Design, Analysis, and Control of a Nitinol Shape Memory Alloy Rotary Actuator for Spacecraft Deployable Structures

by

Mario Melendrez Contreras

Submitted to the
Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology

June 2019

Signature redacted

Signature of Author: _____
Department of Mechanical Engineering
May 13, 2019

Signature redacted

Certified by: _____
Kerri Cahoy
Professor of Aeronautics and Astronautics
Thesis Supervisor

Signature redacted

Accepted by: _____
Maria Yang, PhD
Professor of Mechanical Engineering
Undergraduate Officer

Design, Analysis, and Control of a Nitinol Shape Memory Alloy Rotary Actuator for Spacecraft Deployable Structures

by

Mario Melendrez Contreras

Submitted to the Department of Mechanical Engineering
on May 13, 2019 in Partial Fulfillment of the
Requirements for the Degree of

Bachelor of Science in Mechanical Engineering

Abstract

Small satellites known as CubeSats are becoming more and more popular in the aerospace industry and in academia. The new availability of rockets such as SpaceX's Falcon 9 or even dedicated CubeSat rockets such as Rocket Lab's Electron rocket have provided a new opportunity for many organizations to launch satellites. Depending on the goals of each satellite, they can be configured with many different payloads and mechanisms. Solar panels are one of the most common payloads on CubeSats but are mostly spring-actuated, meaning they cannot be deployed to precise angles. Shape memory alloys have been used to create rotary mechanisms in the past but closed loop control of shape memory alloys in a bending architecture is relatively novel. A rotary shape memory alloy actuator was designed with the use case of precisely pointing solar panels to maximize energy collection. Here we show identification of a system transfer function through multiple step responses and the use of a closed-loop PID control to achieve rise times of about 15 seconds with overshoot errors of 2 to 8 degrees. The experiments also showed the possibility of achieving rapid rise times of less than 2 seconds and accuracy within 2 degrees with some slight changes to the control system. This actuator prototype further develops the possibilities of precision angular actuation in a lightweight, robust, low volume, low power, and simple mechanical system.

Thesis Supervisor: Kerri Cahoy
Title: Professor of Aeronautics and Astronautics

Acknowledgements

I would like to thank the many people who helped me on this long journey of writing my Bachelor's thesis. First, I would like to thank Professor Kerri Cahoy who gave me the amazing opportunity of working on this interesting and exciting project in her lab and her feedback on this thesis. I also would like to thank the FLAPS team for all their hard work and collaboration on this project. I would like to thank Paula do Vale Pereira for her help and hard work in designing revisions to the hinge, helping to manufacture all the prototypes, and brainstorming ideas; Katie Chun for developing the annealing cycles for the hinge, machining, interfacing with the Zero-G staff, and leading the team through meetings; Charles Lindsay for being an indispensable member of the team by designing almost all of the electronic system and helping me write the electronics section of this thesis; Shreeyam Kacker for his help on designing the electronics, writing the Arduino control software, specifying components, and giving advice for the controls system; and Raymond Huffman for developing a data logging scheme. I would also like to thank Christian Haughwout for his initial advice and moving the FLAPS team off the ground and into the project. I also want to give a huge thanks to Professor Harrison Chin of the MIT Mechanical Engineering Department. He gave a huge amount of advice on characterizing the system and advice on how to implement a controller. Without him, this project likely could not have progressed to the point it did. I would also like to thank Franz Hover from the MIT Mechanical Engineering Department for inspiring my interests in controls. His animated lectures made me realize how interesting and useful the subject is. Finally, I would like to thank The Aerospace Corporation for funding this project. Without their help, this project could not have happened.

I would also like to thank my family. To my mother and father, I will never forget your struggles in moving to a different country and I hope to one day have as much courage as you two. All of your sacrifices got me where I am today and I am proud to say that I am your son. Los quiero mucho. To my sister Abby who helped care for me growing up, I appreciate all you've ever done for me and I love you. To my two brothers Sebastian and Abraham, I hope to show you both that you can accomplish anything you set your mind to and to never give up. Something you can always have despite your circumstances is the will and determination to succeed. Échale ganas.

Table of Contents

Table of Figures

List of Tables

1. Introduction

Small satellites known as CubeSats are becoming more and more popular in the aerospace industry and in academia. The new availability of rockets such as SpaceX's Falcon 9 or even dedicated smallsat rockets such as Rocket Lab's Electron rocket have provided a new opportunity for many organizations to launch satellites [1,2]. Depending on the goals of each satellite, they can be configured with different payloads and mechanisms. Currently, one of the most common actuation mechanisms for satellites is to attach deployable solar panels on the sides of CubeSats. They are most commonly attached with spring hinges and are restrained by cords, such as Dyneema [3]. Once the cord is released or burned through, the spring actuates the solar panel to a fixed angle.

If an engineer wishes to actively control the solar panel, they are currently limited to only a few options. Motors are widely understood and easily controllable, but introduce complexity to a satellite and can have backlash. In addition to this, their mass and volume may be significant. Motors can require up to a few watts depending on how much torque or speed is needed for the operation.

Shape memory alloys, or SMAs, are metal alloys that can be "programmed" to remember specific shapes or orientations. They can actuate to these programmed shapes by raising their temperature, which is often done through Joule heating. Using SMAs can decrease the mass of a system, lower complexity, and lower power requirements [4]. In addition to this, shape-memory alloys provide high force per volume, which makes them very useful for mass-constrained launch vehicles [4]. Since they function without moving parts, they are well-adapted to space applications and do not require lubricants or other substances that could outgas to space to function [5]. It is also predictable how the actuator will move. SMAs are typically used in simple axial compression, allowing the designer to create complex movement through the use of power transmission elements. [5]. However, the physical implementation of shape-memory alloys as actuators is still relatively novel.

Shape-memory alloys have the ability to "remember" a certain trained orientation. Examples of alloys that exhibit the shape memory effect are Ni-Ti, Ti-Nb, Ti-Mo, Cu-Zn-Al, Cu-Al-Ni, and Fe-Mi-Si, with Ni-Ti or nickel titanium being one of the most common [6]. When the alloy is heated up to the austenitic phase, it returns to its trained orientation (the mechanism of this phenomenon is explained more thoroughly in Section 2). This action of returning to a certain trained orientation can be used for deployables on CubeSats, for example.

The use of shape-memory alloys may decrease the mass and power requirements of actuators, but also presents challenges. For example, the hysteresis and nonlinear nature of the alloys are difficult to control. In addition to this, the alloy throw, or actuation distance or angle, is limited depending on the method in which the shape memory alloy is trained. Typical shape memory alloys used for actuators actuate through axial compression; the low throw of this movement is difficult to convert to useable work in actuators. Traditional fracture due to fatigue can still occur in shape-memory alloys as the actuator will become constantly stressed and unstressed over its lifetime, although it occurs at longer lifetimes than traditional mechanical actuators. In addition to this, functional fatigue, or the loss of the shape memory effect can also occur. Thermal control is important for SMA actuators because an increase in ambient temperature will begin to actuate it; this means that unintended actuation can occur if a system is not well-designed.

SMA actuators that actuate through axial compression have been manufactured and controlled with relatively high speed and precision [7]. Actuators have been designed that allow continuous rotational motion, but most create this motion through mechanical linkages like pulleys [8]. SMA actuators that function through a bending architecture have been manufactured but have not been implemented with a controller so far [9]. A more in-depth review of previous SMA actuators is provided in Section 3.1.

This thesis aims to ***present a control scheme for controlling shape memory alloys that function through a bending architecture***. The background of SMAs, the mechanical and electronics design, the controller design, and results of the experimentation will be presented in this thesis.

Section 2, Background, will describe the use and effects of shape memory alloys in more depth. Section 3, Prior Research, will discuss the different shape memory alloy actuators in the literature and their control schemes, if any. Section 4, Mechanical Design, provides an explanation of the mechanical design of the actuator and the annealing cycle used. Section 5, Electronics Design, provides an overview on the scheme used for actuating the SMA and recording data. Section 6, Controller Design, goes into detail on the data analysis, modeling, and controller design done for the actuator. Section 7, Results and Discussion, provides a detailed analysis of the tests run with this actuator. Section 8, Conclusions, provides the key takeaways provided in this thesis. Section 9, Future Work, outlines the next steps that would help in designing a more advanced shape memory alloy rotary actuator. References are provided in Section 10 and the code used for data analysis is provided in the Appendix, Section 11.

2. Background

Shape memory alloys, as their name suggests, have the interesting property of "remembering" a certain programmed shape. This process is done through annealing the shape memory alloy at a specific annealing temperature. A summary of different SMAs and their transformation temperatures is provided in Table 1. SMAs can also exhibit a condition known as superelasticity. If annealed correctly, an SMA can elastically withstand large deformations that it would not be able to withstand if it were a more typical engineering material, such as steel. The properties of SMA are explained in Section 2.1.

Alloy	Range of Transformation Temperatures (°C)
AgCd	-190 ~ 15
CuAlNi	-140 ~ 100
NiTi	-50 ~ 110
FeMnSi	-200 ~ 150
NiAl	-180 ~ 100

Table 1. [10] Different transformation temperatures for different shape memory alloys.

For example, at NASA's Goddard Space Flight Center (GSFC), engineers are experimenting with rover wheels made out of NiTi SMAs that are trained to be in their superelastic state. The Mars Curiosity Rover's aluminum wheels are infamous for cracking and

failing under cyclic loading due to the rocky terrain of Mars. Engineers at NASA GSFC have designed and tested wheels made up of woven super elastic shape memory alloys. The alloys, which are in their super elastic state, allow large deflections so the rover can drive over sharp rocks and other objects without the wheels failing [11].

Shape memory alloys have also found their way into the medical field. Stents that are placed in a person's blood vessels and expand to prevent medical issues have been designed with SMAs. The alloys are trained to an "expanded" position and their transition temperature is tuned through annealing to be that of the surrounding biological tissue or fluid. Once trained, the SMA stent can be collapsed and inserted into the vessel or location of interest. After a while, the blood's forced convection over the stent or the ambient body temperature triggers the shape memory effect and opens the stent [12].

2.1 Shape Memory Alloys and their Effects

Shape memory alloys (SMAs) exhibit two main effects. The first is the shape memory effect and the second is superelasticity (also referred to as pseudo-elasticity). In the shape memory effect, the SMA is able to remember one or two different "trained" states. This effect can be explained by examining the shape memory alloy material structure. The material structure will also explain the phenomenon of superelasticity, which allows large deflections to occur in the alloy.

A nickel titanium shape memory alloy, or NiTi alloy, exists in two main phases: martensite and austenite. At a lower temperature the NiTi exists in the martensite phase. When heated, the NiTi begins to transform into austenite. A good reference for this transformation process is given in Figure 1. This phase change begins at the *austenite start temperature,* denoted by A_s. The temperature at which the material fully transforms into austenite is known as the *austenite finish temperature,* or A_f[13]. When austenitic NiTi begins to cool, it begins to return to the martensite phase. This process begins at the *martensite start temperature*, M_s, and is completed at the *martensite finish temperature*, M_f [13]. Despite returning to the martensite phase, the metal alloy itself remains in the orientation it was in when it reached the austenite finish temperature. Additional force is required to deform the alloy to a new orientation. This principle can be exploited in antagonistic actuator architectures to create useable work.

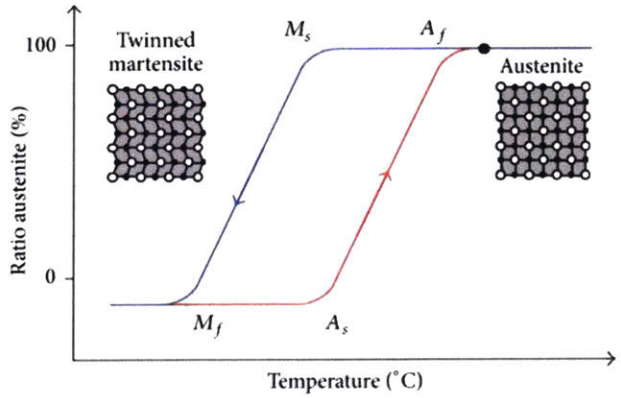

Figure 1. *[14]* SMA transformation hysteresis with transition temperatures. Transition temperature values depend on specific alloy composition.

The properties of the specific shape memory alloy, such as start and finish temperature, can be tuned through the alloy itself and the annealing cycle. One of the most common shape memory alloys used is a Nickel Titanium alloy, coined Nitinol, that consists of a 50% nickel 50% titanium alloying. Nitinol is the specific shape memory alloy used in this thesis due to availability.

An SMA can be annealed at high temperatures to a desired shape. Once the SMA's temperature hits the austenite start temperature, the SMA will begin to actuate to that desired shape. This principle is the basis of all shape memory alloy actuators. It is in the martensite to austenitic transformation that useful work can be done.

Super elasticity can occur when the austenite finish temperature is specifically tuned to the operating environment. For example, if the austenite finish temperature is tuned to be just beneath room temperature, when the alloy is loaded it will deform. When it is unloaded the shape memory effect will cause the alloy to return to its trained configuration, exhibiting super elasticity since the alloy will be able to return to its trained state despite large deflections.

SMAs can also be tuned to exhibit a two-way memory effect, which is to say that they remember both a "hot" and "cold" configuration. This thesis exclusively deals with one-way memory effect but there is a possibility that two-way shape memory can be used effectively to create actuators.

2.2 Shape-Memory Alloy Constitutive Laws

Researchers have created models to try to capture the characteristics of shape memory alloys. A great summary of shape-memory alloy constitutive laws is provided in Yee Harn Teh's doctoral thesis in Chapter 2.2.2: Modeling [7].

Tanaka proposed a model that related stress, strain, temperature, and martensite fraction [15]. These quantities were related by the following equation in the uniaxial case:

$$\dot{\sigma} = D\dot{\epsilon} + \theta\dot{T} + \Omega\dot{\xi} \qquad (Eq.\ 2.2.1)$$

where σ is the uniaxial stress, D is the uniaxial component of the elastic moduli tensor, ϵ is the uniaxial strain, θ is uniaxial component of the thermoelastic tensor, T is temperature, Ω is the uniaxial component of a transformation tensor, and ξ is an internal state variable measuring the ratio of martensite to austenite, known as the martensite fraction [15]. The dots on the variables in Eq. 2.2.1 indicate a time derivative.

It is possible to create a transfer function based off of this constitutive law for controls purposes, but it would be impractical because measuring temperature, strain, martensite fraction, and stress of a shape memory alloy would be difficult. In addition, measuring all of these variables increases complexity, which is to be avoided if possible. Teh avoided using a constitutive law for his controller in his doctoral thesis due to these impracticalities. Instead, he found great results in term of response time and accuracy by characterizing his system with a frequency response and implementing a force control loop within a larger position control loop [7].

3. Prior Research

There has been a large amount of research on SMA actuators and control schemes for them. A large majority of the research involves SMA actuators that actuate through a linear-range of motion, such as compression of an axial bar or a spring [7,16,17]. Rotary SMA actuators are less common but have been researched [8].

3.1 Past SMA Actuators and Controls

Song et al. implemented an active position controller for a composite beam [17]. The applications for this implementation range from active control of structures, such as antennas, to controlling the tips of rotorcraft blades to reduce vibrations. Song embedded a shape memory alloy off-axis on a composite beam so that when the current was sent into the shape memory alloy the SMA contracted and the beam flexed. Song used a robust compensator and PD control to achieve position control of tip position. The controller reached the desired tip displacement of 7.5mm within about 250 seconds but reached 90% of the desired value within 20 seconds [17]. The beam was 30.48 cm long, 5.08cm wide, and 1.32 cm thick.

Teh implemented position control, force control, and a differential force controller for an antagonistic SMA actuator architecture [7]. Teh used PID control to achieve fast position control response for his setup which had two contracting SMAs rotate a pulley. In addition to this, an anti-slack mechanism was implemented in the controller. This mechanism sends a constant power input to both SMAs to make sure both are kept tight and ready to actuate. A rapid heating mechanism was also implemented. Inputting too much power into the SMA has the ability to re-anneal the wires or damage them. Using a resistance curve as a proxy for temperature allows the control system to input as much power as possible without increasing to a temperature that can cause SMA damage [7]. Teh showed that it is possible to implement an accurate and fast control system using an antagonistic SMA wire setup where both wires contract.

Khatsenko designed and tested an open-loop SMA rotary actuator meant to deploy solar panels for CubeSats [9]. He showed that it is possible to use SMA sheets to actuate to large angles of about 90°. The specific implementation of the shape memory alloys induced torsion of the deployable structure but only a very small amount of about 0.27° [9].

Other rotary SMA actuators have also been developed by other researchers. Yuan et al. conducted an extensive review of rotary shape memory actuators [8]. Most of these actuators used an SMA in compression or an SMA spring applying force off-axis or at a lever to create rotational motion. A very interesting design was created by Hwang and Higuchi which uses multiple compressive SMA elements acting along pulleys to create a motor [18]. They were able to create a motor that was able to produce a maximum output torque of 2.6 Nmm [8,18].

4. Mechanical Design of Actuator

The goal of this thesis was to produce a closed-loop control actuator that is able to precisely deploy solar panels from -90° to 90° as depicted in Figure 2. A variety of shape memory alloy geometries and configurations were considered for the actuator. The two most common cross-section types for shape memory alloys are circular and rectangular. The circular cross section has been the most commonly used in actuators as a contracting element. The total throw for different designs was considered in the selection of the mechanical design for the actuator.

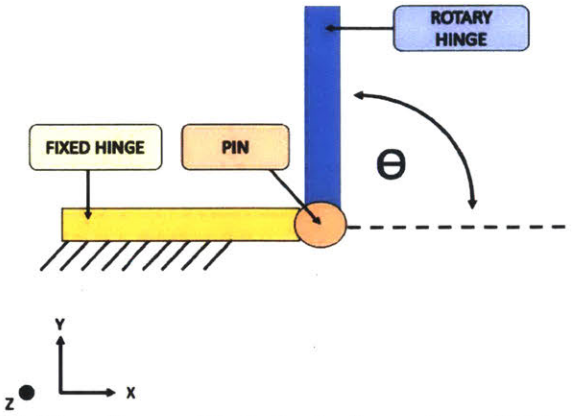

Figure 2. Schematic of SMA Hinge. Range of desired actuation angles is -90° to +90°.

4.1 Design Exploration

There are 3 common types of shape memory alloy actuators. Rectangular and circular cross section wires are the most common. These cross sections are used to actuate through compression of a member, producing useful work. Inducing motion solely through compression results in a low-throw due to the alloys not being able to infinitely compress. To induce rotary motion or more throw, the cross sections must actuate a lever or with a transmission system such as a pulley. SMA springs are also very common and are even sold commercially by vendors like Kellogs Research Lab [19]. Springs are more compressible and also more extensible than simple axial members so it is possible to get more throw with springs. The drawback to using a spring SMA is the fact that there is more length with actuation, so designs cannot be as compact as those that simply use rectangular and circular cross sections.

The actuator in this thesis is intended to rotate loads like solar panels in the space environment so multiple design architectures with different shape memory alloys are considered. Figures 3 demonstrates a rotary actuator design that functions through the use of pulleys and SMA wires. This concept is similar to the setup used by Teh in his dissertation [7]. Figure 4 shows a design similar to Khatsenko's hinge in that it uses an antagonistic SMA architecture that actuates in bending but it instead uses pins to limit the range of motion of the hinge to be purely circular [9]. Figure 5 depicts a design similar to Figure 4 but instead of using an antagonistic SMA architecture it uses solely one SMA actuating against a bias spring to create rotational motion.

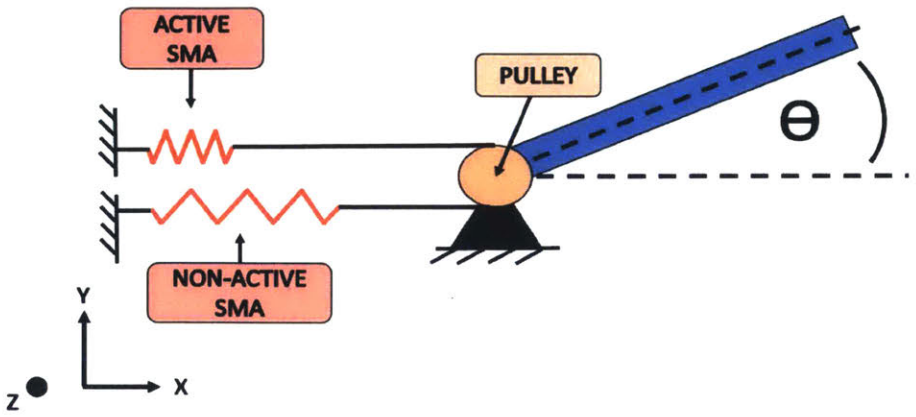

Figure 3. Schematic of antagonistic compressive SMA's attached to a pulley. As one SMA contracts the other is forcibly extended, inducing a rotation of the pulley and the hinge.

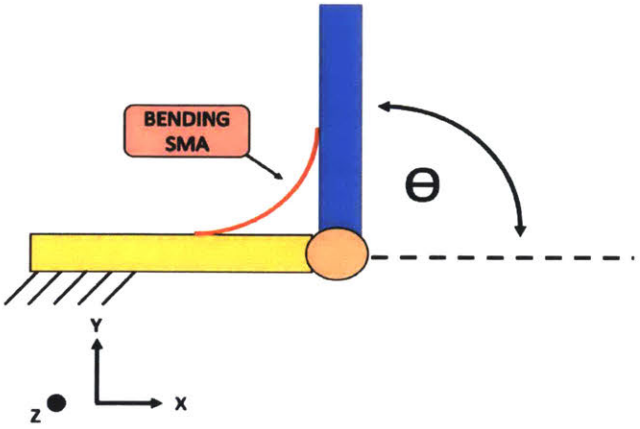

Figure 4. Hinge that utilizes an antagonistic bending SMA architecture. Two SMAs would be placed onto the hinge to actuate against each other.

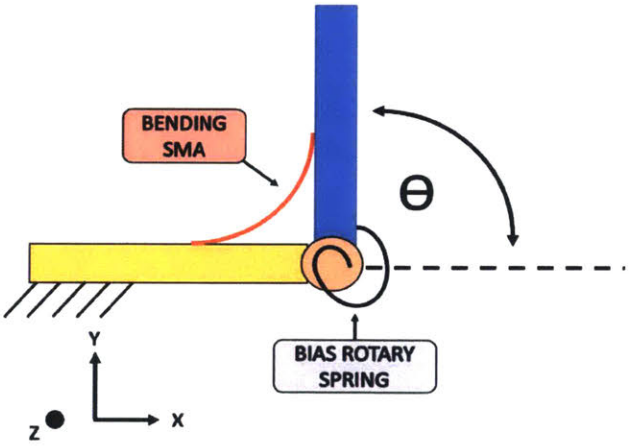

Figure 5. Hinge that utilizes a single SMA and a bias rotary spring.

After considering different SMAs on the market and different designs, we eliminated designs by applying the constraints of the intended use case in space. Ideally, this actuator would be 1) small-volume to minimize the mechanical footprint on the spacecraft, 2) low-weight to open up mass for other valuable instruments onboard, 3) low-power to be able to allocate more power to other instruments, and 4) be reusable.

Any design that uses spring SMAs would be a poor choice for space applications because an unwound spring generally has a large volume and is more massive than a shorter cylindrical or rectangular column. Designs involving pulleys or transmission systems are too complex and have an increased volume and mass. Designs that function through the use of a bias spring were also discarded due to the fact that the bias spring must be carefully tuned to ensure full range of motion. Having a bias spring also means that a constant current has to be supplied to the SMA to apply a force against the rotary spring. If the SMA is allowed to return to room temperature, it

keeps its orientation, but it essentially becomes a passive mechanical element with a bending stiffness specified by its cross-section geometry. A bias spring would therefore pull the spring back to an angle where the bias spring force and the SMA's effective spring force reach equilibrium. This property is not desirable for the application of active solar panel control but it is possible that for certain applications where a neutral unpowered position or angle is desirable, tuning a bias spring and the bending stiffness of an SMA could be an effective option.

These decisions finalized our design to an antagonistic bending SMA architecture, as depicted in Figure 4 and Figure 13. In addition to being mechanically simpler than a transmission system, using a bending architecture in theory allows us to achieve a high range of angles. Using a compressive SMA would require a large pulley to achieve the same range.

However, the cross-sectional geometry of the bending element still needs to be defined. Khatsenko's thesis showed that using SMA ribbon (thin, flat sheets) produced actuation angles of about 90° but seemed to require a large cross section of SMA to actuate [9]. Haughwout et al. created a functional open-loop prototype that could actuate to a full range of 180° (+90° to -90°) [20]. The design used 0.5mm diameter wires to actuate. However, there are some drawbacks to using cylindrical wires to actuate. The first is that it is difficult to establish a secure mechanical connection because the wires are so thin that a bolted solution is impractical and soldering is not a feasible option for Nitinol, the alloy that was chosen for use. The second drawback is that cylindrical wires tend to be stiffer than ribbons in bending.

If an area A is defined for both a rectangular cross-section and a cylindrical cross-section, then:

$$A_{ribbon} = bh \tag{4.1.1}$$

$$A_{cylinder} = \frac{\pi D^2}{4} \tag{4.1.2}$$

Where b is the width of the rectangle, h is the thickness of the rectangle, and D is the diameter of the cylindrical cross-section. The area moment of inertia I_r of a ribbon wire and the area moment of inertia I_c for the cylindrical wire are then:

$$I_r = \frac{bh^3}{12} = \frac{Ah^2}{12} \tag{4.1.3}$$

$$I_c = \frac{\pi D^4}{64} = \frac{\pi \left(\frac{4bh}{\pi}\right)^2}{64} = \frac{b^2h^2}{4\pi} = \frac{A^2}{4\pi} \tag{4.1.4}$$

From equations 4.1.3 and 4.1.4 it is possible to see that for a given area A, the area moment of inertia for a cylinder is fully defined but the inertia for a ribbon can still be tuned by adjusting the thickness. Normalizing for area shows that as long as h is less than some value $h_{critical}$ then $I_r < I_c$.

$$I_r \leq I_c \tag{4.1.5}$$

Plugging in 4.1.3 and 4.1.4 into 4.1.5 ultimately gives:

$$h_{critical} = \sqrt{\frac{3A}{\pi}} \tag{4.1.6}$$

However, area moment of inertia is not the only parameter that is important for actuating the SMA wires. The force each wire generates is also important for determining which geometry and cross-section is most optimal to use. Unfortunately, force or torque per area is not well-defined in the literature for SMA wires that actuate through bending. There is more literature on the force provided by SMA wires that actuate through compression, like the ones used in Teh's dissertation, but as stated previously SMA wires that actuate through compression are not ideal for this specific implementation due to volume and mass constraints and additional mechanical complexity [7].

Initially, the team decided that the actuator should use flat ribbon strips of wire since this would minimize the area moment of inertia. However, minimizing the thickness also minimizes the torque that the wire generates. Since this relationship was not well characterized at early stages the project moved forward using SMA ribbons. A benefit of using flat wire is that the strain in the wire is minimized due to having a smaller thickness than a cylindrical wire, which is good for life and actuation range. In addition, having a flat wire is also directionally easy to work with. If a cylindrical wire is annealed to bend in an "L" shape, it is impossible to tell from later inspection which direction the wire will bend because it is cylindrically symmetric. A flat wire has a clear stiff and non-stiff bending direction. A flat wire allows the designer to increase the width dimension b without forcing an expansion in the thickness dimension t, unlike a cylindrical wire which would expand radially

After the cross-section geometry and actuator architecture are defined, the next step is to determine the sensor feedback for the design. One early idea was to use a ranged time of flight sensor to determine the angle based on distance from a retroflector, as is shown in Figure 6. However, the implementation of the time of flight sensor introduces many complications in the development of a closed-loop rotary actuator. A simple magnetic encoder allows for the faster development of a control system, so a magnetic encoder was chosen for use. To function, a magnet is placed within the rotating object and the magnet's rotation is sensed by an encoder chip which outputs a linear voltage. This voltage can then be converted into an angle. The information on the encoder selected is provided in Section 4.2, Preliminary Mechanical Design. An image of the encoder on the prototype is provided in Figure 13.

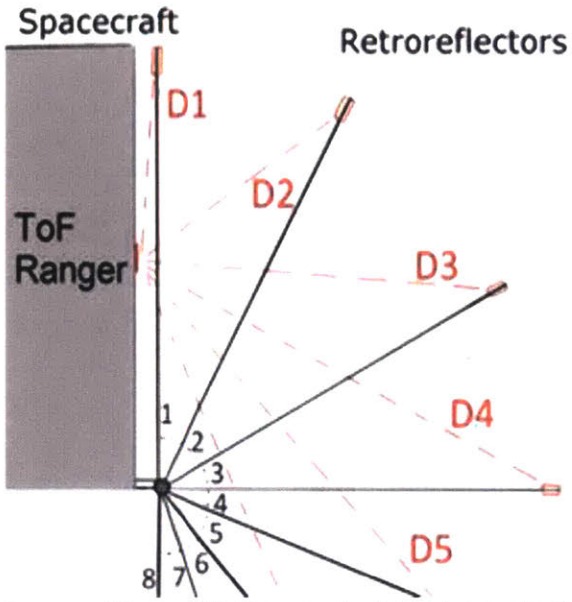

Figure 6. *[20]* Diagram showing use of time of flight sensor to determine angle of a hinge. If a known retroflector position is known, trigonometry can return the angle of the hinge.

Another design choice involves how to mount the shape memory alloys to the hinge itself. Unfortunately, soldering is not an ideal option for NiTi SMAs so using bolts as both the mechanical and electrical interface is chosen. This introduces uncertainty in the resistance of the electrical path due to not electrically connecting at the same point under the bolt head, but for prototype purposes this is acceptable.

4.2 Preliminary Mechanical Design

With the hinge architecture decided, the setup is modeled in SolidWorks as shown in Figures 7 and 11. The architecture consists of an antagonistic pair of flat SMA actuators controlling a hinge. The hinge itself has a pin to allow rotation and constrain x and y travel. An RM08 Super Small Non-Contact Rotary Encoder from Rotary and Linear Motion Sensors (RLS) was selected with 1024 counts per revolution, which provides a resolution of 0.35°, which is adequate for the prototype [21]. The part number of the encoder and magnet pair is RM08VA0010B02L2G00. The magnet is 3mm in diameter, 1mm in height, and has a mass of 0.4g. It is composed of SmCo and has a NiCuNi coating. The encoder body itself is 8mm in diameter, has a mass less than 0.2g, and is made out of aluminum. The encoder runs off 5V, so interfacing with it from an electrical standpoint is simple. The magnet for the encoder is integrated into the hinge with a recessed cavity so that the encoder can measure the angle.

The SMA wires were centered at the neutral axis of the hinge in order to minimize the strain on the wires. A cross-section of this geometry is shown in Figure 8. Pins take the x-direction and y-direction loads of the hinge in order to minimize loading on the SMA wires, which are very compliant. The SMAs are at a temperature of about 80°C when they reach their austenite finish temperature. A lightweight, low outgassing, thermally insulative material would be the best option for the hinge material given space applications. Initially PEEK and nylon were chosen as materials due to their material properties but both were difficult to quickly and precisely machine. Since the prototype was the goal, a 3D-printed hinge was produced using a

MarkForged 3D-printer. The Onyx material was chosen due to its high heat deflection temperature of about 145°C and great strength and stiffness properties [22].

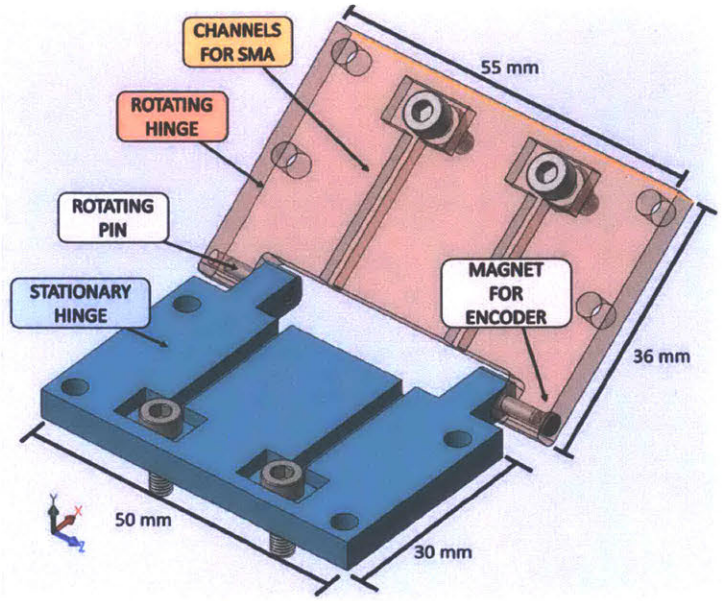

Figure 7. CAD image of version 1 of the SMA hinge. Channels keep SMA in the neutral axis of the hinge to minimize strain. Press fit pins allow hinge rotation. Rotating hinge has a counterbore for encoder magnet.

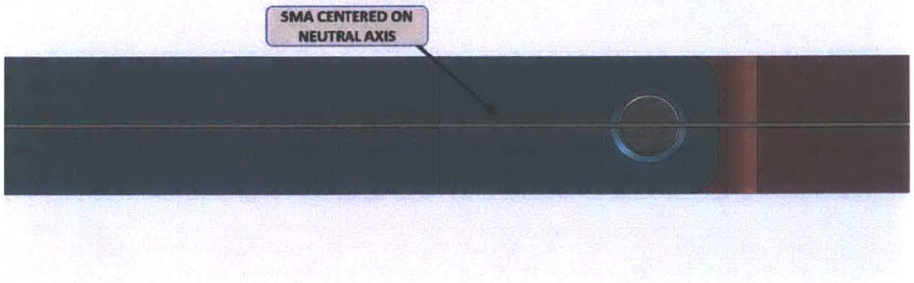

Figure 8. Cross-section view of SMA with hinge. SMA is aligned with the hinge's neutral axis to minimize strain.

A version 1 model of the hinge designed in SolidWorks is seen in Figure 7. When the hinge is flattened and at 0°, as shown in Figure 8, its footprint is approximately 55 mm by 70 mm. The thickness of both hinge elements is 4.5 mm. A prototype hinge was manufactured by 3D-printing the hinge components and purchasing the pins and bolts.

An almost fully assembled model is shown in Figure 9. After the assembly, trained SMAs were attached and actuated using a power supply. Alligator clips were attached to both ends of the SMA and a voltage was applied until the SMAs stopped actuating.

Due to the middle of the SMAs being unconstrained, the passive SMA ended up being buckled by the active SMA. A simple drawing of this situation is provided in Figure 10. Although this did not destroy the passive SMA, this deformation was not ideal and a redesign of the hinge ensued.

Figure 9. 3D-printed model of the version 1 hinge.

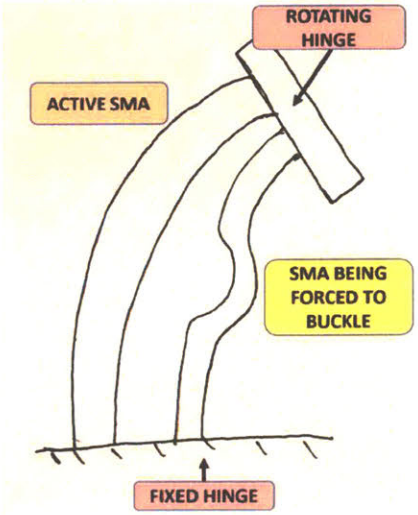

Figure 10. Drawing demonstrating buckling issue of antagonistic SMA wires.

4.3 Design Changes

The redesign included adding buckling straps to the current hinge that would help constrain the bending of the SMA in its midsection and decrease buckling. A computer design of the redesigned hinge is provided in Figure 11. A prototype hinge was 3D-printed in nylon since the MarkForged Onyx printer was down for repairs during this time. An image of this prototype is provided in Figure 12. The hinge had poor surface finish and dimensional accuracy which was not desirable.

More design changes were implemented during the down time of the MarkForged Onyx printer. Bushings were added to the pin-hinge interface to allow smoother rotation and the hinges were made to be symmetric to reduce costs when machining. The previous version without bushings would occasionally get stuck due to friction. The buckling strap was adjusted in width and in location to better prevent buckling. An updated prototype manufactured by 3D-printing with the Onyx material is shown in Figure 13. This version of the prototype hadgood functionality and was able to actuate further to about 18° whereas the original version without buckling straps was limited in range to about 10°. However, this prototype still lacked some key features that will be discussed in Future Work, Section 9.11.

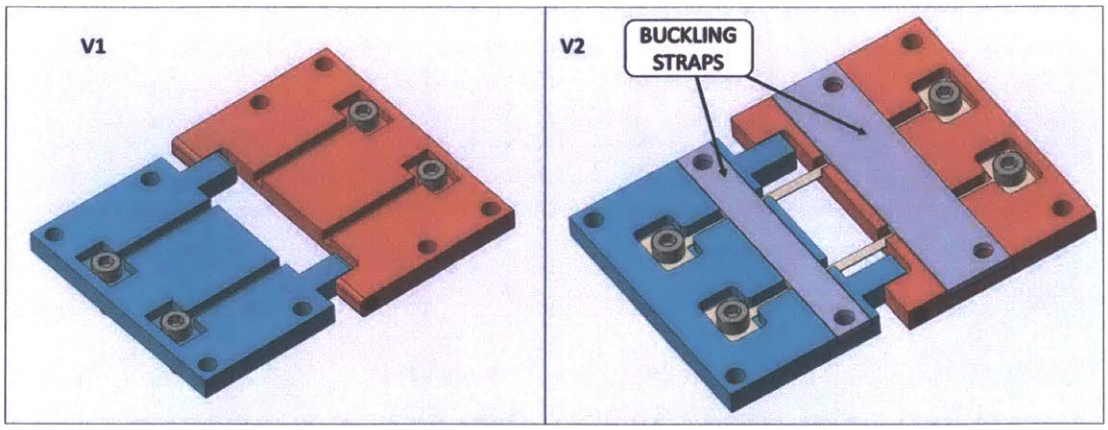

Figure 11. Comparison between Version 1 and Version 2 of SMA hinge. Version 2 includes buckling straps to prevent the buckling at the midsection of the SMAs.

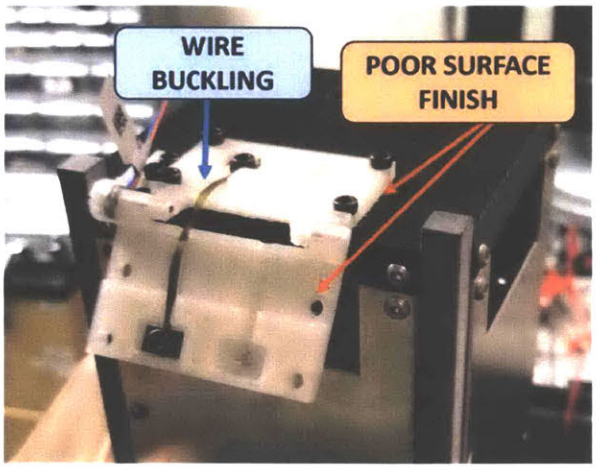

Figure 12. Version 2 hinge 3D-printed in nylon mounted on physical model of a CubeSat bus. The 3D-printing process with nylon results in poor surface finish and dimensional accuracy.

4.4 Annealing Cycle and Shape Memory Alloys

The annealing cycle used and the elemental composition of the shape memory alloy will affect what properties it has. The specific shape memory used in this thesis is Nitinol. Nitinol is a 50-50 split between nickel and titanium. Thin sheets of 0.25 mm Nitinol were sourced from Kellogs Research Laboratory [23]. The dimensions of the actuator are given in an engineering drawing in Figure 14.

Testing was conducted to determine the most optimal angle at which to train the SMAs. Multiple molds were machined out of 6061-T6 Aluminum and tested in a heat oven on campus, courtesy of the MIT Glass Lab. The range of the SMAs was tested by putting the SMAs in a hot cup of water and having them actuate. The angle was then inspected by eye. The mold that provided the best range was the 180° mold. An image of the molds tested is provided in Figure 15.

The annealing cycle consisted of having the actuators in the oven for 30 minutes at 500 °C (932°F). Sometimes when the annealing oven is opened the temperature will drop to

approximately 890°F but returns to 932°F within 5 minutes.

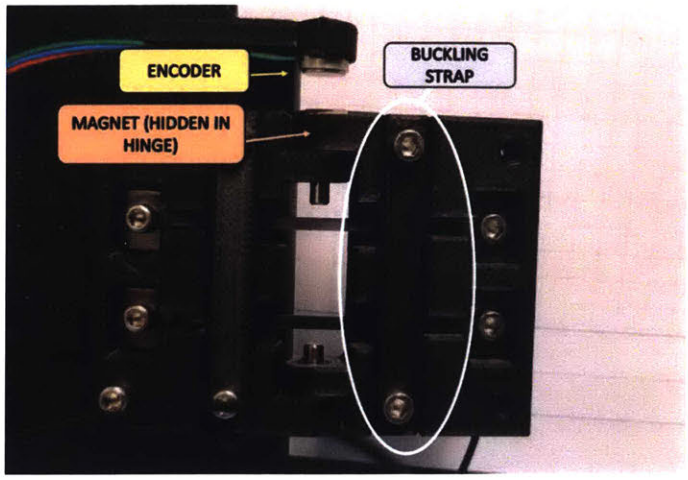

Figure 13. Version 3 of the hinge 3D-printed in Onyx.

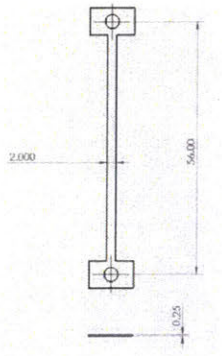

Figure 14. Drawing of Nitinol actuator. All units in mm.

5. Electronics Design of Actuator

The majority of the electronics design for the actuator was done by Charles Lindsay, a Class of 2021 Aeronautics and Astronautics student at MIT. His work is included in this thesis for reference. The electronic schematic is provided in Figure 16.

5.1 PCB and Electronics Layout

The main requirement of the electronics board is to deliver power to the SMAs. The SMAs have a relatively low resistance due to their high conductivity and cross section of 0.25 mm by 2 mm. Current delivered to the SMAs results in the Joule heating effect, increasing their temperature. A power distribution unit on the PCB takes in a voltage provided by the power supply in our lab setup. It steps down the voltage to both 1.8V and 5V to drive the magnetic encoder, accelerometer, and the main computer unit, which is an Arduino Uno. The accelerometer is included in the diagram because this prototype will be flown in a zero-g flight in order to test the hinge in its operational environment.

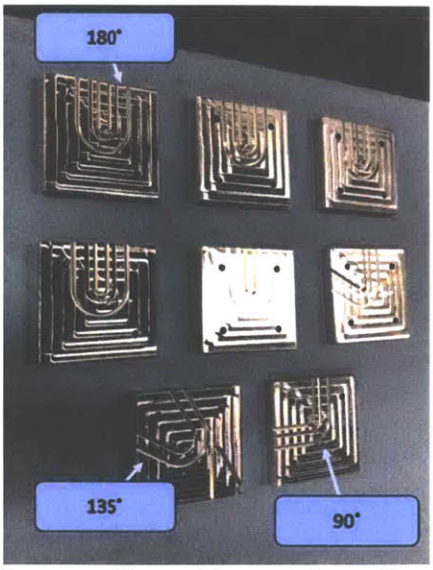

Figure 15. Image showing all the different annealing molds tested. It was found that 180° works best to achieve the intended activation angle. Image credit: Katie Chun

The power through the SMAs can be calculated by knowing the voltage across the SMA actuators and the current going through them. To achieve this, a current shunt of a known resistance is placed on the high side of the SMAs. By varying the duty cycle of the voltage input to the MOSFETs (seen next to I/O #1) it is possible to vary the average voltage across the SMA. The current coming from the 1.8V source must go through both the current shunt and the SMA and to ground. The voltage across the shunt is measured and because the shunt resistance is known, the current through the SMA can be calculated. The voltage across the SMA is calculated by measuring the voltage at the high end of the shunt relative to ground and by knowing the duty cycle. The resistances of all the wire leads are factored into these calculations. The resistance of the SMA is similar to the wire leads, which means the leads need to be included in the calculations. The calculations are done in real time by the Arduino. The control system is also programmed into the Arduino. The power calculation is then logged as data by the Arduino and will be used in future control schemes.

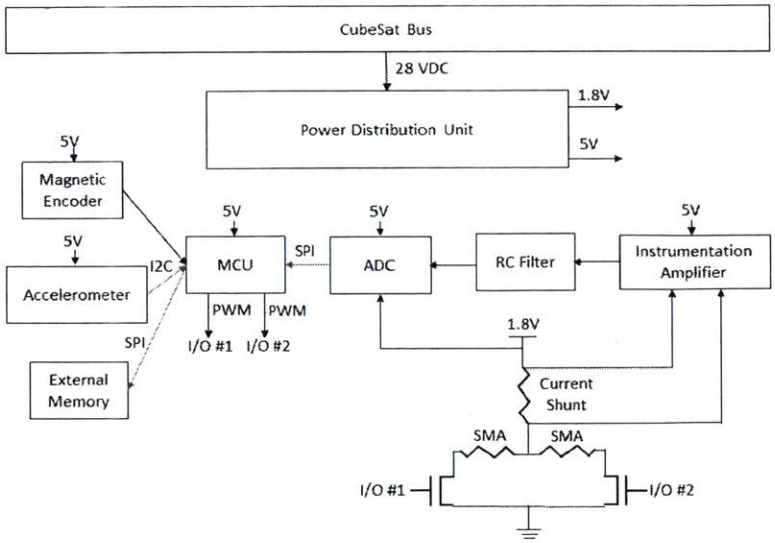

Figure 16. Overall electronic schematic. Image credit: Charles Lindsay

6. Controller Design of Actuator

The controller design of the actuator was based largely on Teh's dissertation [7]. Teh showed that for low power levels an SMA can be approximated to have a linear, time invariant transfer function. This allows for a simple controller design. Step responses were used to characterize the SMA plant. Different powers were sent to the plant and the hinge output angle was measured. A transfer function was defined from the step responses and used to find PID gains. Initial gains were found using MATLAB's robust response time algorithm. These initial gains were then adjusted using root-locus design methods. All the MATLAB code used for data analysis is provided in the Appendix. The Arduino control code is also provided in the Appendix.

6.1 Controller Design Approach

It can be assumed that for small power inputs the transfer function for an SMA is linear time invariant (LTI), which allows us to apply linear control theory. The main goal for the controller tuning is to have no overshoot and to respond as fast as possible. These goals come from the fact that this prototype hinge will be flown on a zero-g flight. The flight has periods of 15 seconds where gravity can be neglected. The actuator must be able to actuate to a set point within this time period in order to test the action of the hinge in a microgravity environment.

6.11 Determining the Transfer Function through a Step-Response

Step responses were measured by the electronic sub-systems. A constant duty cycle command was sent to the SMA through the Arduino which resulted in a varying power signal. The power signal varied due to the changing resistance of the SMA as it heats up. The output angle was measured using the magnetic encoder. Data was collected at a frequency of 285 Hz. Five trials each at 5 different duty cycles were done, which corresponds to 25 total tests. The averaged responses of these tests are shown in Figure 17, with outlier tests excluded. Examples of specific data trials are shown in Figures 18 and 19. The outliers that were excluded did not actuate to the same ballpark angle that tests at the same power level did, with errors from about 5

to 15 degrees. It is assumed that these angle issues were due to the buckling of the other SMA and excluded from the averaging.

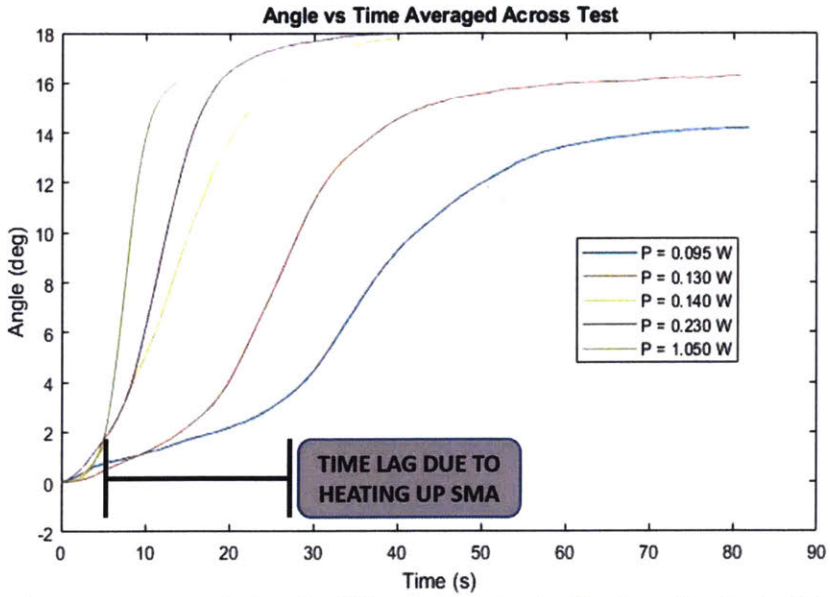

Figure 17. Averaged step response angle data for different power levels. Graph credit: Charles Lindsay

In order to determine a transfer function from step response data, an iddata object needed to be generated in MATLAB. A moving-average filter is applied to the data to smooth out the noise. 200 samples are used in the moving average. To fit a step-response, a system must reach steady state. Because of this, the data is artificially extended as seen in Figure 18 and Figure 19. This is acceptable because when the system reached steady state the data collection was stopped. After filtering, the data was then input into MATLAB to determine a transfer function.

At first, the transfer function was determined by assuming the system was a pure 2nd order system with two poles. After looking at plots of the raw data overlaid with the predicted step response of the estimated transfer function, it was determined that it was necessary to add a lag to compensate for the initial time delay due to the SMA heating up right before the austenite start temperature. The lag was added in the form of a pole-zero pair. The system model with the lag still exhibited a time delay so another lag was added. It was found that fitting the model with two lags (two extra poles and two zeros) produced a step response that was not significantly different from simply modeling with a 2nd order system of only two poles. Because of this increased complexity without much benefit, it was decided that modeling the transfer functions would continue with only two poles. Despite the transfer function not being the best fit, it was assumed that the error term in the controller would be able to account for this inaccuracy. Figure 20 shows a side-by-side of all three system models for a 16DC sample: 2nd order, 2nd order with one lag, and 2nd order with two lags.

It is also important to note that at lower power levels, the 2nd order fit worked better than at higher power levels. This is thought to be due to a non-linear response at higher power levels but more tests are needed to confirm this. In addition, at higher power levels some fits exhibited high levels of error. These data were not used during the controller analysis and design. Due to

this, the controller was tuned around the low-power operating condition at 16DC to better approximate a linear transfer function.

The transfer function determined by using a 2nd order fit is:

$$\theta(s) = \frac{1.144}{s^2 + 0.183s + 0.01597} \qquad (Eq.\,6.11.1)$$

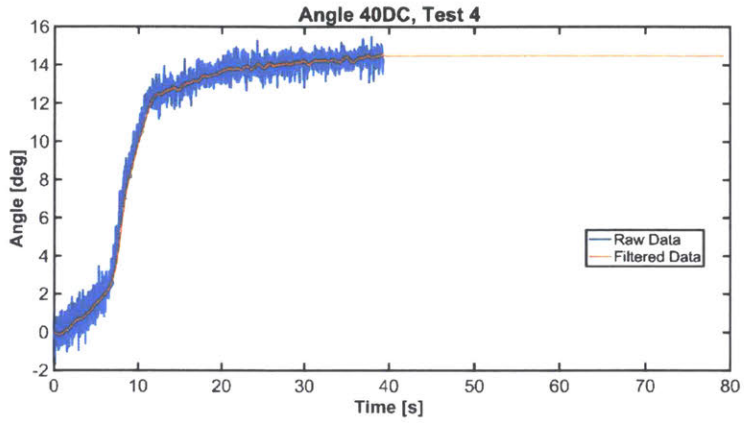

Figure 18. Plot of angle versus time for a duty cycle of 40%.

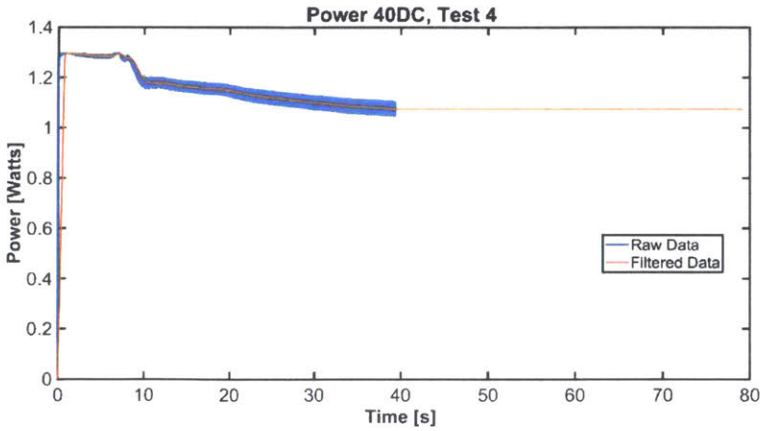

Figure 19. Plot of power versus time for a duty cycle of 40%.

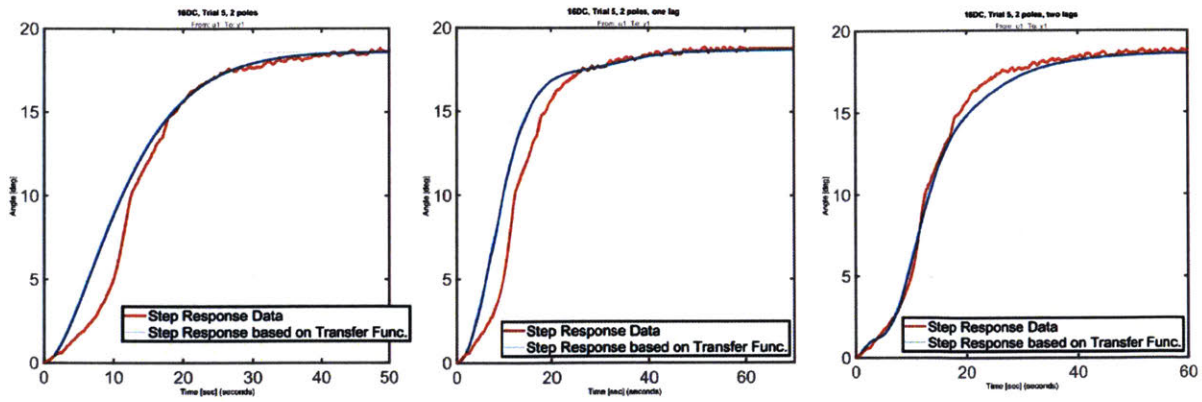

Figure 20. Side by side plot of three different fitting models for step response data.

6.12 PID Control and Gain Tuning

Since it was determined that the SMA transfer function could be approximated by a 2nd order system, PID control was used due to its prevalence in industry, ease of use, and ease of implementing as a discrete controller in Arduino. PID gains are found by auto-tuning with MATLAB's robust response time algorithm. The gains are tuned around the low-power operating condition of 16DC, about 0.14 Watts. The averaged test data was used for the tuning. The gains were then adjusted by a root locus tuning, which is provided in Figure 21. The simulated close-loop step response is provided in Figure 22. Gains were then solved for and recorded. The initial PID gains were: Kp = 0.04379, Ki = 0.002789, and Kd = 0.1442. A schematic for the control system is provided in Figure 23.

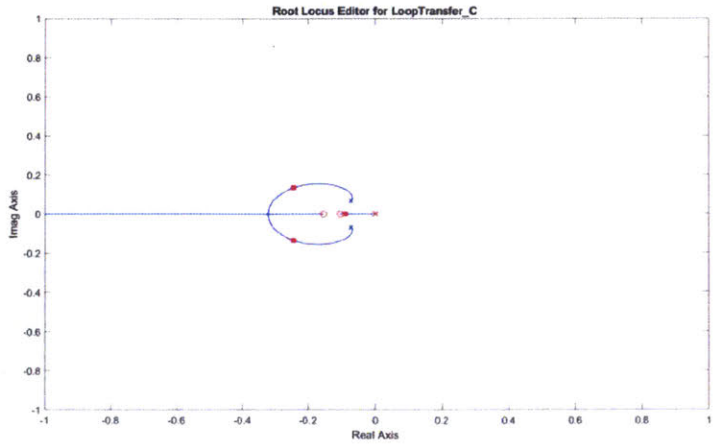

Figure 21. Open-loop root locus plot of controller and plant.

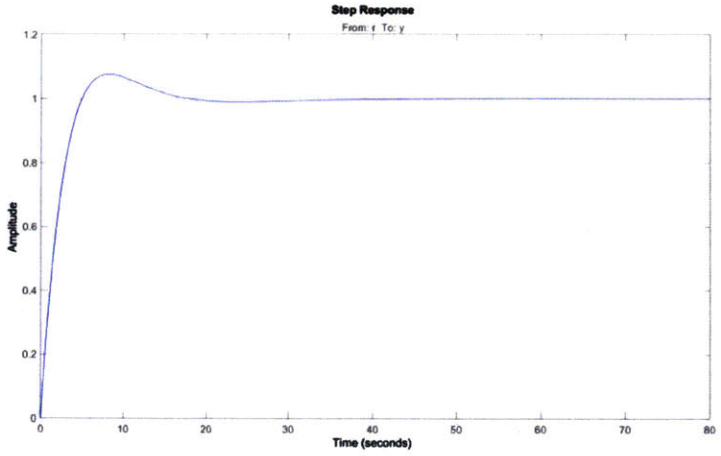

Figure 22. Closed-loop step response.

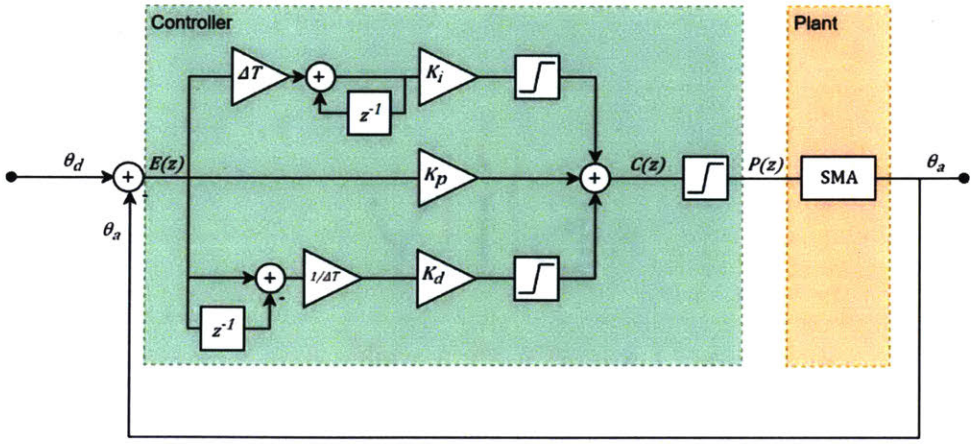

Figure 23. Controls system schematic. Image credit: Shreeyam Kacker

7. Results and Discussion

The conditions needed to maintain the linear time invariant assumption are discussed and explained. The tests to a controlled step response are analyzed and discussed. It is determined that with some modifications to the control system that a fast and accurate closed-loop controlled response is possible.

7.1 Verification of Linear Time Invariant Assumptions

As shown in Figure 17 there is a significant time lag component due to power input. The time lag is due to the SMA heating up before it reaches the austenite start temperature. With a higher power signal, the SMA will heat up faster, decreasing the time lag until actuation. This is clearly demonstrated in Figure 17. However, in the figure one can see that if the time lag is ignored, the output angle increases as the power level increases for low power levels. Eventually, the angle reaches a saturation point at higher power levels. In Figure 17, that saturation point is about 18°. This means that if one controls for the pre-activation temperature then the system can effectively be considered linear, time invariant up to the saturation angle. To control for the pre-activation temperature, a pre-heat control loop is recommended. This will be discussed further in Future Work, Section 9.13: Controls Design.

The saturation angle of 18° can be explained by mechanics. In the mechanical design section, it was assumed that a flat SMA wire would be less stiff in bending and therefore give more range of motion than a cylindrical wire. However, the opposing SMA is not constrained in a pure bending motion. Instead, it tends to be put into compression as the active SMA actuates. This can be seen in Figure 24. The compression stiffness of a member flat wire is much larger than the bending stiffness. The compression reaction force of the flat wire prevents the SMA from actuating past 18° as the active SMA does not have the necessary torque to continue to compress the SMA in order to rotate the hinge. It turns out that the SMA torque and the compression reaction force due to the opposing SMA find an equilibrium at around 18° for the power levels tested. The solution to this issue is described in Future Work, Section 9.11 Mechanical Design and 9.2 Characterizing SMA Torque.

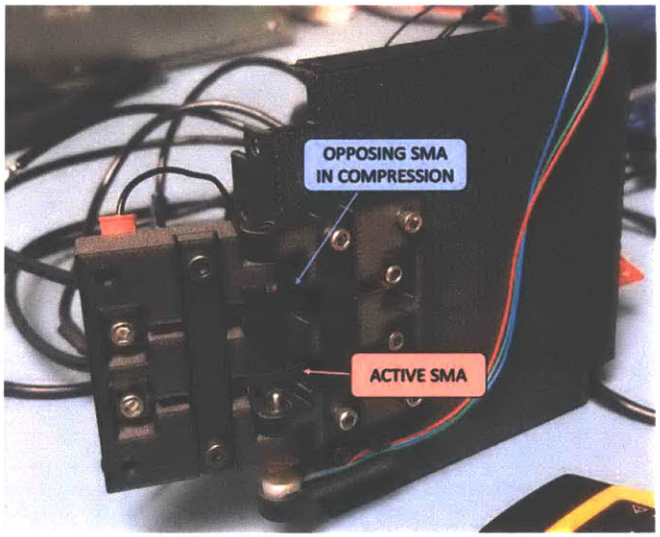

Figure 24. Picture of active SMA compressing opposing SMA. Note the curvature of the active SMA and the lack of curvature on the opposing SMA.

Once all the step responses were recorded and the data was averaged, a pole-zero plot was created to determine if the system could truly be considered LTI. The pole-zero plot is provided in Figure 25. The pole-zero map shows that the transfer functions have complex poles, which means that an open-loop step input would result in an underdamped response with oscillations. Although the MATLAB analysis says that an oscillatory response would occur, that is not physically possible. SMAs actuate with a power step input and will continue actuating to a position until there exists a thermal equilibrium between the power input and the power loss due to heat transfer. The SMA would then stay at that position. The motion described is characteristic of an over-damped or perfectly damped system, not an underdamped one. Despite this, it was assumed that the controller would still be able to control the system since any steady-state errors would be removed by the integral control. Another aspect of the pole-zero plot is that since a pre-heat loop was not used in the experimental setup for determining the transfer function, the transfer functions are not LTI. This is to say that there is not one transfer function that relates power to angle for all power levels. It is expected that if the experiments are repeated controlling for the pre-activation temperature lag then we would see the LTI behavior of Figure 17 when the time lag is removed.

As Figure 20 shows, the fits done by MATLAB do not differ too much in accuracy despite using different pole-zero models. The pure 2nd order system model has a fit accuracy of 92.48%, the 2nd order system with one lag has an accuracy of 96.02%, and the 2nd order system with two lags has an accuracy of 96.8%. Considering that the lags are simply due to the lack of pre-heating of the SMA and that there is not a large difference in the accuracy of fit, it was decided to use the pure 2nd order system model.

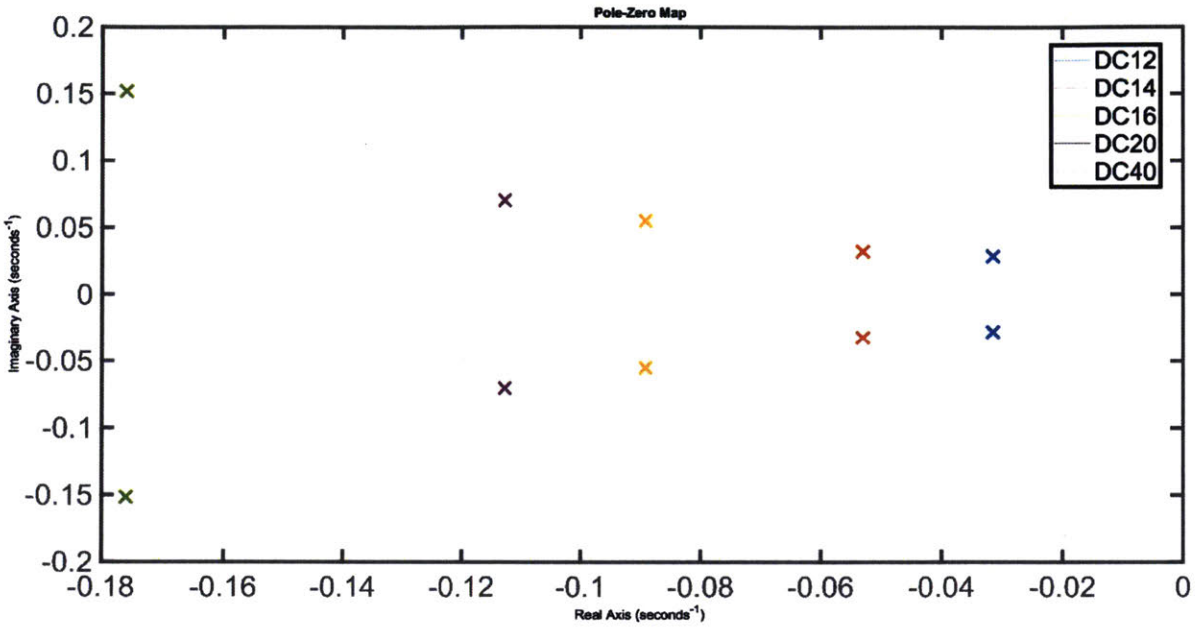

Figure 25. Pole-zero map of averaged step-response data.

In the experiments that were done to characterize the transfer function, the SMA hinge plant was not LTI. However, under the conditions of low power input and controlling for pre-activation temperature the system can be assumed to be linear time invariant.

7.2 Controlled Response to a Step Input

The gains were balanced around the averaged transfer function at the 16 duty cycle power level (which corresponds to about 0.14W) because at low power levels the system appears to be LTI if time lags are ignored. The gains for this initial tuning were Kp = 0.04379, Ki = 0.002789, and Kd = 0.1442. When those gains were input into the system, the SMA actuator was not responding as fast as expected. This was due to a conversion issue in the Arduino between power output and the duty cycle. Because of this issue, the output power was much lower than anticipated.

When the tests were run, the gains were multiplied by 10 or 50, depending on the test, in order to actuate the hinge faster. The gains multiplied by 10 were then Kp10 = 0.4379, Ki10 = 0.02789, Kd10 = 1.442. The gains multiplied by 50 were then Kp50 = 2.1895, Ki50 = 0.13945, and Kd50 = 7.21.

For the 1st trial, a setpoint of -20° was input into the controller with the 50x gains. The angle versus time plot for this trial is provided in Figure 26. The SMA wires were at room temperature at this point (23°C). The wires had to heat up to their austenite start temperature, which for these specific wires was around 80°C. Because of this, the integrator windup error

increased over the duration of the preheat. Once the wire reached the activation temperature the controller was still inputting a large amount of power which caused the SMA to rapidly actuate. It is demonstrated in Figure 26 that the SMA actuated to the setpoint in about 1.5 seconds and then overshot more than 15 degrees. Utilizing a pre-heat loop would eliminate the time lag seen in Figure 26. The overshoot is due to the error term ramping up over the length of the pre-heat. If a new system model is created without the heat-up lag, the system model should be able to accomplish the same actuation time without the overshoot. Improving this response will be discussed more in Section 8: Conclusions and in Future Work and Section 9.13: Controls Design.

In the 2nd trial, the setpoint and gains remained the same but now the wire was preheated due to the 1st trial. The controlled response is shown in Figure 27. Since the wire is preheated there is an instant response. The controller reaches the setpoint at about 15 seconds and overshoots about 2.5°, which is good for a first controlled prototype test. With a more advanced controls implementation, the rise time and error can be reduced substantially. It is important to note that the overshoot was starting to return to the setpoint of -20° after about 20 seconds. This slow error response is due to a relatively small Ki50 but more so due to needing to heat up the opposing SMA. Running a preheat loop on both SMAs will allow the antagonistic SMA to instantly respond to overshoot. It is predicted that if this test was allowed to continue running it would overshoot the -20° setpoint in the positive direction in the same way the 1st trial overshot its setpoint due to the error windup over the duration of the preheat.

In the 3rd trial, the setpoint remained the same but the gains were reduced to the 10x gains. The data and setpoint is provided in Figure 28. Due to the decreased gains there was a larger time to hit the setpoint, about 75 seconds. However, due to these decreased gains there was a decreased overshoot, about less than 2°. As mentioned before, the overshoot was not corrected during the controller test because the other SMA was still at room temperature and still needed to warm up to the activation temperature. In some space applications, rise time is not as important as power draw so actuating with lower control gains could be beneficial depending on applications.

In the 4th trial, a different setpoint of -15° was applied with the 50x gains. The data is shown in Figure 29. As was seen in the 2nd trial, the setpoint was passed around 15 seconds. There was an overshoot of about 5° during this trial. As mentioned before, the overshoot issue can be solved with a preheat loop and a better system model.

In the 5th trial, an even smaller setpoint of -10° was set with the 50x gains. The data is provided in Figure 30. Once again, the setpoint was reached in about 15 seconds, similar to the 2nd and 4th trials. The test overshot about 8°. More tests are needed to confirm whether this extra error was an outlier test. However, as mentioned before, implementing a preheat loop would allow the other SMA to correct the overshoot.

An important aspect of these trials to note is that for all these trials, the SMAs actuated past the hypothetical saturation angle of 18°. This is most likely due to the fact that step responses were not done for higher power levels so the saturation angle was not correctly determined. If the temperature of the SMA was monitored and the SMA was heated until the austenite finish temperature, then at that point it would be known that the SMA had exerted its full force and the saturation angle could be determined. This was not done for these experiments. For future trials, determining the saturation angle is important because it determines the full range of the actuator.

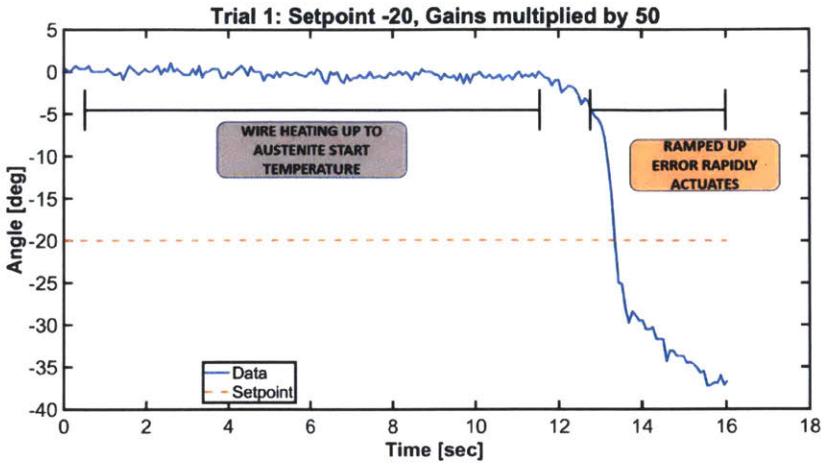

Figure 26. 1st controlled test. Rapidly actuates due to error wind-up.

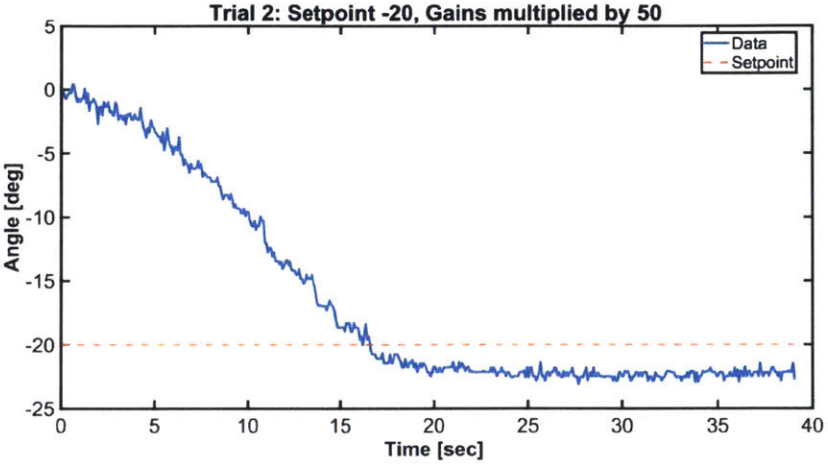

Figure 27. 2nd controlled test. Actuates over longer time span than 1st test due to not as much error windup.

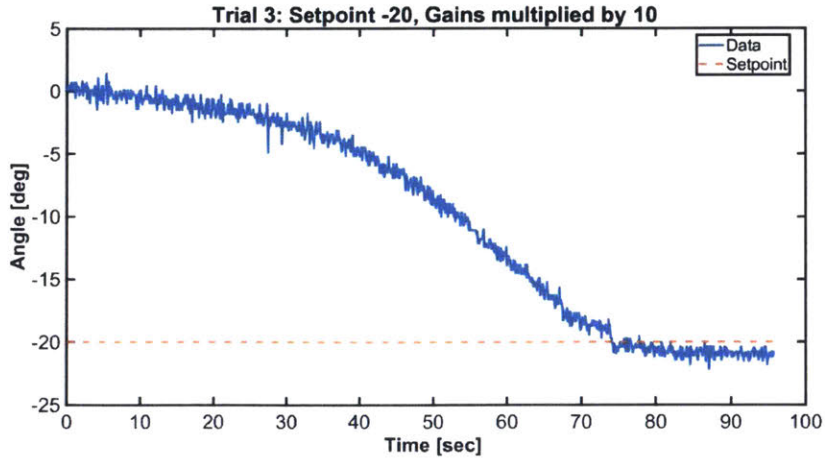

Figure 28. 3rd controlled test. Lowered gains result in longer rising time but lower initial error.

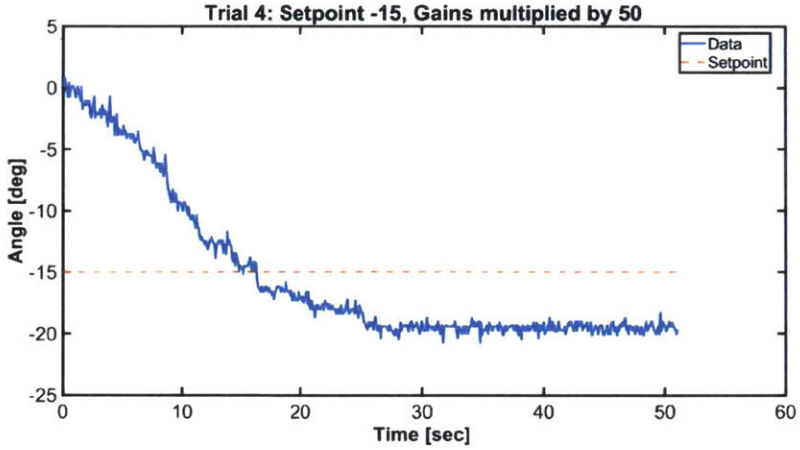

Figure 29. 4th controlled test. A different setpoint still results in the same time to setpoint and error as the 2nd test.

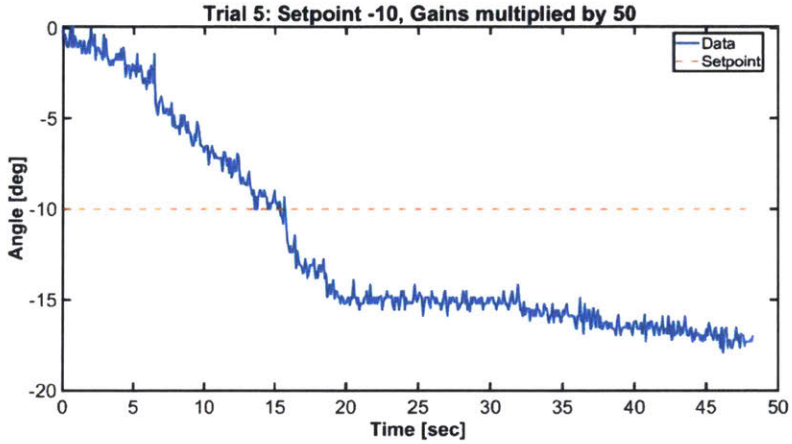

Figure 30. 5th controlled test. A different setpoint with the 50x gains results in the same time to setpoint and error as the 2nd and 4th tests.

8. Conclusions

From the tests it can be concluded that although these experiments did not correctly implement the needed assumptions to model the system as an LTI system, it is possible to treat the antagonistic SMA wires as an LTI system if certain assumptions are met. Those assumptions would be implementing a preheat loop to ensure the wires can immediately respond to control inputs and maintaining relatively low power levels. More experimentation needs to be done to determine which power levels begin to violate LTI assumptions.

In addition to this, it can be seen that despite the system model not being completely accurate due to the preheat time lag and complex poles as described in Section 7.1, the controller was still robust enough to control the system and actuate to the setpoint within the required time of 15 seconds. With a preheat loop and a better system model, more accurate PID gains can be found and a faster response time can be achieved. Another method to increase the response time is to monitor the resistance of the SMA wire. This idea will be discussed further in Future Work, Section 9.13: Controls Design.

These experiments also demonstrate that with an improved controls implementation it is possible to achieve 2 second rise times as shown in Figure 26. These experiments also show that it is possible to control an antagonistic pair of SMA wires to actuate to a setpoint angle. More experiments are necessary to determine how the antagonistic setup will deal with steady-state errors, as these experiments did not allow enough time for the antagonistic SMA to heat up and begin to actuate. It is expected though that since one wire was able to actuate to within 2 degrees of the setpoint, with the antagonistic wire and a better system model, that angle resolutions of sub-2 degrees will be possible. Moreover, increasing the Ki term of the controller should eliminate steady-state errors faster. This opens up possibilities of using this hinge architecture for precision space applications where angular resolution is of high importance.

9. Future Work

These experiments and prototypes have provided valuable knowledge for what design changes need to occur to make the hinge more reliable, precise, and optimal. This section discusses the future work needed to make these improvements and what benefits those improvements will provide.

9.1 Design Recommendations

This section focuses on the improvements in the mechanical, electronic, and controls design of the hinge. These improvements are expected to substantially improve the hinge's reliability and precision.

9.11 Mechanical Design

There are multiple mechanical design changes that can improve the functionality of the current SMA hinge prototype. The first and easiest to accomplish is to machine the hinge out of a space grade material. This would first satisfy the outgassing requirement that many space components have. To ensure the validity of a prototype hinge, it should be manufactured out of space-like materials, which exhibit low outgassing characteristics. In addition to this, machining would give a prototype hinge with better geometric precision, helping to eliminate sources of error.

Another design change that needs to occur is to better constrain the center of the passive SMA to bend with the active SMA. As mentioned before, the SMA gets loaded in compression, increasing the angular stiffness of the hinge and preventing the full SMA range of motion. This is a difficult problem but would expand the hinge to more mechanisms and uses that require a larger range of motion. A potential solution could be to decrease the length of the midsection of the SMA as only the midsection of the SMA is actually contributing to motion. Another solution is to remove a constraint and allow the pins in the hinge to slide within a slot. This would allow the SMAs to curve in whatever shape minimizes energy. Currently, the SMAs are geometrically constrained with the pin pivot, causing the passive SMA to be loaded in compression and sometimes buckle.

In addition to this, the electrical contact of the hinge needs to be improved. Currently, the SMAs are bolted onto the hinge and current is transmitted through the bolt head into the SMA. This method of electrical contact is not repeatable, so it does not stand up to space-grade testing. In addition to this, these bolts also increase weight and have significant resistance when compared to the SMA wires, decreasing efficiency. Possible solutions could include a clamping set of copper plates or inserting the wires into a cavity as done by Haughwout et al in their prototype [20].

One of the most important design changes that should occur is the design of an integrated hinge. Currently, the hinge is designed to mount onto the endcap of a CubeSat, as shown in Figure 31. This design increase complexity and currently only allows for two hinges to be mounted onto a CubeSat, whereas many CubeSats opt to use four. An endcap should be designed that is able to slot in multiple hinges on the CubeSat sides without increasing the z-length of the CubeSat as the current design does. Electrical contacts inside the endcap slots would connect these hinges to the control system and power outputs. This would conserve volume and decrease mass while allowing for full control of up to four hinges.

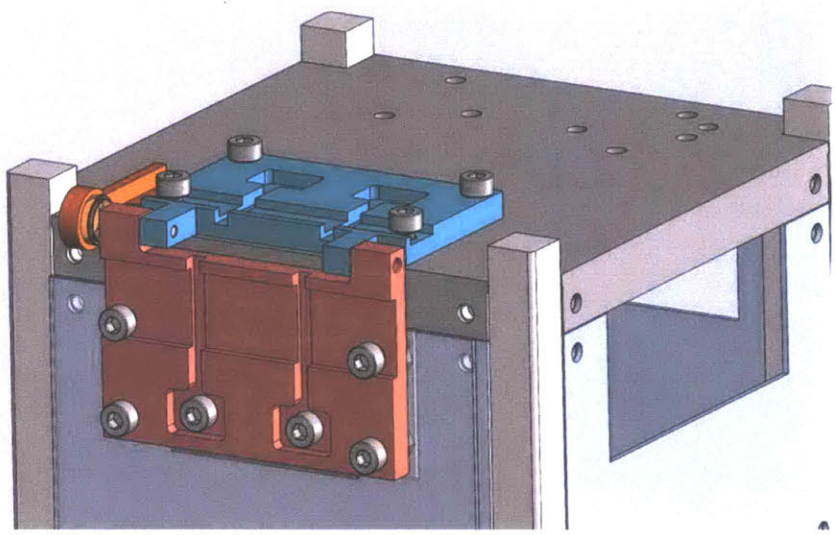

Figure 31. Mockup of how hinge would mount onto a 3U CubeSat. Note the endcap only has space for two hinges with this current design.

9.12 Electronic Design

The electronic design of the actuator could also use improvement. Currently, the test setup has wires connecting to the far end of the actuator (the red hinge seen in Figure 31). This means that during testing the hinge must actuate against the passive force of a wire. The wires should be integrated into the hinge structure instead of being free-floating.

In addition to this, the electrical design is challenging because the SMA wires have relatively low resistance, on the order of tens milliOhms. Currently, the wires that connect the SMA to the power supply are on the same order of resistance as the SMAs, meaning that these wires also experience substantial Joule heating when compared to the SMAs. This means that there is a substantial efficiency loss in the wires transmitting current that cannot be ignored. For space applications, power losses and efficiency are incredibly important. Several solutions, including embedding low resistance conductive plates or wires into the hinge, have been considered for future versions of the hinge.

The power distribution unit is an off-the-shelf DC-DC converter. High frequency noise was observed in several tests across the shunt resistor near the output of the DC-DC converter, which was amplified by the Arduino. The presence of this noise in addition to some lower frequency noise due to the PWM drive signal not being completely attenuated by the low pass filter created significant noise in power measurement. A plot of the power versus time for one of the controlled trials is provided in Figure 32. In addition to this, the power command of the

controller was being updated at a frequency of 50 Hz whereas logging of the power only occurred at about 12 Hz. This means that the power measurements could also suffer from aliasing. Because of this noise and potential aliasing issue, no useful controller input power data can be attained so no findings with regards to power were discussed in this thesis.

A DC-DC converter should be designed from the ground up in order to minimize the noise caused by the measurement circuitry. The magnetic encoder's angular reading is also afflicted by noise. This results in errors of about 1 degree, as seen during the heating up phase of Figure 26. Currently there is no low-pass filter on the encoder output but implementing one would help eliminate this error which would improve accuracy of the controller.

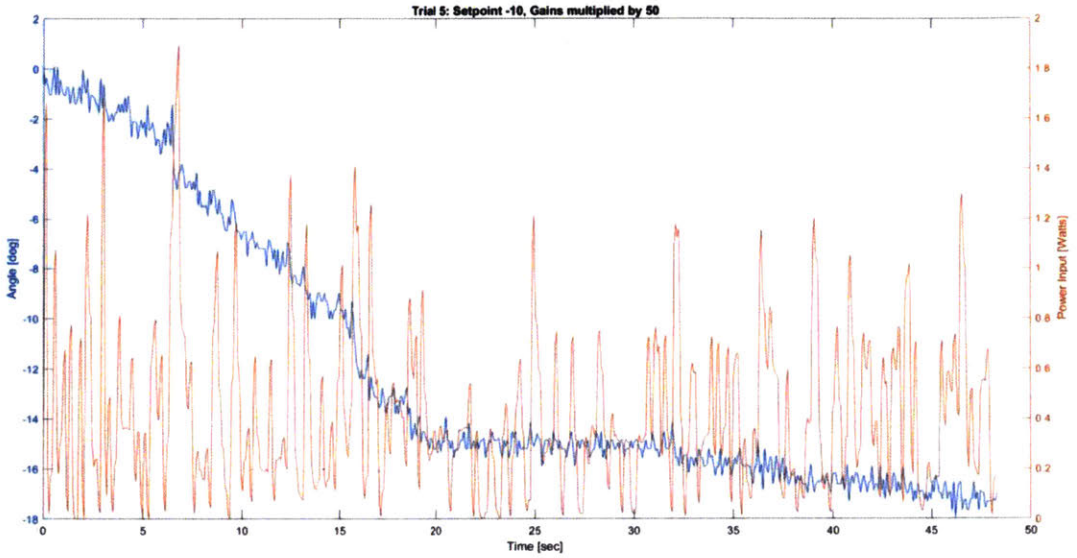

Figure 32. Plot of controller power input and angle for the 5th trial. There is substantial noise in the power input measurement which prevents analysis on what power signals the controller is sending.

9.13 Controls Design

Many controller design improvements could increase the responsiveness of the system and increase its accuracy. Teh describes a great controller in his dissertation. He uses a pre-heat loop (he refers to it as an anti-slack mechanism) that keeps the wires ready to exert force [7]. This would greatly help with the responsiveness of the system by allowing the passive SMA to actuate against the overshoot of the first wire. Tests should be run with this preheat mechanism to determine how long it takes the system to reach steady state.

Teh also improved the response of his system and achieved incredibly fast response times of less than 1 second [7]. He did this by creating a resistance monitoring loop. Resistance changes as a function of SMA temperature. Characterizing the temperature resistance curve allows us to use the SMA resistance measurement as a proxy for a temperature measurement. This fact was exploited by Teh in his dissertation to create a control loop that calculated how much power could be input into the SMA before damaging it [7]. If the reannealing temperature or a temperature that will damage the SMA is known, a resistance can be mapped to that specific temperature. A controller could then calculate the maximum amount of power that could be put into the system without overheating the SMA and it could also monitor the SMA resistance. If the controller detects that the SMA resistance is reaching a point where the temperature of the

SMA is getting dangerously high, it could be programmed to cut off all power to the SMA. In this way, a controller would be able to achieve rapid responses similar to the one in Figure 26 after the preheat. A controller with high gains could exhibit fast response times and still not damage the SMA since it would be monitoring its temperature by using a resistance characterization curve. Teh implemented this method in his dissertation for his actuator setup [7].

Improving the characterization of the system is also an avenue for improving the actuator. Doing a frequency response of the SMA would give better results than subjecting the SMA to step responses and trying to fit curves to these responses. In addition, a frequency response would produce a Bode plot, allowing us to see any resonances with certain frequencies. Teh characterized his SMAs with a frequency response and achieved fast actuation times by creating a very accurate system model [7]. It may also be of value to run a feedback loop with a simple proportional gain of Kp = 1 to observe the system dynamics with error feedback.

9.2 Characterizing SMA Torque

An important aspect of the SMA is characterizing how much torque it can be produce. Currently, the transfer function that was determined for the system only applies to the specific system setup. If any significant component in the system changes, then the transfer function is no longer valid and experimentation is needed once again to characterize the system. In order to expand the SMAs to more mechanisms, it is necessary to characterize the relationship between SMA torque and temperature or resistance. Once this is characterized, it would be possible to model different mechanisms, such as steerable antennas and other deployable structures. A dynamic model could then be constructed and different system configurations could be tested in MATLAB or another modeling software before fabrication.

In addition to this, it is important to characterize the relationship between torque, area, and geometry of different SMAs. Knowing how circular and rectangular cross-sections differ in torque and power input per area would simplify the design process. If an SMA torque is known, it would be possible to check if that torque was enough to deform the antagonistic SMA through solid mechanics calculations. That process would inform the range of motion of different SMAs. Using that range of motion would then be extremely valuable for designing different spacecraft deployable structures and mechanisms.

10. References

[1] spacexcmsadmin, 2012, "Capabilities & Services," SpaceX [Online]. Available: https://www.spacex.com/about/capabilities. [Accessed: 01-Apr-2019].

[2] "Updates," Rocket Lab [Online]. Available: https://www.rocketlabusa.com/news/updates/. [Accessed: 01-Apr-2019].

[3] "About," dyneema [Online]. Available: https://www.dsm.com/products/dyneema/en_GB/about.html. [Accessed: 08-May-2019].

[4] "Shape Memory Alloys" [Online]. Available: https://technology.nasa.gov/patent/TOP3-407. [Accessed: 01-Apr-2019].

[5] Cederström, J., and Van Humbeeck, J., 1995, "Relationship Between Shape Memory Material Properties and Applications," Le Journal de Physique IV, **05**(C2), pp. C2-335-C2-341.

[6] "Other Alloys Exhibiting Shape Memory Effects," Confluent Medical [Online]. Available: https://nitinol.com/reference/other-alloys-exhibiting-shape-memory-effects/. [Accessed: 08-May-2019].

[7] Teh, Y. H., "Fast, Accurate Force and Position Control of Shape Memory Alloy Actuators," p. 205.
[8] Yuan, H., Fauroux, J., Chapelle, F., and Balandraud, X., 2017, "A Review of Rotary Actuators Based on Shape Memory Alloys," Journal of Intelligent Material Systems and Structures, **28**(14), pp. 1863–1885.
[9] Khatsenko, M. O., 2017, "A Rotary Shape Memory Alloy Actuator for CubeSat Deployable Structures," Thesis, Massachusetts Institute of Technology.
[10] Huang, W., "Shape Memory Alloys and Their Application to Actuators for Deployable Structures," p. 192.
[11] "Reinventing the Wheel" [Online]. Available: https://www.nasa.gov/specials/wheels/. [Accessed: 01-Apr-2019].
[12] Petrini, L., and Migliavacca, F., 2011, "Biomedical Applications of Shape Memory Alloys," Journal of Metallurgy [Online]. Available: https://www.hindawi.com/journals/jm/2011/501483/. [Accessed: 02-Apr-2019].
[13] Mihálcz, I., 2001, "FUNDAMENTAL CHARACTERISTICS AND DESIGN METHOD FOR NICKEL-TITANIUM SHAPE MEMORY ALLOY," 1, **45**(1), pp. 75–86.
[14] "Illustration of the One-Way Shape Memory Effect (a) and the Hysteresis...," ResearchGate [Online]. Available: https://www.researchgate.net/figure/Illustration-of-the-one-way-shape-memory-effect-a-and-the-hysteresis-of-martensitic_fig3_279224513. [Accessed: 02-Apr-2019].
[15] Tanaka, K., 1990, "A Phenomenological Description on Thermomechanical Behavior of Shape Memory Alloys," Journal of Pressure Vessel Technology, **112**(2), p. 158.
[16] Bergamasco, M., Salsedo, F., and Dario, P., 1989, "A Linear SMA Motor as Direct-Drive Robotic Actuator," *1989 International Conference on Robotics and Automation Proceedings*, pp. 618–623 vol.1.
[17] Songa, G., and Agrawa, B. N., "Active Position Control of a Shape Memory Alloy Wire Actuated Composite Beam," p. 12.
[18] Hwang, D., and Higuchi, T., 2014, "A Cycloidal Wobble Motor Driven by Shape Memory Alloy Wires," Smart Mater. Struct., **23**(5), p. 055023.
[19] "Nitinol Innovation, Development and Supply - Kellogg's Research Labs" [Online]. Available: https://www.kelloggsresearchlabs.com/. [Accessed: 30-Apr-2019].
[20] Haughwout, C., Barnes, D., Khatsenko, M., Clark, J., and Cahoy, K., "Folded Lightweight Aperture Precision System."
[21] "Super Small Non-Contact Rotary Encoder RM08 - Www.Rls.Si" [Online]. Available: https://www.rls.si/eng/rm08-super-small-non-contact-rotary-encoder. [Accessed: 01-May-2019].
[22] "Onyx 3D Printer Filament and Printing Material | Markforged" [Online]. Available: https://markforged.com/materials/onyx/. [Accessed: 01-May-2019].
[23] "Nitinol Innovation, Development and Supply - Kellogg's Research Labs" [Online]. Available: https://www.kelloggsresearchlabs.com/. [Accessed: 30-Apr-2019].

11. Appendix

```
extendData.m. Used to extend data for analysis
function [pwr, angle, time] = extendData(power, angle, time, extra_time)
    pwr = power; %Stores variables
    angle = angle;
```

```
    time = time;
    extra_time = extra_time;
    s = 0.0035; %Sets spacing between time measurements at this value to
mimic data collec. rate
    x1 = time(end);
    x2 = x1 + extra_time; %Sets new endpoint
    n = ((x2 - x1)/s) + 1; %Calculates number of points between x1 and x2
    n = floor(n); %Brings n down to an integer for linspace and ones
functions
    pwr = [pwr; ones(n,1)*pwr(end)];   %Extends angle and power to improve
fitting
    angle = [angle; ones(n,1)*angle(end)];
    time = [time; linspace(x1,x2,n)']; %Extends time to x2 seconds to improve
fitting
end

cleanDataNoExt.m Used to 'clean' data without extending it. It simply
calibrates out the initial value reading and puts time in the seconds scale.
function [pwr, ang, t] = cleanDataNoExt(power, angle, time)
    pwr = power; %Stores variables
    ang = angle;
    t = time;
    pwr = pwr - ones(size(pwr))*pwr(1); % calibrates values by
subtracting first number
    ang = ang - ones(size(ang))*ang(1);
    t = t - ones(size(t))*t(1);
    t = t/1000; %converts to seconds
end

cleanData.m. Does both extendData.m and cleanDataNoExt.m.
function [pwr, angle, time] = cleanData(power, angle, time, extra_time)
    pwr = power; %Stores variables
    angle = angle;
    time = time;
    extra_time = extra_time;
    pwr = pwr - ones(size(pwr))*pwr(1); % calibrates values by subtracting
first number
    angle = angle - ones(size(angle))*angle(1);
    time = time - ones(size(time))*time(1);
    time = time/1000; %converts to seconds
    s = 0.0035; %Sets spacing between time measurements at this value to
mimic data collec. rate
    x1 = time(end);
    x2 = x1 + extra_time; %Sets new endpoint
    n = ((x2 - x1)/s) + 1; %Calculates number of points between x1 and x2
    n = floor(n); %Brings n down to an integer for linspace and ones
functions
    pwr = [pwr; ones(n,1)*pwr(end)];   %Extends angle and power to improve
fitting
    angle = [angle; ones(n,1)*angle(end)];
    time = [time; linspace(x1,x2,n)']; %Extends time to x2 seconds to improve
fitting
end
```

```
TestDataScript2.m. Loads datasets, creates step response plots and fits
transfer functions, creates pole zero maps, and calculated averaged step
responses.
%% Load Data
load('FlatWire3rdTesting.mat'); %Loads clean and unclean data
%% Step Response Plots & Pole-Zero Plots of System
disp('This will take a while. Please wait.');
names = {'T1','T2','T3', 'T4', 'T5'};
DataSet = PreppedDataT3{2,3}; %DataSet input goes here
for i = 1:5
    cleandata = DataSet{2,i};
    power = cleandata(:,1);
    angle = cleandata(:,2);
    time = cleandata(:,3);

    data = iddata(angle, power, 0.0035); %creates data object
    sys = tfest(data,4,2); %estimates tf
    opt = stepDataOptions; %creates options object
    opt.StepAmplitude = power(end); %Sets amplitude of step to power steady
state
    tf.(names{i}) = sys; %Put transfer func. into structures tf

    figure(i);
    plot(time,angle,'r');
    hold on;
    step(sys,opt);
    ylabel('Angle [deg]');
    xlabel('Time [sec]');
    legend('Step Response Data', 'Step Response based on Transfer Func.');
    improvePlot();
end
figure(1);
title('Trial 1');
figure(2);
title('Trial 2');
figure(3);
title('Trial 3');
figure(4);
title('Trial 4');
figure(5);
title('Trial 5');

figure(6);
pzmap(tf.T1,tf.T2, tf.T3, tf.T4, tf.T5);
legend('T1', 'T2', 'T3', 'T4', 'T5');
improvePlot();
%% Pole Zero Map of all Tests
for i = i:5
    tfs = Tf_cell(2,i);
    pzmap(tfs.T1,tfs.T2,tfs.T3,tfs.T4,tfs.T5);
    hold on;
end
legend('20DC', '40DC');
improvePlot();
%% Averaged Step Responses Plots and PZ Map
disp('This will take a while. Please wait.');
names = {'DC12','DC14','DC16','DC20','DC40'};
```

```
for i = 1:5
    cleandata = averaged_data{2,i};
    power = cleandata(:,1);
    angle = cleandata(:,2);
    time = cleandata(:,3);
    [power,angle,time] = extendData(power,angle,time,80.0);

    data = iddata(angle, power, 0.0035); %creates data object
    sys = tfest(data,2,0); %estimates tf
    opt = stepDataOptions; %creates options object
    opt.StepAmplitude = power(end); %Sets amplitude of step to power steady
state
    tf.(names{i}) = sys; %Put transfer func. into structures tf

    figure(i);
    plot(time,angle,'r');
    hold on;
    step(sys,opt);
    ylabel('Angle [deg]');
    xlabel('Time [sec]');
    legend('Step Response Data', 'Scaled Step Response based on Transfer
Func.');
    improvePlot();
end
figure(1);
title('12DC Step');
figure(2);
title('14DC Step');
figure(3);
title('16DC Step');
figure(4);
title('20DC Step');
figure(5);
title('40DC Step');

figure(6);
pzmap(tf.DC12,tf.DC14,tf.DC16,tf.DC20,tf.DC40);
legend('DC12', 'DC14', 'DC16', 'DC20', 'DC40');
improvePlot();
```

DataOrganizer.m. Got raw data into useable forms by applying different functions. Some of this was written by Charles Lindsay.

```
    %% Import Data
cd 'D:\MIT Undergrad D\Senior Year\Thesis\Controls\Test Data 3rd Round';
files = dir('*.txt');
%% Create Data Cells
PreppedData12DC = cell(2,5);
n = 200; %samples in moving average
window = ones(1,n)./n;
for i = 1:5
    cd 'D:\MIT Undergrad D\Senior Year\Thesis\Controls\Test Data 3rd Round';
    name = files(i).name;
    fileID = fopen(name);
    info = textscan(fileID,'%s %n %s %n %s %n');
    fclose(fileID);
    [power, angle, time] = deal(info{2},info{4}, info{6});
```

```
    if angle(length(angle)) < 0
        angle = -1*angle;
    end

    cd 'D:\MIT Undergrad D\Senior Year\Thesis\Controls\'
%   [power,angle,time] = cleanDataNoExt(power,angle,time); %Cleans data
    [power,angle,time] = cleanData(power,angle,time,20.0); %Cleans data
    angle = filter(window,1,angle);
    power = filter(window,1,power);

    PreppedData12DC{2,i} = [power, angle, time];
end
PreppedData12DC{1,1} = 'Test 1';
PreppedData12DC{1,2} = 'Test 2';
PreppedData12DC{1,3} = 'Test 3';
PreppedData12DC{1,4} = 'Test 4';
PreppedData12DC{1,5} = 'Test 5';
%% Plotting Test
index = 5;
setdirty = DirtyDataTesting3{2,index};
setclean = PreppedDataT3{2,index};
    for j = 4
        trialdirty = setdirty{2,j};
        trialclean = setclean{2,j};

        figure();
        plot(trialdirty(:,3),trialdirty(:,1));
        hold on;
        plot(trialclean(:,3),trialclean(:,1));
        legend('Raw Data', 'Filtered Data');
        xlabel('Time [s]');
        ylabel('Power [Watts]');
        improvePlot();

        figure();
        plot(trialdirty(:,3),trialdirty(:,2));
        hold on;
        plot(trialclean(:,3),trialclean(:,2));
        legend('Raw Data', 'Filtered Data');
        xlabel('Time [s]');
        ylabel('Angle [deg]');
        improvePlot();
    end
%% DirtyData to PreppedData
n = 200; %samples in moving average
window = ones(1,n)./n;
PreppedDataT3 = DirtyDataTesting3;
for i = 1:5
    set = DirtyDataTesting3{2,i};
    for j = 1:5
        pat = set{2,j};
        power = pat(:,1);
        angle = pat(:,2);
        time = pat(:,3);

        angle = filter(window,1,angle);
        power = filter(window,1,power);
```

```
        [power,angle,time] = extendData(power,angle,time,40.0); %Extends data

        set{2,j} = [power,angle,time];
    end
    PreppedDataT3{2,i} = set;
end
%% Load Average Data
load('Avg_Ang_DC_40.mat');
load('Avg_Pwr_DC_40.mat');
load('stand_t_DC_40.mat');

data40 = [average_power,average_angle,stand_t'];
```

ControlsDataTestPlotter.m. Used to plot data from the controlled tests.

```
%% Controlled Response Data Creation
for i = 1:5
    data = ControlsTestDataRaw{2,i};
    [power,angle,time] = cleanDataNoExt(data(:,1),data(:,2),data(:,3));
    vec = [power,angle,time];
    ControlsTestData{2,i} = vec;
end
%%  Plotting Test Data on Same Plot
for i = 1:5
    trial = ControlsTestData{2,i};
    figure(i);
    yyaxis left;
    plot(trial(:,3),trial(:,2));
    xlabel('Time [sec]');
    ylabel('Angle [deg]');
    yyaxis right;
    plot(trial(:,3),trial(:,1));
    ylabel('Power Input [Watts]');
end
figure(1);
title('Trial 1: Setpoint -20, Gains multiplied by 50');
figure(2);
title('Trial 2: Setpoint -20, Gains multiplied by 50');
figure(3);
title('Trial 3: Setpoint -20, Gains multiplied by 10');
figure(4);
title('Trial 4: Setpoint -15, Gains multiplied by 50');
figure(5);
title('Trial 5: Setpoint -10, Gains multiplied by 50');
%% Plotting Angles
setpoints = [-20,-20,-20,-15,-10];
for i = 1:5
    trial = ControlsTestData{2,i};
    time = trial(:,3);
    figure(i);
    %plot(trial(:,3),trial(:,2));
    hold on;
    y = ones(size(time))*setpoints(i);
    plot(time, y,'--');
    xlabel('Time [sec]');
    ylabel('Angle [deg]');
    legend('Data','Setpoint');
    improvePlot();
```

```
end
figure(1);
title('Trial 1: Setpoint -20, Gains multiplied by 50');
figure(2);
title('Trial 2: Setpoint -20, Gains multiplied by 50');
figure(3);
title('Trial 3: Setpoint -20, Gains multiplied by 10');
figure(4);
title('Trial 4: Setpoint -15, Gains multiplied by 50');
figure(5);
title('Trial 5: Setpoint -10, Gains multiplied by 50');
```

hinge_control.ino // code used to run controlled tests

```
#include <Adafruit_ADXL345_U.h>
#include <Adafruit_Sensor.h>
#include <PID_v1.h>

// PID Settings
const float Kp = 0.4379 * 5;
const float Kd = 1.442 * 5;
const float Ki = 0.02789 * 5;

const float f  = 20;
const float dt = 1/f;
float enc_cal = 0;
float t0 = 0;

double set, in, out;

// Pin Definitions
const uint8_t ADC_CS = 7; //Selection Pin
const uint8_t MOSI_LC = 11;//MOSI (Local)
const uint8_t MISO_LC = 12;//MISO
const uint8_t SPICLOCK = 13;//Clock
const uint8_t SMA_H1_R = 9;
const uint8_t SMA_H1_L = 10;
const uint8_t SMA_H2_R = 6;
const uint8_t SMA_H2_L = 5;

// LED pin definitions
const uint8_t P  = 2;
const uint8_t B1_1 = 3;
const uint8_t B2 = 4;

// Miscellaneous setpoints
const float VREF = 4.98; //ADC reference voltage (tied to +5V);
measured with voltmeter across decoupling cap by VREF on 1.24.19
```

```
const int G = 5+5*(51/1); //instrumentation amp gain =
5+5*(R2/R1) where R1 = 1k, R2 = 51kc
const float r_shunt = 0.001; //current shunt resistance (ohms)
const float micro_g_thres = 0.2; //minimum G-Force to be
considered microgravity

const float R_sma = 0.045;
const float R_ext = 0.090;
const float V_drive = 1.8;
const float V_eff = (R_sma/(R_ext + R_sma)) * V_drive;
const float p_max = pow(V_eff, 2)/(R_sma);

// Calibrations
int shunt_cal_1 = 0;
int shunt_cal_2 = 0;

// PID object
PID pid(&in, &out, &set, Kp, Ki, Kd, DIRECT);
Adafruit_ADXL345_Unified accel =
Adafruit_ADXL345_Unified(12345);  //declare accelerometer object

void setup_nvic()
{
  cli(); // stop interrupts
  TCCR1A = 0; // set entire TCCR1A register to 0
  TCCR1B = 0; // same for TCCR1B
  TCNT1  = 0; // initialize counter value to 0
  // set compare match register for 50 Hz increments
  OCR1A = 39999; // = 16000000 / (8 * 50) - 1 (must be <65536)
  // turn on CTC mode
  TCCR1B |= (1 << WGM12);
  // Set CS12, CS11 and CS10 bits for 8 prescaler
  TCCR1B |= (0 << CS12) | (1 << CS11) | (0 << CS10);
  // enable timer compare interrupt
  TIMSK1 |= (1 << OCIE1A);
}

// Charles

void drive_sma(String SMA, int DutyCycle){
  //drive SMA = ('1R' or '1L' or '2R' or '2L') | duty cycle =
(integer 1 to 100)
  //returns immediately if SMA input does not match valid
selection

   if (DutyCycle > 100){ //assume error if input DC > 100, set
to 0
```

```
    DutyCycle = 0;
   }

   int smad;

   if (SMA == "1R") {
     smad = SMA_H1_R;
   } else if (SMA == "1L") {
     smad = SMA_H1_L;
   } else if (SMA == "2R"){
     smad = SMA_H2_R;
   } else if (SMA == "2L"){
     smad = SMA_H2_L;
   } else {
    return;
   }

   analogWrite(smad, (DutyCycle/100.0)*255);
}

bool freefall(){
  //calculate magnitude of net acceleration

  float x = get_accel('x');
  float y = get_accel('y');
  float z = get_accel('z');

  float net_accel = sqrt(x*x+y*y+z*z); //net acceleration, m/s/s

  if (net_accel <= micro_g_thres*9.81){  //returns true if net
acceleration <= micro g threshold
    return true;
  } else {
    return false;
  }

}

float get_p_sma(int DC, int hinge){
  //returns instantaneous power disapated in SMA in Watts |
input DC = 0 - 100 hinge = (1 or 2)
  //returns -1 if incorrect hinge #

  int shunt_cal;

  if (hinge == 1){
    shunt_cal = shunt_cal_1;
```

```
  } else if (hinge == 2){
    shunt_cal = shunt_cal_2;
  } else {
    return -1;
  }

  int V_SHUNT_cal = read_adc(hinge) - shunt_cal;
  int VDRV_SEN = read_adc(0);
  float Vin = (VDRV_SEN/4096.0)*VREF;
  float V_SHUNT = (V_SHUNT_cal/4096.0)*VREF;
  Serial.println(V_SHUNT,4);
  float Psma = ((DC/100.0)*Vin-
(V_SHUNT/G))*(V_SHUNT/(r_shunt*G));
  return Psma; //power in Watts

}

int cal_amp(int hinge){
  //input hinge = (1 or 2)
  Serial.println("calibrating amplifier");

  //turn off MOSFETs
  digitalWrite(SMA_H1_R,LOW);
  digitalWrite(SMA_H1_L,LOW);
  digitalWrite(SMA_H2_R,LOW);
  digitalWrite(SMA_H2_L,LOW);
  //wait for transients
  delay(500);

  int shunt_cal = read_adc(hinge); //read raw output from inst.
amplifier with MOSFETs turned OFF -> stores amp output at 0V to
correct for amp input offset

  return shunt_cal;

}

float get_accel(char axis){
  //input char = (x,y,z); returns acceleration in m/s/s OR
returns 0 if invalid axis input

   sensors_event_t accelEvent;
   accel.getEvent(&accelEvent);

  if (axis == 'x'){
    return accelEvent.acceleration.x;
```

```
  } else if (axis == 'y'){
    return accelEvent.acceleration.y;

  } else if (axis == 'z'){
    return accelEvent.acceleration.z;

  } else {
    return 0;
  }

}

int read_adc(int channel){
  //channel = int 0 - 7; returns raw value from adc 0 to 4096
(12 bits)

  int adcvalue = 0;
  byte commandbits = 0b11000000; //command bits - start, mode,
chn (3), dont care (3)

  //allow channel selection
  commandbits|=((channel)<<3);

  digitalWrite(ADC_CS,LOW); //Select adc
  // setup bits to be written
  for (int i=7; i>=3; i--){
    digitalWrite(MOSI_LC,commandbits&1<<i);
    //cycle clock
    digitalWrite(SPICLOCK,HIGH);
    digitalWrite(SPICLOCK,LOW);
  }

  digitalWrite(SPICLOCK,HIGH);    //ignores 2 null bits
  digitalWrite(SPICLOCK,LOW);
  digitalWrite(SPICLOCK,HIGH);
  digitalWrite(SPICLOCK,LOW);

  //read bits from adc
  for (int i=11; i>=0; i--){
    adcvalue+=digitalRead(MISO)<<i;
    //cycle clock
    digitalWrite(SPICLOCK,HIGH);
    digitalWrite(SPICLOCK,LOW);
  }

  digitalWrite(ADC_CS, HIGH); //turn off device
  return adcvalue;
```

```
}

float get_hinge_angle(int hinge){
  //input hinge = (1 or 2) | returns current hinge angle in
degrees from 0 to 360 deg
  //returns -360 if hinge input not in selection

  float enc_deg;

  if (hinge == 1){
    int enc_raw = read_adc(7);
    enc_deg = (enc_raw/4096.0)*360 - enc_cal;

  } else if (hinge == 2) {

  } else {
    return -360;

  }

  return enc_deg;
}

float measureAngle() {
  return get_hinge_angle(1);
}

void driveSMAs()
{
  int p_out = abs(out);
//  int dc = map((int)(255 * sqrt(p_out/p_max)), 0, 255, 0,
100);
  int dc = map(p_out, 0, 255, 0, 100);

  Serial.print(" Duty Cycle:");
  Serial.print(dc);
  Serial.print(" Power:");
  Serial.println(get_p_sma(dc, 1));

  if(out > 0)
  {
    drive_sma("1L", 0);
    drive_sma("1R", dc);
  }
  else
  {
```

```
    drive_sma("1R", 0);
    drive_sma("1L", dc);
  }
}


void setup_sma()
{
  //Disable SMA Driver
  pinMode(SMA_H1_R, OUTPUT);
  pinMode(SMA_H1_L, OUTPUT);
  pinMode(SMA_H2_R, OUTPUT);
  pinMode(SMA_H2_L, OUTPUT);
  digitalWrite(SMA_H1_R,LOW);
  digitalWrite(SMA_H1_L,LOW);
  digitalWrite(SMA_H2_R,LOW);
  digitalWrite(SMA_H2_L,LOW);

  //LED setup
  pinMode(P,OUTPUT);
  pinMode(B1_1,OUTPUT);
  pinMode(B2,OUTPUT);

  //ADC setup
  pinMode(ADC_CS, OUTPUT);
  pinMode(MOSI_LC, OUTPUT);
  pinMode(MISO_LC, INPUT);
  pinMode(SPICLOCK, OUTPUT);

  //Disable ADC initially
  digitalWrite(ADC_CS,HIGH);
  digitalWrite(MOSI_LC,LOW);
  digitalWrite(SPICLOCK,LOW);

  //Accelerometer calibration & setup
  Serial.println("Calibrating accelerometer");

//  if(!accel.begin()){
//    Serial.println("No accelerometer detected...continuing
program execution");
//  }

//  range_t maxG = ADXL345_RANGE_4_G;
//  accel.setRange(maxG); //set measurement range to sense up to
4G
//  dataRate_t rate = ADXL345_DATARATE_50_HZ;
```

```
//  accel.setDataRate(rate); //set data rate. 6.25 Hz < rate <
100 Hz (greater than 100 Hz = more noise

  //Amplifier calibration
  shunt_cal_1 = cal_amp(1);
  shunt_cal_2 = cal_amp(2);

  Serial.println("Executing main program...");
  Serial.println("");
}


int state = 0;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

//  cli();

//  setup_nvic();

  pid.SetMode(AUTOMATIC);
  pid.SetSampleTime(dt);
  pid.SetOutputLimits(-255, 255);

  setup_sma();

  set = -20; // degrees
  enc_cal = get_hinge_angle(1);

  Serial.println("Completed setup");
}

int lastMillis = 0;

void loop() {
  // put your main code here, to run repeatedly:
  if(Serial.available())
  {
    if(Serial.read() == 's')
    {
//      sei(); // Enable interrupts
      state = 1;
      t0 = millis();
    }
    else if (Serial.read() == 'e')
```

```
    {
//      cli(); // Disable interrupts

      // Clear output
      state = 0;
      out = 0;
    }

  }

  if(state == 1 && millis() - lastMillis > 50) {
    control();
    lastMillis = millis();
  }
}

// 50 hz interrupt (I think)
//ISR(TIMER1_COMPA_vect){
void control() {
//   Serial.println("Is this firing");
   // Compute pwm
   in = get_hinge_angle(1);
   Serial.print("Angle: ");
   Serial.print(in);
   Serial.print(" Commanded: ");
   Serial.print(out);
   Serial.print(" Time: ");
   Serial.print(millis() - t0);

   pid.Compute();
   driveSMAs();
}
```