12-6-2016

Attitude Determination Control Testing System (Helmholtz Cage and Air Bearing)

Nicolas Theoret
Western Michigan University, nicolas.o.theoret@gmail.com

Recommended Citation
Theoret, Nicolas, "Attitude Determination Control Testing System (Helmholtz Cage and Air Bearing)" (2016). *Honors Theses*. 2783.
https://scholarworks.wmich.edu/honors_theses/2783

CubeSat Attitude Determination and Control System Validation and Testing Apparatus

For
Dr. Jennifer Hudson
Dr. Bade Shrestha
Western Michigan University
Kalamazoo, MI

by
Jacob Stevens
Nicolas Theoret
Tyler Chantrenne

Submitted December 13, 2016

DISCLAIMER

Abstract

The WALI team at Western Michigan University requested a test environment to validate their CubeSat's de-tumbling control system and hardware. The test environment required a Helmholtz cage and spherical air bearing. The Helmholtz cage provides an adjustable magnetic field to simulate low earth orbit; the spherical air bearing simulates the friction free environment the CubeSat will experience in space. In conjunction, the two components create an adjustable system that simulates a satellite in low earth orbit.

List of Figures

List of Tables

Table of Contents

1 Introduction

CubeSats are small satellites that are built by various research institutions. They allow research to be conducted in space at a fraction of the cost of a fully built satellite. WMU is currently constructing a CubeSat that will perform plasma plume diagnostics in space. The Western Aerospace Launch Initiative (WALI) is expected to launch in 2017 and is funded by the United States Air Force. To carry the CubeSat into low Earth orbit, a rocket will transport a large quantity of CubeSats to the desired altitude. The CubeSats will then be ejected into space. To correctly orient the satellite, an attitude control system composed of reaction wheels and magnetorquers may be used. This properly orients the satellite relative to the Sun and Earth to accurately conduct the research.

To ensure that the attitude control system works in space conditions, both a friction free environment and magnetic field must be replicated in a laboratory. Our project's goal is to fabricate the attitude control system test environment and the measuring system to accurately calibrate our satellite for space conditions. The two main components include a Helmholtz cage and a spherical air bearing test bed.

2 Objective

Our project objective: simulate earth's magnetic field at an altitude of 500 km in zero gravity conditions. The benchmarks of the project are split into three main components and their constituents:

1. Constructing the Helmholtz cage
    a. Cage frame
    b. Cage coils
    c. Power Supply
2. Constructing the Air Bearing Table
    a. Designing
    b. Manufacturing
    c. Testing
3. Validating Uniformity of magnetic field

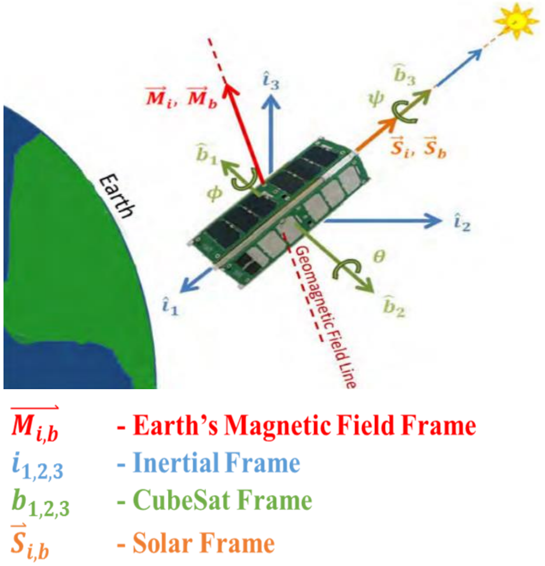


Figure 1: CubeSat Attitude Control System. [2]

3 Helmholtz Cage

3.1 Overview

WALI's satellite stabilizes and orients in space using three components: a sun sensor, a magnetometer, and a magnetorquer. The sun sensor determines the CubeSat's body angles with respect to the sun. The magnetometer detects the orientation and strength of the magnetic field. The data from the sun sensor and magnetometer are used to control the magnetorquer. A magnetorquer controls the attitude of the CubeSat by generating a magnetic field. The field generated applies a torque on the CubeSat as it aligns with the Earth's magnetic field. By switching the magnetorquer on or off, we can “de-tumble” the CubeSat in space over time. The purpose of the Helmholtz cage is to test the magnetometer and magnetorquer of the CubeSat. To do this, the cage must generate a stable and uniform magnetic field in 3 directions. Figure 1 is a representation of the multiple frames used to calculate the proper orientation in space.[1,2,3,4]

3.2 Helmholtz Cage Physics Principles

A Helmholtz cage operates on a single physics principle: Biot-Savart law (Equation 1).

$$dB = \frac{\mu_0 I}{4\pi}\frac{dL\ x\ \vec{r}}{r^3} \qquad (1),$$

where B is the magnitude of the magnetic field measured in Tesla (T), I is the current through the coil, r is the radius of the coil, $\vec{r}$ is the position vector away from the current, and μ_0 is the permeability of free space ($4\pi * 10^{-7}$ N A^{-2}). The law states that current through a wire is directly proportional to the magnetic field it produces (Figure 2). By wrapping that wire into a tight coil, or a square shape, we can generate a magnetic field plane with the maximum magnetic field at the center. The magnetic field of a single coil is calculated using Equation 2.

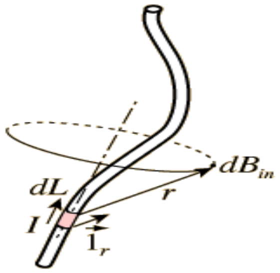


Figure 2: Biot-Savart law applied to current through a wire.[2]

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

where R is the radius of the coil and x is the distance from the coil axis to the radius. Modifying the number of wire turns for the coil and the current through the wire, we can control the magnitude and direction of the magnetic field plane. Additionally, by combining two coils on the same axis, we can control the magnetic field in a volume along that axis, as shown in Figure 3 & 4. This is called a Helmholtz coil pair.

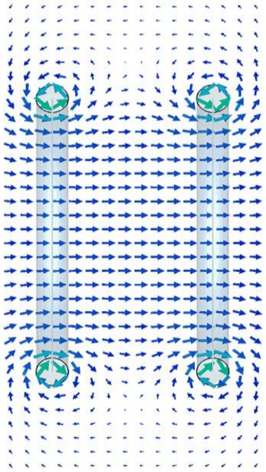


Figure 3: Magnetic field uniformity between two coils (Ref. F3)

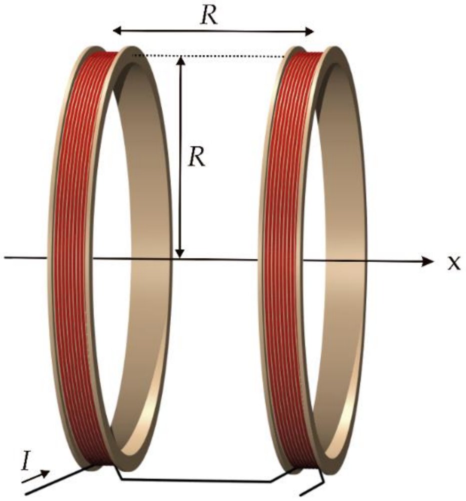


Figure 4: Single axis Helmholtz coil pair. (Ref. F4)

Equation 3 is a modified version of Equation 2 for the magnetic field produced at the center of two adjacent coils along the x-axis:

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

where n is the number of turns and I is the current going through the coils. Although circular coils produce the most uniform magnetic field, square coils are easier to construct for our application. We can accommodate Equation 3 for a square coil application using Equation 4:

$$B(x) = \frac{2\mu_0 n I}{\pi}\left(\frac{a^2}{[(z-d)^2+a]\sqrt{(z-d)^2+2a^2}} + \frac{a^2}{[(z+d)^2+a]\sqrt{(z+d)^2+2a^2}}\right) \qquad (4).$$

Where a is side length of the square coils, d is the distance between the coils, n is the number of wire loops (also known as turns), and I is the current going through the loops. Equation 4 is used to model all three axis of a Helmholtz cage.[12, 13]

3.3 Helmholtz Cage Design

To accurately test the WMU CubeSat attitude control system, the Helmholtz cage design has two constraints. The first constraint is the cage must be capable of generating a magnetic field twice as strong as the Earth's magnetic field. This is for two reasons: First, we need to neutralize the magnetic field in the Kalamazoo area, equivalent to 0.50 Gauss. Second, the cage must generate a magnetic field equivalent to the field measured at 600km of altitude, equivalent to 0.450 Gauss.[15] These two conditions indicate that the cage must produce a minimum magnetic field of .989 Gauss. The second constraint is that the field must be uniform over a large volume to fit a variety of CubeSats. Based on the size of a 6U CubeSat, this requires that the homogenous magnetic field's minimum volume must be .125m^3.[13, 14]

Additionally, a Helmholtz cage must be entirely made from non-ferrous materials. Ferrous materials affect the magnetic fields, which would disturb the magnetic field produced in the Helmholtz cage. Therefore, the frame and the coils must be made entirely from aluminum or synthetic materials. It is also important to remove any ferrous materials inside the cage during testing to ensure a uniform magnetic field.

Multiple Helmholtz Cage designs were researched including University of Michigan, Massachusets Institute of Technology, and Air Force Institute of Technology. We determined that to meet the requirements, our cage would have to be an improved design of the University of Michigan's Helmholtz cage. What we had to consider from their conclusion was that their Helmholtz cage did not meet the required magnetic field strength they had designed for.[13] Therefore, we overdesigned our Helmholtz cage to ensure that we could produce a magnetic field strong enough to obtain a magnitude of 3 Guass. Therefore, our summarized design goals for the Helmholtz Cage are as follows:

1. Maximum magnetic field of 3 Gauss per axis.

2. Homogenous magnetic field in a volume of $0.25m^3$.

3. Total cost of the cage under $2000.

3.4 Helmholtz Cage – Coil Design, Optimization, and Assembly

3.4.1 Coil Size

A Helmholtz cage design begins with designing the coils. According to the Biot-Savart law, the size of the coils, and their distance apart, is proportional to the size of the magnetic field volume we need to produce. Additionally, the coils need to fit within one another to allow for the magnetic field to be produced in three directions. Producing a homogenous field volume of $0.25m^3$ means that the smallest coil pair must produce a magnetic field plane with an area of $0.4m^2$. Therefore, Matlab was used to test different scenario's of coil sizes that could achieve a 3 Gauss magnetic field plane over an area of $0.4m^2$. The code for this Matlab Analysis is displayed in Appendix A.

The results of the analysis indicated that the dimensions of the smallest coils needed to be 1.7m per side. From the minimum coil dimensions, we calculated that the coils for the other two axes would be 1.8m and 1.9m per side, allowing room for the structure of the coils. Additionally, the optimal spacing for all coil pairs was determined from Equation 5:

$$d = .5445L \quad (5),$$

where d is the distance between the coils and L is the length of the coil side.[12] Table 1 summarizes the result of the coil size analysis.

Axis	Coil Side Length	Coil Distance

X	170cm	92.5cm
Y	180cm	98.0cm
Z	190cm	103.5cm

Table 1: Summary of Coil Dimensions

3.4.2 Current, Wire Gauge, and Turns Per Coil

With the coil size determined, the next step is to design and optimize the current, wire gauge, and number of turns for each coil. According to the Biot-Savart law, all three of these parameters are related to the magnetic field (Equation 4). Therefore, to generate a magnetic field with a magnitude of 3 Gauss, each variable is iterated to select the optimal parameter. There are two consideations for coil optimization:

1. maximum current is limited by the wire gauge,
2. wire gauge must be similar for all six coils.

A design matrix was used to identify the optimal parameters, displayed in Appendix B. The MatLab code from Appendix A was also used to optimize the parameters. The results of the analysis is concluded in Table 2. These were chosen as the optimal parameters due to the low number of turns required, safe current levels, and low cost of the copper wire required.

Coil Pair	Coil Side Length	Wire Gauge (AWG)	Max Current	Number of Turns
X-axis	170cm	10	12.5 Amps	17

Y-axis	180cm	10	12.5 Amps	18
Z-axis	190cm	10	12.5 Amps	19

Table 2: Results of current, wire gauge, and turns for each coil.

3.4.3 Coil Assembly

The coil structure was chosen to be made from C-channel aluminum for its rigidity and non-ferrous properties. Each coil is composed of four C-channel aluminum members joined at the corners with triangular gussets, as shown in Figure 5. The gussets and C-channel members were fastened together using nylon fasteners. The coils were wrapped with the 10 AWG magnet wire. During the wrapping phase, we insured that the wire was spaced out evenly along the bottom of the C-channel aluminum and pulled tight along the corners, as shown in Figure 6. This ensures that the integrity of the coils is as close to ideal as possible. Lastly, the outside of each coil is wrapped with electrical tape to fix the wire in place and prohibit any tampering.

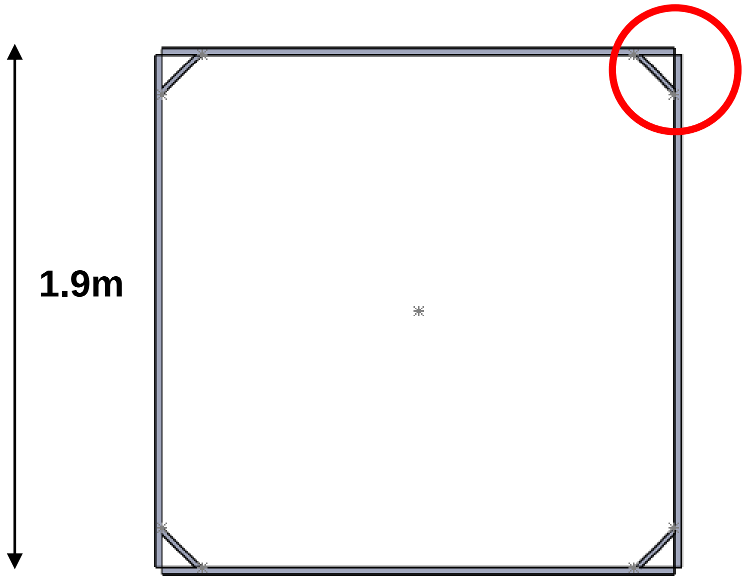


Figure 5: Coils joined at corners with gussets (circled in red).

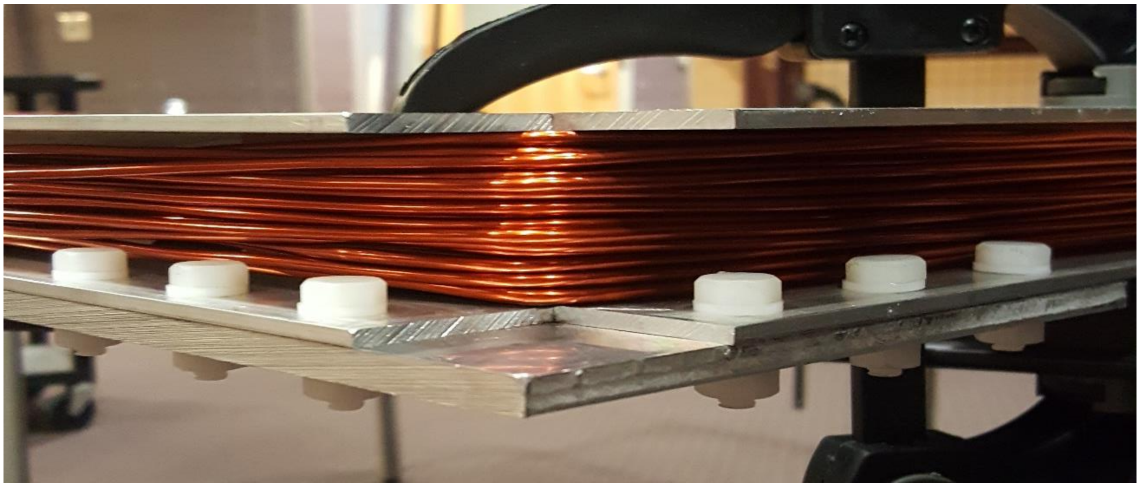


Figure 6: Coils wrapped with copper magnet wire.

3.5 Helmholtz Cage – Frame Design and Assembly

All six coils of the Helmholtz cage must be fixed in a specific configuration. Additionally, the centers of the coil pairs must be concentric and all three axes must meet in the center of the cage. To accomplish this, a solid and easy to adjust frame was required. Our frame was thankfully donated by the University of Michigan which was used previously to support Helmholtz coils. The frame's dimensions are 2m x 2m x 2m which was sufficient to support our coils. The frame is constructed from T-slotted aluminum members, which is ideal for a rigid and easily adjustable frame. The frame also came with caster wheels and support structures for the coils. All materials for the cage are listed in Appendix C, along with the frame assembly instructions.[17]

3.6 Helmholtz Cage – Final Assembly

All coils were assembled into the frame as shown in Figure 7. The X and Y axis coils were hung from the top of the cage and the Z-axis coils were fixed horizontally. The coils were positioned within ¼” of their design position. Extra care was taken to ensure that the distance between the coils matched the design plans. Figure 7 displays the Helmholtz cage fully assembled. The X, Y, and Z axes were all marked and identified. The entire bill of materials for the Helmholtz cage is located in Appendix D.

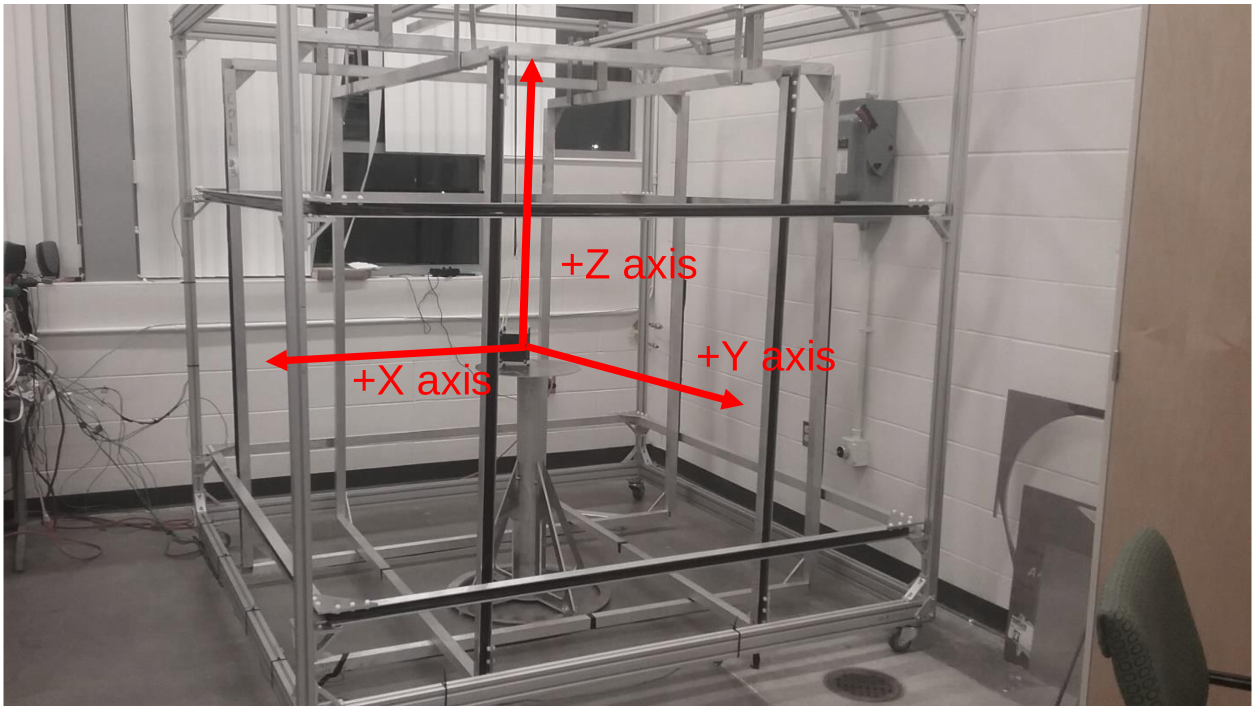


Figure 7: Frame and coils assembled together

4 Power Supply Design, Assembly, and Testing

4.1 Power Supply – Design & Assembly

According to the coil design and optimization, a maximum of 12.5 amps are required to power the coils for a maximum magnetic field of 3 Gauss. There are three constraints for designing the coil power system:

1. must be safe
2. consistent DC output of 12.5 amps,
3. must be user friendly and programmable.

To meet these requirements, we considered the design shown in Appendix E. The design uses an AC to DC power converter, an H-bridge to control the current flow, and an Arduino to control the H-bridges, and current sensor for each individual coil. Appendix E also contains diagrams for the H-bridge, power source, and Arduino pins, including the wire color code. The power system's current direction is controlled with the six switches located on the power source box. The six switches will operate the coils in the positive direction when pointing up, and negative direction when pointing down, also shown in Appendix E. The system is also fused to 12.5 amps to ensure that the current through the coils does not exceed the designed maximum limit. The current sensors allow the system to maintain stability through a feedback loop as shown in Figure 8.

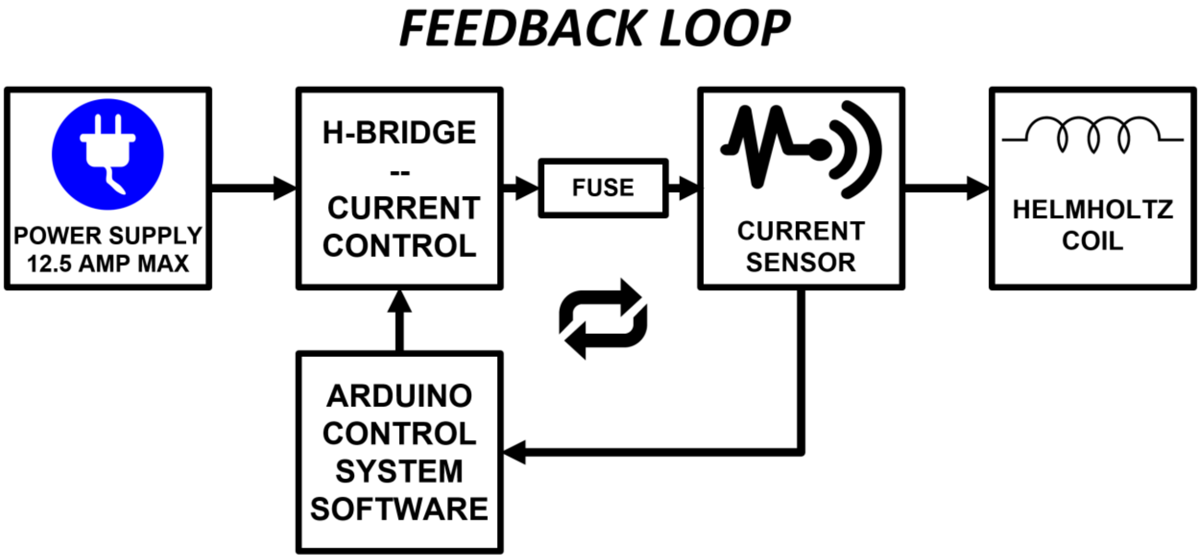


Figure 8: Feedback control of the power system

The Arduino is the controller for the entire system. The Arduino reads the data from the sensor and compares it to the desired current value. If the desired current does not match the measured current through the coil, the Arduino will adjust its PWM output to the H-bridge to increase or decrease the current through the coils accordingly. The control system software used in the Arduino is displayed in Appendix F.

There are two important things to note about the power supply:

1. The current sensor portion of the feedback loop was difficult to calibrate and therefore does not operate correctly.
2. We strongly advise against operating the coils continuously over long periods of time. The H-bridges and power sources will get very hot! Use caution.

4.2 Power Supply – Testing

Each coil of the Helmholtz cage was tested to determine how the PWM value in the Arduino code relates to the current going through the coil. The data from the testing is located in Appendix G including the graphs. According to the data, the PWM value is linearly proportional to the current through each coil according to Equation 6:

$$Current = a * PMW + b \quad (6),$$

where a and b are constants unique to each coil; these are displayed in Appendix G. The instruments used for measuring the current through each coil were unable to detect any noise or variation in the current indicating that the current is constant.

5 Helmholtz Cage Testing

5.1 Coil Validation

The Helmholtz coils must now be validated to prove they are actually creating a magnetic field. The coils were validated with a Gauss meter from Alpha Lab Inc. (model VGM). The meter was first placed outside the cage to read the magnet field generated by earth. Figure 9 displays the Gauss meter and its reading of 0.36 gauss on 12/1/2016. A single coil was then powered and the meter is placed within the coil. To establish a baseline, a gauss reading was taken 3 feet from the edge of a single coil (towards the center). Figure 10 displays the validation of coil 6. The results are reported in table 2.

Figure 9: Earth's Magnetic Field

Figure 10: Coil 6 Validation

	Reading 1 (Gauss)
Date	12/1/2016
Ambient	0.36
Coil 1	1.56
Coil 2	1.48
Coil 3	1.75
Coil 4	1.42
Coil 5	1.47
Coil 6	1.88

Table 3: Single Coil Guess Measurements

5.2 Uniform Field Test Setup

The coils have been proven to produce a sufficient magnetic field, although for proper testing of the ADCS, a uniform field must be present. The question, "Is the CubeSat actually experiencing a uniform magnetic field?" must be addressed. Two magnetometers are incorporated into a testing system in order test the uniformity of the magnetic field. Figure 11 below displays the 1 axis system developed to test magnetic field uniformity. The defined axis was fixed on top of the test stand. The blue foam and sensors were aligned in each of the three directions and measurements were taken for 30 seconds. Tests were then conducted with the Uniform Magnetic Field Testing Procedure in Appendix Q.

An Arduino Uno R3 controlled the sensors and reports the magnetic field measurements to a txt file. The Arduino code, magsensor.ino, can be viewed in Appendix J.

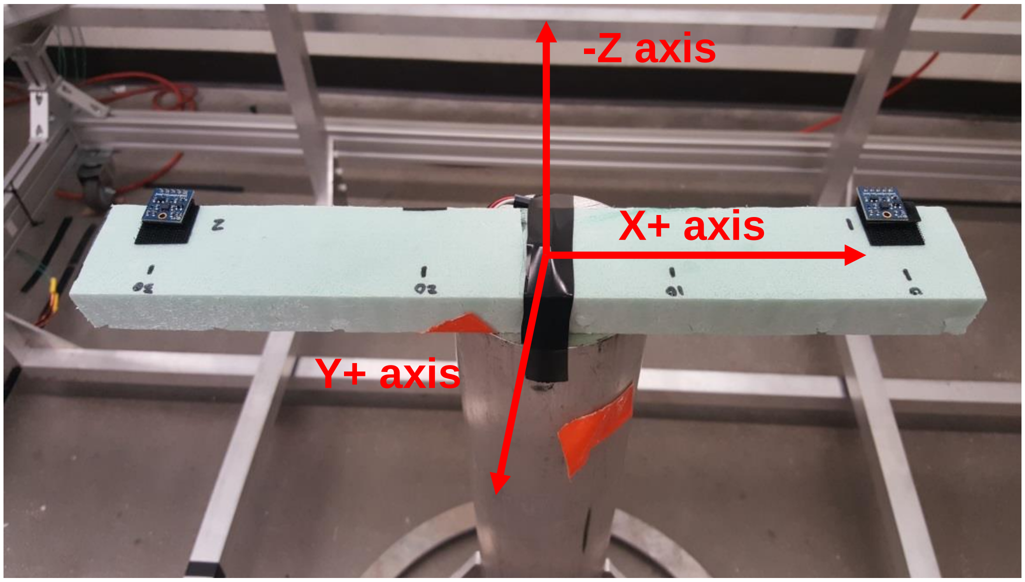


Figure 11: Magnetometers on Testing Stand

The program magsensor.ino separately records two HMC5883L magnetometers, at known distance apart, together to provide a change in strength of magnetic field per unit length in the x, y, and z direction. The CubeSat requires a magnetic field of volume $0.25m^3$, therefore there must be a uniform magnetic field of 0.63m x 0.63m x 0.63m centered at the center of volume of the CubeSat. Figure 12 below shows the Arduino's connection to the computer. The magnetometers have a resolution of 0.73 to 4.35 milli-gauss and a full scale field range of -8 to 8 Gauss. It is important that the output rate remains at 75 Hz, for accurate continuous results.

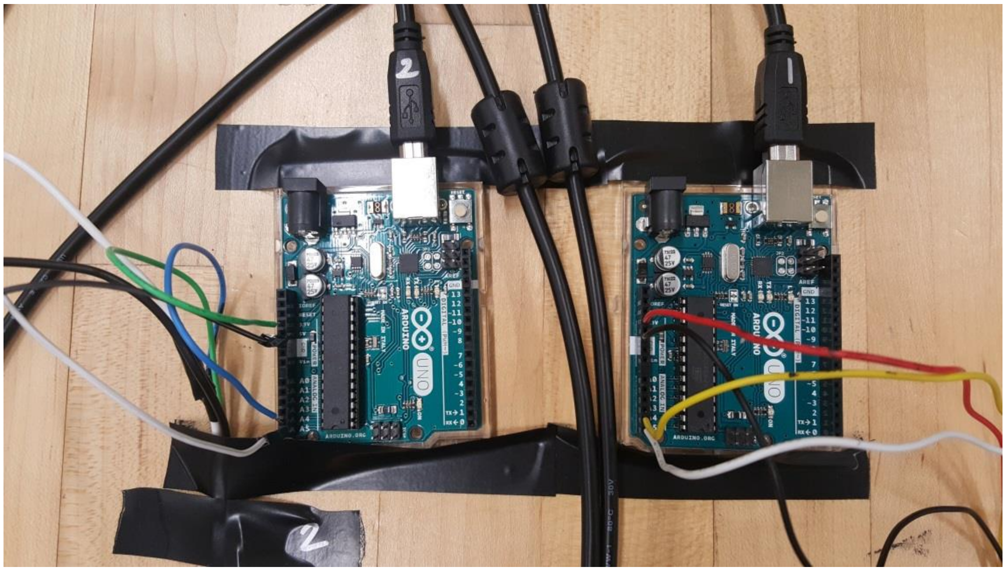


Figure 12: Arduinos Powering Magnetometers

5.3 Uniform Field Testing Results

Testing results were plotted using MATLAB script Difference_plot.m (Appendix K). The script pulled data from testdatav3.mat (Appendix P) which was gathered through two magnetometers. Three plots were created for each set of Helmholtz coils in all three axis directions (See Figure 11 above). The data is displayed in Appendix L, M, N, and O. Overall, magnetic field variation within all coils is less than +/- .03 gauss per decimeter at 7 AMPs of current. Well within acceptable magnetic field variation. Figure 13 below displays a plot of the variation of the magnetic field per decimeter of the X-Coil in the x-axis. The two blue lines display a +/- .05 gauss per decimeter variation. The actual Gauss variation is +/-.02 Gauss per decimeter of the x-axis for the X-Coil.

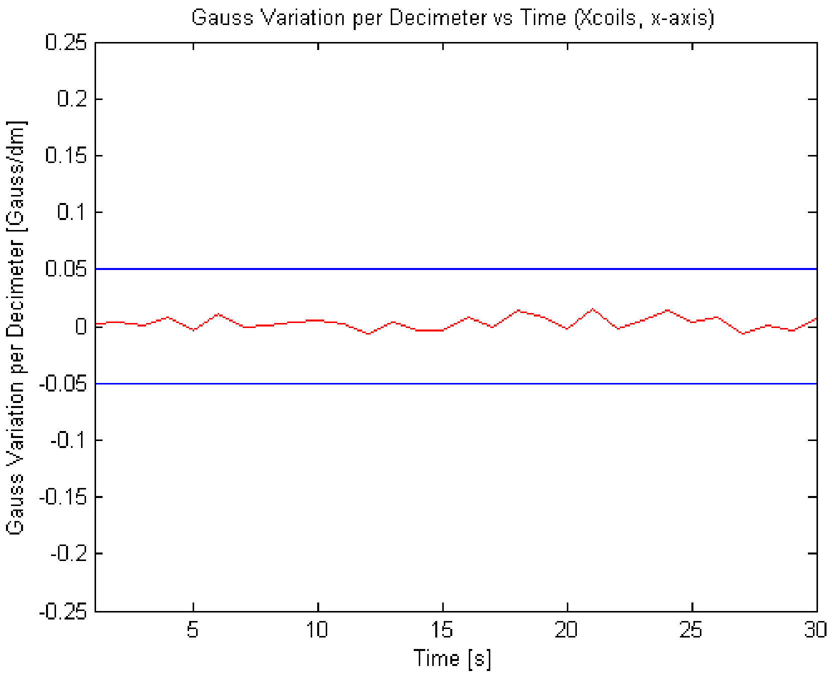


Figure 13: Gauss Variation of X-Coil in x-axis

Magnetometers with better resolution and accuracy will provide better results. Furthermore, initial testing shows an increase in field uniformity as the strength of the magnetic field is increased (increased current). Also, further testing shows an increase in field uniformity when all 6 coils are active.

6 Air Bearing Test Bed

6.1 Overview

The spherical air bearing and test bed provide two things: a frictionless motion environment to allow the CubeSat to tumble freely, and elevates the CubeSat into the uniform magnetic field produced by the Helmholtz cage. The spherical air bearing and test bed both need to be lightweight so they can be easily moved and made of a non-ferrous material so they do not interfere with the magnetic field produced by the Helmholtz cage. This problem was solved by utilizing 6061 aluminum for both the bearing and the test bed.

6.2 Spherical Air Bearing Design

6.2.1 Concepts and Principles

A spherical air bearing provides a frictionless testing environment. The bearing and load are supported by input compressed air. The compressed air creates a thin film below the half hemisphere and provides stiffness. The film of air is approximately 0.0001 inches thick. The air can reach the inside of the cup by multiple methods: a single orifice, multiple orifices, or a porous surface. Figure 13 displays the importance of geometry in maintaining the thin film of air.

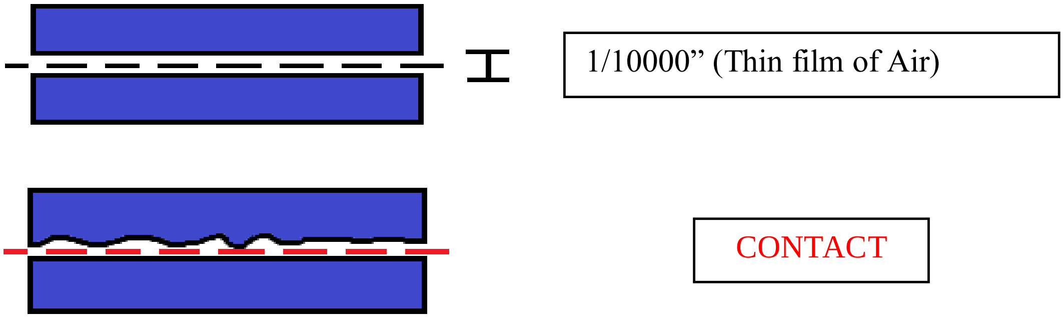


Figure 14: Important of Geometry

Porosity, burrs, roughness, poor handling and other aspects can create interference. Any contact within the bearing cup will render the bearing useless.

The bearing and load is supported by a thin layer of compressed air supplied by a compressor. The supplied air must provide a force greater than the bearing and the load. The force required is calculated from the input pressure times the area. The force from the air film is calculated using Equation 7, and provides only 30 percent efficiency. [18]

$$Force = Pressure * Area * 0.3 \qquad (7).$$

6.2.2 Constraints, Goals, and Decision Making

The spherical air bearing has three requirements:

1. must provide a frictionless motion about 3 axes of rotation,
2. must support at least 38 pounds,
3. must be made of a non-ferrous material.

Additionally, four design goals were determined as follows:

1. the bearing must have a low moment of inertia,
2. cost less than $1000,
2. have a max operating pressure of 90 psi,
4. provide a range of motion of ±45˚.

The team proposed three design paths that can be seen in Figure 14.

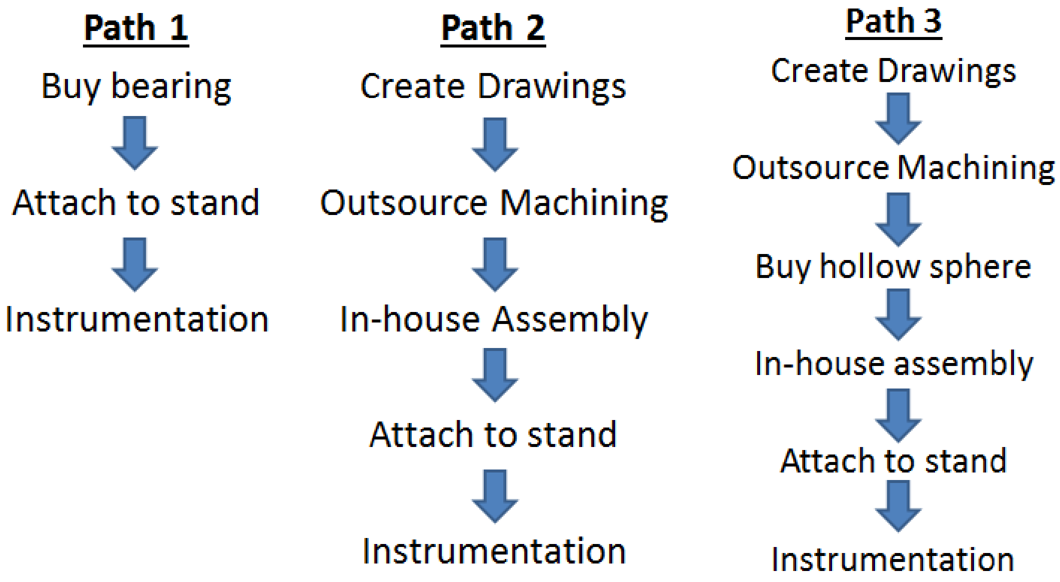


Figure 15: Spherical Air Bearing Design Paths

Path 1 involves buying a bearing directly from a supplier. Path 2 involves the team designing our own bearing and then outsourcing the machining. Path 3 offers a more unique solution to the problem; it offers the fabrication of a new concept, a large hollow sphere that encompasses the CubeSat. The hollow sphere sits within a bearing housing (cup half of the bearing) that provides full 360 degrees of friction-less rotation about all three axes. All three design paths were considered and pursued. The table below shows a comparison of the three paths for cost, ease of fabrication, creativity, and time. A rating from one to three is assigned for every category, one being the worst and three being the most advantageous. Each category was weighted to properly express the most important design factors. The maximum possible score is 300 points.

	Weight %	**Path 1**	**Path 2**	**Path 3**
Cost	30	1	3	2
Fabrication	20	3	2	1
Creativity	10	1	2	3
Time	15	3	1	2
Instrumentation	25	2	3	1
	totals	**195**	**240**	**165**

Table 4: Design Decision

Path 2 is considered the best option with a score of 240 points and is pursued. However, due to manufacturing restrictions, path 1 was ultimately chosen; the design of the bearing for path 2 is described in the following section.

6.2.3 Spherical Air Bearing Design

Based on the constraints, goals, and design concepts discussed in previous sections, a spherical air bearing was designed. The design is displayed in Appendix H. To reduce manufacturing and material costs, a small surface area bearing design was required. As a result, the bearing inlet pressure had to be increased in order to overcome inefficiencies. The designed bearing utilizes a single orifice for the air to flow through the bearing. This single orifice design is similar to that of other smaller spherical air bearings; it was selected due to its simplicity and ability to provide a steady laminar airflow. 6061 aircraft grade aluminum was used in this design because it meets our non-ferrous material requirement. Also, most importantly, 6061 is hard enough to maintain the precision tolerances the bearing requires. The most important of these tolerances is the RMS16 surface tolerance on the rounded faces the two halves of the bearing. This surface tolerance requires an average surface roughness variance of one micron. There are six threaded holes on both the cup and rotating hemisphere of the bearing; these holes are used to mount the cup to the test bed stand, and mount the tabletop to the rotating hemisphere of the bearing.

6.3 Air Filtration System

Surface tolerances and geometry are the most important aspects of our bearing design due to the thin film of air supporting the load and bearing. It is crucial that clean air is provided to the bearing. This clean air needs to be provided through a three stage filtration system.

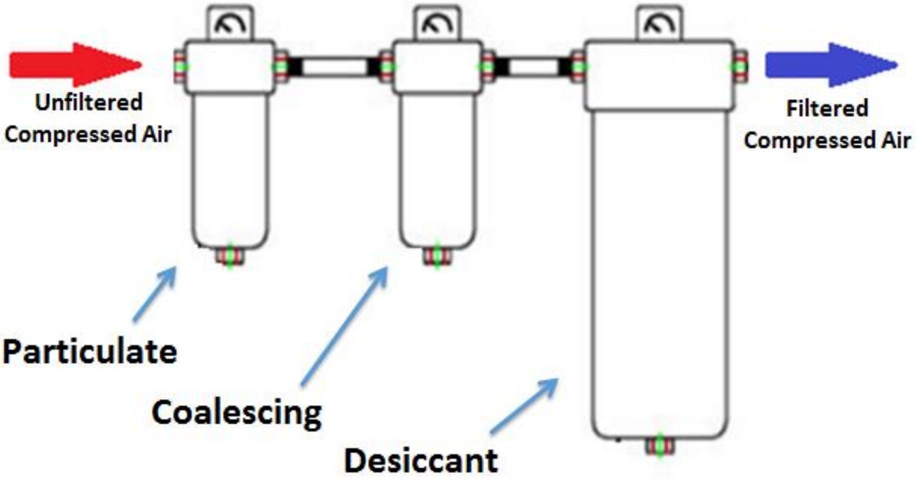


Figure 16: Three Stage Filtration System

Stage one is a particulate filter to remove any dirt and dust particles that may be in the air line. Stage two is a coalescing filter to remove any liquid oil or water that is in the air line. The third and final stage is a desiccant filter that removes any remaining water vapor from the air. It is critical that these three filters are in this order; if incorrect the filters themselves could be damaged, eventually shorting the life of the bearing. The ISO standards associated with the required air quality are below.

ISO8573-1	
Particles	Class 1
Humidity	Class 3
Oil	Class 1

Table 5: Supply Air Quality Standards

The particulate, coalescing, and desiccant filters selected were the Speedaire Intermediate Compressed Air 5-micron particulate filter (Grainger Item #4ZL38), the Speedaire one micron compressed air coalescing filter (Grainger Item #4GNN6), and the Speedaire Desiccant Air Dryer (Grainger Item #5VC89) respectively. These filters were selected because they meet or exceed the air quality requirements for other professionally manufactured spherical air bearings.

6.4 Test Bed

The test bed provides an adjustable leveling elevation system that places the CubeSat in the center of the uniform magnetic field. It is important that the center of mass of the system and the center of rotation are vertically aligned. This is achieved by strategically placing counterweights on or underneath the mounting tabletop of the spherical air bearing.

6.4.1 Constraints and Goals

The test bed design has three constraints:

1. must be made of a non-ferrous material
2. must provide a stable base
3. must be tall enough to place the CubeSat in the middle of the uniform magnetic field.

Additionally, the team set two design goals as follows:

1. stand must be portable so it could be moved by one person
2. must have an adjustable height and leveling system.

6.4.2 Test Bed Design

Once again 6061 aluminum was used in the fabrication of the test bed because it is a non-ferrous material and will not interfere with the magnetic field produced by the Helmholtz cage and it will help make the testbed weight less. The stand without the bearing sits at approximately 39.37”. When the mounting flange and bearing are mounted to the column the CubeSat will sit in the approximate center of the uniform magnetic field. In order to create a stable base a 24” diameter circle was utilized. In order to remove weight, material was removed from the circular base and the 4 supports that were TIG welded to the base and the column. Detailed drawings of the test bed base can be found in appendix I.

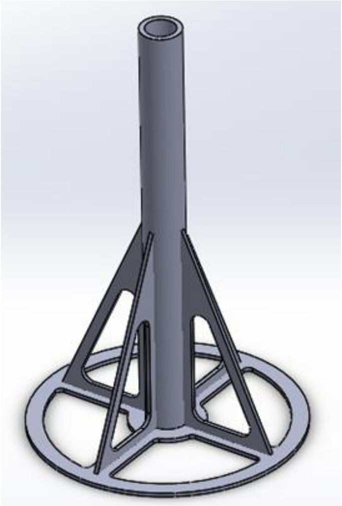


Figure 18: Test Stand Solid Works Model

Figure 17: Final Stand Design

6.5 Recommendations

Unfortunately, we were unable to manufacture a spherical air bearing since many of the local machine shops would have difficulty maintaining some of the tolerances that were required for the bearing; specifically, the precision RMS16 surface finish tolerance. Even if the local machinists could maintain the tolerance there is no way that a company would donate the amount of machine time required to reach such a level of precision. The possibility of purchasing a bearing and filtration system was out of reach due to the limited budget for this project. Until more funds are available it is recommended that the WALI team use a simple string system for their testing. This string system is simple; the CubeSat is hung from a string inside the Helmholtz cage and a motion is imparted upon it when testing as shown in Figure 18.

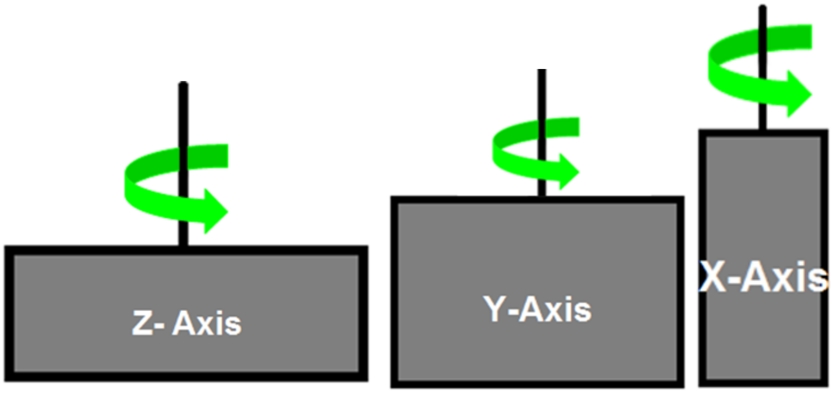


Figure 19: String System

The biggest downfall of this design is that a single axis of rotation can be tested at a time and the satellite must be manually manipulated to change which axis is being tested.

When more funding is available to purchase a spherical air bearing and filtration system it is recommended that they purchase a Nelson A-653 spherical air bearing, this bearing meets and exceeds all of the requirements for this CubeSat and any other CubeSat that WALI may create while remaining cost effective. The air filters recommended earlier meet or exceed the requirements for the Nelson bearing that is currently recommended.

7 Conclusion

A functioning CubeSat attitude testing system has been developed. The Helmholtz cage produces a uniform magnetic field with initial testing showing uniformity better than +/-.03 Gauss per decimeter. A stand has been fabricated and prepped for the installation of a spherical air bearing. A Nelson A-653 air bearing has been recommended along with a fully designed air filtration system. The test bed top plate is fabricated and prepared for the CubeSat and mass balancing system for accurate testing. Utilizing the recommended string system, the test environment is currently ready for testing. Optimal testing conditions can be provided by the aforementioned spherical air bearing and air filtration system recommendations.

Appendix A: MATLAB Code for Coil Design

```
clear all clc;

mu      = 4*pi*10^-7;    %Permeability of Free Space
L       = 2.3368;           %Length of Coil Side
N       = 1:1:100;             %Number of Turns in Coil
z       = 0;            %Distance from Coil
d       = .5445*L           %Distance between Coils (d = .5445L for optimal homogeneity)


filename    = 'WireParameters';
sheet       = 1;
xlRange     = 'A1:A10';
G           = xlsread(filename,sheet,xlRange);

xlRange     = 'B1:B10';
R           = xlsread(filename,sheet,xlRange); %Import resistance

xlRange     = 'C1:C10';
I          = xlsread(filename,sheet,xlRange); %Import 90% Max Current

Title    = ['G(i)    R(i)    I(i)    Gauss     Volts   Loops   Length(m)    Length(ft)'];
disp(Title)

for i = 1:6

    a       = L/2;
    x       = 0;
    y       = 0;
    z       = 0;

    k1      = z^2 + x^2 + (y-a)^2;
    k2      = z^2 + x^2 + (y+a)^2;
    k3      = z^2 + y^2 + (x-a)^2;
    k4      = z^2 + y^2 + (x+a)^2;

    Bz1     = (1./sqrt(k1)).*(((2*(x+a))./(sqrt(k1+(x+a)^2)))-((2*(x-a))./(sqrt(k1+(x-a)^2))));
    Bz2     = (1./sqrt(k2)).*(((2*(x+a))./(sqrt(k2+(x+a)^2)))-((2*(x-a))./(sqrt(k2+(x-a)^2))));
    Bz3     = (1./sqrt(k3)).*(((2*(y+a))./(sqrt(k3+(y+a)^2)))-((2*(y-a))./(sqrt(k3+(y-a)^2))));
    Bz4     = (1./sqrt(k4)).*(((2*(y+a))./(sqrt(k4+(y+a)^2)))-((2*(y-a))./(sqrt(k4+(y-a)^2))));

    N   = 0;
    Gaussz  = 0;
        while Gaussz < 3
            N       = 1 + N;
            Bz      = (mu*N*I(i)/(4*pi))*(Bz1 + Bz2 + Bz3 + Bz4);
            Gaussz  = (Bz)./(10^-4);
        end

    Loops   = N;
    Length  = L*4*N;
    Volts   = I(i)/(L*4*N.*R(i));


    Array   = [G(i), R(i), I(i), Gaussz, Volts, Loops, Length];
    formatSpec  = '%4.2f     %4.2f    %4.2f    %4.2f    %4.2f    %4.2f    %4.2f\n';
```

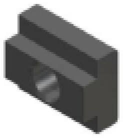
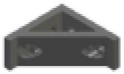
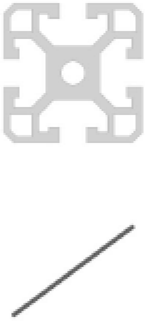
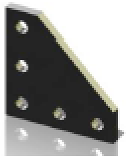
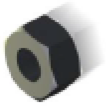
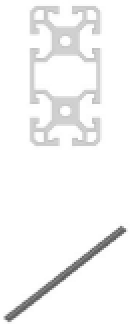
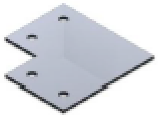
```
    fprintf(formatSpec,Array);

end
```

Appendix B: Design Matrix

Coil Optimization Matrix					
	Loop Side Length (1.9m) (d=1.0345)				
AWG	Diameter (mm)	Ohms/km	Max Current(A)	Loops Required	Wire Length (m)
4	5.189	0.8152	60	5	38
5	4.621	1.028	47	6	45.6
6	4.115	1.296	37	8	60.8
7	3.665	1.634	30	10	76
8	3.264	2.061	24	12	91.2
9	2.906	2.599	19	15	114
10	2.588	3.277	15	19	144.4
11	2.305	4.132	12	24	182.4
12	2.053	5.211	9.3	31	235.6
14	1.628	8.286	5.9	48	364.8
	Loop Side Length (1.8m) (d=0.9801)				
AWG	Diameter (mm)	Ohms/km	Max Current(A)	Loops Required	Wire Length (m)
4	5.189	0.8152	60	5	36
5	4.621	1.028	47	6	43.2
6	4.115	1.296	37	8	57.6
7	3.665	1.634	30	9	64.8
8	3.264	2.061	24	12	86.4
9	2.906	2.599	19	14	100.8
10	2.588	3.277	15	18	129.6
11	2.305	4.132	12	23	165.6
12	2.053	5.211	9.3	29	208.8
14	1.628	8.286	5.9	45	324
	Loop Side Length (1.7m) (d=0.9256)				
AWG	Diameter (mm)	Ohms/km	Max Current(A)	Loops Required	Wire Length (m)
4	5.189	0.8152	60	5	43
5	4.621	1.028	47	6	40
6	4.115	1.296	37	7	47.6
7	3.665	1.634	30	9	61.2
8	3.264	2.061	24	11	74.8
9	2.906	2.599	19	14	95.2
10	2.588	3.277	15	17	115.6
11	2.305	4.132	12	21	142.8
12	2.053	5.211	9.3	27	183.6
14	1.628	8.286	5.9	43	292.4

Appendix C : Frame Components and Assembly Instructions

Component Listing:

	Caster (Parker IPS)		T Slot Nut HD (Parker IPS)
	Gusset Bracket (Parker IPS)		40 Series 40 (Parker IPS)
	Joining Plate Corner (16) (Parker IPS)		
	M8 Hex Nut (Parker IPS)		40 Series 80 (Parker IPS)
	M8x16 Screw (Parker IPS)		
	M8x18 Screw (Parker IPS)		Horizontal Connection (Custom)

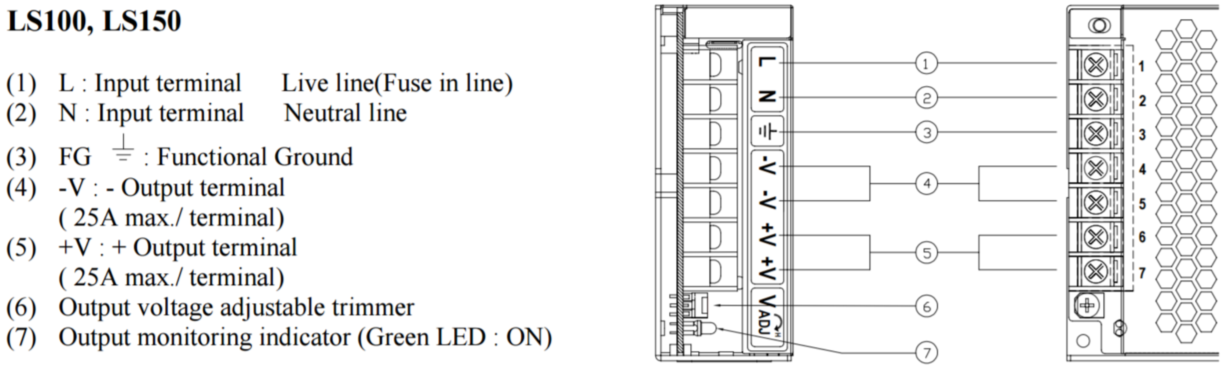
For assembly instructions, please reference Aero 405 Final Report from University of Michigan – Reference 17.

Appendix D: Helmholtz Cage Bill of Materials

Assembly Name		Helmholtz Cage				
Assembly Revision		A				
Approval Date		11/6/2016				
Part Number	**Part Name**	**Qty**	**Unit of Measure**	**Unit $**	**Total $**	**BOM Notes**
100	Helmholtz Frame	1	Each	$ -	$ -	*Portion of frame donated*
101	T-Slot Alum 1.5"x3.0"x76.25"	4	Each	$ -	$ -	
102	T-Slot Alum 1.5"x3.0"x78.75"	4	Each	$ -	$ -	
103	T-Slot Alum 1.5"x1.5"x78.75"	6	Each	$ -	$ -	
104	T-Slot Alum 1.5"x1.5"x76.25"	2	Each	$ -	$ -	
105	T-Slot Frame 45/45 Corners	40	Each	$ -	$ -	
106	T-Slot Frame Flat Corners	16	Each	$ -	$ -	
107	Coil Support Hangers	8	Each	$ -	$ -	
108	T-Slot Nuts	162	Each	$ -	$ -	
109	T-Slot Bolts	162	Each	$ -	$ -	
110	Washers	48	Each	$ -	$ -	
111	Coil Support Plates	8	Each	$ -	$ -	
112	Coil Hangers	4	Each	$ -	$ -	
200	Coil Assemblies					
201	1.9m Coil Square	2	Each	$ -	$ -	
202	C-Channel, 1.9m	4	Each	$ 8.59	$ 68.72	*C1118 Channel 1 x 1 x 1/8 x 74.8031"*
203	Corner Fixture	4	Each	$ 2.72	$ 10.88	*Aluminum Gussets 1/4 x 6 x 6"*
204	1.8m Coil Square	2	Each	$ -	$ -	
205	C-Channel, 1.8m	4	Each	$ 8.19	$ 65.52	*C1118 Channel 1 x 1 x 1/8 x 70.8661"*
206	Corner Fixture	4	Each	$ 2.72	$ 10.88	*Aluminum Gussets 1/4 x 6 x 6"*
207	1.7m Coil Square	2	Each	$ -	$ -	
208	C-Channel, 1.7m	4	Each	$ 7.79	$ 62.32	*C1118 Channel 1 x 3/8 x 1/8 x 66.9291"*
209	Corner Fixture	4	Each	$ 2.72	$ 10.88	*Aluminum Gussets 1/4 x 6 x 6"*
210	Copper Magnet Wire	89	lbs	$ 4.98	$ 443.14	*89lb Copper Spool*
211	Nylon Bolts	72	Each	$ -	$ -	
300	Coil Power Supply					
301	Fuse 12.5A	12	Each	$ 3.95	$ 47.34	*12.5A 500VAC 3AB 3AG*
302	Fuse Holder	6	Each	$ 1.66	$ 9.96	*FUSE HLDR CART 250V 6.3A PNL MNT*
303	Arduino	1	Each	$ 24.95	$ 24.95	*Microcontroller*
304	AC/DC Converter	6	Each	$ 41.09	$ 246.54	*12V 150W*
305	H-Bridge	6	Each	$ 34.95	$ 209.70	*5.5 to 16V, Continous 14A*
306	Connector Bridges	2	Each	$ 8.49	$ 16.98	*Bridges coils to H-bridges*
307	PC Power Supply	1	Each	$ -	$ -	*Recycled PC power supply*

Appendix E: Power System Schematics

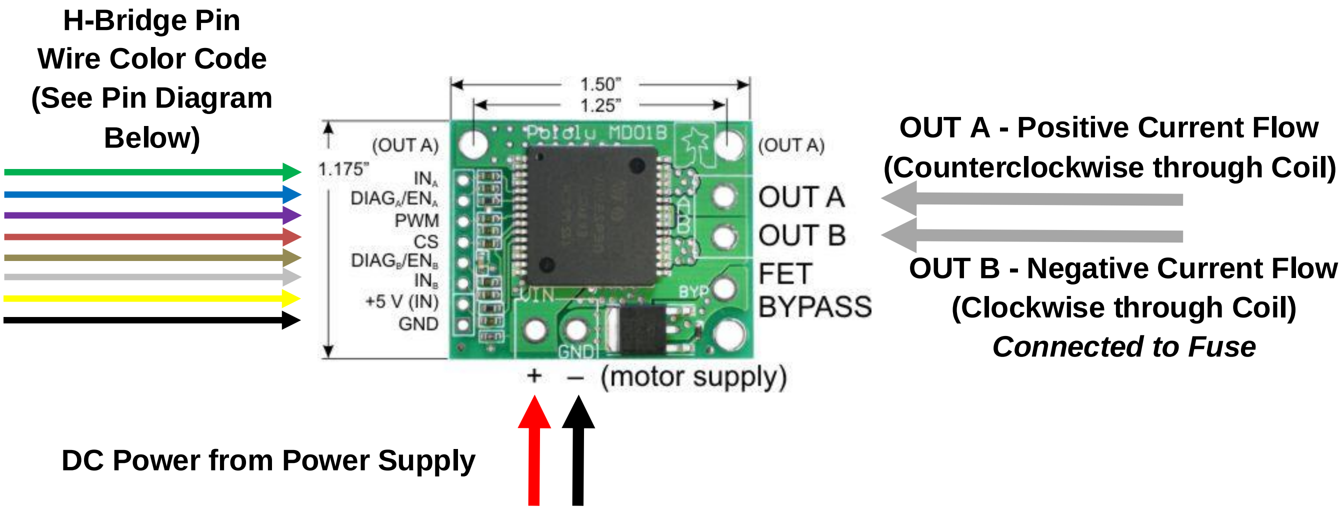
Power Supply Schematic – TDK Lambda LS 25-150 Series



LS100, LS150

(1) L : Input terminal Live line(Fuse in line)
(2) N : Input terminal Neutral line
(3) FG ⏚ : Functional Ground
(4) -V : - Output terminal
(25A max./ terminal)
(5) +V : + Output terminal
(25A max./ terminal)
(6) Output voltage adjustable trimmer
(7) Output monitoring indicator (Green LED : ON)

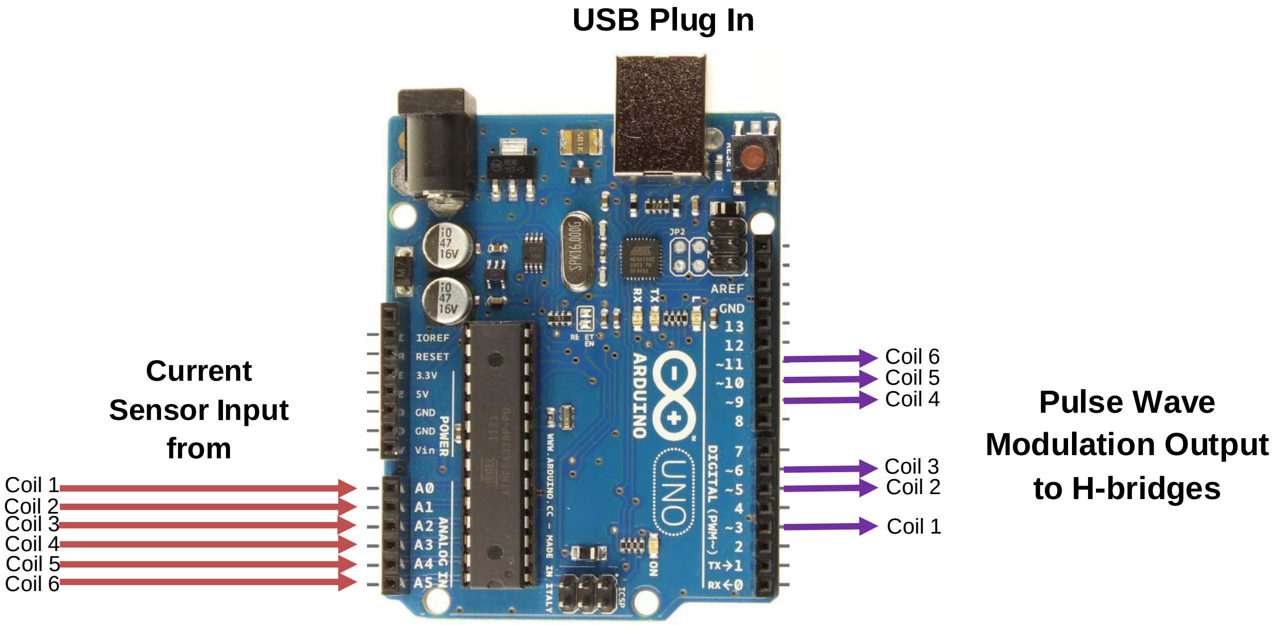
H-bridge Schematic - VNH2SP30 Motor Driver Carrier MD01B



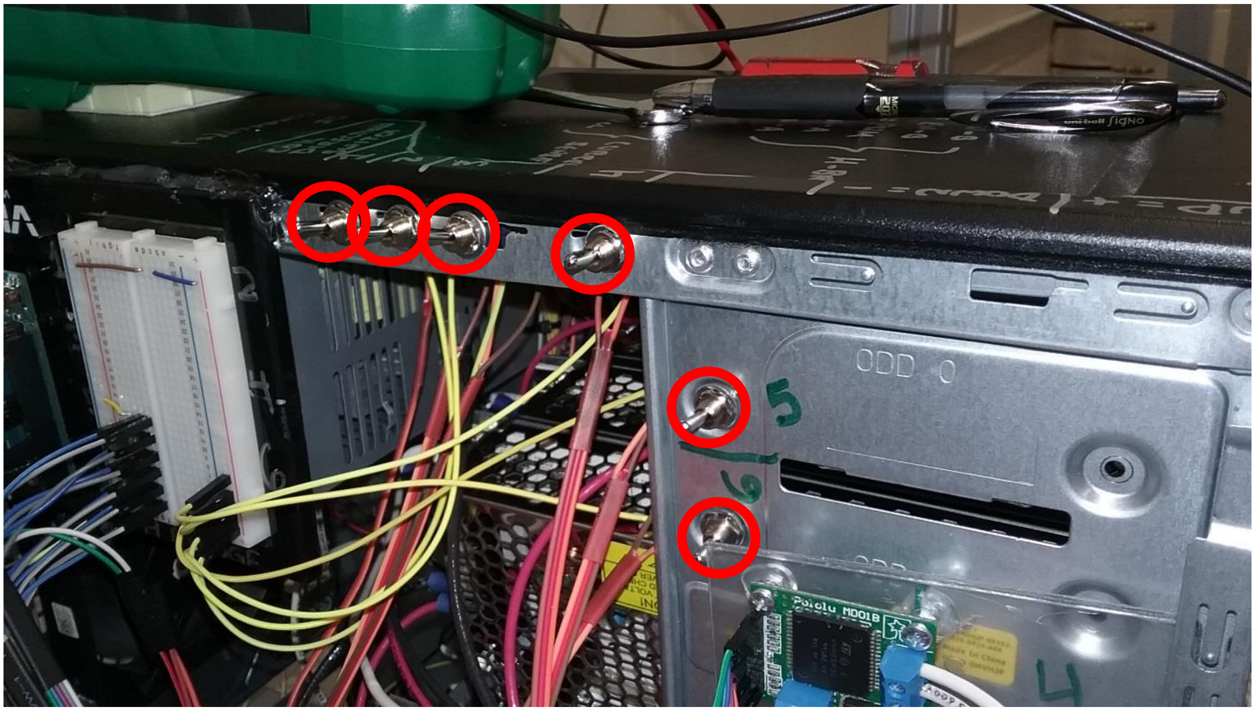
H-Bridge Pin Out Descriptions

IN_A	Current flow direction, counterclockwise if connected to 5V
$\mathrm{DIAG}_A/\mathrm{EN}_A$	Must be connected to 5V for operation
PWM	Pulse Wave Modulation signal from Arduino
CS	Current Sensor (130mV/A) signal output from H-bridge, read by Arduino
$\mathrm{DIAG}_B/\mathrm{EN}_B$	Must be connected to 5V for operation
IN_B	Current flow direction, clockwise if connected to 5V
+V (IN)	H-bridge closed circuit voltage, connected to 12V from PC power supply
GND	H-bridge closed circuit ground, connected to GND from PC power supply

Control System – Arduino Uno R3



Control System – Magnetic Field Direction Switches



Switch in the "up" position will operate the coils in the positive direction.
Switch in the "down" position will operate the coils in the negative direction.

Appendix F: Arduino Control System C++ Software

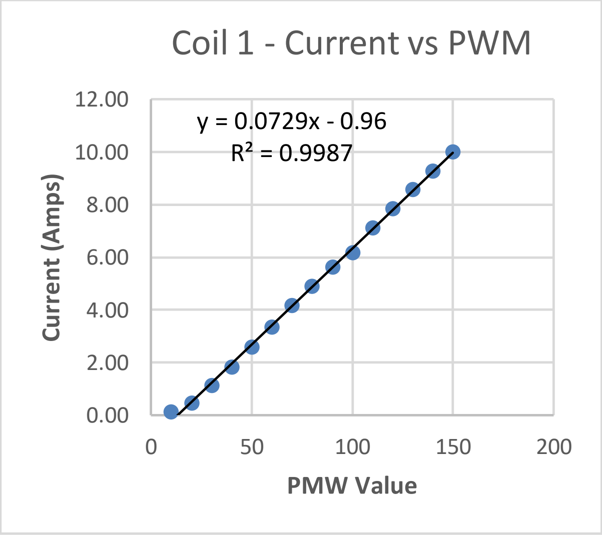
```
/////CURRENT CONTROL TO COILS/////
double Z1_Current = 6.5;                    //Coil 1 current control +/- 50mA
double Z2_Current = 6.5;                    //Coil 2 current control +/- 50mA
double Y3_Current = 6.5;                    //Coil 3 current control +/- 50mA
double Y4_Current = 6.5;                    //Coil 4 current control +/- 50mA
double X5_Current = 6.5;                    //Coil 5 current control +/- 50mA
double X6_Current = 6.5;                    //Coil 6 current control +/- 50mA

int Z1_PWM_val = 0;                    //Coil 1 PWM
int Z2_PWM_val = 0;                    //Coil 2 PWM
int Y3_PWM_val = 0;                    //Coil 3 PWM
int Y4_PWM_val = 0;                    //Coil 4 PWM
int X5_PWM_val = 0;                    //Coil 5 PWM
int X6_PWM_val = 0;                    //Coil 6 PWM

/////DECLARE PIN I/O FOR COIL 1/////
int Z1_AnalogSensorPin = 0;
int Z1_val = 0;                         // Variable used to store "current read" value
int Z1_PWMpin = 3;                          // Pulse Wave Modulation pin
int Z1_readings[numReadings];
double Z1_total = 0;
double Z1_average = 0;
/////DECLARE PIN I/O FOR COIL 2/////
int Z2_AnalogSensorPin = 1;
int Z2_val = 0;                         // Variable used to store "current read" value
int Z2_PWMpin = 5;                          // Pulse Wave Modulation
int Z2_readings[numReadings];
double Z2_total = 0;
double Z2_average = 0;
/////DECLARE PIN I/O FOR COIL 3/////
int Y3_AnalogSensorPin = 2;
int Y3_val = 0;                         // Variable used to store "current read" value
int Y3_PWMpin = 6;                          // Pulse Wave Modulation pin
/////DECLARE PIN I/O FOR COIL 4/////
int Y4_AnalogSensorPin = 3;
int Y4_val = 0;                         // Variable used to store "current read" value
int Y4_PWMpin = 9;                          // Pulse Wave Modulation pin
/////DECLARE PIN I/O FOR COIL 5/////
int X5_AnalogSensorPin = 4;
int X5_val = 0;                         // Variable used to store "current read" value
int X5_PWMpin = 10;                         // Pulse Wave Modulation pin
/////DECLARE PIN I/O FOR COIL 6/////
int X6_AnalogSensorPin = 5;
int X6_val = 0;                         // Variable used to store "current read" value
int X6_PWMpin = 11;                         // Pulse Wave Modulation pin

/////DECLARE VARIABLES FOR CURRENT SENSOR////////////////////////////////////////////////////
int Z1_mVperAmp = 130;                      // use 100 for 20A Module and 66 for 30A Module
int Z1_ACSoffset = 0;
double Z1_Voltage = 0;
double Z1_Amps = 0;

int Z2_mVperAmp = 130;                      // use 100 for 20A Module and 66 for 30A Module
int Z2_ACSoffset = 0;
```

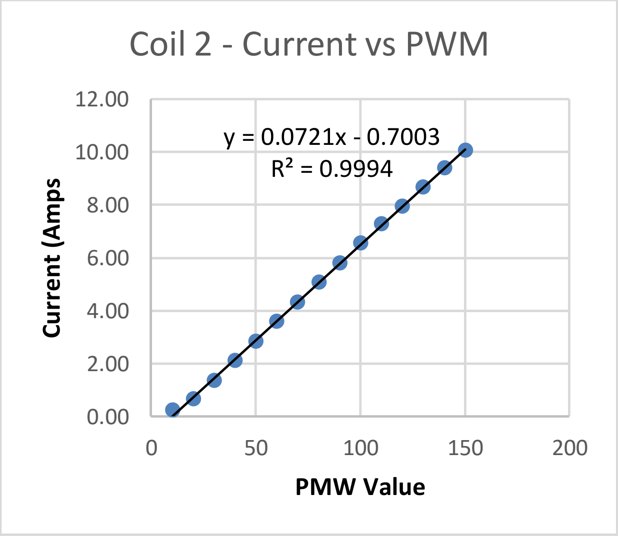
```
double Z2_Voltage = 0;
double Z2_Amps = 0;

int Y3_mVperAmp = 130;                    // use 100 for 20A Module and 66 for 30A Module
int Y3_ACSoffset = 0;
double Y3_Voltage = 0;
double Y3_Amps = 0;

int Y4_mVperAmp = 130;                    // use 100 for 20A Module and 66 for 30A Module
int Y4_ACSoffset = 0;
double Y4_Voltage = 0;
double Y4_Amps = 0;

int X5_mVperAmp = 130;                    // use 100 for 20A Module and 66 for 30A Module
int X5_ACSoffset = 0;
double X5_Voltage = 0;
double X5_Amps = 0;

int X6_mVperAmp = 130;                    // use 100 for 20A Module and 66 for 30A Module
int X6_ACSoffset = 0;
double X6_Voltage = 0;
double X6_Amps = 0;

void setup() {                   // this is a command for the Arduino to set up the pins for use
  Serial.begin(9600);              // Sends read information to the 9600 BAUD syntax readout -- click magnifying
glass in top right corner

  /////COIL 1 OUTPUT/////
  pinMode(Z1_PWMpin, OUTPUT);
  /////COIL 2 OUTPUT/////
  pinMode(Z2_PWMpin, OUTPUT);
  /////COIL 3 OUTPUT/////
  pinMode(Y3_PWMpin, OUTPUT);
  /////COIL 4 OUTPUT/////
  pinMode(Y4_PWMpin, OUTPUT);
  /////COIL 5 OUTPUT/////
  pinMode(X5_PWMpin, OUTPUT);
  /////COIL 6 OUTPUT/////
  pinMode(X6_PWMpin, OUTPUT);
  }

}

void loop(){

  int Z1_PWM_val = (.9600+Z1_Current)/.0745;          //Coil 1 current control
  int Z2_PWM_val = (.7003+Z2_Current)/.0729;          //Coil 2 current control
  int Y3_PWM_val = (.8024+Y3_Current)/.0775;          //Coil 3 current control
  int Y4_PWM_val = (.8337+Y4_Current)/.0765;          //Coil 4 current control
  int X5_PWM_val = (.8337+X5_Current)/.0760;          //Coil 5 current control
  int X6_PWM_val = (.8337+X6_Current)/.0760;          //Coil 6 current control

  if (Z1_PWM_val > 170 || Z2_PWM_val > 170 || Y3_PWM_val > 170 || Y4_PWM_val > 170 || X5_PWM_val >
170 || X6_PWM_val > 170) {
    Serial.println();
    Serial.print("PLEASE LOWER CURRENT VALUE!");
```

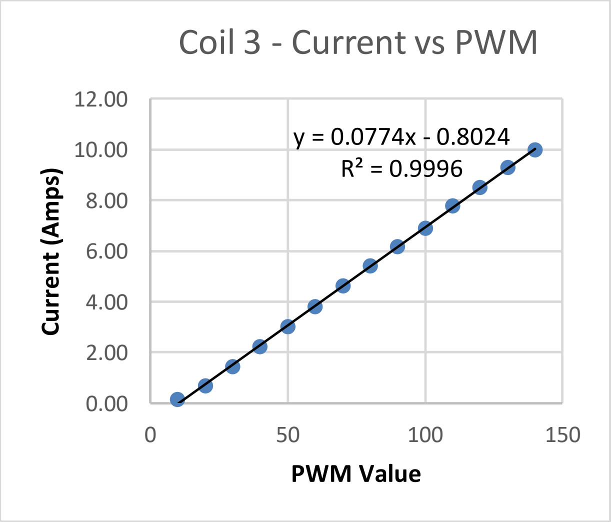
```
    delay(1000);
    Serial.println();
    Serial.print("DUDE, YOU NEED TO LOWER CURRENT VALUE!");
    delay(1000);
    Serial.println();
    Serial.print("...");
    delay(2000);
    Serial.println();
    Serial.print("BUT SERIOUSLY, LOWER THE CURRENT VALUE!");
    while(1) { }
  }
/////COIL 1 PIN OUTPUT CONTROL/////
  analogWrite(Z1_PWMpin,Z1_PWM_val);                //outputs to the PWMpin
/////COIL 2 PIN OUTPUT CONTROL/////
  analogWrite(Z2_PWMpin,Z2_PWM_val);                //outputs to the PWMpin
/////COIL 3 PIN OUTPUT CONTROL/////
  analogWrite(Y3_PWMpin,Y3_PWM_val);                //outputs to the PWMpin
/////COIL 4 PIN OUTPUT CONTROL/////
  analogWrite(Y4_PWMpin,Y4_PWM_val);                //outputs to the PWMpin
/////COIL 5 PIN OUTPUT CONTROL/////
  analogWrite(X5_PWMpin,X5_PWM_val);                //outputs to the PWMpin
/////COIL 6 PIN OUTPUT CONTROL/////
  analogWrite(X6_PWMpin,X6_PWM_val);                //outputs to the PWMpin

//////////////////////////////////////////////////////////////////////////////////////////////////////////
/////COIL 1 SENSOR CALCULATION/////
  Z1_val = analogRead(Z1_AnalogSensorPin);          // value being read from current sensor
  Z1_Voltage = (Z1_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  Z1_Amps = ((Z1_Voltage - Z1_ACSoffset)/Z1_mVperAmp);  // voltage value converted to amps

/////COIL 2 SENSOR CALCULATION/////
  Z2_val = analogRead(Z2_AnalogSensorPin);          // value being read from current sensor
  Z2_Voltage = (Z2_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  Z2_Amps = ((Z2_Voltage - Z2_ACSoffset)/Z2_mVperAmp);  // voltage value converted to amps

/////COIL 3 SENSOR CALCULATION/////
  Y3_val = analogRead(Y3_AnalogSensorPin);          // value being read from current sensor
  Y3_Voltage = (Y3_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  Y3_Amps = ((Y3_Voltage - Y3_ACSoffset)/Y3_mVperAmp);  // voltage value converted to amps

/////COIL 4 SENSOR CALCULATION/////
  Y4_val = analogRead(Y4_AnalogSensorPin);          // value being read from current sensor
  Y4_Voltage = (Y4_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  Y4_Amps = ((Y4_Voltage - Y4_ACSoffset)/Y4_mVperAmp);  // voltage value converted to amps

/////COIL 5 SENSOR CALCULATION/////
  X5_val = analogRead(X5_AnalogSensorPin);          // value being read from current sensor
  X5_Voltage = (X5_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  X5_Amps = ((X5_Voltage - X5_ACSoffset)/X5_mVperAmp);  // voltage value converted to amps

/////COIL 6 SENSOR CALCULATION/////
  X6_val = analogRead(X6_AnalogSensorPin);          // value being read from current sensor
  X6_Voltage = (X6_val/1024.0) * 5000;              // analog reading (0-1024) converted to voltage value
  X6_Amps = ((X6_Voltage - X6_ACSoffset)/X6_mVperAmp);  // voltage value converted to amps
```

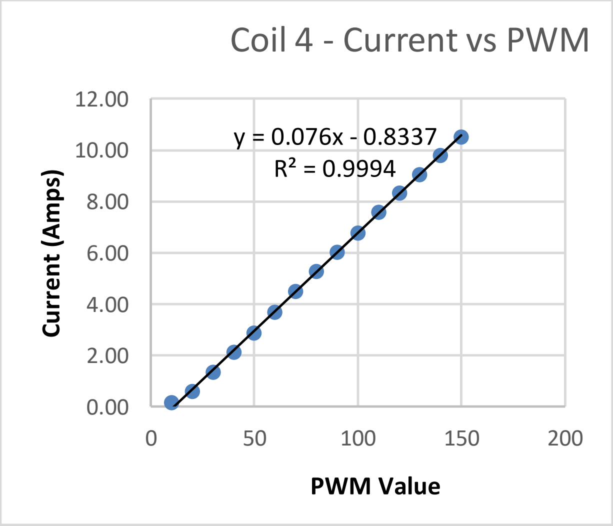
```
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
/////PRINT OUTPUT/////
  Serial.println();
  Serial.print("Sensor 1 (amps) = ");
  Serial.print(Z1_average);
  Serial.print("  Sensor 2 (amps) = ");
  Serial.print(Z2_Amps,3);
  Serial.print("  Sensor 3 (amps) = ");
  Serial.print(Y3_Amps,3);
  Serial.print("  Sensor 4 (amps) = ");
  Serial.print(Y4_Amps,3);
  Serial.print("  Sensor 5 (amps) = ");
  Serial.print(X5_Amps,3);
  Serial.print("  Sensor 6 (amps) = ");
  Serial.print(X6_Amps,3);

  delay(100); //delay between reads
}
```

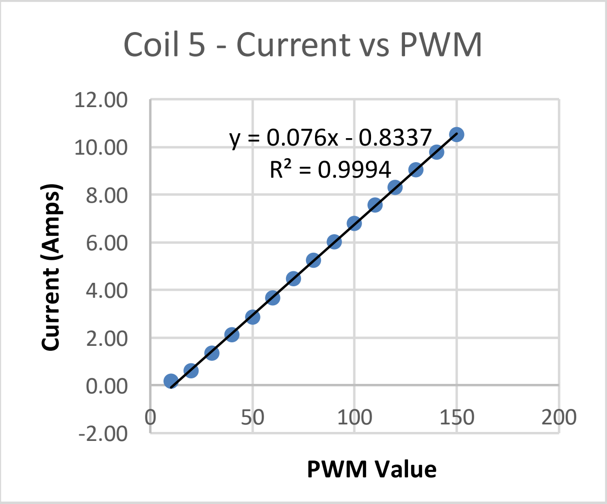
Appendix G: PWM vs Current Coil Value

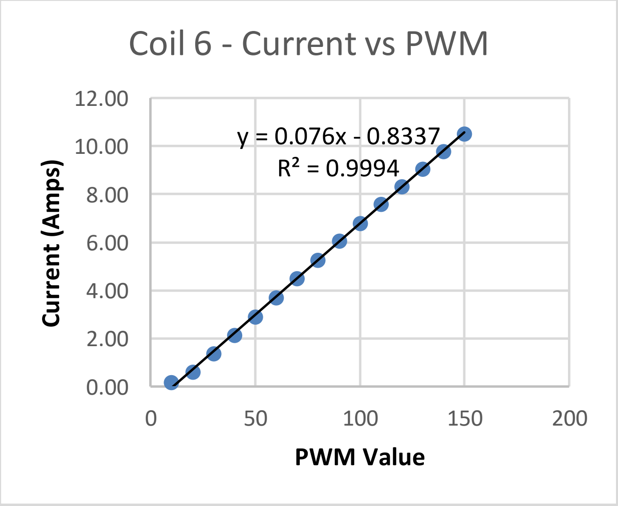




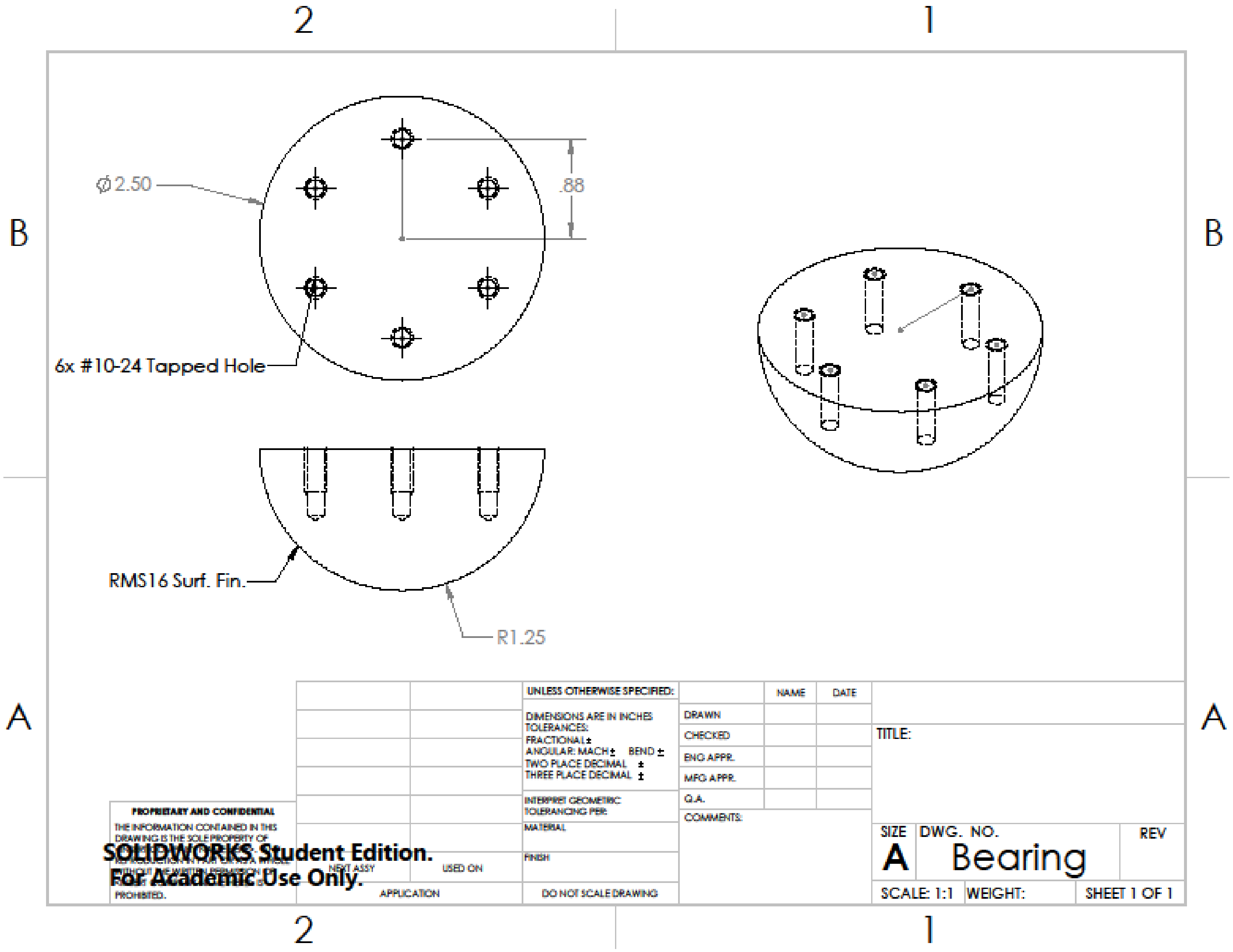


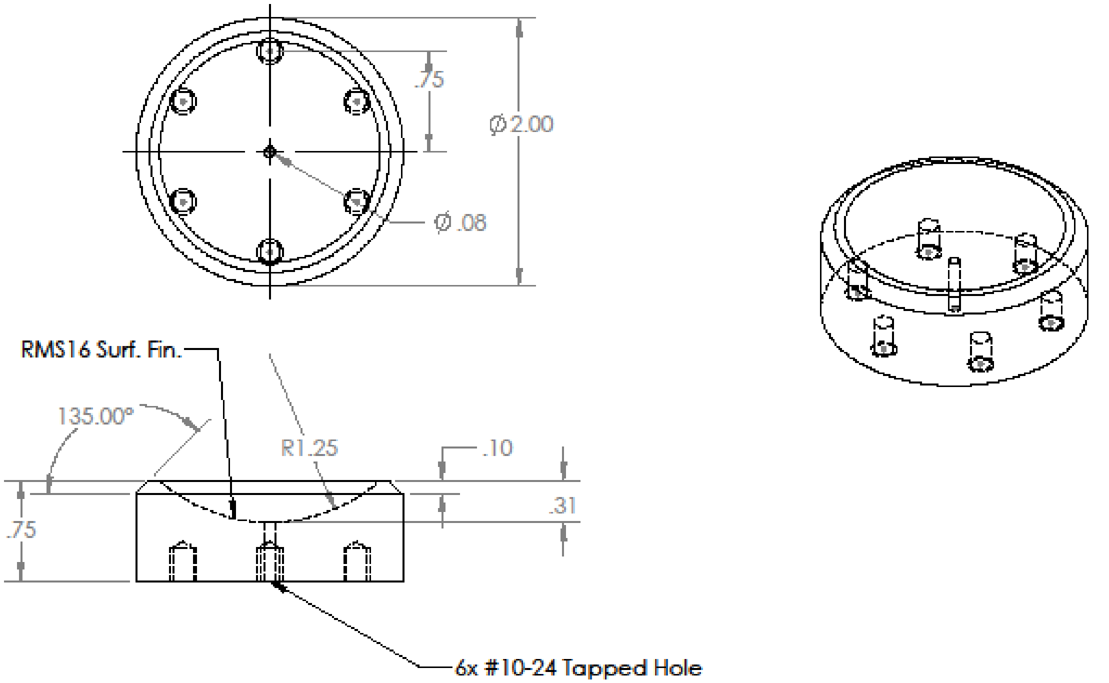






Appendix H: Spherical Air Bearing Technical Drawings





	NAME	DATE
DRAWN		
CHECKED		
ENG APPR.		
MFG APPR.		
Q.A.		
COMMENTS:		

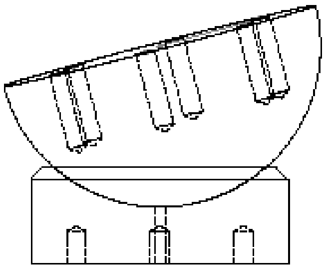
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±

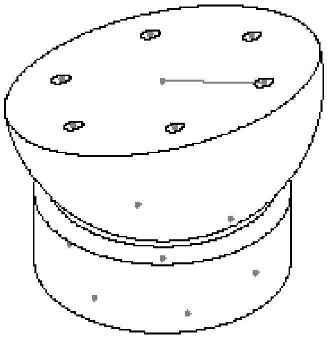
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING

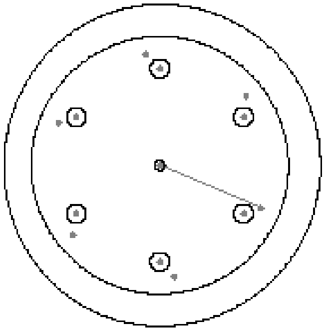
NEXT ASSY
USED ON
APPLICATION

TITLE:
SIZE A
DWG. NO. Cup
REV
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1

B







B

A

		UNLESS OTHERWISE SPECIFIED:		NAME	DATE
		DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN		
			CHECKED		
			ENG APPR.		
			MFG APPR.		
		INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
		MATERIAL	COMMENTS:		
NEXT ASSY	USED ON	FINISH			
APPLICATION		DO NOT SCALE DRAWING			

TITLE:

SIZE	DWG. NO.	REV
A	bearing assem	

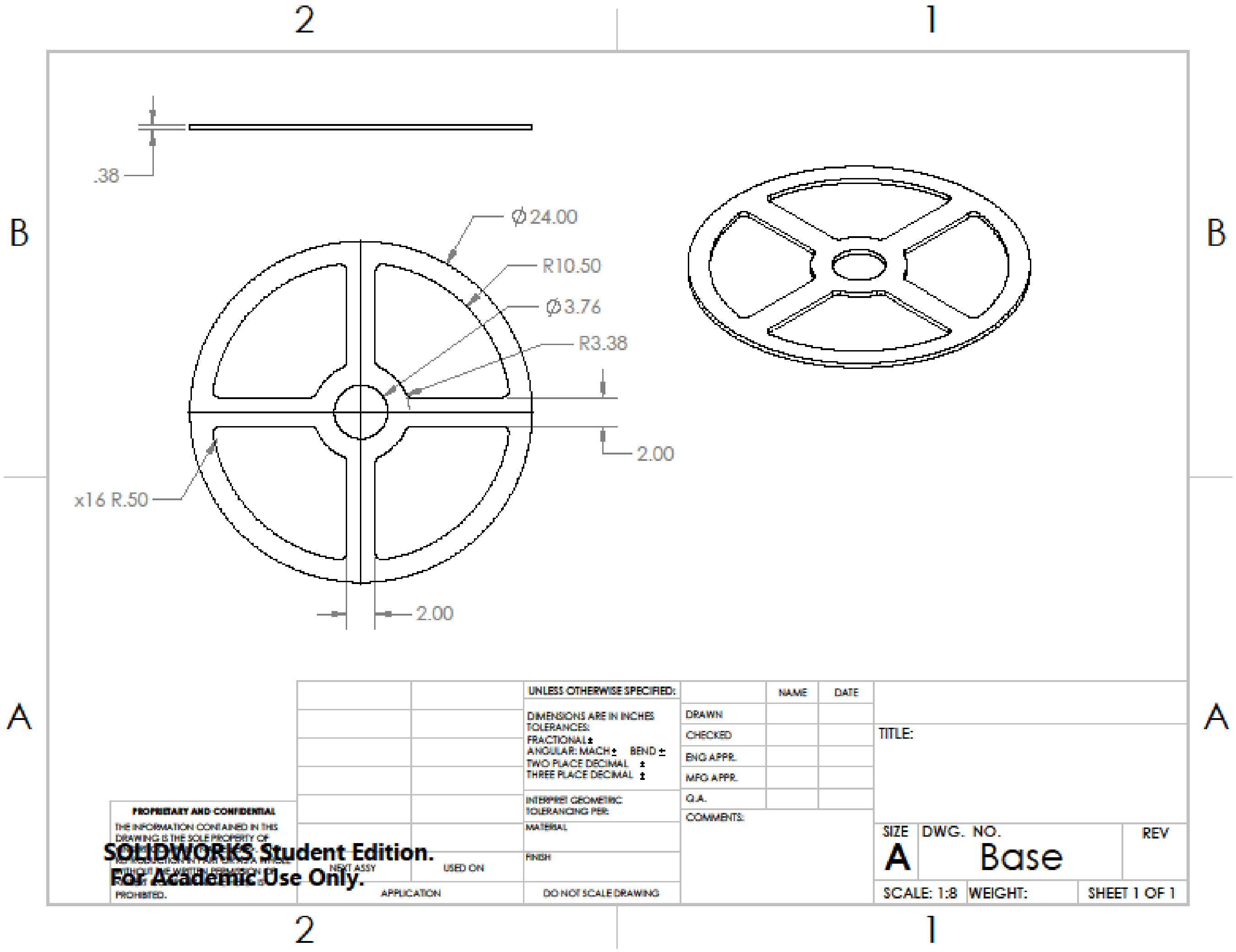
SCALE: 1:2 WEIGHT: SHEET 1 OF 1

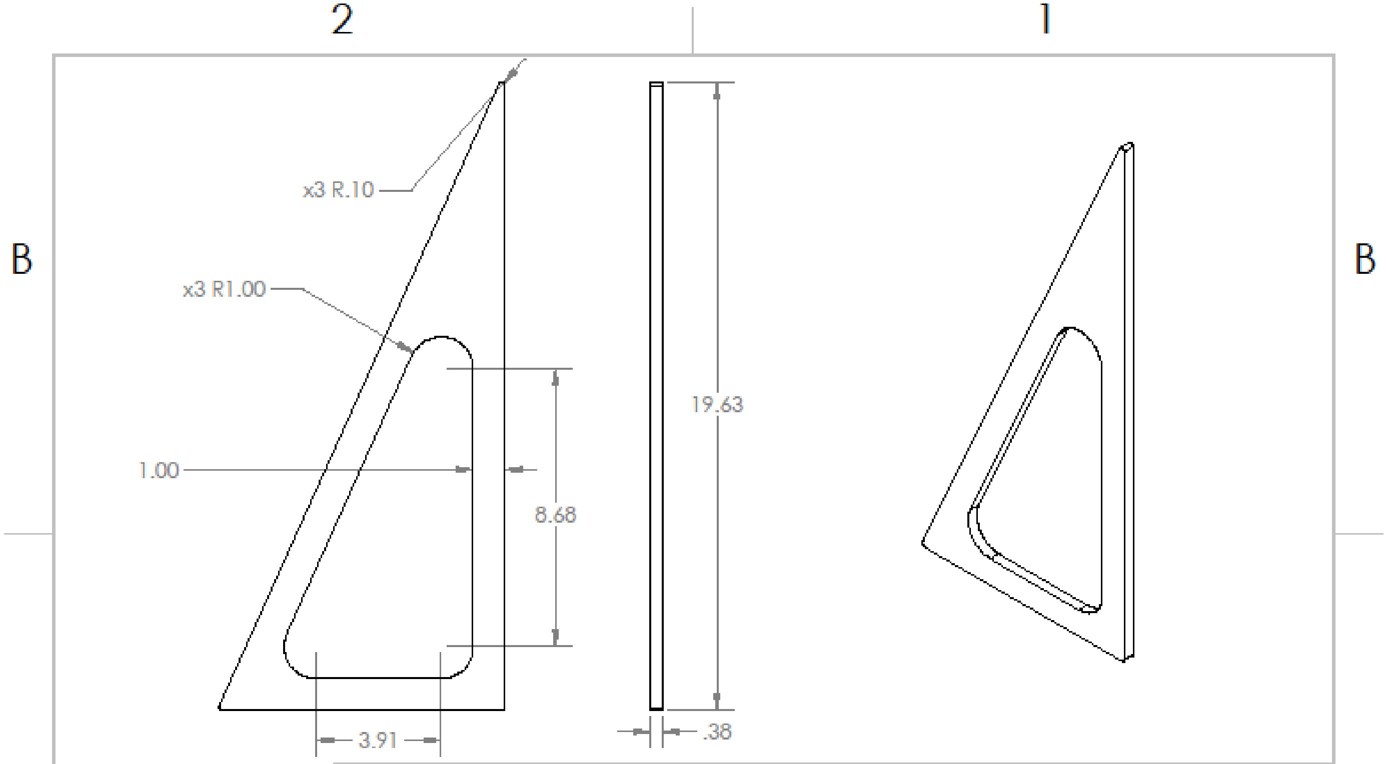
PROPRIETARY AND CONFIDENTIAL

THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF [illegible] REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF [illegible] PROHIBITED.

A

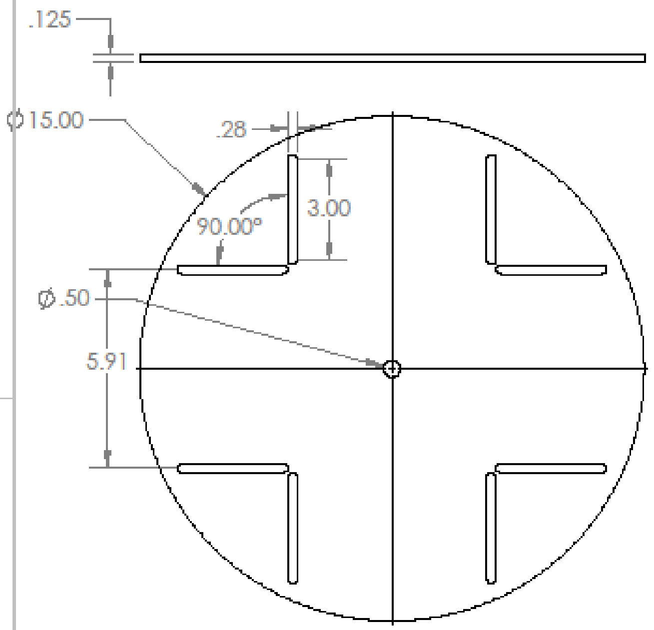
Appendix I: Test Bed Technical Drawings

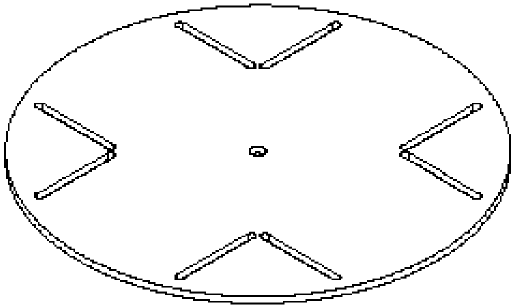




		UNLESS OTHERWISE SPECIFIED:		NAME	DATE	
		DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL± ANGULAR: MACH± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN			TITLE:
			CHECKED			
			ENG APPR.			
			MFG APPR.			
		INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.			
		MATERIAL	COMMENTS:			SIZE **A** DWG. NO. Support REV
NEXT ASSY	USED ON	FINISH				
APPLICATION		DO NOT SCALE DRAWING				SCALE: 1:8 WEIGHT: SHEET 1 OF 1

2 1 A





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±

INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING

	NAME	DATE
DRAWN		
CHECKED		
ENG APPR.		
MFG APPR.		
Q.A.		
COMMENTS:		

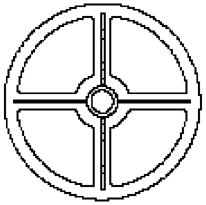
NEXT ASSY
USED ON
APPLICATION

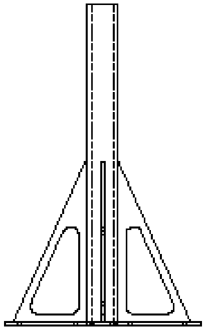
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF [illegible] REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF [illegible] PROHIBITED.

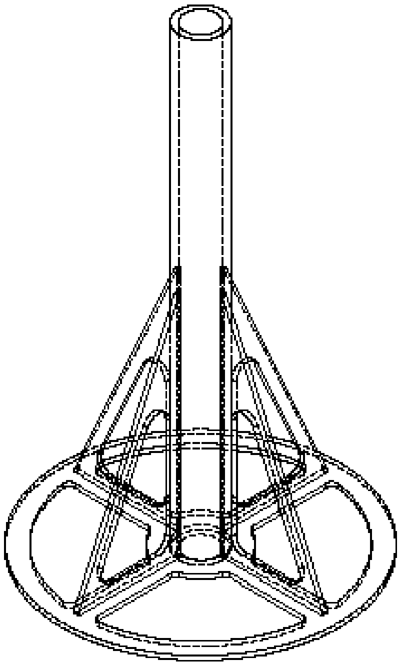
TITLE:

SIZE A
DWG. NO. Top plate
REV

SCALE: 1:8 WEIGHT: SHEET 1 OF 1







PROPRIETARY AND CONFIDENTIAL

THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF … REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF … PROHIBITED.

NEXT ASSY | USED ON

APPLICATION

UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±

INTERPRET GEOMETRIC TOLERANCING PER:

MATERIAL

FINISH

DO NOT SCALE DRAWING

	NAME	DATE
DRAWN		
CHECKED		
ENG APPR.		
MFG APPR.		
Q.A.		

COMMENTS:

TITLE:

SIZE	DWG. NO.	REV
A	Base Assembly	

SCALE: 1:16 | WEIGHT: | SHEET 1 OF 1

Appendix J: Magnetometer Arduino Code

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_HMC5883_U.h>
/* Assign a unique ID to this sensor at the same time */
Adafruit_HMC5883_Unified mag = Adafruit_HMC5883_Unified(12345);
void displaySensorDetails(void)
{
  sensor_t sensor;
  mag.getSensor(&sensor);
  Serial.println("------------------------------------");
  Serial.print  ("Sensor:       "); Serial.println(sensor.name);
  Serial.print  ("Driver Ver:   "); Serial.println(sensor.version);
  Serial.print  ("Unique ID:    "); Serial.println(sensor.sensor_id);
  Serial.print  ("Max Value:    "); Serial.print(sensor.max_value); Serial.println(" uT");
  Serial.print  ("Min Value:    "); Serial.print(sensor.min_value); Serial.println(" uT");
  Serial.print  ("Resolution:   "); Serial.print(sensor.resolution); Serial.println(" uT");
  Serial.println("------------------------------------");
  Serial.println("");
  delay(750);
}
void setup(void)
{
  Serial.begin(9600);
  Serial.println("HMC5883 Magnetometer Test"); Serial.println("");
  /* Initialise the sensor */
  if(!mag.begin())
```

```
  {
    /* There was a problem detecting the HMC5883 ... check your connections */
    Serial.println("Ooops, no HMC5883 detected ... Check your wiring!");
    while(1);
  }
  /* Display some basic information on this sensor */
  displaySensorDetails();
}
void loop(void)
{
  /* Get a new sensor event */
  sensors_event_t event;
  mag.getEvent(&event);
  /* Display the results (magnetic vector values are in micro-Tesla (uT)) */
  Serial.print("X: "); Serial.print(event.magnetic.x); Serial.print("  ");
  Serial.print("Y: "); Serial.print(event.magnetic.y); Serial.print("  ");
  Serial.print("Z: "); Serial.print(event.magnetic.z); Serial.print("  ");Serial.println("uT");
  // Hold the module so that Z is pointing 'up' and you can measure the heading with x&y
  // Calculate heading when the magnetometer is level, then correct for signs of axis.
  float heading = atan2(event.magnetic.y, event.magnetic.x);
  float declinationAngle = 0.22;
  heading += declinationAngle;
  // Correct for when signs are reversed.
  if(heading < 0)
    heading += 2*PI;
  // Check for wrap due to addition of declination.
  if(heading > 2*PI)
```

```
    heading -= 2*PI;
  // Convert radians to degrees for readability.
  float headingDegrees = heading * 180/M_PI;
  Serial.print("Heading (degrees): "); Serial.println(headingDegrees);
  delay(500);
}
```

[/code]

Appendix K: Difference_plot.m

```
%Senior Design Project
%Jacob Stevens

clear all, clc, close all
```

Data

```
load testdatav3

ob      = 30;
num     = linspace(1,ob,ob);    %Observations
```

Difference

```
%X-Coil
diffxx1     = xx1t - xx1a;      %Difference in Test and ambient
diffxx2     = xx2t - xx2a;
diffxxt     = xx2t - xx1t;      %Difference sensor 2 to sensor 1 (test)
diffxxa     = xx2a - xx1a;      %Difference sensor 2 to sensor 1 (ambient)
diffxx      = (diffxxt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffxy1     = xy1t - xy1a;      %Difference in Test and ambient
diffxy2     = xy2t - xy2a;
diffxyt     = xy2t - xy1t;      %Difference sensor 2 to sensor 1 (test)
diffxya     = xy2a - xy1a;      %Difference sensor 2 to sensor 1 (ambient)
diffxy      = (diffxyt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffxz1     = xz1t - xz1a;      %Difference in Test and ambient
diffxz2     = xz2t - xz2a;
diffxzt     = xz2t - xz1t;      %Difference sensor 2 to sensor 1 (test)
diffxza     = xz2a - xz1a;      %Difference sensor 2 to sensor 1 (ambient)
diffxz      = (diffxzt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

%Y-coil

diffyx1     = yx1t - yx1a;      %Difference in Test and ambient
diffyx2     = yx2t - yx2a;
diffyxt     = yx2t - yx1t;      %Difference sensor 2 to sensor 1 (test)
diffyxa     = yx2a - yx1a;      %Difference sensor 2 to sensor 1 (ambient)
diffyx      = (diffyxt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffyy1     = yy1t - yy1a;      %Difference in Test and ambient
diffyy2     = yy2t - yy2a;
diffyyt     = yy2t - yy1t;      %Difference sensor 2 to sensor 1 (test)
diffyya     = yy2a - yy1a;      %Difference sensor 2 to sensor 1 (ambient)
diffyy      = (diffyyt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffyz1     = yz1t - yz1a;      %Difference in Test and ambient
```

```
diffyz2      = yz2t - yz2a;
diffyzt      = yz2t - yz1t;      %Difference sensor 2 to sensor 1 (test)
diffyza      = yz2a - yz1a;      %Difference sensor 2 to sensor 1 (ambient)
diffyz       = (diffyzt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

%Z-coil

diffzx1      = zx1t - zx1a;      %Difference in Test and ambient
diffzx2      = zx2t - zx2a;
diffzxt      = zx2t - zx1t;      %Difference sensor 2 to sensor 1 (test)
diffzxa      = zx2a - zx1a;      %Difference sensor 2 to sensor 1 (ambient)
diffzx       = (diffzxt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffzy1      = zy1t - zy1a;      %Difference in Test and ambient
diffzy2      = zy2t - zy2a;
diffzyt      = zy2t - zy1t;      %Difference sensor 2 to sensor 1 (test)
diffzya      = zy2a - zy1a;      %Difference sensor 2 to sensor 1 (ambient)
diffzy       = (diffzyt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)

diffzz1      = zz1t - zz1a;      %Difference in Test and ambient
diffzz2      = zz2t - zz2a;
diffzzt      = zz2t - zz1t;      %Difference sensor 2 to sensor 1 (test)
diffzza      = zz2a - zz1a;      %Difference sensor 2 to sensor 1 (ambient)
diffzz       = (diffzzt)/3;      %Variation of magnetic field per decimeter (Gauss/dm)
```

Plotting

```
%x-x
figure(1)
plot(num,xx1t, num, xx2t,'--'),title('Sensor Measurments vs Time (Xcoils, x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-2 -1])

figure(2)
plot(num,diffxx1, num, diffxx2,'--'),title('Field Strength Relative to CubeSat vs Time (Xcoils,
x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-2 -1])

hold on
figure(3)
plot(num,diffxx,'r-'),title('Gauss Variation per Decimeter vs Time (Xcoils, x-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off
```

```
%x-y
figure(4)
plot(num,xy1t, num, xy2t,'--'),title('Sensor Measurments vs Time (Xcoils, y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([0 1])

figure(5)
plot(num,diffxy1, num, diffxy2,'--'),title('Field Strength Relative to CubeSat vs Time (Xcoils, y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 1])

hold on
figure(6)
plot(num,diffxy,'r-'),title('Gauss Variation per Decimeter vs Time (Xcoils, y-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%x-z
figure(7)
plot(num,xz1t, num, xz2t,'--'),title('Sensor Measurments vs Time (Xcoils, z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([0 1])

figure(8)
plot(num,diffxz1, num, diffxz2,'--'),title('Field Strength Relative to CubeSat vs Time (Xcoils, z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 1])

hold on
figure(9)
plot(num,diffxz,'r-'),title('Gauss Variation per Decimeter vs Time(Xcoils, z-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off
```

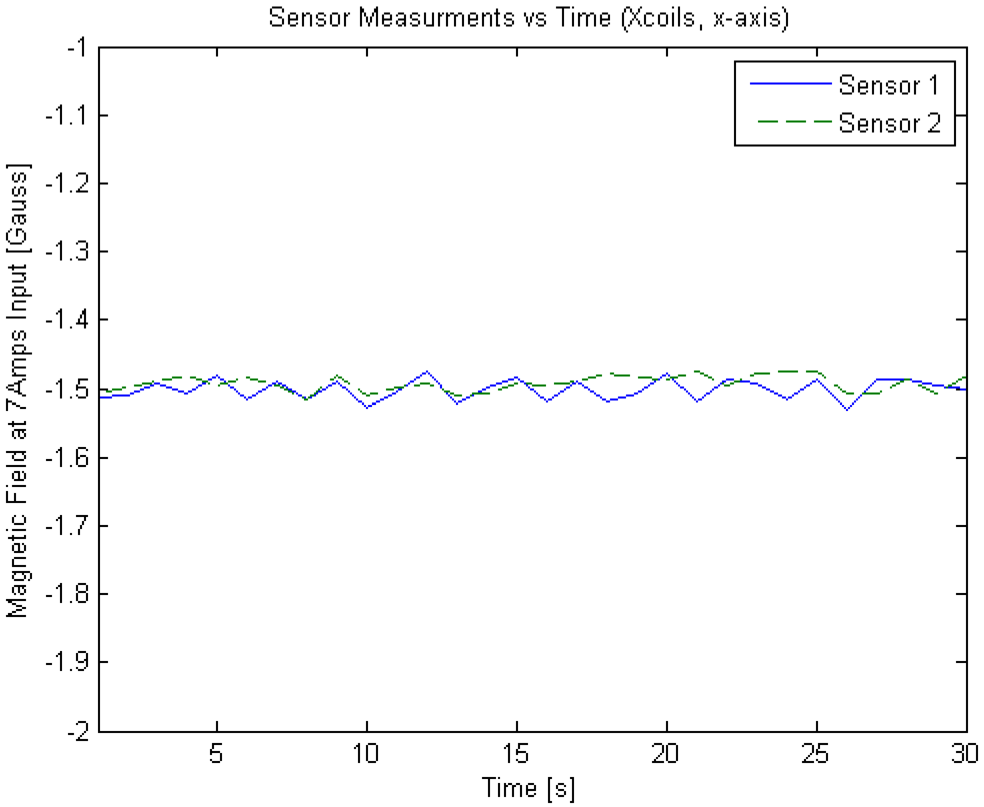
```
%y-x
figure(10)
plot(num,yx1t, num, yx2t,'--'),title('Sensor Measurments vs Time (Ycoils, x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

figure(11)
plot(num,diffyx1, num, diffyx2,'--'),title('Field Strength Relative to CubeSat vs Time (Ycoils,
x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

hold on
figure(12)
plot(num,diffyx,'r-'),title('Gauss Variation per Decimeter vs Time(Ycoils, x-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%y-y
figure(13)
plot(num,yy1t, num, yy2t,'--'),title('Sensor Measurments vs Time (Ycoils, y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-2 -1])

figure(14)
plot(num,diffyy1, num, diffyy2,'--'),title('Field Strength Relative to CubeSat vs Time (Ycoils,
y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-2 -1])

hold on
figure(15)
plot(num,diffyy,'r-'),title('Gauss Variation per Decimeter vs Time(Ycoils, y-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%y-z
```

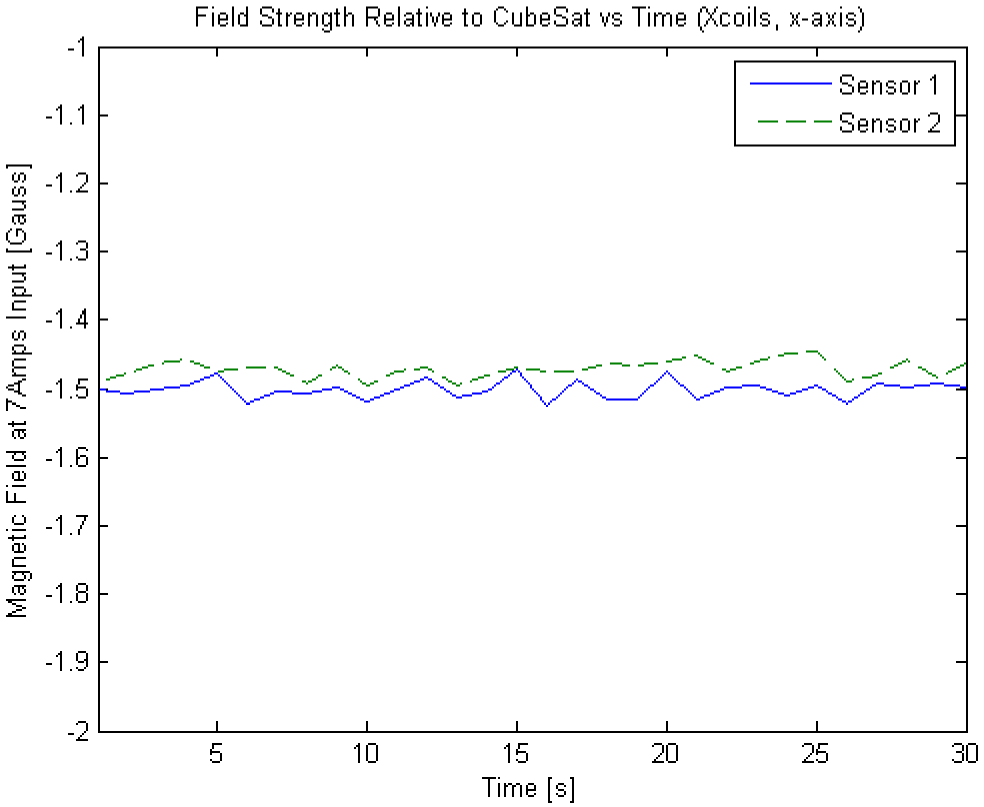
```
figure(16)
plot(num,yz1t, num, yz2t,'--'),title('Sensor Measurments vs Time (Ycoils, z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([0 1])

figure(17)
plot(num,diffyz1, num, diffyz2,'--'),title('Field Strength Relative to CubeSat vs Time (Ycoils,
z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([0 1])

hold on
figure(18)
plot(num,diffyz,'r-'),title('Gauss Variation per Decimeter vs Time(Ycoils, z-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%z-x
figure(19)
plot(num,zx1t, num, zx2t,'--'),title('Sensor Measurments vs Time (Zcoils, x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

figure(20)
plot(num,diffzx1, num, diffzx2,'--'),title('Field Strength Relative to CubeSat vs Time (Zcoils,
x-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

hold on
figure(21)
plot(num,diffzx,'r-'),title('Gauss Variation per Decimeter vs Time(Zcoils, x-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%z-y
figure(22)
```

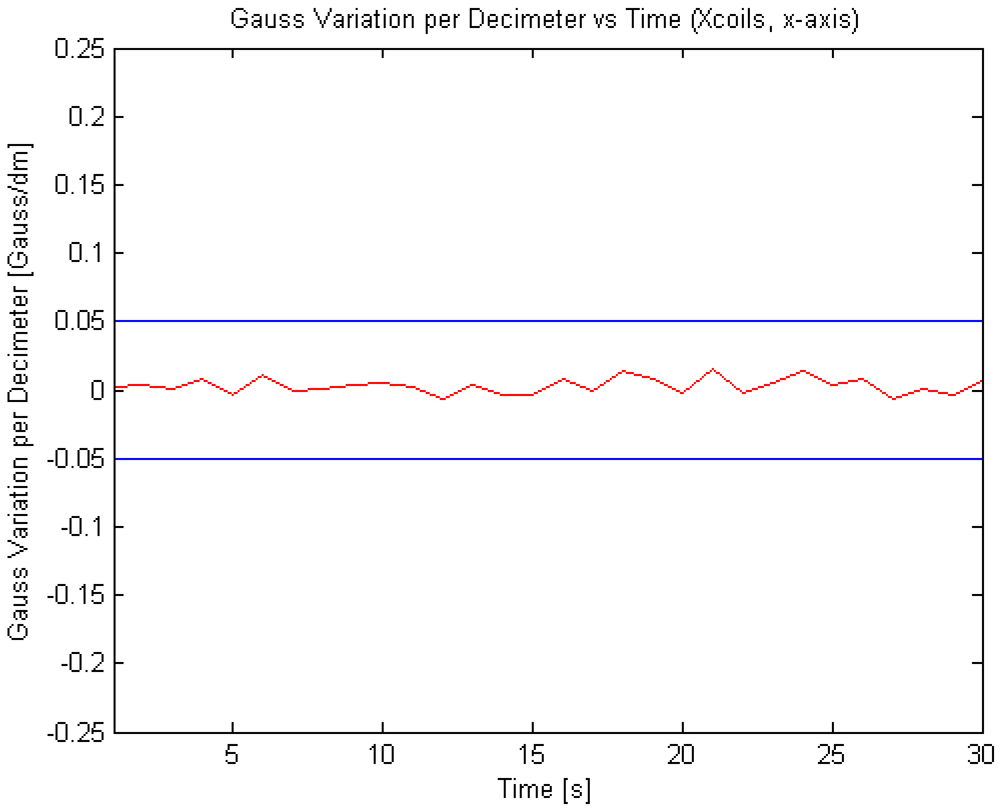
```
plot(num,zy1t, num, zy2t,'--'),title('Sensor Measurments vs Time (Zcoils, y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

figure(23)
plot(num,diffzy1, num, diffzy2,'--'),title('Field Strength Relative to CubeSat vs Time (Zcoils, y-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([-.5 .5])

hold on
figure(24)
plot(num,diffzy,'r-'),title('Gauss Variation per Decimeter vs Time(Zcoils, y-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%z-z
figure(25)
plot(num,zz1t, num, zz2t,'--'),title('Sensor Measurments vs Time (Zcoils, z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([1 2])

figure(26)
plot(num,diffzz1, num, diffzz2,'--'),title('Field Strength Relative to CubeSat vs Time (Zcoils, z-axis)')
xlabel('Time [s]'),ylabel('Magnetic Field at 7Amps Input [Gauss]')
legend('Sensor 1','Sensor 2')
xlim([1 ob])
ylim([1 2])

hold on
figure(27)
plot(num,diffzz,'r-'),title('Gauss Variation per Decimeter vs Time(Zcoils, z-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

%Ambient Variation

hold on
```

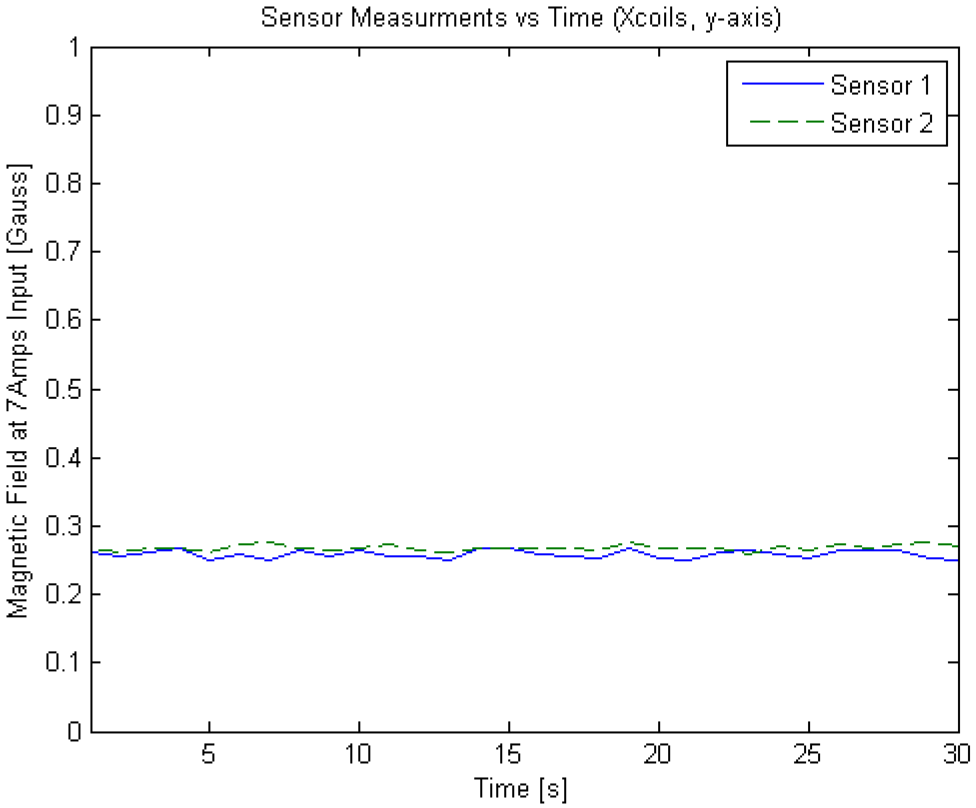
```
figure(28)
plot(num,diffxxa/2,'r-'),title('Gauss Variation per Decimeter vs Time(Ambient, x-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

hold on
figure(29)
plot(num,diffyya/2,'r-'),title('Gauss Variation per Decimeter vs Time(Ambient, y-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off

hold on
figure(30)
plot(num,diffzza/2,'r-'),title('Gauss Variation per Decimeter vs Time(Ambient, z-axis)')
refline(0,.05)
refline(0,-.05)
xlabel('Time [s]'),ylabel('Gauss Variation per Decimeter [Gauss/dm]')
xlim([1 ob])
ylim([-.25 .25])
hold off
```

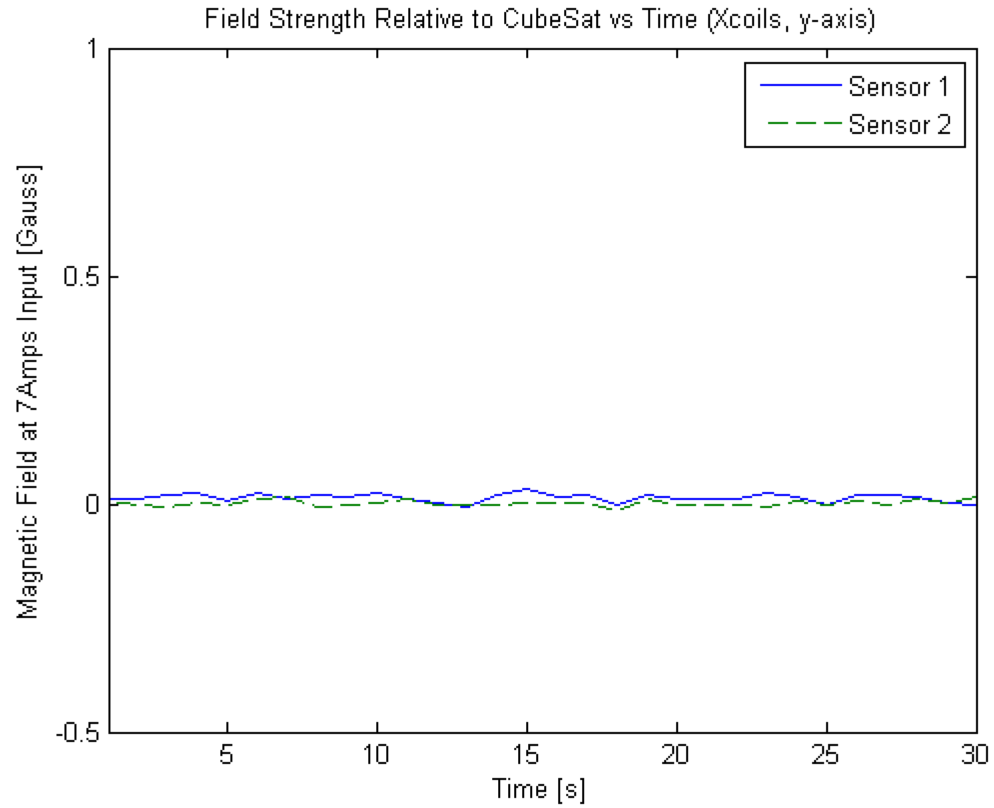
Appendix L: Uniform Magnetic Field Results – X Axis Coils

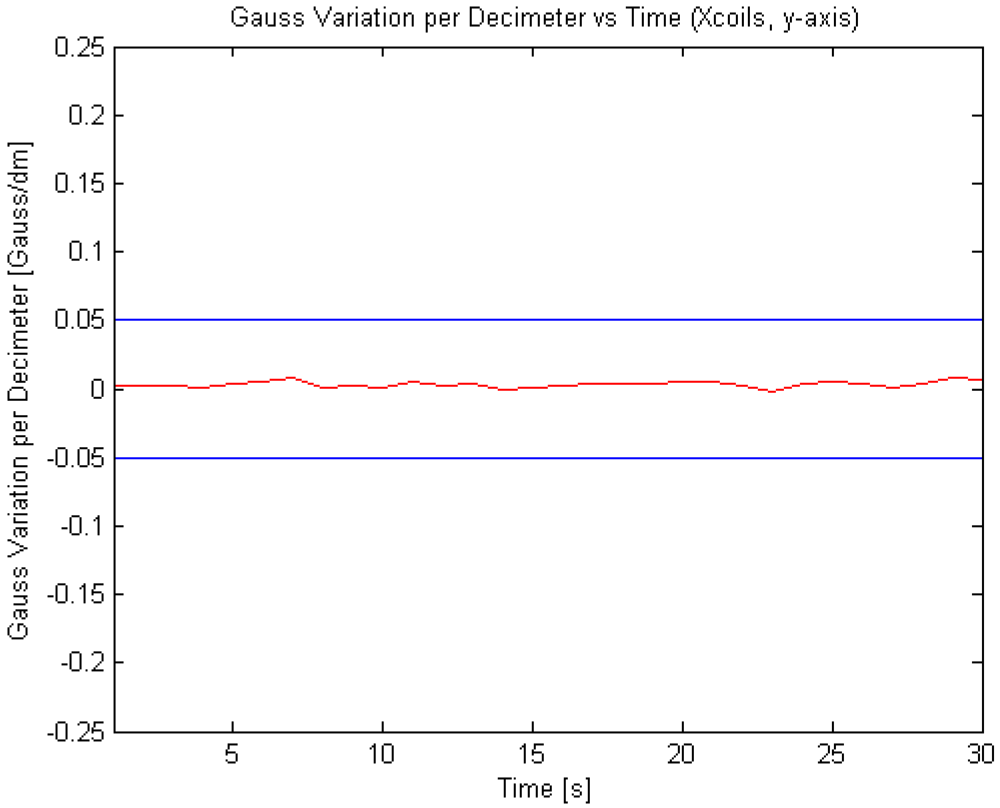


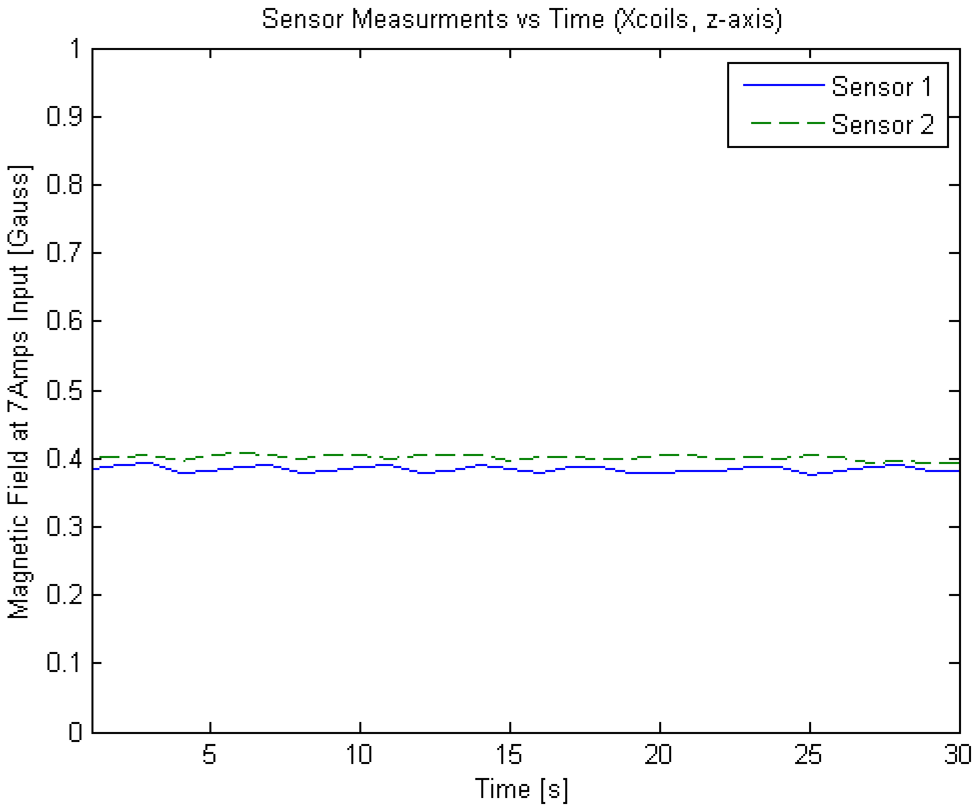



Gauss Variation per Decimeter vs Time (Xcoils, x-axis)
Gauss Variation per Decimeter [Gauss/dm]
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
5
10
15
20
25
30
Time [s]

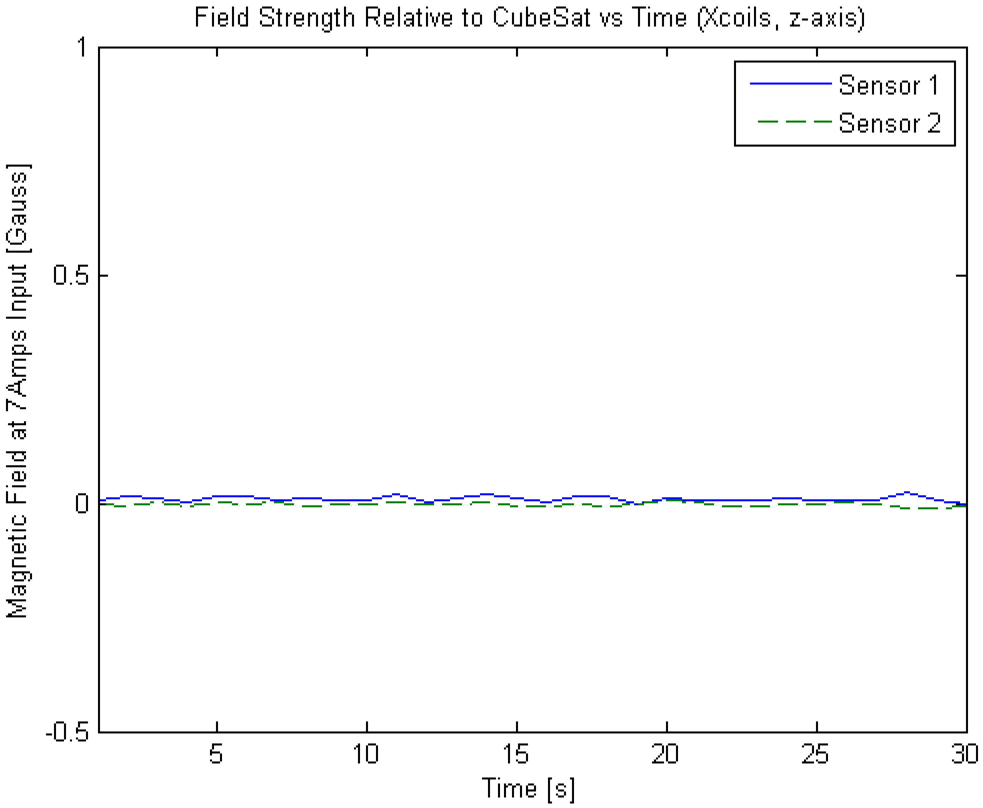

Sensor Measurments vs Time (Xcoils, y-axis)
Magnetic Field at 7Amps Input [Gauss]
Sensor 1
Sensor 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
5
10
15
20
25
30
Time [s]


Field Strength Relative to CubeSat vs Time (Xcoils, y-axis)

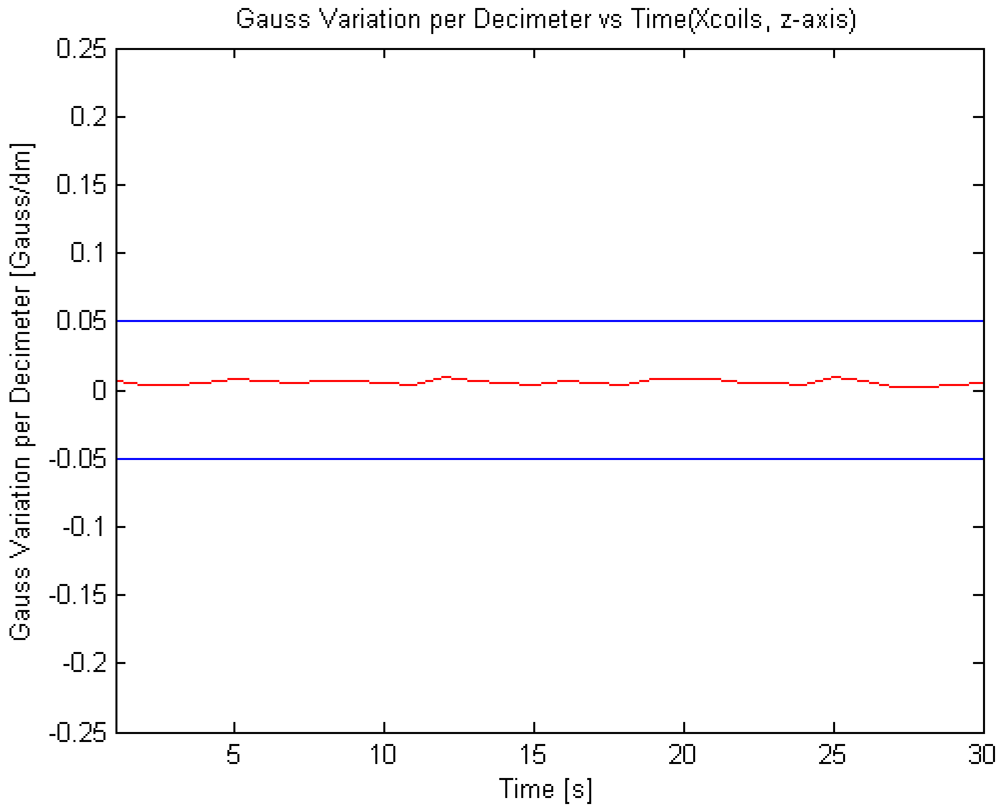

Gauss Variation per Decimeter vs Time (Xcoils, y-axis)



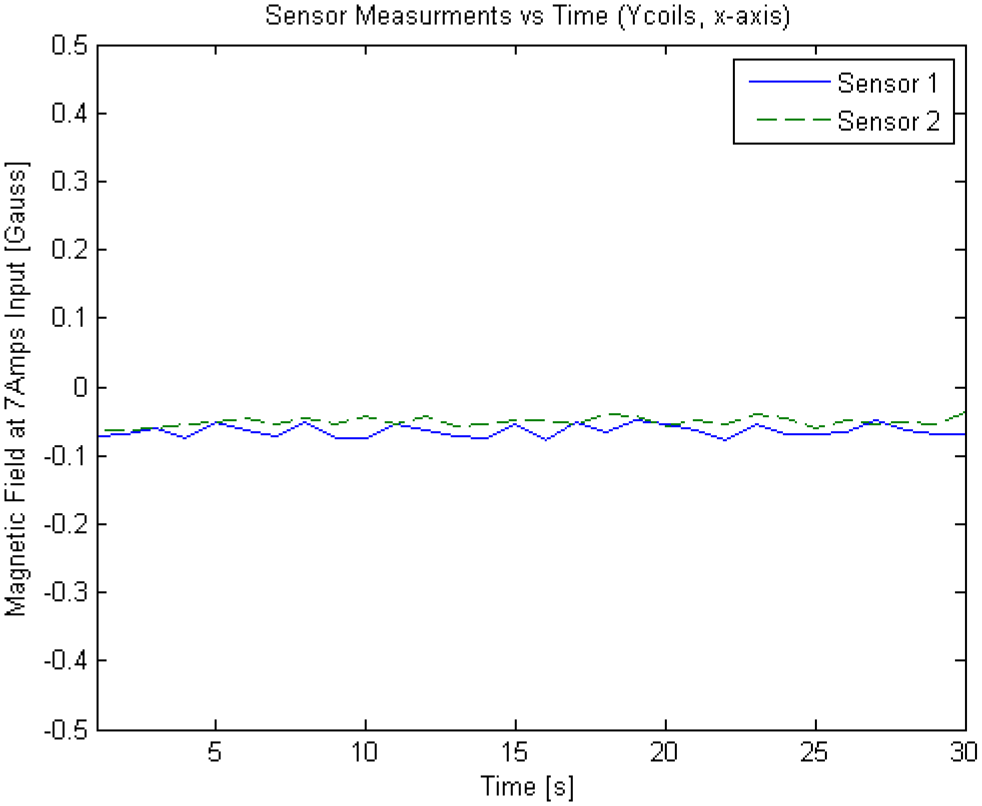
Sensor Measurments vs Time (Xcoils, z-axis)
Sensor 1
Sensor 2
Magnetic Field at 7Amps Input [Gauss]
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
5
10
15
20
25
30
Time [s]

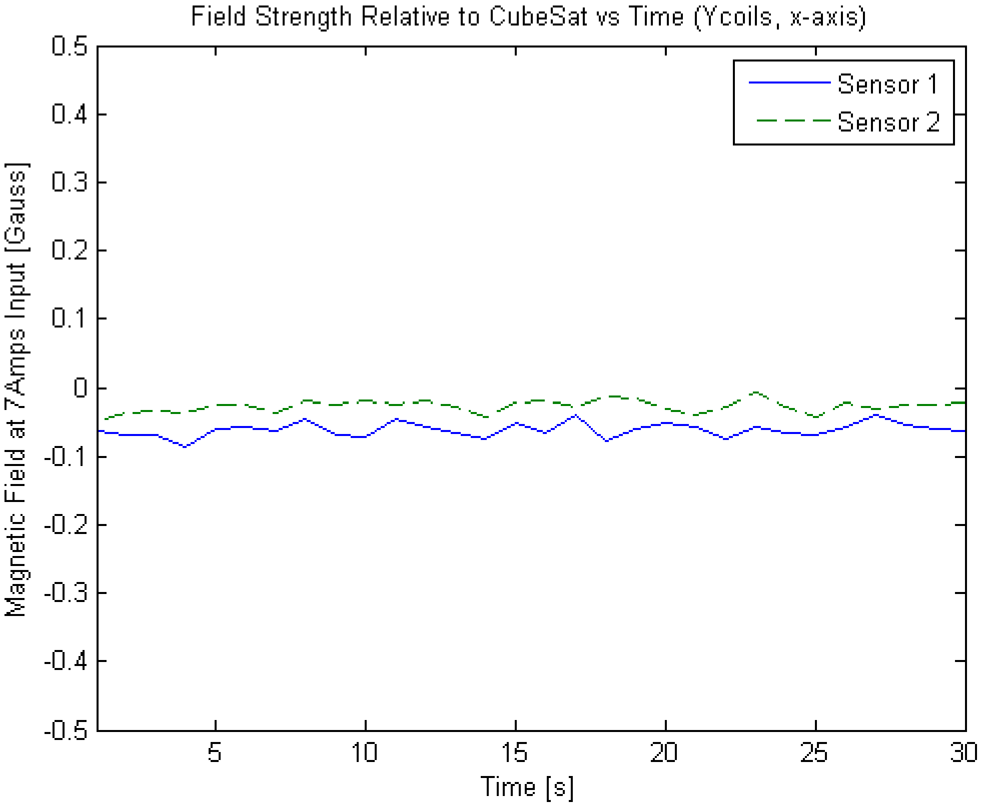


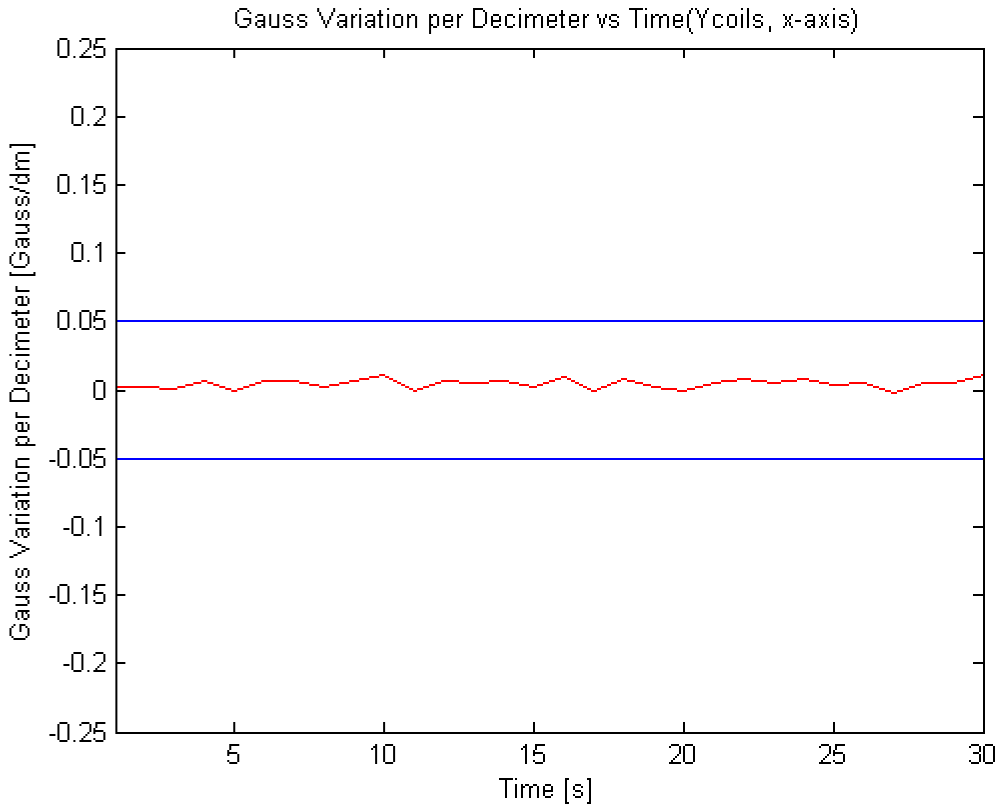
Field Strength Relative to CubeSat vs Time (Xcoils, z-axis)
Sensor 1
Sensor 2
Magnetic Field at 7Amps Input [Gauss]
-0.5
0
0.5
1
5
10
15
20
25
30
Time [s]


Gauss Variation per Decimeter vs Time(Xcoils, z-axis)
Gauss Variation per Decimeter [Gauss/dm]
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
5
10
15
20
25
30
Time [s]

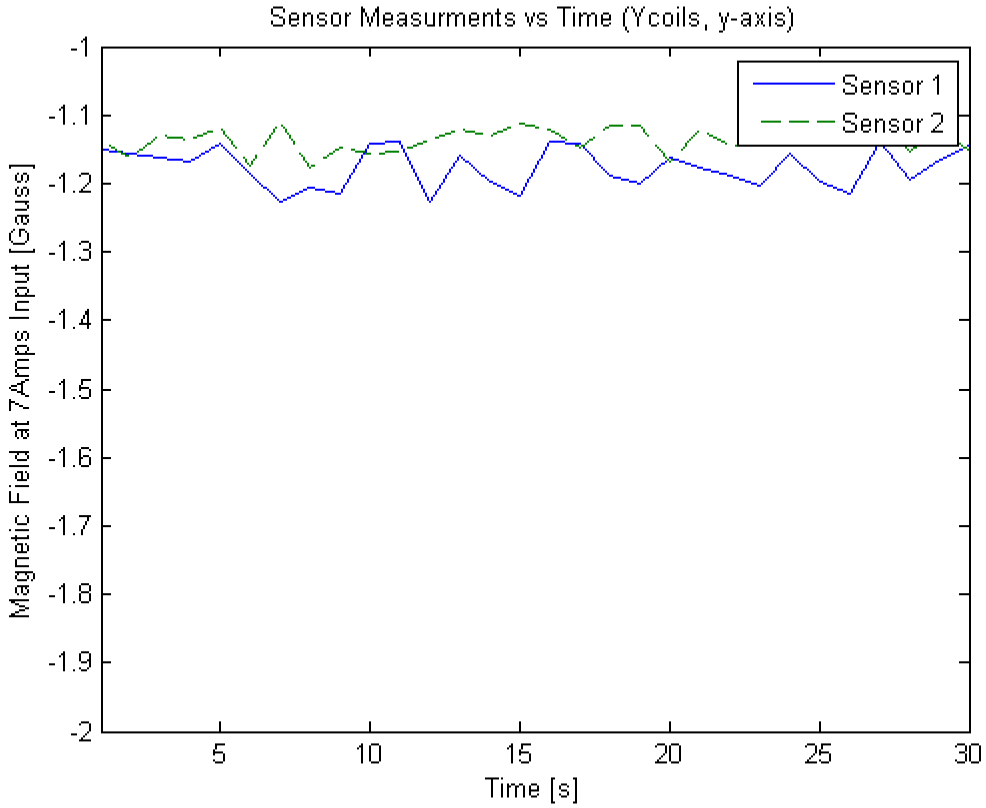
Appendix M: Uniform Magnetic Field Results – Y Axis Coils



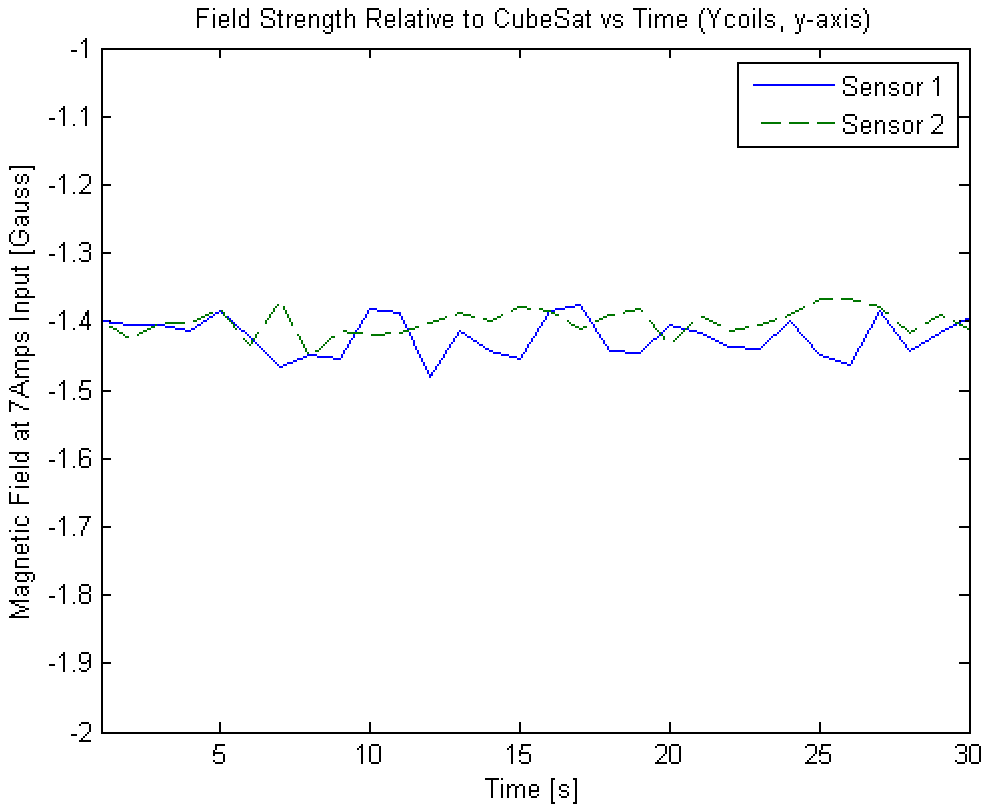


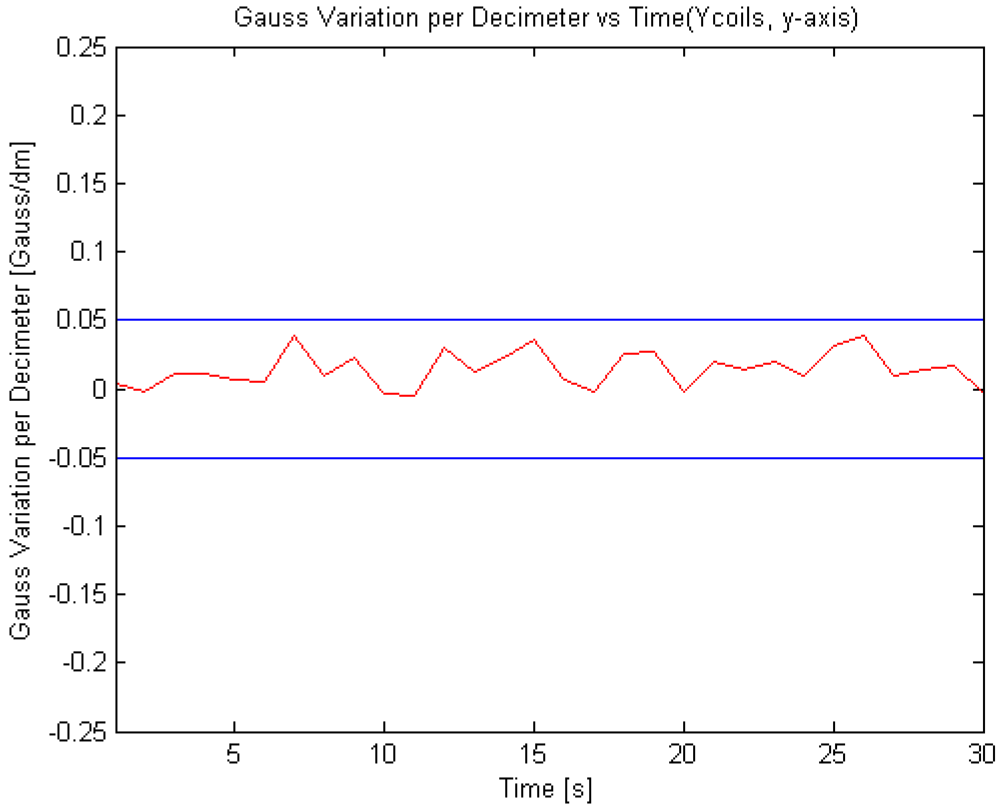


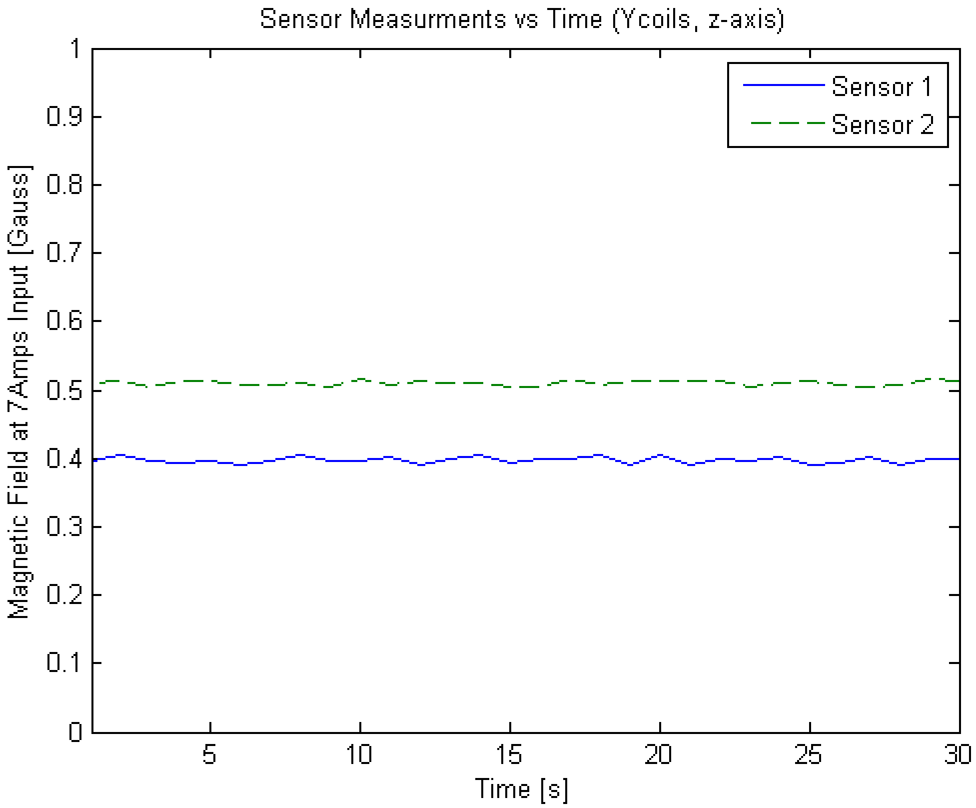
Gauss Variation per Decimeter vs Time(Ycoils, x-axis)

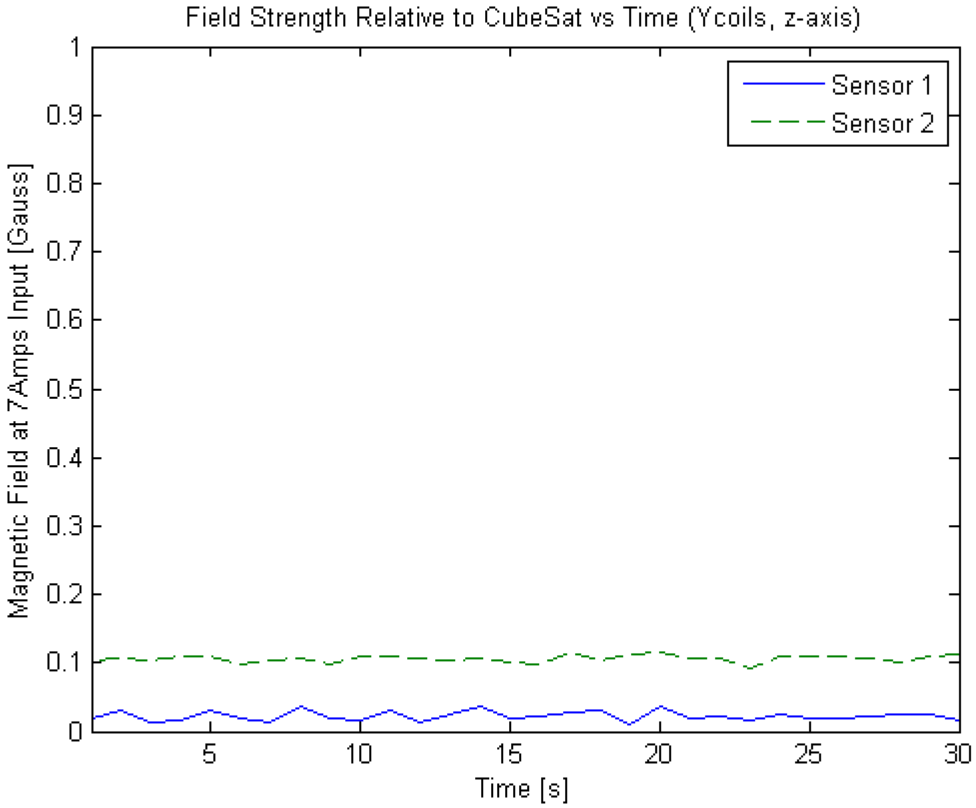


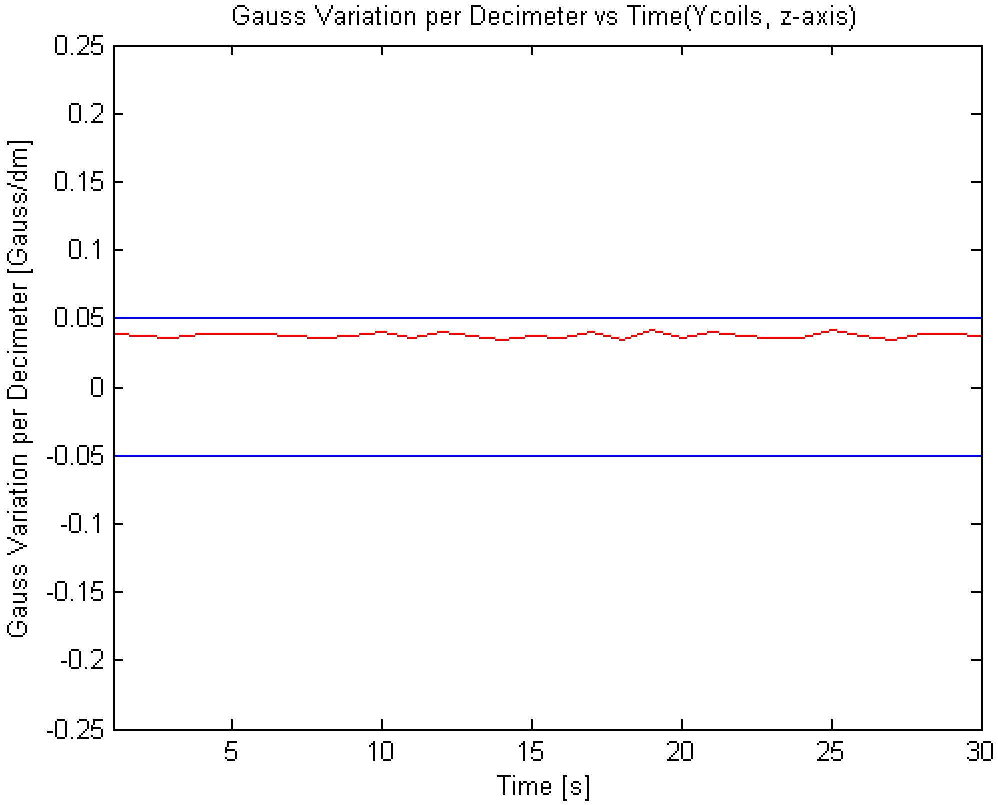
Sensor Measurments vs Time (Ycoils, y-axis)


Field Strength Relative to CubeSat vs Time (Ycoils, y-axis)
Sensor 1
Sensor 2
Magnetic Field at 7Amps Input [Gauss]
-1
-1.1
-1.2
-1.3
-1.4
-1.5
-1.6
-1.7
-1.8
-1.9
-2
5
10
15
20
25
30
Time [s]

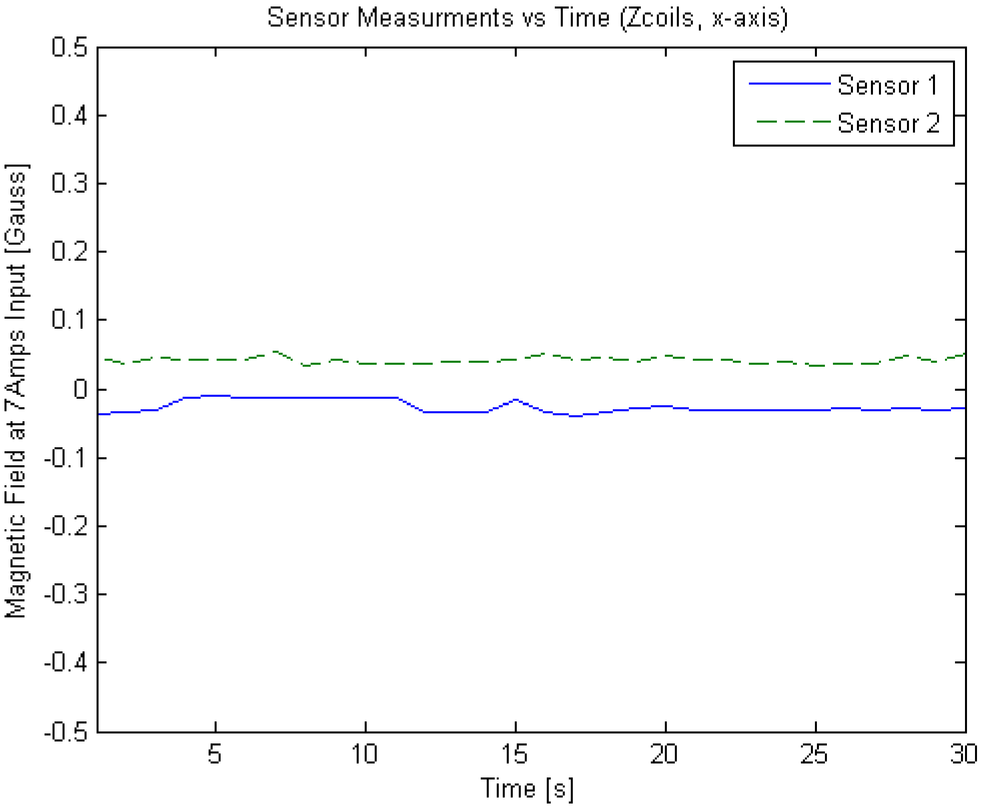

Gauss Variation per Decimeter vs Time(Ycoils, y-axis)
Gauss Variation per Decimeter [Gauss/dm]
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
5
10
15
20
25
30
Time [s]

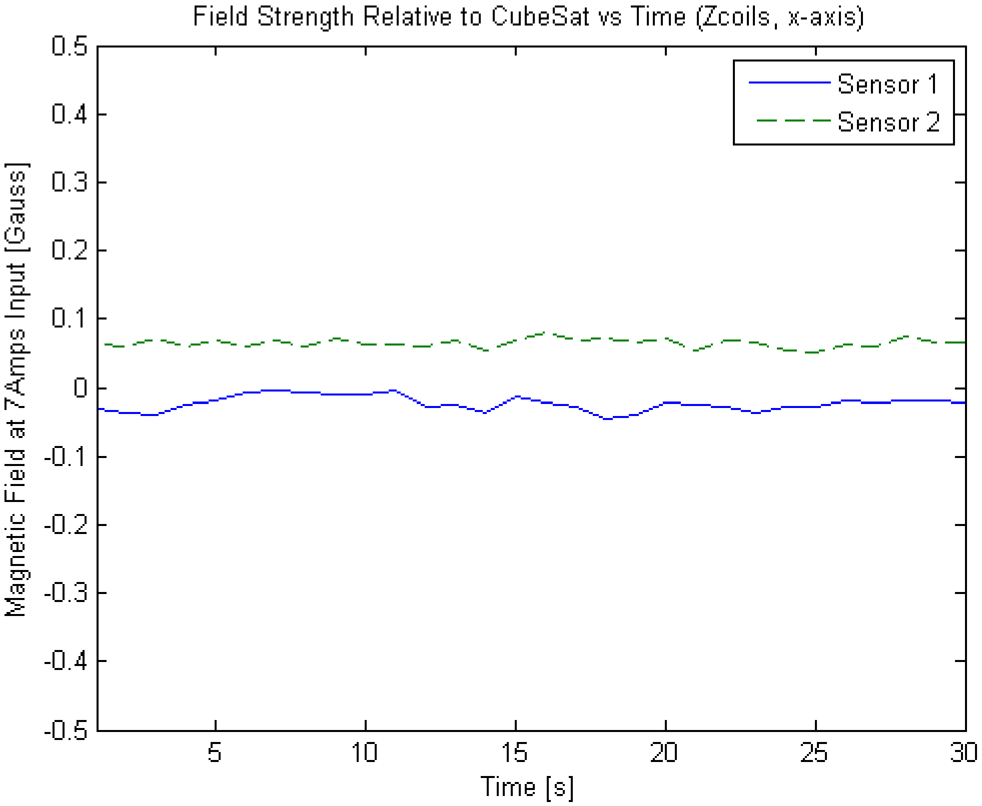

Sensor Measurments vs Time (Ycoils, z-axis)

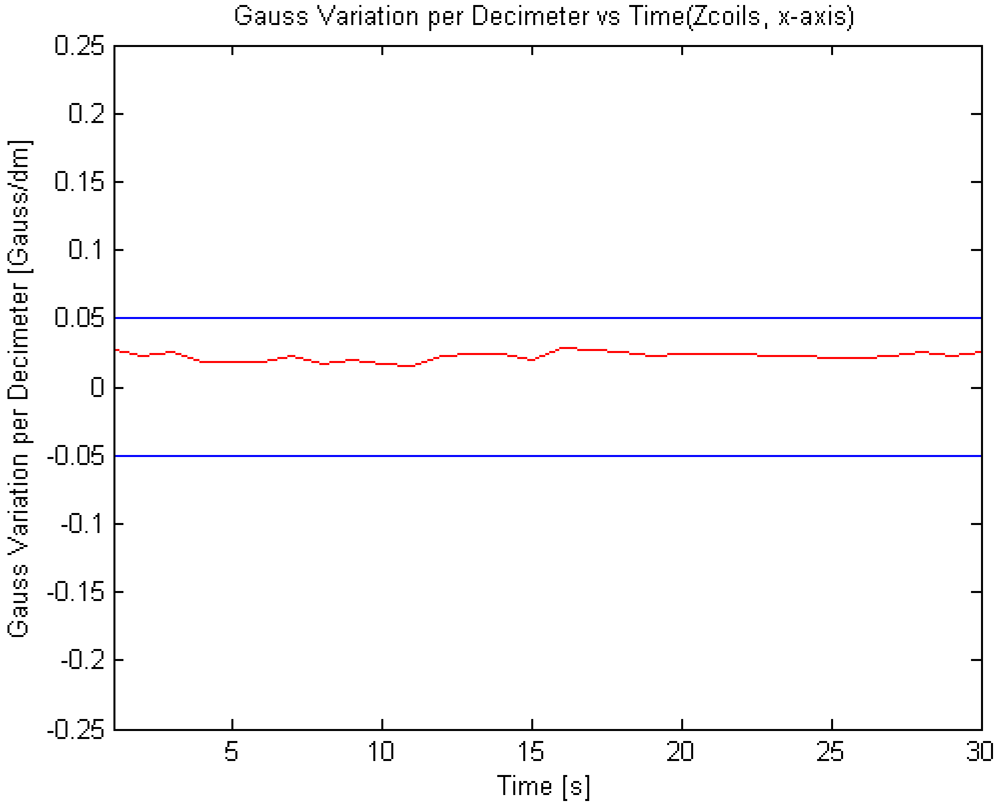

Field Strength Relative to CubeSat vs Time (Ycoils, z-axis)


Gauss Variation per Decimeter vs Time(Ycoils, z-axis)
Gauss Variation per Decimeter [Gauss/dm]
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
5
10
15
20
25
30
Time [s]

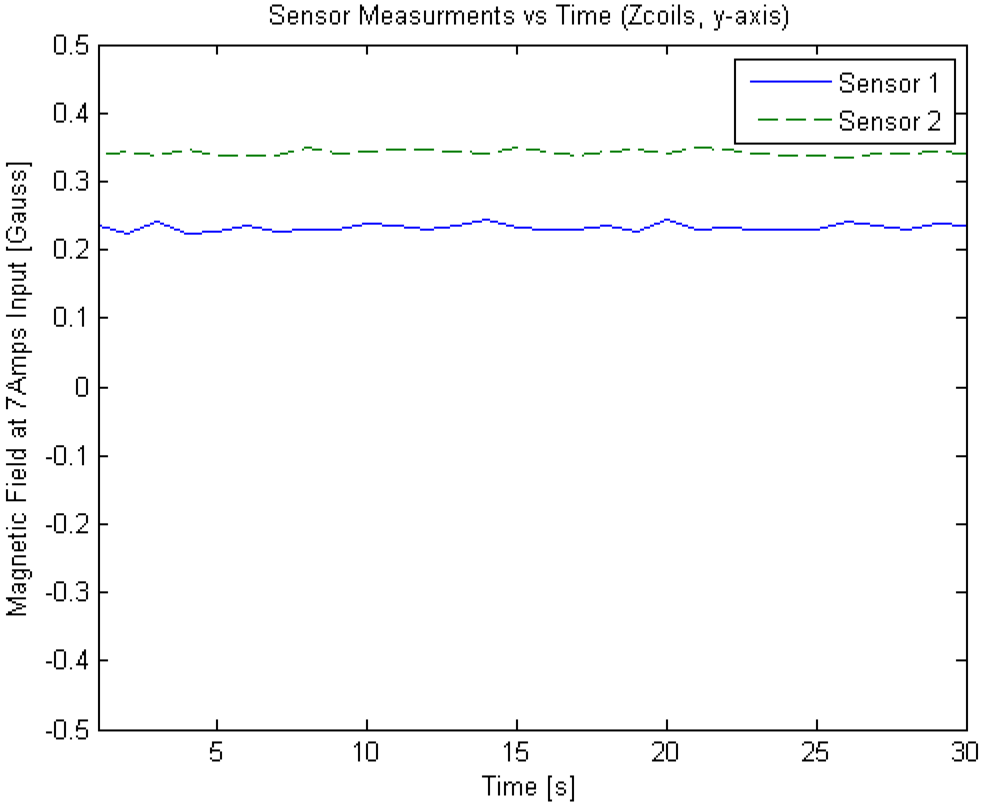
Appendix N: Uniform Magnetic Field Results – Z Axis Coils



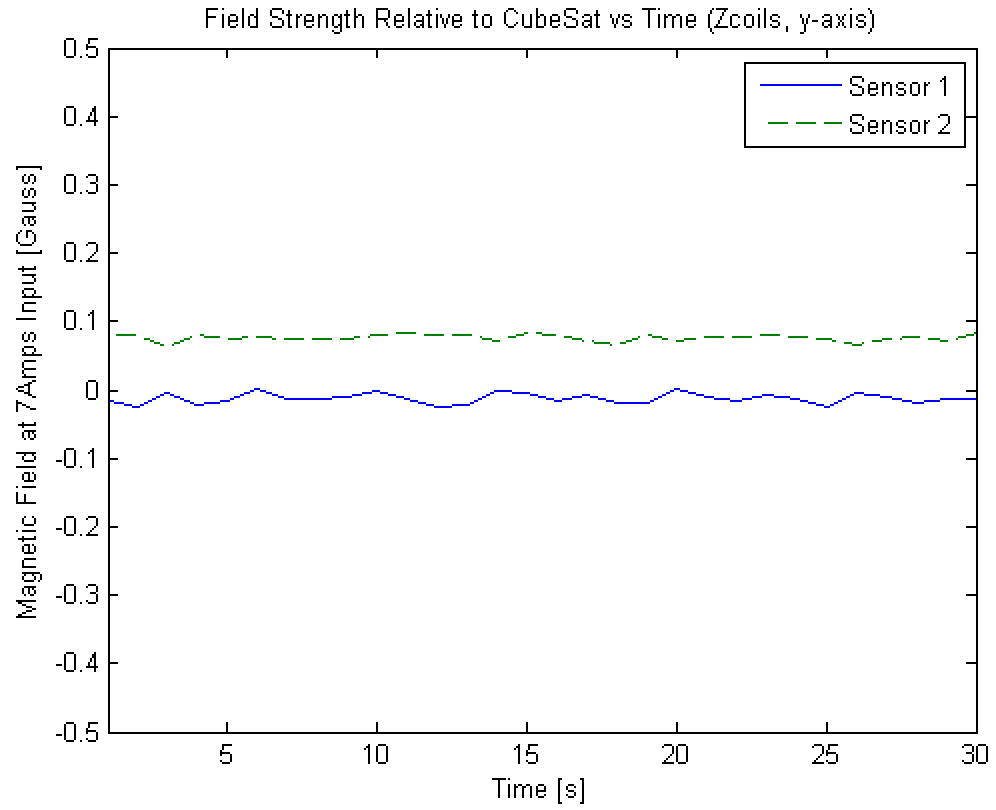


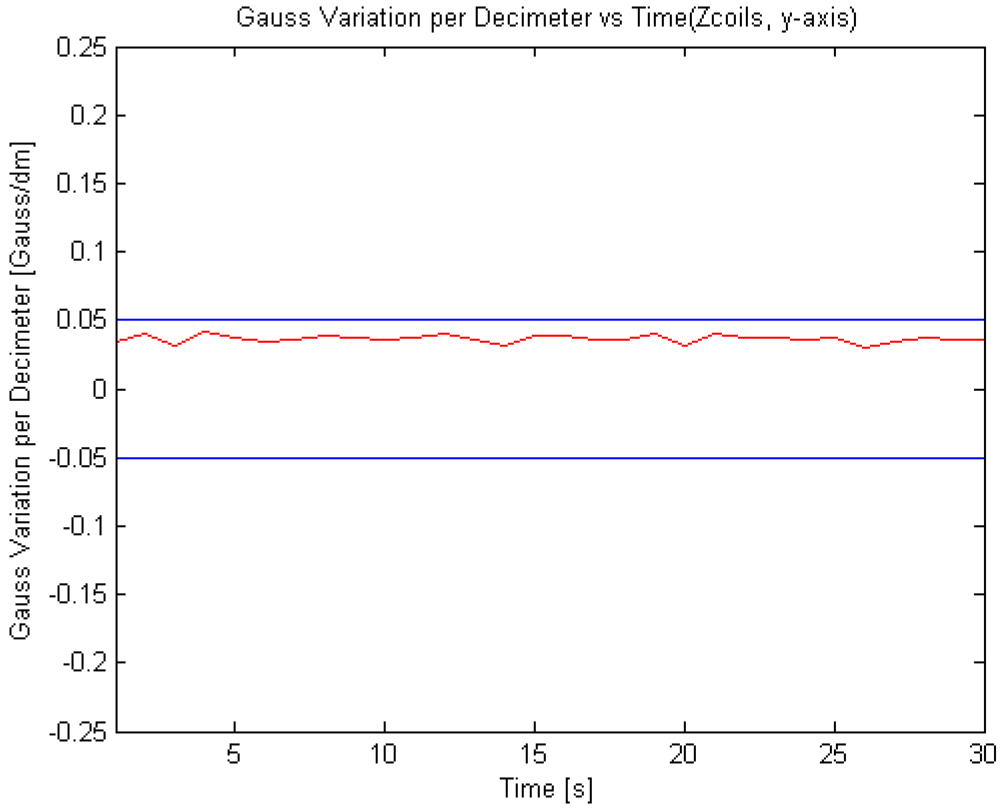


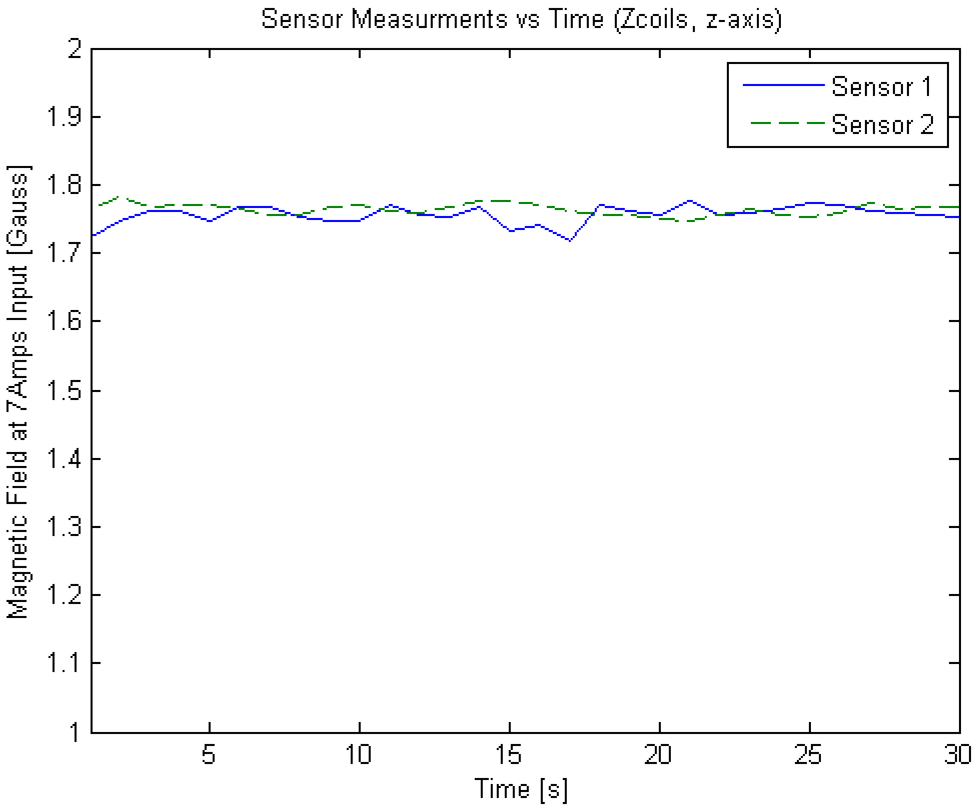
Gauss Variation per Decimeter vs Time(Zcoils, x-axis)

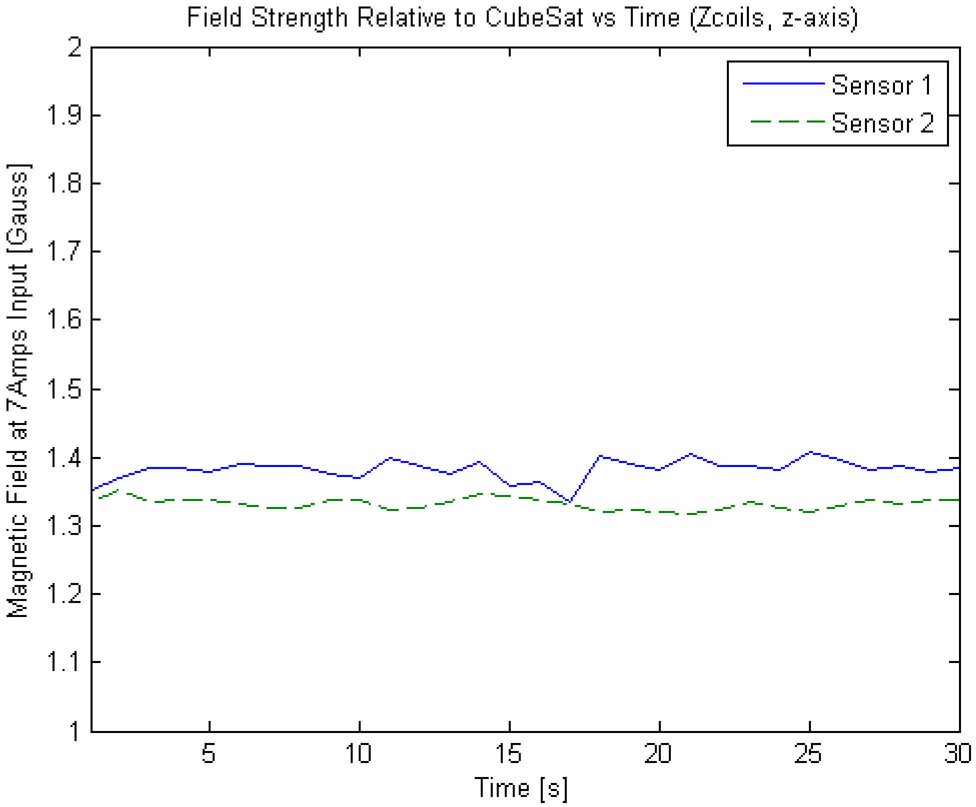


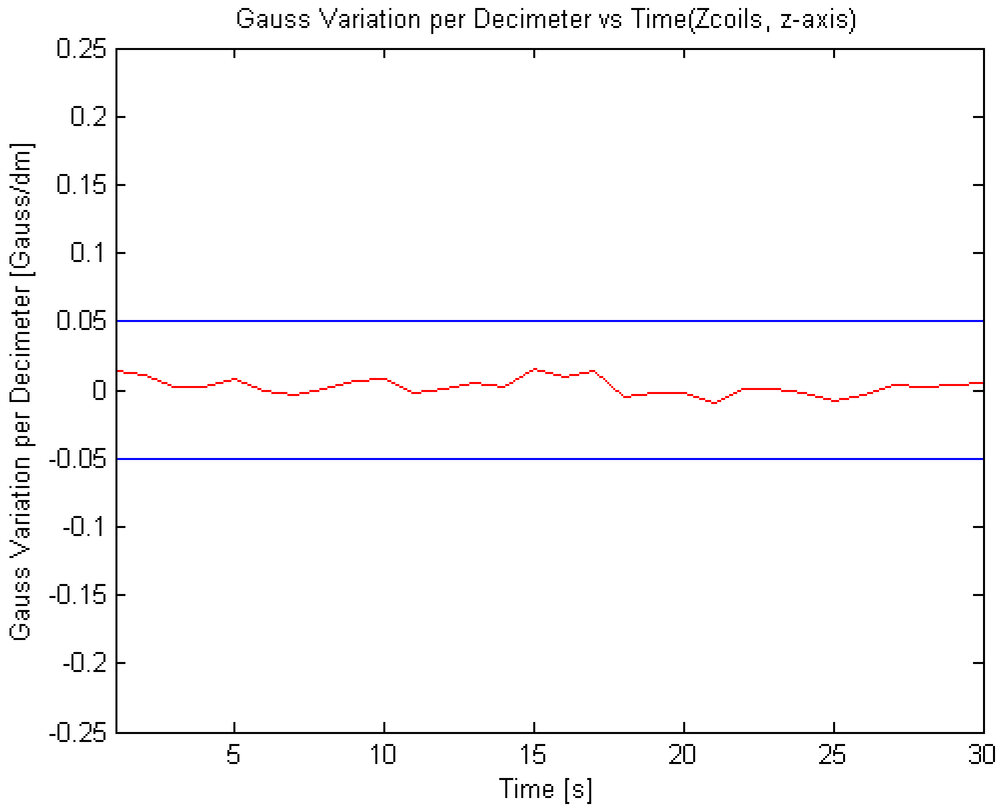
Sensor Measurments vs Time (Zcoils, y-axis)


Field Strength Relative to CubeSat vs Time (Zcoils, y-axis)


Gauss Variation per Decimeter vs Time(Zcoils, y-axis)

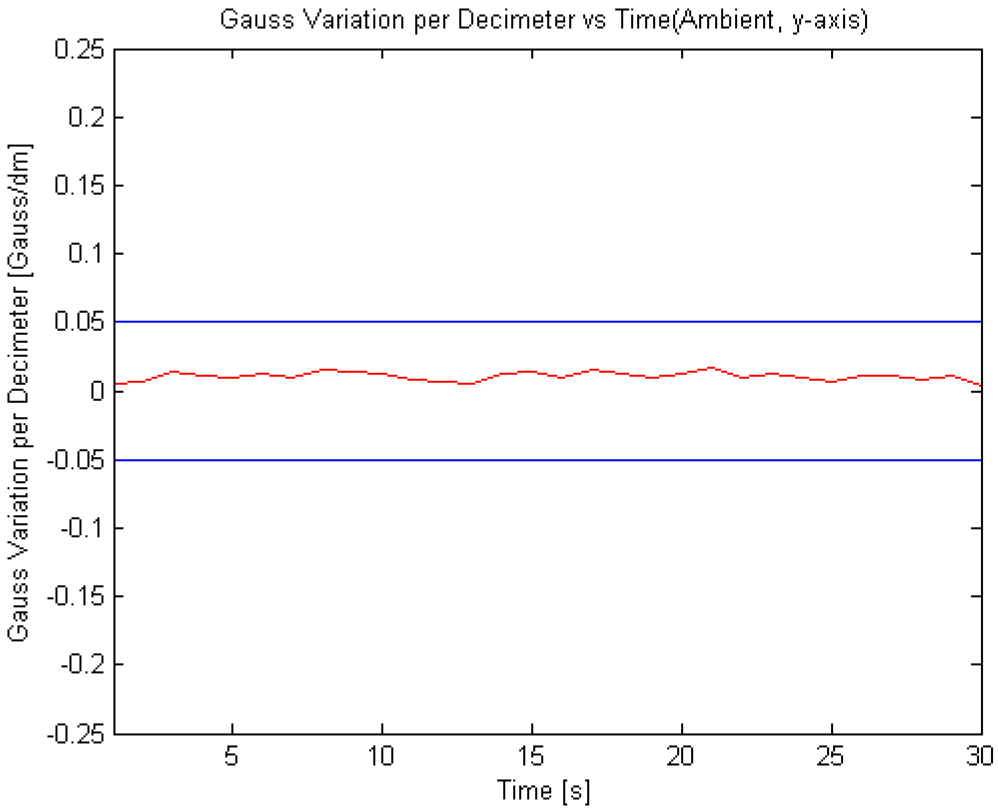

Sensor Measurments vs Time (Zcoils, z-axis)

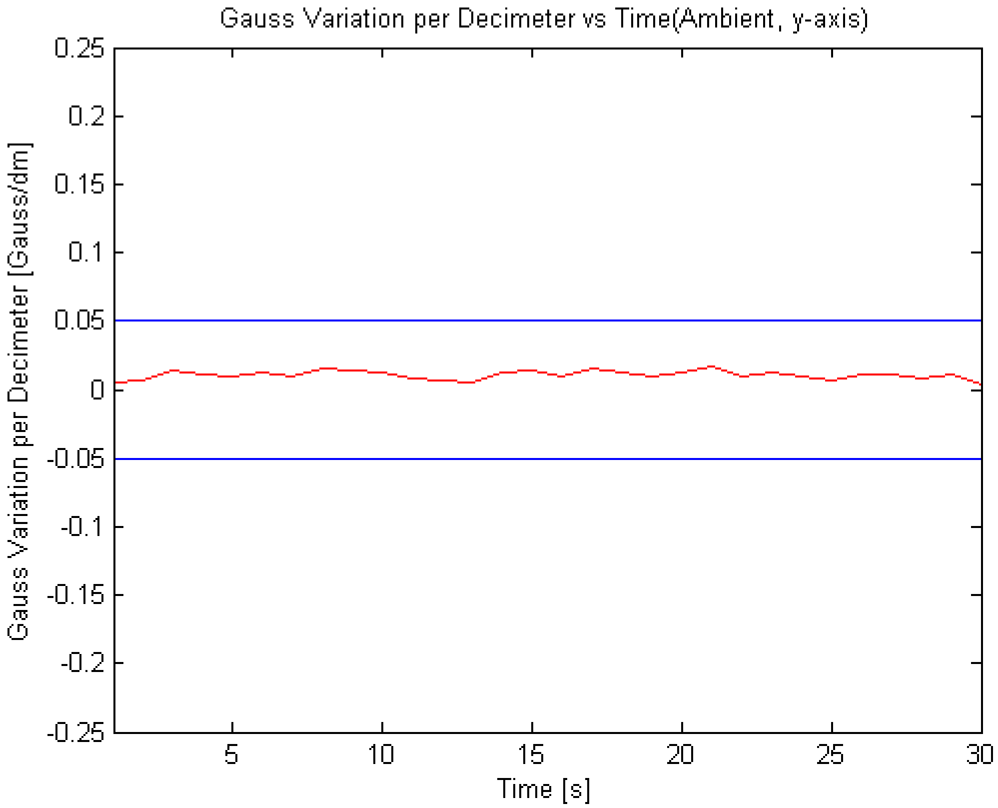

Field Strength Relative to CubeSat vs Time (Zcoils, z-axis)

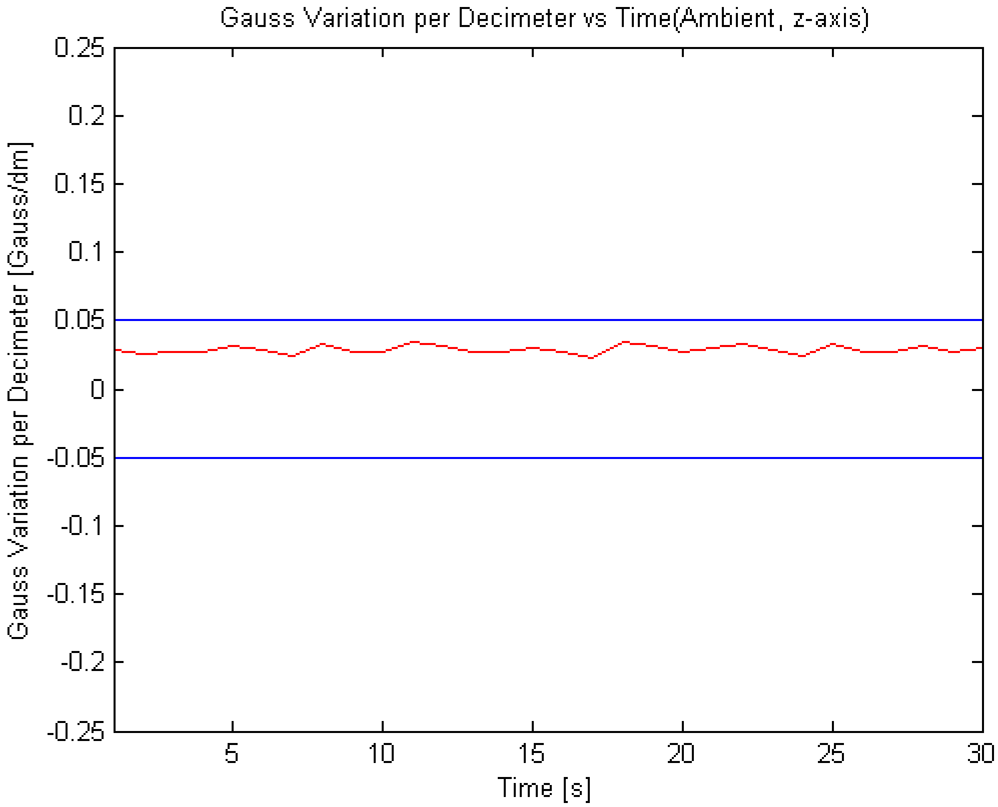


Gauss Variation per Decimeter vs Time(Zcoils, z-axis)

Appendix O: Ambient Gauss Variation






Gauss Variation per Decimeter vs Time(Ambient, z-axis)
Gauss Variation per Decimeter [Gauss/dm]
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
5
10
15
20
25
30
Time [s]

Appendix P: testdatav3.mat

Contents

- Data
- Check vector length

```
%Senior Design Project
% Jacob Stevens
```

Data

```
% Magnetic field in micro tesla [Gauss]
xx1a = [-0.0109 -0.0009 0.0091 -0.0100 -0.0027 0.0045 0.0127 -0.0109 0.0082 -0.0091 -0.0018 0.0082 -0.0109 0.0073 -0.0100 0.0045 -0.0045 -0.0018 0.0091 -0.0036 -0.0045 0.0100 0.0018 -0.0055 0.0109 -0.0091 0.0036 0.0118 -0.0036 -0.0009];
xx2a = [-0.0173 -0.0209 -0.0255 -0.0255 -0.0182 -0.0155 -0.0255 -0.0236 -0.0127 -0.0145 -0.0227 -0.0245 -0.0136 -0.0264 -0.0255 -0.0209 -0.0164 -0.0145 -0.0164 -0.0255 -0.0236 -0.0182 -0.0164 -0.0282 -0.0282 -0.0191 -0.0273 -0.0282 -0.0227 -0.0173 ];
xx1t = [-1.5136 -1.5091 -1.4918 -1.5064 -1.4818 -1.5164 -1.4909 -1.5173 -1.4900 -1.5282 -1.5045 -1.4745 -1.5227 -1.4973 -1.4836 -1.5191 -1.4909 -1.5191 -1.5073 -1.4773 -1.5200 -1.4882 -1.4936 -1.5164 -1.4855 -1.5309 -1.4882 -1.4873 -1.4964 -1.5000 ];
xx2t = [-1.5082 -1.4991 -1.4891 -1.4818 -1.4945 -1.4836 -1.4945 -1.5155 -1.4800 -1.5109 -1.4973 -1.4936 -1.5100 -1.5073 -1.4936 -1.4964 -1.4909 -1.4782 -1.4836 -1.4855 -1.4755 -1.4945 -1.4773 -1.4764 -1.4745 -1.5082 -1.5073 -1.4864 -1.5073 -1.4800 ];
xy1a = [0.2500 0.2482 0.2436 0.2445 0.2436 0.2355 0.2418 0.2445 0.2400 0.2400 0.2473 0.2536 0.2555 0.2455 0.2364 0.2445 0.2345 0.2527 0.2455 0.2418 0.2382 0.2491 0.2382 0.2436 0.2527 0.2473 0.2455 0.2482 0.2500 0.2491 ];
xy2a = [0.2591 0.2618 0.2727 0.2655 0.2636 0.2609 0.2618 0.2755 0.2664 0.2645 0.2636 0.2655 0.2645 0.2691 0.2655 0.2645 0.2645 0.2773 0.2655 0.2673 0.2709 0.2682 0.2627 0.2627 0.2655 0.2691 0.2664 0.2636 0.2727 0.2573 ];
xy1t = [0.2609 0.2573 0.2627 0.2673 0.2500 0.2600 0.2509 0.2655 0.2573 0.2645 0.2573 0.2555 0.2500 0.2673 0.2673 0.2600 0.2555 0.2518 0.2664 0.2527 0.2509 0.2618 0.2636 0.2600 0.2518 0.2655 0.2655 0.2645 0.2536 0.2491 ];
xy2t = [0.2664 0.2618 0.2673 0.2682 0.2618 0.2736 0.2764 0.2682 0.2655 0.2673 0.2736 0.2636 0.2609 0.2673 0.2682 0.2673 0.2682 0.2636 0.2764 0.2664 0.2682 0.2664 0.2582 0.2700 0.2655 0.2745 0.2664 0.2736 0.2764 0.2709 ];
xz1a = [0.3764 0.3745 0.3836 0.3782 0.3673 0.3727 0.3818 0.3682 0.3764 0.3791 0.3709 0.3773 0.3736 0.3700 0.3745 0.3782 0.3691 0.3718 0.3809 0.3682 0.3727 0.3773 0.3809 0.3791 0.3700 0.3745 0.3800 0.3664 0.3764 0.3818 ];
xz2a = [0.4055 0.4073 0.4018 0.4018 0.4018 0.4082 0.4045 0.4055 0.4073 0.4064 0.3991 0.4073 0.4055 0.4018 0.4018 0.4064 0.4018 0.4036 0.4009 0.3964 0.4055 0.4045 0.4109 0.4000 0.4036 0.3973 0.3973 0.4064 0.4064 0.4000 ];
xz1t = [0.3836 0.3900 0.3927 0.3800 0.3809 0.3873 0.3900 0.3782 0.3827 0.3873 0.3900 0.3791 0.3827 0.3900 0.3836 0.3800 0.3864 0.3873 0.3773 0.3791 0.3809 0.3818 0.3873 0.3882 0.3764 0.3809 0.3882 0.3891 0.3818 0.3809 ];
xz2t = [0.4018 0.4018 0.4055 0.3964 0.4045 0.4073 0.4064 0.3982 0.4036 0.4036 0.4000 0.4055 0.4036 0.4036 0.3955 0.4018 0.4018 0.3991 0.4000 0.4045 0.4064 0.3991 0.4027 0.3991 0.4036 0.4009 0.3945 0.3955 0.3945 0.3945 ];
yx1a = [-0.0082 0.0009 0.0091 0.0109 0.0091 -0.0064 -0.0100 -0.0073 -0.0055 -0.0045 -0.0082 -0.0064 -0.0064 0.0018 -0.0018 -0.0109 -0.0109 0.0118 0.0109 -0.0036 -0.0064 -0.0027 0.0045 -0.0027 -0.0018 -0.0082 -0.0091 -0.0109 -0.0100 -0.0073 ];
yx2a = [-0.0155 -0.0255 -0.0255 -0.0182 -0.0264 -0.0182 -0.0173 -0.0264 -0.0273 -0.0245 -0.0273 -0.0255 -0.0282 -0.0145 -0.0264 -0.0291 -0.0273 -0.0264 -0.0255 -0.0255 -0.0109 -0.0264 -0.0300 -0.0155 -0.0173 -0.0273 -0.0236 -0.0255 -0.0273 -0.0155 ];
yx1t = [-0.0718 -0.0691 -0.0600 -0.0745 -0.0500 -0.0636 -0.0727 -0.0518 -0.0745 -0.0755 -0.0527 -0.0636 -0.0727 -0.0736 -0.0536 -0.0764 -0.0518 -0.0664 -0.0491 -0.0545 -0.0636 -0.0773 -0.0536 -0.0682 -0.0700 -0.0645 -0.0491 -0.0636 -0.0700 -0.0700 ];
yx2t = [-0.0636 -0.0618 -0.0591 -0.0545 -0.0500 -0.0445 -0.0545 -0.0464 -0.0536 -0.0436 -0.0527 -0.0436 -0.0573 -0.0555 -0.0473 -0.0491 -0.0555 -0.0409 -0.0427 -0.0564 -0.0491 -0.0545 -0.0382 -0.0445 -0.0609 -0.0482 -0.0555 -0.0500 -0.0527 -0.0373 ];
yy1a = [xy1a];
yy2a = [xy2a];
yy1t = [-1.1491 -1.1555 -1.1609 -1.1691 -1.1409 -1.1864 -1.2255 -1.2055 -1.2155 -1.1427 -1.1391 -1.2264 -1.1582 -1.1964 -1.2182 -1.1400 -1.1418 -1.1891 -1.1991 -1.1618 -1.1782 -1.1882 -1.2018 -1.1555 -1.1964 -1.2155 -1.1391 -1.1936 -1.1664 -1.1445 ];
yy2t = [-1.1382 -1.1627 -1.1300 -1.1373 -1.1191 -1.1727 -1.1082 -1.1764 -1.1473 -1.1555 -1.1536 -1.1364 -1.1218 -1.1291 -1.1127 -1.1209 -1.1473 -1.1145 -1.1164 -1.1682 -1.1209 -1.1455 -1.1409 -1.1282 -1.1018 -1.0991 -1.1109 -1.1527 -1.1173 -1.1545 ];
yz1a = [xz1a];
yz2a = [xz2a];
yz1t = [0.3945 0.4045 0.3973 0.3945 0.3973 0.3909 0.3955 0.4045 0.3955 0.3955 0.4018 0.3918 0.3982 0.4055 0.3927 0.3991 0.3982 0.4036 0.3900 0.4055 0.3918 0.3991 0.3973 0.4027 0.3891 0.3936 0.4018 0.3918 0.4000 0.3991 ];
yz2t = [0.5100 0.5145 0.5045 0.5127 0.5118 0.5064 0.5082 0.5109 0.5055 0.5164 0.5073 0.5145 0.5100 0.5091 0.5036 0.5055 0.5173 0.5082 0.5136 0.5127 0.5118 0.5118 0.5036 0.5091 0.5127 0.5073 0.5036 0.5082 0.5173 0.5118 ];
```

```
zx1a = [yx1a];
zx2a = [yx2a];
zx1t = [-0.0373 -0.0345 -0.0318 -0.0136 -0.0109 -0.0145 -0.0145 -0.0145 -0.0145 -0.0136 -0.0118 -0.0336 -0.0327 -0.0345 -0.0155 -0.0336 -0.0382 -0.0336 -0.0273 -0.0255 -0.0300 -0.0309 -0.0318 -0.0300 -0.0309 -0.0273 -0.0300 -0.0291 -0.0300 -0.0282 ];
zx2t = [0.0464 0.0355 0.0464 0.0418 0.0418 0.0418 0.0527 0.0345 0.0436 0.0373 0.0355 0.0355 0.0409 0.0391 0.0418 0.0518 0.0418 0.0445 0.0409 0.0473 0.0436 0.0427 0.0364 0.0391 0.0345 0.0355 0.0373 0.0491 0.0391 0.0500 ];
zy1a = [xy1a];
zy2a = [xy2a];
zy1t = [0.2355 0.2236 0.2400 0.2227 0.2273 0.2364 0.2273 0.2309 0.2300 0.2382 0.2345 0.2282 0.2345 0.2436 0.2318 0.2291 0.2282 0.2345 0.2255 0.2436 0.2291 0.2336 0.2300 0.2309 0.2282 0.2418 0.2364 0.2300 0.2382 0.2345 ];
zy2t = [0.3382 0.3427 0.3364 0.3464 0.3391 0.3373 0.3364 0.3491 0.3418 0.3445 0.3455 0.3473 0.3436 0.3400 0.3482 0.3436 0.3364 0.3427 0.3473 0.3400 0.3491 0.3464 0.3418 0.3391 0.3391 0.3336 0.3400 0.3418 0.3445 0.3400 ];
zz1a = [0.3718 0.3773 0.3773 0.3773 0.3691 0.3764 0.3809 0.3664 0.3727 0.3773 0.3691 0.3691 0.3764 0.3745 0.3718 0.3773 0.3827 0.3673 0.3700 0.3745 0.3709 0.3682 0.3727 0.3818 0.3673 0.3745 0.3791 0.3709 0.3773 0.3673 ];
zz2a = [0.4291 0.4273 0.4327 0.4309 0.4327 0.4327 0.4300 0.4309 0.4282 0.4300 0.4382 0.4327 0.4318 0.4282 0.4327 0.4318 0.4291 0.4355 0.4318 0.4300 0.4309 0.4327 0.4300 0.4300 0.4318 0.4300 0.4345 0.4336 0.4300 0.4273 ];
zz1t = [1.7223 1.7464 1.7609 1.7613 1.7466 1.7682 1.7669 1.7537 1.7482 1.7463 1.7693 1.7557 1.7531 1.7688 1.7312 1.7418 1.7172 1.7694 1.7606 1.7572 1.7765 1.7557 1.7602 1.7634 1.7747 1.7699 1.7620 1.7575 1.7567 1.7516];
zz2t = [1.7627 1.7809 1.7664 1.7700 1.7691 1.7655 1.7555 1.7573 1.7673 1.7691 1.7609 1.7591 1.7673 1.7755 1.7755 1.7709 1.7609 1.7545 1.7545 1.7491 1.7482 1.7564 1.7636 1.7555 1.7518 1.7591 1.7727 1.7645 1.7664 1.7664 ];
```

Check vector length

```
LENxx1a = length(xx1a)
LENxx2a = length(xx2a)
LENxx1t = length(xx1t)
LENxx2t = length(xx2t)
LENxy1a = length(xy1a)
LENxy2a = length(xy2a)
LENxy1t = length(xy1t)
LENxy2t = length(xy2t)
LENxz1a = length(xz1a)
LENxz2a = length(xz2a)
LENxz1t = length(xz1t)
LENxz2t = length(xz2t)
LENyx1a = length(yx1a)
LENyx2a = length(yx2a)
LENyx1t = length(yx1t)
LENyx2t = length(yx2t)
LENyy1a = length(yy1a)
LENyy2a = length(yy2a)
LENyy1t = length(yy1t)
LENyy2t = length(yy2t)
LENyz1a = length(yz1a)
LENyz2a = length(yz2a)
LENyz1t = length(yz1t)
LENyz2t = length(yz2t)
LENzx1a = length(zx1a)
LENzx2a = length(zx2a)
LENzx1t = length(zx1t)
LENzx2t = length(zx2t)
LENzy1a = length(zy1a)
LENzy2a = length(zy2a)
LENzy1t = length(zy1t)
LENzy2t = length(zy2t)
LENzz1a = length(zz1a)
LENzz2a = length(zz2a)
LENzz1t = length(zz1t)
LENzz2t = length(zz2t)
```

```
LENxx1a =
30
LENxx2a =
30
```

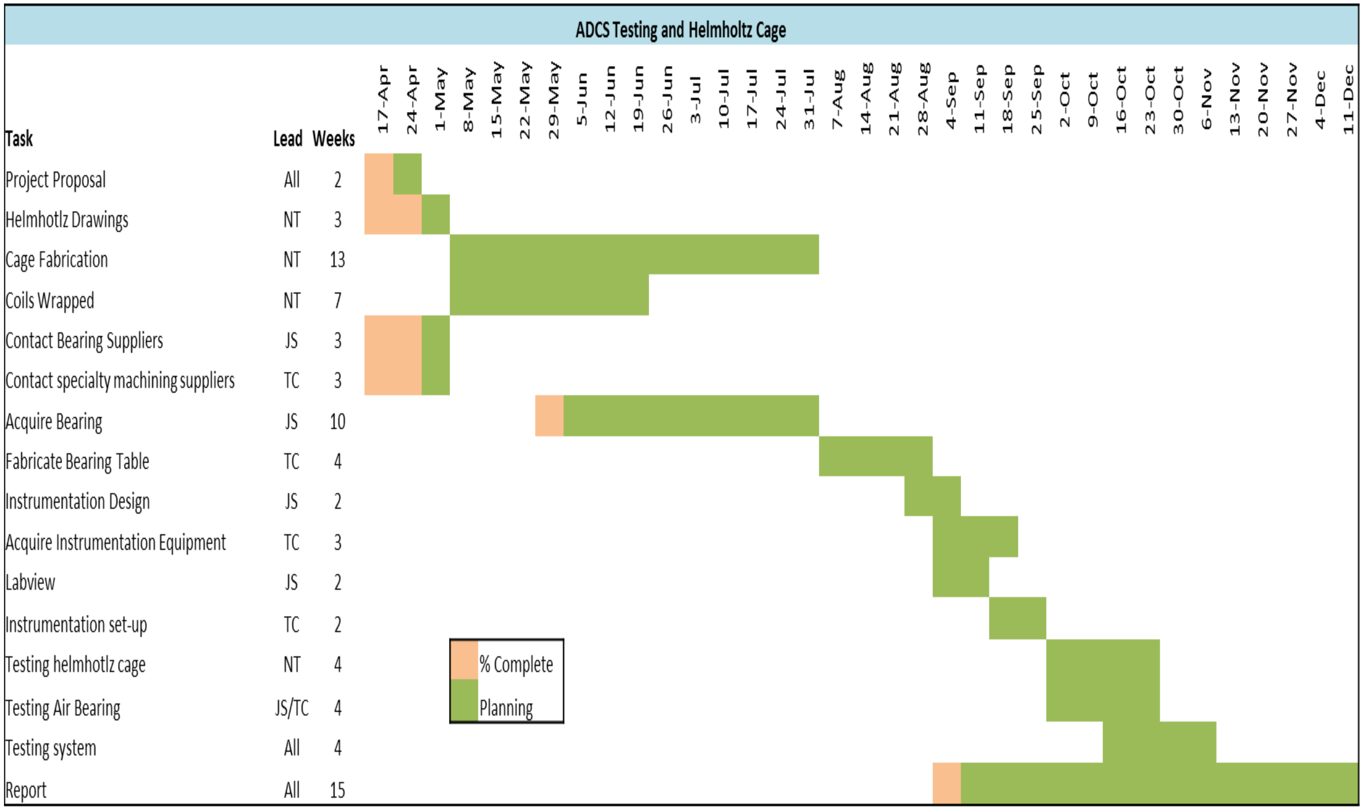
```
LENxx1t =
30
LENxx2t =
30
LENxy1a =
30
LENxy2a =
30
LENxy1t =
30
LENxy2t =
30
LENxz1a =
30
LENxz2a =
30
LENxz2a =
30
LENxz1t =
30
LENxz2t =
30
LENyx1a =
30
LENyx2a =
30
LENyx1t =
30
LENyx2t =
30
LENyy1a =
30
LENyy2a =
30
LENyy1t =
30
LENyy2t =
30
LENyz1a =
30
LENyz2a =
30
LENyz1t =
30
LENyz2t =
30
LENzx1a =
30
LENzx2a =
30
LENzx1t =
30
LENzx2t =
30
LENzy1a =
30
LENzy2a =
30
LENzy1t =
30
LENzy2t =
30
LENzz1a =
30
LENzz2a =
30
LENzz1t =
30
LENzz2t =
30
```

Published with MATLAB® R2014a

Appendix Q: Uniform Magnetic Field Testing Procedure

1. Center sensing apparatus along fixed x-axis on test stand
2. Connect magnetometers to Arduino’s
3. Reset Arduino
4. Record ambient magnetic field measurements for 30 seconds
5. Reset Arduino
6. Provide 7 AMPS of current to X-coils
7. Record magnetic field measurements for 30 seconds
8. Turn off power to coils
9. Reset Arduino
10. Provide 7 AMPS of current to Y-coils
11. Record magnetic field measurements for 30 seconds
12. Reset Arduino
13. Provide 7 AMPS of current to Z-coils
14. Record magnetic field measurements for 30 seconds
15. Reset Arduino
16. Center sensing apparatus along fixed y-axis on test stand
17. Repeat steps 2 – 14
18. Center sensing apparatus along fixed z-axis on test stand
19. Repeat steps 2 – 14

Appendix R: Gantt Chart



ADCS Testing and Helmholtz Cage

Task	Lead	Weeks	17-Apr	24-Apr	1-May	8-May	15-May	22-May	29-May	5-Jun	12-Jun	19-Jun	26-Jun	3-Jul	10-Jul	17-Jul	24-Jul	31-Jul	7-Aug	14-Aug	21-Aug	28-Aug	4-Sep	11-Sep	18-Sep	25-Sep	2-Oct	9-Oct	16-Oct	23-Oct	30-Oct	6-Nov	13-Nov	20-Nov	27-Nov	4-Dec	11-Dec
Project Proposal	All	2	🟧	🟩																																	
Helmhotlz Drawings	NT	3	🟧	🟧	🟩																																
Cage Fabrication	NT	13				🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩																			
Coils Wrapped	NT	7				🟩	🟩	🟩	🟩	🟩	🟩	🟩																									
Contact Bearing Suppliers	JS	3	🟧	🟧	🟩																																
Contact specialty machining suppliers	TC	3	🟧	🟧	🟩																																
Acquire Bearing	JS	10							🟧	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩																			
Fabricate Bearing Table	TC	4																	🟩	🟩	🟩	🟩															
Instrumentation Design	JS	2																				🟩	🟩														
Acquire Instrumentation Equipment	TC	3																					🟩	🟩	🟩												
Labview	JS	2																					🟩	🟩													
Instrumentation set-up	TC	2																							🟩	🟩											
Testing helmhotlz cage	NT	4																									🟩	🟩	🟩	🟩							
Testing Air Bearing	JS/TC	4																									🟩	🟩	🟩	🟩							
Testing system	All	4																											🟩	🟩	🟩	🟩					
Report	All	15																					🟧	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

🟧 % Complete
🟩 Planning

References

1. Farhat, A., Ivase, J., Lu, Y., Snapp, A. (2013). "Attitude Determination and Control System for CubeSat". Worcester Polytechnic Institute. p. 26-63.

2. Want, P., Shtessel, Y., Wang, Y. (1998). "Satellite Attitude Control Using only Magetorquers". University of Alabama. IEEE. p. 500-504.

3. Nicholas, A., Miller, D. (2013). "Attitude and Formation Control Design and System Simulation for a Three-Satellite CubeSat Mission". Massachusetts Institute of Technology.

4. Leomanni, M. (2012). "Comparison of Control Laws for Magnetic Detumbling". University of Siena. Research Gate. https://www.researchgate.net/publication/263008407_Comparison_of_Control_Laws_for_Magnetic_Detumbling

5. Candini, G., Piergentili, F., Santoni, F. (2012). "Miniaturized Attitude Control System for Nanosatellite". Elsevier. *Acta Astronautica, 81*. p. 325-334.

6. Ousaloo, H. (2014). "Magnetic Attitude Control of Dynamically Unbalanced Spinning Spacecraft during Orbit Raising". *American Society of Civil Engineers*. DOI: 10.1061/(ASCE)AS.1943-5525.0000252.

7. Poppenk, F., Amini, R., Brouwer, G. (2004). "Design and Application of a Helmholtz Cage for Testing Nano-Satellites*". ESA-ESTEC 6th International Symposium on Environmental Testing for Space Programmes.*

8. Piergentili, F., Candini, G., Zannoni, M. (2007). "Design, Manufacturing, and Test of a Real-Time, Three-Axis Magnetic Field Simulator". University of Bologna. *IEEE Transactions on Aerospace and Electronic Systems. Volume 47, No. 2.*

9. Taraba, M. (2007). " Boeing's Cubesat Testbed 1 Attitude Determination Design and On-Orbit Experience". *Conference on Small Satellites, 23rd Annual AIAA/USU.*

10. Brewer, M. (2012). "Cubesat Attitude Determination and Helmholtz Cage Design". Air Force Institute of Technology. *AFIT/GAE/ENY/12-M03.*

11. Foley, J. (2012). "Calibration and Characterization of Cubesat Magnetic Sensors Using a Helmholtz Cage". California Polytechnic State University.

12. Pastena, M., Grassi, M. (2002). "Optimum Design of a Three-Axis Magnetic Field Simulator". University of Naples. *IEEE Transactions on Aerospace and Electronic Systems, Vol. 38, No. 2*

13. Klesh, A., Seagraves, S., Bennett, M., et al. (2009). "Dynamically Driven Helmholtz Cage for Experimental Magnetic Attitude Determination". Radio Aurora Explorer. University of Michigan.

14. Cubesat Specifications. Rev 13. The Cubesat Program. Cal Poly SLO.

15. Gerhardt, D., Palo, S. (2010). "Passicve Magnetic Attitude Control for Cubesat Spacecraft". University of Boulder Colorado. *24th Annual AIAA/USU Conference on Small Satellites. SSC10.*

16. Alamgir, A., Fang, J., Gu, C., Han, Z. (2005). "Square Helmholtz Coil with Homogeneous Field for Magnetic Measurement of Longer HTS Tapes". Elsevier. Science Direct, Physica C, 424. DOI: 10.1016/j.physc.2005.04.019.

17. Dontchev, K., Heckathorn C., Light, M., Morris, S., Wloszek, P. (2008). "Fabrication of a 3-D Helmholtz Cage for University of Michigan NanoSat Pipeline". University of Michigan. *Aero 405, Proffesor Geister Class.*

18. Nelson PI. (2015). "PIglide HB: Hemispherical Air Bearing" http://www.pi-usa.us/products/Air_Bearing_Stages/PIglide_HB_Spherical_Air_Bearing.pdf

F1: reference 2
F2: reference 2
F3: https://upload.wikimedia.org/wikipedia/commons/8/8d/Helmholtz_coils.png
F4: https://www.cst.com/Content/Media/academia/Helmholtz-Coil-B-Field.png