March 2018

Design and Testing of a Linear Compliant Mechanism with Adjustable Force Output

William Niemeier
University of South Florida, niemeier.will@gmail.com

Design and Testing of a Linear Compliant Mechanism with Adjustable Force Output

by

William Niemeier

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Mechanical Engineering
Department of Mechanical Engineering
College of Engineering
University of South Florida

Major Professor: Craig Lusk, Ph.D.
Kyle Reed, Ph.D.
Daniel Hess, Ph.D.

Date of Approval:
March 20, 2018

Keywords: hysteresis, variable stiffness,
large deflection, creep, oscillation

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This thesis presents a novel compliant mechanism with adjustable force output. The force comes from the bending of a rectangular cross section beam within the mechanism. By rotating this beam with a stepper motor, the force output of the mechanism changes. A model was made to simulate this mechanism, and a prototype was made based off of this data. A test apparatus was constructed around this mechanism, and a series of tests were performed. These tests adjusted parameters such as beam rotation speed and weight in order to characterize the system. Adjustments were made based on this information and the mechanism was refined. The results suggest the following. The speed has a negligible effect on the behavior of the system, while the weight, length of top link r_3, and position of bottom stop have a significant effect. Also, there is a large, consistent amount of hysteresis in the system. This is likely caused by the beam storing torsion or friction from the slider.

CHAPTER 1: INTRODUCTION

1.1 Objective

The objective of this thesis is to describe the design a compliant low-stiffness mechanism with dynamically adjustable force levels that could be used as a suspension (vibration isolation) system, the design of a test apparatus, and the results of preliminary (static) tests of the mechanism's effectiveness. This novel mechanism is based on Howell's single-degree-of-freedom constant-force mechanism (Howell 2001) but has been modified (Hasara 2018) to have two degrees of freedom. The first degree of freedom (the same as Howell's) allows for a large linear range of motion and passively exerts a low-stiffness, near constant force. The second degree of freedom (Hasara's modification) allows this force level to be changed.

1.2 Motivation

There are several applications for a mechanism of this nature, including in vehicle suspensions (Spelta, et. al. 2010) and minimizing damage from earthquakes (Araki, Asai, & Masui 2009). This suspension reacts dynamically and by storing and releasing the vibrational energy it receives, may prevent the transmission of potentially damaging oscillations and shocks. This could be significant in earthquake-prone regions, on watercraft, and noisy environments like helicopters.

1.3 Scope

The scope of this thesis is to present the design and prototype of a compliant mechanism that is low stiffness, has adjustable force levels, and has properties that are well suited for use in vibration isolation, the design and prototyping of a test apparatus, and the preliminary (static) test results. The suitability of the mechanism as a vibration isolator is addressed through the design process and by adjusting various parameters during testing.

1.4 Overview

The outline of this thesis is as follows. Chapter 2 is the background. It covers the relevant theory from compliant mechanisms, including Pseudo-Rigid-Body Models (PRBMs), Howell's original constant-force mechanism, Hasara's modifications, and the numerical model of this mechanism. Chapter 3 discusses the theory that went into designing the mechanism, and describes the mechanism that was built. Chapter 4 describes the entire test apparatus of which the mechanism is a part. Chapter 5 addresses the testing process and procedures. Chapter 6 presents the results of the testing. Chapter 7 provides discussion on the results and gives concluding remarks. Chapter 8 lists possible next steps that should follow the work presented here. Chapter 9 summarizes the contributions made in this thesis. Finally, there are 3 Appendices. Appendix A contains technical drawings of the different mechanical components, Appendix B has additional tables made from results data, and Appendix C lists the code for the Arduino programs that were used.

CHAPTER 2: BACKGROUND

The background information for this thesis is in the area of compliant mechanisms; including Pseudo-Rigid-Body-Models (PRBMs), Howell's constant-force mechanism, and Hasara's modifications.

2.1 Compliant Mechanism Theory

Like rigid-link mechanisms, a compliant mechanism transfers or transforms motion, force, or energy. However, compliant mechanisms gain at least some of their mobility from the deflection of flexible members rather than from movable joints only (Howell 2001). Compliant mechanisms have many design benefits, which can be grouped into two main categories: increased performance (increased precision, increased reliability, reduced wear, reduced weight, reduced maintenance) and reduced cost (part-count reduction, reduced assembly time, simplified manufacturing processes) (Howell 2001).

Even though they offer many advantages, compliant mechanisms also have their share of challenges. One of the biggest challenges is the difficulty of analyzing and designing the mechanism (Howell 2001). By nature, the motion of compliant mechanisms involves large deflections of flexible beams. This means that the linearized beam equations that assume small deflections cannot be used. Other methods of analysis must be resorted to. These include: trial and error, large deflection analysis, and the Pseudo-Rigid-Body Model.

The Pseudo-Rigid-Body Model (PRBM) technique is used to predict the deflection path and force-deflection relationships of a flexible beam (Howell 2001). Compared to large-deflection analysis, it is both simpler and more efficient. It works by modeling a flexible beam as two rigid links pinned together. At the pin joint, there is a torsional spring which represents the beam's resistance to deflection. An illustration of a flexible beam and its corresponding PRBM can be seen in Figure 2.1 and Figure 2.2 respectively.

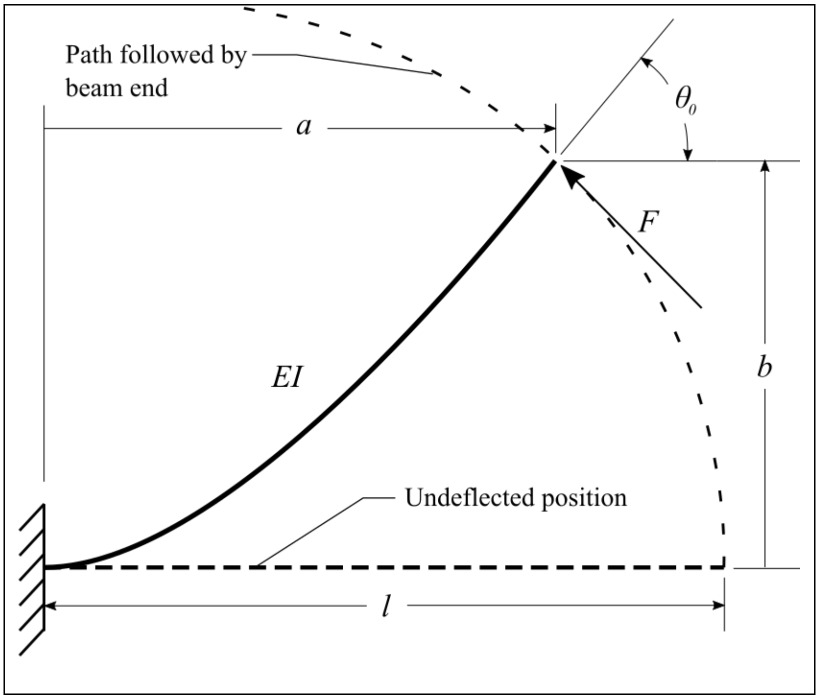


Figure 2.1 Model of a deflected beam, adapted from (Howell 2001).

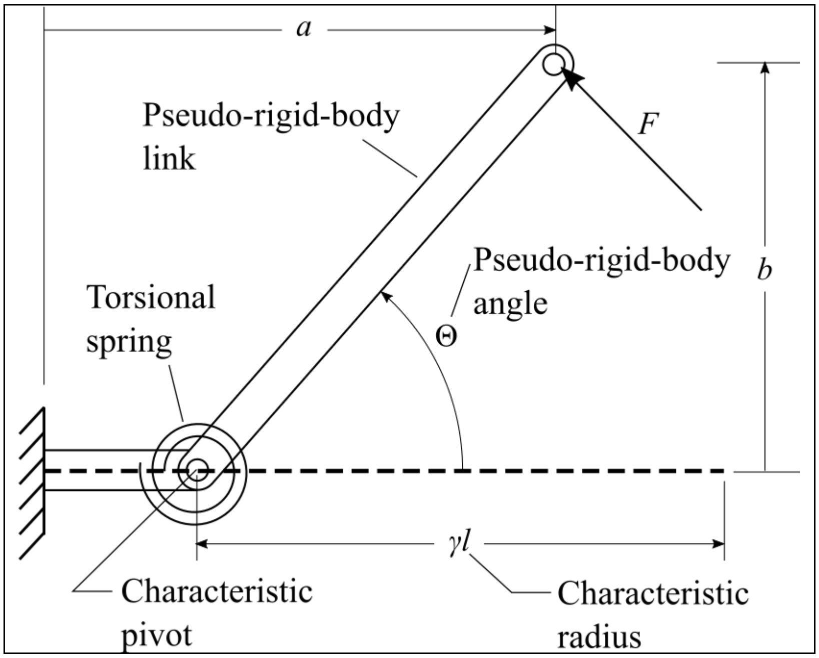


Figure 2.2 Pseudo-rigid body model of the beam in Figure 2.1, adapted from (Howell 2001).

One of the main drawbacks of the PRBM is that since it is an approximation, there is a known error. Also, it is only applicable to a given set of conditions, such as the fixed-pinned beam shown in the figures above. However, these conditions are general enough for the purposes of this thesis. For this reason, the PRBM method is used in this thesis.

2.2 Constant Force Mechanisms

An important property of compliant mechanisms is their ability to store energy. This can be harnessed to create mechanisms with special purposes, like the constant-force mechanism. Constant-force mechanisms produce a reaction force at the output port that does not change for a large range of input motion (Howell 2001). Rigid-link versions of these mechanisms also exist, but their compliant counterparts have the advantage of requiring fewer parts.

The idea behind this mechanism is that as the vertical displacement Δy gets larger, the mechanical advantage increases and balances the increasing moment from the torsional spring. In order for this to work properly, the ratio

$$R = \frac{r_3}{r_2}$$

must be an acceptable value. For the constant force mechanism shown in Figure 2.3, R = 0.8853.

2.3 Previous Design

Much of the design for the mechanism presented in this thesis comes from a load-adjustable compliant constant-force mechanism designed by Steven Hasara (Hasara 2018). This mechanism is a crank-slider, and features a compliant beam that can be rotated. The beam is framed by two sets of rigid links, one set on either side. The lengths of these links match those of the PRBM of the mechanism. This allows the path of the rigid links to match the natural path of the deflecting beam. The particularly clever part of this mechanism is that the compliant beam can rotate. The beam has a rectangular (1:2 aspect ratio) cross-section, and thus has a different stiffness depending on how the beam is rotated when it is deflected. When implemented in the mechanism, the constant-force value of the mechanism changes with the beam rotation. This design offers a theoretical 4x increase of force output from 90° beam rotation (Hasara 2018). The beam was rotated through the use of a NEMA 17 stepper motor with a 99.05:1 planetary gearbox. Its constant-force values were tested at various rotations using a tensile test machine.

The biggest differences between Hasara's mechanism and the one described in this thesis are as follows. First, the new mechanism is designed so that it can be attached to the crosshead of a tensile test machine and vertically oscillated. Second, a distance sensor and microcontroller have been added. The controller gets data from the sensor and outputs to the stepper motor that rotates the compliant beam. Thirdly, the rigid-link lengths used in testing are different than Hasara's mechanism, and they are designed to be adjustable. The motor on this mechanism is larger, a NEMA 34, and has no gearbox attached to it. Also, the new mechanism is much more sturdy and robust, made of mostly aluminum instead of wood.

2.4 Numerical Model

The mechanism discussed in this thesis is a modification of a compliant constant force mechanism. The skeleton diagram of the mechanism and its equivalent PRBM are shown below in Figure 2.3.

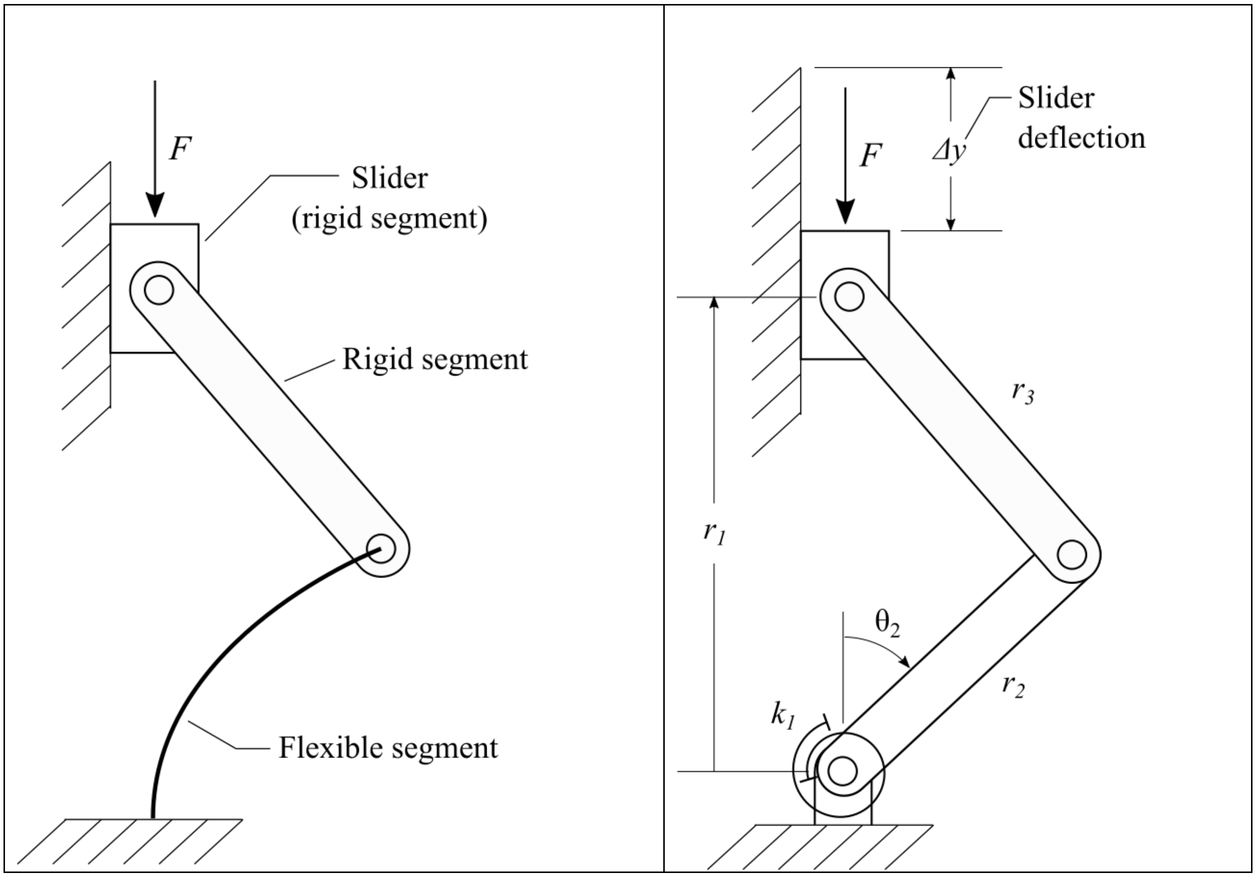


Figure 2.3 Simplified diagram of mechanism (left); PRBM of simplified mechanism (right).

A numerical model was made of the pseudo rigid body model (PRBM). It initially calculated force, *F*, from slider displacement, *Δy*. Given *Δy, r₂, r₃,* and *γ* (characteristic radius factor), the following was calculated:

$$r_1 = r_2 + r_3 - \Delta y, \qquad K_\Theta = \pi\gamma$$

where K_Θ is the stiffness coefficient. The second moment of area, *I*, is needed to calculate the stiffness of the torsional spring k_1, but this is more complicated than usual. The beam, which has a rectangular cross section, is being bent at different rotations. This means the second moment of area is different at each beam angle *Γ* and is calculated as follows:

$$I_{zz} = \frac{bh^3}{12}, I_{yy} = \frac{b^3h}{12}, \qquad I = \frac{I_{zz} + I_{yy}}{2} - \frac{I_{yy} - I_{zz}}{2}\cos 2\Gamma$$

where *b* is the base of beam cross section, and *h* is the height of beam cross section. This method is part of the 3D PRBM for Rectangular Cantilever Beams developed by Jairo Chimento (Chimento 2014). Next, the angular positions of the links (θ_2 and θ_3) and k_1 are calculated:

$$\theta_2 = \cos^{-1}\left(\frac{r_1^2 + r_2^2 - r_3^2}{2r_1r_2}\right), \qquad \theta_3 = \sin^{-1}\left(\frac{-r_2\sin\theta_2}{r_3}\right), \qquad k_1 = \frac{\gamma K_\Theta EI}{L}$$

where *E* is the elastic modulus and *L* is the length of the original compliant link ($L = r_2/\gamma$) . Finally, the force *F* at a given *Δy* is calculated using the following equation:

$$F = \frac{r_3\cos(\theta_3)k_1\theta_2}{r_2r_3\sin(\theta_2 - \theta_3)}$$

This process is repeated at increasing displacements up to 40% of the total length, and all data is stored in Excel. The result is graphed as force over displacement, as is shown in Figure 2.4. Here, *L* = 0.3m, γ = 0.83, r_2 = 0.249m, r_3 = 0.220m, *E* = 1.7GPa, *b* = 0.015m, and *h* = 0.0075m. Each line represents a different beam rotation ranging from *Γ* = 0° to *Γ* = 90°.

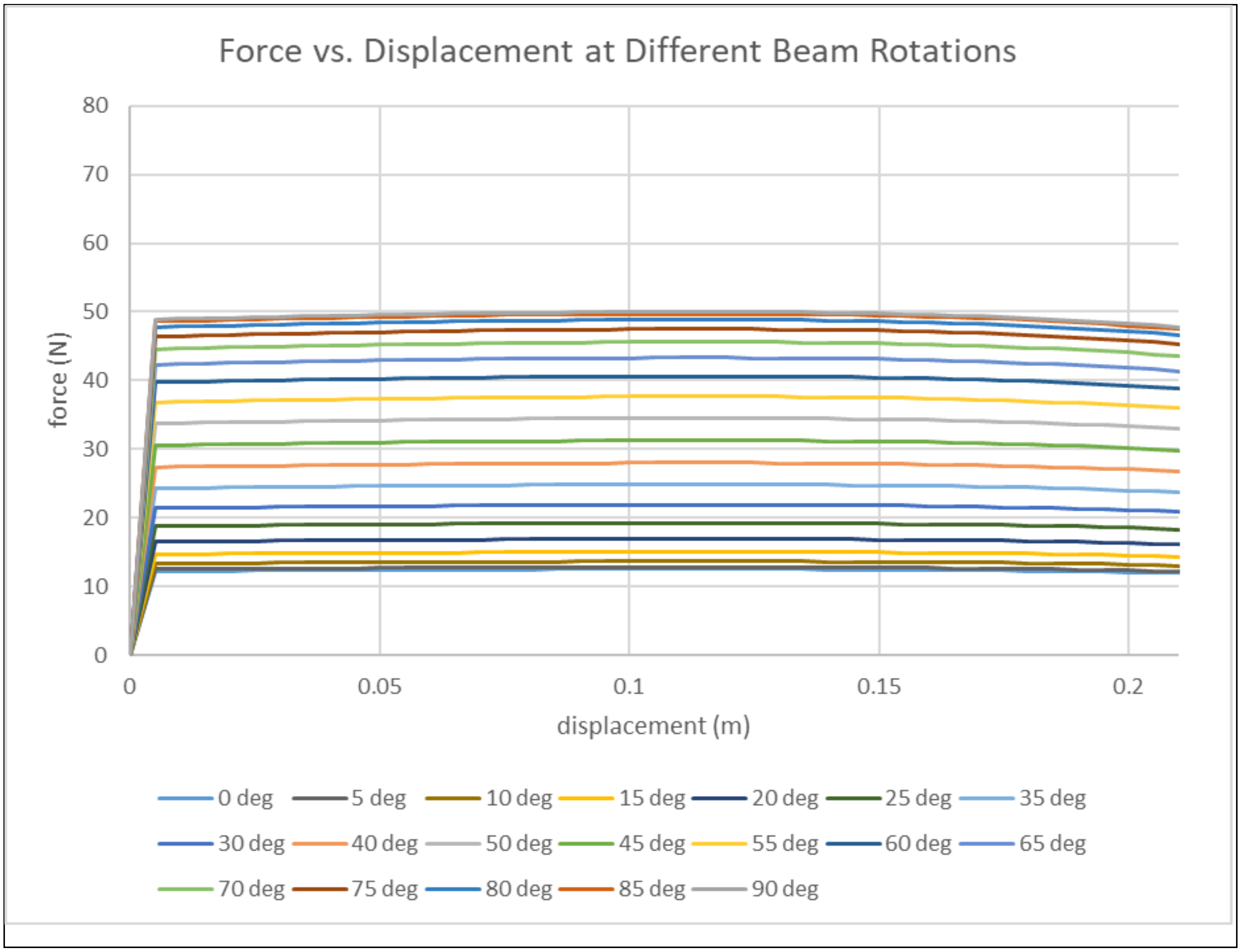


Figure 2.4 Plot from the numerical model. Each line represents a different beam rotation. 0° is the lowest, and 90° is the highest. The maximum displacement shown is 40% of the total length. R = 0.8853

Using this numerical model, design parameters like the length ratio, *R*, can be adjusted. The length ratio was defined above as the ratio between r_3 and r_2, and for a constant force mechanism it equals 0.8853. The plot in Figure 2.4 was made using this value, and it shows the horizontal lines that are to be expected from a constant-force mechanism. If *R* is decreased, the lines curve downward, which means the system exhibits negative stiffness. While negative stiffness has been used successfully in vibration isolation, the negative stiffness element is usually just part of a larger system which also contains positive stiffness elements (Le & Ahn 2013) (Yang, Xiong, & Xing 2013) (Shahadat et. al. 2015).

CHAPTER 3: PROTOTYPE

Based on the theory just discussed, the prototype was designed. There are many components that went into the overall structure of the apparatus, but in this chapter the focus is narrowed to the crank-slider mechanism portion. This includes: the side links, the joints, the compliant beam, the slider, and their attachments to the test apparatus. The labeled CAD model of the mechanism is shown below in Figure 3.1. The rest of the test apparatus will be discussed in the next chapter. This chapter gives qualitative descriptions and the rationale for the design choices. The dimensioned drawings of each component are given in Appendix A.

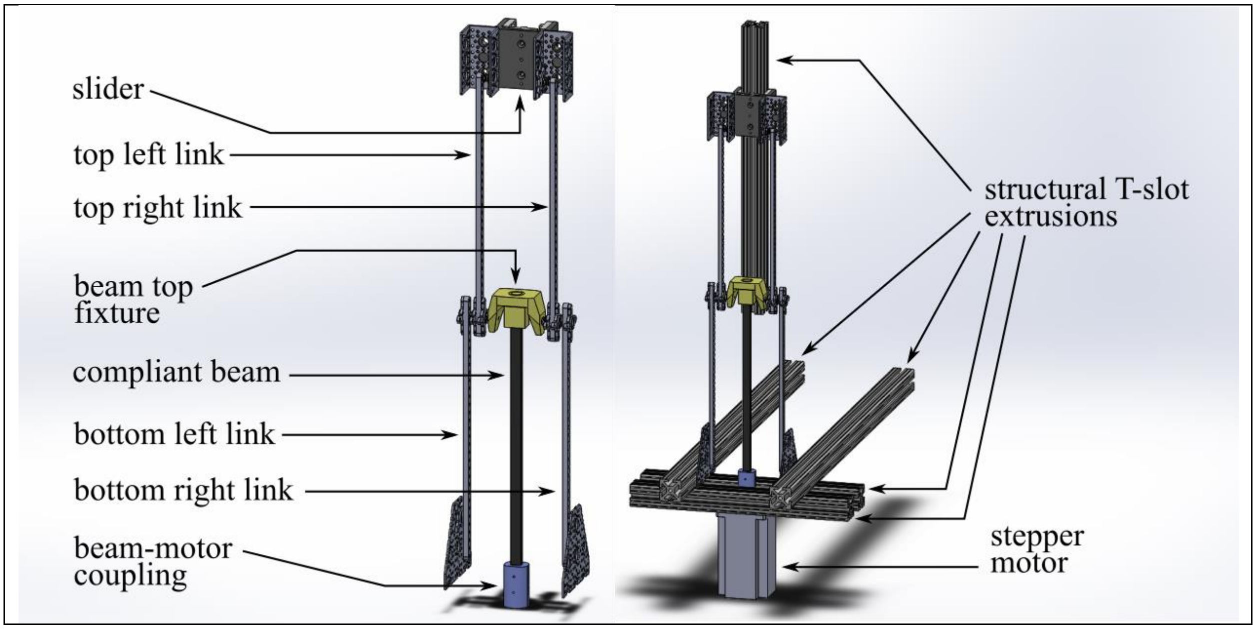
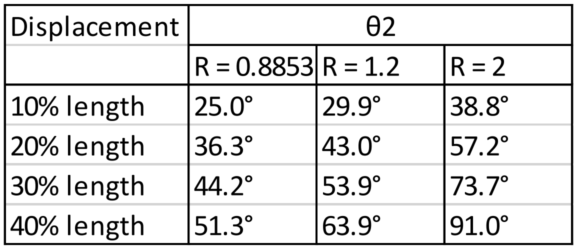


Figure 3.1 The mechanism is shown bare (left) and with the relevant structure (right).

3.1 Links and Joints

Before ordering the links, a value for the length ratio, *R*, was chosen. Because the system discussed in this thesis does not have additional elements to compliment the negative stiffness that comes from $R < 0.8853$, negative stiffness was determined to be undesirable and values of $R < 0.8853$ were ignored. Increasing R makes the lines curve upward, resulting in a desirable increase in stiffness. However, there is an upper limit. If R is increased too much, the mechanism gets too long with respect to the length of the beam, and the beam bends a lot to achieve the same displacement. In Table 3.1, three different R values are compared. For each one, the bending angle of the beam (θ_2) is shown for 4 different displacements.

Table 3.1 Bending angle of beam, θ_2, vs displacement, Δy, for different length ratios, R.



Displacement	θ2		
	R = 0.8853	R = 1.2	R = 2
10% length	25.0°	29.9°	38.8°
20% length	36.3°	43.0°	57.2°
30% length	44.2°	53.9°	73.7°
40% length	51.3°	63.9°	91.0°

When the beam bends past 60°, excessive plastic deformation occurs. Also, the design does not require that the max displacement get above 40%. Based on this data, I decided on a length ratio value of 1.2. This gives the highest stiffness without overbending the beam. The same plot from Figure 2.4 is shown below in Figure 3.2, but with R changed to 1.2.

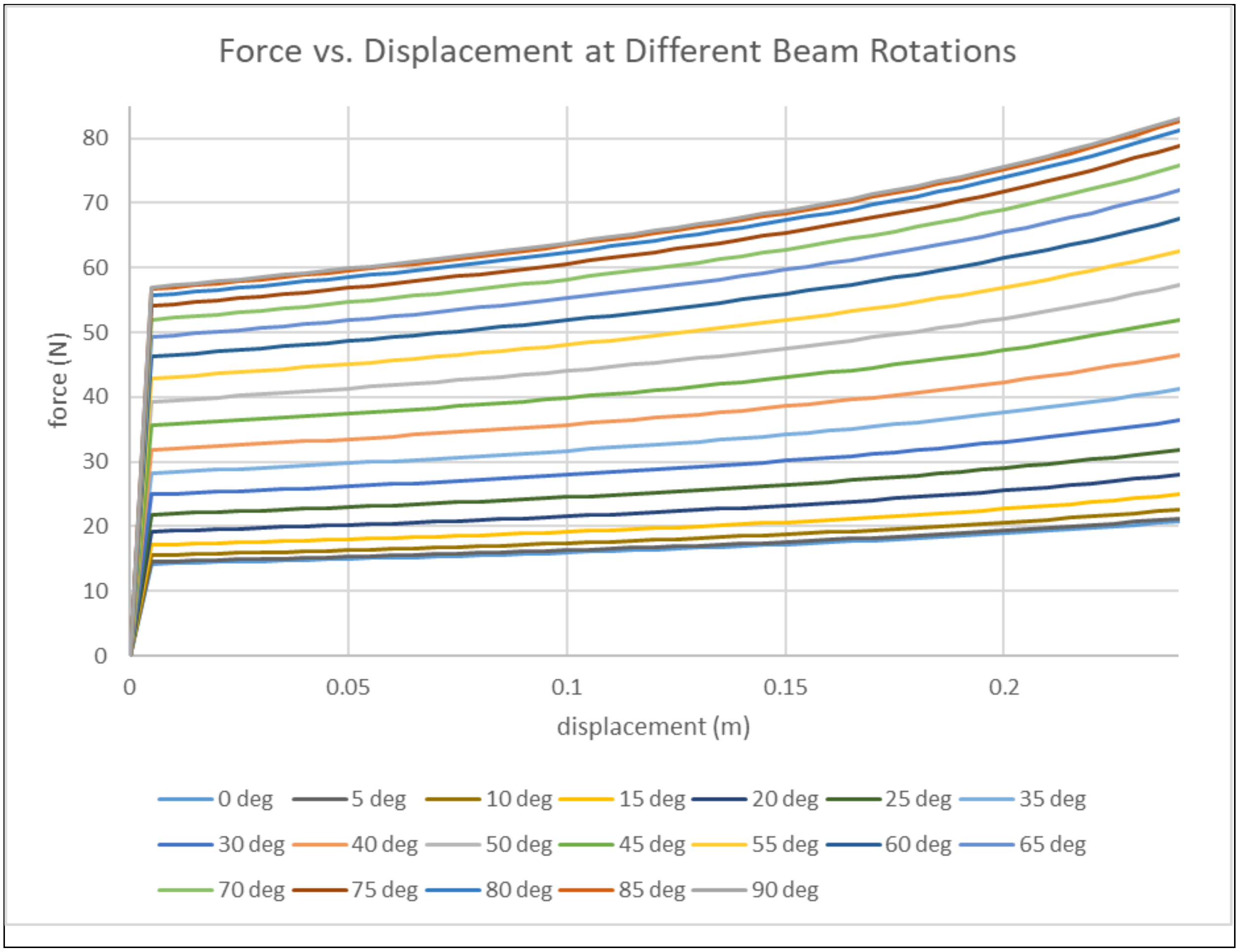


Figure 3.2 Plot from the numerical model. Each line represents a different beam rotation. 0° is the lowest, and 90° is the highest. The maximum displacement shown is 40% of the total length. *R* = 1.2

The next challenge was finding a source for the four links (top left, top right, bottom left, and bottom right). The links were required to be stiff, light, strong, preferably adjustable, and have solid joints that could rotate freely with minimal friction. This last requirement was particularly constraining. The joint needed to be tightened without increasing friction. I chose to use an assembly of Actobotics 33-Hole Aluminum Beams and Face Thru-Hole Pillow Blocks. A total of ten pillow blocks were used, and were fastened to the links at either end, allowing free rotation because of their bearings. To make a joint, a screw was passed through selected holes in the two links and small washers were used to keep the links from touching. Then, a locknut was fastened behind the link to hold the joint together. The holes in the links allowed for convenient adjustability in link length by moving the attachment position of the pillow blocks. A CAD model of a link with an attached pillow block is shown in Figure 3.3.

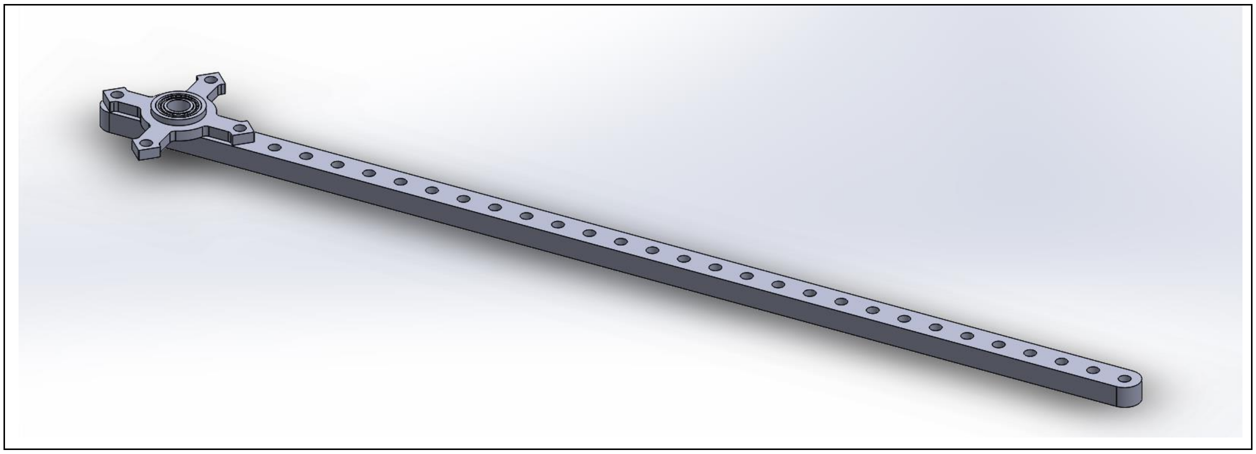


Figure 3.3 A rigid link (Actobotics 33-hole aluminum beam) with a face thru-hole pillow block on the end.

3.2 Compliant Beam

Next, a compliant beam was chosen. Two materials were initially considered: polypropylene and steel. Polypropylene (PP) beams were used for two reasons. The first reason was the ease of producing PP beams. They can be cut out of a sheet using USF's Compliant Mechanisms Research Lab (CMRL) laser cutter. Figure 3.4 shows one of the laser-cut PP beams. The second reason is that the Lab has a supply of PP sheets and they are relatively inexpensive. This allows for both mass production and easy adjustability. It became very practical to have a new beam for every test, and it was easy to cut a different beam if the current one is not desirable due to excessive plastic deformation. By contrast, the steel beams are expensive and difficult to machine to specific cross-sectional areas, making it impractical to obtain even one steel beam, much less multiple beams for testing purposes.

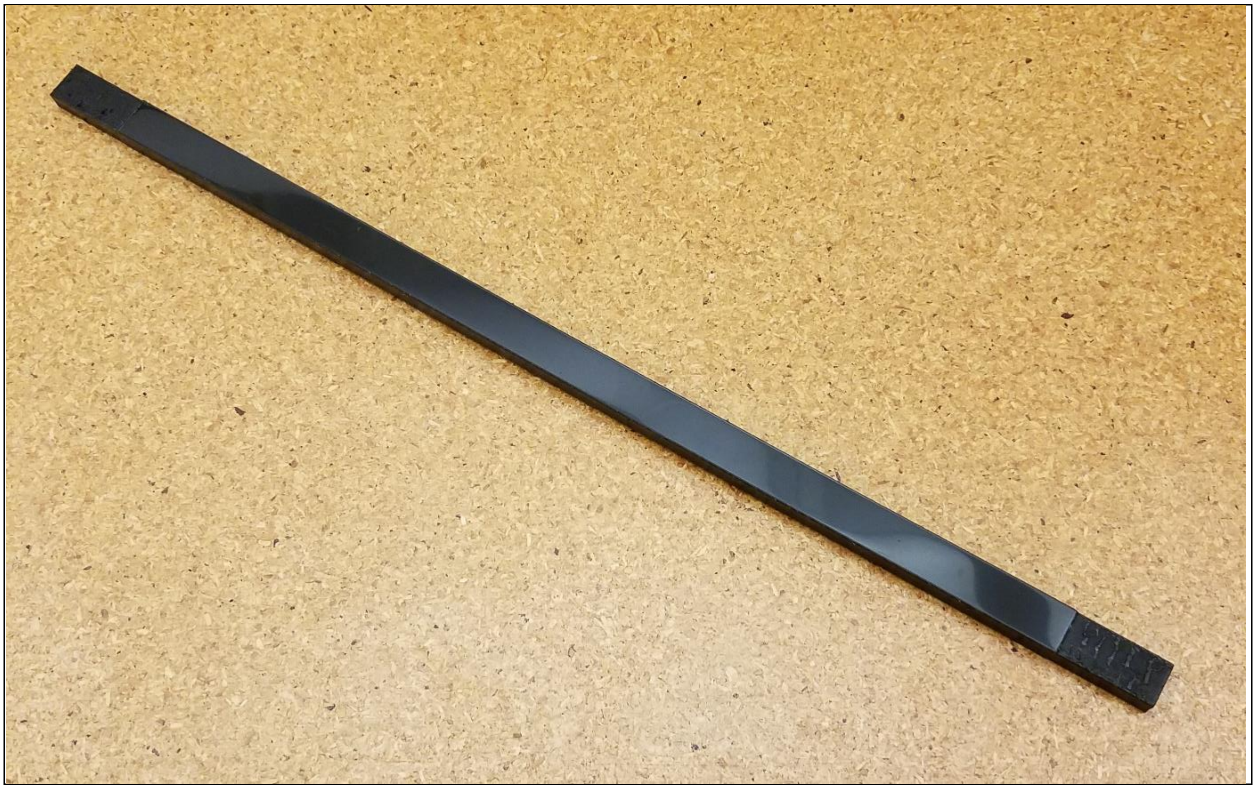


Figure 3.4 One of the polypropylene beams made using the laser cutter.

3.3 Beam Couplings

The bottom of the beam needed to be attached to the motor and the beam top attached to the top end of the bottom link. These attachments required fixtures that were custom made using 3D printing (Fused Deposition Modeling i.e. FDM) in USF's Advanced Visualization Center. The bottom fixture was a simple coupling that joined the PP beam to the shaft of the stepper motor. Both the beam and the motor shaft are press fit into the coupling. A CAD model of the coupling is shown in Figure 3.5.

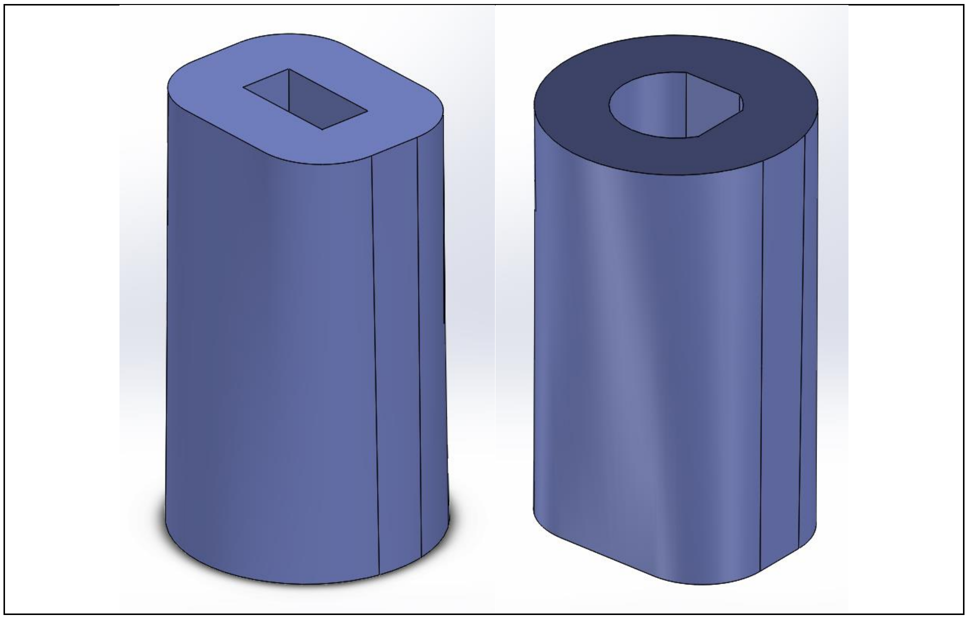


Figure 3.5 Two views of the beam-motor coupling. In the left image, the aperture that accepts the rectangular compliant beam is shown. In the right image, the aperture that accepts the motor shaft is shown.

The top fixture has two separate parts. The smaller part is press fit onto the top of the compliant beam and ends in a cylindrical boss that fits loosely into a bearing contained by the second, larger part. This larger part is joined on both the left and right sides to the joint between the top and bottom links, and can freely rotate about the joint's axis. This fixture helps constrain the bending of the beam. The free rotation constrains the position of the top of the beam, but not the angle, while the boss loosely fitting into the bearing lets the beam slip slightly so that it is never in compression. Both parts of this fixture are shown in Figure 3.6.

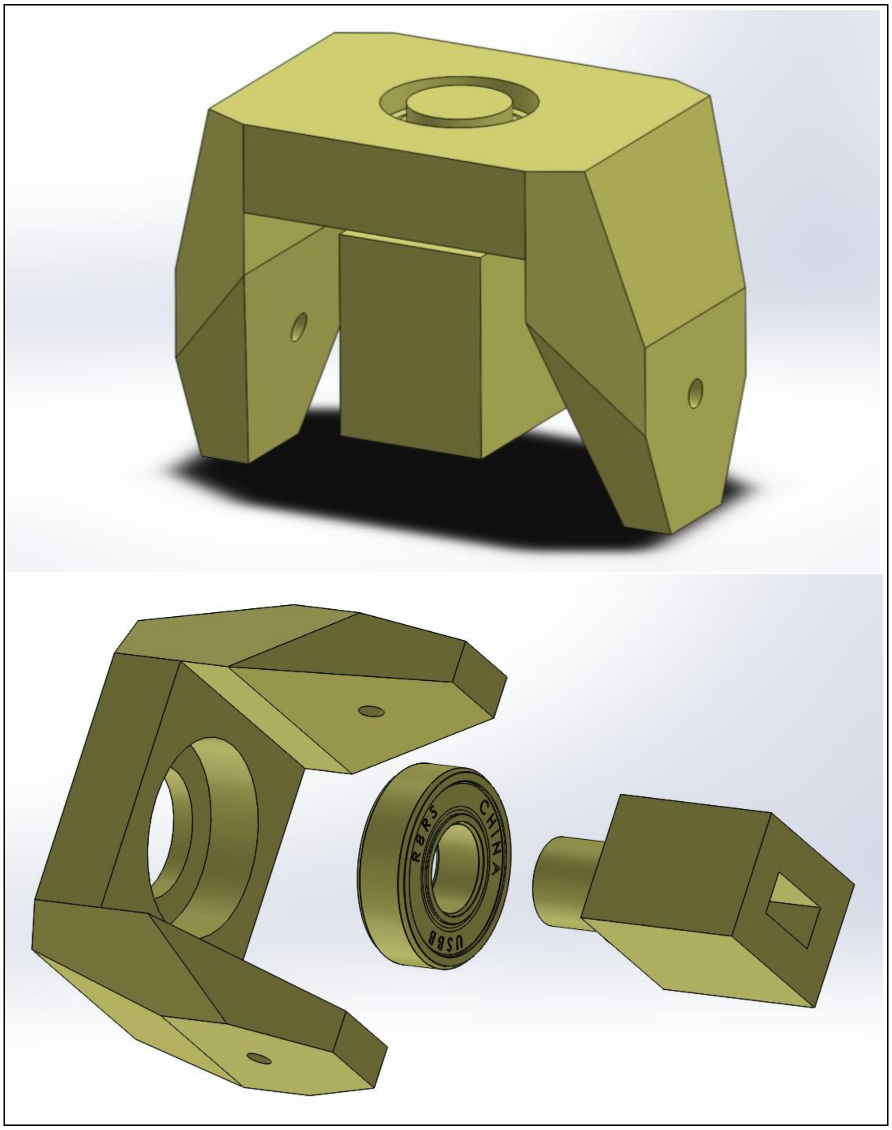


Figure 3.6 Assembled (top) and exploded (bottom) view of the fixture for the beam top.

3.4 Slider

The slider was part of the Futura T-slots system that most of the test structure is made out of. It is designed to slide along one of the aluminum structural members. The contact surface is made of ultra-high-molecular-weight polyethylene (UHMWPE). This material has a very low coefficient of friction, lower than nylon and comparable to PTFE (Teflon). However, it has better abrasion resistance than PTFE. On top of that, it is self-lubricating. These properties make it ideal for use as a slider in this mechanism, and it can be seen in Figure 3.7. In order to connect it to the links, two 3” aluminum channels with pillow blocks were fastened to the slider. The links were then fastened to the pillow blocks.

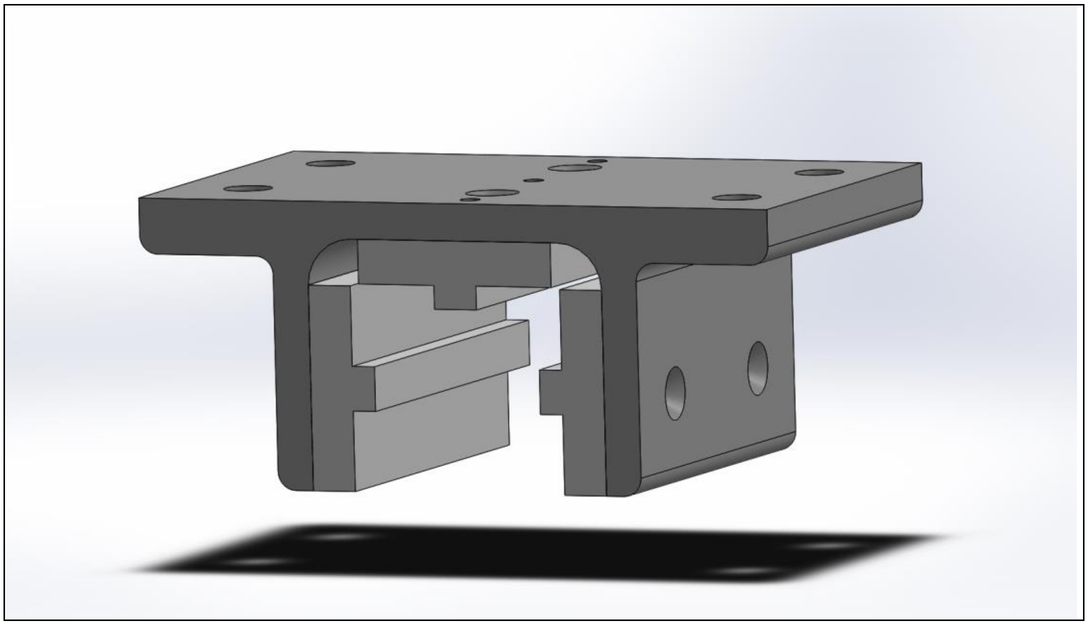


Figure 3.7 A CAD model of the slider.

CHAPTER 4: TEST APPARATUS

Now that the prototype's components have been discussed, this chapter describes the test apparatus. The test apparatus components are both mechanical and electrical.

4.1 Mechanical Components

Perhaps one of the biggest early decisions was choosing the stepper motor. The shaft of the motor is coupled to the beam, allowing the motor to control the beam's twist angle. Based on the numerical model that was made of the system in Chapter 2, it was predicted that the motor may have to withstand torques of up to 10-12 Nm. Because of this, I chose the Nema 34 Hybrid Stepper Motor which has a holding torque of 13 Nm. This conservative choice gives more than enough power to control the beam and deal with any loads it may endure.

Another important choice was the distance sensors. We needed a distance sensor that was accurate, fast, had a range of about two feet, was compatible with the Arduino microprocessor, was relatively small, and was reasonably priced. The Migatron LCU-40APW was chosen. It is an analog ultrasonic sensor with a range of 4"-40", takes about 100 measurements per second, and has a resolution of 0.039". It also was relatively low cost, and fits comfortably in the palm of your hand. The resolution mentioned is calculated with respect to the system discussed in this thesis, which uses an Arduino to measure the analog voltage output of the sensor. It is sufficient for this thesis, but a finer resolution may be possible with another system.

As mentioned briefly in the previous chapter, most of the structure of the apparatus was made using parts from the Futura T-slots system. Its main feature is precision aluminum extrusions which can be cut and joined together to make strong, stable structures. All of the supporting framework in this test apparatus is made of extrusions with a cross section that measures 1.5” square. They are fastened to each other along their slots, which is important because it means that the joints can be loosened and the components can slide to a new location if desired. This makes design and assembly less difficult, and also allows for easy modifications to the apparatus. These were both desirable qualities. The T-slot frame supports the bottom links of the mechanism, the slider at the top, the sensor fixture, and the stepper motor.

The mechanism’s range of motion needed to be limited for a few reasons. First is that with the current design, the slider will fall off of its extrusion if it goes too low, and the mechanism will hit a dead center if it reaches the upper limit of its range. From the dead center, the mechanism could bend “elbow in” instead of “elbow out”. This will cause the top beam fixture to interfere with the structure, and is not the intended use of the mechanism. But the main reason to limit the range is to avoid overstressing the beam. The more the beam is allowed to bend, the quicker it will fatigue. To reduce the motion range, two hard stops were placed on the slider extrusion. Each hard stop is a large nut that is bolted to the slot, blocking the motion of the slider. The top stop was placed just below the upper limit of the mechanism, not allowing it to fully straighten out. The bottom stop was placed about halfway down initially, but was tested at different positions. Because of the T-slots, the hard stops could be securely fastened anywhere along the length of the extrusion.

The complete test apparatus (electrical components not pictured) is shown in Figure 4.1.

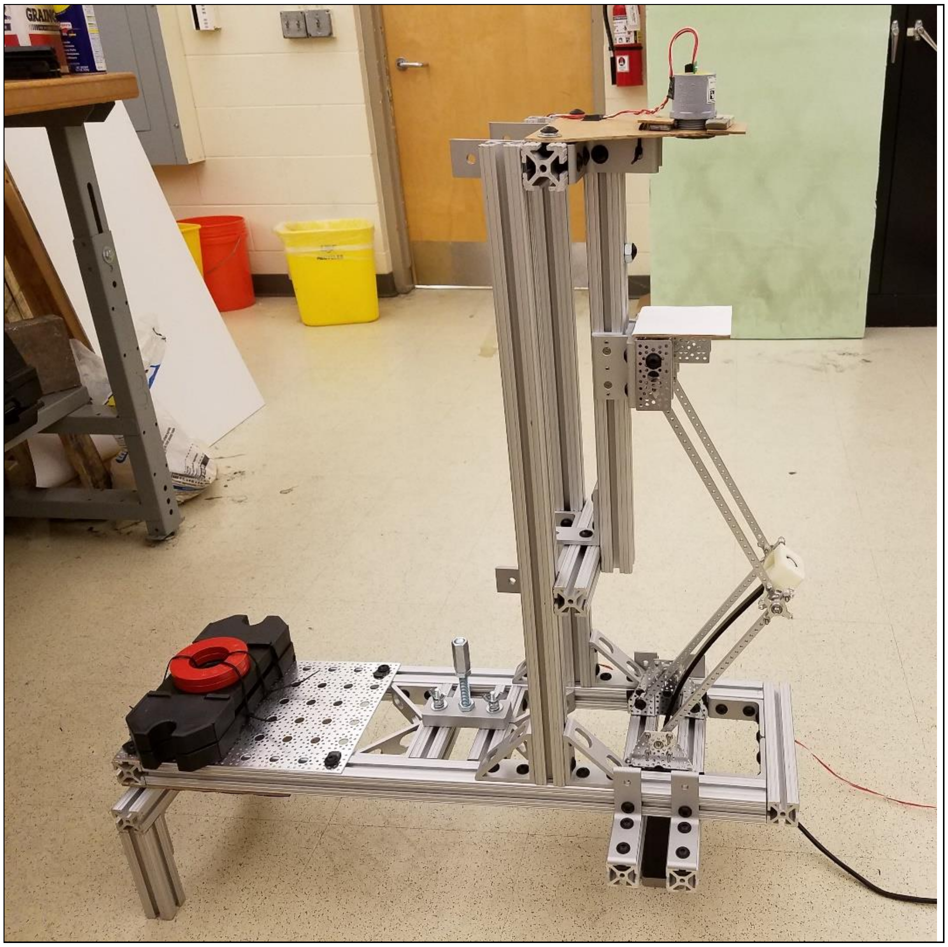
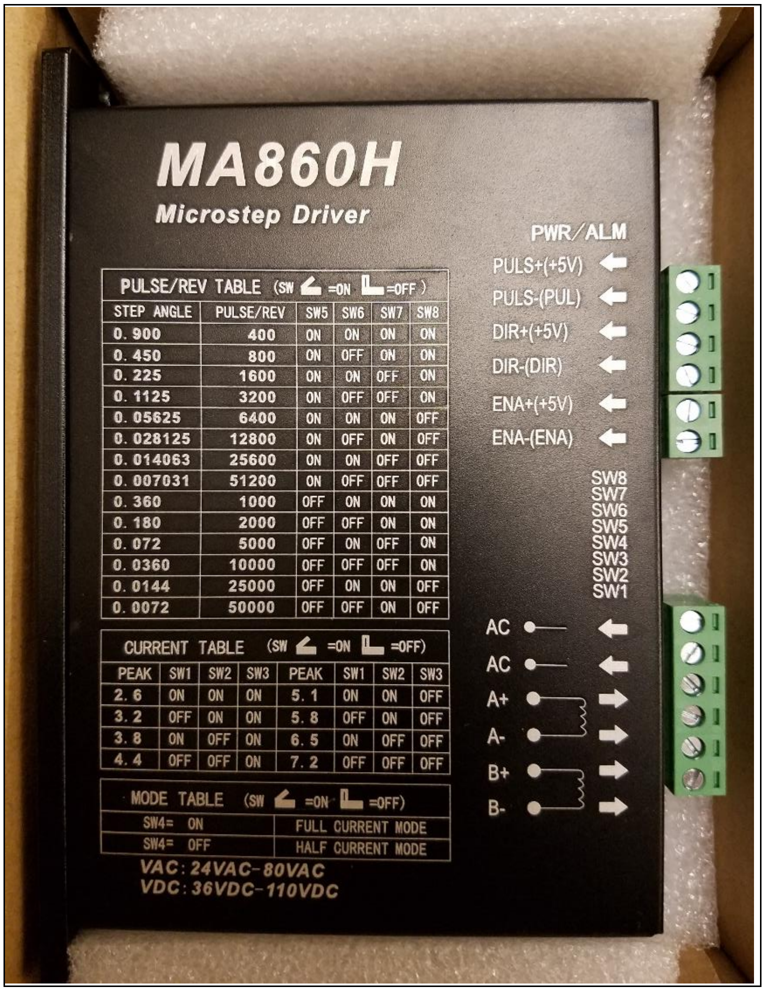


Figure 4.1 Completed test apparatus.

4.2 Electrical Components

The electrical components of the system are gathered together for organization in a laser-cut wooden case. The electrical components have two functions: to control the stepper motor, and to measure displacements.

The two biggest parts are the power supply and the stepper motor driver. The power supply receives 120V AC from a standard outlet and outputs 48V DC at 7.3A. This powers the driver and in turn, the stepper motor. The driver is an MA860H Microstep Driver, which is shown in Figure 4.2. It has 8 inputs and 4 outputs. Two inputs receive power from the power supply and the remaining 6 come from the Arduino and are used to control the motor. They are split into 3 +/- pairs, labeled “PULS”, “DIR”, and “ENA”. The “+” is the actual input, and the “-” is its ground. Each input is either low or high. ENA stands for Enable, and controls the power state of the driver. As long as this is low, the driver is powered on. DIR stands for direction, and controls the direction the motor steps in. A low signal sets the direction to clockwise, while a high signal is counterclockwise. Finally, PULS stands for pulse, and providing an electrical pulse (low to high) advances the motor one step in the chosen direction. The 4 outputs on the driver correspond to the 4 wires from the stepper motor. They are labeled A+, A-, B+, B-. They match with the motor’s colored wires as follows: A+ = red, A- = green, B+ = yellow, B- = blue. The wires that run from the power supply and from the stepper motor to the driver are 18 gauge. This is thicker than the rest of the wiring, which is either 28 or 24 gauge. The driver has several different resolution options for the stepper motor, from 400 to 51200 pulses/revolution. I chose the lowest option, 400 pulse/rev, because it was the fastest and having 0.9° per step was fine enough.



PULSE/REV TABLE (SW ◢ =ON ▙ =OFF)					
STEP ANGLE	PULSE/REV	SW5	SW6	SW7	SW8
0. 900	400	ON	ON	ON	ON
0. 450	800	ON	OFF	ON	ON
0. 225	1600	ON	ON	OFF	ON
0. 1125	3200	ON	OFF	OFF	ON
0. 05625	6400	ON	ON	ON	OFF
0. 028125	12800	ON	OFF	ON	OFF
0. 014063	25600	ON	ON	OFF	OFF
0. 007031	51200	ON	OFF	OFF	OFF
0. 360	1000	OFF	ON	ON	ON
0. 180	2000	OFF	OFF	ON	ON
0. 072	5000	OFF	ON	OFF	ON
0. 0360	10000	OFF	OFF	OFF	ON
0. 0144	25000	OFF	ON	ON	OFF
0. 0072	50000	OFF	OFF	ON	OFF

CURRENT TABLE (SW ◢ =ON ▙ =OFF)							
PEAK	SW1	SW2	SW3	PEAK	SW1	SW2	SW3
2. 6	ON	ON	ON	5. 1	ON	ON	OFF
3. 2	OFF	ON	ON	5. 8	OFF	ON	OFF
3. 8	ON	OFF	ON	6. 5	ON	OFF	OFF
4. 4	OFF	OFF	ON	7. 2	OFF	OFF	OFF

MODE TABLE (SW ◢ =ON ▙ =OFF)	
SW4= ON	FULL CURRENT MODE
SW4= OFF	HALF CURRENT MODE

VAC: 24VAC-80VAC
VDC: 36VDC-110VDC

Figure 4.2 MA860H Microstep Driver

The other main electronic component is the microcontroller. This had to serve as the heart of the electronic system, being able to interact with all the components. Although there were other worthy candidates, the Arduino UNO was chosen mostly because of my own familiarity with it. However, its popularity means that there is a lot of online support which was very helpful during the coding process.

One of the early electrical challenges was limiting the voltage output of the distance sensors. They output 1-10V, which corresponded linearly with their 4-40” range. However, the Arduino can only handle 5V maximum. To fix this, I made a voltage divider circuit, which scales the voltage by a factor of 0.5. This made the new voltage range 0.5-5V, which was perfect for the system. The diagram of the voltage divider is shown in Figure 4.3.

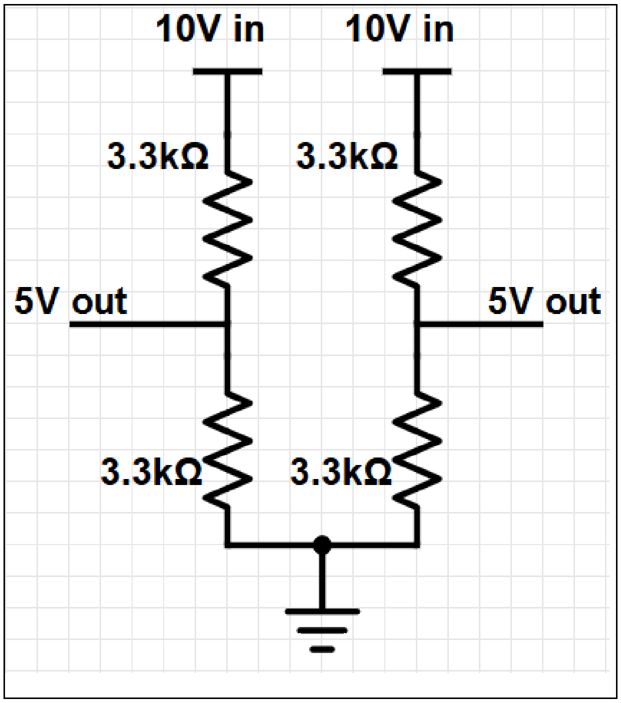


Figure 4.3 Diagram of the voltage divider circuit used in the system. It scales the output voltage of the sensors by a factor of 0.5.

All of the electronic components were mounted in a wooden case, both for ease of transport and to protect them. The case was custom built from laser-cut wood, and has 6 ports: one each for the two distance sensors, and one for each of the four stepper motor wires. The finished electronics assembly is shown in Figure 4.4 and the corresponding wiring diagram can be seen in Figure 4.5.

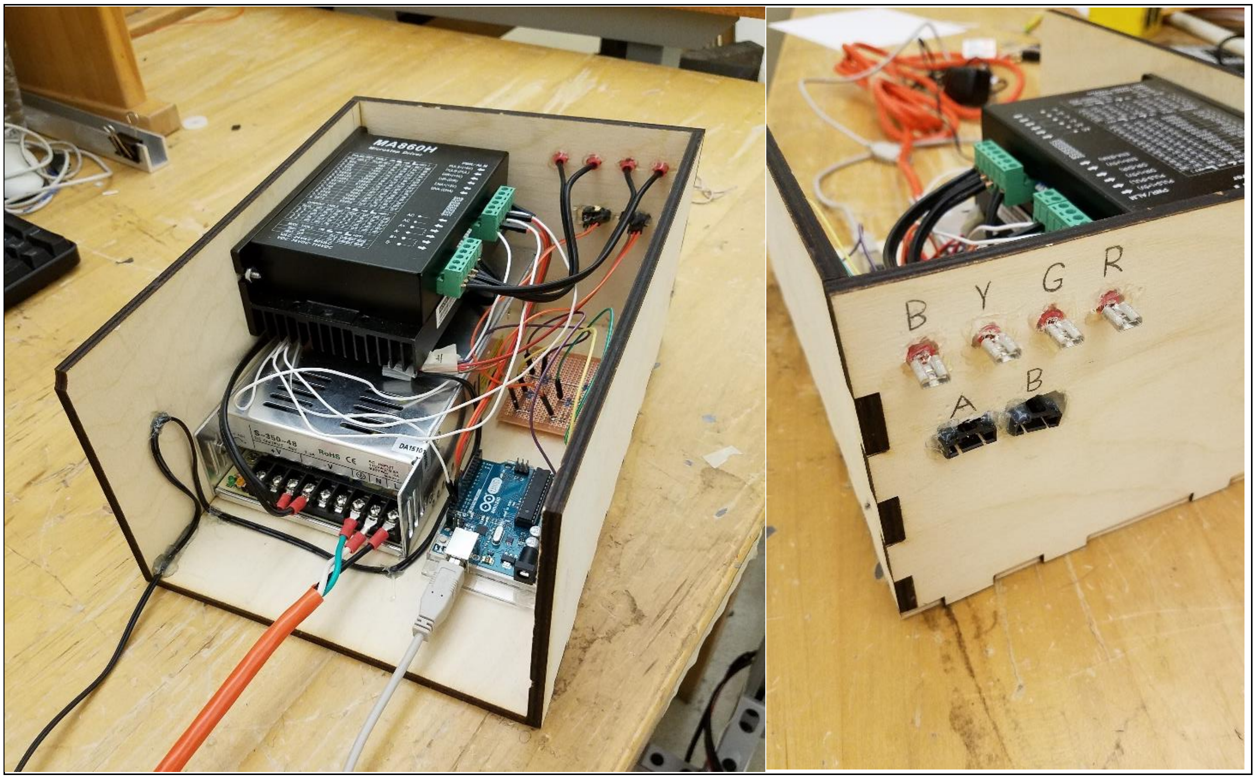


Figure 4.4 The electrical components in the laser-cut case. The left image shows the front of the case, while the right image shows the ports on the back.

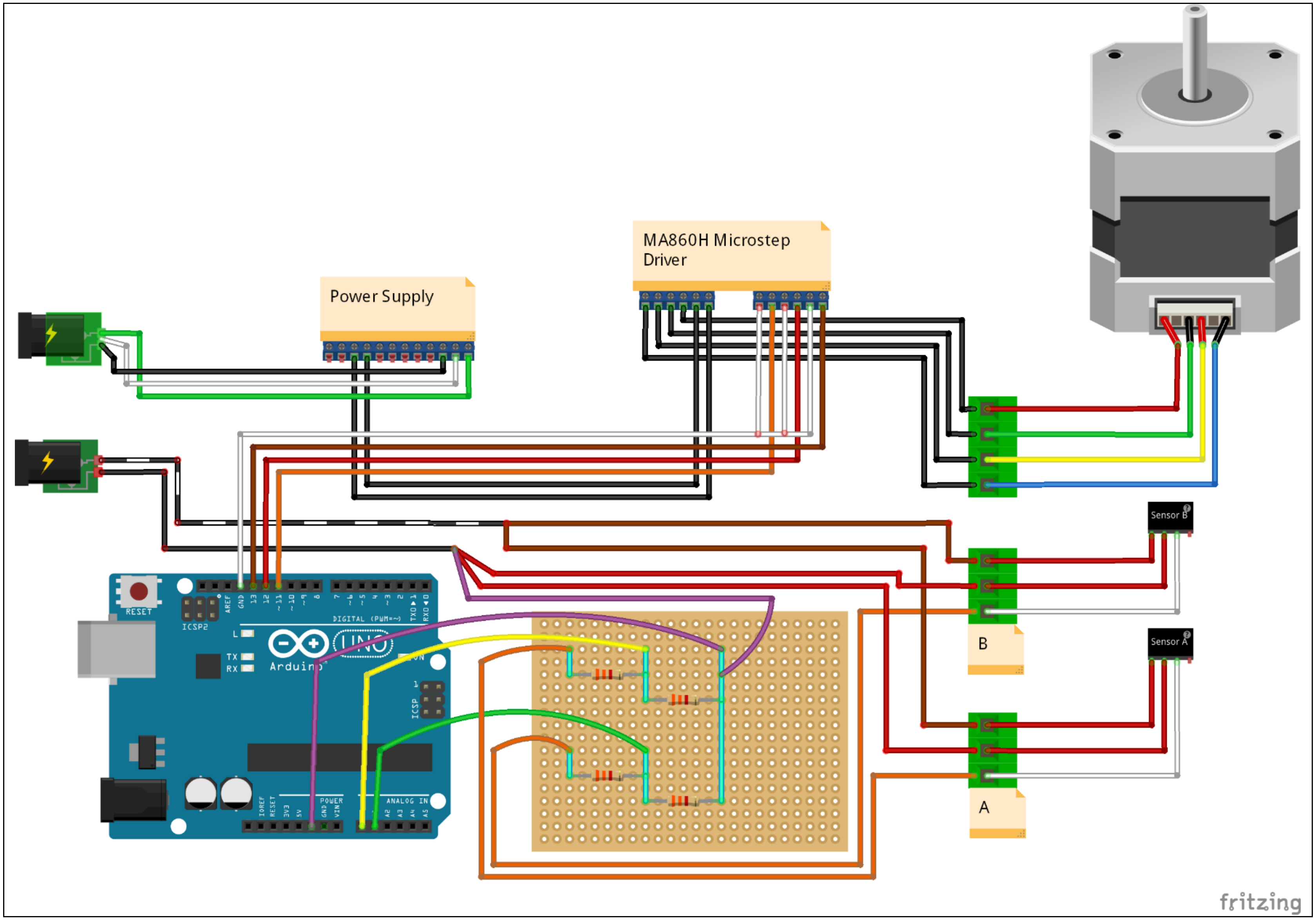


Figure 4.5 Wiring diagram for all electronic components.

CHAPTER 5: TEST PROCEDURES

Ten types of tests were performed on the system. They were all based on the same Arduino code, but with small variations. The basic process of the original code was as follows. The motor twists the base of the beam from 0° and 108° and back, three times, while collecting distance data from the sensor. During the actual test each program ran 3 times back to back, so that the system completed 9 loops total. The various tests performed used this template but changed some of the test parameters, including: the position of bottom hard stop, the speed of beam rotation, length of top link (r_3), the weight of top platform, and the twist reversal angle. Before each test, the zero position of the beam is set manually using a special program.

The parameters of Test 1 will be described in detail, and the other tests are described in less detail, assuming that the non-specified features are identical with this first test. Starting at a twist angle, $\Gamma = 1$, the motor steps up to 120 steps (108°) and back down to one at a speed of 0.978 steps/s. This loop is repeated 9 times. During each step, 10 readings are taken from the distance sensor and averaged, then this value is logged with the current step number. This process is repeated 10 times, resulting in total 100 measurements and 10 logged values for every step. The top link r_3 is 11.55" long, while the bottom link r_2 is 9.625". The bottom stop is 14.375" above the bottom joint of r_3, which allows for the slider to move up to about 29% of the effective mechanism length. The top stop is 22.75" above the same joint. This is about 1" less than the maximum platform height at dead center. The polypropylene beam is 13.25" long with an effective length of 11.25" (it is inserted into 1" deep slots at either end), and a cross section of 0.5" by 0.25". It is also new, meaning that it has not been stressed or used in any previous tests.

The data is collected and imported into Excel. The voltage value reported by the Arduino (0-1023) is converted into the distance r_1, which is measured in inches from the bottom joint of link r_2 to the top joint of link r_3. This is the conversion equation:

$$r_1 = 25.4375 - \left(\frac{V + 5.4577}{25.433}\right)$$

where V is voltage.

In Test 2 two things were changed. The bottom stop was moved up 1.5” to a position of 15.875”. Also, the joint near the top of the beam where the cylindrical boss mates with the bearing was lubricated with WD40.

For Test 3, the bottom stop was moved up 1.5” like in Test 2, and the bearing joint was lubricated. The top link r_3 was shortened by 0.77” to a length of 10.78”. The top stop was moved down 0.8” to a height of 21.95” to compensate for the shortening of the top link.

Test 4 had the bottom stop moved up 3” to a position of 17.375”. The bearing joint was lubricated.

In Test 5, the bottom stop was moved up 3”, the bearing joint was lubricated, and the speed was increased. A new Arduino code was used that decreased the delay between measurements from 100ms to 50ms. This increased the rotation speed to 1.919 steps/s.

In Test 6, the bottom stop was moved up 3”, the bearing joint was lubricated, and the speed was increased again. A new Arduino code decreased the delay between measurements to 25ms, which made the rotation speed 3.676 steps/s.

Test 7 was the final speed test. The bottom stop was moved up 3” and the bearing joint was lubricated. The new Arduino code set the delay between measurements to 10ms, which made the rotation speed 6.803 steps/s.

Test 8 was the first hysteresis test. The bottom stop was moved up 3" and the bearing joint was lubricated. A new Arduino code was used, and the delay between measurements was set back to 100ms. However, instead of rotating between 1 and 120 steps, it rotates up to 80, then down to 60, then back up to 80. It rotates between 60 and 80 six times total, before descending from 80 back down to 1. This code is only run once.

Test 9 was the second hysteresis test. It is the same as Test 8, but the rotation range was changed. The bottom stop was moved up 3", the bearing joint was lubricated, and a new Arduino code was used. This time, the motor rotates up to 75, then goes back and forth between 35 and 75 6 times. After, it goes from 75 back down to 1.

Test 10 was the weight test. It is identical to Test 4 but with weight added. The bottom stop was moved up 3" and the bearing joint was lubricated. Two weight plates weighing 0.25lbs each were fastened to the platform, for a total of 0.5lbs of added weight.

CHAPTER 6: RESULTS

Now that the test procedures have been described, the data collected from each test are presented. The first test, Test 1, consists of an initial run, Test 1.1, and its replicate, Test 1.2, which can be seen in Figures 6.1 and 6.2, respectively.

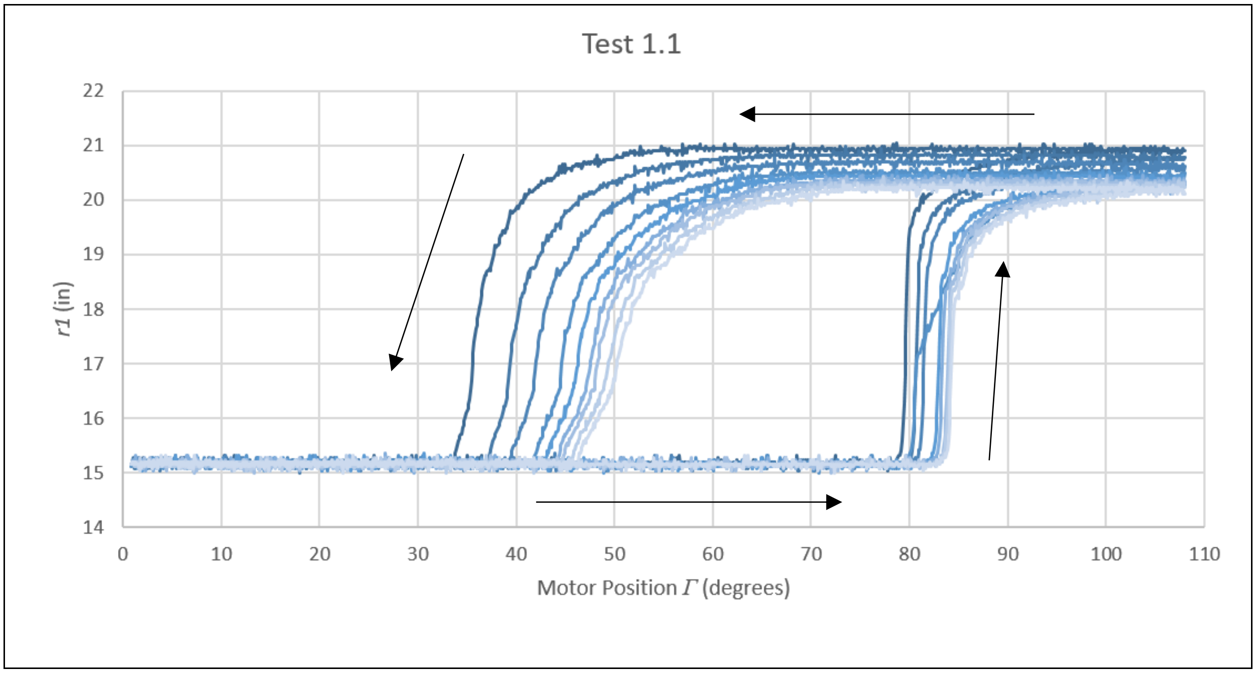


Figure 6.1 Plot of Test 1.1. *R1* is plotted over *Γ*. The line color gets lighter with each loop. The arrows show the path of the mechanism.

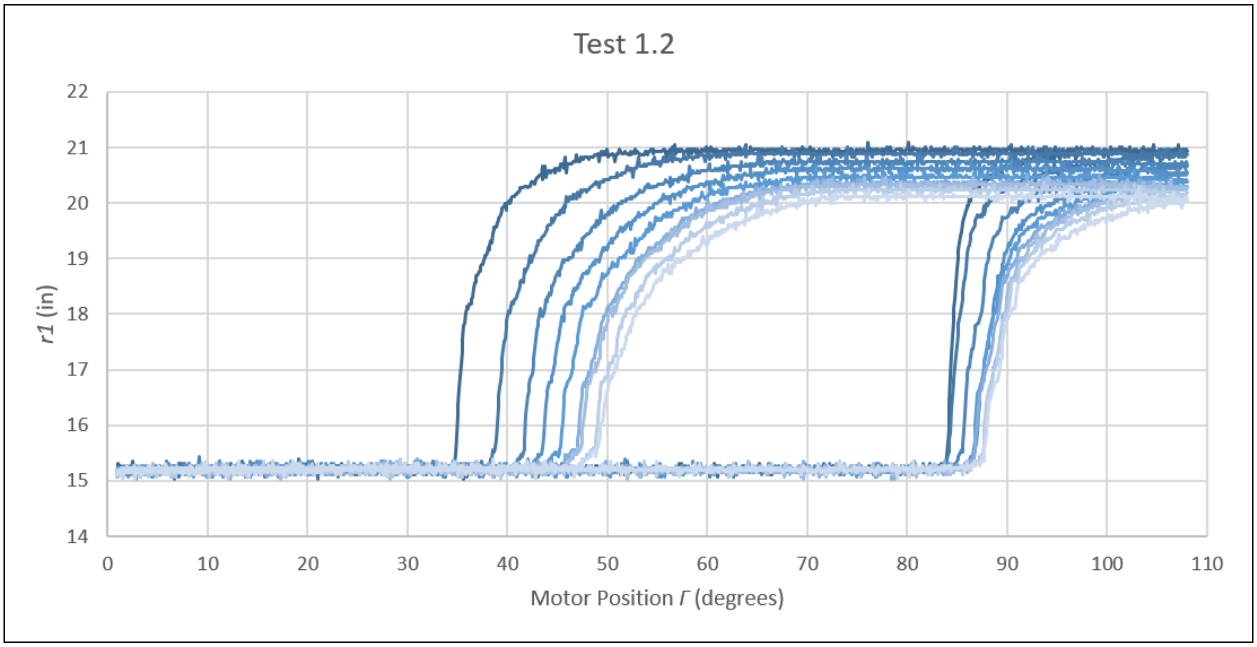


Figure 6.2 Plot of Test 1.2. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

Next, the results of Test 2 are shown in Figure 6.3. For this test, the bottom stop was moved up 1.5”. The test was performed only once.

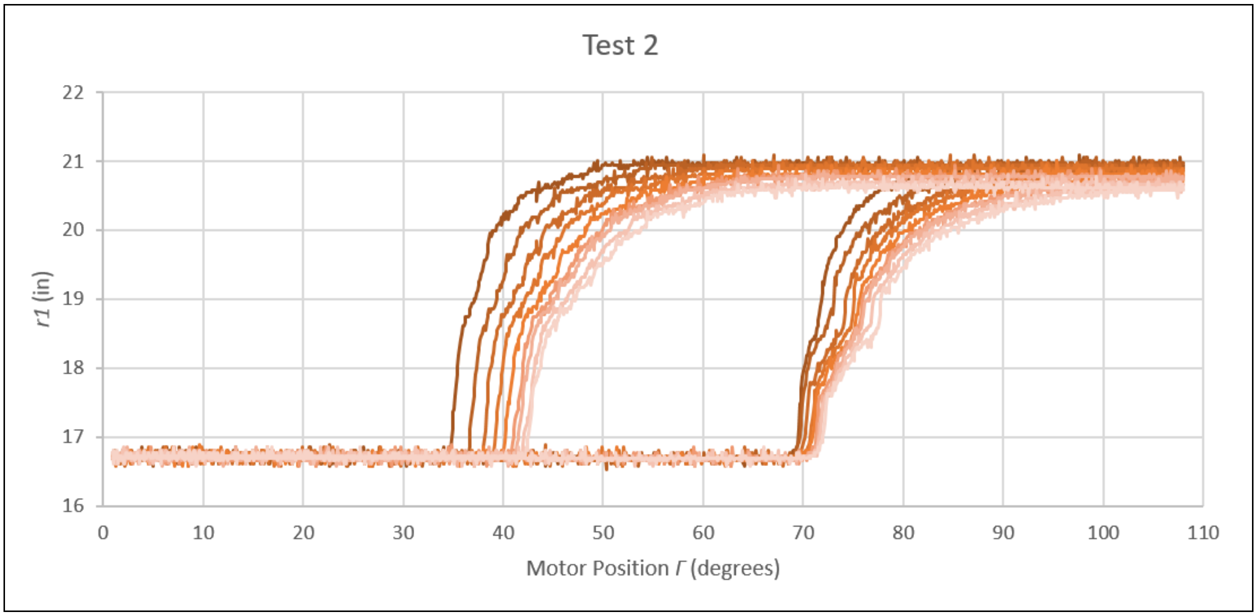


Figure 6.3 Plot of Test 2. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

For Test 3, the bottom stop was left unchanged from Test 2 but the top link was shortened. These results can be seen in Figure 6.4.

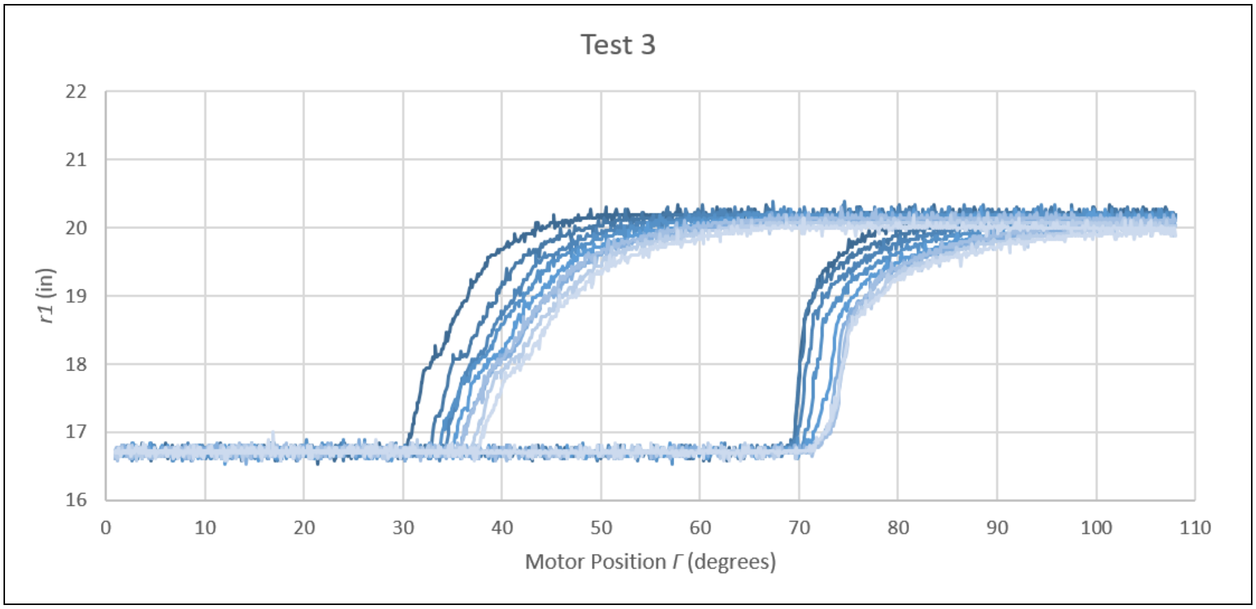


Figure 6.4 Plot of Test 3. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

Test 4 had the bottom stop moved up 3”. There was an initial run, and 2 replicates. Results are shown in Figures 6.5, 6.6, and 6.7.

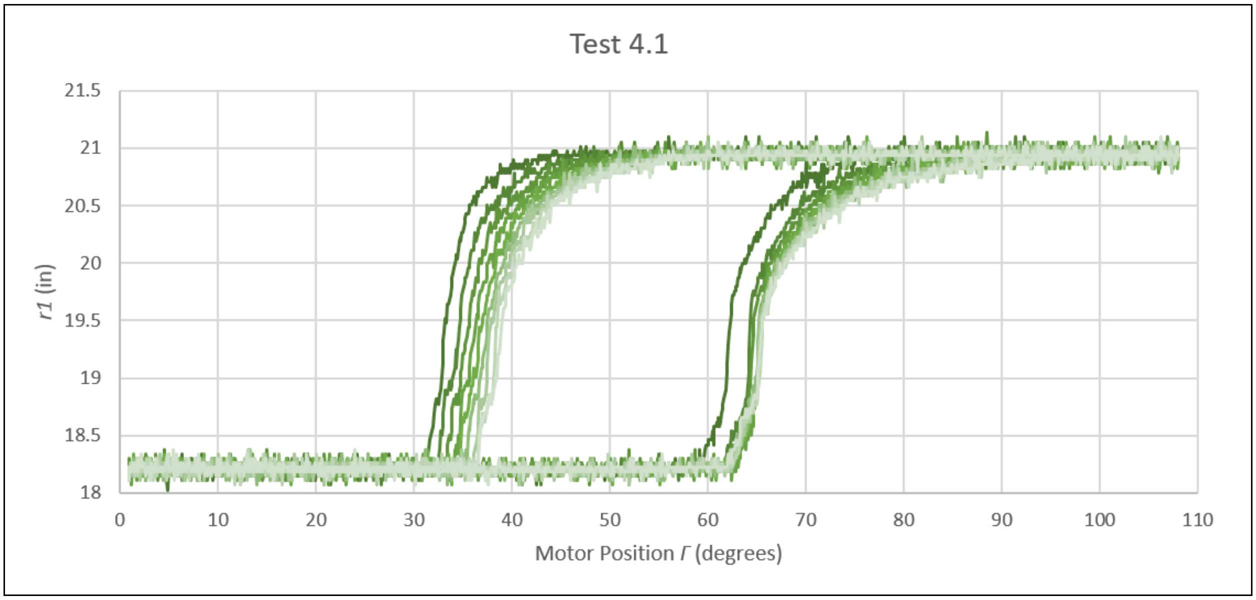


Figure 6.5 Plot of Test 4.1. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

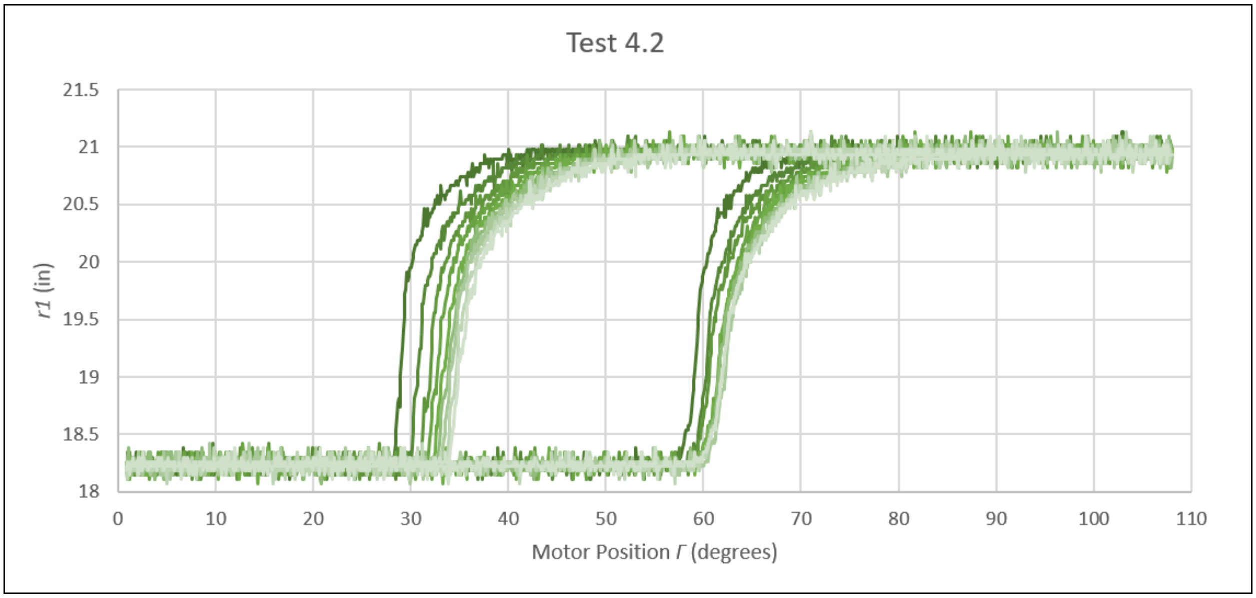


Figure 6.6 Plot of Test 4.2. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

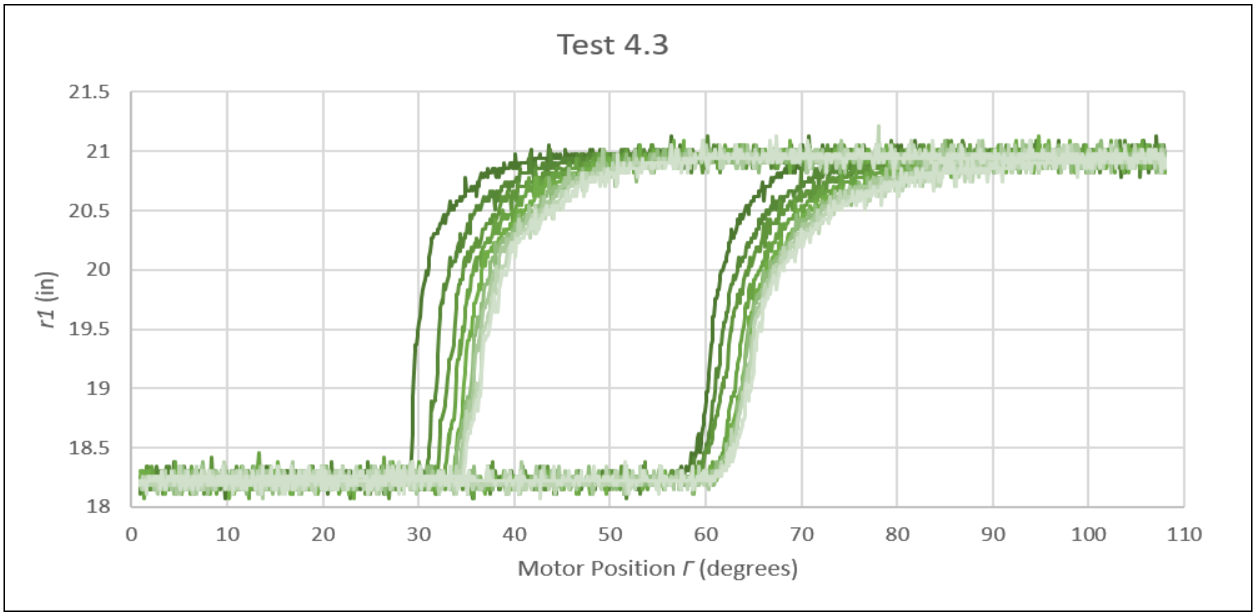


Figure 6.7 Plot of Test 4.3. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

For Test 5, speed was increased to 1.919 steps/s. Results are shown in Figure 6.8.

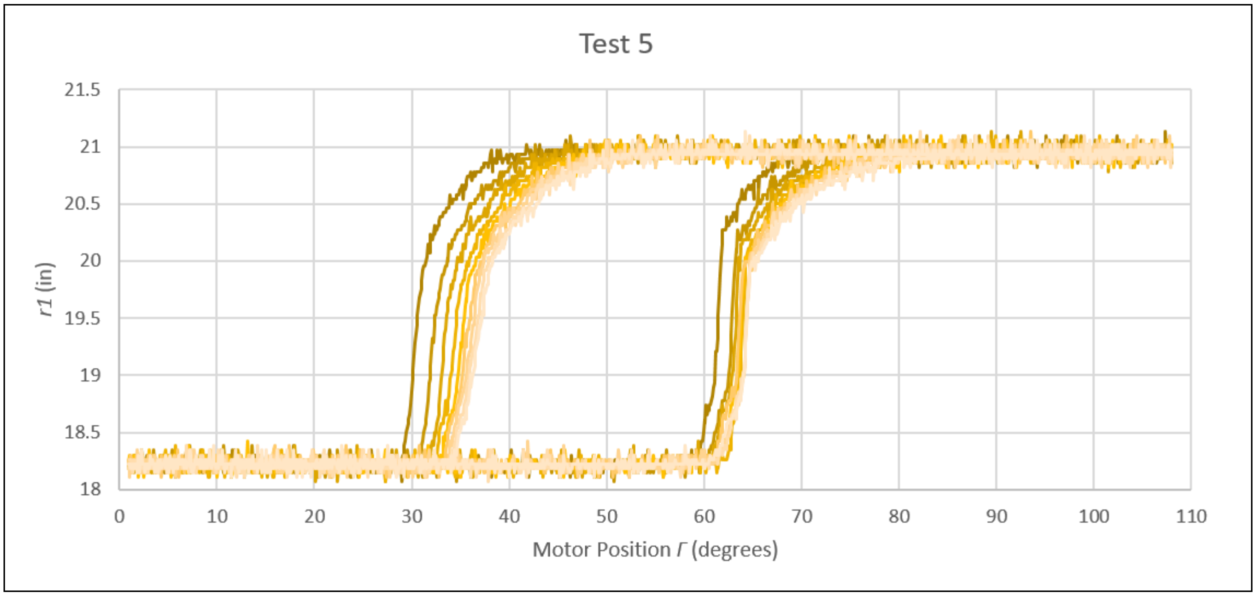


Figure 6.8 Plot of Test 5. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

For Test 6, speed was increased to 3.676 steps/s. Results are shown in Figure 6.9.

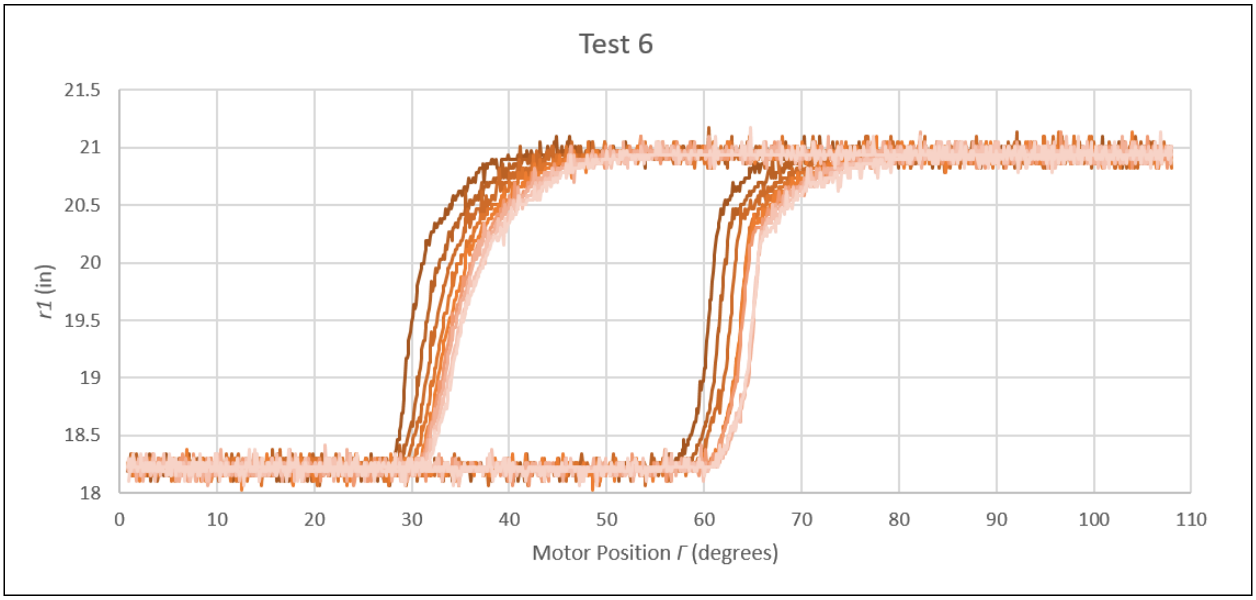


Figure 6.9 Plot of Test 6. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

For Test 7, speed was increased to 6.803 steps/s. There was an initial run, and 2 replicates. Results are shown in Figures 6.10, 6.11, and 6.12.

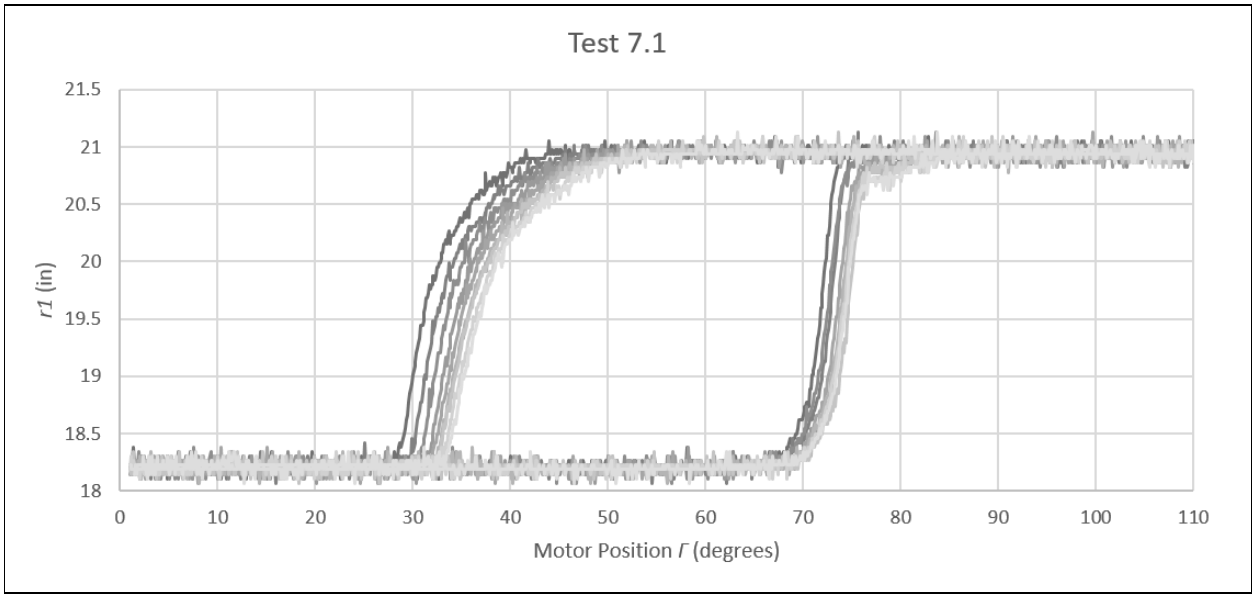


Figure 6.10 Plot of Test 7.1. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

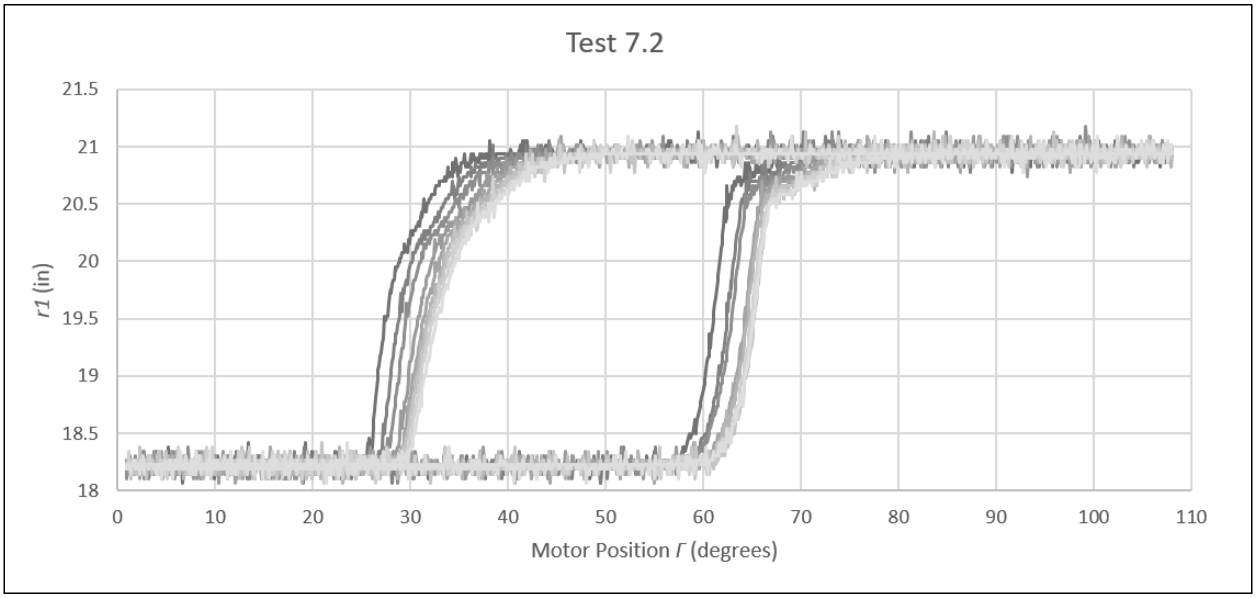


Figure 6.11 Plot of Test 7.2. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

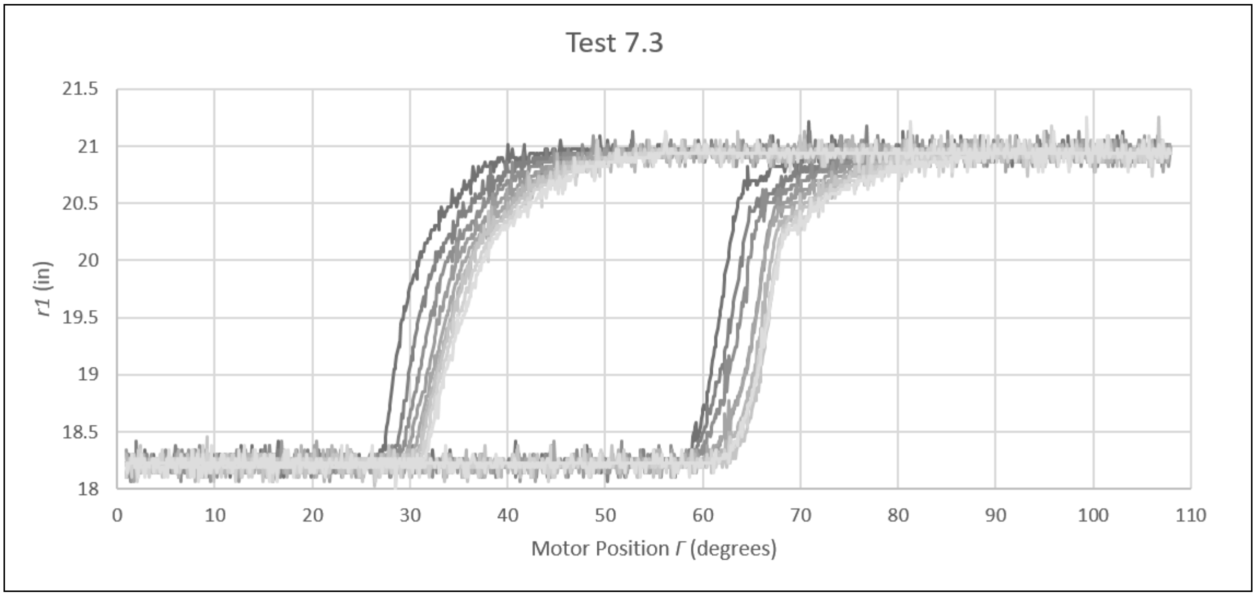


Figure 6.12 Plot of Test 7.3. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

Test 8 rotated the motor back and forth in a 20-step range from 60-80. Figure 6.13 shows the results.

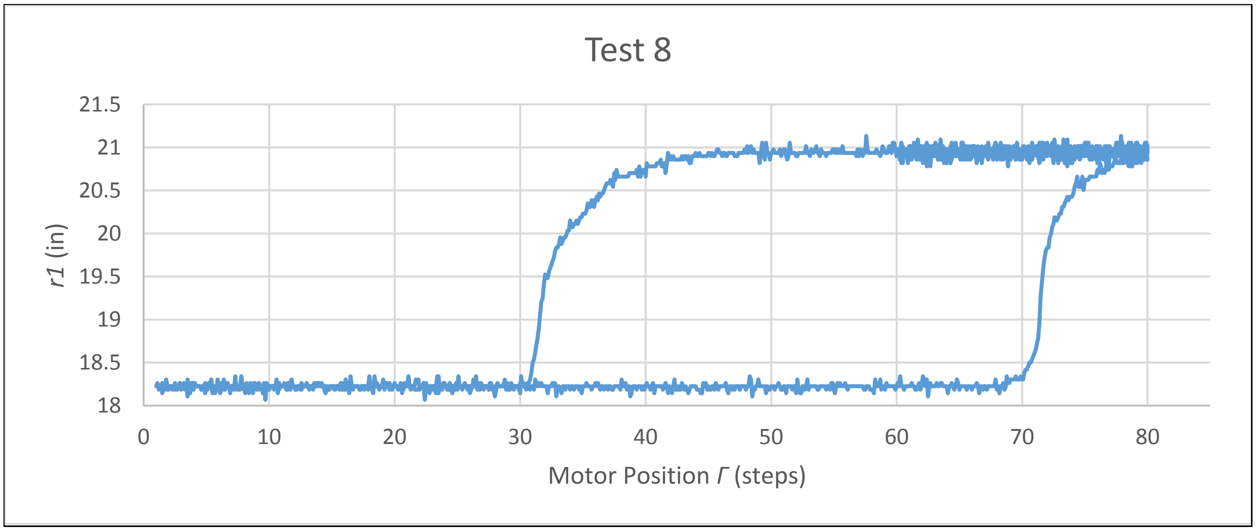


Figure 6.13 Plot of Test 8. *R1* is plotted over *Γ*. The lines overlap in the top right corner.

Test 9 rotated the motor back and forth in a 40-step range between 35 and 75 steps. The results are shown in Figure 6.14.

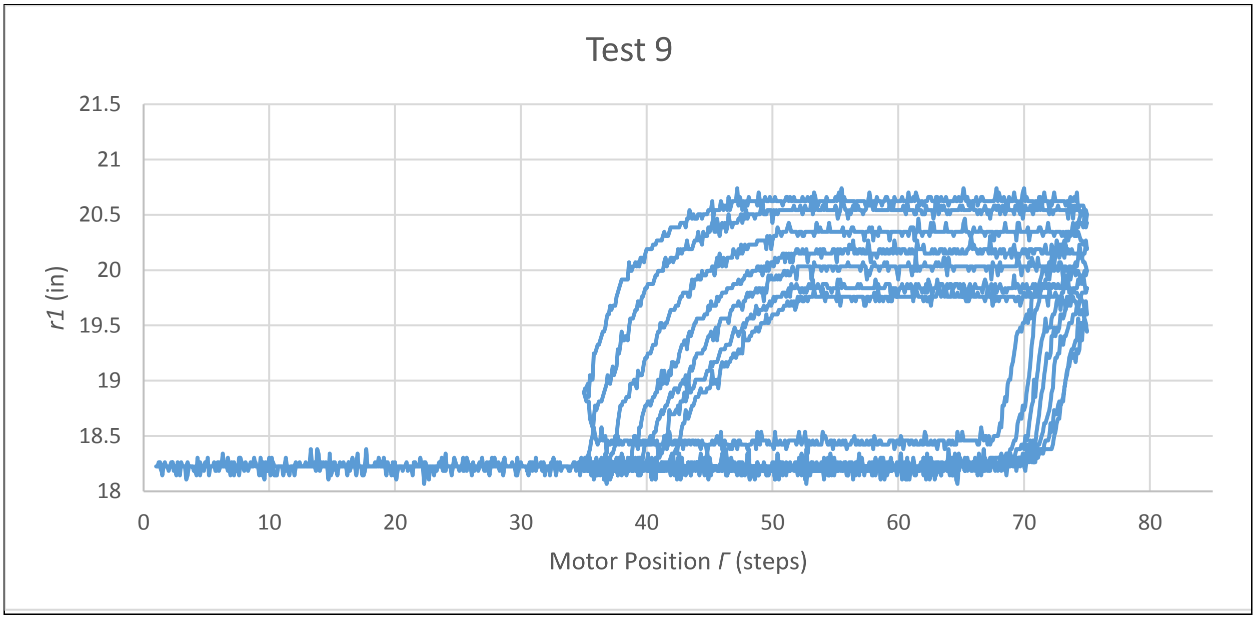


Figure 6.14 Plot of Test 9. *R1* is plotted over *Γ*.

Test 10 added 0.5 lbs of weight to the platform. Figure 6.15 shows the results.

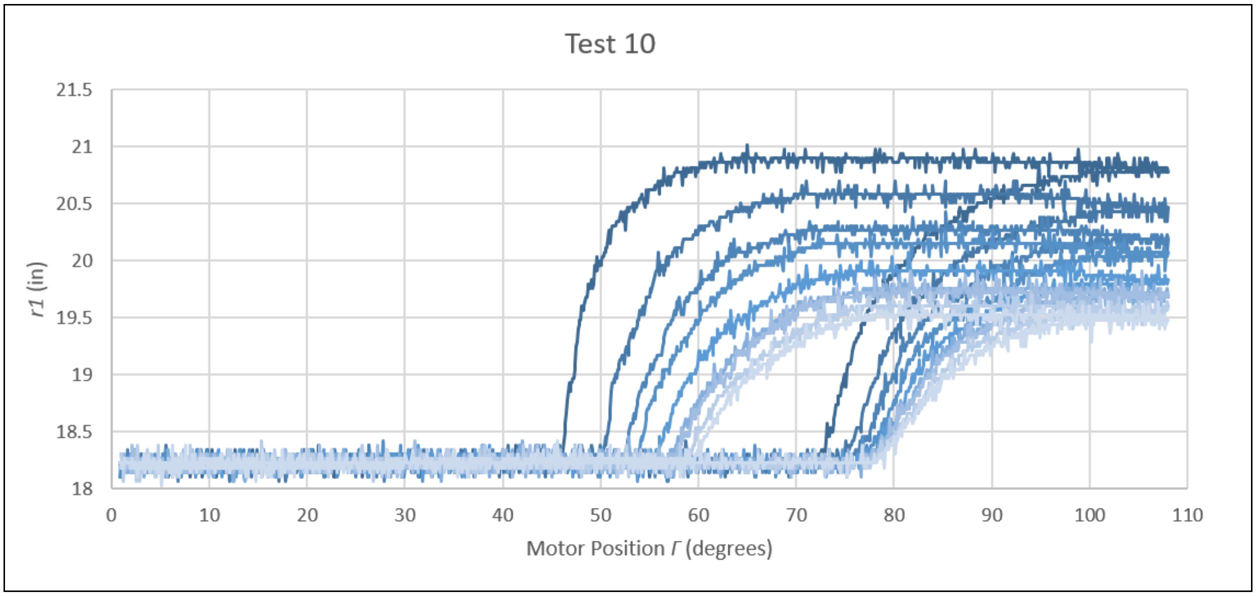


Figure 6.15 Plot of Test 10. *R1* is plotted over *Γ*. The line color gets lighter with each loop.

CHAPTER 7: DISCUSSION AND CONCLUSIONS

Now that the test data has been visualized, it will be analyzed and discussed. In the previous chapter, each test was accompanied with a plot which showed vertical position (r_1) against motor position (Γ) for a series of 9 loops. The majority of these loops had nearly identical maximum and minimum vertical positions. This range was taken and split into equal quartiles for each loop, and the corresponding motor position (steps) at each of those values was recorded. Since the tests showed separate but consistent behavior for the platform rising compared to falling, this process was applied to both directions separately. This is shown for Test 4.1 in Table 7.1. Then this data was visualized by using a dual series of boxplots. The upper row represents rising, and the lower row represents falling. This can be seen in Figure 7.1. The tables for every other test will not be shown here, but can be found in Appendix B.

In this chapter, the tests will be compared to quantify the effect of 5 different parameters: speed of motor rotation, length of the top link r_3, position of the bottom stop, weight of the top platform, and twist reversal angle.

Table 7.1 Min, max, midpoint, and quartile values with their corresponding motor positions for the initial run of Test 4.

Test 4.1									
Up									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.932	20.936	20.932	20.932	20.924	20.933	20.928	20.933	20.927
Beam Angle	85.1	90.3	93.6	95.9	99.3	101.2	101.3	103.8	105.9
Min	18.203	18.206	18.210	18.203	18.203	18.204	18.196	18.202	18.202
Beam Angle	66.3	68.1	69.2	70.2	69.4	69.5	69.5	69.6	69
Midpoint	19.568	19.571	19.571	19.567	19.564	19.569	19.562	19.568	19.565
Beam Angle	69.4	71.6	72.1	72.7	72.7	72.6	73	73.1	73.2
1/4 point	18.885	18.888	18.890	18.885	18.884	18.887	18.879	18.885	18.884
Beam Angle	68.7	71.2	71.5	72.3	72.2	72.1	72.2	72.4	72.3
3/4 point	20.250	20.253	20.252	20.249	20.244	20.251	20.245	20.250	20.246
Beam Angle	72.2	75	75.7	76.3	76.6	77.2	77.4	77.7	78.1
Down									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.946	20.944	20.943	20.943	20.942	20.946	20.941	20.940	20.945
Beam Angle	47.5	54.2	57.4	58.7	60.4	60.9	62.3	63.5	64.4
Min	18.222	18.226	18.224	18.224	18.229	18.224	18.220	18.228	18.227
Beam Angle	34.8	36	36.7	37.9	38.5	39.2	39.9	40.5	40.4
Midpoint	19.584	19.585	19.583	19.584	19.585	19.585	19.581	19.584	19.586
Beam Angle	37	38.6	39.7	40.6	41.4	41.8	42.4	42.8	43.4
1/4 point	18.903	18.905	18.903	18.904	18.907	18.904	18.900	18.906	18.906
Beam Angle	36.4	37.3	38.5	39	39.8	40.7	41.5	42.3	42.4
3/4 point	20.265	20.265	20.263	20.264	20.263	20.265	20.261	20.262	20.266
Beam Angle	38.7	40.5	41.7	43	44.3	44.8	45.7	46.5	46.8

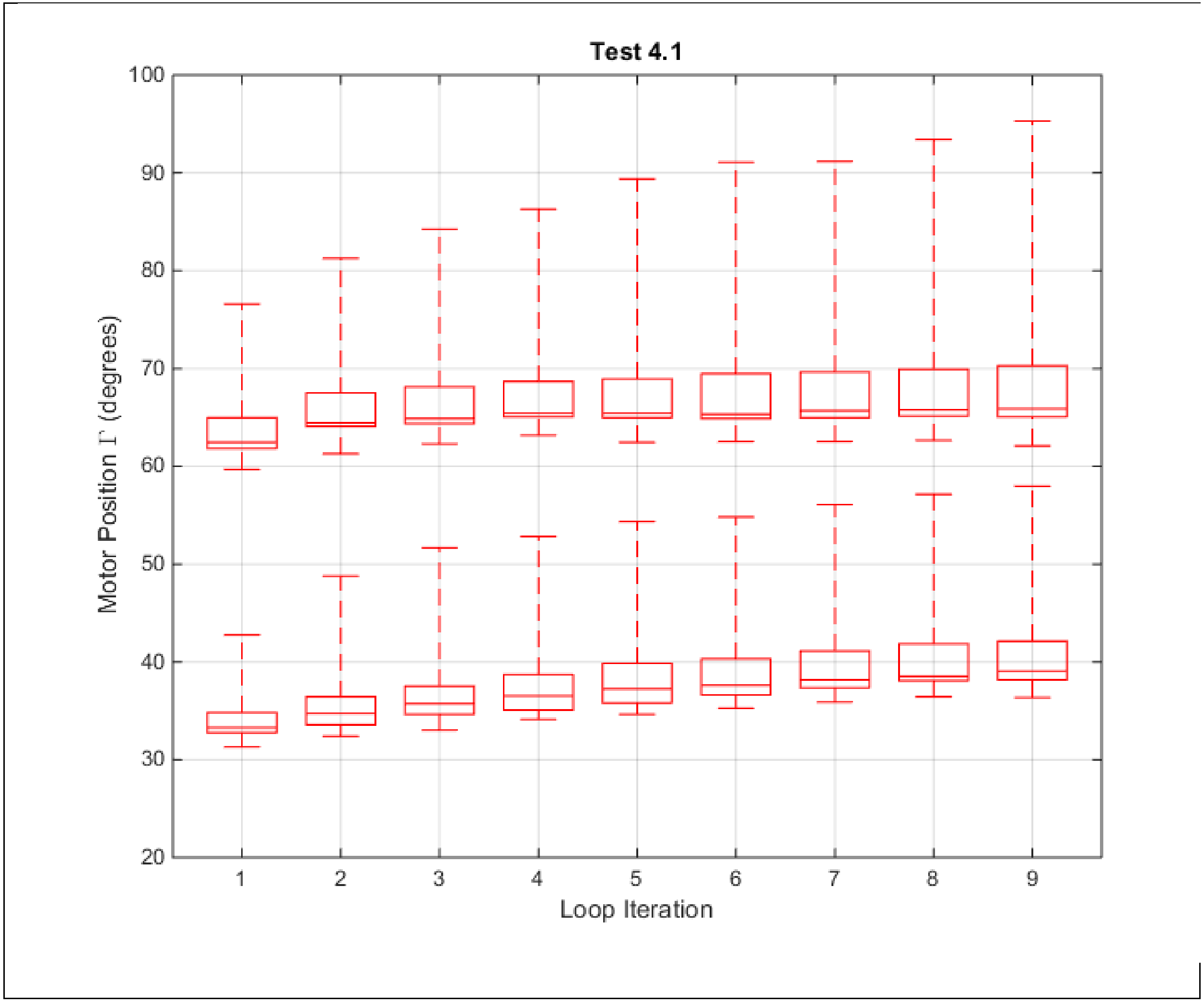


Figure 7.1 Boxplots of the initial run of Test 4. Each one represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down.

7.1 Motor Rotation Speed

Using this visualization, Test 4 was compared against Test 7. These were alike in procedure except for speed. Test 4 had the slowest speed (0.978 steps/s) while Test 7 had the fastest speed (6.803 steps/s). First, all 3 runs of Test 4 were compared, as seen in Figure 7.2. The same was done in Figure 7.3 for the 3 runs of Test 7.

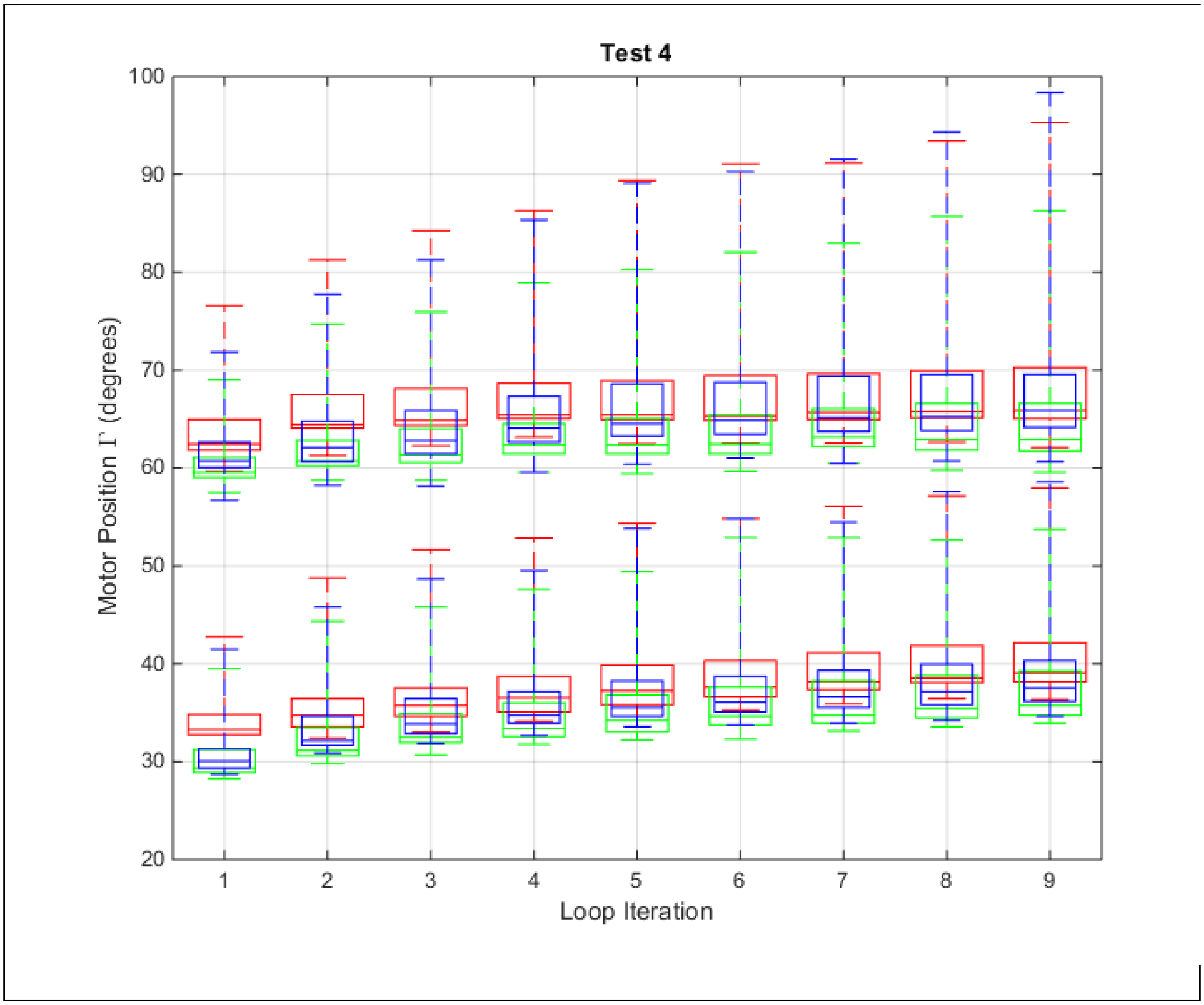


Figure 7.2 Boxplots comparing the initial run of Test 4 and its replicates. Each plot represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down. Each color represents a different run: Test 4.1 = red, Test 4.2 = green, Test 4.3 = blue.

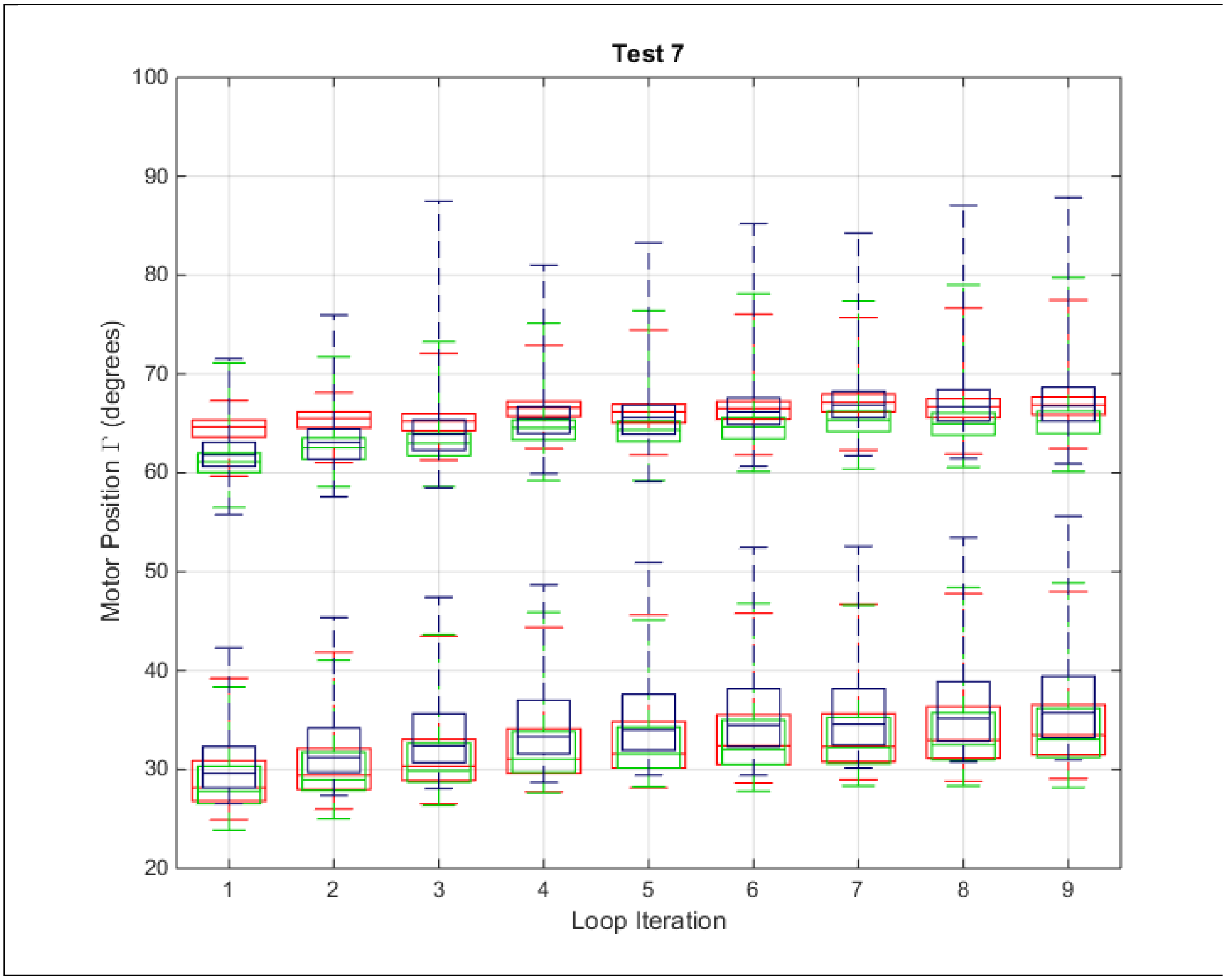


Figure 7.3 Boxplots comparing the initial run of Test 7 and its replicates. Each one represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down. Each color represents a different run: Test 7.1 = red, Test 7.2 = green, Test 7.3 = blue.

First, it is seen that the top row of boxes in Test 4 are taller than in Test 7. This means that the amount of motor steps between the first and third quartiles is greater in Test 4 when the mechanism is rising. This distance between quartiles is comparable to the slope of the line during its rise or fall in the plots shown in Chapter 6. Having the slope near vertical is not ideal for a controllable system, so having short boxes is not ideal either. Looking at the bottom row, Test 7 is the one with the taller boxes. This is interesting, because the boxes in Test 7's bottom row are also much taller than its top row. This indicates a difference in slope in Test 7 between ascending and descending. This adds complication to a controllable system and is not ideal.

Below in Figure 7.4, Tests 4 and 7 are compared on the same graph. It is a minimal version of the previous two plots, with just the midpoint values being displayed. The midpoint values give a good one-number summary of the position of the rise and fall curves. The plot shows clearly the difference in motor position between the two tests, and the difference between the rise and fall of each test.

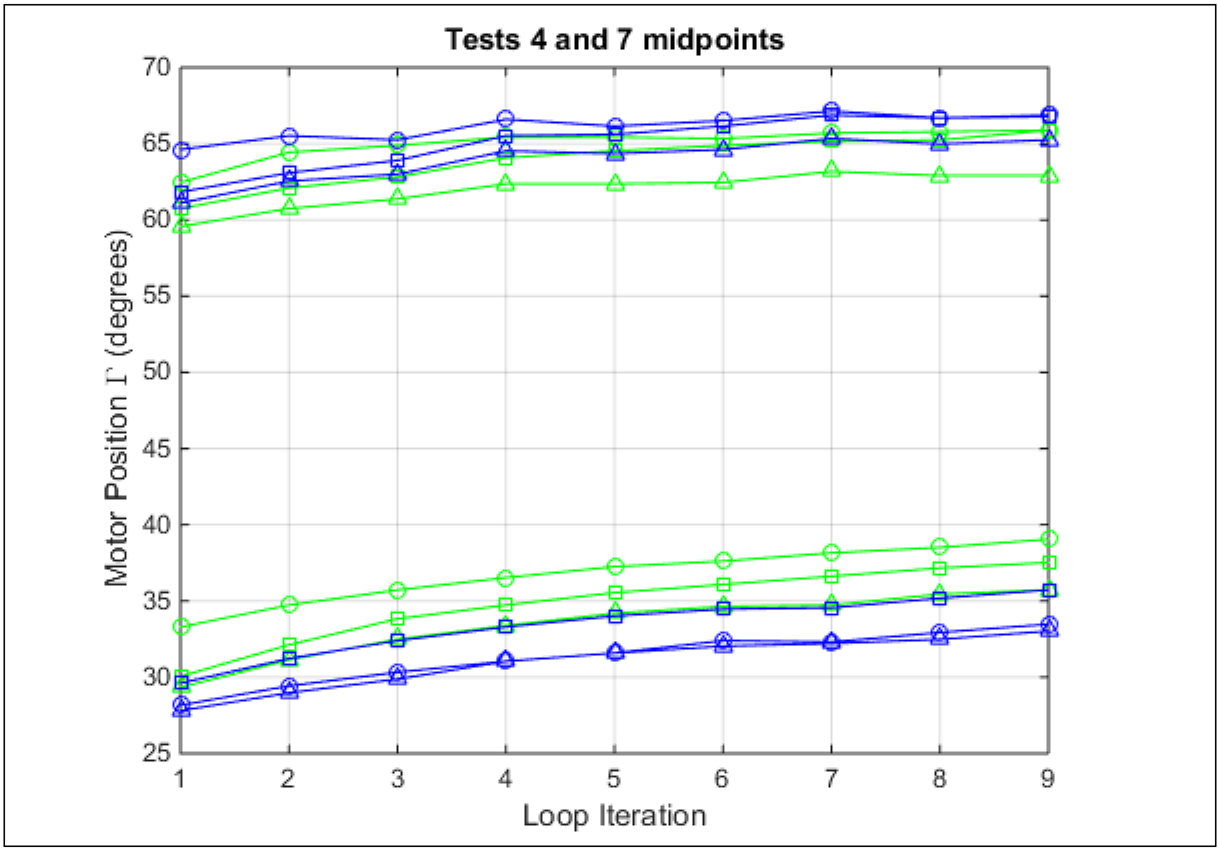


Figure 7.4 Plot of the midpoints of all runs of Tests 4 and 7 at each loop iteration. Here, green represents Test 4 and blue is Test 7. Circle markers show the initial run, triangles show replicate 1, and square markers show replicate 2.

The plot seems to show that Test 7 is lagging behind Test 4. It takes 2-4 degrees more for Test 7 to reach the same point as Test 4, on both rise and fall. This makes the effects of hysteresis more pronounced for Test 7. The amount of hysteresis can be roughly estimated by the size of the gap between the top midpoint lines and bottom midpoint lines of the same color and style. But other factors should be taken into account before making any solid conclusions. The difference between the lines of Test 4 and those of Test 7 is comparable to differences within each test. Specifically, the difference between the first and last loop iteration of each run and the difference between runs.

Speed of motor rotation cannot be determined to significantly affect the motor position of the rise and fall curves. However, higher speeds may increase the difference in slope between rise and fall curves. This could lead to control problems in the future.

7.2 Position of Bottom Stop

Next, I look at how the position of the bottom stop affects the behavior of the system. Tests 1, 2, and 4 are compared to isolate this parameter. The difference between these three tests is only the position of the bottom stop. This can be seen in Figure 7.5, where one run is chosen from each test, and their respective plots from Chapter 6 are superimposed on each other. The minimum r_1 value for each color show where that test's bottom stop is.

Note that each color makes a similar shape, but that shape's dimensions vary between colors as the bottom stop moves upward. Not only does the shape get shorter, it gets skinnier and also shifts to the left. The height of the shape describes the vertical range of the mechanism. A large range is desirable, so shorter is not necessarily better. However, the shifting to the left is a good thing and may outweigh that. The shape's horizontal position indicates the active range of the motor rotation. It is desirable to have this motor range not extend past 90 degrees. This is because the beam should not have to rotate past 90° to move the platform to its maximum height. The beam also fatigues more quickly the lower the bottom stop is. This is shown in the figure in two ways. First, the lines for each loop are more spaced out the lower the bottom stop is. Also, in the tests that use the lower two positions (red and blue on the plot), the max height decreases with each loop. It makes sense that the beam fatigues more rapidly with a lower bottom stop because it is being bent further and therefore experiences greater stress.

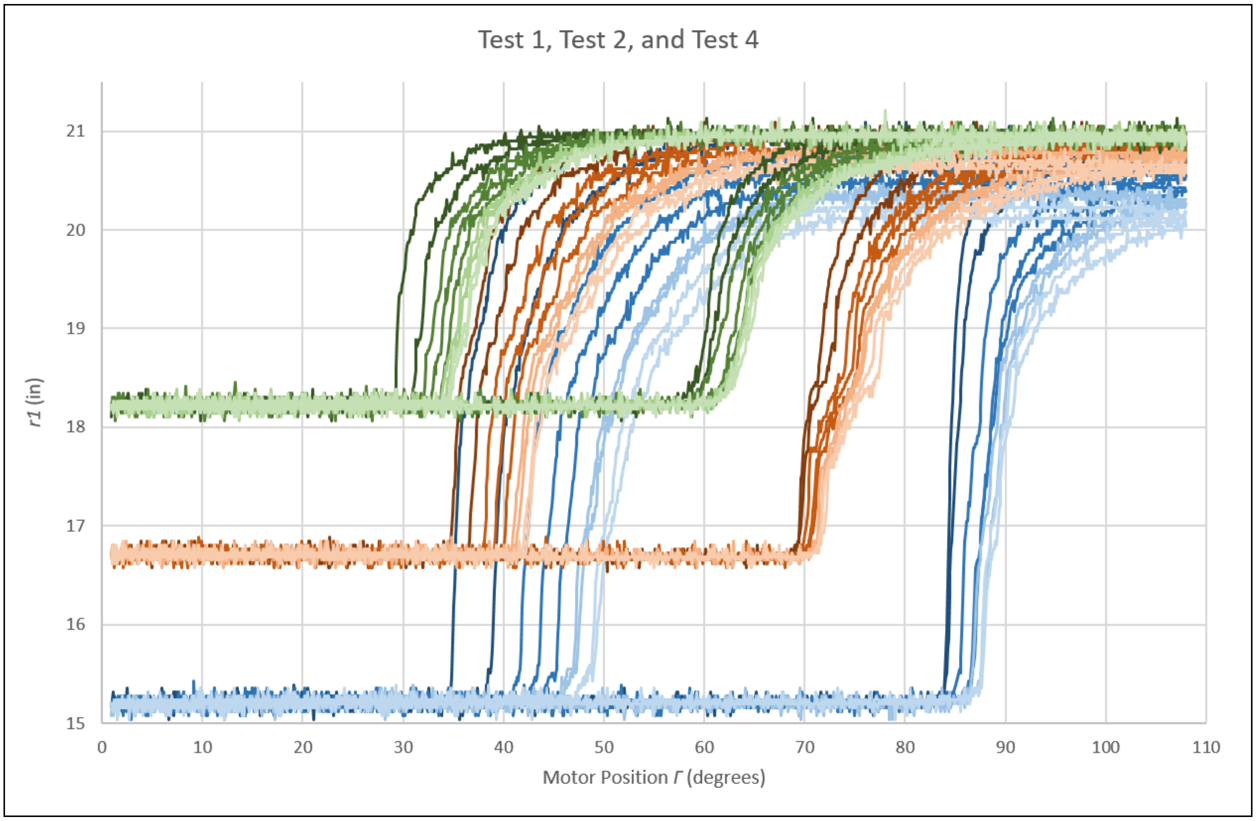


Figure 7.5 Superimposed plots of Test 1.2 (blue), Test 2 (red), and Test 4.3 (green).

Now, the slopes of the lines during each test will be compared. The boxplots shown in Figures 7.6, 7.7, and 7.2 correspond to Tests 1, 2, and 4 respectively. Examining these plots shows a weak correlation between box height and bottom stop position. Interestingly, the middle position (Test 2) has the tallest boxes. However, the plot of Test 2 shown in Figure 7.5 has a unique bend in the lines about halfway up, and this may be affecting the boxplots. The midpoint lines in the top row of boxes are halfway up or higher, which is unusual because they are normally near the bottom of the box next to the 25% line. Also, Test 2 was only performed once so it may not be a good idea to make conclusions on this data. Something that the boxplots do not account for is the vertical range, which affects the slope. Slope = y/x, and the box height only provides the x. The vertical range for Test 4 is about half that of Test 1, so if we doubled the height of Test 4’s boxes with respect to Test 1, that would be a more accurate estimation of the slope. Since Test 4 and Test 1’s boxes are similar in height as displayed, this doubling would easily show that the slopes of Test 4 is gentler and more desirable than those of Test 1.

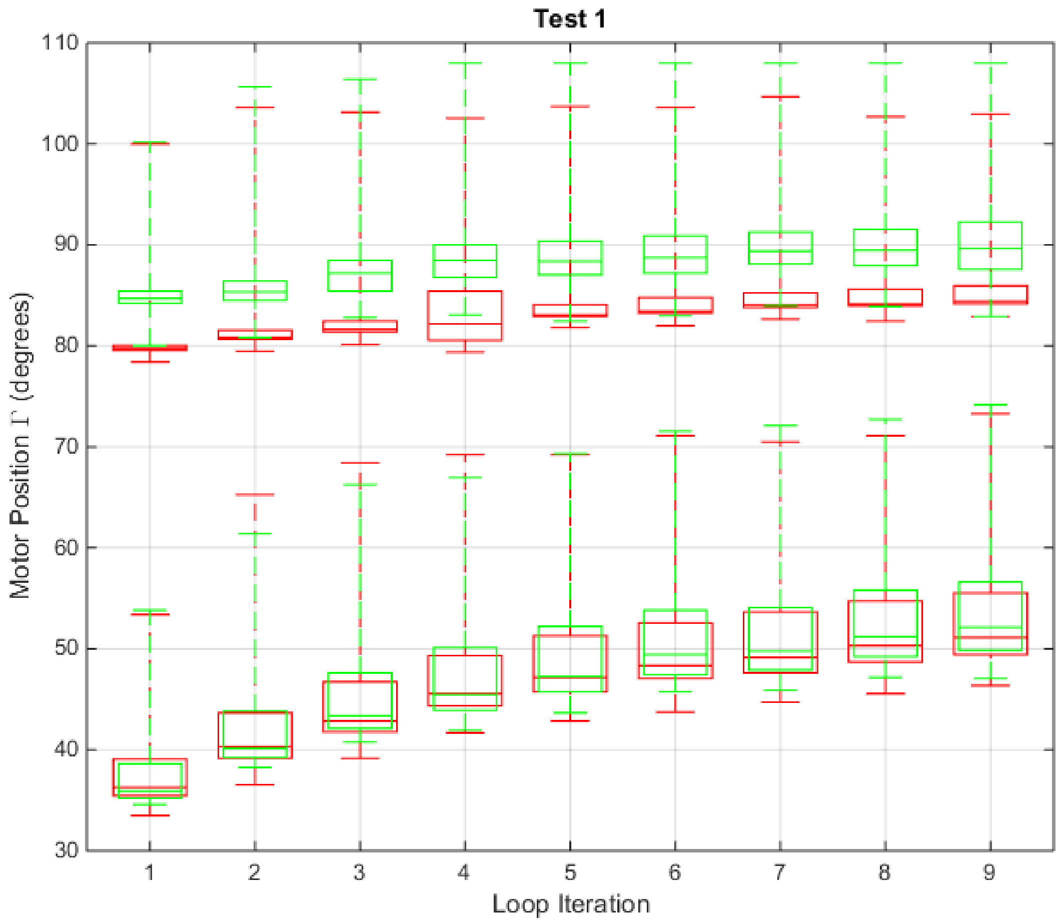


Figure 7.6 Boxplots comparing all runs of Test 1. Each boxplot represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down. Test 1.1 is red, and Test 1.2 is green.

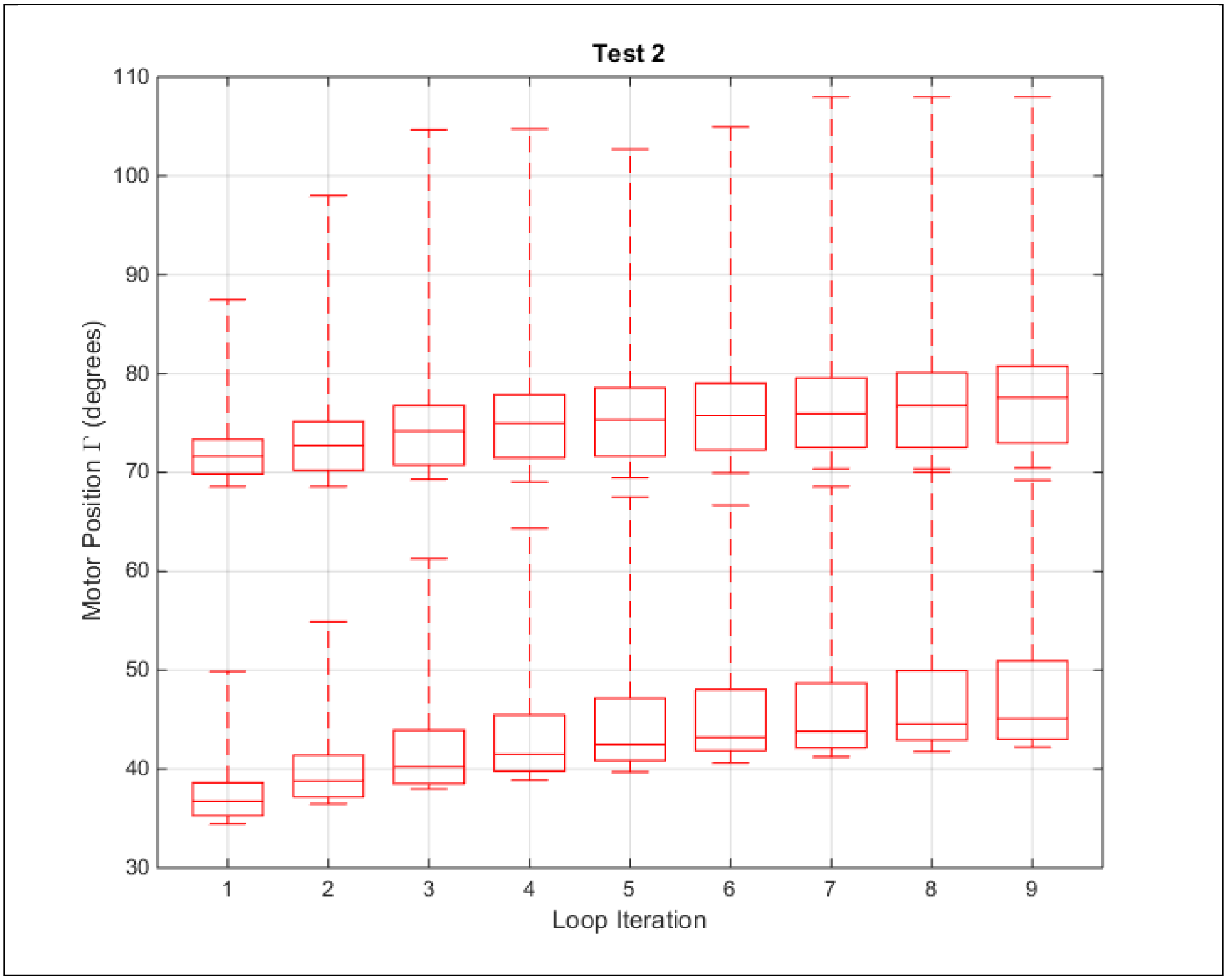


Figure 7.7 Boxplots of Test 2. Each one represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down.

Finally, the positions of the rise/fall curves are compared using a different plot. The midpoints from all three tests were plotted and can be seen in Figure 7.8. The vertical shifting between colors reflects the horizontal shifting in Figure 7.5. The other thing to notice is that the distance between rise/fall lines within colors slightly decreases as the bottom stop moves up. This suggests that a higher bottom stop position reduces the effect of hysteresis.

Because of these reasons, the highest bottom stop position (as used in Test 4) is determined to be the most desirable. The range of motion is the lowest of the three tests but other factors make up for it. The full range of motion is possible without the motor exceeding 100 steps, the beam fatigues the least quickly, the slopes of the rise/fall curves are the gentlest, and the effect of hysteresis is minimized.

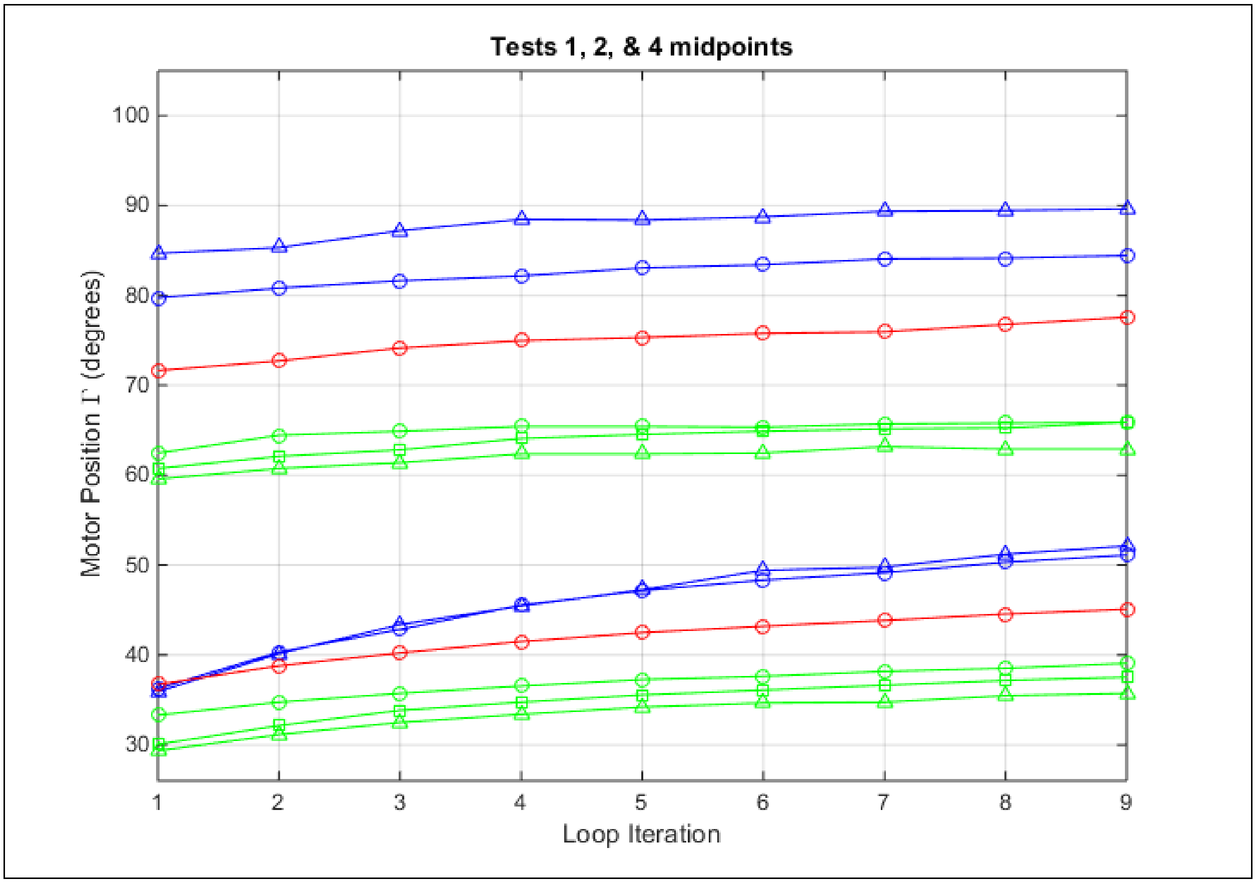


Figure 7.8 Plot of the midpoints of all runs of Tests 1, 2, and 4 at each loop iteration. Here, Test 1 is blue, Test 2 is red, and Test 4 is green. Circle markers show the initial run, triangles show replicate 1, and square markers show replicate 2.

7.3 Length of Top Link

Now I will explore how the length of the top link, r_3, affects the system. This variable is isolated by comparing Test 2 and Test 3. In Test 2 r_3 is 11.55” long, and in Test 3 it gets reduced by 0.77” to be 10.78” long. This means the ratio R = 1.2 in Test 2, while R = 1.12 in Test 3. The plots of Tests 2 and 3 from Figures 6.3 and 6.4 are superimposed below in Figure 7.9. Shortening r_3 decreases the vertical range of the system, as seen in the figure.

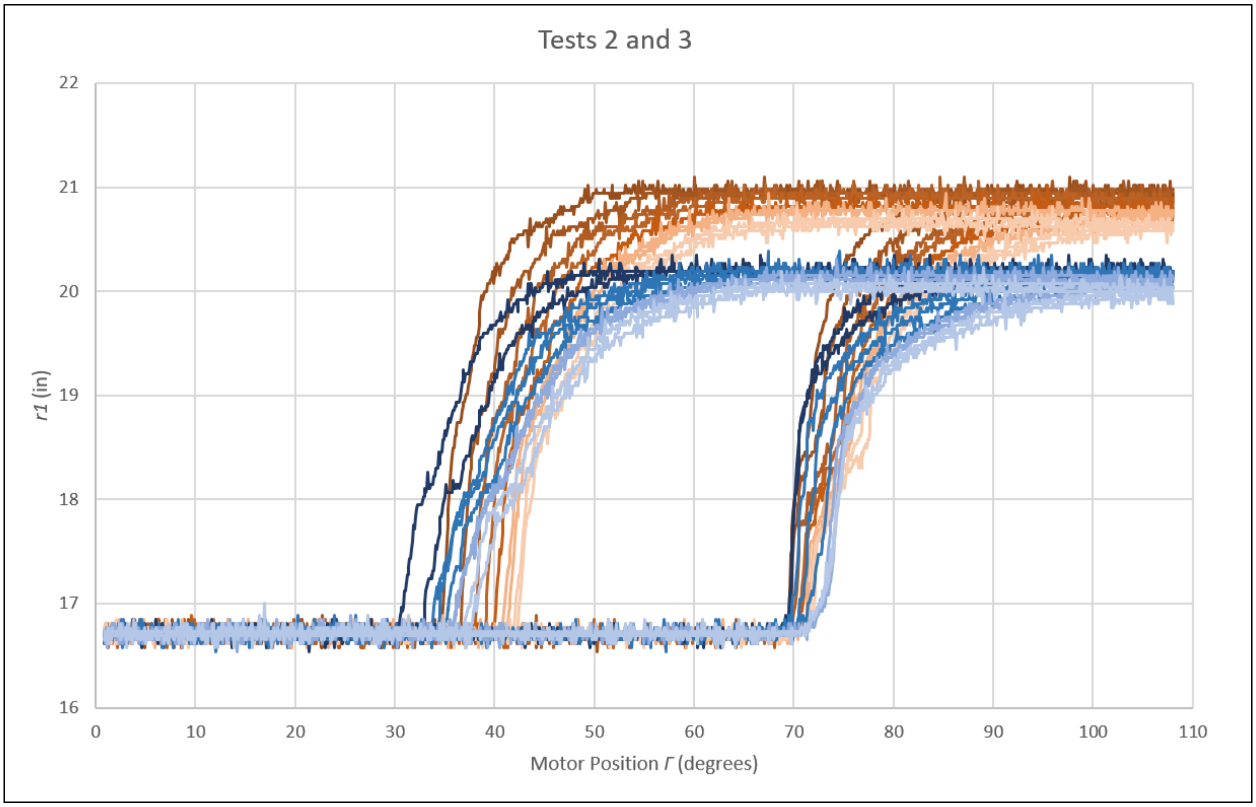


Figure 7.9 Superimposed plots of Test 2 (red) and Test 3 (blue).

The slopes of the fall curves seem steeper in Test 2, while the rise curves have similar slopes in both tests. The difference between rise slopes and fall slopes is greater in Test 3, and is not ideal. This can be seen clearly in the boxplots of Test 3, shown below in Figure 7.10. The boxes in the top row are much shorter than the boxes in the bottom row.

Because of the decrease in range of motion and the increase in slope difference between rise and fall, Test 3 gave a less controllable system than Test 2. For this reason, r_3 = 11.55” and R = 1.2 are determined to be superior.

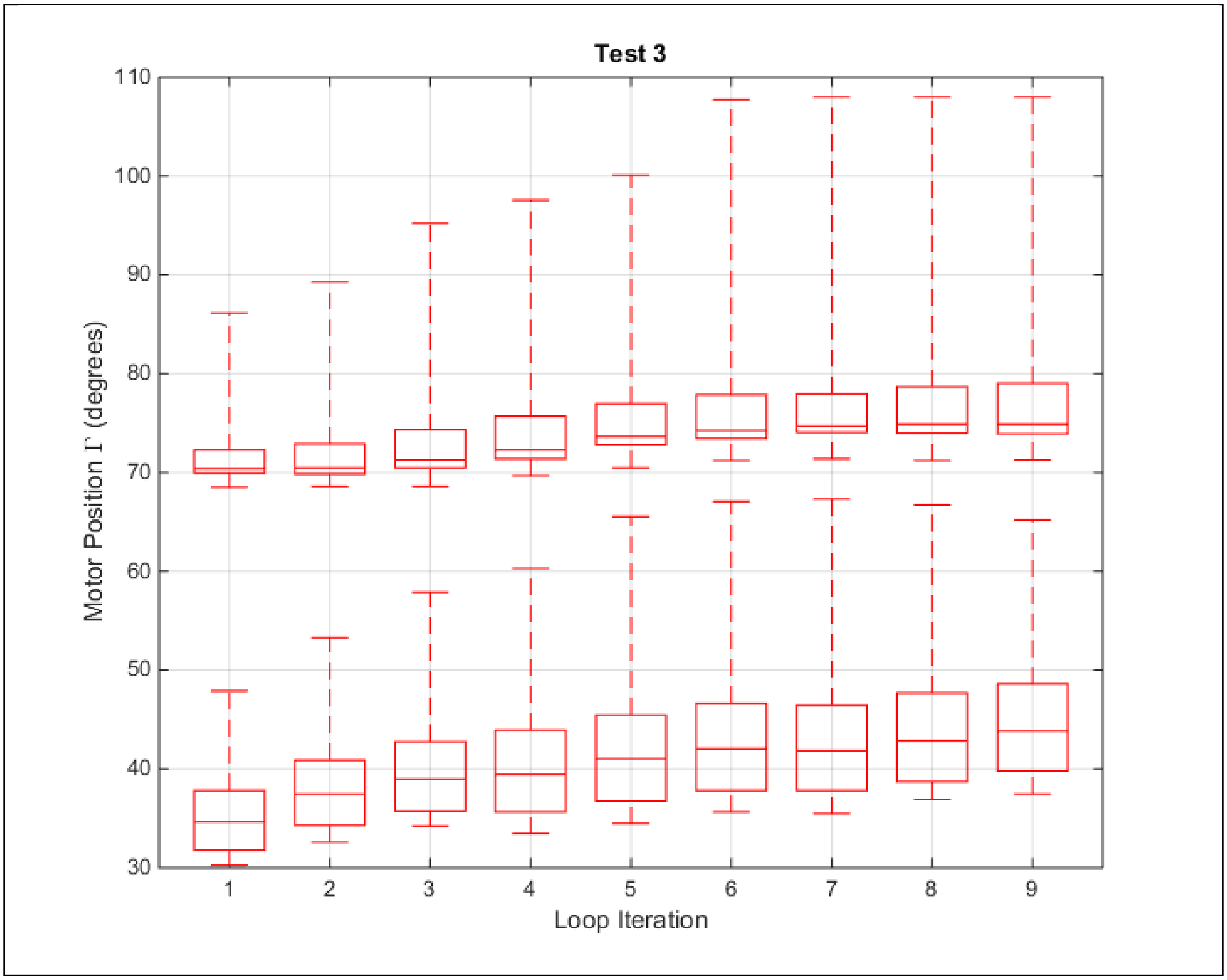


Figure 7.10 Boxplots of Test 3. Each one represents the motor position at different values of r_1 (min, 25%, midpoint, 75%, max). The top row describes the system moving up at each iteration, and the bottom row describes it moving down.

7.4 Weight Added to Platform

Now I will analyze the effect of added weight on the system. Test 4 and Test 10 will be compared to isolate the weight parameter. The only difference between these tests is that Test 10 has 0.5 lbs. added to the top platform, while Test 4 has no additional weight added. The plots of Test 4.3 and Test 10 shown in Figures 6.7 and 6.17 are superimposed and shown below in Figure 7.11. The differences between the two are very prominent.

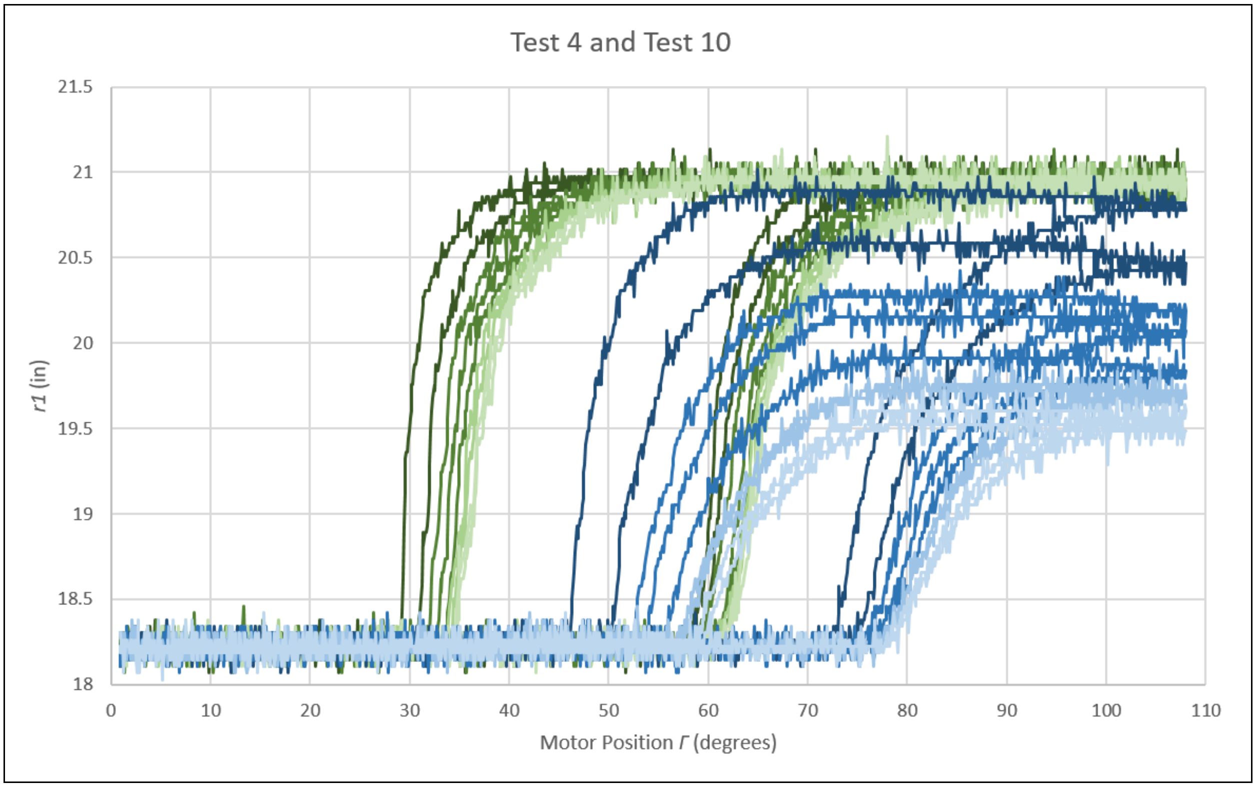


Figure 7.11 Superimposed plots of Test 4.3 (green) and Test 10 (blue).

First, the rise/fall curves are shifted 15-25 degrees to the right. This makes the active motor range about 45-100 degrees for +0.5 lbs. It is heavily skewed towards the top of the preferred motor range, and this is not ideal. This is compounded by the amount of meaningful motion past 90 degrees shown by Test 10. Ideally, r_1 would reach the top stop (maximum) before 90 degrees, and Test 10 fails that objective.

The curves of Test 10 have more change between loop iterations compared to Test 4. This is especially prominent on the falling curves. They are spaced out more, and the maximum height decreases with each loop. This is a strong indication of fatigue, which is unwanted.

Adding weight to the system shifts the active motor range past what is acceptable and increases the rate of fatigue. For these reasons, increased weight is not desirable for the system.

7.5 Twist Reversal Angle

The last parameter to analyze is the twist reversal angle. This is the point at which the motor switches directions. In every test discussed so far, the motor has switched directions at 120 steps and 1 step. To isolate this parameter, Tests 4, 8, and 9 will be compared. The plots for Test 4 are shown in Figures 6.5, 6.6, and 6.7. The plots for Tests 8 and 9 are shown in Figures 6.13 and 6.14, respectively. Test 4 changes direction at 1 and 120 steps, Test 8 changes direction at 60 and 80 steps, and Test 9 changes direction at 35 and 75 steps. While Test 4 executes nine loops, Tests 8 and 9 execute only six.

The plot of Test 8 may look like one big, single loop, but it actually overlaps itself many times in the top right corner. A more descriptive visualization is provided below in Figure 7.12, where r_1 is plotted against time instead of motor position. The two plots together paint the complete picture. The platform starts at the minimum position, and then hits maximum by the time the motor reaches 80 steps. The motor then starts turning back and forth in a 20 step range, but the platform doesn't move until it leaves that range and starts heading back down to step 1.

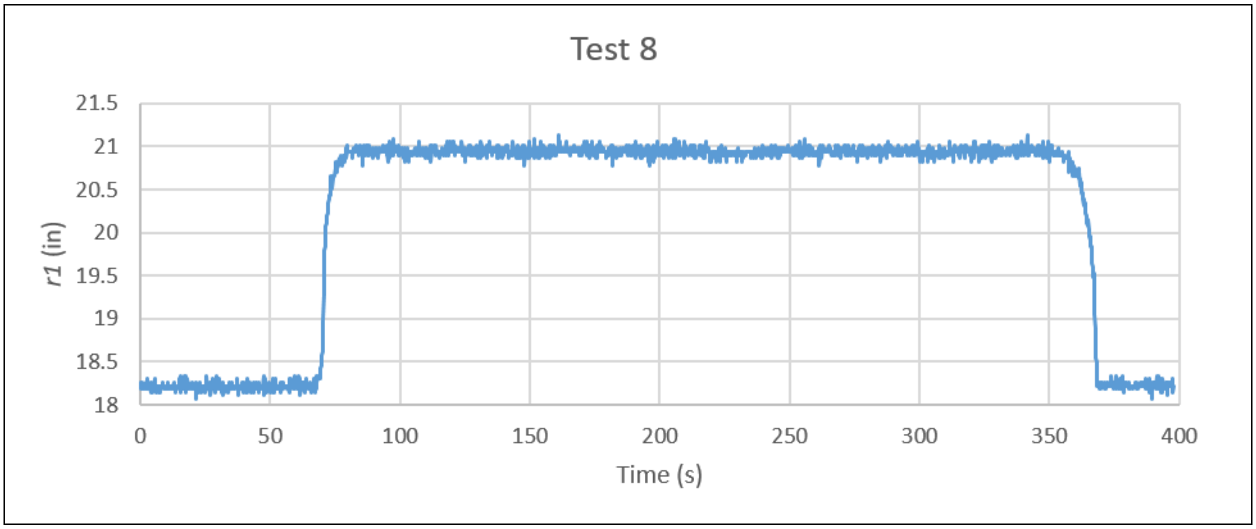


Figure 7.12 Alternate plot of Test 8. *R1* is plotted over time (s).

Test 9 shows a more interesting plot. The platform starts to rise on each loop but doesn't make it to the top. When the motor reverses direction, the platform does not follow suit. Rather, it stays still for about 20 steps before finally starting to move back down. This is shown in Figure 6.14, but the overlap makes it hard to differentiate between loops. Below, in Figure 7.13, Test 9 is plotted against time. Here, the individual loops are clearly seen.

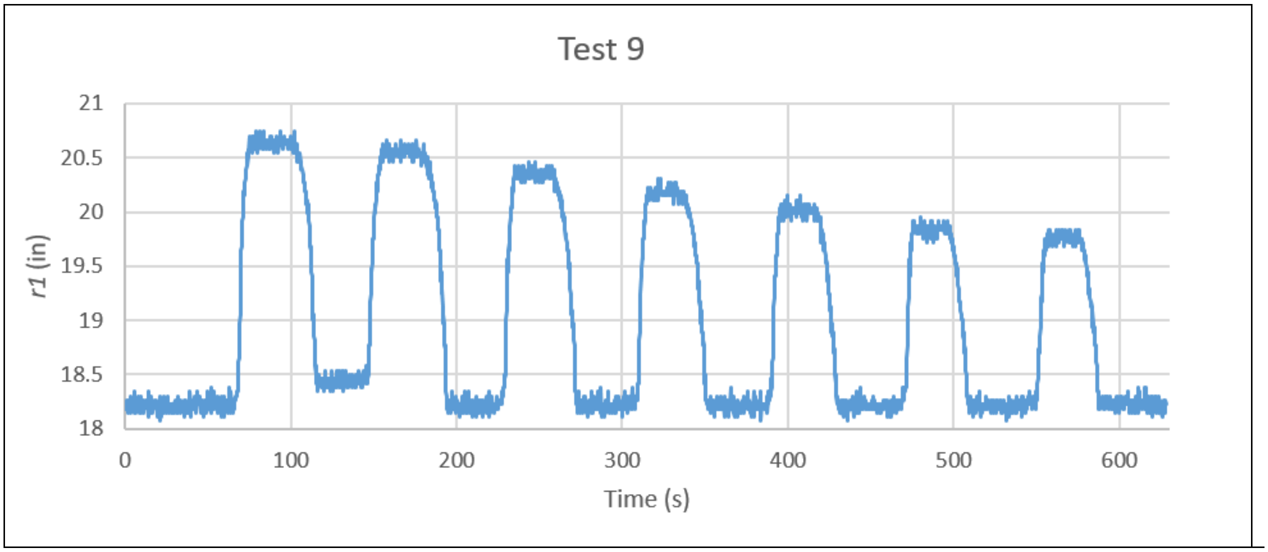


Figure 7.13 Alternate plot of Test 9. *R1* is plotted over time (s).

The system is showing an interesting behavior. If the direction of the motor is reversed, the platform does not change position until some number of steps have been taken in the other direction. This happens regardless of where the platform is when the motor changes direction, or whether it was going up or down. This behavior indicates hysteresis in the system. At any moment, the system has a sensitive direction, and a resilient direction. This makes the system both hard to model and hard to control. However, improving this has proven to be difficult. This behavior is present in every test so far. Hysteresis results from energy being stored in the system somewhere. This stored energy could be from torsion in the PP beam, friction on the slider, or something else that is undetermined so far.

CHAPTER 8: FUTURE WORK

The work presented in this thesis is only a part of the development of this mechanism. Going forward, there will be changes made and new phases of testing. Vibration isolation is the ultimate goal, and to achieve that there will need to be dynamic testing done. The mechanism was designed with this in mind, and the static tests have helped push it closer to the dynamic testing. In order to be successful, however, I believe some additional parameters need to be explored. First, the slider friction will be reduced either by adding lubrication or by switching out the slider. Second, the joint at the bearing will be kept from sticking either by redesign or adding a different lubrication. Hopefully, changing these two parameters will reduce the hysteresis of the system. Finally, the beam material will be switched from polypropylene to something with less creep, such as steel. This will extend the life and improve the reliability of the beam, and will hopefully reduce or eliminate the twisting problem. Polypropylene has fairly prominent hysteretic behavior (Zrida et al. 2009), and solely switching beam materials may minimize hysteresis to a manageable level.

For dynamic testing, I have considered two different vibration isolation approaches. The first will be a closed loop active control. Ideally this will emulate a "sky-hook", where the platform stays still relative to ground. The second approach involves tuning the mechanism such that the oscillatory motion is cancelled out by the deflection of the beam. The payload of the mechanism behaves as a tuned mass damper and reduces harmonic vibrations. This assumes that the oscillations experienced by the mechanism are both sinusoidal and uniform.

Existing versions of vibration isolation technology employ semi-active suspension using the sky-hook concept (Savaresi & Spelta 2007). This means that the payload maintains a stable position even when the mechanism itself is oscillating, as though it was supported by an imaginary hook in the sky. This is achieved through control of the damping and stiffness.

There has also been success in vibration isolation by using a tuned mass damper. This component stabilizes motion caused by harmonic vibration. It is usually calibrated for a specific frequency range, making it a passive control approach. However, through the use of variable stiffness and damping, Solase and others have made a mechanism that uses this concept and can adapt to different frequencies (Solase et. al. 2013).

CHAPTER 9: THESIS CONTRIBUTIONS

Several things are now known about the system that were not known before. There seems to be a significant amount of hysteresis, which was not predicted. The cause is not certain, but it is certainly a property of the system. The polypropylene beams fatigue quickly, and another material may handle the cyclic loading better. The behavior of the system can be improved for controllability by raising the bottom stop, setting the top link length such that $R = 1.2$, and reducing the weight of the top platform. This thesis has helped to understand the behavior of the system and pushed it towards its final goal, vibration isolation.

WORKS CITED

Araki, Yoshikazu, et al. “Vertical vibration isolator having piecewise-Constant restoring force.” *Earthquake Engineering & Structural Dynamics*, vol. 38, 26 Mar. 2009, pp. 1505–1523. *Wiley InterScience*, doi:10.1002/eqe.915.

Chimento, Jairo, et al. “A 3-D Pseudo-Rigid Body Model for Rectangular Cantilever Beams With an Arbitrary Force End-Load.” *Volume 5A: 38th Mechanisms and Robotics Conference*, 2014, doi:10.1115/detc2014-34292.

Gao, J., et al. “Modelling and open-Loop performance of a novel active suspension for a road vehicle.” *UKACC International Conference on CONTROL '96*, vol. 427, 2 Sept. 1996, pp. 1160–1165., doi:10.1049/cp:19960717.

Hasara, Steven and Craig Lusk. “A Load-Adjustable Constant-Force Mechanism.” *ASME International Design Engineering Technical Conferences*, 2018.

Howell, Larry L. *Compliant mechanisms*. John Wiley & Sons, 2001.

Le, Thanh Danh, and Kyoung Kwan Ahn. “Experimental investigation of a vibration isolation system using negative stiffness structure.” *International Journal of Mechanical Sciences*, vol. 70, May 2013, pp. 99–112. *ScienceDirect*, Elsevier, doi:10.1016/j.ijmecsci.2013.02.009.

Savaresi, Sergio M., and Cristiano Spelta. “Mixed Sky-Hook and ADD: Approaching the Filtering Limits of a Semi-Active Suspension.” *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 4, July 2007, pp. 382–392., doi:10.1115/1.2745846.

Shahadat, Mhia Md. Zaglul, et al. “Active vibration isolation using negative stiffness and displacement cancellation controls: Comparison based on vibration isolation performance.” *Control Engineering Practice*, vol. 37, Apr. 2015, pp. 55–66. *ScienceDirect*, Elsevier, doi:10.1016/j.conengprac.2014.12.004.

Solase, R. B., et al. “Design of Variable Stiffness and Variable Damping Vibration Absorber.” *International Journal for Scientific Research & Development*, vol. 1, no. 2, 2013, pp. 178–182.

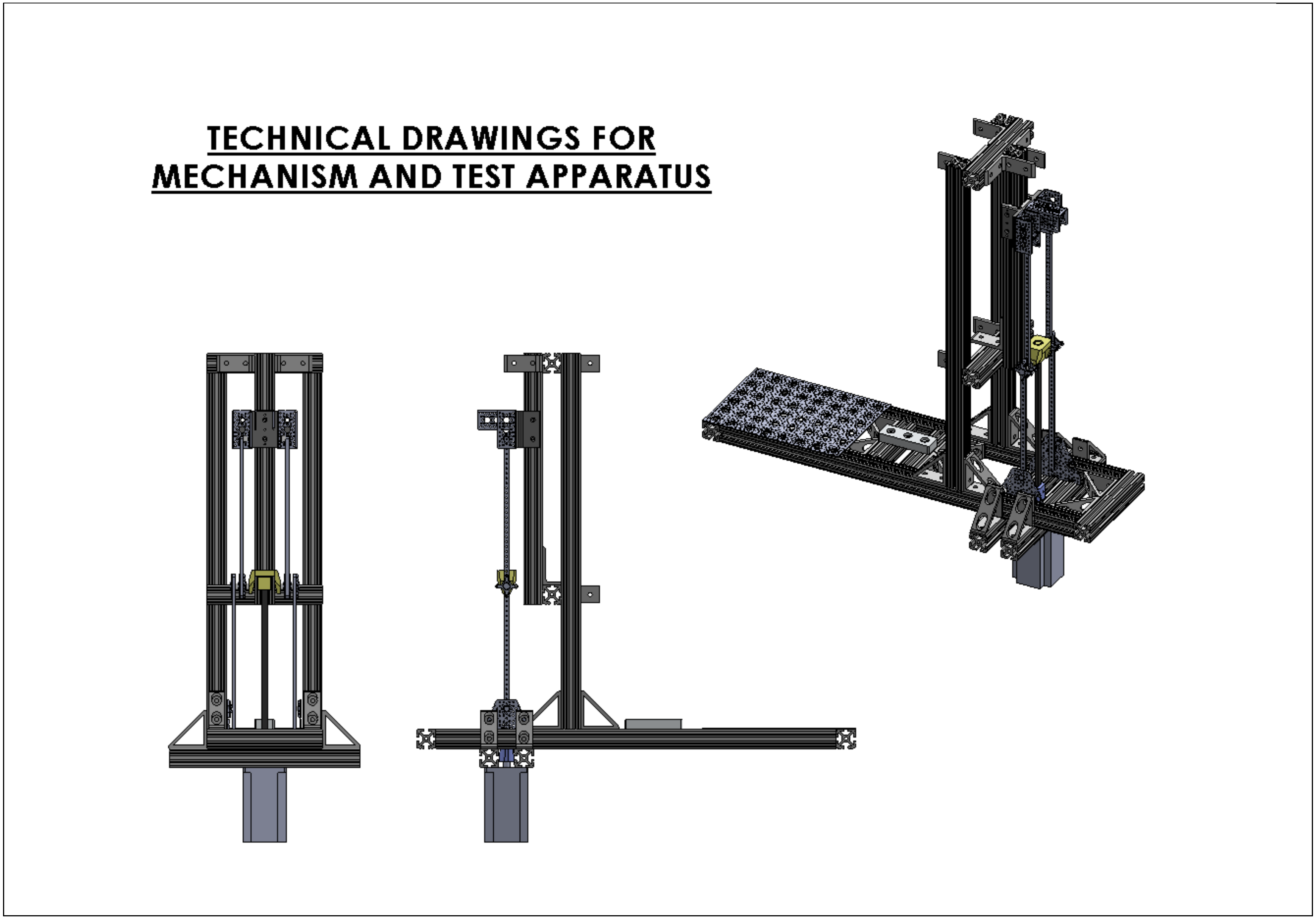
Spelta, Cristiano, et al. “Performance analysis of semi-Active suspensions with control of variable damping and stiffness.” *Vehicle System Dynamics*, vol. 49, no. 1-2, 29 July 2010, pp. 237–256. *Taylor & Francis*, doi:10.1080/00423110903410526.

Yang, J., et al. “Dynamics and power flow behaviour of a nonlinear vibration isolation system with a negative stiffness mechanism.” *Journal of Sound and Vibration*, vol. 332, no. 1, 7 Jan. 2013, pp. 167–183. *ScienceDirect*, Elsevier, doi:10.1016/j.jsv.2012.08.010.

Zrida, M., et al. “Experimental and numerical study of polypropylene behavior using an hyper-Visco-Hysteresis constitutive law.” *Computational Materials Science*, vol. 45, no. 2, 2009, pp. 516–527. *ScienceDirect*, Elsevier, doi:10.1016/j.commatsci.2008.11.017.

APPENDIX A: CAD DRAWINGS

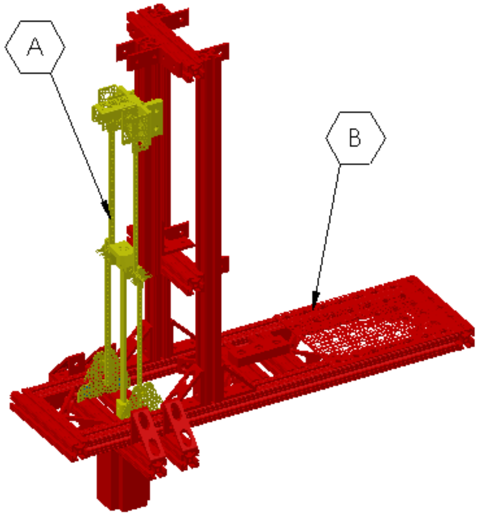
This Appendix contains all technical drawings for the components of the mechanism and test apparatus.



TECHNICAL DRAWINGS FOR
MECHANISM AND TEST APPARATUS

Figure A.1 The front page of the technical drawings.

PART LIST			
PART ID	PART NAME	DESCRIPTION	QTY.
AA	Link	Actobotics 33-hole Aluminum Beam	4
AB	Pillow Block	Actobotics Face Thru-Hole Pillow Block	10
AC	Compliant Beam	Laser-cut polypropylene beam	1
AD	Beam-Motor Coupling	3D-printed coupling between beam and stepper motor	1
AE	Beam Top Fixture	3D-printed fixture attached to top of beam and bottom of top links	1
AF	Slider	TSLOTS: Deluxe Double Flange Single Wide Linear Bearing - Short, #657097	1
AG	Channel	Actobotics Std. 3" Aluminum Channel	3
AH	Mount Plate	Cut from Actobotics 9"x12" Aluminum Pattern Plate, connects Mechanism to Test Apparatus	2
AJ	Washer	Std. #6 flat washer	38
BA	Stepper Motor	Nema 34 CNC High Torque Stepper Motor 13Nm	1
BB	Extrusion (80cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	2
BC	Extrusion (75cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	2
BD	Extrusion (50cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	1
BE	Extrusion (38cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	2
BF	Extrusion (23cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	4
BG	Extrusion (15cm)	TSLOTS: TS15-15 GR, #650080, Aluminum	2
BH	Corner Bracket	TSLOTS: 4 Hole Vertical Inside Corner Bracket, #653135	7
BJ	Corner Gusset	TSLOTS: 4 Hole Vertical Inside Corner Gusset, #653169	16
BK	Counterweight Plate	Actobotics 9"x12" Aluminum Pattern Plate	1
BL	Mount Block	TSLOTS: 3 Hole Blank Base Plate, #655660	1



A MECHANISM

B TEST APPARATUS

Figure A.2 The parts list for the technical drawings.

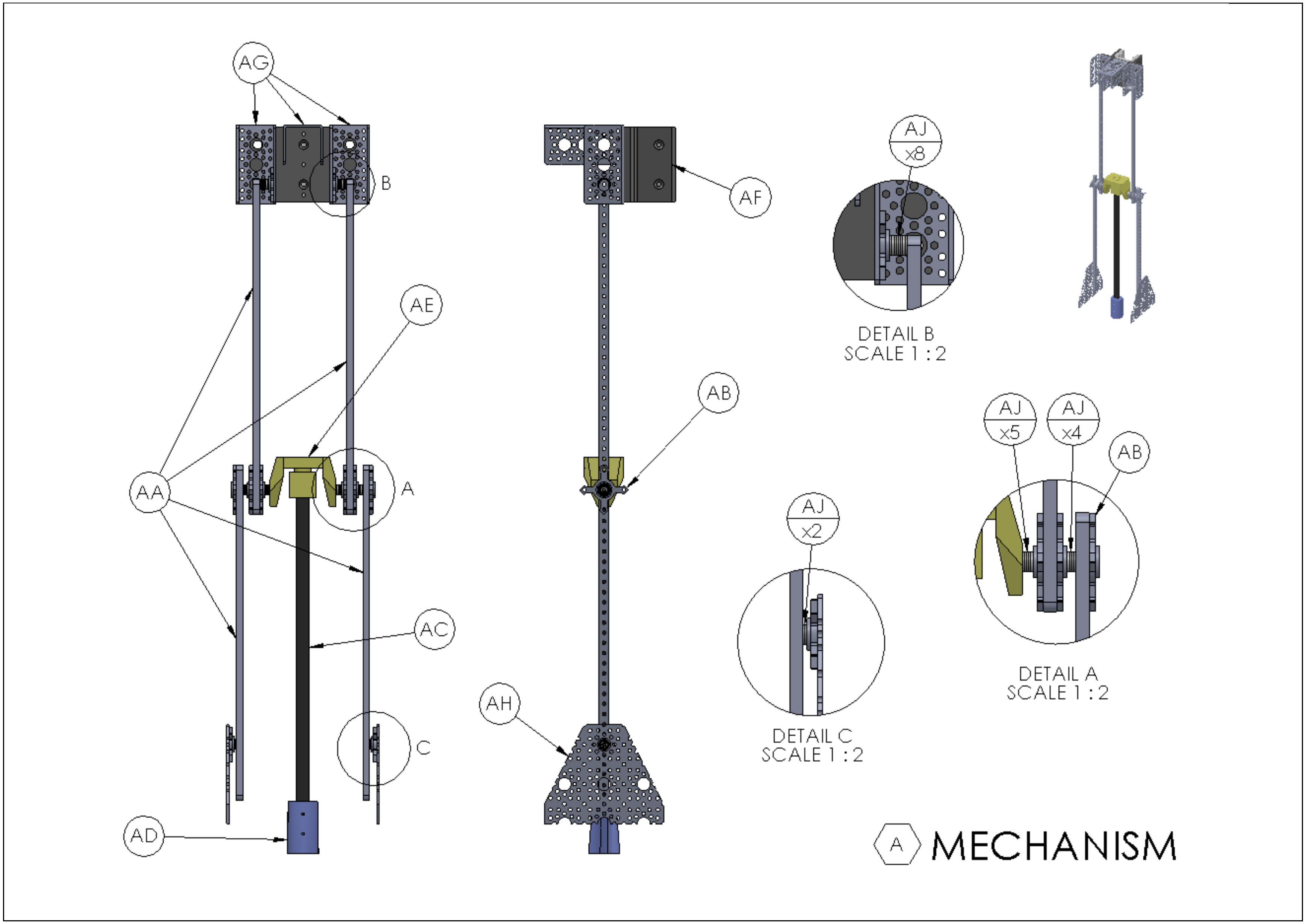


Figure A.3 Views of just the mechanism with callouts labeling the different parts.

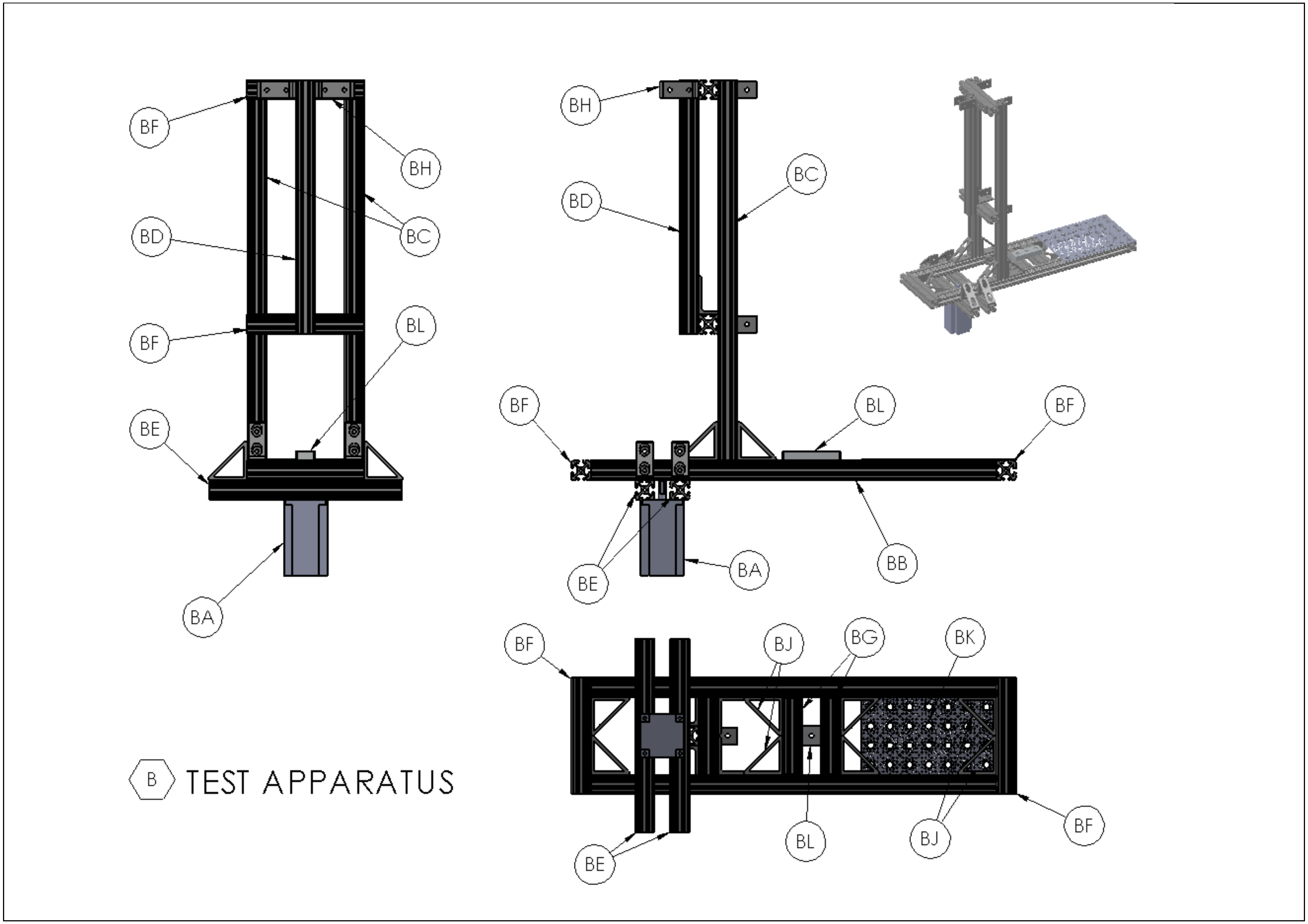


Figure A.4 Views of just the test apparatus with callouts labeling the different parts.

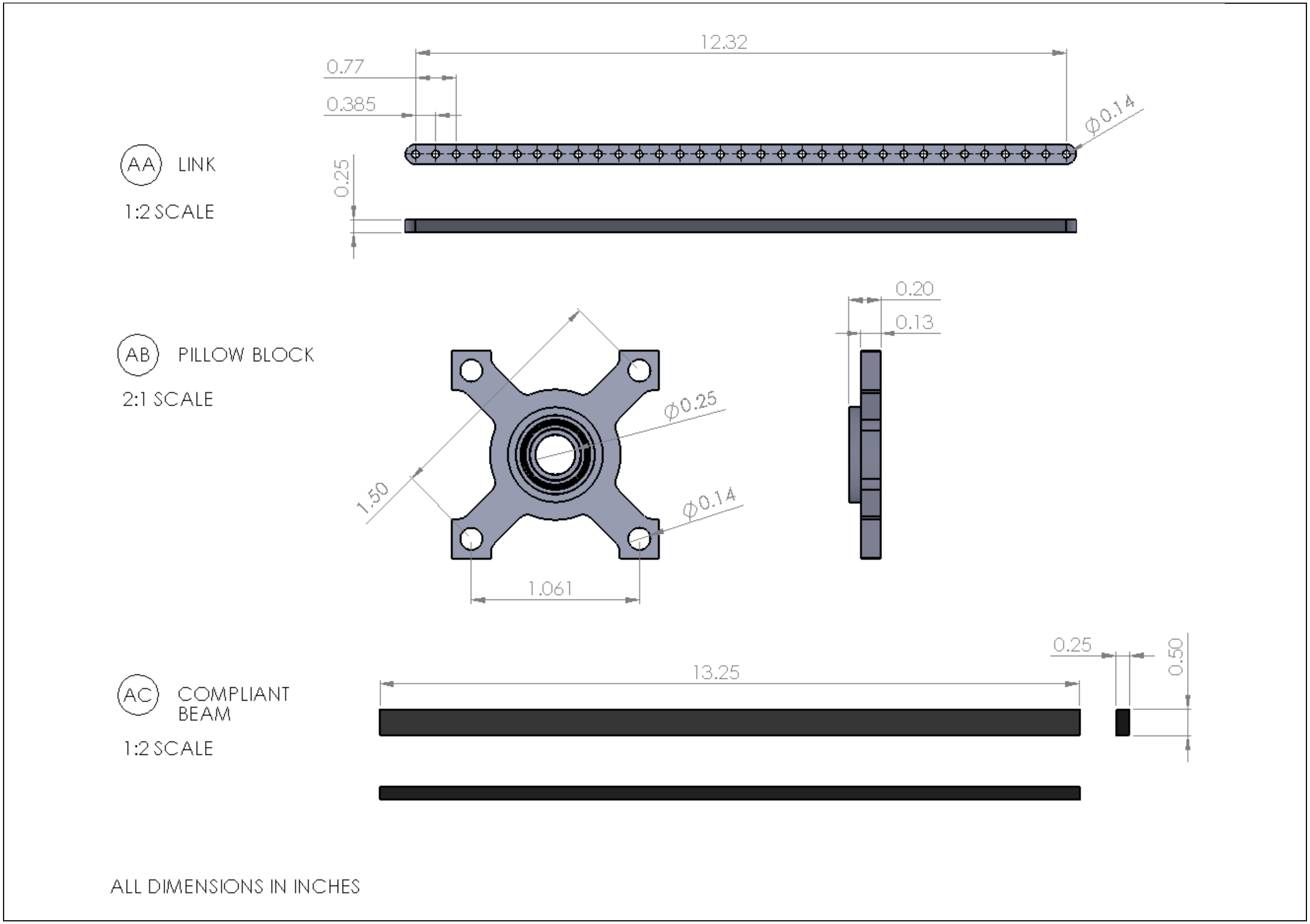


Figure A.5 Drawings for the link, pillow block, and compliant beam.

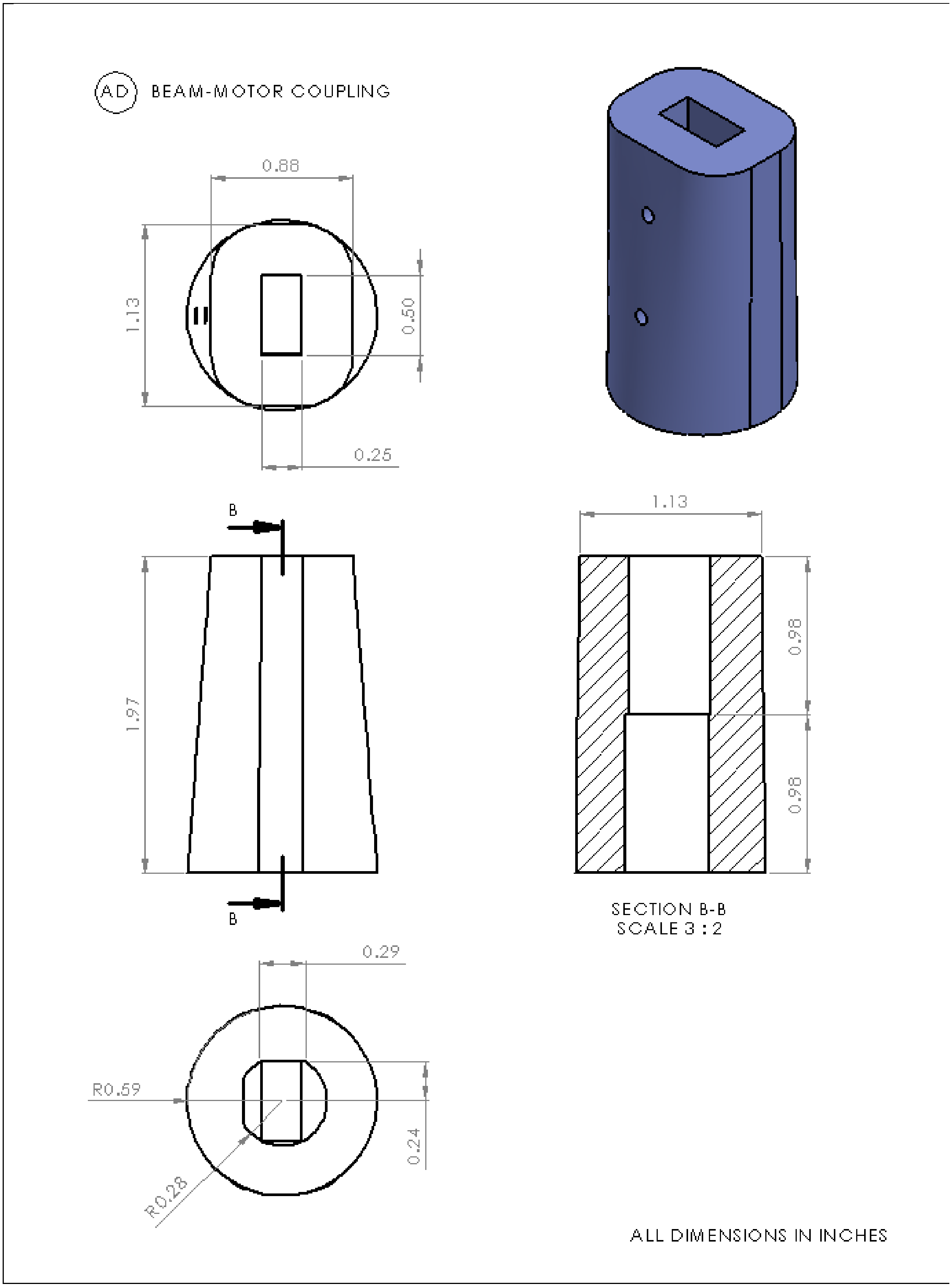


Figure A.6 Drawing for the beam-motor coupling.

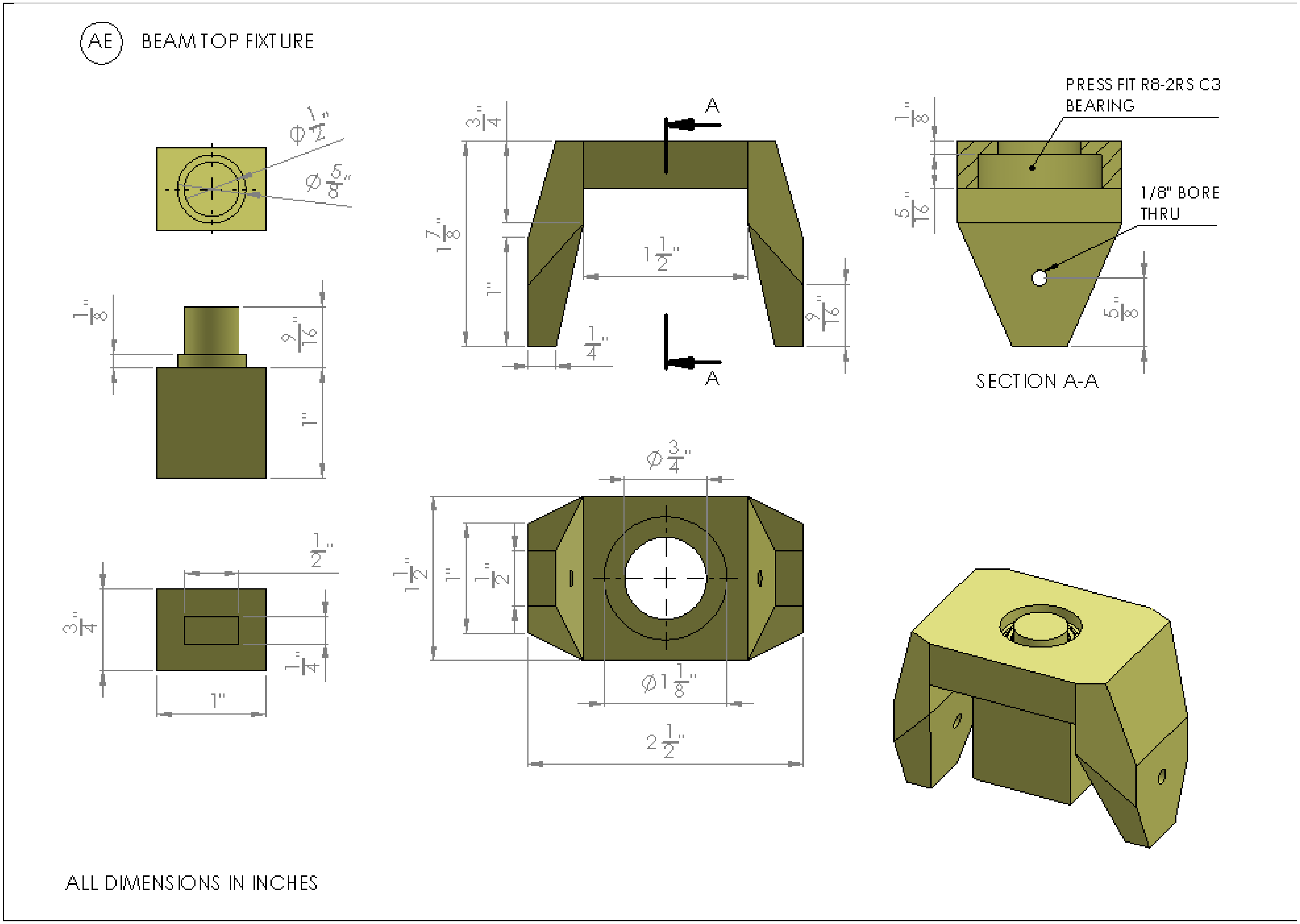


Figure A.7 Drawing for the beam top fixture.

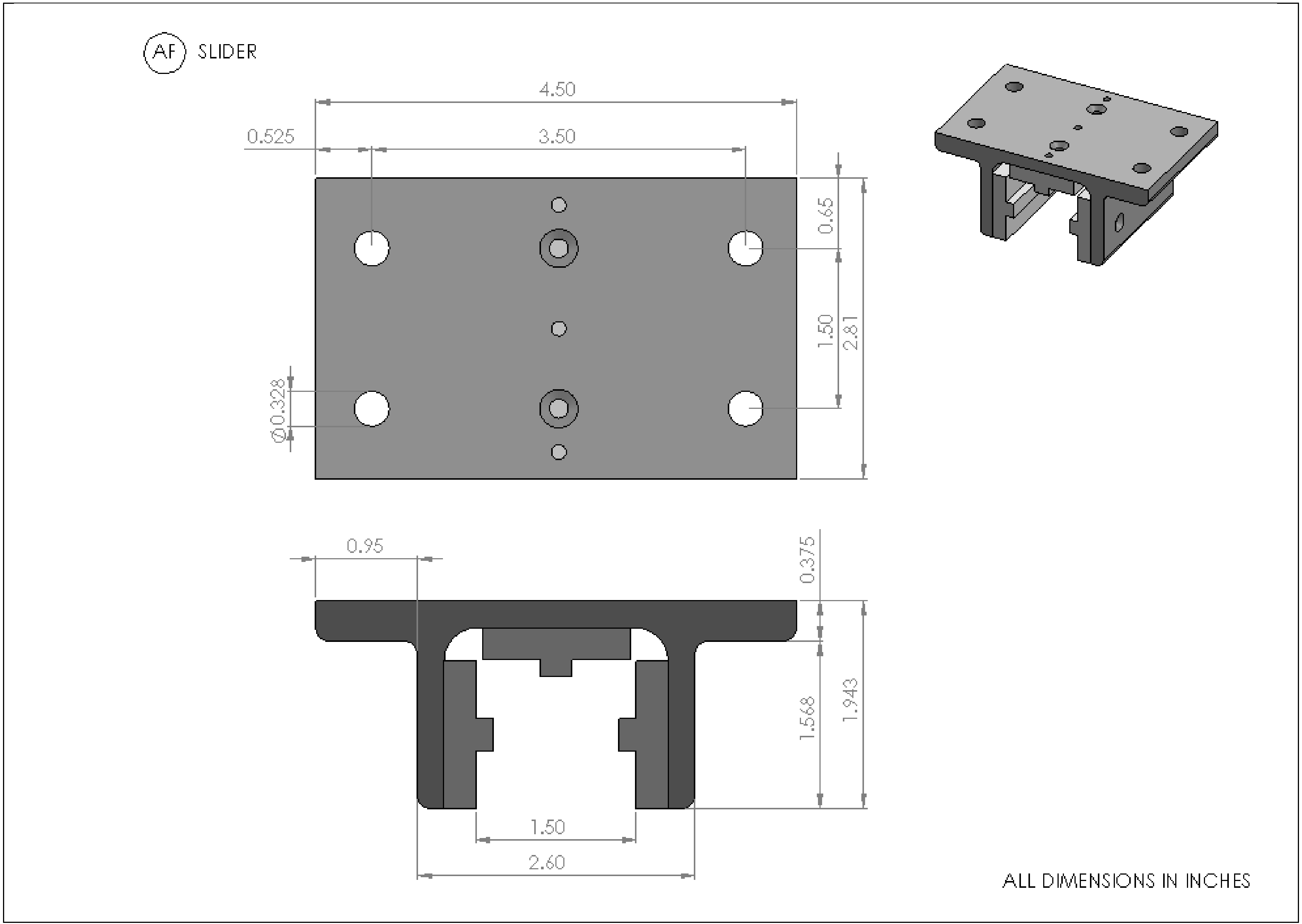


Figure A.8 Drawing for the slider.

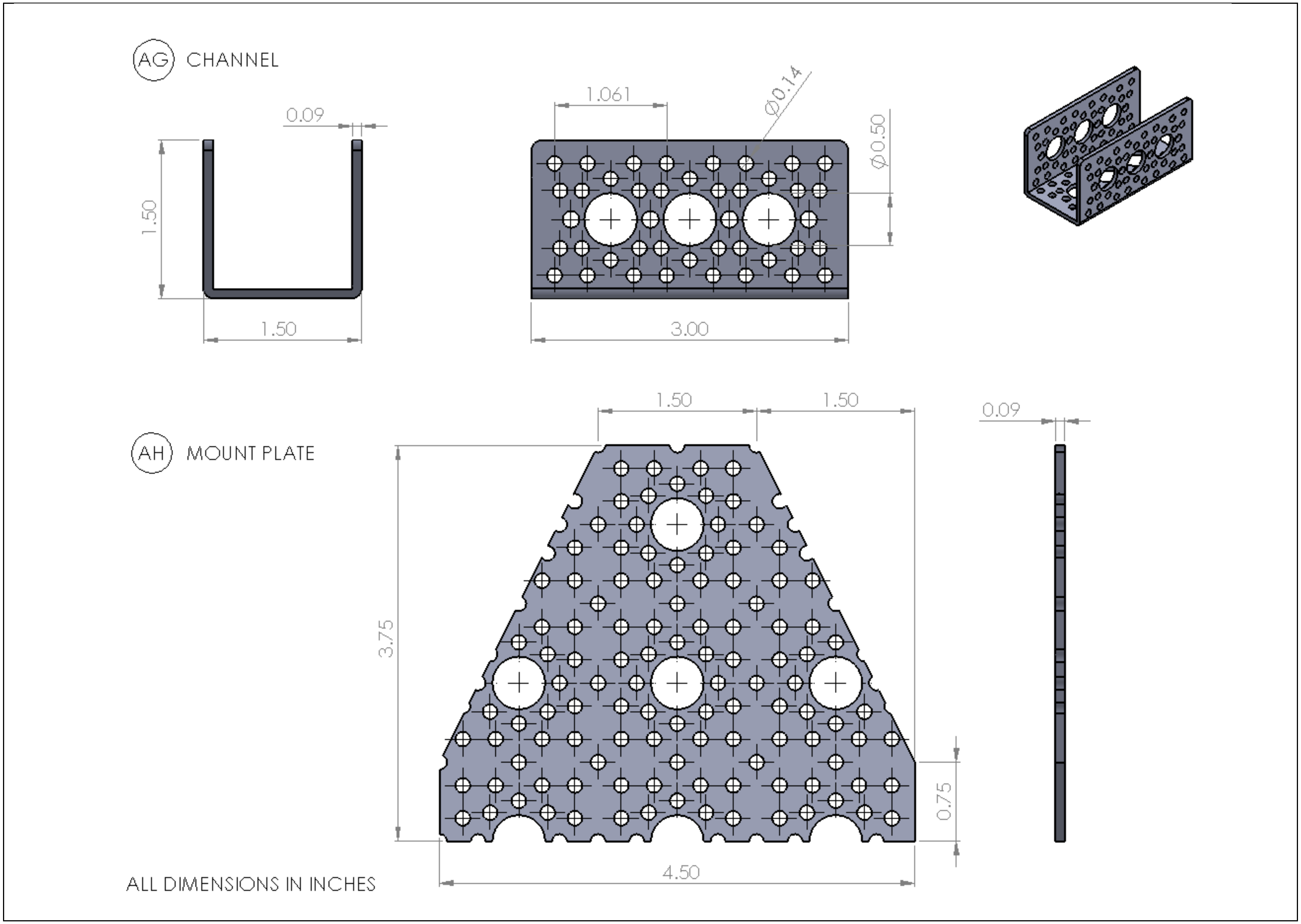


Figure A.9 Drawings for the channel and mount plate.

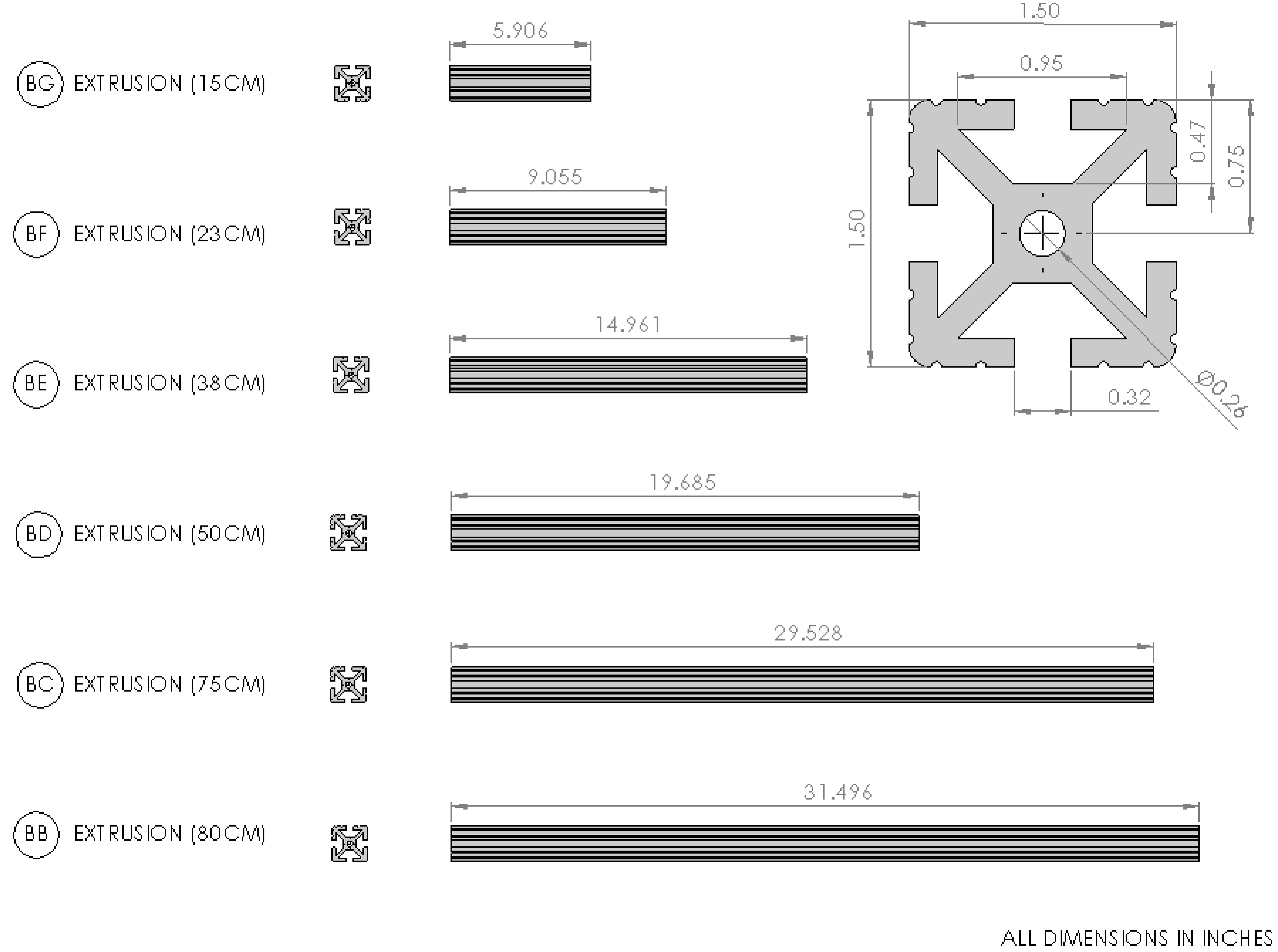


Figure A.10 Drawings for the various lengths of extrusions.

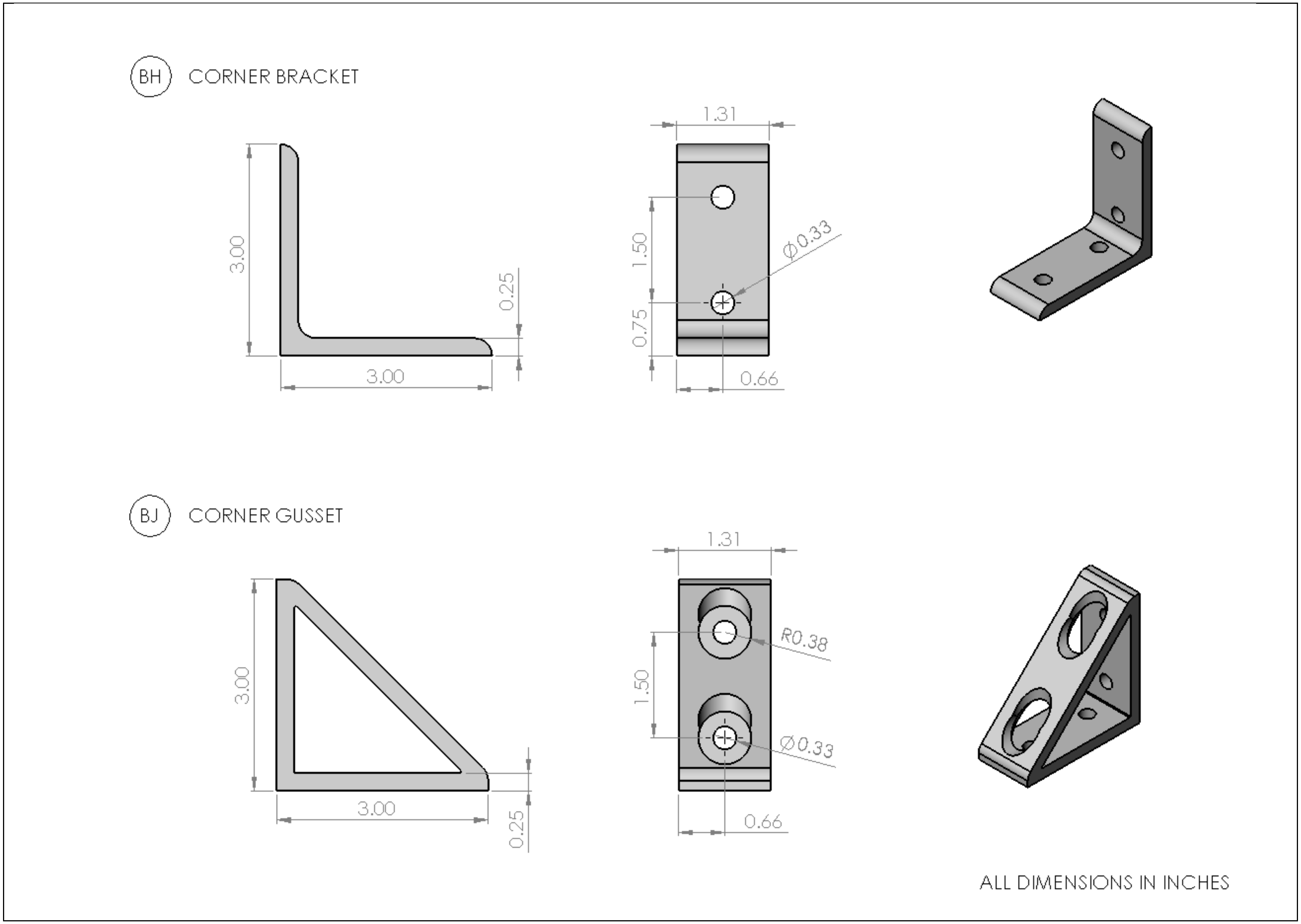


Figure A.11 Drawings for the corner bracket and corner gusset.

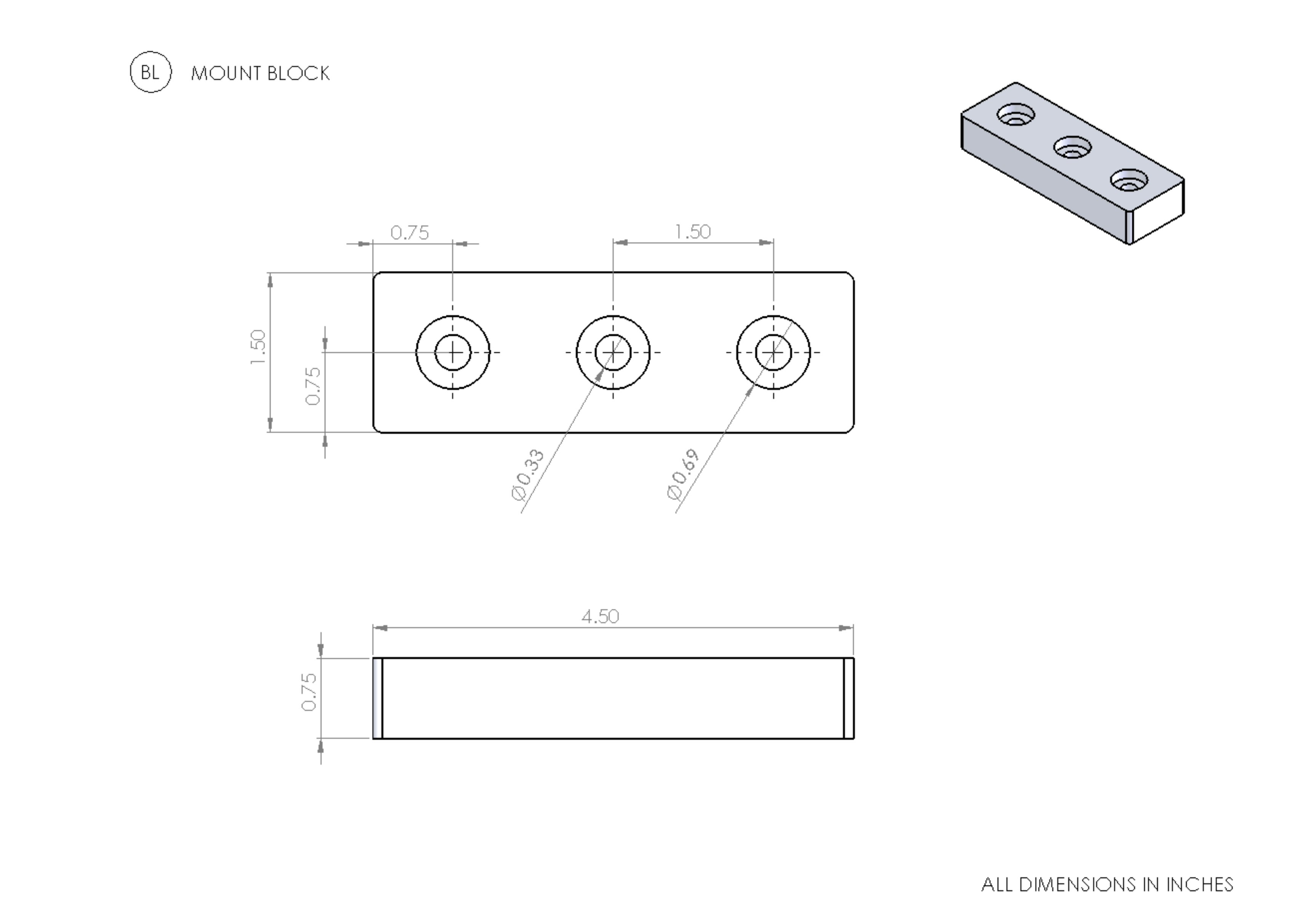


Figure A.12 Drawing for the mount block.

APPENDIX B: TESTING TABLES

This Appendix contains the tables similar to Table 7.1 for each test that had boxplots made of it. These tables are made from the test data and used to make the boxplots and midpoint plots shown in Chapter 7.

Table B.1 Min, max, midpoint, and quartile values with their corresponding motor positions for the initial run of Test 1.

Test 1.1

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.883	20.744	20.598	20.406	20.342	20.275	20.274	20.207	20.170
Beam Angle	111.1	115.1	114.6	113.9	115.2	115.1	116.3	114.1	114.4
Min	15.167	15.145	15.150	15.152	15.149	15.154	15.150	15.143	15.154
Beam Angle	87.1	88.3	89	88.2	90.9	91.1	91.8	91.6	92.1
Midpoint	18.025	17.945	17.874	17.779	17.746	17.715	17.712	17.675	17.662
Beam Angle	88.6	89.8	90.7	91.3	92.3	92.7	93.4	93.5	93.8
1/4 point	16.596	16.545	16.512	16.466	16.447	16.435	16.431	16.409	16.408
Beam Angle	88.4	89.6	90.4	89.5	92.1	92.5	93.1	93.3	93.5
3/4 point	19.454	19.345	19.236	19.093	19.044	18.995	18.993	18.941	18.916
Beam Angle	88.9	90.6	91.6	94.9	93.4	94.2	94.7	95.1	95.5

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.923	20.809	20.678	20.496	20.424	20.346	20.317	20.260	20.212
Beam Angle	59.3	72.5	76	76.9	76.9	79	78.3	79	81.4
Min	15.176	15.180	15.186	15.183	15.189	15.181	15.179	15.181	15.177
Beam Angle	37.2	40.6	43.5	46.3	47.6	48.6	49.7	50.6	51.5
Midpoint	18.050	17.995	17.932	17.839	17.807	17.764	17.748	17.721	17.694
Beam Angle	40.3	44.8	47.6	50.6	52.4	53.7	54.6	55.9	56.8
1/4 point	16.613	16.588	16.559	16.511	16.498	16.472	16.463	16.451	16.436
Beam Angle	39.4	43.5	46.4	49.3	50.8	52.3	52.9	54.1	54.9
3/4 point	19.486	19.402	19.305	19.167	19.116	19.055	19.032	18.990	18.953
Beam Angle	43.4	48.5	51.9	54.8	57	58.4	59.6	60.8	61.7

Table B.2 Min, max, midpoint, and quartile values with their corresponding motor positions for the first replicate of Test 1.

Test 1.2									
Up									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.929	20.845	20.644	20.506	20.350	20.258	20.222	20.106	20.000
Beam Angle	111.3	117.4	118.2	120	120	120	120	120	120
Min	15.210	15.197	15.197	15.200	15.180	15.197	15.187	15.191	15.196
Beam Angle	88.9	89.8	92	92.3	91.6	92.2	93.3	93.2	92.1
Midpoint	18.070	18.021	17.920	17.853	17.765	17.727	17.704	17.648	17.598
Beam Angle	94.1	94.8	96.9	98.3	98.2	98.6	99.3	99.4	99.6
1/4 point	16.640	16.609	16.558	16.526	16.473	16.462	16.446	16.419	16.397
Beam Angle	93.6	93.9	94.9	96.4	96.7	96.9	97.9	97.7	97.3
3/4 point	19.499	19.433	19.282	19.179	19.057	18.993	18.963	18.877	18.799
Beam Angle	94.9	96	98.3	100	100.4	101	101.4	101.7	102.5
Down									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.953	20.889	20.727	20.603	20.466	20.355	20.328	20.222	20.108
Beam Angle	59.8	68.2	73.6	74.4	77	79.5	80.1	80.8	82.4
Min	15.232	15.229	15.212	15.216	15.227	15.205	15.209	15.216	15.222
Beam Angle	38.4	42.5	45.3	46.6	48.5	50.8	51	52.4	52.3
Midpoint	18.092	18.059	17.969	17.909	17.846	17.780	17.769	17.719	17.665
Beam Angle	39.9	44.6	48.2	50.5	52.5	54.9	55.3	56.9	57.9
1/4 point	16.662	16.644	16.591	16.563	16.536	16.492	16.489	16.468	16.444
Beam Angle	39.1	43.6	46.8	48.8	50.8	52.7	53.2	54.7	55.4
3/4 point	19.523	19.474	19.348	19.256	19.156	19.068	19.048	18.971	18.887
Beam Angle	42.9	48.7	52.9	55.7	58	59.8	60.1	62	62.9

Table B.3 Min, max, midpoint, and quartile values with their corresponding motor positions for Test 2.

Test 2

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.934	20.929	20.884	20.842	20.770	20.726	20.710	20.624	20.588
Beam Angle	97.2	108.9	116.3	116.4	114.1	116.6	120	120	120
Min	16.688	16.697	16.685	16.700	16.698	16.699	16.696	16.696	16.701
Beam Angle	76.2	76.2	77	76.7	77.2	77.7	78.2	77.8	78.3
Midpoint	18.811	18.813	18.784	18.771	18.734	18.713	18.703	18.660	18.644
Beam Angle	79.6	80.8	82.4	83.3	83.7	84.2	84.4	85.3	86.2
1/4 point	17.750	17.755	17.734	17.735	17.716	17.706	17.699	17.678	17.672
Beam Angle	77.6	78	78.6	79.4	79.6	80.3	80.6	80.6	81.1
3/4 point	19.873	19.871	19.834	19.806	19.752	19.720	19.706	19.642	19.616
Beam Angle	81.5	83.5	85.3	86.5	87.3	87.8	88.4	89	89.7

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.942	20.947	20.925	20.899	20.822	20.777	20.772	20.681	20.631
Beam Angle	55.4	61	68.1	71.5	75	74.1	76.2	78.2	76.9
Min	16.719	16.713	16.722	16.719	16.725	16.725	16.727	16.728	16.726
Beam Angle	38.3	40.5	42.2	43.2	44.1	45.1	45.8	46.4	46.9
Midpoint	18.831	18.830	18.824	18.809	18.773	18.751	18.750	18.705	18.678
Beam Angle	40.8	43.1	44.7	46.1	47.2	48	48.7	49.5	50.1
1/4 point	17.775	17.772	17.773	17.764	17.749	17.738	17.739	17.717	17.702
Beam Angle	39.2	41.3	42.8	44.2	45.4	46.5	46.8	47.7	47.8
3/4 point	19.887	19.888	19.874	19.854	19.797	19.764	19.761	19.693	19.654
Beam Angle	42.9	46	48.8	50.5	52.4	53.4	54.1	55.5	56.6

Table B.4 Min, max, midpoint, and quartile values with their corresponding motor positions for Test 3.

Test 3

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.184	20.188	20.164	20.130	20.082	20.035	20.083	19.998	19.946
Beam Angle	95.7	99.2	105.8	108.4	111.2	119.7	120	120	120
Min	16.701	16.699	16.700	16.696	16.698	16.704	16.696	16.699	16.703
Beam Angle	76.1	76.2	76.2	77.4	78.3	79.1	79.3	79.1	79.2
Midpoint	18.442	18.443	18.432	18.413	18.390	18.369	18.389	18.349	18.324
Beam Angle	78.2	78.3	79.2	80.3	81.8	82.5	83	83.2	83.2
1/4 point	17.572	17.571	17.566	17.554	17.544	17.536	17.543	17.524	17.513
Beam Angle	77.7	77.6	78.3	79.3	80.9	81.6	82.3	82.2	82.1
3/4 point	19.313	19.316	19.298	19.272	19.236	19.202	19.236	19.173	19.135
Beam Angle	80.3	81	82.6	84.1	85.5	86.5	86.6	87.4	87.8

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.195	20.199	20.191	20.176	20.141	20.097	20.137	20.057	20.005
Beam Angle	53.2	59.2	64.3	67	72.8	74.5	74.8	74.1	72.4
Min	16.727	16.731	16.725	16.725	16.727	16.727	16.725	16.729	16.731
Beam Angle	33.6	36.2	38	37.2	38.3	39.6	39.4	41	41.6
Midpoint	18.461	18.465	18.458	18.451	18.434	18.412	18.431	18.393	18.368
Beam Angle	38.5	41.6	43.3	43.8	45.6	46.7	46.5	47.6	48.7
1/4 point	17.594	17.598	17.591	17.588	17.581	17.569	17.578	17.561	17.550
Beam Angle	35.3	38.1	39.7	39.6	40.8	42	42	43	44.2
3/4 point	19.328	19.332	19.324	19.314	19.288	19.254	19.284	19.225	19.187
Beam Angle	42	45.4	47.5	48.8	50.5	51.8	51.6	53	54

Table B.5 Min, max, midpoint, and quartile values with their corresponding motor positions for the first replicate of Test 4.

Test 4.2

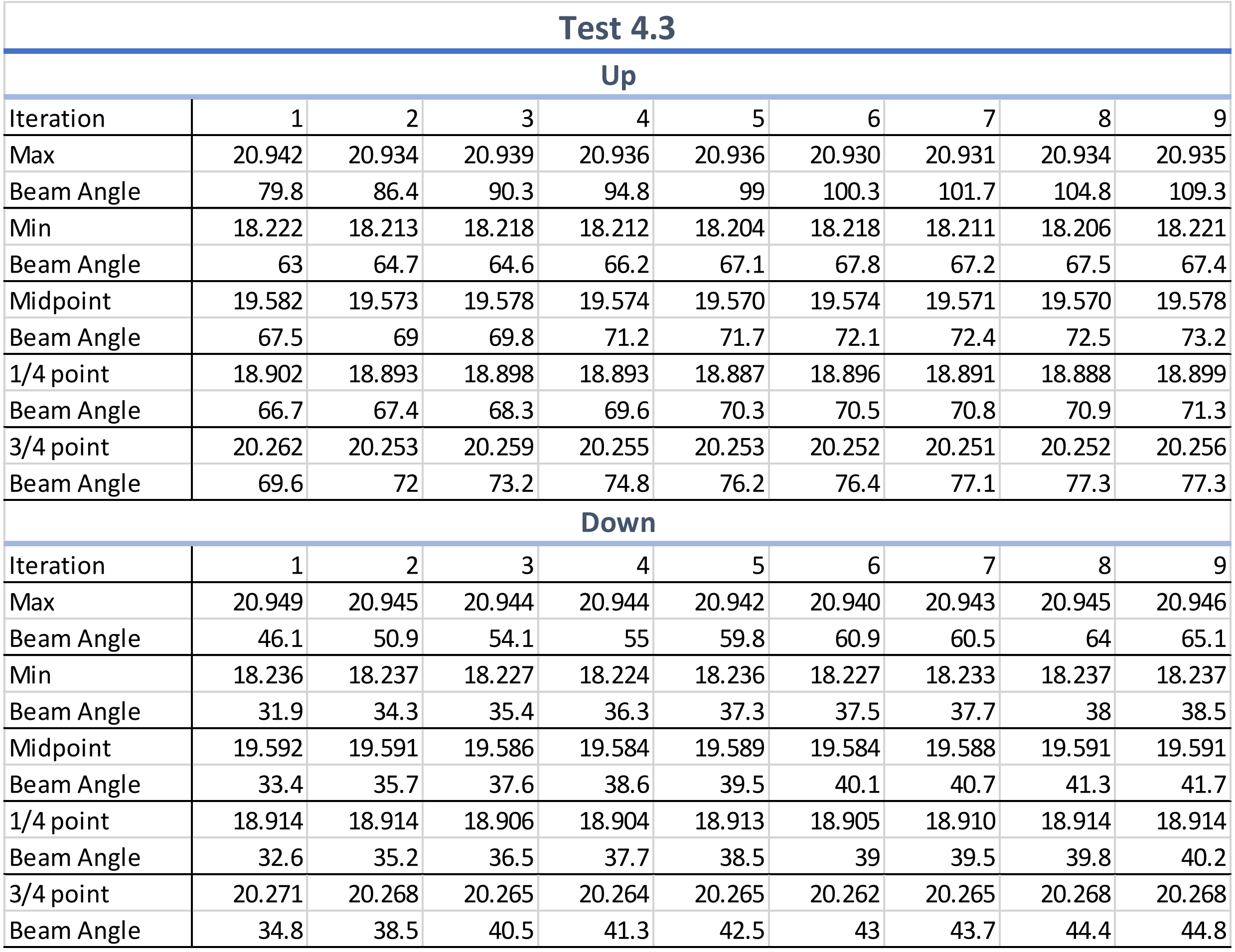
Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.944	20.937	20.939	20.936	20.942	20.936	20.941	20.934	20.938
Beam Angle	76.7	83	84.4	87.7	89.2	91.2	92.2	95.2	95.9
Min	18.223	18.225	18.231	18.226	18.219	18.231	18.225	18.216	18.228
Beam Angle	63.9	65.3	65.3	66.2	66	66.3	67.2	66.4	66.2
Midpoint	19.584	19.581	19.585	19.581	19.580	19.584	19.583	19.575	19.583
Beam Angle	66.2	67.5	68.2	69.3	69.3	69.4	70.2	69.9	69.9
1/4 point	18.903	18.903	18.908	18.904	18.900	18.907	18.904	18.895	18.905
Beam Angle	65.6	66.9	67.3	68.3	68.3	68.3	69.1	68.7	68.6
3/4 point	20.264	20.259	20.262	20.259	20.261	20.260	20.262	20.254	20.260
Beam Angle	67.9	69.8	71.1	71.7	72.3	72.7	73.4	74	74

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.965	20.964	20.961	20.957	20.956	20.949	20.952	20.953	20.942
Beam Angle	43.9	49.3	50.9	52.9	54.9	58.8	58.8	58.5	59.7
Min	18.246	18.245	18.234	18.247	18.252	18.237	18.245	18.249	18.239
Beam Angle	31.4	33.1	34.1	35.3	35.8	35.9	36.8	37.3	37.7
Midpoint	19.606	19.605	19.598	19.602	19.604	19.593	19.599	19.601	19.591
Beam Angle	32.6	34.6	36.1	37.1	38	38.5	38.6	39.4	39.7
1/4 point	18.926	18.925	18.916	18.925	18.928	18.915	18.922	18.925	18.915
Beam Angle	32.1	34	35.5	36.2	36.7	37.5	37.7	38.3	38.6
3/4 point	20.285	20.284	20.280	20.280	20.280	20.271	20.275	20.277	20.267
Beam Angle	34.7	37.2	38.8	40	40.9	41.8	42.5	43.1	43.6

Table B.6 Min, max, midpoint, and quartile values with their corresponding motor positions for the second replicate of Test 4.



Test 4.3

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.942	20.934	20.939	20.936	20.936	20.930	20.931	20.934	20.935
Beam Angle	79.8	86.4	90.3	94.8	99	100.3	101.7	104.8	109.3
Min	18.222	18.213	18.218	18.212	18.204	18.218	18.211	18.206	18.221
Beam Angle	63	64.7	64.6	66.2	67.1	67.8	67.2	67.5	67.4
Midpoint	19.582	19.573	19.578	19.574	19.570	19.574	19.571	19.570	19.578
Beam Angle	67.5	69	69.8	71.2	71.7	72.1	72.4	72.5	73.2
1/4 point	18.902	18.893	18.898	18.893	18.887	18.896	18.891	18.888	18.899
Beam Angle	66.7	67.4	68.3	69.6	70.3	70.5	70.8	70.9	71.3
3/4 point	20.262	20.253	20.259	20.255	20.253	20.252	20.251	20.252	20.256
Beam Angle	69.6	72	73.2	74.8	76.2	76.4	77.1	77.3	77.3

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.949	20.945	20.944	20.944	20.942	20.940	20.943	20.945	20.946
Beam Angle	46.1	50.9	54.1	55	59.8	60.9	60.5	64	65.1
Min	18.236	18.237	18.227	18.224	18.236	18.227	18.233	18.237	18.237
Beam Angle	31.9	34.3	35.4	36.3	37.3	37.5	37.7	38	38.5
Midpoint	19.592	19.591	19.586	19.584	19.589	19.584	19.588	19.591	19.591
Beam Angle	33.4	35.7	37.6	38.6	39.5	40.1	40.7	41.3	41.7
1/4 point	18.914	18.914	18.906	18.904	18.913	18.905	18.910	18.914	18.914
Beam Angle	32.6	35.2	36.5	37.7	38.5	39	39.5	39.8	40.2
3/4 point	20.271	20.268	20.265	20.264	20.265	20.262	20.265	20.268	20.268
Beam Angle	34.8	38.5	40.5	41.3	42.5	43	43.7	44.4	44.8

Table B.7 Min, max, midpoint, and quartile values with their corresponding motor positions for the initial run of Test 7.

Test 7.1

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.939	20.927	20.937	20.931	20.929	20.942	20.932	20.939	20.931
Beam Angle	74.8	75.7	80.1	81	82.7	84.5	84.1	85.2	86.1
Min	18.205	18.197	18.202	18.192	18.195	18.195	18.193	18.191	18.192
Beam Angle	66.3	67.8	68.1	69.4	68.7	68.7	69.2	68.8	69.4
Midpoint	19.572	19.562	19.569	19.561	19.562	19.568	19.563	19.565	19.562
Beam Angle	71.8	72.8	72.5	74	73.5	73.9	74.6	74.1	74.3
1/4 point	18.888	18.879	18.886	18.877	18.879	18.882	18.878	18.878	18.877
Beam Angle	70.7	71.7	71.4	73	72.3	72.7	73.5	72.9	73.2
3/4 point	20.256	20.244	20.253	20.246	20.246	20.255	20.248	20.252	20.246
Beam Angle	72.6	73.5	73.3	74.7	74.4	74.7	75.5	75	75.2

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.953	20.940	20.945	20.940	20.941	20.942	20.938	20.940	20.948
Beam Angle	43.6	46.5	48.3	49.3	50.7	50.9	51.9	53.1	53.3
Min	18.222	18.220	18.339	18.218	18.220	18.332	18.219	18.222	18.340
Beam Angle	27.7	28.9	29.5	30.8	31.3	31.8	32.2	32	32.3
Midpoint	19.588	19.580	19.642	19.579	19.580	19.637	19.578	19.581	19.644
Beam Angle	31.3	32.7	33.7	34.5	35.1	36	35.9	36.6	37.2
1/4 point	18.905	18.900	18.991	18.899	18.900	18.985	18.899	18.902	18.992
Beam Angle	29.8	31.1	32.1	32.9	33.5	33.9	34.2	34.6	35
3/4 point	20.271	20.260	20.293	20.260	20.261	20.290	20.258	20.261	20.296
Beam Angle	34.3	35.7	36.7	37.9	38.7	39.5	39.6	40.4	40.6

Table B.8 Min, max, midpoint, and quartile values with their corresponding motor positions for the first replicate of Test 7.

Test 7.2

Up

Iteration	1	2	3	4	5	6	7	8	9
Max	20.933	20.938	20.936	20.927	20.934	20.936	20.934	20.938	20.938
Beam Angle	79	79.7	81.4	83.5	84.9	86.8	86	87.8	88.6
Min	18.208	18.211	18.207	18.206	18.207	18.207	18.210	18.209	18.206
Beam Angle	62.8	65.1	65.1	65.8	65.8	66.8	67.1	67.3	66.8
Midpoint	19.571	19.574	19.571	19.566	19.570	19.572	19.572	19.573	19.572
Beam Angle	67.9	69.5	70	71.7	71.5	71.8	72.6	72.2	72.5
1/4 point	18.889	18.893	18.889	18.886	18.889	18.889	18.891	18.891	18.889
Beam Angle	66.7	68.2	68.6	70.4	70.2	70.5	71.3	70.9	71.1
3/4 point	20.252	20.256	20.253	20.247	20.252	20.254	20.253	20.256	20.255
Beam Angle	68.9	70.6	71.1	72.6	72.5	72.9	73.6	73.4	73.6

Down

Iteration	1	2	3	4	5	6	7	8	9
Max	20.943	20.943	20.944	20.943	20.940	20.940	20.944	20.941	20.940
Beam Angle	42.6	45.6	48.5	51	50.1	52	51.8	53.8	54.3
Min	18.230	18.228	18.345	18.224	18.225	18.345	18.226	18.227	18.343
Beam Angle	26.5	27.8	29.3	30.7	31.4	30.9	31.5	31.5	31.3
Midpoint	19.586	19.586	19.645	19.584	19.582	19.643	19.585	19.584	19.642
Beam Angle	30.9	32.2	33.2	34.5	35.1	35.6	35.8	36.1	36.7
1/4 point	18.908	18.907	18.995	18.904	18.904	18.994	18.905	18.905	18.992
Beam Angle	29.5	31	31.9	33	33.5	33.9	34	34.4	34.7
3/4 point	20.264	20.264	20.294	20.264	20.261	20.292	20.264	20.263	20.291
Beam Angle	33.7	35.3	36.3	37.6	38.1	38.9	39.2	39.7	40.2

Table B.9 Min, max, midpoint, and quartile values with their corresponding motor positions for the second replicate of Test 7.

Test 7.3									
Up									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.945	20.939	20.932	20.935	20.933	20.930	20.935	20.932	20.937
Beam Angle	79.5	84.4	87.2	90	92.5	94.7	93.6	96.7	97.6
Min	18.220	18.211	18.215	18.205	18.208	18.208	18.206	18.209	18.210
Beam Angle	62	64	65	66.6	65.7	67.4	68.6	68.3	67.7
Midpoint	19.583	19.575	19.573	19.570	19.570	19.569	19.570	19.570	19.573
Beam Angle	68.7	70.1	71	72.8	72.9	73.5	74.3	74.1	74.2
1/4 point	18.901	18.893	18.894	18.887	18.889	18.888	18.888	18.890	18.891
Beam Angle	67.4	68.2	69.2	71.1	71	72.1	72.9	72.5	72.5
3/4 point	20.264	20.257	20.253	20.252	20.252	20.250	20.253	20.251	20.255
Beam Angle	70.1	71.6	72.6	74.1	74.3	75.1	75.8	76	76.3
Down									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.965	20.945	20.948	20.942	20.944	20.940	20.940	20.941	20.942
Beam Angle	47	50.4	52.7	54.1	56.6	58.3	58.4	59.4	61.8
Min	18.231	18.230	18.348	18.223	18.232	18.346	18.227	18.228	18.343
Beam Angle	29.5	30.4	31.2	31.9	32.7	32.7	33.5	34.2	34.4
Midpoint	19.598	19.587	19.648	19.583	19.588	19.643	19.583	19.584	19.643
Beam Angle	32.9	34.7	36	37	37.8	38.3	38.4	39.1	39.7
1/4 point	18.914	18.909	18.998	18.903	18.910	18.994	18.905	18.906	18.993
Beam Angle	31.3	33	34.1	35.1	35.5	35.9	36.1	36.5	36.9
3/4 point	20.282	20.266	20.298	20.263	20.266	20.291	20.262	20.263	20.292
Beam Angle	35.9	38	39.6	41.1	41.8	42.4	42.4	43.2	43.8

Table B.10 Min, max, midpoint, and quartile values with their corresponding motor positions for Test 10.

Test 10									
Up									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.805	20.436	20.177	20.046	19.813	19.685	19.688	19.537	19.486
Beam Angle	112	111.9	113.7	110.2	109.6	107.6	109.8	110	110.8
Min	18.218	18.212	18.218	18.211	18.211	18.215	18.212	18.210	18.214
Beam Angle	80.2	82.3	83.1	84.7	85.5	85.1	85.6	86.4	86.5
Midpoint	19.511	19.324	19.197	19.129	19.012	18.950	18.950	18.874	18.850
Beam Angle	85.7	88.5	89.6	90.8	91.4	92	92.4	92.7	93.4
1/4 point	18.864	18.768	18.707	18.670	18.611	18.583	18.581	18.542	18.532
Beam Angle	82.8	85.9	87.2	88.4	89.2	89.1	89.2	89.5	90
3/4 point	20.158	19.880	19.687	19.587	19.412	19.318	19.319	19.206	19.168
Beam Angle	91.9	94.4	96.3	97.6	96.8	96.6	97.7	97.1	97.8
Down									
Iteration	1	2	3	4	5	6	7	8	9
Max	20.870	20.571	20.275	20.147	19.913	19.748	19.727	19.601	19.518
Beam Angle	70.6	77.6	78.5	80.7	83.8	85.4	85.1	84.7	84.9
Min	18.229	18.230	18.227	18.226	18.228	18.228	18.229	18.227	18.225
Beam Angle	51.2	55.7	58.5	59.7	61.8	63.4	63.9	65.5	66.2
Midpoint	19.549	19.400	19.251	19.187	19.070	18.988	18.978	18.914	18.871
Beam Angle	53.5	59.5	62.6	64.1	66.5	68.7	69.7	70.3	71.2
1/4 point	18.889	18.815	18.739	18.706	18.649	18.608	18.604	18.570	18.548
Beam Angle	52.1	56.8	59.7	61	63.6	65.6	66.2	67.4	68
3/4 point	20.210	19.986	19.763	19.667	19.492	19.368	19.352	19.257	19.194
Beam Angle	56.6	63	66.8	68.7	71.3	73.8	74.3	75.6	76.6

APPENDIX C: ARDUINO CODE

This appendix contains all the Arduino code used for testing.

C.1 Tests 1-4, 10

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}

void loop() {
  int steps = 1;

 for(int z = 0; z < 3; z++)
 {
  digitalWrite(dir,HIGH); //ccw
  int direc = 1;

  for(int y = 0; y < 119; y++)
```

```
{
  int iter = 0;
  for(int x = 0; x < 10; x++)
  {
    for (int j=0; j<10; j++)
    {
      e5=e4;
      e4=e3;
      e3=e2;
      e2=e1;
      e1=e;
      e=d;
      d=c;
      c=b;
      b=a;
      a = analogRead(sensB);
    }
    avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
    //channel[0] = analogRead(sensA);
    channel[1] = analogRead(sensB);
    Serial.print(moPos);
    Serial.print("\t");
    Serial.print(moPos+(iter*.1*direc));
    Serial.print("\t");
    Serial.println(avg);
    iter = iter + 1;
    delay(100);
  }

  for(int x = 0; x < steps; x++)
  {
    digitalWrite(puls, HIGH);
    delay(1);
    digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
    delay(1);
  }
  moPos = moPos + steps;
}

digitalWrite(dir, LOW); //cw
direc = -1;

for(int y = 0; y < 119; y++)
{
  int iter = 0;
  for(int x = 0; x < 10; x++)
  {
    for (int j=0; j<10; j++)
    {
      e5=e4;
      e4=e3;
      e3=e2;
      e2=e1;
      e1=e;
      e=d;
      d=c;
      c=b;
```

```
        b=a;
        a = analogRead(sensB);
      }
      avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
      //channel[0] = analogRead(sensA);
      channel[1] = analogRead(sensB);
      Serial.print(moPos);
      Serial.print("\t");
      Serial.print(moPos+(iter*.1*direc));
      Serial.print("\t");
      Serial.println(avg);
      iter = iter + 1;
      delay(100);
    }

    for(int x = 0; x < steps; x++)
    {
      digitalWrite(puls, HIGH);
      delay(1);
      digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
      delay(1);
    }
    moPos = moPos - steps;
  }
 }

  exit(0);
}
```

C.2 Test 5

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}

void loop() {
  int steps = 1;

   for(int z = 0; z < 3; z++)
   {
    digitalWrite(dir,HIGH); //ccw
    int direc = 1;

    for(int y = 0; y < 119; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(50);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos + steps;
    }
    digitalWrite(dir, LOW); //cw
    direc = -1;

    for(int y = 0; y < 119; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(50);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos - steps;
    }
   }

    exit(0);
}
```

C.3 Test 6

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}

void loop() {
  int steps = 1;

   for(int z = 0; z < 3; z++)
   {
    digitalWrite(dir,HIGH); //ccw
    int direc = 1;

    for(int y = 0; y < 119; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(25);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos + steps;
    }

    digitalWrite(dir, LOW); //cw
    direc = -1;

    for(int y = 0; y < 119; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(25);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
```

```
        digitalWrite(puls, LOW);     // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos - steps;
    }
   }

    exit(0);
}
```

C.4 Test 7

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}

void loop() {
  int steps = 1;

   for(int z = 0; z < 3; z++)
   {
    digitalWrite(dir,HIGH); //ccw
    int direc = 1;

    for(int y = 0; y < 119; y++)
    {
```

```
  int iter = 0;
  for(int x = 0; x < 10; x++)
  {
    for (int j=0; j<10; j++)
    {
      e5=e4;
      e4=e3;
      e3=e2;
      e2=e1;
      e1=e;
      e=d;
      d=c;
      c=b;
      b=a;
      a = analogRead(sensB);
    }
    avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
    //channel[0] = analogRead(sensA);
    channel[1] = analogRead(sensB);
    Serial.print(moPos);
    Serial.print("\t");
    Serial.print(moPos+(iter*.1*direc));
    Serial.print("\t");
    Serial.println(avg);
    iter = iter + 1;
    delay(10);
  }

  for(int x = 0; x < steps; x++)
  {
    digitalWrite(puls, HIGH);
    delay(1);
    digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
    delay(1);
  }
  moPos = moPos + steps;
}

digitalWrite(dir, LOW); //cw
direc = -1;

for(int y = 0; y < 119; y++)
{
  int iter = 0;
  for(int x = 0; x < 10; x++)
  {
    for (int j=0; j<10; j++)
    {
      e5=e4;
      e4=e3;
      e3=e2;
      e2=e1;
      e1=e;
      e=d;
      d=c;
      c=b;
      b=a;
```

```
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(10);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos - steps;
    }
   }

    exit(0);
}
```

C.5 Test 8

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}

void loop() {
  int steps = 1;

//up to hys loop beginning
  digitalWrite(dir,HIGH); //ccw
  int direc = 1;

  for(int y = 0; y < 79; y++)
  {
    int iter = 0;
    for(int x = 0; x < 10; x++)
    {
      for (int j=0; j<10; j++)
      {
        e5=e4;
        e4=e3;
        e3=e2;
        e2=e1;
        e1=e;
        e=d;
        d=c;
        c=b;
        b=a;
        a = analogRead(sensB);
      }
      avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
      //channel[0] = analogRead(sensA);
      channel[1] = analogRead(sensB);
      Serial.print(moPos);
      Serial.print("\t");
      Serial.print(moPos+(iter*.1*direc));
      Serial.print("\t");
      Serial.println(avg);
      iter = iter + 1;
      delay(100);
    }

    for(int x = 0; x < steps; x++)
    {
      digitalWrite(puls, HIGH);
      delay(1);
      digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
      delay(1);
    }
    moPos = moPos + steps;
  }
//start hys loop
   for(int z = 0; z < 6; z++)
   {
    digitalWrite(dir, LOW); //cw
    direc = -1;

    for(int y = 0; y < 20; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(100);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos - steps;
    }

//up
    digitalWrite(dir,HIGH); //ccw
    int direc = 1;

    for(int y = 0; y < 20; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(100);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos + steps;
    }
   }

//end loop, back down
  digitalWrite(dir, LOW); //cw
  direc = -1;

  for(int y = 0; y < 79; y++)
  {
    int iter = 0;
    for(int x = 0; x < 10; x++)
    {
      for (int j=0; j<10; j++)
      {
        e5=e4;
        e4=e3;
        e3=e2;
        e2=e1;
        e1=e;
        e=d;
        d=c;
        c=b;
        b=a;
        a = analogRead(sensB);
      }
      avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
      //channel[0] = analogRead(sensA);
      channel[1] = analogRead(sensB);
      Serial.print(moPos);
      Serial.print("\t");
      Serial.print(moPos+(iter*.1*direc));
      Serial.print("\t");
      Serial.println(avg);
      iter = iter + 1;
      delay(100);
    }

    for(int x = 0; x < steps; x++)
    {
      digitalWrite(puls, HIGH);
      delay(1);
      digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
      delay(1);
    }
    moPos = moPos - steps;
  }

    exit(0);
}
```

C.6 Test 9

```
int ena = 11;
int dir = 12;
int puls = 13;
int sensA = A0;
int sensB = A1;
int channel[2];
int moPos = 1;
int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;
int e1 = 0;
int e2 = 0;
int e3 = 0;
int e4 = 0;
int e5 = 0;

int avg = 0;

void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(puls, OUTPUT);
  pinMode(sensA, INPUT);
  pinMode(sensB, INPUT);
  digitalWrite(ena, LOW); //activates motor
  digitalWrite(dir,HIGH); //ccw
  Serial.begin(9600);

}
```

```
void loop() {
  int steps = 1;

//up to hys loop beginning
  digitalWrite(dir,HIGH); //ccw
  int direc = 1;

  for(int y = 0; y < 74; y++)
  {
    int iter = 0;
    for(int x = 0; x < 10; x++)
    {
      for (int j=0; j<10; j++)
      {
        e5=e4;
        e4=e3;
        e3=e2;
        e2=e1;
        e1=e;
        e=d;
        d=c;
        c=b;
        b=a;
        a = analogRead(sensB);
      }
      avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
      //channel[0] = analogRead(sensA);
      channel[1] = analogRead(sensB);
      Serial.print(moPos);
      Serial.print("\t");
      Serial.print(moPos+(iter*.1*direc));
      Serial.print("\t");
      Serial.println(avg);
      iter = iter + 1;
      delay(100);
    }

    for(int x = 0; x < steps; x++)
    {
      digitalWrite(puls, HIGH);
      delay(1);
      digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
      delay(1);
    }
    moPos = moPos + steps;
  }
//start hysteresis loop
   for(int z = 0; z < 6; z++)
   {
    digitalWrite(dir, LOW); //cw
    direc = -1;

    for(int y = 0; y < 40; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(100);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos - steps;
    }

//up
    digitalWrite(dir,HIGH); //ccw
    int direc = 1;

    for(int y = 0; y < 40; y++)
    {
      int iter = 0;
      for(int x = 0; x < 10; x++)
      {
        for (int j=0; j<10; j++)
        {
          e5=e4;
          e4=e3;
          e3=e2;
          e2=e1;
          e1=e;
          e=d;
          d=c;
          c=b;
          b=a;
          a = analogRead(sensB);
        }
        avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
        //channel[0] = analogRead(sensA);
        channel[1] = analogRead(sensB);
        Serial.print(moPos);
        Serial.print("\t");
        Serial.print(moPos+(iter*.1*direc));
        Serial.print("\t");
        Serial.println(avg);
        iter = iter + 1;
        delay(100);
      }

      for(int x = 0; x < steps; x++)
      {
        digitalWrite(puls, HIGH);
        delay(1);
        digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
        delay(1);
      }
      moPos = moPos + steps;
    }
   }

//end of loop, back down
  digitalWrite(dir, LOW); //cw
  direc = -1;

  for(int y = 0; y < 74; y++)
  {
    int iter = 0;
    for(int x = 0; x < 10; x++)
    {
      for (int j=0; j<10; j++)
      {
        e5=e4;
        e4=e3;
        e3=e2;
        e2=e1;
        e1=e;
        e=d;
        d=c;
        c=b;
        b=a;
        a = analogRead(sensB);
      }
      avg = (a+b+c+d+e+e1+e2+e3+e4+e5)/10;
      //channel[0] = analogRead(sensA);
      channel[1] = analogRead(sensB);
      Serial.print(moPos);
      Serial.print("\t");
      Serial.print(moPos+(iter*.1*direc));
      Serial.print("\t");
      Serial.println(avg);
      iter = iter + 1;
      delay(100);
    }
    for(int x = 0; x < steps; x++)
    {
      digitalWrite(puls, HIGH);
      delay(1);
      digitalWrite(puls, LOW);    // emits a 2ms pulse, advancing 1 step
      delay(1);
    }
    moPos = moPos - steps;
  }

    exit(0);
}
```