Active Surface Shaping for Reflectors
Final Report

EML 4551C – Senior Design – Spring 2013 Deliverable

Team # 9

Cameron Duncan (cyd07@my.fsu.edu), Akeem Jordan (aij08@my.fsu.edu),
Raymond Mak (rtm07@my.fsu.edu)

Department of Mechanical Engineering, Florida State University, Tallahassee, FL

Project Sponsor
Mr. Gustavo Toledo

Project Advisor(s)
Dr. Kamal Amin, Ph.D.
Department of Mechanical Engineering

Dr. William S. Oates, Ph.D.
Department of Mechanical Engineering

Reviewed by Advisor(s):

Table of Contents

Abstract

The Harris Corporation currently uses manually adjustable mechanisms in order to fine tune their mesh reflectors to the desired geometry. Harris gave senior design team nine the opportunity to design, construct, and test a fully automated adjuster mechanism which would replace the previous manual mechanism. A slightly altered manual mechanism was provided to team nine in order to showcase the parallel pull design that Harris is switching to for each adjuster. This parallel pull design was then further modified through several iterations to be able to house three independent, micro stepper motors controlled by an Arduino Nano. To meet the required resolution for each adjustment, a 2:1 gear ratio was used on each motor. The central bore for the gears were threaded in order to translate the threaded rods which pass through the gears as they are rotated. The translation of the threaded rods is what provided the cord adjustments.

As these adjustments must be very precise, a means of measuring the achieved displacement had to be implemented. To simulate the adjustments that would be made on an actual reflector, the prototype mechanism was mounted to a cubic frame along with two string potentiometers attached to the adjustment cords. As adjustments are made, the string potentiometers send signals into the data acquisition module where the signals are conditioned and adapted into linear distances.

After rigorous testing, it was determined that the prototype was able to achieve a linear resolution of 0.003” and 0.007” when doing a consecutive adjustment in the opposite direction. To increase the overall linear resolution accuracy, the team has recommended improvements for future work such as higher precision gears, higher step count motors, and use of a higher precision 3D printer for the base and other components with a higher strength material.

Introduction

The Harris Corporation is an international communications and information technology company dedicated to developing best-in-class assured communications products, systems, and services. One of the many systems they produce are mesh reflectors designed to send and receive communication signals for military and commercial applications. These mesh reflectors must first be adjusted to specific surface geometries on ground, then stowed inside of a shuttle before they are deployed in space. A typical reflector described to the team by our Harris sponsor has eight ribs stemming from its center which constitute the main structural support of the mesh material as seen in Figure 1. Along each of these ribs, there are seventeen adjustment mechanisms that are responsible for three adjustment points. In total, there are 136 adjustment mechanisms located on a reflector with 408 individual adjustments. For certain high frequency applications, this process becomes even more complex since the number of adjustments are greatly increased as well as the resolution of these adjustments. Achieving desired geometrical surface shape is labor intensive and time consuming due the current process which requires manual adjustments hundreds, and possibly thousands of times.

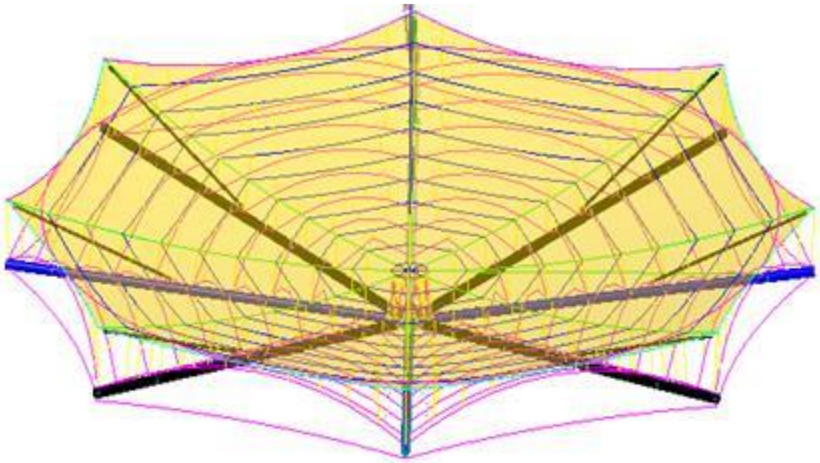


Figure 1 Eight rib reflector

Background

Figure 2 shows a two dimensional cross sectional view of the adjustment mechanism attached to the reflector. The blue represents the surface mesh of the reflector, black is the structural rib of the reflector where the mechanism is physically mounted onto, and the green lines are supportive cords. The yellow and red lines are the adjustable components of the reflector to change the surface geometry of the mesh. The yellow lines are straws that are adjusted vertically and red lines are the adjustment cords that are tightened and loosed until desired. When the adjustments are finalized, the cords are crisscrossed and bonded together, then the mechanism can be removed from the points indicated by the yellow arrows seen in Figure 4.

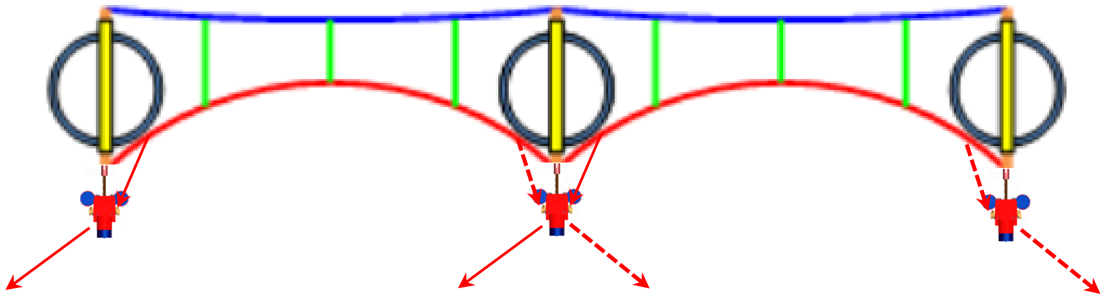


Figure 2 2D cross sectional view of adjustment mechanism configuration

A previous adjustment mechanism, shown in Figure 3, was presented to the team by the Harris sponsor that utilized an angled configuration to pull the cords. This design interfered with adjacent components when stowed away and also required many different adjusters to accommodate varying cord angles along the reflector rib. To solve this problem, Harris altered the configuration of the adjustment mechanism to a parallel pull configuration seen in Figure 4. This permitted the cord pull angle to be independent of the cord orientation, allowing for a more universal adjustment mechanism for various types of mesh reflectors.

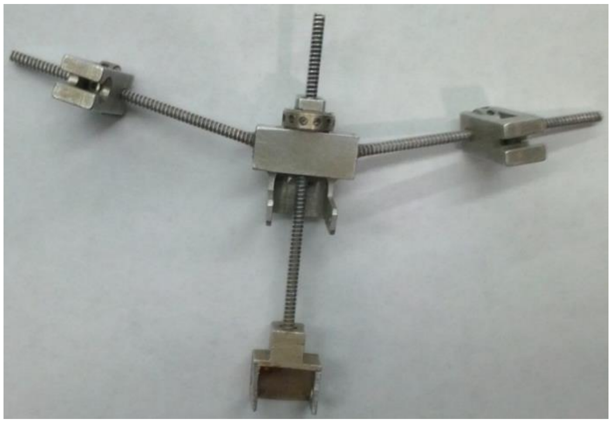


Figure 3 Previous adjustment mechanism with angled configuration

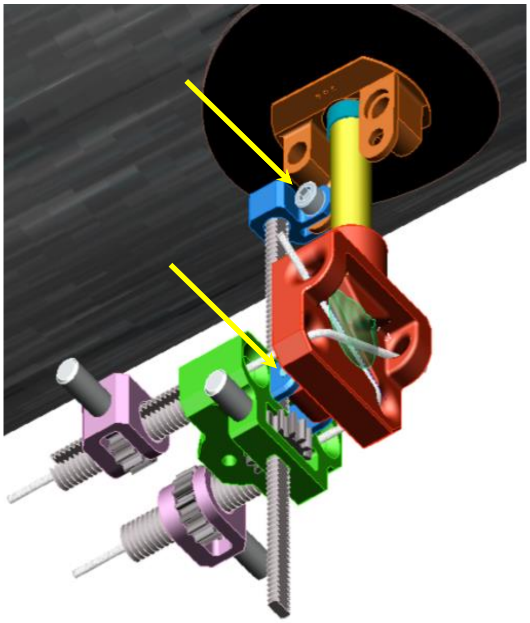


Figure 4 Adjustment mechanism with parallel pull configuration

Project Scope

The project given to team nine by Harris Corporation is to generate the necessary mechanism and control logic to make automatic hands-free adjustments of a reflector surface. With an older adjustment mechanism shown to the team, and CAD drawings provided of a current design, the team is to automate the existing parallel-pull mechanism. The primary goals are to build one high precision mechanism and construct a visual display to demonstrate that the mechanism is capable of high linear resolution as required from Harris. Given that there is a weight tolerance for additional components, hardware may be added to provide wireless capability and an integrated power supply to make the mechanism fully wireless.

Customer Needs

Given the task to build one high precision adjustment mechanism, certain constraints were given by the Harris sponsor that the mechanism must be able to perform within. The linear range for each adjustment location must be 0.100" with a linear resolution of 0.001" and lifespan of 10,000 linear inches. It is desired that each of the adjustment mechanisms be lightweight as possible, preferably under 80 grams and cost $800 per unit. The visual display should have linear displacement sensor such as a linear variable differential transformer (LVDT) or string potentiometer (string pot) to correspond to each adjustment cord, which measures the displacement accurately and displays this measurement on an output screen.

Function Analysis

The diagram in Figure 5 shows how the automated adjustment mechanism works, which components are controlled by an input, and how the displacement is measured. It is broken up into 2 subsystems, the adjustment mechanism and the tabletop visual demonstration used to test it. The motor controls the drive system that consists of the internally threaded gears, the 4-40 all thread rods, the cord guide, and cord anchors which are connected to the cord that measures the displacement using a string pot.

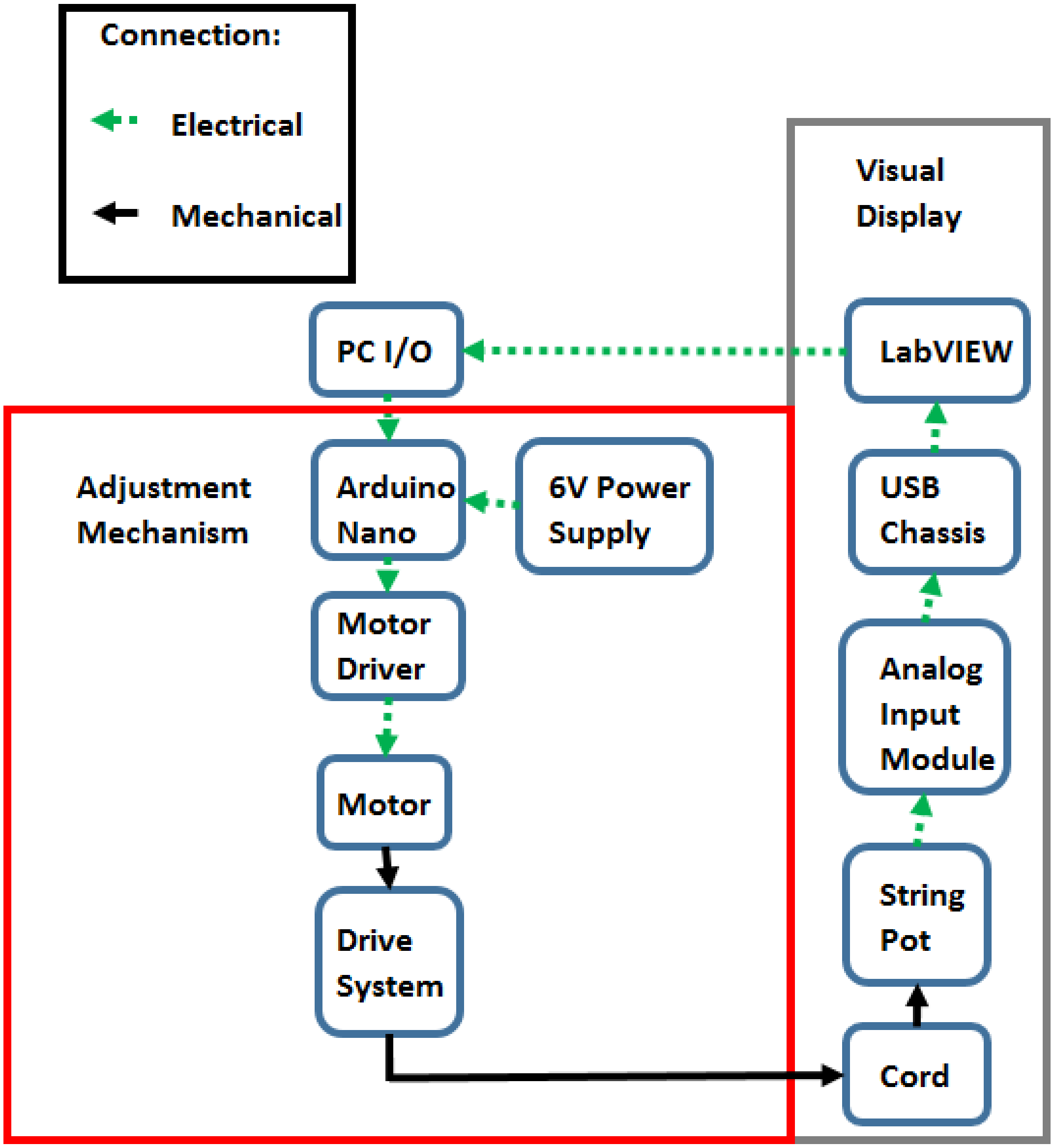


Figure 5 Function Analysis of Adjustment Mechanism

Concept Generation

With the CAD files of the parallel pull mechanism provided by the team's sponsor, the design process of the adjustment mechanism began with redesigning the base of the mechanism with slots to house the motors. The team decided that motors directly mounted to the base would lead to less complications in the future. The finalized design seen in Figure 6 was based on CAD files provided by Harris for dimensions such as spacing between the cords and location for the mount. Not many concept designs were under consideration given the limited configurations available for the motors to be placed.

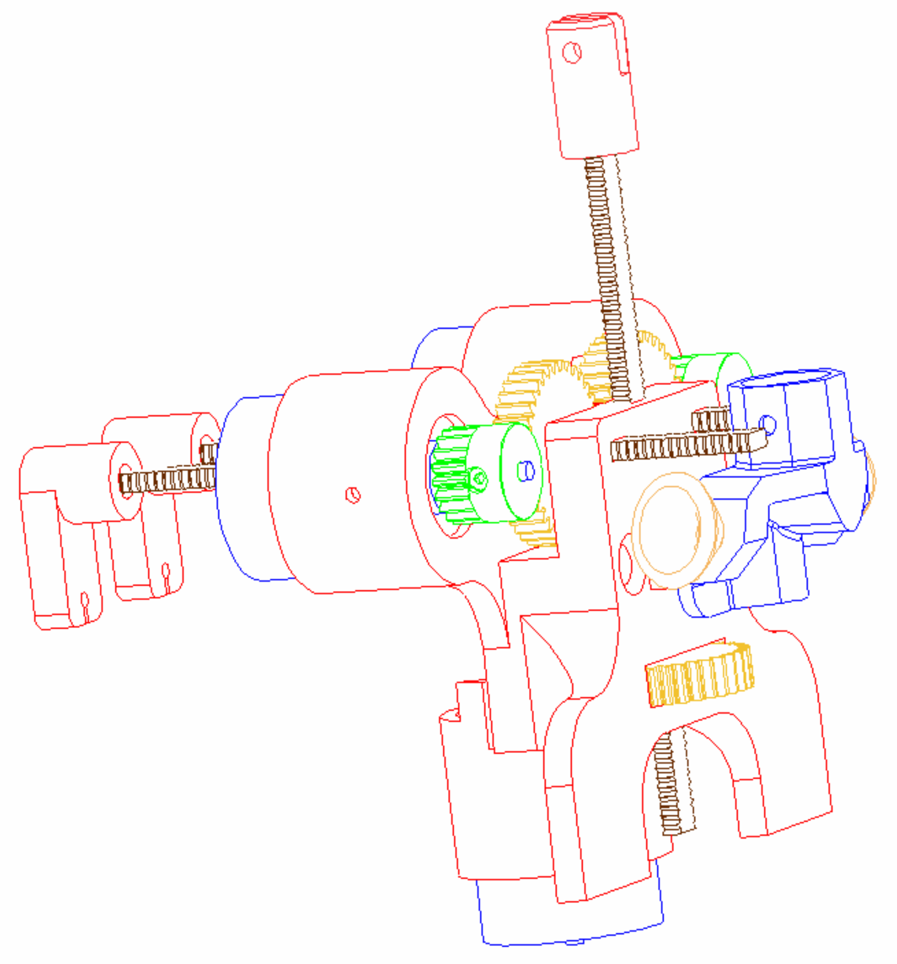


Figure 6 CAD of Automated Adjustment Mechanism

Adjuster Component Selection

Material Selection

Before the mechanism was designed, research was done on the best available materials to construct the mechanism that would provide an economical design. With aluminum and acrylonitrile butadiene styrene (ABS) plastic under careful consideration due to their lightweight properties and availability, it was determined that ABS plastic would be the best material for the primary components of the mechanism. Initial calculations based on the volume of material used from ProEngineer had the Al6061 base weighing 17.33g compared to ABS plastic weighing only 6.53g. ABS plastic is used as the generic 3D printing material, which provided a low density, durable, and cheap material that is used to print the complex structures of the mechanism base. Other components to be fabricated out of ABS plastic are the cord and straw anchors which can be seen better in Figure 18 in Appendix B, which shows an exploded view of the mechanism.

The materials of other essential components for the mechanism to be selected were for the 4-40 all thread rod, gears, and washers. Aluminum was selected for the 4-40 all thread rods because of its lightweight material properties and availability from McMaster-Carr. Premade brass spur gear stock were selected for the drive system that consists of the gear and pinion due to its weight, cost, and durability. The gear stock was lathed down into useable gears for the mechanism as opposed to custom made, high precision gears that would have exceeded the team's budget. Low friction, nylon spacers were added between base and gears in order to reduce wear on the base of the mechanism from the gears. Utilizing these shims also eliminates possibilities of foreign object debris (FOD), a major concern in the aerospace industry, from the gears wearing away the base material.

Materials for the visual demonstration consist of 80/20 aluminum beams for the frame. This material was chosen since it is sturdy which will aid in accurately displaying measurements as well as it is more aesthetically pleasing and offers a professional look. Non-stretch braided fishing line will be used as a substitute for the typical quartz cord used in reflector adjustments. This was done in an effort to reduce cost and decrease lead time for demonstration purposes.

Motor Selection

For this application, a high precision open control system was needed. Stepper motors rotate at specific angular increments, called steps from digital pulses, compared to conventional motors that run continuously when a voltage is applied. Therefore the use of this type of motor was the logical choice for this system because they offer precise rotation control.

In order to select a proper motor for this application, several requirements needed to be addressed namely, torque requirements and linear resolution of adjustments. Utilizing 4-40 all-thread as the heart of the mechanical adjustments, a captive geared nut creates translational movements. Using the lead of the thread and a simple proportion equation (Equation 3 in Appendix B), it is determined that a step angle of 14.4 degrees is needed to reach a .001" linear resolution. Torque requirements were calculated using two different equations which are both located in Appendix B, Equation 1 from a professor and Equation 2 was provided by the Harris sponsor. Both of these equations have empirical values, coefficient of friction and nut fitting factor, that lead to a slight degree of uncertainty. Nonetheless, both calculations yielded similar values around five milli-newton-meters, which provides a high level of confidence in their accuracy. Exploring the market as to what is available in micro-stepper motors, it was determined that the best selection would be the Faulhaber AM1524 as seen in Figure 7. Although this motor only outputs six milli-newton-meters of torque and has slightly poor resolution at fifteen degrees per step, it is quite affordable and very lightweight at twelve grams as compared to other micro-steppers on the market. To achieve greater linear resolution and increase the torque output of the motor, a 2:1 gear reduction was used to overcome additional frictional losses in the system.

Figure 7 Faulhaber AM1524 Stepper Motor

Electrical Components

To automate the control of the reflector surface adjustment mechanism, certain electrical components need to be integrated into the assembly. The mechanism that is currently used by Harris has three individual points that can be adjusted, two for the opposing cords and one for the vertical straw. For each of these points, a micro stepper motor is to be employed to allow the ability to actuate the respective cord remotely.

A microcontroller is the main and most essential electrical component to be used to handle various tasks for this project. The selected microcontroller to be utilized by this project is the Arduino Nano seen in Figure 8 due to its lightweight attribute of only nine grams, small size, and open source platform. The advantage of the Arduino environment is that it allows electronic prototyping that is flexible and easy to use with a widespread development community. The language requirements to program an Arduino based system is in C/C++, which all members of the group have familiarity with, instead of having to learn a brand new programming language.

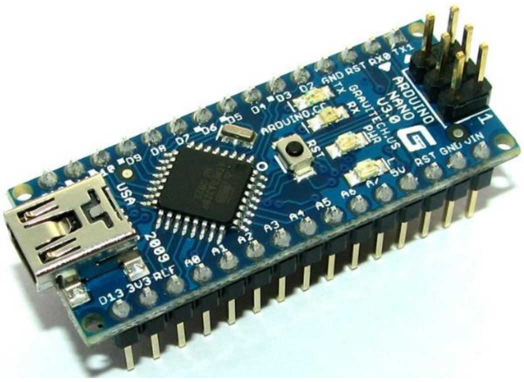


Figure 8 Arduino Nano Microcontroller

Motors are the next important components to actually perform the physical movement at the adjustment points. The type of motor selected are stepper motors to ensure that there is accurate displacement as discussed earlier. On the Arduino microcontroller, the pins can only provide a maximum output current of 40mA, when the necessary current to drive our Faulhaber stepper motor is 150mA. If the motor wires were to be directly connected to the microcontroller, the outputs could potentially get damaged. Also, a stepper motor is not like a common DC motor that will spin when a current is applied. The current has to be applied in a sequence across the four motor wires, along with the polarity to provide direction of the spin. A solution to these problems are to use a H-Bridge motor driver, which provides more available drive current for the stepper and ability to switch current drive polarity. There were two motor driver chips under consideration, the Texas Instrument SN754410 and SGS-Thomson L293D that were similarly priced at ~$3. Both have the same pin layout, so they are easily interchangeable with each other without any hardware or code changes. Also, both have a built in flyback diodes to protect the driver that minimize inductive voltage spikes when the current changed too quickly. The only difference is that the SN754410 can provide a current of 1A compared to the 0.6A from the L293D. For the scope of this project, either motor drive can be used, but the SN754410 seen in Figure 9 was selected due higher current output and availability from professor. The L293D can be used as a backup if the SN754410 does not function as intended. With the addition of an H-Bridge, the lifespan of the Arduino microcontroller will be increased with protection from voltage spikes.

Figure 9 TI SN754410 Motor Driver

For the testing phase of the electrical components, the Arduino Nano, motor driver chips, and motors will be put onto a breadboard to be programmed seen in Figure 10. For each stepper motor, there is a corresponding motor driver chip to amplify the current.

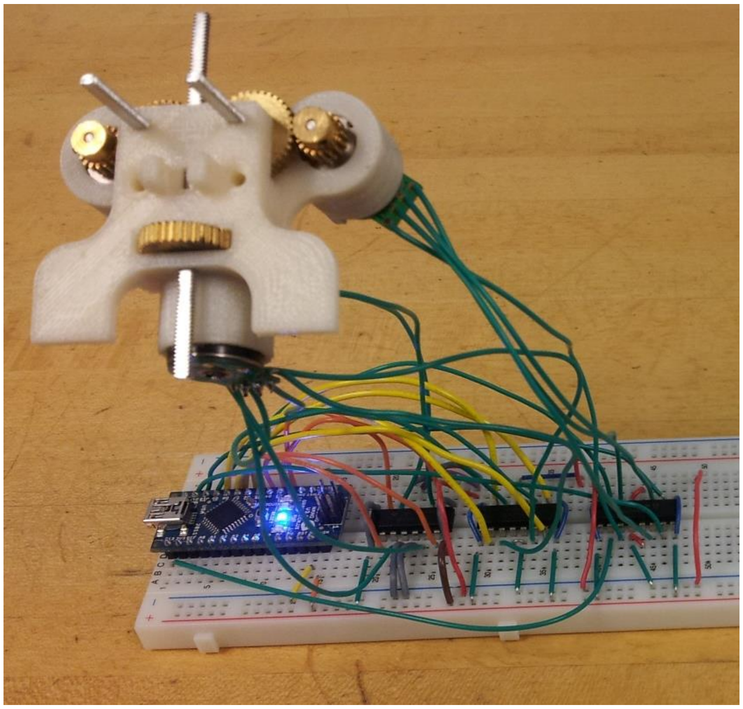


Figure 10 Initial Electrical Testing Setup on Breadboard

Once the programming and electrical setup was finalized, the components were soldered onto a proto board to compact the electrical systems and further reduce the weight seen in Figure 11. Figure 11 below shows the basic layout of the electronics system to compact them. The motor driver chips inputs are connected to the digital I/O pins of the Arduino. The motors are then connected to the corresponding output pins of the motor driver chips. The mini USB port is used to directly connect the microcontroller to a computer USB in order to use the software interface for input adjustments. With these components, only seven pins will be used out of the thirteen available on the Nano, which leaves four pins for Wi-Fi to be implemented in the future for a possible future project.

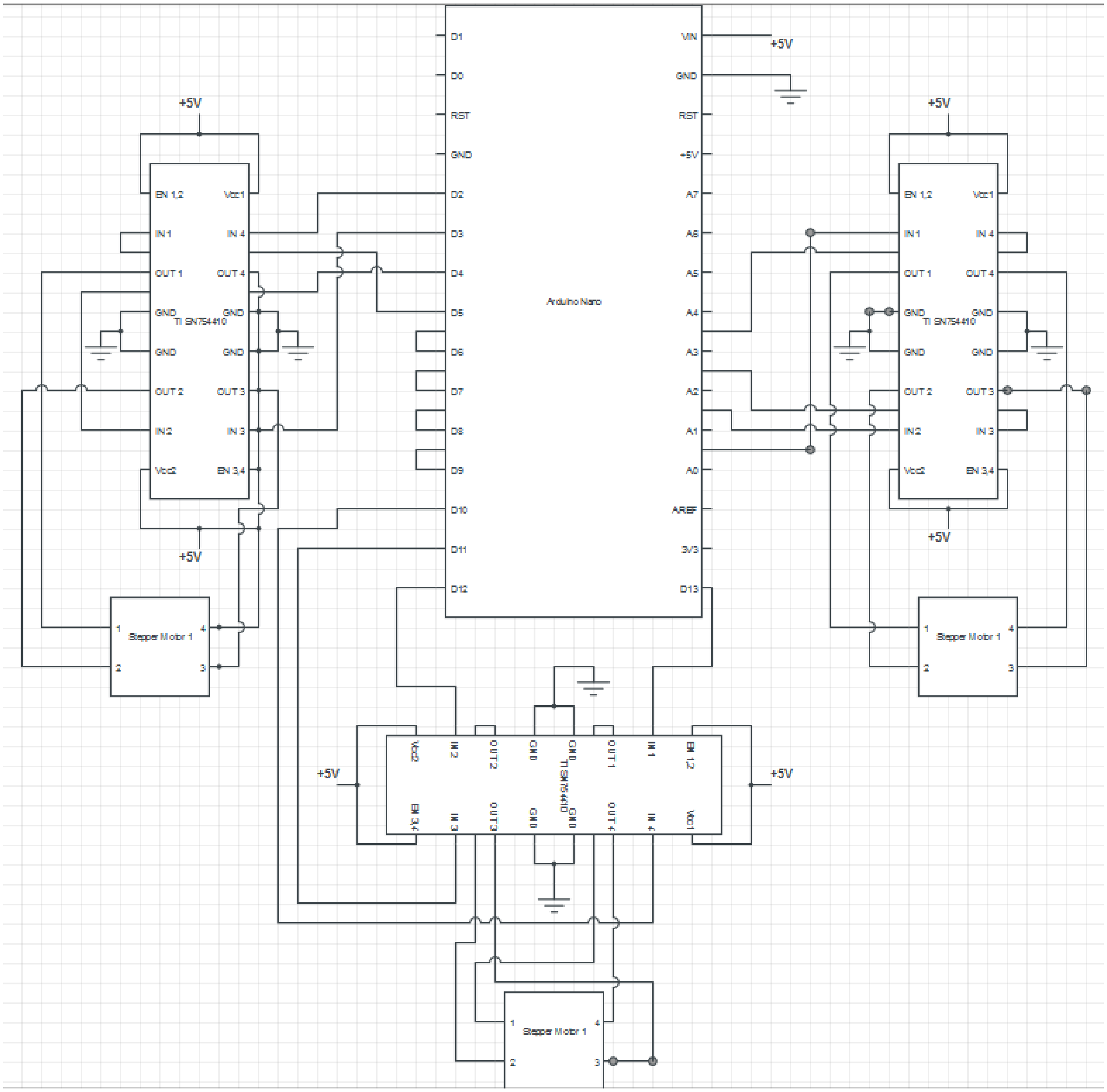


Figure 11 Electrical Systems Schematic for Adjustment Mechanism

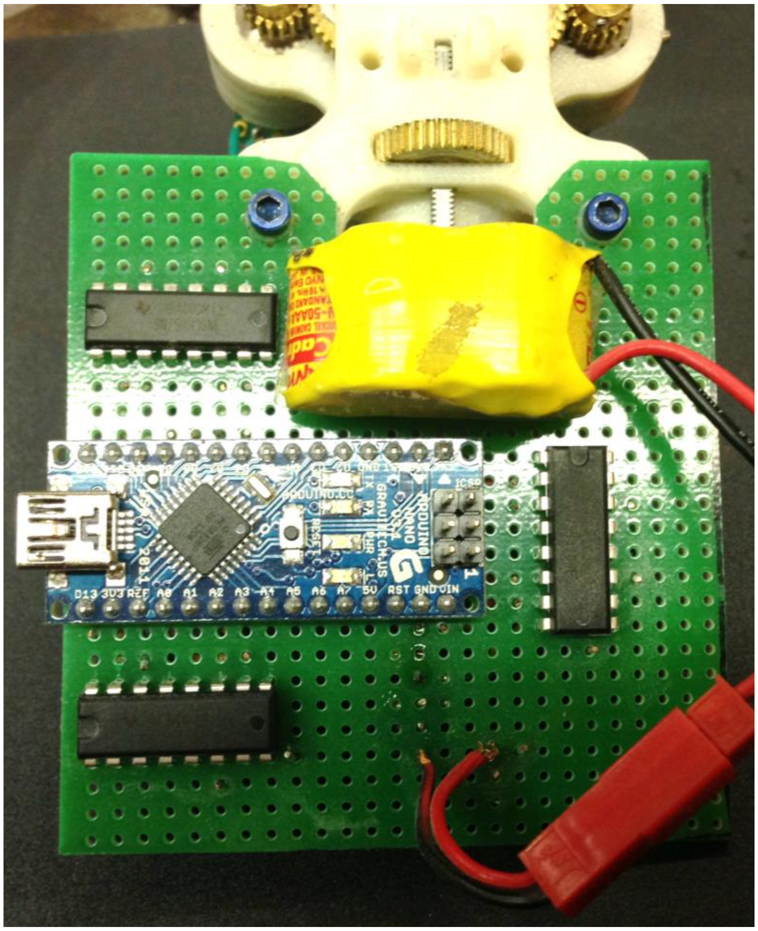


Figure 12 Electrical Setup on Perma-Proto Board

The primary goal was to get the motors connected to the microcontroller then be able to control them remotely. Secondary goals are to implement wireless capability and have an integrated power supply given that there is an allowable tolerance for the extra weight. Given the time constraint of the project, only the task of integrating a power supply was complete. For the future, an Arduino Wi-Fi Shield may be connected to the Arduino Nano to give the mechanism a wireless connectivity that can connect to a standard 802.11b/g wireless network. A nickel cadmium battery supply was used due to the ability to recharge, low-cost, the small size, and its high energy density (power to weight ratio).

Manufacturing

In order to create a prototype of the adjustment mechanism in the most efficient manner, the team had to consider various manufacturing processes when designing the mechanism. Considerations were taken to help reduce the overall production cost, minimize the complexity of the manufacturing, and acknowledge that there was a limit on product tolerances by the necessary manufacturing processes.

Manufacturing Process

With the complex shape and curves in the geometry of the adjustment mechanism base, a specialized process would be necessary to create the base. So in our case, the 3D printer was ideal due to the material that it printed with, along with the available resource of a Dimension SST 1200 3D printer located on the campus.

Other components on the mechanism to be machined were the 4-40 all thread rods that needed to be milled flat on the top and bottom, along with the gear stock that needed to be lathed down into thinner pieces to fit within the mechanism base.

The 80/20 aluminum beams used to build the tabletop visual demonstration for the open house were cut in order to fashion a cubic frame. Once pieces are cut, they will be fastened together using associated L-brackets and t-head bolts and nuts

Manufacturing Location

The machine shop on campus was unsuccessful in the initial attempt to get the 4-40 all thread rods milled flat on the top and bottom of the thread. They encountered many problems that included the thread popping out of the clamps, bending of the threads which gave it an uneven profile, and ultimately broke. After informing the sponsor at Harris of our problem, they offered to machine these parts for us. With allocated funds for the senior design projects with special circumstances, Harris's machine shop was able to provide them at a cost of $100 each.

The gear stock was submitted to the machine shop on campus which lathed it into thinner pieces useable for the mechanism. The base of the mechanism, along with other components were printed on campus as well with the 3D printer.

The visual display will be built at Cameron Duncan's shop at his home, with access to metal band saws, a tungsten inert gas (TIG) welder, and various other machines will allow for adequate and timely construction of the display frame. Brackets for string potentiometers will also be fabricated there and the display will be assembled in its entirety on-site.

Design Changes / Manufacturing Challenges

Upon completion of the initial prototype, it was apparent that some changes in design were need for the unit to perform optimally. The passage that runs longitudinally through the base for the straw adjustment needed to have an access hole to allow for the solution to dissolve the 3D printer support material. Also, resolution of printed parts were quite poor due to low node count of STL file. The number of elements in the mesh was increased in order to have finer resolution. Holes were undersized on the first prototype, which led to a need for slightly oversized holes in the second prototype. Gear pockets were also widened in order to accommodate thrust washers. Provisions for mounting the electronics were added to the lower portion of the mechanism base.

Other manufacturing challenges as mentioned previously, include the machining of the 4-40 all-thread rods from FAMU-FSU College of Engineering machine shop. Due to the small size of the rods, the machine shop’s clamps were unable to securely hold them for machining which resulted in the rods popping out during machining and becoming mangled and bent. This issue was resolved by contacting Harris Corporation who had machined similar pieces in the past with great success, and the company was gracious enough to machine these for the team successfully and with great precision.

Although not a huge issue, the gears which were machine by the college, varied in width by ±0.01” which made shimming of the gear somewhat of a necessity to ensure proper gear alignment with the pinions.

Reliability

To prolong the lifespan of the adjustment mechanism prototype, special attention is necessary to observe whether there is any wear to any of the essential components. Hazards to be aware of can be caused by basic functions of the mechanism from any of the moving parts. A few problems will be listed, but the operations manual will go in depth with more problems and solutions on how to solve them. The guide holes for the 4-40 thread may eventually wear away from the piece translating back and forth. This may have to do with the material used for the base of the mechanism. If observed, the base should be replaced, preferably with a stronger ABS plastic material that can withstand more cycles. To avoid wear of the base from the gears, low friction nylon washers were implemented. Over time, if the washers are noticeably degrading, it is recommended that they be replaced. Along with the replacement of the washers, the drive system that consists of the gears, pinions, and threads should be regularly cleaned to ensure that there is no debris that may clog the system, and then lubricated to ensure smooth interaction. Set screws are used to maintain the position of the motors and the pinions to the motor shafts, occasionally they may need to be tightened to ensure consistent results and functionality.

Economics

Since there is a need for over 130 adjustment devices for a single reflector with 7 ribs, keeping the mechanisms cheap and simple is a foremost concern. An example of how this was accomplished, was using a 3D printed main base. When compared to a fully CNC’d piece is much less costly and faster to manufacture. Rather than having a third party manufacture the required gears to the needed specifications, raw gear stock was purchased and machined in house to match the application needs, dramatically reducing cost per unit and avoiding costly re-tooling fees and expenses. Using the Arduino platform was also a cost-saving move, as these units are quite affordable and provide sufficient control functions to run the adjustment mechanism.

Data Acquisition Hardware

Linear Displacement Sensor Selection

To measure the displacement that the adjustment mechanism is capable of, multiple linear displacement sensors with high resolution were necessary for accurate measurement. The main criteria for the potentiometer purchase from the sponsor was that it must have a resolution capability of 0.0005”. The school did not have the necessary equipment that the team needed to measure such a small resolution of only ±0.1in with a resolution of 0.001in. To find the best option for this project, research was done for the best possible potentiometer that the team could purchase with factors under consideration such as cost and resolution. After extensive research, the search was narrowed down to string pots or LVDTs. In the end, the Celesco M150 string pot was selected due to the low price of $400 each, 13% of the 1.5” full-stroke utilization and most importantly the variable measuring angle. The LVDT would have given higher resolution for displacement, but the cost would have been double of the string pot and it was only uniaxial in measurement direction which would have caused inconsistent measurements when mounted.

Figure 13 Celesco M150 String Potentiometer

Data Acquisition

After the string pots were selected for the displacement measurement devices, data acquisition hardware was necessary to capture the data. With no data acquisition hardware available for the team to use, a proposal was written to the mechanical engineering department to purchase equipment for the team that would also benefit senior design teams in the future. The most important criteria for the DAQ equipment for the team was that it should be 16-bit analog to provide the necessary resolution and able to read 10V from the string pot. With the proposal accepted, a National Instruments (NI) cDAQ-9174 4-slotted USB chassis seen in Figure 14 and NI 9205 analog input module in Figure 15 that the string pots hookup to were purchased. The chassis would house the analog input module, then can be connected to a computer via USB. With all the connections made, the analog input signal from the potentiometers could be converted to a displacement using a simple formula shown in Equation 6 and displayed on screen through LabVIEW.

Figure 14 National Instruments cDAQ-9174 4-slotted USB chassis

Figure 15 National Instruments 9205 Analog Input Module

Visual Demonstration Setup

To demonstrate the prototype of the adjustment mechanism, a table-top visual will be constructed. The visual setup will consist of a fully-functioning automated adjustment mechanism, housed in an 80/20 structural aluminum stand with mounts for string pots as seen in the center of Figure 16. Adjustments of the cords and straw can be input through the Arduino serial monitor on the left side of Figure 17 and the displacement can be monitored via a string pot at each adjustment where the linear distance will be output onto a computer screen via data acquisition software such as LabVIEW on the right side of Figure 17. This will allow for demonstration of linear resolution and repeatability of the adjustments being made. The connections for the string pot to the DAQ and code for the LabVIEW program can be found in Appendix C, Figure 20 and Figure 20 respectively.

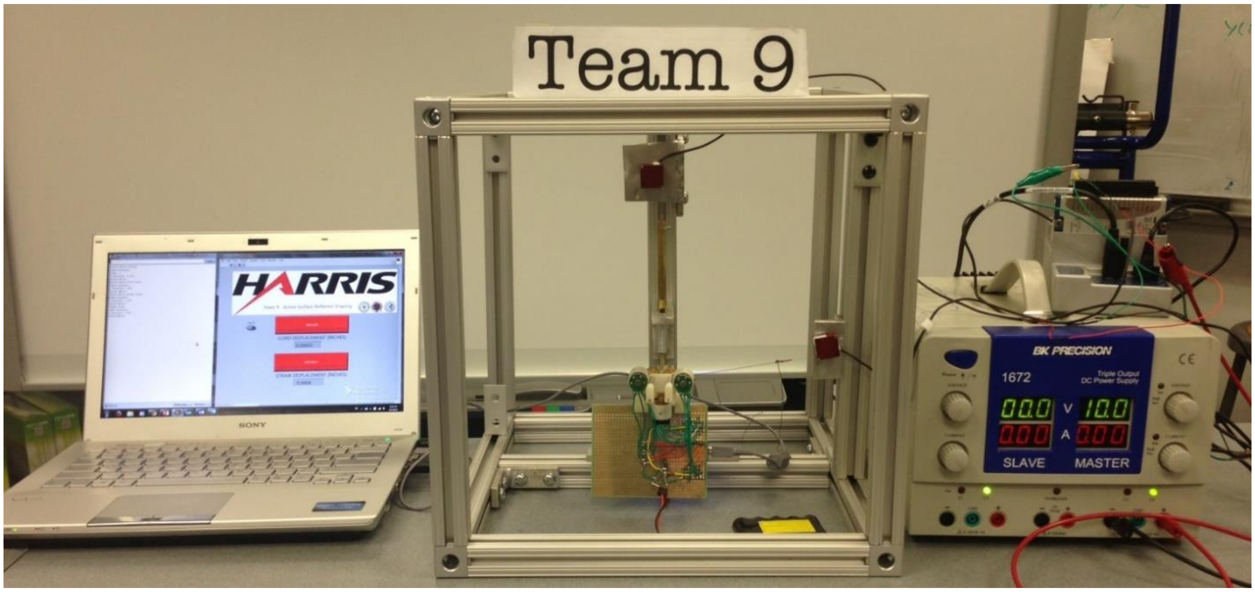
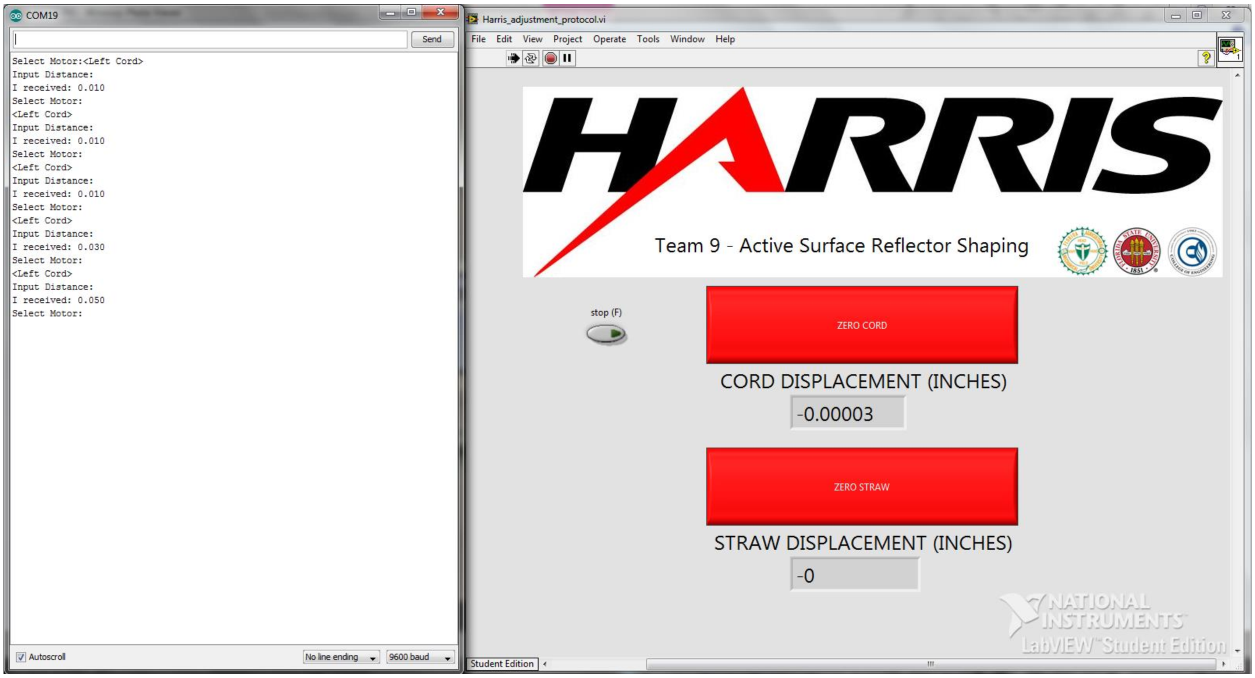


Figure 16 Visual Display Setup



```
Select Motor:<Left Cord>
Input Distance:
I received: 0.010
Select Motor:
<Left Cord>
Input Distance:
I received: 0.010
Select Motor:
<Left Cord>
Input Distance:
I received: 0.010
Select Motor:
<Left Cord>
Input Distance:
I received: 0.030
Select Motor:
<Left Cord>
Input Distance:
I received: 0.050
Select Motor:
```

CORD DISPLACEMENT (INCHES) -0.00003

STRAW DISPLACEMENT (INCHES) -0

Figure 17 Software Interface - Arduino Serial Monitor and LabVIEW Output

Bill of Materials

Table 1 below shows the budget expenditure of team 9, which is broken up into two sections: the adjustment mechanism and the visual display which cost $500 and $1000 respectively. It lists what components were purchased, the purpose, unit cost, total cost, and where they were purchased from are listed. The team was given $2,500 from Harris to build an adjustment mechanism and a testing platform to measure the accuracy of the displacement, of which only $1,500 was spent on the entire project. It should be noted that services such as the 3D printing located on campus, the 4-40 all thread rods that were milled flat from Harris, and the data acquisition hardware purchased by the school were not included in this budget and this table should be used as a rough estimate.

Component	Purpose	Quantity	Unit Cost	Total Cost	Vendor
Adjustment Mechanism					
Arduino Nano	Microcontroller	1	42.79	42.79	Digikey
Faulhaber AM1524	Stepper motor	3	117.6	352.8	Micromo
Texas Insturments SN754410	Motor driver chip	2	2.35	4.7	Digikey
4-40 all thread	Gearing system	1	3.13	3.13	McMaster
18 Teeth, 64 D.P., 20° Pressure Angle, 1' Foot Long Brass Pinion Wire Stock.	Pinion	1	23.73	23.73	SDP/SI
36 Teeth, 64 D.P., 20° Pressure Angle, 1' Foot Long Brass Pinion Wire Stock.	Gear	1	50.71	50.71	SDP/SI
400mAh Polymer Lithium Ion Battery	Power supply	1	7.95	7.95	Sparkfun
Texas Insturments SN754410	Motor driver chip	4	2.35	9.4	Sparkfun
Perma Proto Board	Circuit Board	1	5	5	Fouraker
Visual Demonstration/Testing Platform					
Extrusions for Aluminum T-Slotted Framing	Visual Demonstration frame			209.87	McMaster
Celesco M150	String Pot	2	358.2	716.4	Celesco

LabVIEW Student Edition	Data acquisition software	1	59.95	59.95	Studica
		Total Budget			
		Cost of Adjustment Mechanism		500.21	
		Cost of Visual Demonstration		986.22	
		Budget Available		1013.57	

Table 1 Budget Expenditure

Results and Conclusion

The project given to team nine by Harris Corporation is to generate the necessary mechanism and control logic to make automatic hands-free adjustments of a reflector surface. The current process is very time consuming with the possibility of thousands of adjustment manually until the desired surface geometry of the reflector is achieved. The primary goals were to build one high precision mechanism and construct a visual display to demonstrate that the mechanism is capable of high linear resolution as required from Harris. This was be done by employing three stepper motors at each of the adjustment locations and controlled with a microcontroller.

For the visual demonstration, a structural frame made out of 80-20 aluminum was constructed to house the mechanism. With the mechanism mounted to the structure, string pots were mounted on the structure to measure displacement corresponding to each individual cord.

After much testing, it was observed that the resolution of the adjuster mechanism developed was within ±0.003", slightly outside the Harris preferred envelope of ±0.001". It was determined that this diminished resolution is due to several contributors. The largest contributor is most likely the low-precision gears, which had to be used due to the expensive nature of custom made gears. Another issue was the low step count (24 steps) motors, since the motors needed to be lightweight and small these steppers were the best choice. Another factor in the poor resolution could also be the base tolerances and some slight friction between the sliding thread and the base. The final adjuster mechanism weighed in just over 99 grams, slightly above Harris' preferred weight of 80g.

Future Work

Possible improvements for the automated adjustment prototype mechanism or possibly future projects include having custom gears made of delrin, this would reduce weight and increase precision. This would be more feasible with high production quantities and there would most-likely be a price break associated. Also using higher step count motors would allow for finer adjustments and great overall resolution. Printing the base structure out with a higher strength material such as Taulman 645 on a higher fidelity printer can give greater life to the mechanism and more precision with tighter tolerances.

To address the weight of the mechanism further, several items could be changed slightly to reduce weight. First and foremost, replacing the brass gears with delrin gears would decrease overall weight dramatically. Other ways to reduce the weight would be to have a custom circuit board printed to house all the electrical components and create a more compact package. Also hollowing out areas that are not structurally-critical can also reduce weight.

It would be advantageous for future iterations of this project to focus on implementing wireless command capabilities such as Wi-Fi or Bluetooth. Also producing several adjustment mechanisms and proving the control logic to ensure synchronous operation without any channel interference is key to the final implementation of this mechanism. A good way to demonstrate this would be to produce several mechanisms and implement them on a scaled down representative reflector.

References

"Arduino - ArduinoBoardNano." Arduino - ArduinoBoardNano. N.p., n.d. Web. 10 Dec. 2012. <http://arduino.cc/en/Main/ArduinoBoardNano>.

"Celesco Transducer Products, Setting the Standard for Easy and Reliable Linear and Rotational Measurement." *M150 Miniature String Pot*. N.p., n.d. Web. 23 Apr. 2013. <http://www.celesco.com/m150/>.

"Civilian Government Managing One of the World's Most Reliable Communications Networks." Harris Corporation. N.p., n.d. Web. 10 Dec. 2012. <http://harris.com/>.

"Leading Manufacturer of Small Precision Mechanical Components." Precision Mechanical Components Timing Belts Pulleys Spur Gears Couplings Bearings Sprockets Retaining Rings Brakes Helical Gears Clutches Universal Joints. N.p., n.d. Web. 10 Dec. 2012.

"NI 9205." *- 32-Ch ±200 MV to ±10 V, 16-Bit, 250 KS/s Analog Input Module*. N.p., n.d. Web. 23 Apr. 2013. <http://sine.ni.com/nips/cds/view/p/lang/en/nid/208800>.

"NI CDAQ-9174." *NI CompactDAQ 4-Slot USB Chassis*. N.p., n.d. Web. 23 Apr. 2013. <http://sine.ni.com/nips/cds/view/p/lang/en/nid/207535>.

"Product Index Integrated Circuits (ICs) PMIC - MOSFET, Bridge Drivers - External Switch SN754410NE." SN754410NE Texas Instruments. N.p., n.d. Web. 10 Dec. 2012.

"Rush Gears Inc." Rush Gears Inc. N.p., n.d. Web. 10 Dec. 2012. <http://www.rushgears.com/>.

Faulhaber. Stepper Motors 6.0mNm. N.p.: Faulhaber, n.d. Micromo. Web. 10 Dec. 2012. <http://www.faulhaber.com/uploadpk/EN_AM1524_PCS.pdf>.

Jordan, Akeem. "Team 9 - Harris Active Reflector Surface Shaping." Team 9 - Harris Active Reflector Surface Shaping. FSU CoE, n.d. Web. 10 Dec. 2012. <http://eng.fsu.edu/me/senior_design/2013/team9/>.

Appendix A - Calculations

Torque Calculations

$$T_{raise} = \frac{\mathrm{F}d_m}{2}\left(\frac{l+\pi\mu d_m}{\pi d_m-\mu l}\right) = .0374\ \mathrm{in} * \mathrm{lbf} = 4.23\ \mathrm{mN} * \mathrm{m}$$

Equation 1

F = required pull force = 2 lbf

d_m = mean diameter = 0.095 in

μ = coefficient of friction = 0.3

l = lead = # of Starts * Pitch = 0.025 in

Pitch = 1/threads per inch = 0.025 in

$$\mathrm{T} = \mathrm{kFd} = .0448\ \mathrm{in} * \mathrm{lb} = 5.06\ \mathrm{mN} * \mathrm{m}$$

Equation 2

k = fitting factor = 0.2

F = required pull force = 2 lbf

d = diameter = 0.112 in

Gear Calculations

Required step angle from motor to obtain 0.001” resolution:

$$\frac{0.025\ in}{360\ deg} = \frac{0.001\ in}{x} \Rightarrow x = 14.4°$$

Equation 3

Actual step from Faulhaber AM1524 motor:

$$\frac{0.025\ in}{360°} = \frac{x}{15°} \Rightarrow x = \ 0.00104\ in \text{ Linear resolution}$$

Equation 4

Using 2:1 gear ratio:

$$\frac{0.025"}{360°} = \frac{x}{7.5°} \Rightarrow x = 0.000521 in \text{ Linear resolution}$$

Equation 5

Displacement Formula from Input Voltage

$$\mathrm{d} = \left(\frac{1.5\ \mathrm{in}}{10\ \mathrm{V}}\right) * \mathrm{V}$$

Equation 6

$\mathrm{V} = \text{input voltage from string pot}$

Appendix B – Additional Pictures

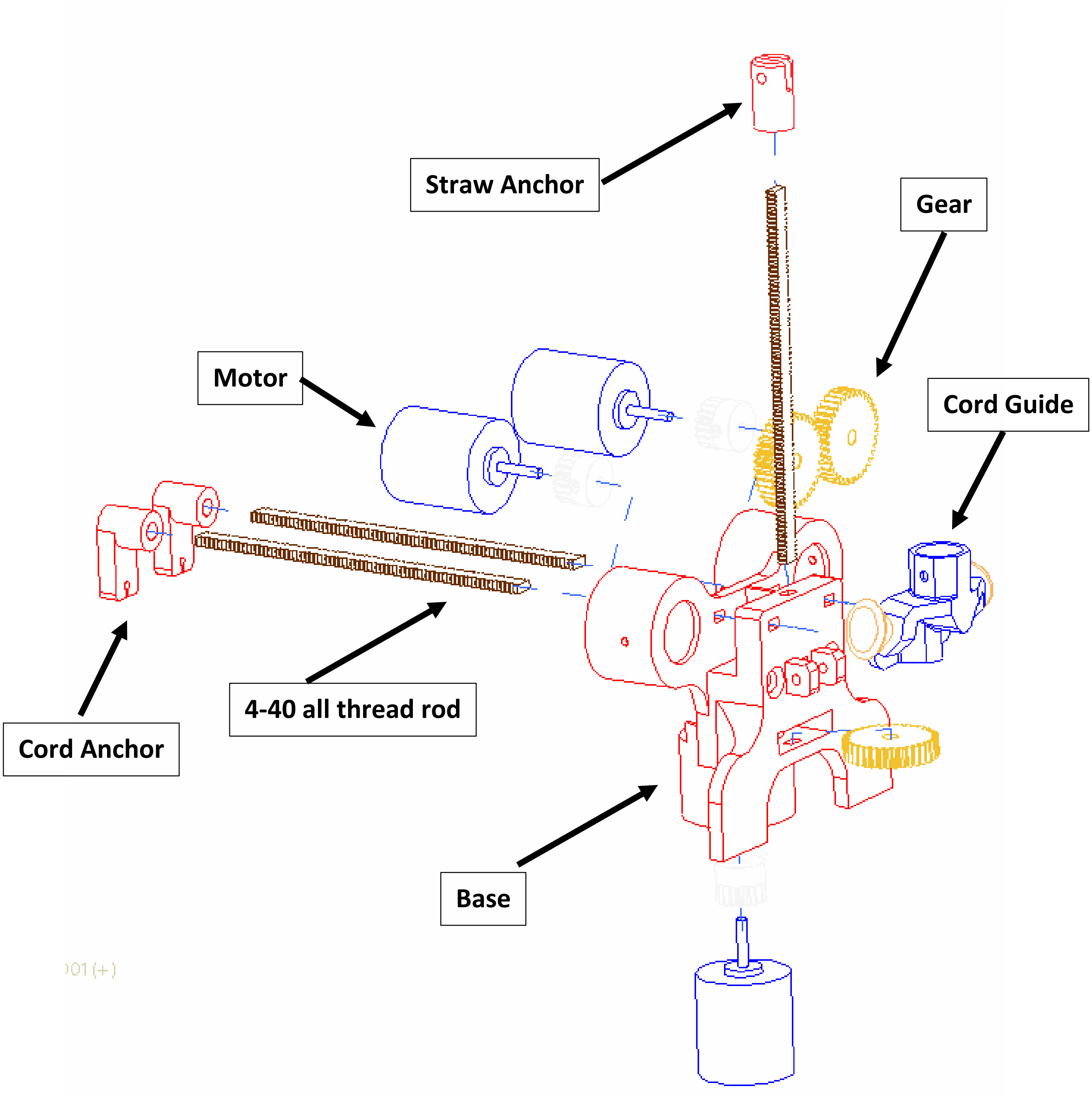


Figure 18 Exploded View of Adjustment Mechanism

Appendix C – DAQ Associated Information

String Pot Hardware Connection

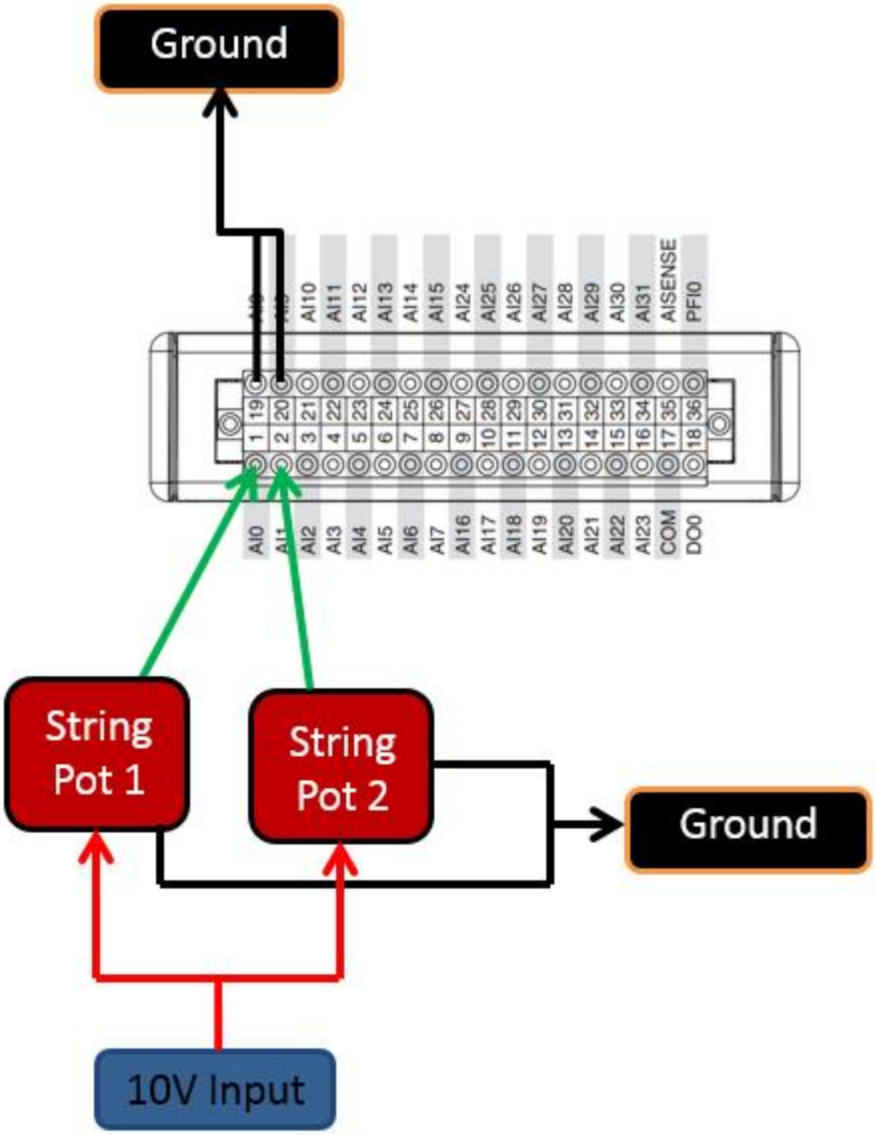


Figure 19 String Pot Connections to DAQ Hardware

LabVIEW Block Diagram

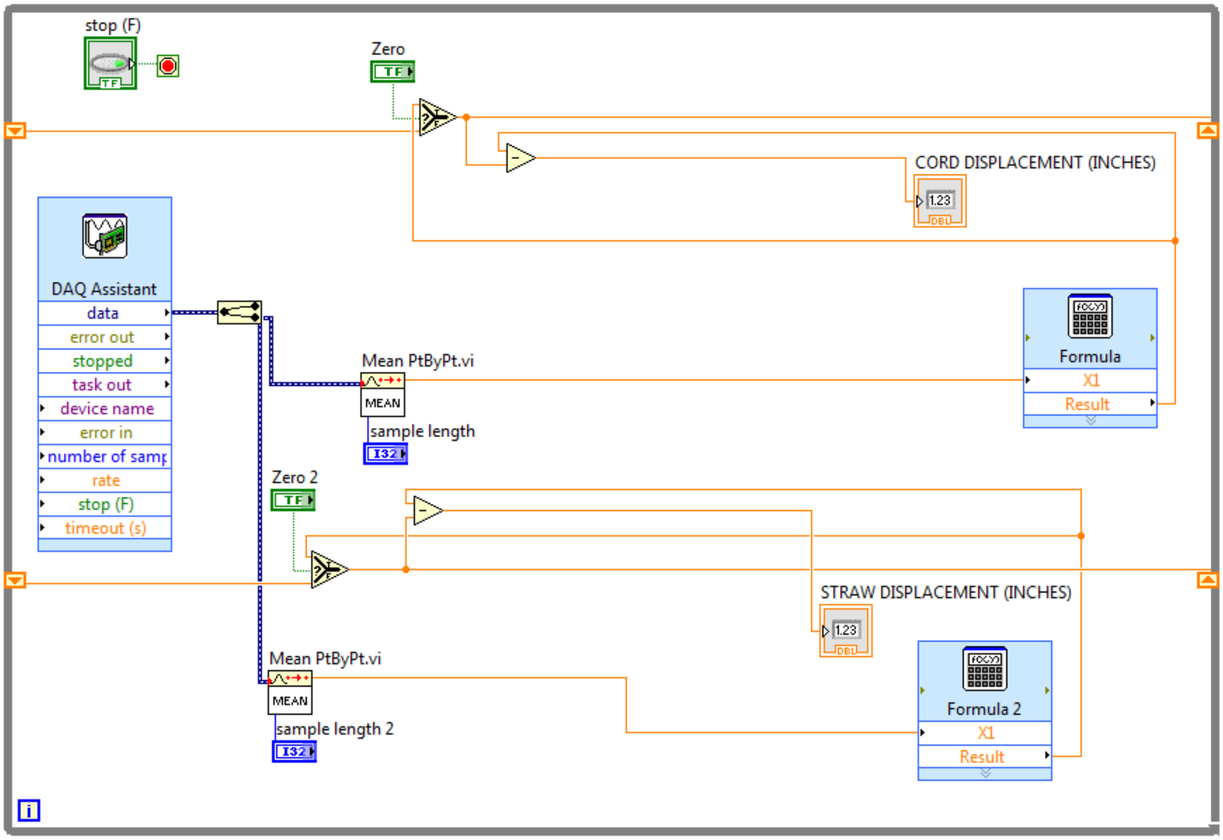


Figure 20 LabVIEW Block Diagram

Arduino Adjustment Code

```
#include <Stepper.h>
```

```
#include <SoftwareSerial.h>

#define motorSteps 24
             //Steps per revolution of motor


#define motorPin1 5
            //motor 1
#define motorPin2 4
#define motorPin3 3
#define motorPin4 2

#define motorPin5 9
            //motor 2
#define motorPin6 8
#define motorPin7 7
#define motorPin8 6

#define motorPin9 13
            //motor 3
#define motorPin10 12
#define motorPin11 11
#define motorPin12 10


byte input;
int sign = 1;
char motor;
int state = 0;
boolean negative = false;
boolean ended = false;



Stepper myStepper1(motorSteps, motorPin1,motorPin2,motorPin3,motorPin4);
//Initialize of the Stepper library
Stepper myStepper2(motorSteps, motorPin5,motorPin6,motorPin7,motorPin8);
Stepper myStepper3(motorSteps, motorPin9,motorPin10,motorPin11,motorPin12);

float num1, num2;
float complNum, negNum, counter;
boolean mySwitch = false;

void setup()
{
  myStepper1.setSpeed(100);
//Set the motor speed
  myStepper2.setSpeed(100);
  myStepper3.setSpeed(100);

  Serial.begin(9600);
//Initialize the Serial port:

  num1=0;
  num2=0;
  complNum=0;
```

```
 Serial.print("Select Motor:");
//Request motor selection
 }

void loop()
{
      char motor = (char)Serial.read();
//Incoming character designates motor selection


       if (motor == 'l')
//Selects left cord motor
       {
         Serial.println("<Left Cord>");
         Serial.println("Input Distance: ");
         delay(50);

         while (motor == 'l')
         {
           while (Serial.available())
    {
    input = Serial.read();
//Incoming serial input is the desired linear translational distance

    if (input > 47 && input < 58)
//If incoming ASCII values are between 47 & 48 take them in
    {

        if (!mySwitch)
//If decimal point received input is a float
        {
          num1 = (num1*10) + (input -
 48);                                //Take in first number after decimal
point
          delay(100);
        }
        else
        {
          num2 = (num2*10) + (input -
 48);                                //Take in next numbers after decimal
point
          delay(100);
        }
    }
    else if (input == 45)
//If negative sign received mototr steps in opposite direction
        {
          negative = true;
          sign = -1.0;
          delay(100);
        }

         if (input == 61)
//When Equal sign received input is translated into a distance
          {
            complNum = (num1 + (num2/(counter)))*sign;
            Serial.print("I received: ");
```

```
            Serial.println(complNum,3);
            myStepper1.step(((360/0.025)*complNum)/6.18);
            state = 0;
       //Cut power to motor after distance is reached
            num1 = 0;
       //Reset all variables
            num2 = 0;
            complNum = 0;
            sign = 1;
            mySwitch = false;
            motor = 0;

            if (state == 0)
//If state = 0 left cord motor is off
              {
                digitalWrite(motorPin1, LOW);
                digitalWrite(motorPin2, LOW);
                digitalWrite(motorPin3, LOW);
                digitalWrite(motorPin4, LOW);
              }

            Serial.println("Select Motor:");
//Request motor selection

              }
              else if (input == 46)
//If decimal point received myswitch is true
              {
                  mySwitch = true;
              }
           }
        }

    }

    if (motor == 'r')
//Selects right cord motor
       {

         Serial.println("<Right Cord>");
         Serial.println("Input Distance: ");
         delay(50);

         while (motor == 'r')
         {
           while (Serial.available())
         {

          input = Serial.read();
//Incoming serial input is the desired linear translational distance

          if (input > 47 && input < 58)
//If incoming ASCII values are between 47 & 48 take them in
          {
              if (!mySwitch)
//If decimal point received input is a float
              {
```

```
                num1 = (num1*10) + (input -
 48);                                           //Take in first number after
decimal point
                delay(100);
              }
              else
              {
                num2 = (num2*10) + (input -
 48);                                           //Take in next numbers after
decimal point
                delay(100);
              }
          }
          else if (input == 45)
//If negative sign received motor steps in opposite direction
              {
                negative = true;
                sign = -1.0;
                delay(100);
              }

         if (input == 61)
//When Equal sign received input is translated into a distance
          {
            complNum = (num1 + (num2/(counter)))*sign;
            Serial.print("I received: ");
            Serial.println(complNum,3);
            myStepper3.step(((360/0.025)*complNum)/7.5);
            state = 2;
               //Cut power to motor after distance is reached
            num1 = 0;
               //Reset all variables
            num2 = 0;
            complNum = 0;
            sign = 1;
            mySwitch = false;
            motor = 0;

            if (state == 2)
//If state = 2 right cord motor is off
        {
          digitalWrite(motorPin9, LOW);
          digitalWrite(motorPin10, LOW);
          digitalWrite(motorPin11, LOW);
          digitalWrite(motorPin12, LOW);
        }

            Serial.println("Select Motor:");
//Request motor selection

          }
          else if (input == 46)
//If decimal point received myswitch is true
          {
              mySwitch = true;
          }
        }
```

```
       }
    }

    if (motor == 's')
//Selects straw motor
       {

         Serial.println("<Straw>");
         Serial.println("Input Distance: ");
         delay(50);

         while (motor == 's')
         {
           while (Serial.available())
         {

            input = Serial.read();
//Incoming serial input is the desired linear translational distance

            if (input > 47 && input < 58)
//If incoming ASCII values are between 47 & 48 take them in
            {
                if (!mySwitch)
//If decimal point received input is a float
                {
                  num1 = (num1*10) + (input -
 48);                                              //Take in first number
after decimal point
                  delay(100);
                }
                else
                {
                  num2 = (num2*10) + (input -
 48);                                              //Take in next numbers
after decimal point
                  delay(100);
                }
            }
            else if (input == 45)
//If negative sign received motor steps in opposite direction
                {
                  negative = true;
                  sign = -1.0;
                  delay(100);
                }

           if (input == 61)
//When Equal sign received input is translated into a distance
            {
              complNum = (num1 + (num2/(counter)))*sign;
              Serial.print("I received: ");
              Serial.println(complNum,3);
              myStepper2.step(((360/0.025)*complNum)/6.378);
              state = 1;
                   //Cut power to motor after distance is reached
              num1 = 0;
                   //Reset all variables
```

```
              num2 = 0;
              complNum = 0;
              sign = 1;
              mySwitch = false;
              motor = 0;

              if (state == 1)
//If state = 2 right cord motor is off
          {
            digitalWrite(motorPin5, LOW);
            digitalWrite(motorPin6, LOW);
            digitalWrite(motorPin7, LOW);
            digitalWrite(motorPin8, LOW);
          }

              Serial.println("Select Motor:");
//Request motor selection

            }
            else if (input == 46)
//If decimal point received myswitch is true
            {
                mySwitch = true;
            }
           }
         }
    }
}
```