg mm^2
$I_{yy} = 3.66 \times 10^4$ kg mm^2
$I_{zz} = 4.96 \times 10^3$ kg mm^2

Products of inertia: $I_{xy} = -4.77 \times 10^2$ kg mm^2
$I_{xz} = 1.12 \times 10^3$ kg mm^2
$I_{yz} = -4.81 \times 10^2$ kg mm^2

Mass and center of mass values are in compliance with requirement 3.2.4. Variations in the final mass properties of no more than ±10% are anticipated.

ELaNa V CubeSat Verification Report

Venting Analysis

Revision:	1
Date:	7/15/2013
Authors:	Zachary Manchester Email: zrm3@cornell.edu Phone: 607-279-1358
Requirements Verified:	3.2.8

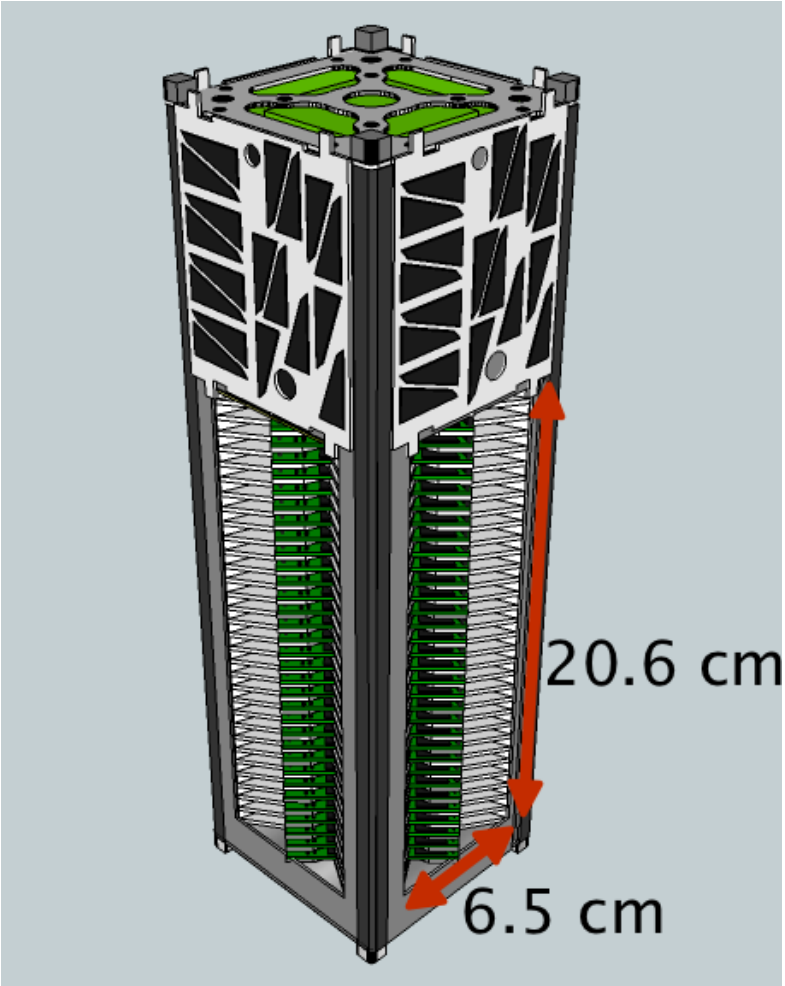


Figure 1: *Exterior of KickSat Spacecraft*

The following analysis considers the (conservatively large) full volume of the 3U CubeSat in the volume-to-vent-area ratio calculation.

$$\text{Total Volume: } V = (34\text{ cm}) * (10\text{ cm}) * (10\text{ cm}) = 3400\text{ cm}^3$$

The vent area is taken to be the four large cutouts on the +X, –X, +Y, and –Y faces of the CubeSat depicted in figure 1. Note that this area is conservatively small as it neglects several smaller cutouts.

$$\text{Total Area: } A = 4 * (6.5\text{ cm}) * (20.6\text{ cm}) = 535.6\text{ cm}^2$$

The resulting volume-to-vent-area ratio meets requirement 3.2.8 specified in the ELaNa V CubeSat to P-POD Interface Control Document:

$$\frac{V}{A} = \frac{3400\text{ cm}^3}{535.6\text{ cm}^2} \approx 6.348\text{ cm} \approx 2.5\text{ inches} < 2000\text{ inches}$$

ELaNa V CubeSat Verification Report

Functional Test

Revision:	2
Date:	11/6/2013
Authors:	Zachary Manchester Email: zrm3@cornell.edu Phone: 607-279-1358
Requirements Verified:	3.2.5.2, 3.3.1.1, 3.3.1.4, 3.3.5

Overview:

A functional (day in the life) test was performed on the KickSat spacecraft to ensure compliance with ELaNa V ICD requirements 3.4.5.2 (deployable time delay), 3.3.1.1 (deployment switch function), 3.3.1.4 (deployment switch toggle), and 3.3.5 (transmission exclusion).

Test Setup:

Functional tests were performed in building 17 at NASA Ames Research Center. An electronics bench with grounded mats and wrist straps was used and proper ESD precautions were followed. CubeSat power status was verified using power status LEDs on the avionics boards as shown in figure 1. Times were recorded using a wall clock.

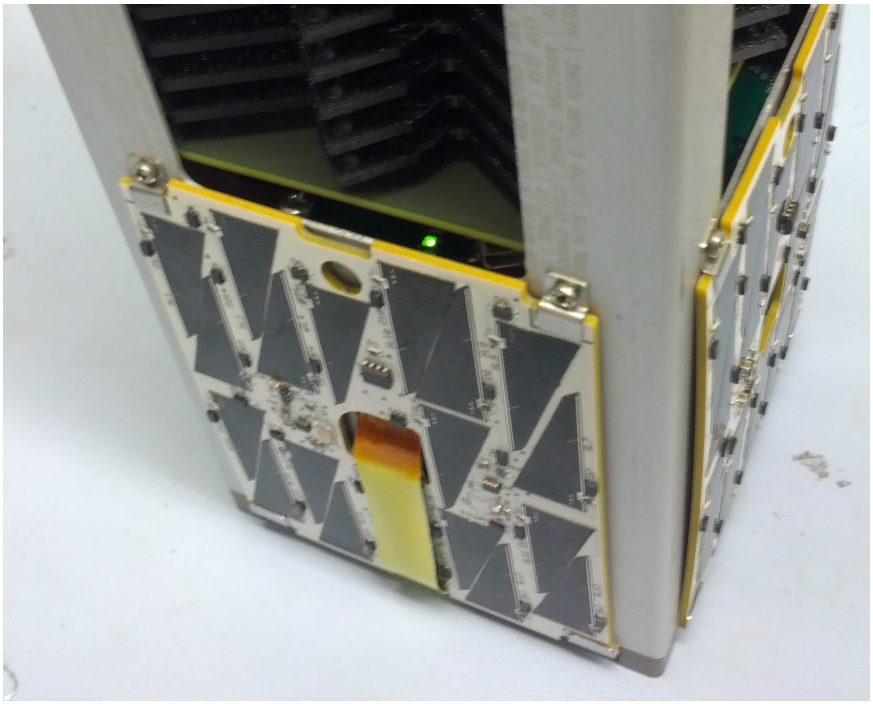


Figure 1: *CubeSat Power Status LED*

Test Procedure:

The following table lists the steps performed during the test along with the times they were performed.

Step	Time
1. Ensure ESD precautions are being followed.	12:37 PM
2. Place KickSat on test bench. Ensure satellite is powered off with RBF pin inserted and deployment switch engaged.	12:38 PM

3. Remove RBF pin. Ensure deployment switch remains engaged.	12:38 PM
4. Verify satellite remains powered off.	12:38 PM
5. Re-insert RBF pin. Ensure deployment switch remains engaged.	12:38 PM
6. Verify that satellite remains powered off.	12:38 PM
7. Remove RBF pin. Ensure deployment switch remains engaged.	12:39 PM
8. Verify that the satellite remains powered off.	12:39 PM
9. Release deployment switch and wait 5 minutes.	12:40 PM
10. Re-engage deployment switch. Ensure satellite is powered off.	12:45 PM
11. Release deployment switch.	12:47 PM
12. Record time at antenna deployment.	1:37 PM
13. Record time of first RF transmission.	1:37 PM
14. Re-insert RBF pin. Ensure that the satellite is powered off.	1:38 PM
15. Re-engage deployment switch.	1:38 PM
16. Functional test complete.	1:38 PM

Results:

All ICD requirements were satisfied. Requirement 3.4.5.2 specifies a minimum of 15 minutes between ejection from the P-POD and any deployments. A delay of 50 minutes was measured. Requirements 3.3.1.1 and 3.3.1.4 specify that the deployment switch should cut all power to the CubeSat and that the switch should tolerate toggling between on and off states. This was verified by actuating the switch multiple times. Requirement 3.3.5 specifies a minimum time of 45 minutes between ejection from the P-POD and any RF transmissions. A delay of 50 minutes was measured.

ELaNa V CubeSat Verification Report

Vibration Test

Revision:	2
Date:	10/09/2013
Authors:	Zachary Manchester Email: zrm3@cornell.edu Phone: 607-279-1358
Requirements Verified:	3.4.3, 3.4.4, 3.4.8

Overview:

KickSat underwent vibration testing from August 26-28, 2013 at Space Systems Loral in Palo Alto, California. The testing process conformed to requirements 3.4.3, 3.4.4, and 3.4.8 and was performed to the proto-flight levels listed in table 8 of appendix D of the ICD.

Test Setup:

All tests were performed on a Ling shaker table. A 1" thick aluminum adapter plate was machined to bolt the TestPOD to the 2" by 2" bolt hole pattern on the shaker table. A mechanical drawing of this plate is provided in Appendix A. Three 3-axis accelerometers were used – one attached to the outside corner of the TestPOD, one attached to the CubeSat structure, and a control accelerometer attached to the adapter plate as shown in figure 1.

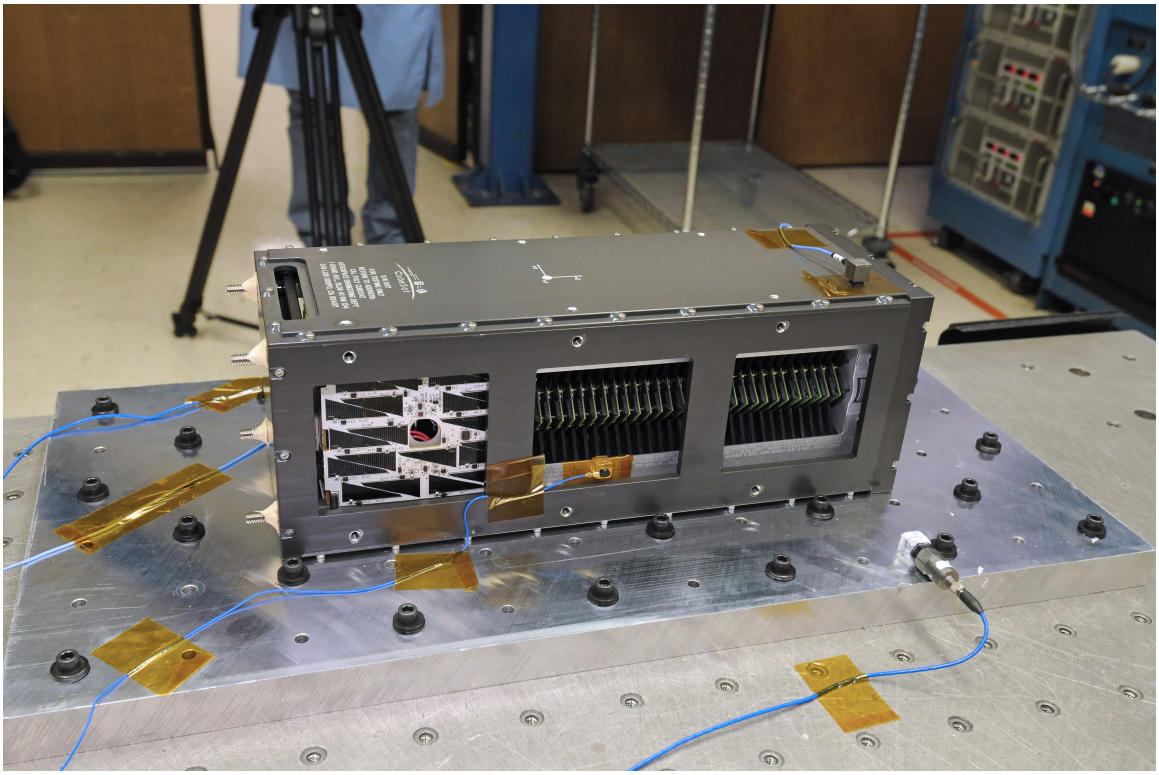


Figure 1: *Test Setup Showing Placement of Accelerometers*

Y-axis tests were performed with the shaker in a vertical orientation and the TestPOD bolted to a head expander as shown in figure 2, while X-axis and Z-axis tests were performed with the shaker in a horizontal orientation using a slip table as shown in figures 3 and 4.

Figure 2: *Y-Axis Test Setup*

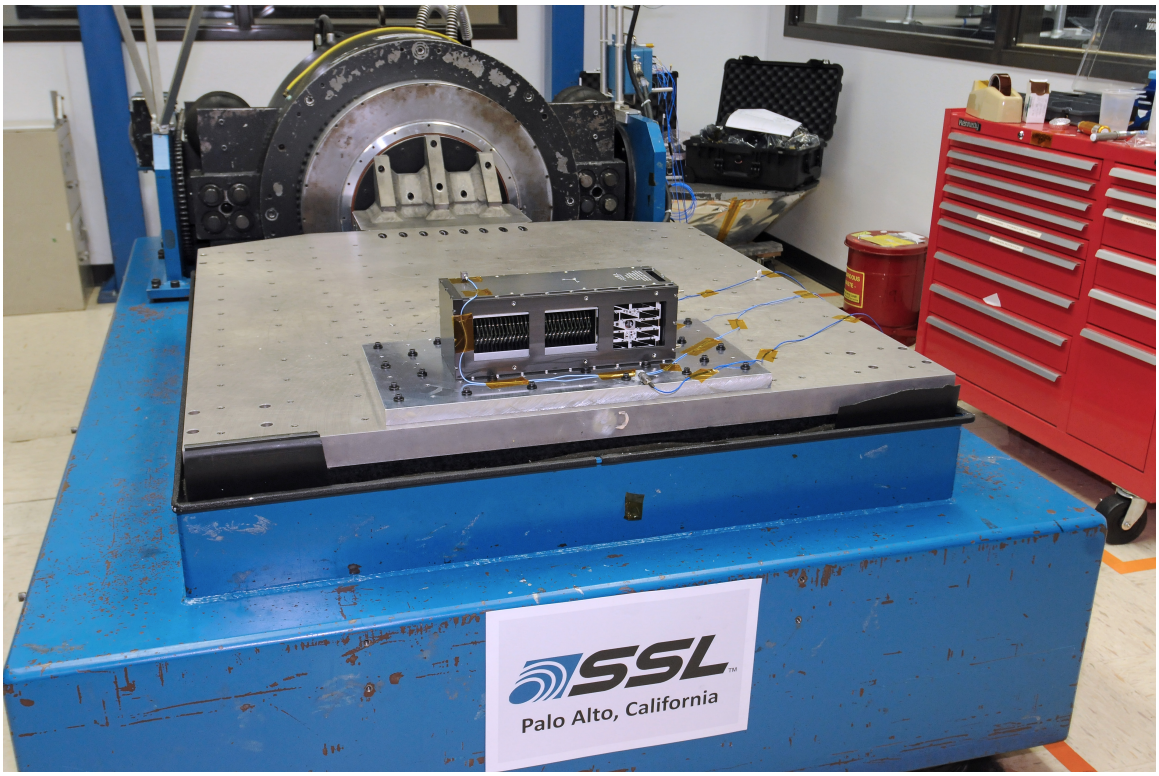


Figure 3: *X-Axis Test Setup*

Figure 4: *Z-Axis Test Setup*

Test Procedure:

The following table lists the steps performed during the test along with the approximate times they were performed.

Step	Date/Time
1. Perform functional test of spacecraft	8/26 10:10 AM
2. Perform CubeSat Acceptance Checklist	8/26 11:00 AM
3. Install CubeSat into TestPOD	8/26 1:15 AM
4. Bolt TestPOD to adapter plate	8/26 2:30 PM
5. Attach adapter plate to head expander for Y-axis test	8/26 3:00 PM
6. Attach accelerometers	8/26 3:15 PM
7. Verify that radio is not transmitting	8/26 3:45 PM
8. Perform initial Y-axis sine sweep	8/26 3:57 PM
9. Perform Y-axis short duration random vibration test	8/26 4:09 PM
10. Perform Y-axis steady state random vibration test	8/26 4:34 PM
11. Perform post Y-axis sine sweep	8/26 4:42 PM
12. Compare pre/post sine sweeps	8/26 4:45 PM
13. Verify that radio is not transmitting	8/26 4:48 PM
14. Remove test fixture from head expander	8/27 9:05 AM
15. Inspect TestPOD and KickSat for damage	8/27 9:15 AM
16. Set up slip table for X and Z-axis tests	8/27 9:30 AM
17. Mount test fixture to slip table for X-axis tests	8/27 10:30 AM

18. Verify that radio is not transmitting	8/27 11:00 AM
19. Perform initial X-axis sine sweep	8/27 11:16 AM
20. Perform X-axis short duration random vibration test	8/27 1:13 PM
21. Perform X-axis steady state random vibration test	8/27 1:23 PM
22. Perform post X-axis sine sweep	8/27 1:39 PM
23. Compare pre/post sine sweeps	8/27 1:43 PM
24. Verify that radio is not transmitting	8/27 1:45 PM
25. Remove test fixture from slip table	8/27 1:50 PM
26. Inspect TestPOD and KickSat for damage	8/27 2:15 PM
27. Mount test fixture to slip table for Z-axis tests	8/28 10:40 AM
28. Verify that radio is not transmitting	8/28 11:50 AM
29. Perform initial Z-axis sine sweep	8/28 11:57 AM
30. Perform Z-axis short duration random vibration test	8/28 1:10 PM
31. Perform Z-axis steady state random vibration test	8/28 1:40 PM
32. Perform post Z-axis sine sweep	8/28 1:49 PM
33. Compare pre/post sine sweeps	8/28 1:55 PM
34. Verify that radio is not transmitting	8/28 2:00 PM
35. Remove test fixture from slip table	8/28 2:05 PM
36. Remove TestPOD from adapter plate	8/28 2:25 PM
37. Remove KickSat from TestPOD	8/28 2:30 PM
38. Inspect KickSat and TestPOD for damage	8/28 3:20 PM
39. Perform functional test	8/28 4:45 PM
40. Test complete	8/28 5:00 PM

Results:

KickSat had no visible or audible signs of damage during vibration testing. The beacon radio remained off while in the TestPOD, and pre/post sine sweep plots were consistent. KickSat passed both pre- and post-vibration functional tests. Frequency response plots taken from the accelerometer mounted to the top corner of the TestPOD as well as control accelerometer data for random vibe tests for the axis under test are given below.

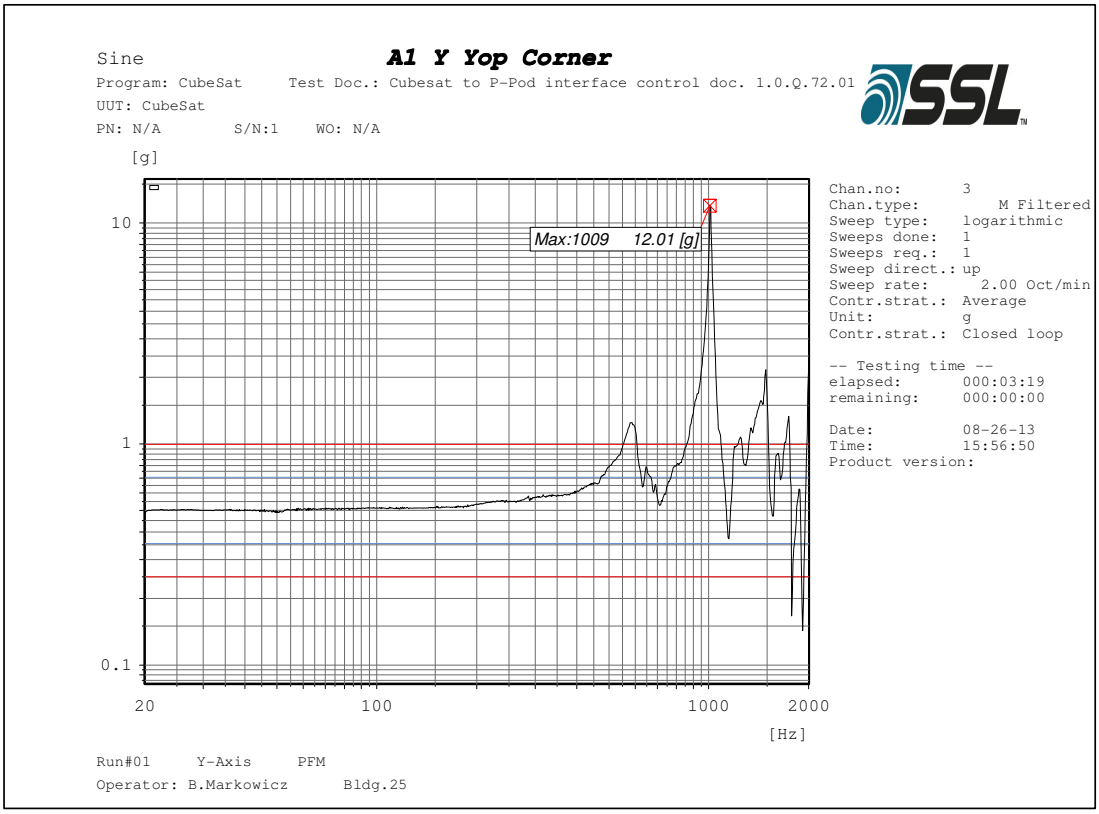


Figure 5: *Y-Axis Initial Sine Sweep*

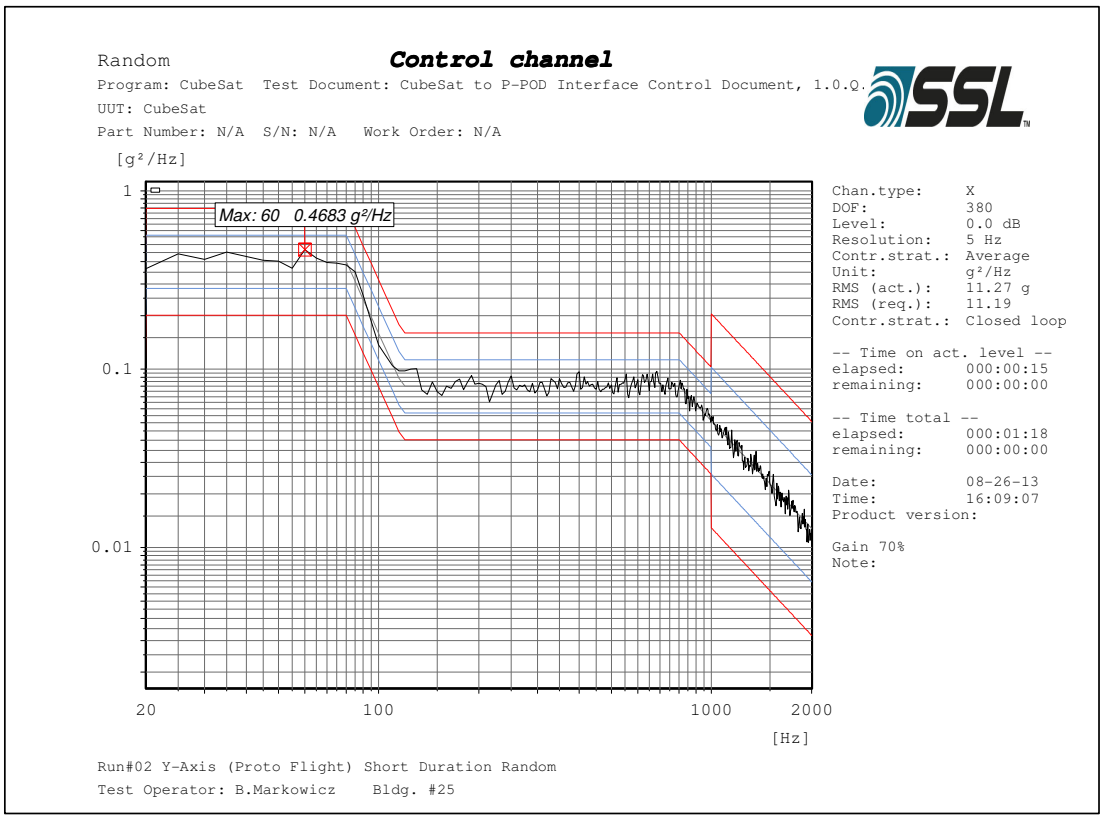


Figure 6: *Y-Axis Short Duration Random*

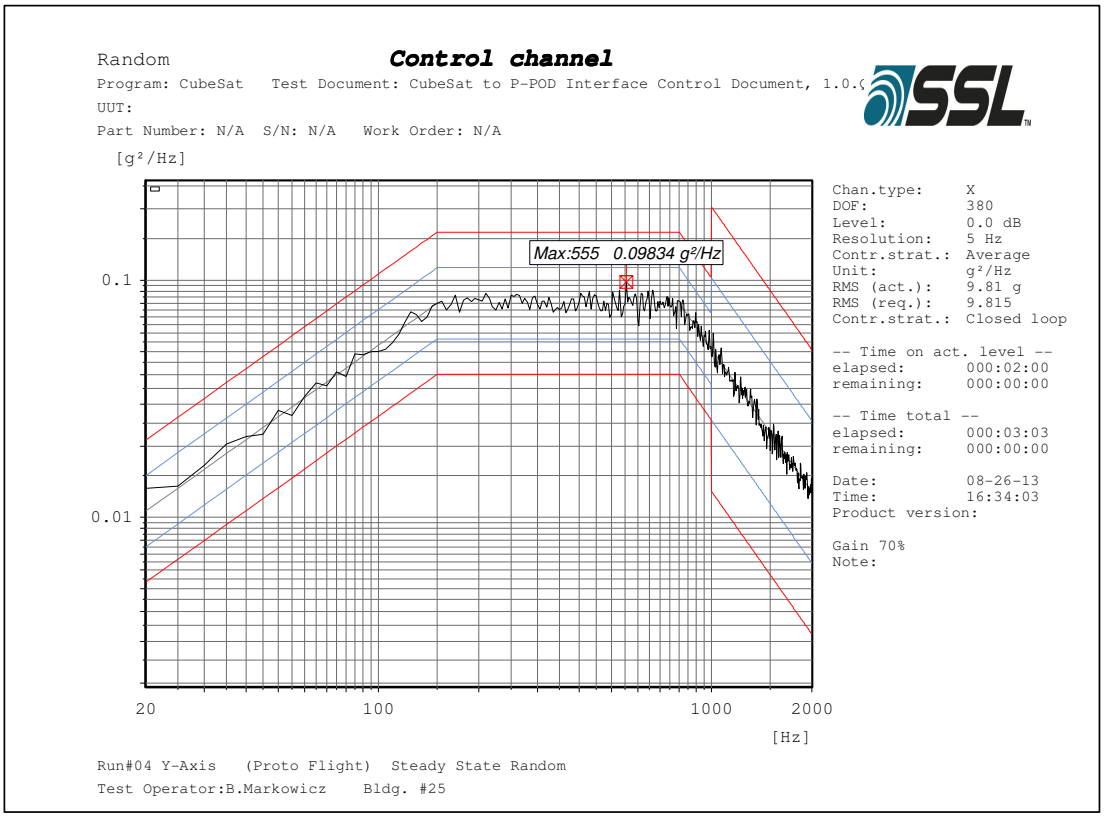


Figure 7: *Y-Axis Steady State Random*

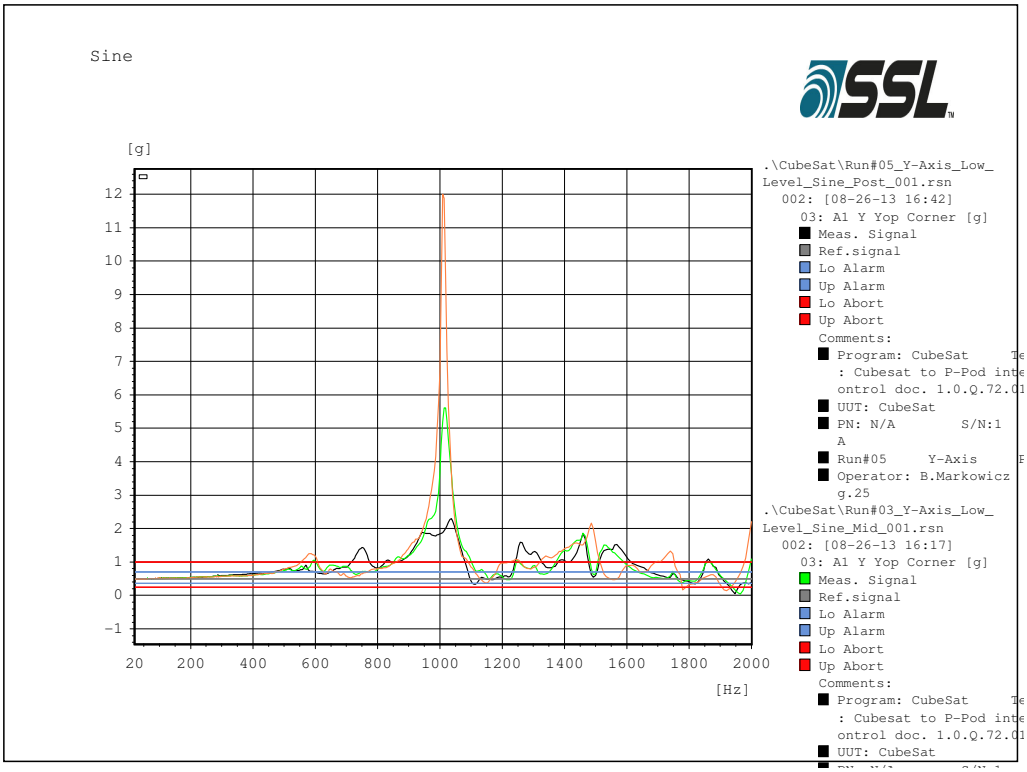


Figure 8: *Y-Axis Pre/Post Sine Sweep Overlay*

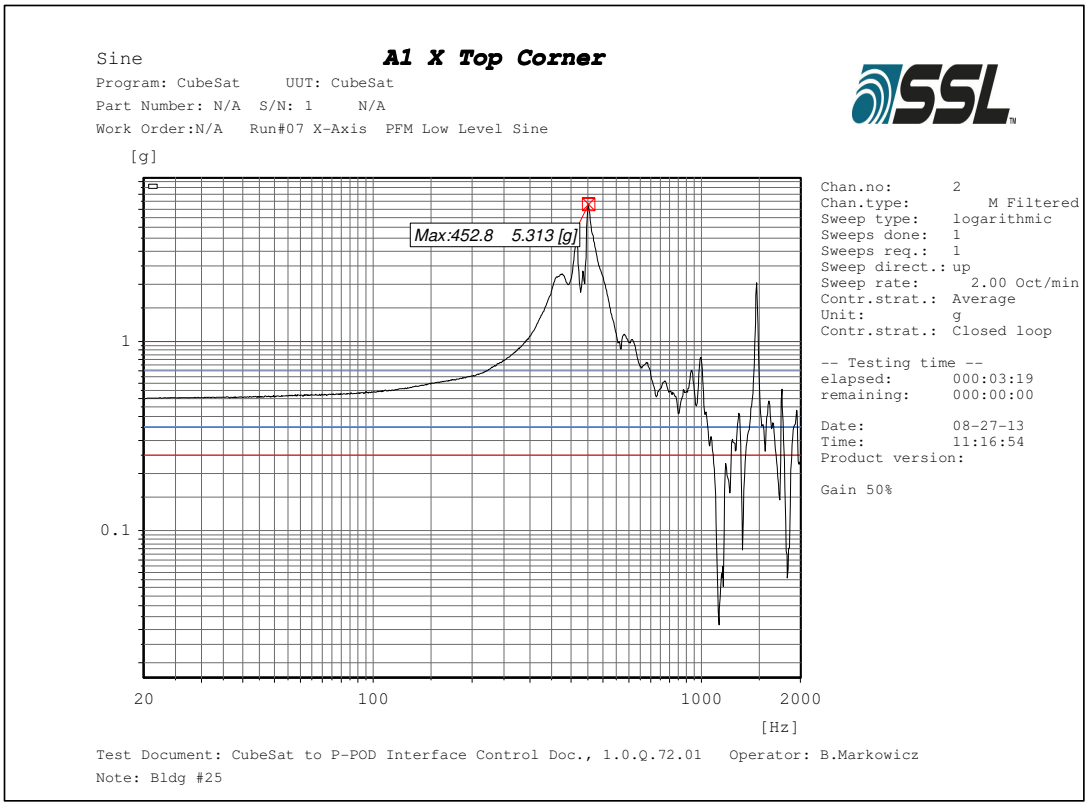


Figure 9: *X-Axis Initial Sine Sweep*

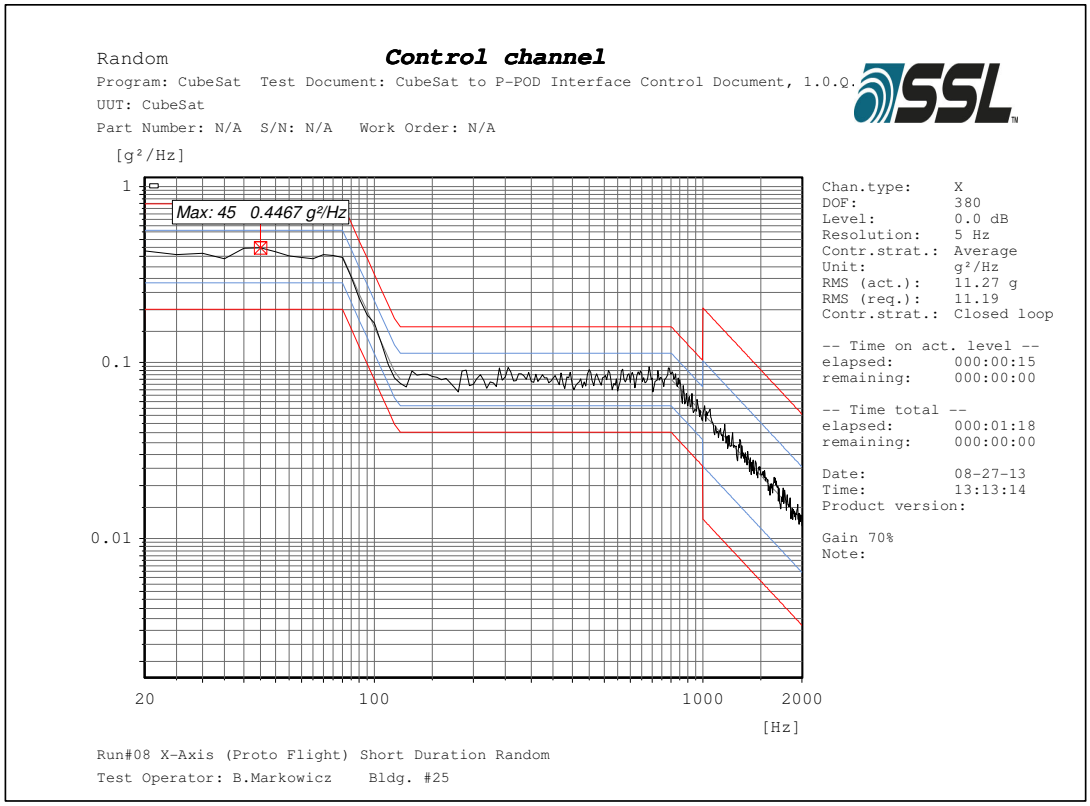


Figure 10: *X-Axis Short Duration Random*

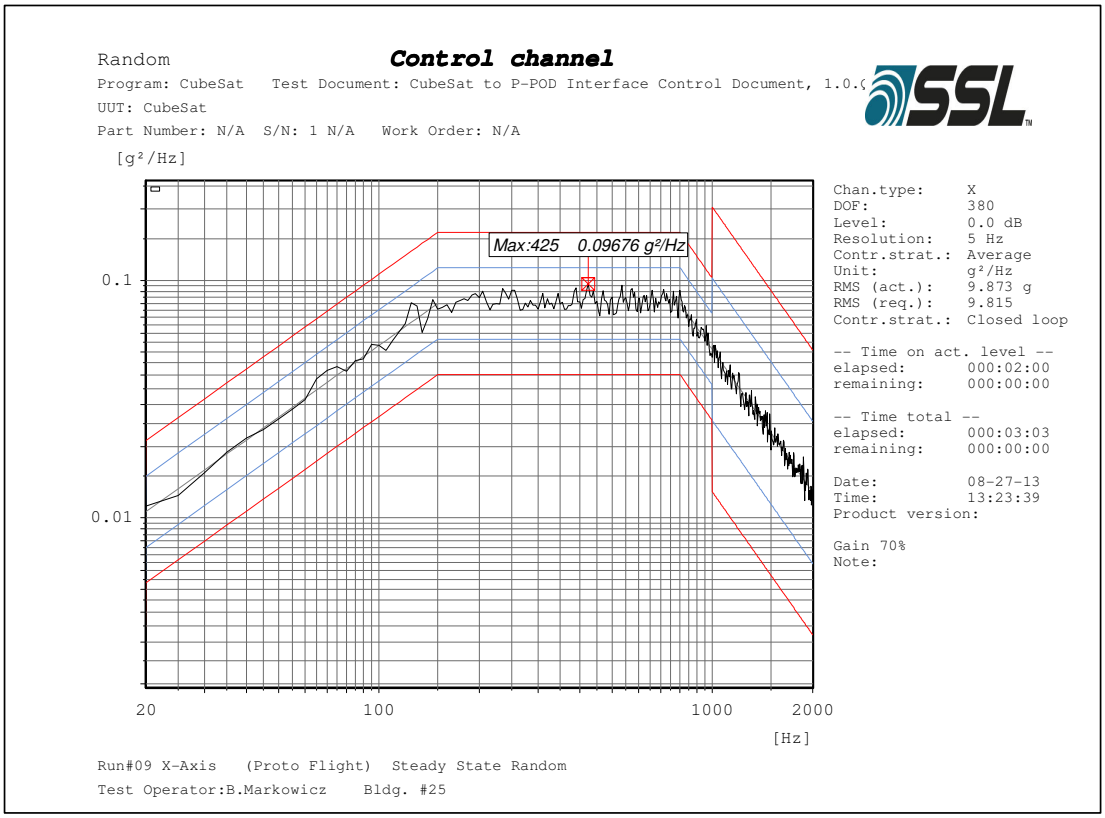


Figure 11: *X-Axis Steady State Random*

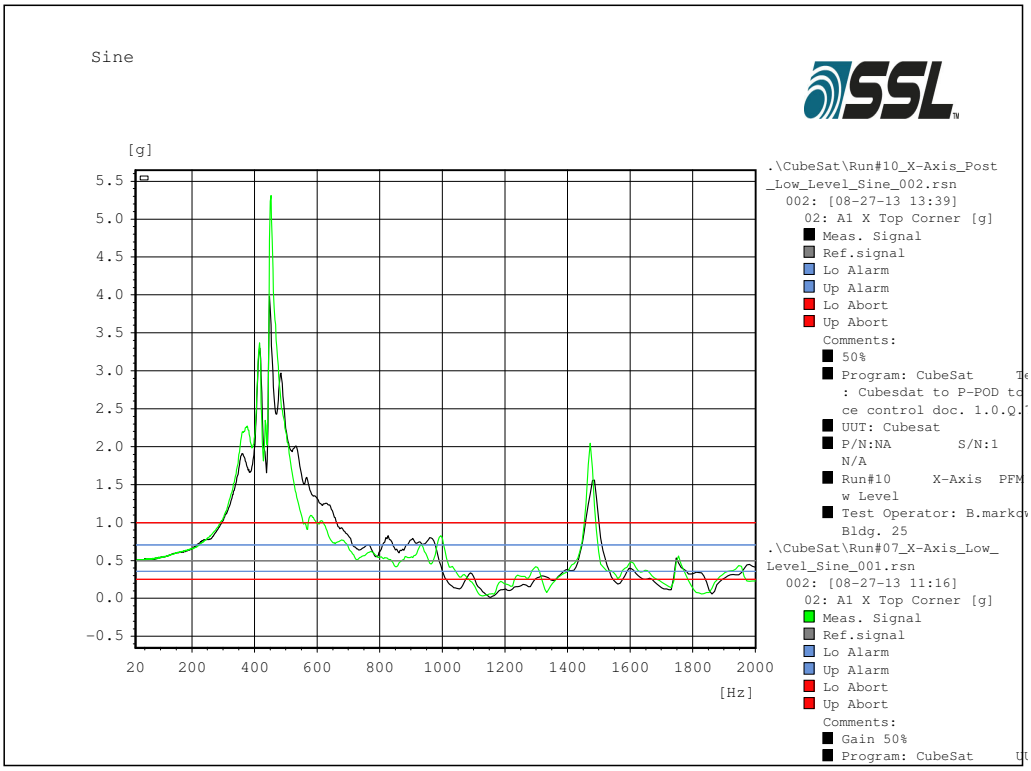


Figure 12: *X-Axis Pre/Post Sine Sweep Overlay*

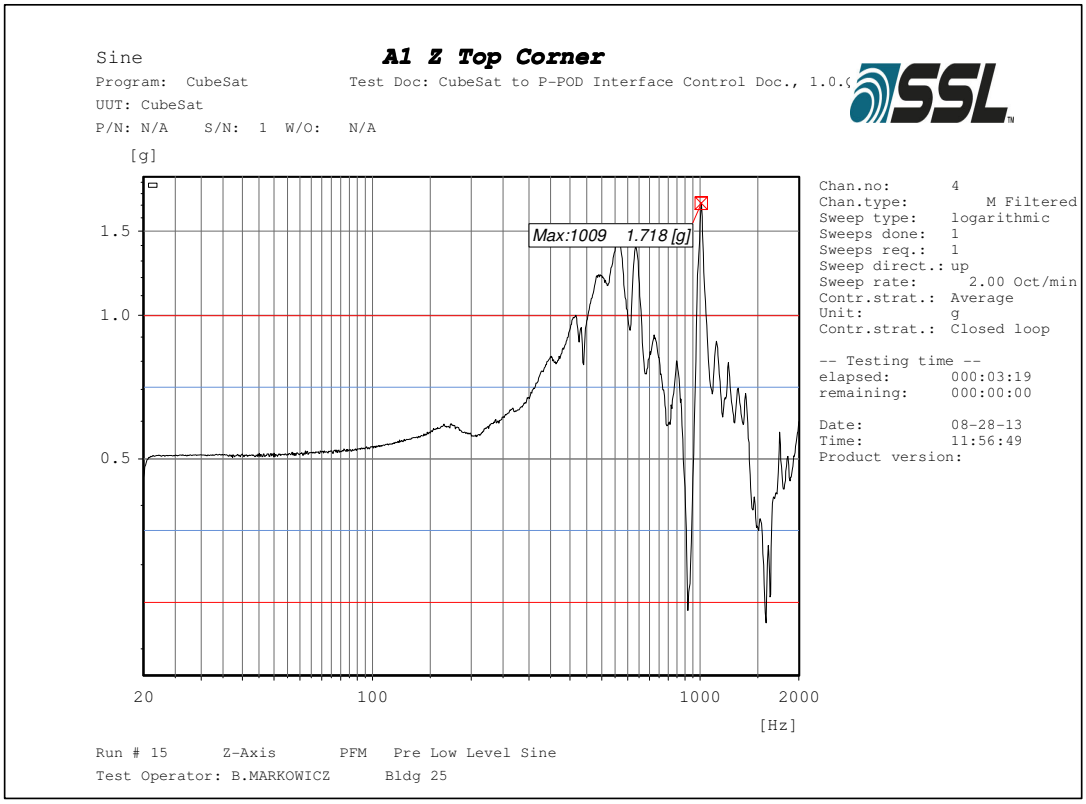


Figure 13: *Z-Axis Initial Sine Sweep*

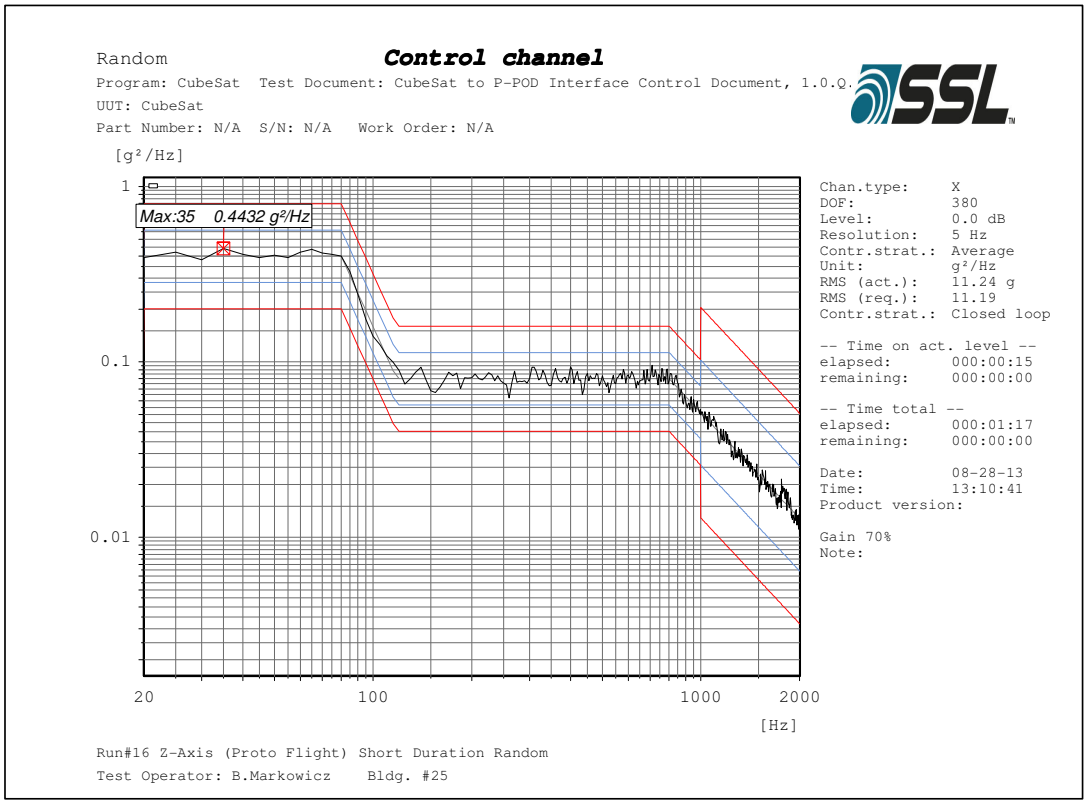


Figure 14: *Z-Axis Short Duration Random*

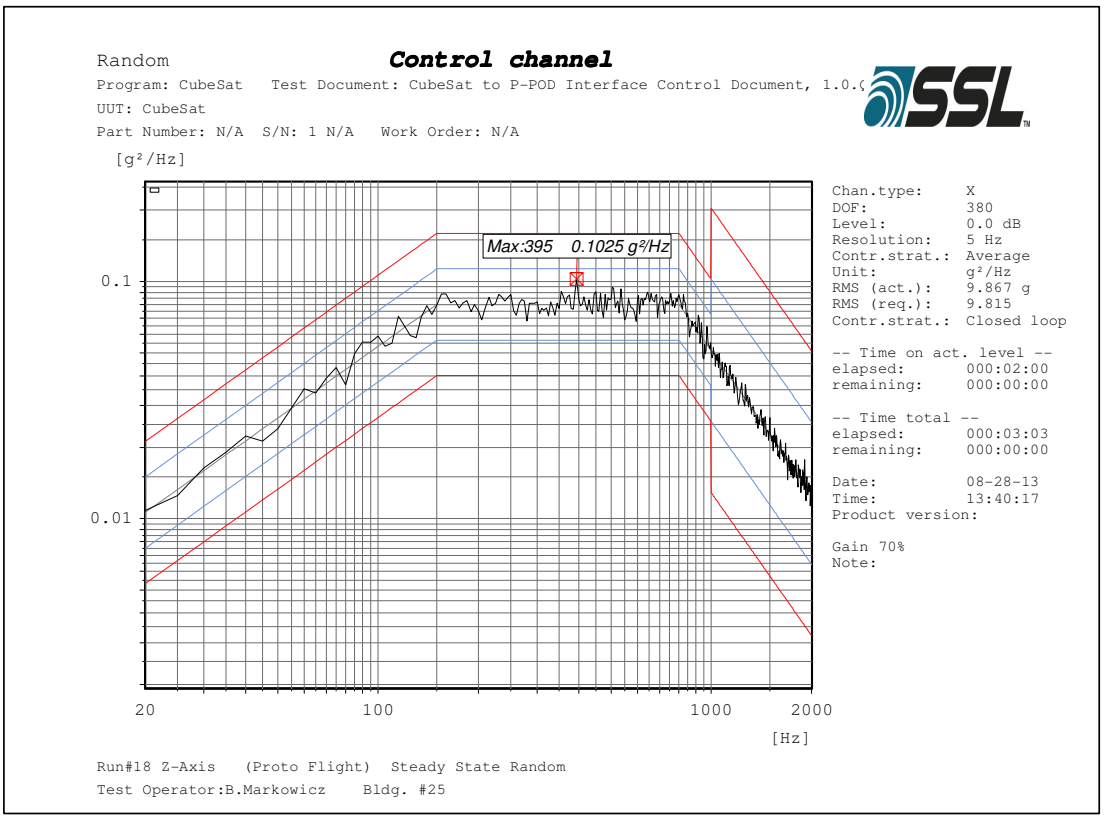


Figure 15: *Z-Axis Steady State Random*

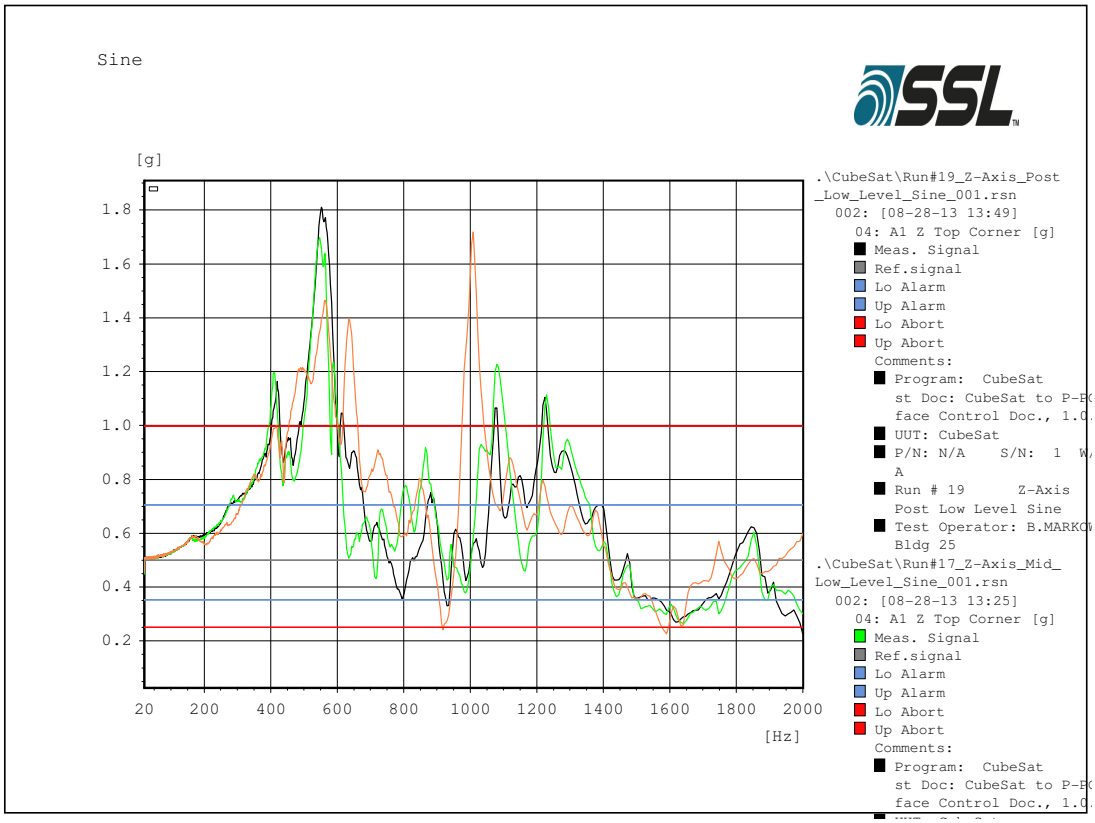
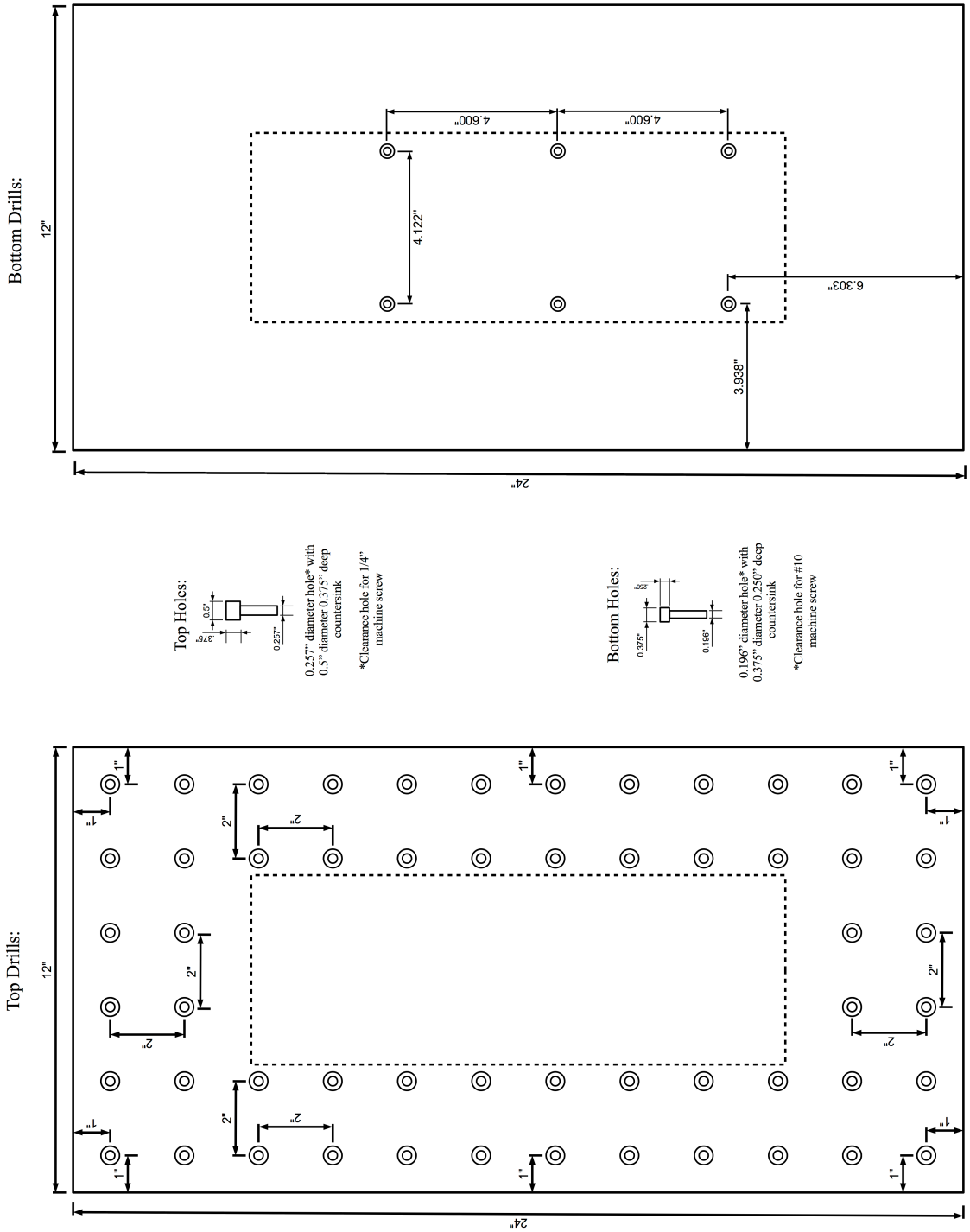


Figure 16: *Z-Axis Pre/Post Sine Sweep Overlay*

Appendix A:

TestPOD Adapter Plate Mechanical Drawing



ELaNa V CubeSat Verification Report

Thermal Vacuum Test

Revision:	2
Date:	8/28/2013
Authors:	Zachary Manchester Email: zrm3@cornell.edu Phone: 607-279-1358
Requirements Verified:	3.4.5, 3.4.5.1

Overview:

KickSat underwent bakeout testing overnight from August 21-22, 2013 at the NASA Ames Research Center Engineering Evaluation Laboratory. The bakeout process conformed to requirement 3.4.5 by baking at a nominal pressure of 10^{-4} Torr and conformed to requirement 3.4.5.1 by baking at a nominal temperature of 60° C for six hours with a temperature ramp rate of less than 5° C per minute.

Test Setup:

Figure 1 shows the vacuum chamber used for KickSat's bakeout.

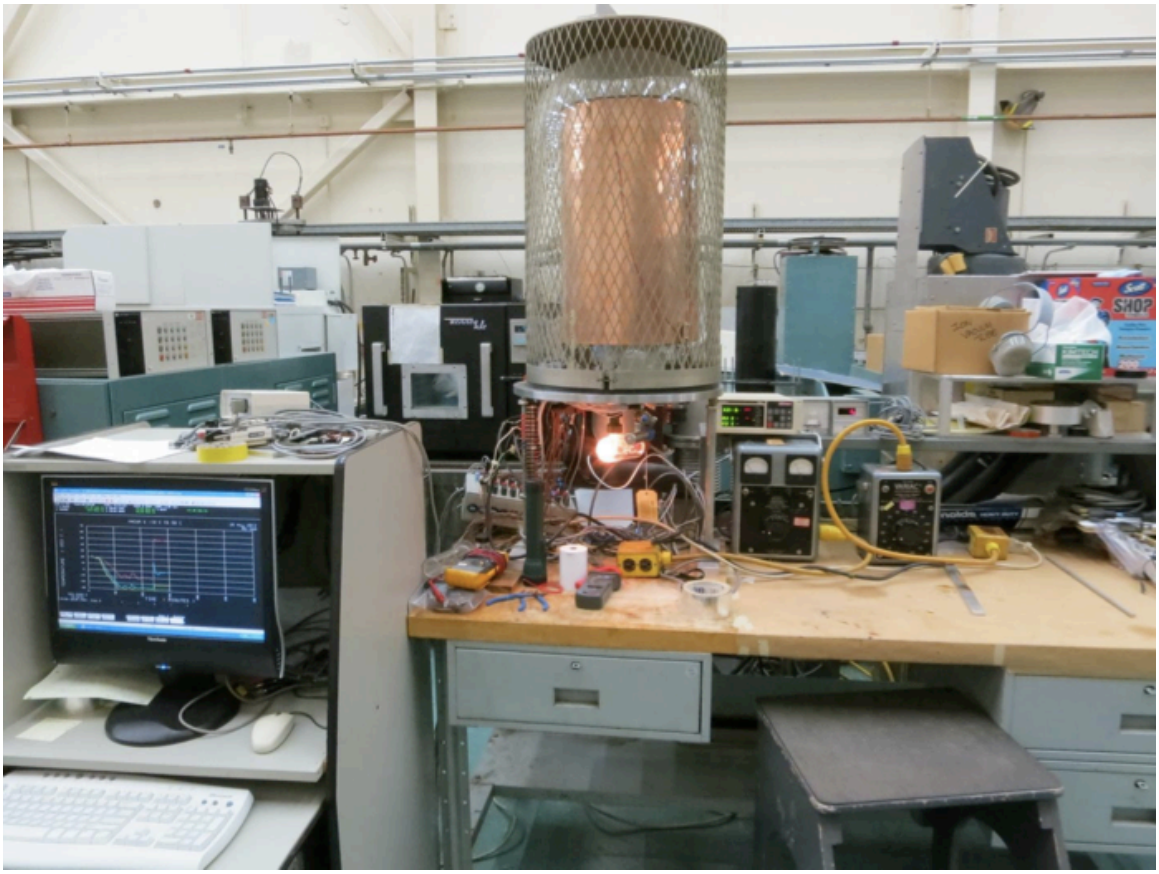


Figure 1: *Vacuum Chamber*

KickSat was fitted with two thermocouples – one each on the +x and –x exterior faces. The thermocouples were attached to the anodized aluminum surface of the CubeSat structure with kapton tape as shown in Figure 2.

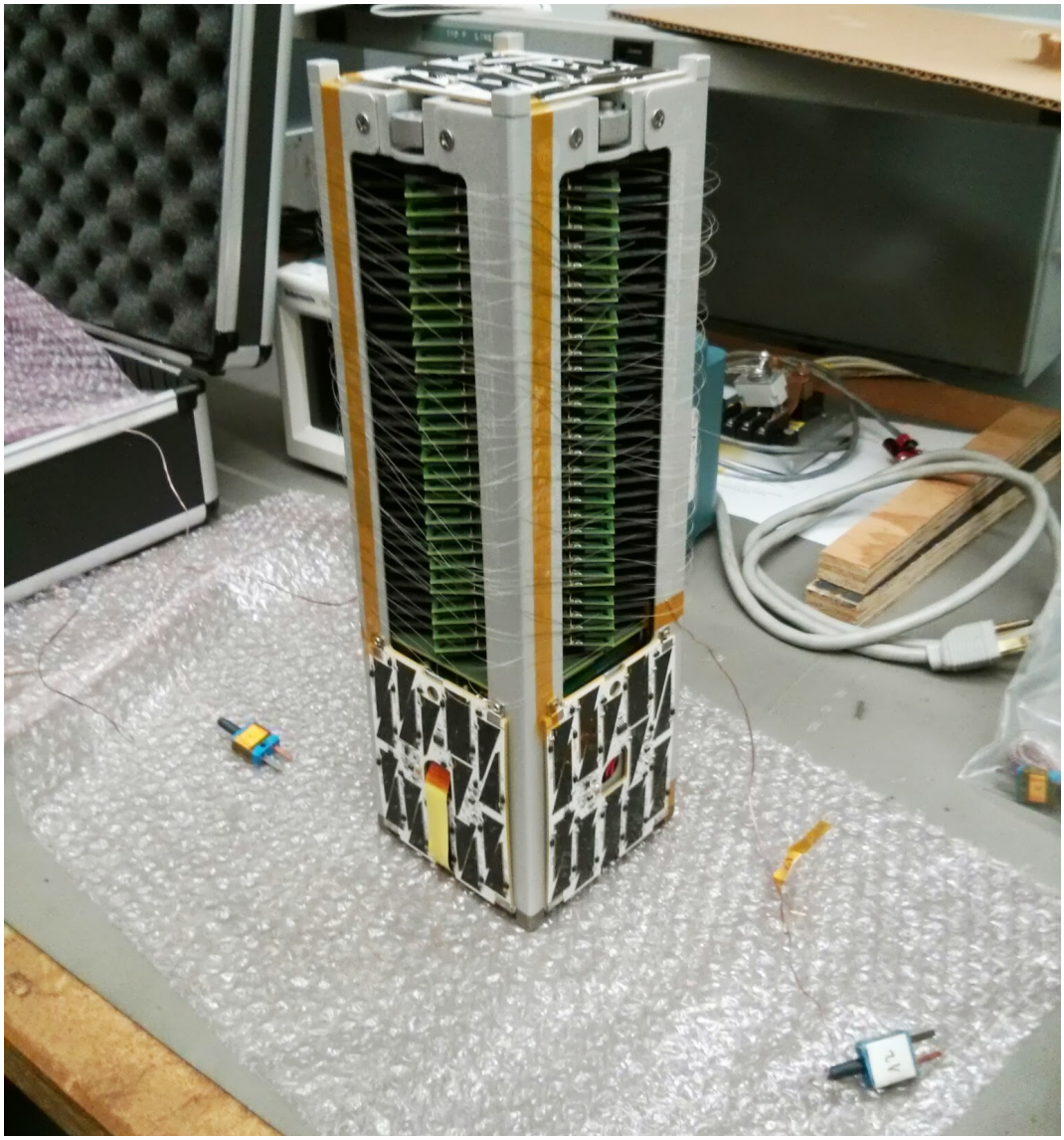


***Figure 2**: KickSat with Thermocouples Fitted*

Figure 3 shows KickSat inside the vacuum chamber with thermocouples attached prior to lowering the bell jar.

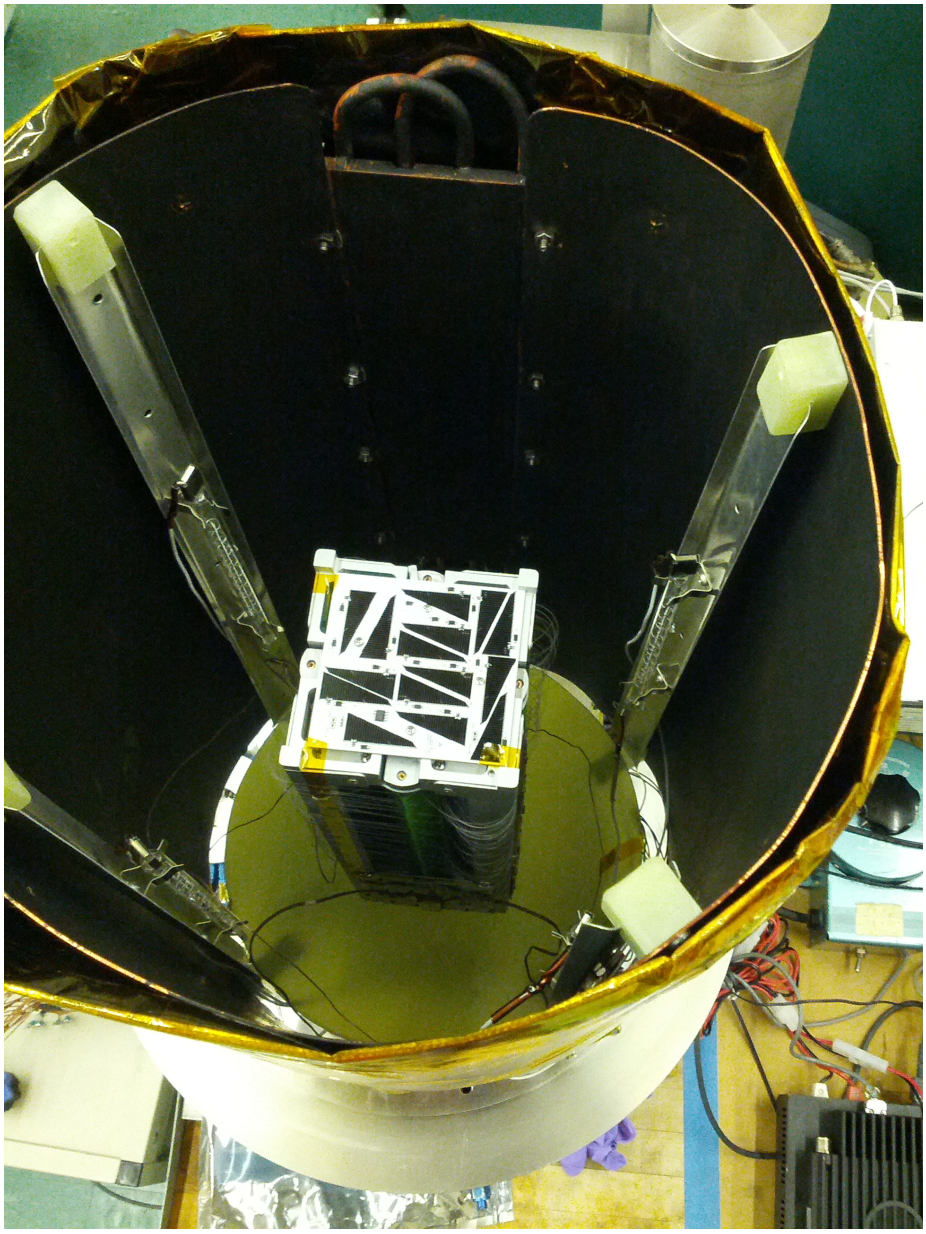


Figure 3: *KickSat in Vacuum Chamber*

Test Procedure:

The following table lists the steps performed during the test along with the approximate times they were performed.

Step	Time
1. Attach thermocouples to CubeSat and note locations	3:50 PM
2. Check thermocouple readings to ensure they are close to room temperature	3:53 PM

3. Perform functional test of CubeSat a. Power on CubeSat b. Receive radio packets from Stensat c. Receive radio packets from MHX2420 d. Verify reaction wheels turn e. Power off CubeSat	3:55 PM
4. Clean hardware surfaces	4:00 PM
5. Place satellite into vacuum chamber	4:07 PM
6. Connect thermocouples	4:08 PM
7. Seal vacuum chamber	4:10 PM
8. Bring vacuum chamber pressure to 1×10^{-4} Torr	11:05 PM
9. Ensure data logging is enabled	11:05 PM
10. Slowly raise the heating element temperature while ensuring the chamber pressure does not rise above 5×10^{-4} Torr	11:06 PM
11. Maintain spacecraft temperature between 60° C and 65° C for 6 hours	2:56 AM
12. Slowly lower the heating element temperature until the spacecraft temperature reaches 30° C	9:45 AM
13. Return vacuum chamber to atmospheric pressure	2:10 PM
14. Remove CubeSat from vacuum chamber	2:15 PM
15. Perform functional test of CubeSat a. Power on CubeSat b. Receive radio packets from Stensat c. Receive radio packets from MHX2420 d. Verify reaction wheels turn e. Power off CubeSat	2:18 PM
16. Remove thermocouples	2:25 PM
17. Ensure all data is recorded	2:27 PM
18. Test complete	2:30 PM

Results:

Figure 4 shows the temperature and pressure data recorded during the bakeout. The raw pressure and temperature data is tabulated in Appendix A. The temperature readings of the two thermocouples agreed to within 2 degrees Celsius throughout the test. Significant outgassing was observed once the heaters were turned on, causing the pressure to rise to a maximum of 3.7×10^{-4} Torr before slowly decreasing to a minimum of 6.7×10^{-5} Torr at the end of the 6 hour bakeout period. In total, a minimum temperature of 60° C and maximum pressure of 3.7×10^{-4} were maintained for 6 hours and 50 minutes.

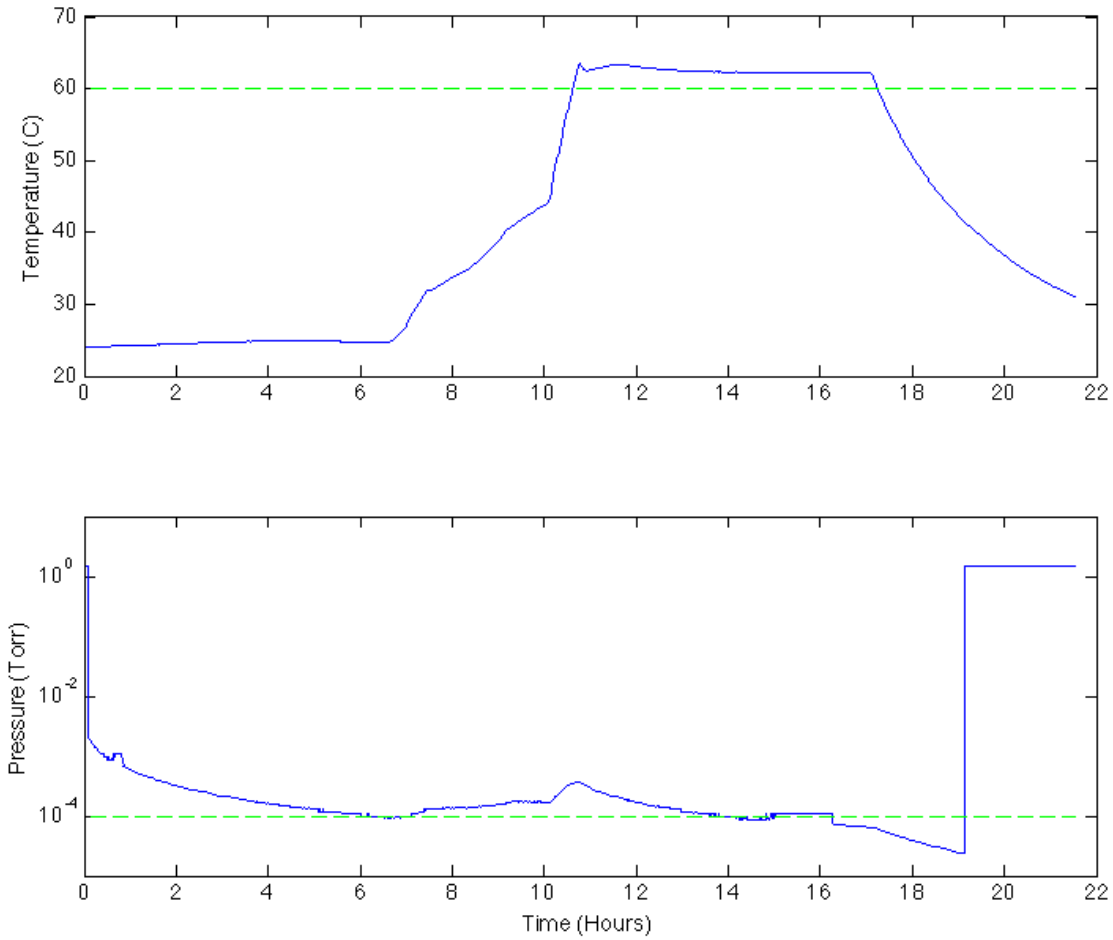


Figure 4: *Temperature and Pressure Plots*

Appendix A:

KICKSAT BAKEOUT
60 C for 6 HOURS
Service Request: RD-472
FILE: RD472 KICKSAT-1

TEMPERATURE	TEMPERATURE	PRESSURE	TIME
TCT	TCT	TERRA-VAC	586,2,0
4	5	16	
TC # 1	TC # 2	VACUUM	TIME
C	C	TORR	SECONDS
24.01	24.15	1.47E+00	8/21/13 16:32
24	24.13	1.47E+00	8/21/13 16:33
24	24.12	1.98E-03	8/21/13 16:34
24.01	24.13	1.88E-03	8/21/13 16:35
24.03	24.13	1.78E-03	8/21/13 16:36
24.02	24.12	1.78E-03	8/21/13 16:37
24.04	24.13	1.68E-03	8/21/13 16:38
24.03	24.13	1.58E-03	8/21/13 16:39
24.03	24.13	1.48E-03	8/21/13 16:40
24.04	24.12	1.48E-03	8/21/13 16:41
24.04	24.12	1.38E-03	8/21/13 16:42
24.03	24.12	1.38E-03	8/21/13 16:43
24.04	24.13	1.38E-03	8/21/13 16:44
24.04	24.13	1.28E-03	8/21/13 16:45
24.06	24.13	1.28E-03	8/21/13 16:46
24.06	24.14	1.18E-03	8/21/13 16:47
24.06	24.13	1.18E-03	8/21/13 16:48
24.07	24.14	1.18E-03	8/21/13 16:49
24.06	24.13	1.08E-03	8/21/13 16:50
24.07	24.14	1.08E-03	8/21/13 16:51
24.07	24.13	1.08E-03	8/21/13 16:52
24.08	24.12	1.08E-03	8/21/13 16:53
24.09	24.15	9.78E-04	8/21/13 16:54
24.08	24.15	9.79E-04	8/21/13 16:55
24.09	24.13	9.81E-04	8/21/13 16:56
24.08	24.14	9.81E-04	8/21/13 16:57
24.1	24.14	8.80E-04	8/21/13 16:58
24.08	24.16	9.80E-04	8/21/13 16:59
24.1	24.15	8.81E-04	8/21/13 17:00
24.09	24.15	8.81E-04	8/21/13 17:01
24.09	24.15	8.79E-04	8/21/13 17:02
24.1	24.16	8.81E-04	8/21/13 17:03
24.11	24.15	8.80E-04	8/21/13 17:04
24.1	24.17	8.81E-04	8/21/13 17:05
24.09	24.15	1.10E-03	8/21/13 17:06
24.1	24.17	8.80E-04	8/21/13 17:07
24.11	24.17	1.10E-03	8/21/13 17:08
24.12	24.17	1.10E-03	8/21/13 17:09
24.11	24.17	1.10E-03	8/21/13 17:10
24.12	24.17	1.10E-03	8/21/13 17:11
24.12	24.17	1.10E-03	8/21/13 17:12

24.12	24.18	1.10E-03	8/21/13 17:13
24.13	24.17	1.09E-03	8/21/13 17:14
24.13	24.18	1.09E-03	8/21/13 17:15
24.13	24.19	1.09E-03	8/21/13 17:16
24.14	24.19	1.09E-03	8/21/13 17:17
24.13	24.17	6.98E-04	8/21/13 17:18
24.13	24.18	6.79E-04	8/21/13 17:19
24.14	24.19	6.68E-04	8/21/13 17:20
24.14	24.2	6.58E-04	8/21/13 17:21
24.14	24.19	6.48E-04	8/21/13 17:22
24.15	24.2	6.38E-04	8/21/13 17:23
24.16	24.21	6.28E-04	8/21/13 17:24
24.15	24.22	6.19E-04	8/21/13 17:25
24.17	24.22	6.08E-04	8/21/13 17:26
24.16	24.22	5.99E-04	8/21/13 17:27
24.15	24.22	5.89E-04	8/21/13 17:28
24.17	24.22	5.89E-04	8/21/13 17:29
24.16	24.21	5.79E-04	8/21/13 17:30
24.17	24.23	5.69E-04	8/21/13 17:31
24.17	24.24	5.59E-04	8/21/13 17:32
24.19	24.25	5.49E-04	8/21/13 17:33
24.18	24.25	5.49E-04	8/21/13 17:34
24.19	24.25	5.39E-04	8/21/13 17:35
24.2	24.26	5.29E-04	8/21/13 17:36
24.2	24.26	5.29E-04	8/21/13 17:37
24.2	24.26	5.19E-04	8/21/13 17:38
24.21	24.28	5.09E-04	8/21/13 17:39
24.22	24.27	5.09E-04	8/21/13 17:40
24.22	24.28	4.99E-04	8/21/13 17:41
24.22	24.28	4.99E-04	8/21/13 17:42
24.22	24.28	4.89E-04	8/21/13 17:43
24.24	24.3	4.89E-04	8/21/13 17:44
24.24	24.29	4.79E-04	8/21/13 17:45
24.24	24.3	4.79E-04	8/21/13 17:46
24.26	24.3	4.69E-04	8/21/13 17:47
24.25	24.31	4.59E-04	8/21/13 17:48
24.25	24.32	4.59E-04	8/21/13 17:49
24.26	24.31	4.59E-04	8/21/13 17:50
24.25	24.32	4.49E-04	8/21/13 17:51
24.26	24.32	4.39E-04	8/21/13 17:52
24.25	24.33	4.39E-04	8/21/13 17:53
24.27	24.33	4.39E-04	8/21/13 17:54
24.27	24.35	4.29E-04	8/21/13 17:55
24.28	24.34	4.29E-04	8/21/13 17:56
24.28	24.34	4.29E-04	8/21/13 17:57
24.28	24.36	4.29E-04	8/21/13 17:58
24.3	24.35	4.19E-04	8/21/13 17:59
24.29	24.37	4.09E-04	8/21/13 18:00
24.31	24.36	4.09E-04	8/21/13 18:01
24.32	24.37	3.99E-04	8/21/13 18:02
24.31	24.36	3.99E-04	8/21/13 18:03
24.34	24.39	3.99E-04	8/21/13 18:04
24.33	24.38	3.99E-04	8/21/13 18:05

24.32	24.39	3.89E-04	8/21/13 18:06
24.33	24.38	3.89E-04	8/21/13 18:07
24.35	24.39	3.89E-04	8/21/13 18:08
24.34	24.41	3.79E-04	8/21/13 18:09
24.36	24.41	3.79E-04	8/21/13 18:10
24.36	24.43	3.79E-04	8/21/13 18:11
24.36	24.41	3.69E-04	8/21/13 18:12
24.37	24.43	3.69E-04	8/21/13 18:13
24.37	24.43	3.59E-04	8/21/13 18:14
24.38	24.44	3.59E-04	8/21/13 18:15
24.38	24.42	3.59E-04	8/21/13 18:16
24.39	24.44	3.49E-04	8/21/13 18:17
24.39	24.45	3.49E-04	8/21/13 18:18
24.38	24.44	3.49E-04	8/21/13 18:19
24.4	24.45	3.39E-04	8/21/13 18:20
24.41	24.45	3.39E-04	8/21/13 18:21
24.41	24.45	3.39E-04	8/21/13 18:22
24.41	24.46	3.29E-04	8/21/13 18:23
24.41	24.46	3.29E-04	8/21/13 18:24
24.41	24.47	3.29E-04	8/21/13 18:25
24.43	24.48	3.29E-04	8/21/13 18:26
24.42	24.47	3.29E-04	8/21/13 18:27
24.43	24.49	3.29E-04	8/21/13 18:28
24.44	24.49	3.19E-04	8/21/13 18:29
24.44	24.49	3.09E-04	8/21/13 18:30
24.46	24.51	3.09E-04	8/21/13 18:31
24.44	24.49	3.17E-04	8/21/13 18:32
24.44	24.5	3.09E-04	8/21/13 18:33
24.47	24.52	3.09E-04	8/21/13 18:34
24.46	24.51	3.09E-04	8/21/13 18:35
24.46	24.5	2.99E-04	8/21/13 18:36
24.46	24.51	2.99E-04	8/21/13 18:37
24.45	24.52	2.99E-04	8/21/13 18:38
24.47	24.53	2.89E-04	8/21/13 18:39
24.48	24.53	2.99E-04	8/21/13 18:40
24.49	24.54	2.89E-04	8/21/13 18:41
24.48	24.53	2.89E-04	8/21/13 18:42
24.49	24.54	2.89E-04	8/21/13 18:43
24.49	24.54	2.89E-04	8/21/13 18:44
24.5	24.56	2.79E-04	8/21/13 18:45
24.49	24.55	2.79E-04	8/21/13 18:46
24.51	24.56	2.79E-04	8/21/13 18:47
24.51	24.55	2.79E-04	8/21/13 18:48
24.51	24.56	2.79E-04	8/21/13 18:49
24.51	24.57	2.79E-04	8/21/13 18:50
24.53	24.58	2.69E-04	8/21/13 18:51
24.52	24.57	2.69E-04	8/21/13 18:52
24.53	24.59	2.69E-04	8/21/13 18:53
24.55	24.59	2.69E-04	8/21/13 18:54
24.56	24.6	2.69E-04	8/21/13 18:55
24.54	24.59	2.69E-04	8/21/13 18:56
24.55	24.6	2.59E-04	8/21/13 18:57
24.55	24.61	2.59E-04	8/21/13 18:58

24.56	24.62	2.59E-04	8/21/13 18:59
24.58	24.63	2.59E-04	8/21/13 19:00
24.57	24.62	2.49E-04	8/21/13 19:01
24.57	24.62	2.49E-04	8/21/13 19:02
24.58	24.63	2.49E-04	8/21/13 19:03
24.58	24.62	2.49E-04	8/21/13 19:04
24.58	24.64	2.49E-04	8/21/13 19:05
24.6	24.64	2.49E-04	8/21/13 19:06
24.59	24.63	2.49E-04	8/21/13 19:07
24.59	24.63	2.49E-04	8/21/13 19:08
24.6	24.64	2.39E-04	8/21/13 19:09
24.61	24.64	2.39E-04	8/21/13 19:10
24.6	24.66	2.39E-04	8/21/13 19:11
24.61	24.67	2.39E-04	8/21/13 19:12
24.63	24.67	2.39E-04	8/21/13 19:13
24.63	24.67	2.39E-04	8/21/13 19:14
24.62	24.67	2.29E-04	8/21/13 19:15
24.62	24.68	2.29E-04	8/21/13 19:16
24.62	24.68	2.29E-04	8/21/13 19:17
24.65	24.67	2.29E-04	8/21/13 19:18
24.63	24.69	2.29E-04	8/21/13 19:19
24.64	24.69	2.29E-04	8/21/13 19:20
24.64	24.69	2.29E-04	8/21/13 19:21
24.64	24.69	2.19E-04	8/21/13 19:22
24.64	24.69	2.19E-04	8/21/13 19:23
24.66	24.71	2.19E-04	8/21/13 19:24
24.65	24.71	2.19E-04	8/21/13 19:25
24.66	24.72	2.19E-04	8/21/13 19:26
24.68	24.73	2.19E-04	8/21/13 19:27
24.67	24.72	2.19E-04	8/21/13 19:28
24.67	24.7	2.19E-04	8/21/13 19:29
24.68	24.73	2.19E-04	8/21/13 19:30
24.69	24.73	2.09E-04	8/21/13 19:31
24.68	24.73	2.09E-04	8/21/13 19:32
24.72	24.74	2.09E-04	8/21/13 19:33
24.69	24.75	2.09E-04	8/21/13 19:34
24.69	24.74	2.09E-04	8/21/13 19:35
24.71	24.74	2.09E-04	8/21/13 19:36
24.72	24.75	2.09E-04	8/21/13 19:37
24.72	24.77	2.09E-04	8/21/13 19:38
24.72	24.77	2.09E-04	8/21/13 19:39
24.73	24.77	1.99E-04	8/21/13 19:40
24.72	24.78	1.99E-04	8/21/13 19:41
24.74	24.77	1.99E-04	8/21/13 19:42
24.73	24.78	1.99E-04	8/21/13 19:43
24.73	24.77	1.99E-04	8/21/13 19:44
24.74	24.77	1.99E-04	8/21/13 19:45
24.75	24.77	1.99E-04	8/21/13 19:46
24.74	24.78	1.99E-04	8/21/13 19:47
24.75	24.78	1.99E-04	8/21/13 19:48
24.75	24.79	1.99E-04	8/21/13 19:49
24.75	24.79	1.89E-04	8/21/13 19:50
24.74	24.79	1.89E-04	8/21/13 19:51

24.77	24.81	1.89E-04	8/21/13 19:52
24.76	24.8	1.89E-04	8/21/13 19:53
24.76	24.8	1.89E-04	8/21/13 19:54
24.77	24.8	1.89E-04	8/21/13 19:55
24.77	24.8	1.89E-04	8/21/13 19:56
24.78	24.82	1.89E-04	8/21/13 19:57
24.77	24.82	1.89E-04	8/21/13 19:58
24.77	24.81	1.89E-04	8/21/13 19:59
24.79	24.83	1.89E-04	8/21/13 20:00
24.79	24.83	1.79E-04	8/21/13 20:01
24.79	24.83	1.79E-04	8/21/13 20:02
24.79	24.83	1.79E-04	8/21/13 20:03
24.79	24.83	1.79E-04	8/21/13 20:04
24.8	24.82	1.79E-04	8/21/13 20:05
24.8	24.82	1.79E-04	8/21/13 20:06
24.8	24.84	1.79E-04	8/21/13 20:07
24.8	24.83	1.79E-04	8/21/13 20:08
24.81	24.85	1.79E-04	8/21/13 20:09
24.8	24.85	1.79E-04	8/21/13 20:10
24.82	24.84	1.69E-04	8/21/13 20:11
24.81	24.85	1.69E-04	8/21/13 20:12
24.83	24.85	1.79E-04	8/21/13 20:13
24.82	24.85	1.69E-04	8/21/13 20:14
24.84	24.86	1.79E-04	8/21/13 20:15
24.83	24.88	1.69E-04	8/21/13 20:16
24.83	24.87	1.69E-04	8/21/13 20:17
24.82	24.87	1.69E-04	8/21/13 20:18
24.84	24.87	1.69E-04	8/21/13 20:19
24.83	24.87	1.69E-04	8/21/13 20:20
24.83	24.86	1.69E-04	8/21/13 20:21
24.84	24.87	1.69E-04	8/21/13 20:22
24.84	24.87	1.69E-04	8/21/13 20:23
24.85	24.88	1.69E-04	8/21/13 20:24
24.85	24.87	1.59E-04	8/21/13 20:25
24.84	24.87	1.59E-04	8/21/13 20:26
24.85	24.88	1.69E-04	8/21/13 20:27
24.84	24.87	1.69E-04	8/21/13 20:28
24.85	24.88	1.69E-04	8/21/13 20:29
24.84	24.86	1.59E-04	8/21/13 20:30
24.87	24.88	1.59E-04	8/21/13 20:31
24.84	24.88	1.59E-04	8/21/13 20:32
24.84	24.87	1.59E-04	8/21/13 20:33
24.85	24.88	1.59E-04	8/21/13 20:34
24.84	24.88	1.59E-04	8/21/13 20:35
24.85	24.87	1.59E-04	8/21/13 20:36
24.85	24.86	1.59E-04	8/21/13 20:37
24.86	24.89	1.59E-04	8/21/13 20:38
24.86	24.88	1.59E-04	8/21/13 20:39
24.86	24.89	1.59E-04	8/21/13 20:40
24.86	24.88	1.49E-04	8/21/13 20:41
24.86	24.88	1.59E-04	8/21/13 20:42
24.86	24.89	1.49E-04	8/21/13 20:43
24.86	24.87	1.49E-04	8/21/13 20:44

24.86	24.88	1.49E-04	8/21/13 20:45
24.85	24.88	1.49E-04	8/21/13 20:46
24.85	24.88	1.49E-04	8/21/13 20:47
24.87	24.89	1.49E-04	8/21/13 20:48
24.86	24.87	1.49E-04	8/21/13 20:49
24.86	24.88	1.49E-04	8/21/13 20:50
24.85	24.88	1.49E-04	8/21/13 20:51
24.87	24.89	1.49E-04	8/21/13 20:52
24.86	24.89	1.49E-04	8/21/13 20:53
24.86	24.88	1.49E-04	8/21/13 20:54
24.86	24.87	1.49E-04	8/21/13 20:55
24.85	24.89	1.39E-04	8/21/13 20:56
24.86	24.88	1.49E-04	8/21/13 20:57
24.86	24.89	1.39E-04	8/21/13 20:58
24.86	24.89	1.39E-04	8/21/13 20:59
24.86	24.88	1.39E-04	8/21/13 21:00
24.86	24.89	1.39E-04	8/21/13 21:01
24.86	24.89	1.39E-04	8/21/13 21:02
24.86	24.87	1.39E-04	8/21/13 21:03
24.85	24.88	1.39E-04	8/21/13 21:04
24.86	24.88	1.39E-04	8/21/13 21:05
24.87	24.88	1.39E-04	8/21/13 21:06
24.86	24.88	1.39E-04	8/21/13 21:07
24.86	24.87	1.39E-04	8/21/13 21:08
24.86	24.87	1.39E-04	8/21/13 21:09
24.86	24.87	1.39E-04	8/21/13 21:10
24.85	24.86	1.39E-04	8/21/13 21:11
24.86	24.88	1.39E-04	8/21/13 21:12
24.86	24.88	1.29E-04	8/21/13 21:13
24.86	24.89	1.39E-04	8/21/13 21:14
24.86	24.88	1.39E-04	8/21/13 21:15
24.87	24.88	1.29E-04	8/21/13 21:16
24.85	24.87	1.29E-04	8/21/13 21:17
24.87	24.88	1.29E-04	8/21/13 21:18
24.87	24.88	1.29E-04	8/21/13 21:19
24.85	24.87	1.29E-04	8/21/13 21:20
24.85	24.87	1.29E-04	8/21/13 21:21
24.86	24.87	1.39E-04	8/21/13 21:22
24.84	24.87	1.29E-04	8/21/13 21:23
24.84	24.86	1.29E-04	8/21/13 21:24
24.86	24.86	1.29E-04	8/21/13 21:25
24.85	24.85	1.29E-04	8/21/13 21:26
24.85	24.85	1.29E-04	8/21/13 21:27
24.85	24.87	1.29E-04	8/21/13 21:28
24.84	24.86	1.29E-04	8/21/13 21:29
24.83	24.86	1.29E-04	8/21/13 21:30
24.84	24.85	1.29E-04	8/21/13 21:31
24.85	24.87	1.29E-04	8/21/13 21:32
24.85	24.86	1.29E-04	8/21/13 21:33
24.84	24.84	1.19E-04	8/21/13 21:34
24.84	24.85	1.29E-04	8/21/13 21:35
24.85	24.85	1.19E-04	8/21/13 21:36
24.84	24.85	1.19E-04	8/21/13 21:37

24.84	24.85	1.19E-04	8/21/13 21:38
24.85	24.86	1.19E-04	8/21/13 21:39
24.83	24.85	1.29E-04	8/21/13 21:40
24.85	24.86	1.19E-04	8/21/13 21:41
24.84	24.85	1.19E-04	8/21/13 21:42
24.84	24.83	1.19E-04	8/21/13 21:43
24.82	24.84	1.19E-04	8/21/13 21:44
24.83	24.83	1.19E-04	8/21/13 21:45
24.83	24.84	1.19E-04	8/21/13 21:46
24.83	24.86	1.19E-04	8/21/13 21:47
24.82	24.85	1.19E-04	8/21/13 21:48
24.83	24.84	1.19E-04	8/21/13 21:49
24.84	24.84	1.19E-04	8/21/13 21:50
24.83	24.84	1.19E-04	8/21/13 21:51
24.83	24.83	1.19E-04	8/21/13 21:52
24.82	24.83	1.19E-04	8/21/13 21:53
24.83	24.83	1.19E-04	8/21/13 21:54
24.83	24.83	1.19E-04	8/21/13 21:55
24.83	24.84	1.19E-04	8/21/13 21:56
24.82	24.83	1.19E-04	8/21/13 21:57
24.83	24.83	1.19E-04	8/21/13 21:58
24.82	24.83	1.19E-04	8/21/13 21:59
24.82	24.83	1.19E-04	8/21/13 22:00
24.82	24.83	1.19E-04	8/21/13 22:01
24.83	24.82	1.19E-04	8/21/13 22:02
24.82	24.83	1.19E-04	8/21/13 22:03
24.81	24.81	1.09E-04	8/21/13 22:04
24.81	24.82	1.09E-04	8/21/13 22:05
24.81	24.82	1.09E-04	8/21/13 22:06
24.8	24.82	1.09E-04	8/21/13 22:07
24.82	24.83	1.09E-04	8/21/13 22:08
24.81	24.82	1.09E-04	8/21/13 22:09
24.8	24.81	1.09E-04	8/21/13 22:10
24.8	24.81	1.09E-04	8/21/13 22:11
24.8	24.8	1.09E-04	8/21/13 22:12
24.8	24.81	1.09E-04	8/21/13 22:13
24.8	24.81	1.09E-04	8/21/13 22:14
24.79	24.8	1.09E-04	8/21/13 22:15
24.8	24.8	1.09E-04	8/21/13 22:16
24.79	24.79	1.09E-04	8/21/13 22:17
24.79	24.79	1.09E-04	8/21/13 22:18
24.8	24.79	1.09E-04	8/21/13 22:19
24.79	24.79	1.09E-04	8/21/13 22:20
24.79	24.79	1.09E-04	8/21/13 22:21
24.78	24.8	1.09E-04	8/21/13 22:22
24.79	24.79	1.09E-04	8/21/13 22:23
24.77	24.79	1.09E-04	8/21/13 22:24
24.78	24.78	1.09E-04	8/21/13 22:25
24.77	24.77	1.09E-04	8/21/13 22:26
24.78	24.78	1.09E-04	8/21/13 22:27
24.77	24.8	1.09E-04	8/21/13 22:28
24.77	24.78	1.09E-04	8/21/13 22:29
24.77	24.77	9.90E-05	8/21/13 22:30

24.76	24.77	9.90E-05	8/21/13 22:31
24.76	24.77	9.88E-05	8/21/13 22:32
24.77	24.77	9.88E-05	8/21/13 22:33
24.77	24.76	9.89E-05	8/21/13 22:34
24.76	24.77	9.89E-05	8/21/13 22:35
24.74	24.75	9.88E-05	8/21/13 22:36
24.75	24.76	1.09E-04	8/21/13 22:37
24.74	24.75	9.88E-05	8/21/13 22:38
24.74	24.74	1.09E-04	8/21/13 22:39
24.74	24.74	9.87E-05	8/21/13 22:40
24.74	24.74	9.87E-05	8/21/13 22:41
24.74	24.75	9.89E-05	8/21/13 22:42
24.74	24.73	9.89E-05	8/21/13 22:43
24.72	24.73	9.90E-05	8/21/13 22:44
24.71	24.73	9.88E-05	8/21/13 22:45
24.72	24.72	9.89E-05	8/21/13 22:46
24.72	24.73	9.90E-05	8/21/13 22:47
24.72	24.72	9.88E-05	8/21/13 22:48
24.71	24.72	9.89E-05	8/21/13 22:49
24.71	24.73	9.87E-05	8/21/13 22:50
24.71	24.72	9.89E-05	8/21/13 22:51
24.71	24.71	9.90E-05	8/21/13 22:52
24.72	24.71	9.89E-05	8/21/13 22:53
24.7	24.71	9.88E-05	8/21/13 22:54
24.7	24.71	9.89E-05	8/21/13 22:55
24.69	24.71	9.87E-05	8/21/13 22:56
24.71	24.71	9.90E-05	8/21/13 22:57
24.69	24.71	9.90E-05	8/21/13 22:58
24.71	24.7	9.90E-05	8/21/13 22:59
24.69	24.68	8.90E-05	8/21/13 23:00
24.69	24.7	9.90E-05	8/21/13 23:01
24.68	24.69	9.90E-05	8/21/13 23:02
24.68	24.69	8.88E-05	8/21/13 23:03
24.69	24.69	9.87E-05	8/21/13 23:04
24.73	24.71	8.89E-05	8/21/13 23:05
24.79	24.76	8.88E-05	8/21/13 23:06
24.86	24.8	8.88E-05	8/21/13 23:07
24.94	24.89	8.88E-05	8/21/13 23:08
25.02	24.93	8.88E-05	8/21/13 23:09
25.11	24.98	9.88E-05	8/21/13 23:10
25.18	25.05	8.88E-05	8/21/13 23:11
25.27	25.12	8.87E-05	8/21/13 23:12
25.34	25.19	8.88E-05	8/21/13 23:13
25.48	25.3	9.89E-05	8/21/13 23:14
25.63	25.42	9.89E-05	8/21/13 23:15
25.79	25.52	8.88E-05	8/21/13 23:16
25.94	25.64	9.88E-05	8/21/13 23:17
26.09	25.78	9.88E-05	8/21/13 23:18
26.2	25.88	9.89E-05	8/21/13 23:19
26.33	25.98	8.88E-05	8/21/13 23:20
26.47	26.09	9.89E-05	8/21/13 23:21
26.59	26.21	8.87E-05	8/21/13 23:22
26.7	26.3	9.87E-05	8/21/13 23:23

26.81	26.4	9.87E-05	8/21/13 23:24
26.95	26.52	9.89E-05	8/21/13 23:25
27.14	26.68	9.90E-05	8/21/13 23:26
27.32	26.85	9.88E-05	8/21/13 23:27
27.52	27	9.87E-05	8/21/13 23:28
27.79	27.24	9.87E-05	8/21/13 23:29
28.11	27.5	9.87E-05	8/21/13 23:30
28.43	27.77	9.89E-05	8/21/13 23:31
28.73	28.02	9.87E-05	8/21/13 23:32
29.02	28.26	9.88E-05	8/21/13 23:33
29.29	28.49	9.88E-05	8/21/13 23:34
29.53	28.7	1.09E-04	8/21/13 23:35
29.73	28.88	1.09E-04	8/21/13 23:36
29.89	29.03	1.09E-04	8/21/13 23:37
30.06	29.21	1.09E-04	8/21/13 23:38
30.22	29.37	1.09E-04	8/21/13 23:39
30.39	29.54	1.09E-04	8/21/13 23:40
30.57	29.7	1.09E-04	8/21/13 23:41
30.73	29.86	1.09E-04	8/21/13 23:42
30.88	30.02	1.09E-04	8/21/13 23:43
31.07	30.2	1.19E-04	8/21/13 23:44
31.22	30.35	1.19E-04	8/21/13 23:45
31.37	30.51	1.19E-04	8/21/13 23:46
31.53	30.67	1.19E-04	8/21/13 23:47
31.69	30.82	1.19E-04	8/21/13 23:48
31.85	30.97	1.19E-04	8/21/13 23:49
32.03	31.14	1.19E-04	8/21/13 23:50
32.17	31.3	1.29E-04	8/21/13 23:51
32.32	31.44	1.19E-04	8/21/13 23:52
32.47	31.59	1.29E-04	8/21/13 23:53
32.61	31.73	1.29E-04	8/21/13 23:54
32.69	31.82	1.29E-04	8/21/13 23:55
32.68	31.85	1.29E-04	8/21/13 23:56
32.66	31.87	1.29E-04	8/21/13 23:57
32.65	31.91	1.29E-04	8/21/13 23:58
32.63	31.92	1.29E-04	8/21/13 23:59
32.64	31.96	1.29E-04	8/22/13 0:00
32.66	32	1.29E-04	8/22/13 0:01
32.69	32.04	1.29E-04	8/22/13 0:02
32.73	32.09	1.29E-04	8/22/13 0:03
32.77	32.14	1.29E-04	8/22/13 0:04
32.8	32.18	1.29E-04	8/22/13 0:05
32.88	32.26	1.29E-04	8/22/13 0:06
32.95	32.33	1.29E-04	8/22/13 0:07
33.01	32.4	1.29E-04	8/22/13 0:08
33.09	32.47	1.29E-04	8/22/13 0:09
33.16	32.54	1.29E-04	8/22/13 0:10
33.22	32.6	1.29E-04	8/22/13 0:11
33.29	32.67	1.29E-04	8/22/13 0:12
33.36	32.74	1.38E-04	8/22/13 0:13
33.44	32.81	1.29E-04	8/22/13 0:14
33.5	32.87	1.29E-04	8/22/13 0:15
33.56	32.93	1.39E-04	8/22/13 0:16

33.63	33	1.39E-04	8/22/13 0:17
33.69	33.05	1.29E-04	8/22/13 0:18
33.77	33.13	1.29E-04	8/22/13 0:19
33.83	33.18	1.39E-04	8/22/13 0:20
33.89	33.24	1.39E-04	8/22/13 0:21
33.97	33.32	1.39E-04	8/22/13 0:22
34.02	33.37	1.39E-04	8/22/13 0:23
34.09	33.44	1.39E-04	8/22/13 0:24
34.14	33.5	1.39E-04	8/22/13 0:25
34.21	33.56	1.39E-04	8/22/13 0:26
34.26	33.61	1.29E-04	8/22/13 0:27
34.32	33.69	1.39E-04	8/22/13 0:28
34.39	33.73	1.39E-04	8/22/13 0:29
34.46	33.79	1.39E-04	8/22/13 0:30
34.52	33.85	1.29E-04	8/22/13 0:31
34.57	33.91	1.39E-04	8/22/13 0:32
34.64	33.97	1.39E-04	8/22/13 0:33
34.69	34.02	1.39E-04	8/22/13 0:34
34.75	34.08	1.39E-04	8/22/13 0:35
34.81	34.14	1.39E-04	8/22/13 0:36
34.86	34.18	1.39E-04	8/22/13 0:37
34.92	34.25	1.39E-04	8/22/13 0:38
34.98	34.3	1.38E-04	8/22/13 0:39
35.03	34.34	1.39E-04	8/22/13 0:40
35.08	34.42	1.39E-04	8/22/13 0:41
35.14	34.47	1.38E-04	8/22/13 0:42
35.19	34.52	1.39E-04	8/22/13 0:43
35.25	34.56	1.38E-04	8/22/13 0:44
35.3	34.62	1.39E-04	8/22/13 0:45
35.36	34.67	1.39E-04	8/22/13 0:46
35.42	34.71	1.29E-04	8/22/13 0:47
35.48	34.77	1.39E-04	8/22/13 0:48
35.59	34.87	1.39E-04	8/22/13 0:49
35.69	34.96	1.39E-04	8/22/13 0:50
35.8	35.05	1.39E-04	8/22/13 0:51
35.92	35.15	1.39E-04	8/22/13 0:52
36.02	35.24	1.39E-04	8/22/13 0:53
36.11	35.32	1.39E-04	8/22/13 0:54
36.22	35.42	1.39E-04	8/22/13 0:55
36.3	35.5	1.39E-04	8/22/13 0:56
36.4	35.58	1.39E-04	8/22/13 0:57
36.49	35.67	1.39E-04	8/22/13 0:58
36.57	35.74	1.39E-04	8/22/13 0:59
36.65	35.82	1.39E-04	8/22/13 1:00
36.73	35.9	1.39E-04	8/22/13 1:01
36.82	35.98	1.39E-04	8/22/13 1:02
36.89	36.05	1.39E-04	8/22/13 1:03
36.99	36.13	1.49E-04	8/22/13 1:04
37.05	36.21	1.49E-04	8/22/13 1:05
37.16	36.3	1.39E-04	8/22/13 1:06
37.34	36.46	1.49E-04	8/22/13 1:07
37.5	36.6	1.49E-04	8/22/13 1:08
37.66	36.73	1.49E-04	8/22/13 1:09

37.8	36.85	1.49E-04	8/22/13 1:10
37.94	36.97	1.49E-04	8/22/13 1:11
38.08	37.08	1.49E-04	8/22/13 1:12
38.19	37.21	1.49E-04	8/22/13 1:13
38.31	37.32	1.49E-04	8/22/13 1:14
38.42	37.43	1.49E-04	8/22/13 1:15
38.53	37.52	1.49E-04	8/22/13 1:16
38.65	37.64	1.49E-04	8/22/13 1:17
38.76	37.74	1.49E-04	8/22/13 1:18
38.86	37.84	1.49E-04	8/22/13 1:19
38.96	37.95	1.49E-04	8/22/13 1:20
39.08	38.04	1.49E-04	8/22/13 1:21
39.19	38.14	1.49E-04	8/22/13 1:22
39.29	38.25	1.49E-04	8/22/13 1:23
39.4	38.34	1.49E-04	8/22/13 1:24
39.49	38.44	1.59E-04	8/22/13 1:25
39.58	38.55	1.58E-04	8/22/13 1:26
39.69	38.64	1.59E-04	8/22/13 1:27
39.79	38.74	1.59E-04	8/22/13 1:28
39.89	38.82	1.49E-04	8/22/13 1:29
39.98	38.93	1.59E-04	8/22/13 1:30
40.16	39.09	1.59E-04	8/22/13 1:31
40.43	39.31	1.59E-04	8/22/13 1:32
40.71	39.55	1.59E-04	8/22/13 1:33
40.97	39.76	1.59E-04	8/22/13 1:34
41.11	39.89	1.59E-04	8/22/13 1:35
41.2	39.98	1.59E-04	8/22/13 1:36
41.29	40.09	1.69E-04	8/22/13 1:37
41.39	40.18	1.58E-04	8/22/13 1:38
41.48	40.27	1.69E-04	8/22/13 1:39
41.56	40.37	1.69E-04	8/22/13 1:40
41.61	40.44	1.69E-04	8/22/13 1:41
41.67	40.5	1.59E-04	8/22/13 1:42
41.74	40.58	1.69E-04	8/22/13 1:43
41.79	40.67	1.68E-04	8/22/13 1:44
41.87	40.73	1.69E-04	8/22/13 1:45
41.93	40.8	1.69E-04	8/22/13 1:46
42.01	40.88	1.79E-04	8/22/13 1:47
42.09	40.96	1.79E-04	8/22/13 1:48
42.16	41.05	1.79E-04	8/22/13 1:49
42.24	41.13	1.79E-04	8/22/13 1:50
42.32	41.2	1.79E-04	8/22/13 1:51
42.39	41.28	1.79E-04	8/22/13 1:52
42.47	41.36	1.79E-04	8/22/13 1:53
42.54	41.43	1.69E-04	8/22/13 1:54
42.62	41.52	1.79E-04	8/22/13 1:55
42.69	41.58	1.69E-04	8/22/13 1:56
42.77	41.66	1.79E-04	8/22/13 1:57
42.84	41.73	1.79E-04	8/22/13 1:58
42.91	41.81	1.79E-04	8/22/13 1:59
42.98	41.87	1.79E-04	8/22/13 2:00
43.06	41.95	1.69E-04	8/22/13 2:01
43.13	42.02	1.79E-04	8/22/13 2:02

43.2	42.08	1.79E-04	8/22/13 2:03
43.28	42.16	1.79E-04	8/22/13 2:04
43.36	42.24	1.79E-04	8/22/13 2:05
43.42	42.31	1.69E-04	8/22/13 2:06
43.48	42.37	1.69E-04	8/22/13 2:07
43.57	42.44	1.79E-04	8/22/13 2:08
43.64	42.52	1.79E-04	8/22/13 2:09
43.7	42.58	1.78E-04	8/22/13 2:10
43.77	42.66	1.71E-04	8/22/13 2:11
43.86	42.72	1.69E-04	8/22/13 2:12
43.91	42.77	1.69E-04	8/22/13 2:13
43.99	42.86	1.79E-04	8/22/13 2:14
44.04	42.92	1.69E-04	8/22/13 2:15
44.13	42.99	1.79E-04	8/22/13 2:16
44.18	43.06	1.79E-04	8/22/13 2:17
44.25	43.12	1.69E-04	8/22/13 2:18
44.32	43.19	1.79E-04	8/22/13 2:19
44.38	43.27	1.69E-04	8/22/13 2:20
44.46	43.32	1.79E-04	8/22/13 2:21
44.51	43.38	1.79E-04	8/22/13 2:22
44.57	43.45	1.69E-04	8/22/13 2:23
44.64	43.51	1.79E-04	8/22/13 2:24
44.71	43.57	1.69E-04	8/22/13 2:25
44.78	43.63	1.69E-04	8/22/13 2:26
44.84	43.7	1.69E-04	8/22/13 2:27
44.91	43.76	1.69E-04	8/22/13 2:28
44.97	43.82	1.69E-04	8/22/13 2:29
45.03	43.88	1.69E-04	8/22/13 2:30
45.1	43.95	1.69E-04	8/22/13 2:31
45.29	44.1	1.69E-04	8/22/13 2:32
45.54	44.31	1.69E-04	8/22/13 2:33
45.78	44.52	1.69E-04	8/22/13 2:34
46.02	44.72	1.69E-04	8/22/13 2:35
46.25	44.9	1.69E-04	8/22/13 2:36
47.14	45.7	1.79E-04	8/22/13 2:37
48.23	46.57	1.79E-04	8/22/13 2:38
49.26	47.35	1.89E-04	8/22/13 2:39
50.19	48.09	1.99E-04	8/22/13 2:40
51	48.76	1.99E-04	8/22/13 2:41
51.74	49.38	2.09E-04	8/22/13 2:42
52.42	49.97	2.19E-04	8/22/13 2:43
52.77	50.27	2.19E-04	8/22/13 2:44
52.95	50.52	2.29E-04	8/22/13 2:45
53.15	50.77	2.29E-04	8/22/13 2:46
53.4	51.06	2.29E-04	8/22/13 2:47
54	51.67	2.39E-04	8/22/13 2:48
54.72	52.3	2.49E-04	8/22/13 2:49
55.41	52.91	2.49E-04	8/22/13 2:50
56.04	53.46	2.59E-04	8/22/13 2:51
56.63	54.01	2.69E-04	8/22/13 2:52
57.18	54.55	2.79E-04	8/22/13 2:53
57.72	55.05	2.79E-04	8/22/13 2:54
58.27	55.56	2.89E-04	8/22/13 2:55

58.78	56.05	2.98E-04	8/22/13 2:56
59.28	56.54	3.09E-04	8/22/13 2:57
59.43	56.67	3.09E-04	8/22/13 2:58
59.57	56.97	3.19E-04	8/22/13 2:59
60.09	57.49	3.19E-04	8/22/13 3:00
60.64	58.01	3.29E-04	8/22/13 3:01
61.17	58.51	3.29E-04	8/22/13 3:02
61.68	58.99	3.38E-04	8/22/13 3:03
62.18	59.45	3.39E-04	8/22/13 3:04
62.64	59.91	3.39E-04	8/22/13 3:05
63.1	60.37	3.49E-04	8/22/13 3:06
63.52	60.79	3.49E-04	8/22/13 3:07
64.11	61.32	3.59E-04	8/22/13 3:08
64.58	61.79	3.59E-04	8/22/13 3:09
65.06	62.25	3.59E-04	8/22/13 3:10
65.53	62.69	3.69E-04	8/22/13 3:11
65.84	63.03	3.69E-04	8/22/13 3:12
66.11	63.3	3.69E-04	8/22/13 3:13
66.33	63.57	3.69E-04	8/22/13 3:14
65.98	63.34	3.69E-04	8/22/13 3:15
65.56	63.1	3.59E-04	8/22/13 3:16
65.21	62.93	3.49E-04	8/22/13 3:17
64.94	62.81	3.49E-04	8/22/13 3:18
64.74	62.71	3.38E-04	8/22/13 3:19
64.6	62.63	3.39E-04	8/22/13 3:20
64.47	62.58	3.29E-04	8/22/13 3:21
64.38	62.55	3.19E-04	8/22/13 3:22
64.31	62.52	3.19E-04	8/22/13 3:23
64.25	62.51	3.09E-04	8/22/13 3:24
64.27	62.53	3.09E-04	8/22/13 3:25
64.29	62.56	2.99E-04	8/22/13 3:26
64.31	62.58	2.99E-04	8/22/13 3:27
64.32	62.6	2.99E-04	8/22/13 3:28
64.35	62.63	2.89E-04	8/22/13 3:29
64.36	62.64	2.89E-04	8/22/13 3:30
64.39	62.66	2.79E-04	8/22/13 3:31
64.41	62.68	2.79E-04	8/22/13 3:32
64.41	62.71	2.69E-04	8/22/13 3:33
64.45	62.72	2.69E-04	8/22/13 3:34
64.47	62.76	2.69E-04	8/22/13 3:35
64.5	62.77	2.59E-04	8/22/13 3:36
64.52	62.8	2.69E-04	8/22/13 3:37
64.54	62.82	2.59E-04	8/22/13 3:38
64.56	62.84	2.58E-04	8/22/13 3:39
64.59	62.85	2.59E-04	8/22/13 3:40
64.62	62.87	2.49E-04	8/22/13 3:41
64.64	62.89	2.49E-04	8/22/13 3:42
64.67	62.93	2.49E-04	8/22/13 3:43
64.68	62.95	2.39E-04	8/22/13 3:44
64.71	62.98	2.38E-04	8/22/13 3:45
64.74	63	2.39E-04	8/22/13 3:46
64.77	63.03	2.39E-04	8/22/13 3:47
64.8	63.05	2.39E-04	8/22/13 3:48

64.83	63.07	2.29E-04	8/22/13 3:49
64.87	63.1	2.29E-04	8/22/13 3:50
64.88	63.13	2.29E-04	8/22/13 3:51
64.92	63.16	2.29E-04	8/22/13 3:52
64.95	63.18	2.29E-04	8/22/13 3:53
64.99	63.21	2.19E-04	8/22/13 3:54
65.02	63.23	2.19E-04	8/22/13 3:55
65.03	63.25	2.19E-04	8/22/13 3:56
65.04	63.26	2.19E-04	8/22/13 3:57
65.05	63.27	2.19E-04	8/22/13 3:58
65.06	63.28	2.18E-04	8/22/13 3:59
65.06	63.3	2.09E-04	8/22/13 4:00
65.07	63.31	2.09E-04	8/22/13 4:01
65.07	63.32	2.09E-04	8/22/13 4:02
65.1	63.32	2.09E-04	8/22/13 4:03
65.12	63.34	2.09E-04	8/22/13 4:04
65.13	63.34	2.09E-04	8/22/13 4:05
65.15	63.36	1.99E-04	8/22/13 4:06
65.15	63.38	1.99E-04	8/22/13 4:07
65.15	63.38	1.99E-04	8/22/13 4:08
65.18	63.4	1.99E-04	8/22/13 4:09
65.2	63.42	1.99E-04	8/22/13 4:10
65.19	63.4	1.99E-04	8/22/13 4:11
65.14	63.39	1.99E-04	8/22/13 4:12
65.11	63.35	1.89E-04	8/22/13 4:13
65.07	63.33	1.89E-04	8/22/13 4:14
65.05	63.3	1.89E-04	8/22/13 4:15
65.02	63.29	1.79E-04	8/22/13 4:16
64.99	63.26	1.89E-04	8/22/13 4:17
64.98	63.24	1.78E-04	8/22/13 4:18
64.94	63.21	1.89E-04	8/22/13 4:19
64.93	63.21	1.79E-04	8/22/13 4:20
64.9	63.18	1.78E-04	8/22/13 4:21
64.9	63.17	1.79E-04	8/22/13 4:22
64.87	63.14	1.79E-04	8/22/13 4:23
64.85	63.12	1.79E-04	8/22/13 4:24
64.85	63.11	1.79E-04	8/22/13 4:25
64.83	63.1	1.79E-04	8/22/13 4:26
64.82	63.07	1.68E-04	8/22/13 4:27
64.81	63.06	1.69E-04	8/22/13 4:28
64.8	63.04	1.69E-04	8/22/13 4:29
64.79	63.03	1.69E-04	8/22/13 4:30
64.78	63.01	1.69E-04	8/22/13 4:31
64.76	63.01	1.69E-04	8/22/13 4:32
64.75	63	1.69E-04	8/22/13 4:33
64.74	62.99	1.69E-04	8/22/13 4:34
64.73	62.95	1.69E-04	8/22/13 4:35
64.72	62.96	1.59E-04	8/22/13 4:36
64.72	62.95	1.59E-04	8/22/13 4:37
64.71	62.94	1.59E-04	8/22/13 4:38
64.69	62.92	1.58E-04	8/22/13 4:39
64.69	62.91	1.58E-04	8/22/13 4:40
64.67	62.9	1.59E-04	8/22/13 4:41

64.67	62.9	1.59E-04	8/22/13 4:42
64.67	62.89	1.59E-04	8/22/13 4:43
64.66	62.87	1.49E-04	8/22/13 4:44
64.65	62.87	1.59E-04	8/22/13 4:45
64.64	62.86	1.48E-04	8/22/13 4:46
64.62	62.84	1.49E-04	8/22/13 4:47
64.63	62.83	1.48E-04	8/22/13 4:48
64.62	62.82	1.48E-04	8/22/13 4:49
64.6	62.82	1.49E-04	8/22/13 4:50
64.6	62.8	1.49E-04	8/22/13 4:51
64.58	62.79	1.49E-04	8/22/13 4:52
64.58	62.79	1.49E-04	8/22/13 4:53
64.6	62.81	1.39E-04	8/22/13 4:54
64.57	62.78	1.49E-04	8/22/13 4:55
64.56	62.76	1.49E-04	8/22/13 4:56
64.57	62.77	1.39E-04	8/22/13 4:57
64.53	62.74	1.49E-04	8/22/13 4:58
64.55	62.73	1.39E-04	8/22/13 4:59
64.54	62.73	1.38E-04	8/22/13 5:00
64.54	62.71	1.39E-04	8/22/13 5:01
64.52	62.71	1.39E-04	8/22/13 5:02
64.52	62.7	1.39E-04	8/22/13 5:03
64.51	62.69	1.39E-04	8/22/13 5:04
64.51	62.67	1.39E-04	8/22/13 5:05
64.5	62.67	1.39E-04	8/22/13 5:06
64.5	62.67	1.39E-04	8/22/13 5:07
64.48	62.65	1.39E-04	8/22/13 5:08
64.48	62.65	1.39E-04	8/22/13 5:09
64.46	62.62	1.39E-04	8/22/13 5:10
64.46	62.62	1.29E-04	8/22/13 5:11
64.45	62.63	1.39E-04	8/22/13 5:12
64.44	62.6	1.29E-04	8/22/13 5:13
64.44	62.6	1.29E-04	8/22/13 5:14
64.43	62.59	1.29E-04	8/22/13 5:15
64.43	62.59	1.29E-04	8/22/13 5:16
64.41	62.58	1.29E-04	8/22/13 5:17
64.4	62.56	1.28E-04	8/22/13 5:18
64.41	62.57	1.29E-04	8/22/13 5:19
64.4	62.55	1.29E-04	8/22/13 5:20
64.39	62.55	1.29E-04	8/22/13 5:21
64.4	62.54	1.29E-04	8/22/13 5:22
64.39	62.53	1.28E-04	8/22/13 5:23
64.37	62.53	1.29E-04	8/22/13 5:24
64.37	62.52	1.29E-04	8/22/13 5:25
64.35	62.51	1.29E-04	8/22/13 5:26
64.35	62.52	1.19E-04	8/22/13 5:27
64.35	62.5	1.19E-04	8/22/13 5:28
64.35	62.51	1.19E-04	8/22/13 5:29
64.34	62.49	1.19E-04	8/22/13 5:30
64.32	62.48	1.19E-04	8/22/13 5:31
64.32	62.47	1.19E-04	8/22/13 5:32
64.33	62.47	1.19E-04	8/22/13 5:33
64.31	62.46	1.19E-04	8/22/13 5:34

64.31	62.44	1.19E-04	8/22/13 5:35
64.3	62.45	1.19E-04	8/22/13 5:36
64.29	62.44	1.19E-04	8/22/13 5:37
64.28	62.42	1.19E-04	8/22/13 5:38
64.28	62.43	1.19E-04	8/22/13 5:39
64.28	62.43	1.19E-04	8/22/13 5:40
64.27	62.42	1.18E-04	8/22/13 5:41
64.28	62.42	1.19E-04	8/22/13 5:42
64.28	62.42	1.09E-04	8/22/13 5:43
64.28	62.41	1.19E-04	8/22/13 5:44
64.26	62.4	1.19E-04	8/22/13 5:45
64.26	62.4	1.19E-04	8/22/13 5:46
64.27	62.4	1.19E-04	8/22/13 5:47
64.25	62.4	1.09E-04	8/22/13 5:48
64.26	62.39	1.09E-04	8/22/13 5:49
64.26	62.39	1.09E-04	8/22/13 5:50
64.25	62.39	1.08E-04	8/22/13 5:51
64.25	62.38	1.09E-04	8/22/13 5:52
64.25	62.38	1.09E-04	8/22/13 5:53
64.25	62.37	1.09E-04	8/22/13 5:54
64.24	62.39	1.09E-04	8/22/13 5:55
64.25	62.38	1.08E-04	8/22/13 5:56
64.24	62.38	1.09E-04	8/22/13 5:57
64.24	62.37	1.08E-04	8/22/13 5:58
64.23	62.36	1.09E-04	8/22/13 5:59
64.24	62.38	1.09E-04	8/22/13 6:00
64.24	62.35	1.09E-04	8/22/13 6:01
64.23	62.35	1.09E-04	8/22/13 6:02
64.21	62.35	1.09E-04	8/22/13 6:03
64.21	62.34	9.85E-05	8/22/13 6:04
64.21	62.33	9.84E-05	8/22/13 6:05
64.23	62.33	1.09E-04	8/22/13 6:06
64.21	62.34	1.09E-04	8/22/13 6:07
64.21	62.33	1.09E-04	8/22/13 6:08
64.22	62.33	1.09E-04	8/22/13 6:09
64.22	62.32	9.86E-05	8/22/13 6:10
64.22	62.34	1.09E-04	8/22/13 6:11
64.21	62.34	1.09E-04	8/22/13 6:12
64.22	62.32	9.85E-05	8/22/13 6:13
64.21	62.33	9.85E-05	8/22/13 6:14
64.22	62.33	1.09E-04	8/22/13 6:15
64.2	62.31	9.86E-05	8/22/13 6:16
64.2	62.32	9.84E-05	8/22/13 6:17
64.2	62.33	9.86E-05	8/22/13 6:18
64.21	62.33	9.84E-05	8/22/13 6:19
64.2	62.32	9.86E-05	8/22/13 6:20
64.2	62.32	9.84E-05	8/22/13 6:21
64.2	62.3	9.84E-05	8/22/13 6:22
64.19	62.31	9.85E-05	8/22/13 6:23
64.2	62.3	9.84E-05	8/22/13 6:24
64.2	62.32	9.85E-05	8/22/13 6:25
64.2	62.3	9.85E-05	8/22/13 6:26
64.2	62.31	9.86E-05	8/22/13 6:27

64.19	62.31	9.86E-05	8/22/13 6:28
64.19	62.3	9.85E-05	8/22/13 6:29
64.2	62.31	9.84E-05	8/22/13 6:30
64.2	62.32	9.86E-05	8/22/13 6:31
64.2	62.33	9.85E-05	8/22/13 6:32
64.21	62.31	9.87E-05	8/22/13 6:33
64.22	62.32	9.86E-05	8/22/13 6:34
64.22	62.32	9.86E-05	8/22/13 6:35
64.23	62.32	9.85E-05	8/22/13 6:36
64.23	62.31	9.86E-05	8/22/13 6:37
64.22	62.32	9.84E-05	8/22/13 6:38
64.23	62.33	8.86E-05	8/22/13 6:39
64.22	62.33	9.84E-05	8/22/13 6:40
64.22	62.31	9.87E-05	8/22/13 6:41
64.22	62.32	9.85E-05	8/22/13 6:42
64.22	62.32	8.85E-05	8/22/13 6:43
64.21	62.32	8.87E-05	8/22/13 6:44
64.21	62.31	8.85E-05	8/22/13 6:45
64.23	62.33	9.85E-05	8/22/13 6:46
64.21	62.32	8.85E-05	8/22/13 6:47
64.23	62.32	8.85E-05	8/22/13 6:48
64.22	62.31	8.87E-05	8/22/13 6:49
64.22	62.32	8.86E-05	8/22/13 6:50
64.2	62.31	9.87E-05	8/22/13 6:51
64.21	62.31	8.86E-05	8/22/13 6:52
64.2	62.3	9.84E-05	8/22/13 6:53
64.2	62.31	8.86E-05	8/22/13 6:54
64.21	62.3	8.86E-05	8/22/13 6:55
64.19	62.3	8.86E-05	8/22/13 6:56
64.2	62.3	8.85E-05	8/22/13 6:57
64.19	62.29	8.86E-05	8/22/13 6:58
64.18	62.29	8.85E-05	8/22/13 6:59
64.19	62.29	8.87E-05	8/22/13 7:00
64.18	62.29	8.85E-05	8/22/13 7:01
64.19	62.29	8.86E-05	8/22/13 7:02
64.19	62.28	8.83E-05	8/22/13 7:03
64.17	62.28	8.84E-05	8/22/13 7:04
64.17	62.27	8.86E-05	8/22/13 7:05
64.17	62.28	8.86E-05	8/22/13 7:06
64.17	62.26	8.85E-05	8/22/13 7:07
64.16	62.28	8.86E-05	8/22/13 7:08
64.16	62.26	8.86E-05	8/22/13 7:09
64.17	62.27	8.84E-05	8/22/13 7:10
64.17	62.26	8.87E-05	8/22/13 7:11
64.14	62.25	8.85E-05	8/22/13 7:12
64.16	62.25	8.86E-05	8/22/13 7:13
64.15	62.25	8.84E-05	8/22/13 7:14
64.16	62.25	8.85E-05	8/22/13 7:15
64.14	62.24	8.85E-05	8/22/13 7:16
64.14	62.24	8.85E-05	8/22/13 7:17
64.15	62.24	1.10E-04	8/22/13 7:18
64.14	62.24	8.86E-05	8/22/13 7:19
64.13	62.24	8.86E-05	8/22/13 7:20

64.13	62.24	8.86E-05	8/22/13 7:21
64.13	62.25	8.85E-05	8/22/13 7:22
64.11	62.22	1.10E-04	8/22/13 7:23
64.12	62.22	1.10E-04	8/22/13 7:24
64.12	62.22	8.87E-05	8/22/13 7:25
64.11	62.22	1.10E-04	8/22/13 7:26
64.11	62.22	1.10E-04	8/22/13 7:27
64.11	62.21	8.86E-05	8/22/13 7:28
64.12	62.22	1.10E-04	8/22/13 7:29
64.11	62.2	1.10E-04	8/22/13 7:30
64.1	62.21	1.10E-04	8/22/13 7:31
64.11	62.2	1.10E-04	8/22/13 7:32
64.11	62.2	1.10E-04	8/22/13 7:33
64.09	62.2	1.10E-04	8/22/13 7:34
64.1	62.19	1.10E-04	8/22/13 7:35
64.09	62.2	1.10E-04	8/22/13 7:36
64.09	62.2	1.10E-04	8/22/13 7:37
64.1	62.19	1.10E-04	8/22/13 7:38
64.09	62.19	1.10E-04	8/22/13 7:39
64.09	62.2	1.10E-04	8/22/13 7:40
64.08	62.19	1.10E-04	8/22/13 7:41
64.08	62.18	1.10E-04	8/22/13 7:42
64.09	62.19	1.10E-04	8/22/13 7:43
64.08	62.19	1.10E-04	8/22/13 7:44
64.08	62.18	1.10E-04	8/22/13 7:45
64.09	62.18	1.10E-04	8/22/13 7:46
64.08	62.18	1.10E-04	8/22/13 7:47
64.09	62.19	1.10E-04	8/22/13 7:48
64.1	62.2	1.10E-04	8/22/13 7:49
64.09	62.18	1.10E-04	8/22/13 7:50
64.1	62.18	1.10E-04	8/22/13 7:51
64.12	62.2	1.10E-04	8/22/13 7:57
64.1	62.19	1.10E-04	8/22/13 7:59
64.1	62.18	1.10E-04	8/22/13 8:00
64.1	62.18	1.10E-04	8/22/13 8:01
64.1	62.19	1.10E-04	8/22/13 8:02
64.09	62.19	1.10E-04	8/22/13 8:03
64.09	62.19	1.10E-04	8/22/13 8:04
64.1	62.19	1.09E-04	8/22/13 8:05
64.1	62.18	1.10E-04	8/22/13 8:06
64.1	62.18	1.09E-04	8/22/13 8:07
64.1	62.18	1.09E-04	8/22/13 8:08
64.09	62.18	1.10E-04	8/22/13 8:09
64.1	62.18	1.09E-04	8/22/13 8:10
64.09	62.17	1.10E-04	8/22/13 8:11
64.09	62.19	1.09E-04	8/22/13 8:12
64.09	62.19	1.09E-04	8/22/13 8:13
64.08	62.16	1.09E-04	8/22/13 8:14
64.09	62.17	1.10E-04	8/22/13 8:15
64.08	62.18	1.10E-04	8/22/13 8:16
64.07	62.17	1.09E-04	8/22/13 8:17
64.09	62.19	1.09E-04	8/22/13 8:18
64.07	62.17	1.09E-04	8/22/13 8:19

64.09	62.18	1.09E-04	8/22/13 8:20
64.07	62.16	1.09E-04	8/22/13 8:21
64.08	62.16	1.09E-04	8/22/13 8:22
64.08	62.16	1.10E-04	8/22/13 8:23
64.07	62.16	1.09E-04	8/22/13 8:24
64.07	62.17	1.09E-04	8/22/13 8:25
64.07	62.15	1.09E-04	8/22/13 8:26
64.07	62.15	1.09E-04	8/22/13 8:27
64.07	62.16	1.09E-04	8/22/13 8:28
64.07	62.16	1.09E-04	8/22/13 8:29
64.07	62.16	1.09E-04	8/22/13 8:30
64.05	62.14	1.09E-04	8/22/13 8:31
64.05	62.16	1.09E-04	8/22/13 8:32
64.06	62.15	1.09E-04	8/22/13 8:33
64.06	62.15	1.09E-04	8/22/13 8:34
64.07	62.16	1.09E-04	8/22/13 8:35
64.07	62.15	1.09E-04	8/22/13 8:36
64.06	62.15	1.09E-04	8/22/13 8:37
64.07	62.15	1.09E-04	8/22/13 8:38
64.06	62.15	1.09E-04	8/22/13 8:39
64.07	62.15	1.09E-04	8/22/13 8:40
64.06	62.15	1.09E-04	8/22/13 8:41
64.07	62.15	1.09E-04	8/22/13 8:42
64.07	62.16	1.09E-04	8/22/13 8:43
64.06	62.16	1.09E-04	8/22/13 8:44
64.05	62.15	1.09E-04	8/22/13 8:45
64.06	62.14	1.09E-04	8/22/13 8:46
64.07	62.15	1.09E-04	8/22/13 8:47
64.06	62.15	1.09E-04	8/22/13 8:48
64.06	62.15	1.09E-04	8/22/13 8:49
64.07	62.16	7.50E-05	8/22/13 8:50
64.06	62.15	7.39E-05	8/22/13 8:51
64.05	62.14	7.30E-05	8/22/13 8:52
64.06	62.14	7.29E-05	8/22/13 8:53
64.06	62.14	7.30E-05	8/22/13 8:54
64.05	62.15	7.19E-05	8/22/13 8:55
64.05	62.14	7.19E-05	8/22/13 8:56
64.05	62.13	7.20E-05	8/22/13 8:57
64.05	62.15	7.09E-05	8/22/13 8:58
64.06	62.15	7.10E-05	8/22/13 8:59
64.05	62.15	7.10E-05	8/22/13 9:00
64.05	62.15	7.10E-05	8/22/13 9:01
64.05	62.14	7.00E-05	8/22/13 9:02
64.05	62.15	7.00E-05	8/22/13 9:03
64.06	62.13	7.00E-05	8/22/13 9:04
64.06	62.14	6.99E-05	8/22/13 9:05
64.06	62.15	7.00E-05	8/22/13 9:06
64.04	62.15	6.99E-05	8/22/13 9:07
64.05	62.14	6.90E-05	8/22/13 9:08
64.05	62.13	6.90E-05	8/22/13 9:09
64.06	62.14	6.90E-05	8/22/13 9:10
64.07	62.16	6.90E-05	8/22/13 9:11
64.05	62.15	6.90E-05	8/22/13 9:12

64.06	62.15	6.80E-05	8/22/13 9:13
64.05	62.14	6.80E-05	8/22/13 9:14
64.05	62.13	6.80E-05	8/22/13 9:15
64.06	62.14	6.80E-05	8/22/13 9:16
64.05	62.14	6.80E-05	8/22/13 9:17
64.06	62.14	6.70E-05	8/22/13 9:18
64.06	62.13	6.70E-05	8/22/13 9:19
64.06	62.14	6.70E-05	8/22/13 9:20
64.05	62.15	6.70E-05	8/22/13 9:21
64.05	62.13	6.70E-05	8/22/13 9:22
64.05	62.14	6.60E-05	8/22/13 9:23
64.06	62.14	6.69E-05	8/22/13 9:24
64.06	62.14	6.70E-05	8/22/13 9:25
64.06	62.14	6.59E-05	8/22/13 9:26
64.07	62.15	6.60E-05	8/22/13 9:27
64.05	62.13	6.60E-05	8/22/13 9:28
64.06	62.15	6.60E-05	8/22/13 9:29
64.05	62.13	6.60E-05	8/22/13 9:30
64.05	62.13	6.60E-05	8/22/13 9:31
64.05	62.13	6.60E-05	8/22/13 9:32
64.05	62.15	6.57E-05	8/22/13 9:33
64.06	62.15	6.50E-05	8/22/13 9:34
64.04	62.13	6.50E-05	8/22/13 9:35
64.06	62.13	6.50E-05	8/22/13 9:36
64.05	62.13	6.49E-05	8/22/13 9:37
64.05	62.14	6.50E-05	8/22/13 9:38
64.05	62.13	6.50E-05	8/22/13 9:39
64.04	62.12	6.50E-05	8/22/13 9:40
64.04	62.13	6.40E-05	8/22/13 9:41
63.94	62.04	6.40E-05	8/22/13 9:42
63.67	61.82	6.40E-05	8/22/13 9:43
63.32	61.54	6.40E-05	8/22/13 9:44
62.89	61.22	6.30E-05	8/22/13 9:45
62.45	60.89	6.30E-05	8/22/13 9:46
62.06	60.59	6.20E-05	8/22/13 9:47
61.69	60.28	6.20E-05	8/22/13 9:48
61.34	59.99	6.10E-05	8/22/13 9:49
61.03	59.73	6.10E-05	8/22/13 9:50
60.7	59.44	6.00E-05	8/22/13 9:51
60.42	59.19	5.90E-05	8/22/13 9:52
60.12	58.95	5.90E-05	8/22/13 9:53
59.86	58.68	5.80E-05	8/22/13 9:54
59.59	58.43	5.70E-05	8/22/13 9:55
59.31	58.19	5.69E-05	8/22/13 9:56
59.06	57.95	5.60E-05	8/22/13 9:57
58.82	57.71	5.60E-05	8/22/13 9:58
58.57	57.49	5.50E-05	8/22/13 9:59
58.32	57.23	5.40E-05	8/22/13 10:00
58.1	57.02	5.40E-05	8/22/13 10:01
57.86	56.79	5.30E-05	8/22/13 10:02
57.61	56.56	5.30E-05	8/22/13 10:03
57.4	56.36	5.20E-05	8/22/13 10:04
57.19	56.12	5.20E-05	8/22/13 10:05

56.96	55.89	5.10E-05	8/22/13 10:06
56.76	55.69	5.10E-05	8/22/13 10:07
56.54	55.48	5.00E-05	8/22/13 10:08
56.35	55.27	5.00E-05	8/22/13 10:09
56.12	55.07	5.00E-05	8/22/13 10:10
55.92	54.87	4.90E-05	8/22/13 10:11
55.72	54.67	4.80E-05	8/22/13 10:12
55.51	54.46	4.80E-05	8/22/13 10:13
55.3	54.25	4.70E-05	8/22/13 10:14
55.11	54.05	4.70E-05	8/22/13 10:15
54.93	53.85	4.70E-05	8/22/13 10:16
54.73	53.67	4.60E-05	8/22/13 10:17
54.53	53.47	4.50E-05	8/22/13 10:18
54.36	53.27	4.50E-05	8/22/13 10:19
54.17	53.09	4.50E-05	8/22/13 10:20
53.98	52.91	4.40E-05	8/22/13 10:21
53.8	52.72	4.40E-05	8/22/13 10:22
53.61	52.53	4.30E-05	8/22/13 10:23
53.43	52.34	4.30E-05	8/22/13 10:24
53.24	52.15	4.30E-05	8/22/13 10:25
53.05	51.98	4.20E-05	8/22/13 10:26
52.88	51.79	4.20E-05	8/22/13 10:27
52.7	51.61	4.10E-05	8/22/13 10:28
52.54	51.44	4.10E-05	8/22/13 10:29
52.35	51.26	4.10E-05	8/22/13 10:30
52.19	51.1	4.00E-05	8/22/13 10:31
52.01	50.93	4.00E-05	8/22/13 10:32
51.84	50.75	4.00E-05	8/22/13 10:33
51.66	50.59	3.90E-05	8/22/13 10:34
51.5	50.42	3.90E-05	8/22/13 10:35
51.33	50.25	3.80E-05	8/22/13 10:36
51.15	50.07	3.80E-05	8/22/13 10:37
50.98	49.91	3.80E-05	8/22/13 10:38
50.82	49.74	3.70E-05	8/22/13 10:39
50.66	49.57	3.70E-05	8/22/13 10:40
50.49	49.43	3.70E-05	8/22/13 10:41
50.34	49.27	3.60E-05	8/22/13 10:42
50.18	49.11	3.60E-05	8/22/13 10:43
50.02	48.95	3.60E-05	8/22/13 10:44
49.87	48.8	3.60E-05	8/22/13 10:45
49.7	48.65	3.50E-05	8/22/13 10:46
49.56	48.49	3.50E-05	8/22/13 10:47
49.4	48.34	3.40E-05	8/22/13 10:48
49.25	48.19	3.40E-05	8/22/13 10:49
49.1	48.04	3.40E-05	8/22/13 10:50
48.95	47.9	3.40E-05	8/22/13 10:51
48.79	47.75	3.30E-05	8/22/13 10:52
48.64	47.6	3.30E-05	8/22/13 10:53
48.49	47.44	3.30E-05	8/22/13 10:54
48.35	47.31	3.20E-05	8/22/13 10:55
48.21	47.17	3.20E-05	8/22/13 10:56
48.06	47.02	3.20E-05	8/22/13 10:57
47.92	46.86	3.20E-05	8/22/13 10:58

47.77	46.74	3.20E-05	8/22/13 10:59
47.63	46.59	3.10E-05	8/22/13 11:00
47.49	46.44	3.10E-05	8/22/13 11:01
47.34	46.3	3.10E-05	8/22/13 11:02
47.21	46.18	3.10E-05	8/22/13 11:03
47.07	46.04	3.10E-05	8/22/13 11:04
46.93	45.9	3.10E-05	8/22/13 11:05
46.8	45.78	3.00E-05	8/22/13 11:06
46.66	45.62	3.00E-05	8/22/13 11:07
46.52	45.5	3.00E-05	8/22/13 11:08
46.39	45.37	3.00E-05	8/22/13 11:09
46.26	45.25	3.00E-05	8/22/13 11:10
46.14	45.12	3.00E-05	8/22/13 11:11
45.99	44.99	2.90E-05	8/22/13 11:12
45.85	44.85	2.90E-05	8/22/13 11:13
45.74	44.74	2.90E-05	8/22/13 11:14
45.61	44.61	2.80E-05	8/22/13 11:15
45.48	44.48	2.90E-05	8/22/13 11:16
45.36	44.36	2.80E-05	8/22/13 11:17
45.22	44.23	2.80E-05	8/22/13 11:18
45.11	44.12	2.80E-05	8/22/13 11:19
45	44	2.80E-05	8/22/13 11:20
44.86	43.87	2.80E-05	8/22/13 11:21
44.74	43.78	2.70E-05	8/22/13 11:22
44.61	43.65	2.70E-05	8/22/13 11:23
44.5	43.53	2.70E-05	8/22/13 11:24
44.38	43.42	2.70E-05	8/22/13 11:25
44.27	43.3	2.60E-05	8/22/13 11:26
44.14	43.19	2.60E-05	8/22/13 11:27
44.03	43.07	2.60E-05	8/22/13 11:28
43.92	42.96	2.60E-05	8/22/13 11:29
43.8	42.85	2.50E-05	8/22/13 11:30
43.68	42.73	2.50E-05	8/22/13 11:31
43.57	42.62	2.50E-05	8/22/13 11:32
43.45	42.51	2.40E-05	8/22/13 11:33
43.35	42.41	2.50E-05	8/22/13 11:34
43.22	42.29	2.40E-05	8/22/13 11:35
43.12	42.18	2.40E-05	8/22/13 11:36
43.01	42.09	2.40E-05	8/22/13 11:37
42.9	41.97	2.40E-05	8/22/13 11:38
42.8	41.86	2.40E-05	8/22/13 11:39
42.71	41.75	2.40E-05	8/22/13 11:40
42.63	41.65	2.40E-05	8/22/13 11:41
42.52	41.56	2.40E-05	8/22/13 11:42
42.42	41.45	1.47E+00	8/22/13 11:43
42.32	41.35	1.47E+00	8/22/13 11:44
42.22	41.24	1.47E+00	8/22/13 11:45
42.14	41.15	1.47E+00	8/22/13 11:46
42.05	41.04	1.47E+00	8/22/13 11:47
41.94	40.96	1.47E+00	8/22/13 11:48
41.84	40.86	1.47E+00	8/22/13 11:49
41.74	40.78	1.47E+00	8/22/13 11:50
41.63	40.68	1.47E+00	8/22/13 11:51

41.53	40.58	1.47E+00	8/22/13 11:52
41.44	40.49	1.47E+00	8/22/13 11:53
41.34	40.42	1.47E+00	8/22/13 11:54
41.24	40.31	1.47E+00	8/22/13 11:55
41.13	40.21	1.47E+00	8/22/13 11:56
41.03	40.13	1.47E+00	8/22/13 11:57
40.94	40.02	1.47E+00	8/22/13 11:58
40.84	39.94	1.47E+00	8/22/13 11:59
40.73	39.84	1.47E+00	8/22/13 12:00
40.64	39.75	1.47E+00	8/22/13 12:01
40.54	39.65	1.47E+00	8/22/13 12:02
40.45	39.57	1.47E+00	8/22/13 12:03
40.34	39.49	1.47E+00	8/22/13 12:04
40.26	39.39	1.47E+00	8/22/13 12:05
40.15	39.29	1.47E+00	8/22/13 12:06
40.04	39.2	1.47E+00	8/22/13 12:07
39.95	39.11	1.47E+00	8/22/13 12:08
39.86	39.02	1.47E+00	8/22/13 12:09
39.76	38.94	1.47E+00	8/22/13 12:10
39.66	38.84	1.47E+00	8/22/13 12:11
39.56	38.75	1.47E+00	8/22/13 12:12
39.48	38.66	1.47E+00	8/22/13 12:13
39.38	38.58	1.47E+00	8/22/13 12:14
39.27	38.48	1.47E+00	8/22/13 12:15
39.18	38.4	1.47E+00	8/22/13 12:16
39.08	38.29	1.47E+00	8/22/13 12:17
38.99	38.21	1.47E+00	8/22/13 12:18
38.89	38.13	1.47E+00	8/22/13 12:19
38.8	38.04	1.47E+00	8/22/13 12:20
38.7	37.94	1.47E+00	8/22/13 12:21
38.62	37.86	1.47E+00	8/22/13 12:22
38.54	37.77	1.47E+00	8/22/13 12:23
38.44	37.68	1.47E+00	8/22/13 12:24
38.35	37.6	1.47E+00	8/22/13 12:25
38.25	37.51	1.47E+00	8/22/13 12:26
38.16	37.43	1.47E+00	8/22/13 12:27
38.06	37.35	1.47E+00	8/22/13 12:28
37.99	37.24	1.47E+00	8/22/13 12:29
37.89	37.16	1.47E+00	8/22/13 12:30
37.81	37.09	1.47E+00	8/22/13 12:31
37.73	37	1.47E+00	8/22/13 12:32
37.63	36.92	1.47E+00	8/22/13 12:33
37.55	36.84	1.47E+00	8/22/13 12:34
37.46	36.76	1.47E+00	8/22/13 12:35
37.37	36.69	1.47E+00	8/22/13 12:36
37.3	36.6	1.47E+00	8/22/13 12:37
37.2	36.52	1.47E+00	8/22/13 12:38
37.11	36.44	1.47E+00	8/22/13 12:39
37.03	36.36	1.47E+00	8/22/13 12:40
36.94	36.29	1.47E+00	8/22/13 12:41
36.87	36.21	1.47E+00	8/22/13 12:42
36.79	36.13	1.47E+00	8/22/13 12:43
36.72	36.05	1.47E+00	8/22/13 12:44

36.62	35.97	1.47E+00	8/22/13 12:45
36.54	35.89	1.47E+00	8/22/13 12:46
36.46	35.83	1.47E+00	8/22/13 12:47
36.38	35.74	1.47E+00	8/22/13 12:48
36.31	35.67	1.47E+00	8/22/13 12:49
36.23	35.6	1.47E+00	8/22/13 12:50
36.14	35.53	1.47E+00	8/22/13 12:51
36.06	35.45	1.47E+00	8/22/13 12:52
35.98	35.37	1.47E+00	8/22/13 12:53
35.89	35.29	1.47E+00	8/22/13 12:54
35.83	35.22	1.47E+00	8/22/13 12:55
35.74	35.15	1.47E+00	8/22/13 12:56
35.67	35.07	1.47E+00	8/22/13 12:57
35.6	35.02	1.47E+00	8/22/13 12:58
35.52	34.94	1.47E+00	8/22/13 12:59
35.44	34.87	1.47E+00	8/22/13 13:00
35.37	34.79	1.47E+00	8/22/13 13:01
35.3	34.73	1.47E+00	8/22/13 13:02
35.22	34.67	1.47E+00	8/22/13 13:03
35.16	34.59	1.47E+00	8/22/13 13:04
35.07	34.52	1.47E+00	8/22/13 13:05
35.01	34.46	1.47E+00	8/22/13 13:06
34.92	34.39	1.47E+00	8/22/13 13:07
34.86	34.32	1.47E+00	8/22/13 13:08
34.8	34.25	1.47E+00	8/22/13 13:09
34.72	34.2	1.47E+00	8/22/13 13:10
34.66	34.11	1.47E+00	8/22/13 13:11
34.59	34.06	1.47E+00	8/22/13 13:12
34.51	33.99	1.47E+00	8/22/13 13:13
34.45	33.93	1.47E+00	8/22/13 13:14
34.38	33.87	1.47E+00	8/22/13 13:15
34.31	33.81	1.47E+00	8/22/13 13:16
34.25	33.73	1.47E+00	8/22/13 13:17
34.18	33.68	1.47E+00	8/22/13 13:18
34.12	33.62	1.47E+00	8/22/13 13:19
34.07	33.56	1.47E+00	8/22/13 13:20
33.99	33.5	1.47E+00	8/22/13 13:21
33.93	33.42	1.47E+00	8/22/13 13:22
33.87	33.38	1.47E+00	8/22/13 13:23
33.8	33.32	1.47E+00	8/22/13 13:24
33.74	33.25	1.47E+00	8/22/13 13:25
33.67	33.19	1.47E+00	8/22/13 13:26
33.6	33.12	1.47E+00	8/22/13 13:27
33.56	33.06	1.47E+00	8/22/13 13:28
33.5	33.01	1.47E+00	8/22/13 13:29
33.43	32.96	1.47E+00	8/22/13 13:30
33.38	32.89	1.47E+00	8/22/13 13:31
33.3	32.83	1.47E+00	8/22/13 13:32
33.25	32.79	1.47E+00	8/22/13 13:33
33.18	32.73	1.47E+00	8/22/13 13:34
33.12	32.67	1.47E+00	8/22/13 13:35
33.06	32.61	1.47E+00	8/22/13 13:36
33.01	32.55	1.47E+00	8/22/13 13:37

32.94	32.49	1.47E+00	8/22/13 13:38
32.89	32.44	1.47E+00	8/22/13 13:39
32.84	32.4	1.47E+00	8/22/13 13:40
32.77	32.33	1.47E+00	8/22/13 13:41
32.7	32.29	1.47E+00	8/22/13 13:42
32.66	32.23	1.47E+00	8/22/13 13:43
32.61	32.18	1.47E+00	8/22/13 13:44
32.55	32.12	1.47E+00	8/22/13 13:45
32.5	32.07	1.47E+00	8/22/13 13:46
32.44	32.01	1.47E+00	8/22/13 13:47
32.39	31.96	1.47E+00	8/22/13 13:48
32.34	31.92	1.47E+00	8/22/13 13:49
32.28	31.86	1.47E+00	8/22/13 13:50
32.22	31.79	1.47E+00	8/22/13 13:51
32.17	31.77	1.47E+00	8/22/13 13:52
32.12	31.71	1.47E+00	8/22/13 13:53
32.08	31.67	1.47E+00	8/22/13 13:54
32	31.61	1.47E+00	8/22/13 13:55
31.97	31.57	1.47E+00	8/22/13 13:56
31.92	31.53	1.47E+00	8/22/13 13:57
31.88	31.48	1.47E+00	8/22/13 13:58
31.83	31.43	1.47E+00	8/22/13 13:59
31.78	31.38	1.47E+00	8/22/13 14:00
31.71	31.33	1.47E+00	8/22/13 14:01
31.68	31.29	1.47E+00	8/22/13 14:02
31.63	31.24	1.47E+00	8/22/13 14:03
31.58	31.2	1.47E+00	8/22/13 14:04
31.53	31.16	1.47E+00	8/22/13 14:05
31.47	31.11	1.47E+00	8/22/13 14:06
31.45	31.07	1.47E+00	8/22/13 14:07

BIBLIOGRAPHY

[1] S. Janson, "Mass-producible silicon spacecraft for 21st century missions," in *Proceedings of the AIAA Space Technology Conference and Exposition*, Albuquerque, NM, 1999.

[2] J. A. Atchison and M. A. Peck, "A millimeter-scale lorentz-propelled spacecraft," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, AIAA, Hilton Head, SC, 2007.

[3] D. J. Barnhart, T. Vladimirova, and M. N. Sweeting, "Very-small-satellite design for distributed space missions," *Journal of Spacecraft and Rockets*, vol. 44, no. 6, pp. 1294–1306, 2007.

[4] J. A. Atchison, Z. R. Manchester, and M. A. Peck, "Microscale atmospheric reentry sensors," in *Proceedings of the 7th International Planetary Probe Workshop*, Barcelona, Spain, 2010.

[5] Z. R. Manchester and M. A. Peck, "Stochastic space exploration with microscale spacecraft," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Portland, OR, 2011.

[6] C. Escoubet, R Schmidt, and M. Goldstein, "Cluster – science and mission overview," *Space Science Reviews*, vol. 79, no. 1–2, pp. 11–32, 1997.

[7] V. Angelopoulos, "The THEMIS mission," *Space Science Reviews*, vol. 141, no. 1–4, pp. 5–34, 2008.

[8] J. A. Atchison and M. A. Peck, "Length scaling in spacecraft dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 1, pp. 231–246, 2011.

[9] C. Colombo and C. R. McInnes, "Orbital dynamics of earth-orbiting smart-dust spacecraft under the effects of solar radiation pressure and aerodynamic drag," in *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference*, Toronto, 2010.

[10] D. J. Barnhart, T. Vladimirova, A. M. Baker, and M. N. Sweeting, "A low-cost femtosatellite to enable distributed space missions," *Acta Astronautica*, vol. 64, pp. 1123–1143, 2009.

[11] *Triangular advanced solar cells*, Spectrolab Inc., Sylmar, CA, Apr. 2002.

[12] M. Noca, F. Jorndan, N. Steiner, T. Choueiri, F. George, G. Roethlisberger, N. Scheidegger, H. Peter-Contesse, M. Borgeaud, R. Krpoun, and H. Shea, "Lessons learned from the first swiss pico-satellite: Swisscube," in *Proceedings of the AIAA/USU Conference on Small Satellites*, Logan, UT, 2009.

[13] B. Klofas and K. Leveque, "A survey of CubeSat communication systems: 2009–2012," in *CubeSat Developers' Workshop*, Barcelona, Spain, Apr. 2013.

[14] J. R. Wertz and W. J. Larson, *Space Mission Analysis and Design*, 3rd ed. Torrance, CA: Microcosm Press, 1999.

[15] B. Sklar, *Digital Communications*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2001.

[16] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed. Lincoln, MA: Ganga-Jamuna Press, 2010.

[17] L. Welch, "Lower bounds on the maximum cross correlation of signals," *IEEE Transactions on Information Theory*, vol. 20, no. 3, pp. 297–399, 1974.

[18] R. Gold, "Optimal binary sequences for spread spectrum multiplexing," *IEEE Transactions on Information Theory*, vol. 13, no. 4, pp. 619–621, 1967.

[19] T. K. Moon, *Error Correction Coding*, 1st ed. Hoboken, NJ: Wiley, 2005.

[20] *CC1101 low-power sub-1 GHz RF transceiver*, Texas Instruments, Dallas, TX, 2014.

[21] R. Barker, "Group synchronizing of binary digital systems," in *Communication Theory*, W Jackson, Ed., New York: Accademic Press, 1953, pp. 279–287.

[22] E. Blossom, "Gnu radio: Tools for exploring the radio frequency spectrum," *Linux Journal*, vol. 122, Jun. 2004.

[23] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed. Cambridge, UK: Cambridge University Press, 2007.

[24] *USRP manual*, Ettus Research, Inc., Santa Clara, CA, May 2015.

[25] M. Kravchenko, M. J. O'Rourke, J. Golden, M. Finckenor, M. Leatherwood, and J. Alred, "MISSE 6, 7 and 8 materials sample experiments from the international space station materials and processes team," in *Proceedings of the National Space and Missile Materials Symposium*, Scottsdale, AZ, 2010.

[26] *CubeSat design specification*, Rev. 13, California Polytechnic State University, San Luis Obispo, CA, 2014.

[27] L. Bell, "NASA completes successful phonesat mission," *NASA Tech Briefs*, vol. 37, no. 7, p. 8, 2013.

[28] P. C. Hughes, *Spacecraft Attitude Dynamics*. Mineola, New York: Dover Publications, 2004.

[29] K. H. Wiener, "The role of mass properties measurement in the space mission," in *Proceedings of the European Conference on Spacecraft Structures, Materials, and Mechanical Testing*, Noordwijk, The Netherlands, May 2005.

[30] T. Kane and G. Tseng, "Dynamics of the bifilar pendulum," *International Journal of Mechanical Sciences*, vol. 9, no. 2, pp. 83–96, 1967.

[31] M. R. Jardin and E. R. Mueller, "Optimized measurements of uav mass moment of inertia with a bifilar pendulum," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Hilton Head, SC, 2007.

[32] H. Goldstein, C. Poole, and J. Safko, *Classical Mechanics*, 3rd ed. San Francisco: Addison Wesley, 2001.

[33] "Program level dispenser and CubeSat requirements document," John F. Kennedy Space Center, FL, Tech. Rep. LSP-REQ-317.01, Jan. 2014.

[34] "General environmental verification standard," Greenbelt, MD, Tech. Rep. GSFC-STD-7000, Apr. 2013.

[35] E. V. Bergmann, B. K. Walker, and D. R. Levy, "Mass property estimation for control of asymmetrical satellites," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 5, pp. 483–491, 1987.

[36] J. Ahmed, V. T. Coppola, and D. S. Bernstein, "Adaptive asymptotic tracking of spacecraft attitude motion with inertia matrix identification," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 5, pp. 684–691, 1998.

[37] M. L. Psiaki, "Estimation of a spacecraft's attitude dynamics parameters by using flight data," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 4, pp. 594–603, 2005.

[38] S. Tanygin and T. Williams, "Mass property estimation using coasting maneuvers," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 4, pp. 625–632, 1997.

[39] M. C. Norman, M. A. Peck, and D. J. O'Shaughnessy, "In-orbit estimation of inertia and momentum-actuator alignment parameters," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 6, pp. 1798–1814, 2011.

[40] J. A. Keim, B. A. Acikmese, and J. F. Shields, "Spacecraft inertia estimation via constrained least squares," in *Proceedings of the IEEE Aerospace Conference*, IEEE, 2006.

[41] M. A. Peck, "Uncertainty models for physically realizable inertia dyadics," *The Journal of the Astronautical Sciences*, vol. 54, no. 1, pp. 1–16, 2006.

[42] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2009.

[43] S. Altmann, *Rotations, Quaternions, and Double Groups*, ser. Oxford Science Publications. Oxford, UK: Clarendon Press, 1986.

[44] S. P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. SIAM, 1994, vol. 15.

[45] P. Nikravesh, R. Wehage, and O. Kwon, "Euler parameters in computational kinematics and dynamics. part 1," *Journal of Mechanical Design*, vol. 107, no. 3, pp. 358–365, 1985.

[46] F. E. Udwadia and A. D. Schutte, "An alternative derivation of the quaternion equations of motion for rigid-body rotational dynamics," *Journal of Applied Mechanics*, vol. 77, no. 4, p. 044 505, 2010.

[47] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*, 2nd ed. Reston, VA: AIAA, Oct. 2009.

[48] A. M. Fosbury and C. K. Nebelecky, "Spacecraft actuator alignment estimation," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Chicago, 2009, pp. 10–13.

[49] J. E. Marsden and M. West, "Discrete mechanics and variational integrators," *Acta Numerica*, vol. 10, pp. 357–514, May 2001, ISSN: 1474-0508.

[50] Z. R. Manchester and M. A. Peck, "Quaternion variational integrators for spacecraft dynamics," *Journal of Guidance, Control, and Dynamics*, 2015.

[51] J. Marsden and T. Ratiu, *Introduction to Mechanics and Symmetry*. New York: Springer-Verlag, 1994.

[52] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[53] F. L. Markley, "Attitude error representations for kalman filtering," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[54] L. Trefethen and D. Bau, *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1997.

[55] G. Golub and C. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 2012.

[56] *ITG-3200 product specification*, InvenSense Inc., Sunnyvale, CA, Mar. 2010.

[57] J. F. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 625–653, 1999.

[58] H. D. Mittelmann, "An independent benchmarking of SDP and SOCP solvers," *Mathematical Programming*, vol. 95, no. 2, pp. 407–430, 2003.

[59] R. N. Bracewell and O. K. Garriott, "Rotation of artificial earth satellites," *Nature*, vol. 182, pp. 760–762, Sep. 1958.

[60] W. J. Devey, C. F. Field, and L. Flook, "An active nutation control system for spin stabilized satellites," *Automatica*, vol. 13, pp. 161–172, 1977.

[61] C. S. Fish, C. M. Swenson, *et al.*, "Design, development, implementation, and on-orbit performance of the dynamic ionosphere cubesat experiment mission," *Space Science Reviews*, vol. 181, pp. 61–120, Feb. 2014.

[62] Z. Manchester, M. Peck, and A. Filo, "Kicksat: A crowd-funded mission to demonstrate the world's smallest spacecraft," *AIAA/USU Conference on Small Satellites*, 2013.

[63] E. D. Wise, C. M. Pong, *et al.*, "A dual-spinning, three-axis-stabilized cubesat for earth observations," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Boston, Aug. 2013.

[64] P. M. Barba, N. Furumoto, and I. P. Leliakov, "Techniques for flat-spin recovery of spinning satellites," in *Proceedings of the AIAA Guidance and Control Conference*, Key Biscayne, FL, Aug. 1973, pp. 859–867.

[65] J. R. Gebman and D. L. Mingori, "Perturbation solution for the flat spin recovery of a dual-spin spacecraft," *AIAA Journal*, vol. 14, no. 7, pp. 859–867, 1976.

[66] D. A. Lawrence and T. E. Holden, "Essentially global asymptotically stable nutation control using a single reaction wheel," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 6, pp. 1783–1793, 2007.

[67] C. D. Rahn and P. M. Barba, "Reorientation maneuver for spinning spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 4, pp. 724–728, Jul. 1991.

[68] N. H. Beachley and J. J. Uicker, "Wobble-spin technique for spacecraft inversion and earth photography," *Journal of Spacecraft and Rockets*, vol. 6, no. 3, pp. 245–248, Mar. 1969.

[69] N. H. Beachley, "Inversion of spin-stabilized spacecraft by mass translation - some practical aspects," *Journal of Spacecraft and Rockets*, vol. 8, no. 10, pp. 1078–1080, Oct. 1971.

[70] H.-S. Myung and H. Bang, "Predictive nutation and spin inversion control of spin-stabilized spacecraft," *Journal of Spacecraft and Rockets*, vol. 47, no. 6, pp. 1010–1022, 2010.

[71] P. Lu, "Nonlinear predictive controllers for continuous systems," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 3, pp. 553–560, 1994.

[72] H. Khalil, *Nonlinear Systems*, 3rd ed. Englewood Cliffs, NJ: Prentice Hall, 2002.

[73] J. Slotine and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice Hall, 1991.

[74] C Masaitis, "On the motion of two linked bodies," *Archive for Rational Mechanics and Analysis*, vol. 8, no. 1, pp. 23–35, 1961.

[75] J. Wittenburg, *Dynamics of Systems of Rigid Bodies*, 1st ed. Stuttgart: Teubner, 1977.

[76] S. P. Bhat and D. S. Bernstein, "A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon," *Systems and Control Letters*, vol. 39, no. 1, pp. 63–70, 2000.

[77] L. F. Shampine and M. W. Reichelt, "The Matlab ODE suite," *SIAM Journal on Scientific Computing*, vol. 18, no. 1, pp. 1–22, 1997.

[78] E. Hairer, C. Lubich, and G. Wanner, "Geometric numerical integration illustrated by the stormer-verlet method," *Acta Numerica*, vol. 12, pp. 399–450, May 2003, ISSN: 1474-0508.

[79] T. Lee, N. McClamroch, and M. Leok, "A lie group variational integrator for the attitude dynamics of a rigid body with applications to the 3D pendulum," in *Proceedings of the IEEE Conference on Control Applications*, Toronto: IEEE, Aug. 2005, pp. 962–967.

[80] I. Hussein, M. Leok, A. Sanyal, and A. Bloch, "A discrete variational integrator for optimal control problems on SO(3)," in *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego: IEEE, Dec. 2006, pp. 6636–6641.

[81] T. Lee, M. Leok, and N. H. McClamroch, "Lie group variational integrators for the full body problem in orbital mechanics," *Celestial Mechanics and Dynamical Astronomy*, vol. 98, no. 2, pp. 121–144, 2007.

[82] E. Barth and B. Leimkuhler, "Symplectic methods for conservative multibody systems," *Fields Institute Communications*, vol. 10, pp. 25–43, 1993.

[83] R. I. McLachlan, "Explicit Lie-Poisson integration and the Euler equations," *Physical Review Letters*, vol. 71, pp. 3043–3046, 19 Nov. 1993.

[84] I. P. Omelyan, "Algorithm for numerical integration of the rigid-body equations of motion," *Physical Review E*, vol. 58, no. 1, p. 1169, 1998.

[85] P Krysl, "Direct time integration of rigid body motion with discrete-impulse midpoint approximation: Explicit Newmark algorithms," *Communications in Numerical Methods in Engineering*, vol. 22, no. 5, pp. 441–451, 2006.

[86] D. D. Holm, *Geometric Mechanics: Rotating, Translating, and Rolling*, ser. Geometric Mechanics. London: Imperial College Press, 2011, vol. 2.

[87] C. Lanczos, *The Variational Principles of Mechanics*, 4th ed. Mineola, New York: Dover Publications, 1986.

[88] H. S. Morton Jr, "Hamiltonian and Lagrangian formulations of rigid-body rotational dynamics based on the Euler parameters," *Journal of the Astronautical Sciences*, vol. 41, pp. 569–591, 1993.

[89] U. Ascher and C. Greif, *A First Course on Numerical Methods*, ser. Computational Science and Engineering. Philadelphia: SIAM, 2011.

[90] T. Kane and P. Barba, "Effects of energy dissipation on a spinning satellite," *AIAA Journal*, vol. 4, no. 8, pp. 1391–1394, 1966.

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

[92] M. Villano, "Small donations in large numbers, with online help," *The New York Times*, Mar. 14, 2010.

[93] R. Walker, "The trivialities and transcendence of Kickstarter," *The New York Times*, Aug. 5, 2011.