University of Southern Indiana
Pott College of Science, Engineering, and Education
Engineering Department

8600 University Boulevard
Evansville, Indiana 47712

Concentrated Solar

Implementation of Small Scale Parabolic Trough Heliostat

Michael Tunny, Nicholas Wester
ECE 491 Senior Design
May 5, 2022

Approved by: _____ _____
Faculty Advisor: Jenna Kloosterman, Ph.D. Date

Approved by: _____ _____
Department Chair: Paul Kuban, Ph.D. Date

ACKNOWLEDGEMENTS

This project would not have been possible without the knowledge and resources from the University of Southern Indiana. A special thank you is given to Mr. Justin Amos for his time teaching the group how to use many of the machines in the Applied Engineering Center to manufacture the parts necessary for this project. We extend are gratitude to Dr. Todd Nelson, Mr. David Ellert, and Mr. Russel Denton for their support with mechanical engineering challenges throughout this project. We appreciate Dr. Jenna Kloosterman for her time and commitment to the project as an advisor. Mrs. Jamie Curry is also appreciated for her help in purchasing components that were needed during the course of this project. Lastly, we thank our friends and family for their ongoing support.

ABSTRACT

This project aims to create a small scale, low-cost concentrated solar parabolic trough system to further research into renewable energy technologies at the University of Southern Indiana. This system uses a reflective mylar film bonded to a Lexan substrate as a cost-effective solution to traditional glass mirrors. 3D modeling is used to develop a plywood base and parabolic frame. Computer aided manufacturing is used with a computer navigated control router to produce most of the components that are needed. The control system is a novel design that uses light dependent resistors, 3D printing, and an Arduino embedded system to track the sun throughout the day. Mechanical rotation is provided by a stepper motor and worm gearbox. The system was successful in automatically tracking the sun with an average tracking error of $0.655^\circ \pm 0.1^\circ$. The system also had a maximum temperature of 394° F with an average temperature of 290.3° using air as a heat transfer fluid. However, these temperature results only provide a baseline, as a true thermodynamic analysis would need to consider fluid dynamics. Additionally, delamination of the reflective film is expected in this system in the future. More research and experimentation is needed to provide a better solution for bonding of the reflective mylar film to a substrate. This project was successful in providing a platform for other senior design projects in renewable energy systems for the future.

Table of Contents

List of Figures **v**

List of Tables **vii**

1 Introduction **1**

2 Background **2**
2.1 CSP Types . . . 2
2.1.1 Central Tower Systems . . . 2
2.1.2 Parabolic Dish Systems . . . 5
2.1.3 Linear Fresnel Systems . . . 6
2.1.4 Parabolic Trough Systems . . . 7

3 Design Considerations **8**

4 Project Management and Teamwork **13**

5 System Architecture **14**
5.1 Level 0 . . . 14
5.2 Level 1 . . . 15
5.3 Level 2 . . . 16

6 Material Selection **17**

7 3D Modeling **18**
7.1 Iteration 1 . . . 18
7.2 Iteration 2 . . . 19
7.3 Iteration 3 . . . 20

8 CNC and Assembly **23**
8.1 MasterCam . . . 23
8.2 CNC Router . . . 25
8.3 Final Preparations and Paint . . . 26
8.4 Assembly . . . 27

9 Controls **35**
9.1 Hardware . . . 36
9.1.1 Controller . . . 36
9.1.2 Solar Tracking . . . 39
9.1.3 Actuators . . . 45
9.1.4 User Interface . . . 47
9.1.5 Hardware Integration . . . 50
9.2 Programming . . . 54
9.2.1 Configuration . . . 54
9.2.2 Main Loop . . . 55
9.2.3 Manual Mode . . . 57
9.2.4 Rotate . . . 58
9.2.5 E-Stop and Limits . . . 59
9.3 Control Testing . . . 60
9.3.1 Bench Testing . . . 60

9.3.2 Testing Individual Hardware Components 64

10 Results **66**

11 Conclusions and Recommendations **67**
11.1 Lessons Learned 67
11.2 Future Recommendations 67

Appendix **70**
Appendix A - Bill of Material 70
Appendix B - Design Factor Locations 71
Appendix C - Arduino Code 72
Appendix D - Resources 79

List of Figures

1 Final implementation of parabolic trough concentrated solar system 1
2 Example central tower CSP system [1] 2
3 (a) External cavity receiver (b) Cavity receiver [3] 3
4 A diagram of the Rankine cycle [5] 4
5 An example of a parabolic dish CSP system [7] 5
6 An example of a linear Fresnel CSP system [9] 6
7 An example of a parabolic trough CSP system [10] 7
8 Mathematical explanation of parabolic shape [11] 10
9 Differences in parabolic and spherical geometries [12] 11
10 Differences in parabolic focal points [11] 11
11 Gantt chart that shows tasks that needed to be complete 13
12 Level 0 system architecture 14
13 Level 1 system architecture 15
14 Level 2 system architecture 16
15 1st iteration of 3D model 18
16 2nd iteration of 3D model 19
17 3rd iteration of 3D model 20
18 Nested sheet 1 and nested sheet 2 21
19 Nested sheet 3 22
20 MasterCam toolpaths and parameters 23
21 G-code generated for use with CNC router 24
22 CNC router with parts being cut 25
23 Rounded and sanded parts 26
24 Painted parts 27
25 Assembly of the top section of the parabolic trough 28
26 Fitting Lexan substrate to the top section of trough 29
27 One of the assembled base pieces showing the upper base rail 30
28 The partially assembled parabolic trough 31
29 Aluminum bracing added to the top of the trough 31
30 Applying spray adhesive to the trough and film 32
31 Applying mylar film to the trough with squeegee 33
32 Fully assembled system 34
33 Control architecture 35
34 Arduino Mega connected to breadboard with LDRs for testing 36
35 Controller block 37
36 Example of an Adafruit 161 LDR 39
37 Circuit schematic for single LDR 40
38 Solar tracking block 41
39 Top-down view of solar tracking assembly with LDR locations in red 42
40 First iteration of tracking enclosure 42
41 Models of second iteration enclosure components 43
42 Assembled and painted second iteration enclosure with caps installed 44
43 Final solar tracking enclosure mounted on trough 45
44 Actuator block 46
45 3.0 Nm NEMA 23 stepper motor connected to NMRV 80:1 wormdrive gearbox . . . 46
46 Limit switch to prevent over rotation for CCW direction 47
47 User interface block 47
48 Completed user interface in final enclosure 49

49 Technical schematic 50
50 Symbolic schematic 51
51 Entire control system before mounting on parabolic trough 52
52 Entire control system after mounting on parabolic trough 53
53 Main loop flowchart 56
54 Manual mode function flowchart 57
55 Rotate Function flowchart 58
56 Emergency stop interrupt flowchart 59
57 Rotation limit switch function flowchart 59
58 Four LDRs in tubes connected to Arduino with plot on computer 60
59 Test bed to determine value of R_{LDR} connected to Arduino with plot on computer . 61
60 Board with all eight adjustible R_{LDR} resistors 62
61 First iteration tracker connected to Arduino with output of all eight LDRs on plot . 63
62 Majority of control system components connected on breadboard 64
63 Individual hardware test for user interface 65

List of Tables

1 Criteria Weights 8
2 Technology comparisons with geometric mean and normalization values 9
3 Summary of weights and related values 9
4 Final AHP results showing parabolic trough as winner 10
5 Calculations for parabolic geometry 12
6 Arduino pin allocation and wire color 38
7 Test results for individual hardware components and respective Arduino configuration 65
8 Temperature results for parabolic trough April 19th 2022 66
Bill of Materials 71

1 Introduction

Concentrated solar power (CSP) systems concentrate visible light energy from the sun in the form of heat. These types of systems generally require a vast amount of land area and require significant capital to construct. This project aims to solve some of these cost issues by creating a small scale, low cost CSP parabolic trough system. Costs are reduced by using inexpensive and readily available materials such as plywood, plastics, and thin film technology. Reducing the cost of renewable technologies is key in advancing power generation from green energy.

The construction of the system starts with a thin Lexan substrate being used to shape the parabola, while a thin film of aluminized mylar is used to create a reflective surface. As light is reflected in the parabola, it is concentrated in a receiver tube located at the focal point of the parabolic reflector. The parabolic trough system also tracks the sun's movement throughout the day to maintain a high efficiency using a stepper motor, worm gearbox, Arduino Mega microcontroller, and light dependent resistors (LDR).

Figure 1 shows the final implementation of the parabolic trough CSP system.

Figure 1: Final implementation of parabolic trough concentrated solar system

2 Background

There are 4 different types of CSP systems, each with their advantages and disadvantages. However, each system still uses the same basic principles of operation. Light is reflected from a movable mirror called a heliostat. This mirror reflects light to a receiver where heat transfer fluids (HTF) are typically cycled. The receiver is the component of a CSP system that collects the focused solar energy. Receiver types vary greatly depending on the type of heat transfer fluid that is used in a given system. As concentrated heat energy is collected, the heat is transferred into a heat transfer fluid. This heat can then be used to do work using different power cycles or can be stored for later use. These systems can be found all over the globe in arid desert environments, and range in size from kilowatts to gigawatts. Each type of CSP system achieves these requirements in different ways. These are discussed in detail below.

2.1 CSP Types

2.1.1 Central Tower Systems

Figure 2: Example central tower CSP system [1]

Central tower, often referred to as power tower, CSP systems use plane heliostats to reflect light to a central tower. These heliostats are often large and range from 2 to 180 m^2. Large plane mirrors are made from individual smaller mirrors and need to be canted to provide overlapping reflected images on the receiver [2]. Power tower systems reflect light onto a point and require dual axis sun tracking for each heliostat to accomplish this. The light is reflected onto a tower where a receiver is located. Central tower receivers are typically tubular cavity receivers because of the liquid heat transfer fluids that are used. However, other types of receivers are used for air and particle mediums. Figure 3 displays two cavity receiver types used in power towers.

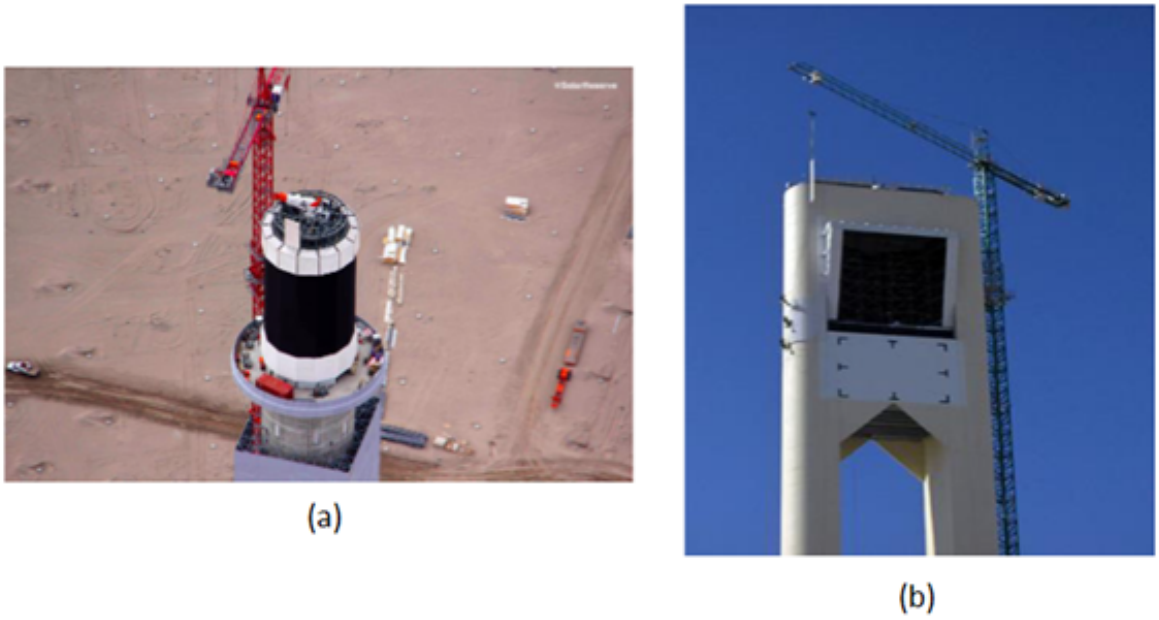

Figure 3: (a) External cavity receiver (b) Cavity receiver [3]

Newer power tower systems use liquid molten salts or sodium mixtures as a HTF, but other systems are experimenting with falling particle or film mediums. These HTFs have desirable thermal characteristics such as thermal efficiencies of greater than 90%. They are also able to be stored directly with minimal thermal losses [4]. Central tower systems that use liquid HTFs operate on the Rankine power cycle [4]. The Rankine cycle uses water vapor to generate work. In the case of modern power generation this vapor is steam, and steam turbines are used to generate the electricity. This cycle can produce more energy at high temperatures. This cycle can be done directly if the HTF is water or indirectly through a heat exchanger such as a boiler for molten salt and sodium HTFs [3].

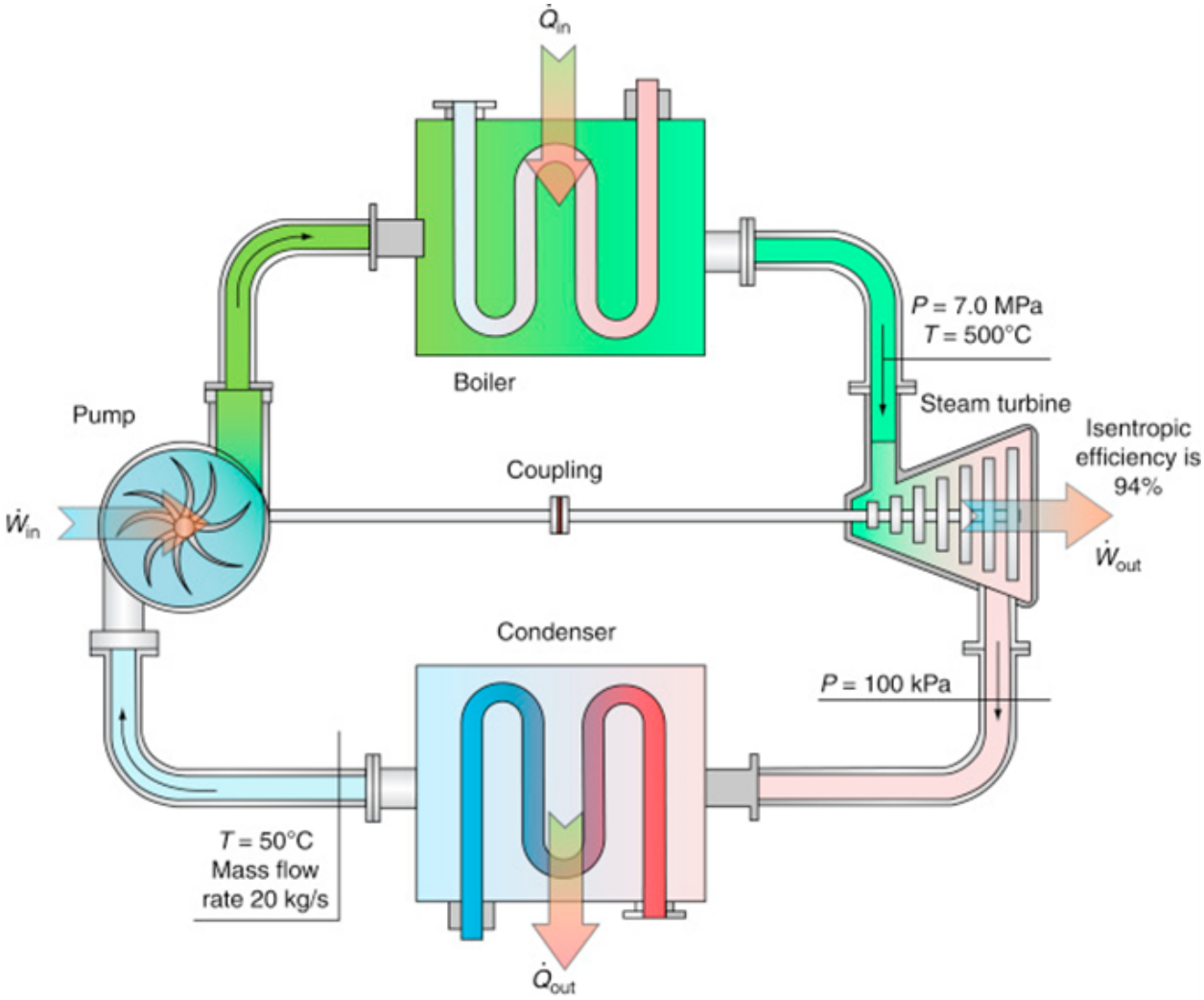

Figure 4: A diagram of the Rankine cycle [5]

These HTFs have great thermal characteristics but are not without their disadvantages. Molten salts require a minimum temperature of 220° C to avoid freezing. This makes molten salts HTFs unusable in lower heat systems. These salts are also corrosive at high temperatures [6].

2.1.2 Parabolic Dish Systems

Figure 5: An example of a parabolic dish CSP system [7]

Parabolic dish CSP systems use a parabolic shaped disc to reflect light into a power conversion unit (PCU). The PCU is located at the focal point of the dish and is comprised of the receiver and a heat engine. These systems more commonly use air, hydrogen, or helium HTFs with a tubular receiver. In rarer occasions fluid HTFs can be boiled and condensed with heat pipes [8]. A sterling engine is usually selected as the heat engine, but there are some micro gas turbine technologies in use today. Parabolic dish systems can achieve much higher temperatures than all other types of CSP technology, though the total thermal output is generally lower due to the size of each dish and lower thermal capacities of the HTFs. Parabolic dish CSP system also require dual axis sun tracking since they are reflecting light to a point.

2.1.3 Linear Fresnel Systems

Figure 6: An example of a linear Fresnel CSP system [9]

Linear Fresnel CSP systems use either flat or slightly curved mirrors to reflect sunlight to a receiver tube located at the focal line. The receiver can be a copper pipe, steel pipe, or evacuated tube. A secondary reflector is needed above the receiver to capture reflected light efficiently. These systems differ from previously mentioned technologies mainly due to only needing single axis tracking, though multiple actuators are still often needed for each linear array of mirrors. These systems typically have lower thermal conversion efficiencies but with the benefit of much lower cost. Linear fresnel systems can use a wide array of HTFs. The most common HTFs for these systems are water or a synthetic oil like biphenyl-dipenyloxide. Synthetic oils have great thermal properties. They are however expensive, flammable, and have negative environmental impacts [6].

2.1.4 Parabolic Trough Systems

Figure 7: An example of a parabolic trough CSP system [10]

Parabolic trough CSP systems use a parabolic shaped mirror to reflect light onto a receiver tube located at the focal line. These mirrors are traditionally constructed of large curved tempered glass with a reflective coating of aluminum or silver. This reflective coating can be bonded to the glass in many different ways including physical vapor deposition, sputter deposition, or thermal evaporation. Because paraboloids focus all wavelengths to the same point, parabolic mirrors have high spectral reflectance, which is the largest factor to overall efficiency in a CSP system. The combination of large, curved mirrors and expensive deposition techniques does come at a high cost. Other mirror technologies have been tested such as reflective thin film technologies on inexpensive substrate materials, but the lifespan of these films is generally less than glass mirrors [2]. The receiver tube can be copper pipe, steel pipe, or an evacuated tube depending on the HTF used and size of the system. Much like the linear Fresnel systems, parabolic troughs only need single axis sun tracking due to the geometry of the parabola. Unlike the linear Fresnel technology, only a single actuator is often needed to move the parabolic trough. Parabolic troughs can use a wide array of HTFs such as air, water, synthetic oils, and molten salts in larger applications.

3 Design Considerations

This project was designed around the ideas of using lower cost and readily available materials. The goals of this project were to create a low cost, small scale CSP system that could be used for future engineering students at USI.

One of the first design considerations was selecting the appropriate CSP technology to meet the project goals. An analytical hierarchy process (AHP) was used to accomplish this selection. First 10 criteria were selected and weighted in importance. Each criteria was compared against each other to develop a weight of importance, then normalized. Table 1 displays the weighted criteria.

	Cost	Efficiency	Technical Complexity	Material Availability	Time to Implement	Testing Time	Mobility	Scalability	Size	Enviromental	GM	**Norm**
Cost	1.00	5.00	5.00	7.00	5.00	7.00	5.00	7.00	7.00	7.00	4.35	**0.31**
Efficiency	0.20	1.00	3.00	5.00	5.00	7.00	3.00	7.00	7.00	7.00	2.87	**0.20**
Technical Complexity	0.33	0.20	1.00	0.33	1.00	1.00	0.20	3.00	3.00	5.00	0.86	**0.06**
Material Availabilty	0.14	0.20	3.00	1.00	5.00	5.00	3.00	7.00	3.00	7.00	1.86	**0.13**
Time to Implement	0.20	0.20	1.00	0.20	1.00	1.00	3.00	3.00	3.00	5.00	1.01	**0.07**
Testing Time	0.14	0.14	1.00	0.20	1.00	1.00	0.33	3.00	3.00	1.00	0.67	**0.05**
Mobility	0.20	0.33	5.00	0.33	0.33	3.00	1.00	3.00	3.00	7.00	1.19	**0.09**
Scalability	0.14	0.14	0.33	0.14	0.33	0.33	0.33	1.00	1.00	3.00	0.44	**0.03**
Size	0.14	0.14	0.33	0.33	0.33	0.33	0.33	1.00	1.00	5.00	0.49	**0.04**
Enviromental	0.14	0.14	0.20	0.14	0.20	1.00	0.14	0.33	0.20	1.00	0.29	**0.02**

Table 1: Criteria Weights

Table 1 shows how each criteria is weighted in importance to what is most critical for the project goals. A higher number correlates to a higher importance. Next each CSP technology was compared together based on these weighted criteria. A geometric mean was found and then each selection was normalized. It should also be noted that the parabolic dish system was not considered due to lower thermal capabilities when compared to other technologies. Table 2 shows the normalized comparisons of each technology that was considered.

Raw Comparisons

	Cost	Efficiency	Technical Complexity	Material Availabilty	Time to Implement	Testing Time	Mobility	Scalability	Size	Enviromental
Tower	7.00	29.00	8.00	7.00	7.00	5.00	1.00	10.00	10.00	5.00
Trough	3.00	17.00	4.00	5.00	5.00	5.00	7.00	5.00	3.00	5.00
Linear	5.00	17.00	6.00	6.00	5.00	5.00	5.00	4.00	8.00	5.00

Geometric Mean

Tower	0.43	1.00	0.50	1.00	0.71	1.00	0.14	1.00	0.30	1.00
Trough	1.00	0.59	1.00	0.71	1.00	1.00	1.00	0.50	1.00	1.00
Linear	0.60	0.59	0.67	0.86	1.00	1.00	0.71	0.40	0.38	1.00

Normal

Tower	0.21	0.46	0.23	0.39	0.26	0.33	0.08	0.53	0.18	0.33
Trough	0.49	0.27	0.46	0.28	0.37	0.33	0.54	0.26	0.60	0.33
Linear	0.30	0.27	0.31	0.33	0.37	0.33	0.38	0.21	0.22	0.33

Table 2: Technology comparisons with geometric mean and normalization values

Lastly the weighted values of each criteria were applied to the normal values found in Table 2. Table 3 shows these weighted importances when compared to each technology. This resulted in the parabolic trough technology winning with 39.74%. Table 4 shows these final results.

	Weight	Tower		Trough		Linear	
Cost	**0.31**	0.21	0.07	0.49	0.15	0.30	0.09
Efficiency	**0.20**	0.46	0.09	0.27	0.06	0.27	0.06
Technical Complexity	**0.06**	0.23	0.01	0.46	0.03	0.31	0.02
Material Availabilty	**0.13**	0.39	0.05	0.28	0.04	0.33	0.04
Time to Implement	**0.07**	0.26	0.02	0.37	0.03	0.37	0.03
Testing Time	**0.05**	0.33	0.02	0.33	0.02	0.33	0.02
Mobility	**0.09**	0.08	0.01	0.54	0.05	0.38	0.03
Scalability	**0.03**	0.53	0.02	0.26	0.01	0.21	0.01
Size	**0.04**	0.18	0.01	0.60	0.02	0.22	0.01
Enviromental	**0.02**	0.33	0.01	0.33	0.01	0.33	0.01

Table 3: Summary of weights and related values

Results	
Tower	29.63%
Trough	39.74%
Linear	30.64%

Winner
Parabloic Trough

Table 4: Final AHP results showing parabolic trough as winner

With the parabolic trough now selected as the CSP technology of choice, the next design consideration was choosing a parabolic shape. "Geometrically, a parabola is a locus of points that lie on equal distance from a line (directrix) and a point" [11]. Mathematically this is represented as Equation 1, and geometrically this is seen in Figure 8.

$$DR = FR \tag{1}$$

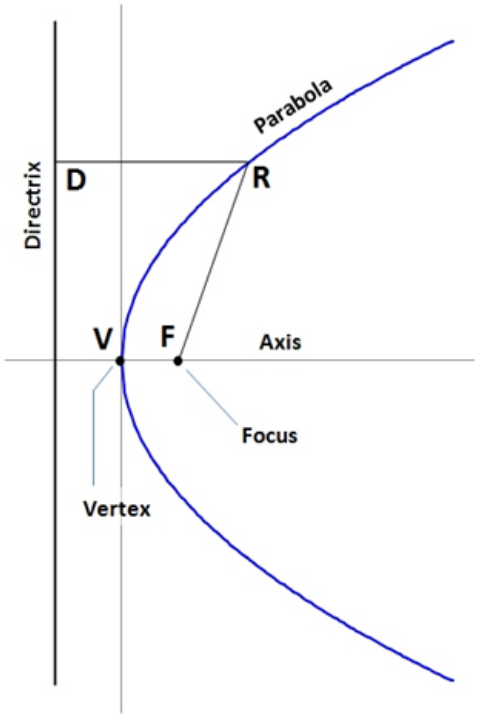

Figure 8: Mathematical explanation of parabolic shape [11]

One design consideration was to decide what geometry should be chosen for the best performance. Physics and mathematics show how parabolas differ from spherical geometries greatly in terms of incident beams of light being reflected to a specific focal line. Parabolic shapes reflect all incident light to a single focal point, while spherical shapes have multiple focal points. Parabolic geometries are vastly superior to spherical geometries in CSP for this reason. Figure 9 displays these differences.

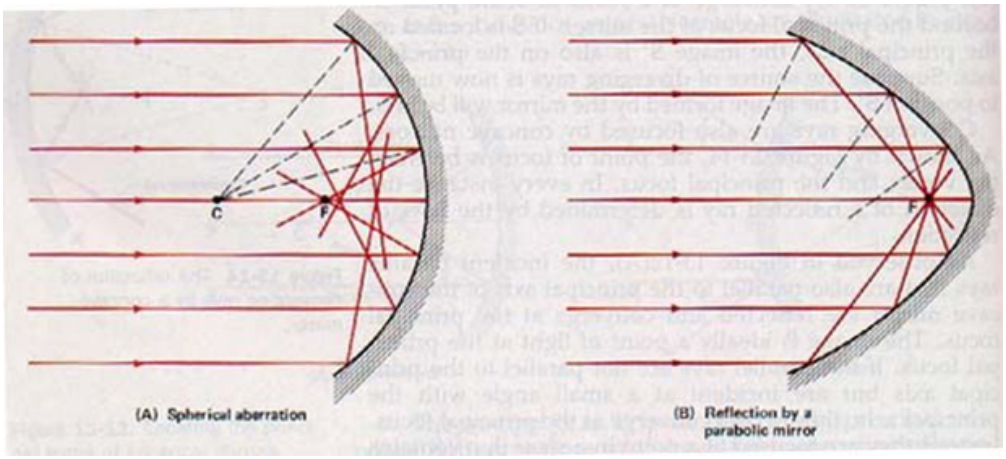

Figure 9: Differences in parabolic and spherical geometries [12]

Designing a spherical mirror would have been much simpler than a parabolic equivalent, but the inefficiency of spherical geometries was much too high to be considered. A parabolic geometry was selected as the mirror shape for this reason. The specific parabolic geometry needed to be defined next. As the width of a parabola widens, the distance to the focal point increases and the rim angle decreases. Wider parabolic shapes have larger areas to collect light in which is an advantage. However, as the width increases energy flux concentration ratio drops and larger receiver tube diameters are needed. There are also increasing mechanical considerations for the center of gravity, framing designs, and dynamic wind loading with wider aperture areas. Figure 10 shows how these parameters are interrelated.

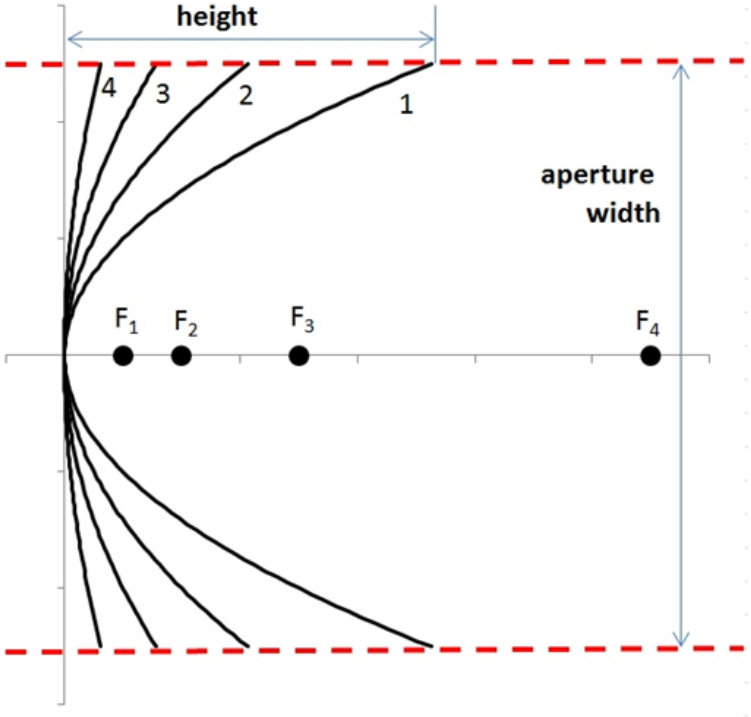

Figure 10: Differences in parabolic focal points [11]

Typically, the rim angle is referred to as the defining parameter of the parabolic shape that other parameters can then be derived from. The equations below define many of the parameters that need to be selected when defining the shape of the parabola for this project.

$$r = \frac{2f}{1 + \cos(\phi_r)} \tag{2}$$

Equation 2 is a formula showing how the radius (r) of the parabola is related to the focal point (f) and rim angle (ϕ_r).

$$D = 2r_r \sin(\theta_m) \tag{3}$$

Equation 3 is a formula showing how the diameter of a receiver pipe (D) is related to the max rim distance (r_r) and half acceptance angle (θ_m)

$$\text{Energy Flux Ratio } = \frac{\text{aperture area}}{\text{focal distance}} \tag{4}$$

Equation 4 is a formula for energy flux ratio of parabola

With all the mechanical, geometric, and energy flux concentration ratio parameters considered, a rim angle of 84° was chosen. The maximum radius of the parabolic shape could be solved mathematically, but a simpler string method was used to define the endpoints of the parabola. A a plot of the parabolic shape was printed, and a string scaled to 48 inches was placed on top of the plot with the midpoint of the string and the midpoint of the parabola matched. Using this method, the endpoints of the parabola were found to be 21.426 inches. This value is useful for finding the total area of the aperture. Using the equations listed above and some other basic algebra, parameters were calculated. These calculations are shown in Table 5

Rim Angle (°)	Area (ft^2)	Receiver Diameter (in)	Max Height (in)	Focal Point (in)	Energy Flux Ratio
84	28.568	0.2	7.72	12	3.571

Table 5: Calculations for parabolic geometry

These parameters define the geometry of the parabolic shape that was designed. Equation 5 also describes the shape of the parabola as a function of x and y points. This equation was used when starting the 3D modeling process.

$$y = \frac{1}{48}x^2, \qquad \text{With Endpoints At } \pm 21.426 \text{ in} \tag{5}$$

4 Project Management and Teamwork

This project was quite large in scope and required a schedule to be completed successfully. The team was comprised of two people. Project management was done through a website called Hive, which uses tools that are very similar to Microsoft Project. Each person created tasks with start dates, end dates, and notes on what needed to be complete via a Gantt chart. Figure 11 shows the project schedule that was developed.

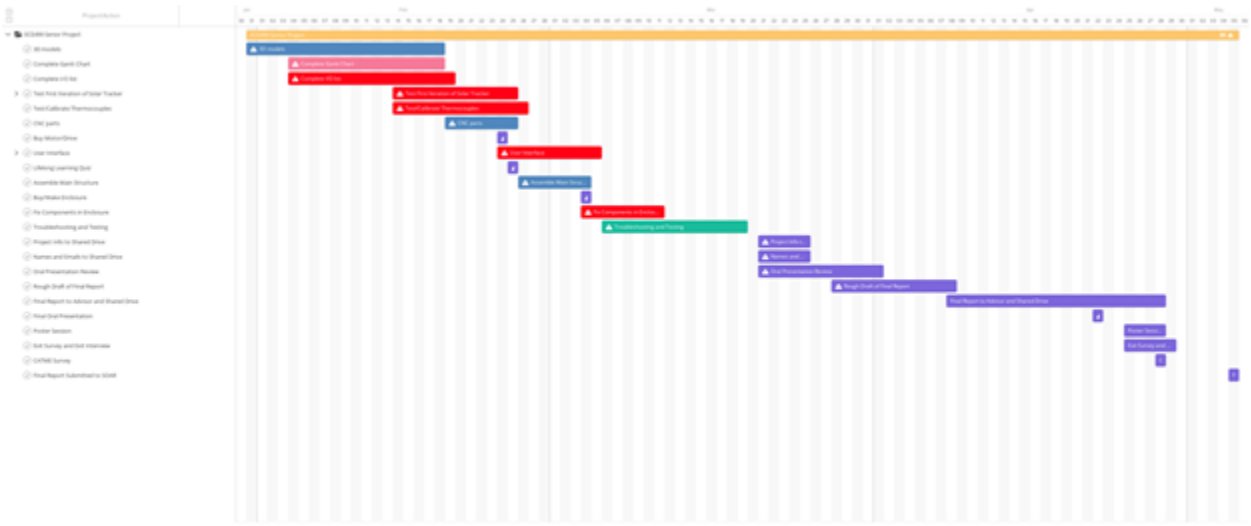

Figure 11: Gantt chart that shows tasks that needed to be complete (blue = modeling and manufacturing, red = controls development, purple = deliverables for ECE 491, green = testing and integration)

A full size version of this chart can be found using the link in Appendix D. This link connects to a shared drive that contains all of the documents, code, modeling, and research that was gathered for this project. This collaborative work environment was created on Onedrive. It was important to have a singular location for files in order to minimize confusion and keep communication clear. Email and phone were also both used for communication as well. The team operated cohesively with only a few disagreements. These disagreements were discussed with an open mind and with the intent of making the best project possible.

5 System Architecture

The system architecture consists of different levels. Level 0 is the most simple, rudimentary level which shows the most basic operation of the system. As the levels increase, so does the level of detail represented by the diagram. The levels stop when the system has been broken down to a component level.

5.1 Level 0

The figure below represents the level 0 system architecture. Here there are two inputs and a single output. For now the inner workings of the block are unknown, but function is known. The block takes a user input along with solar radiation and turns them into thermal energy at the output.

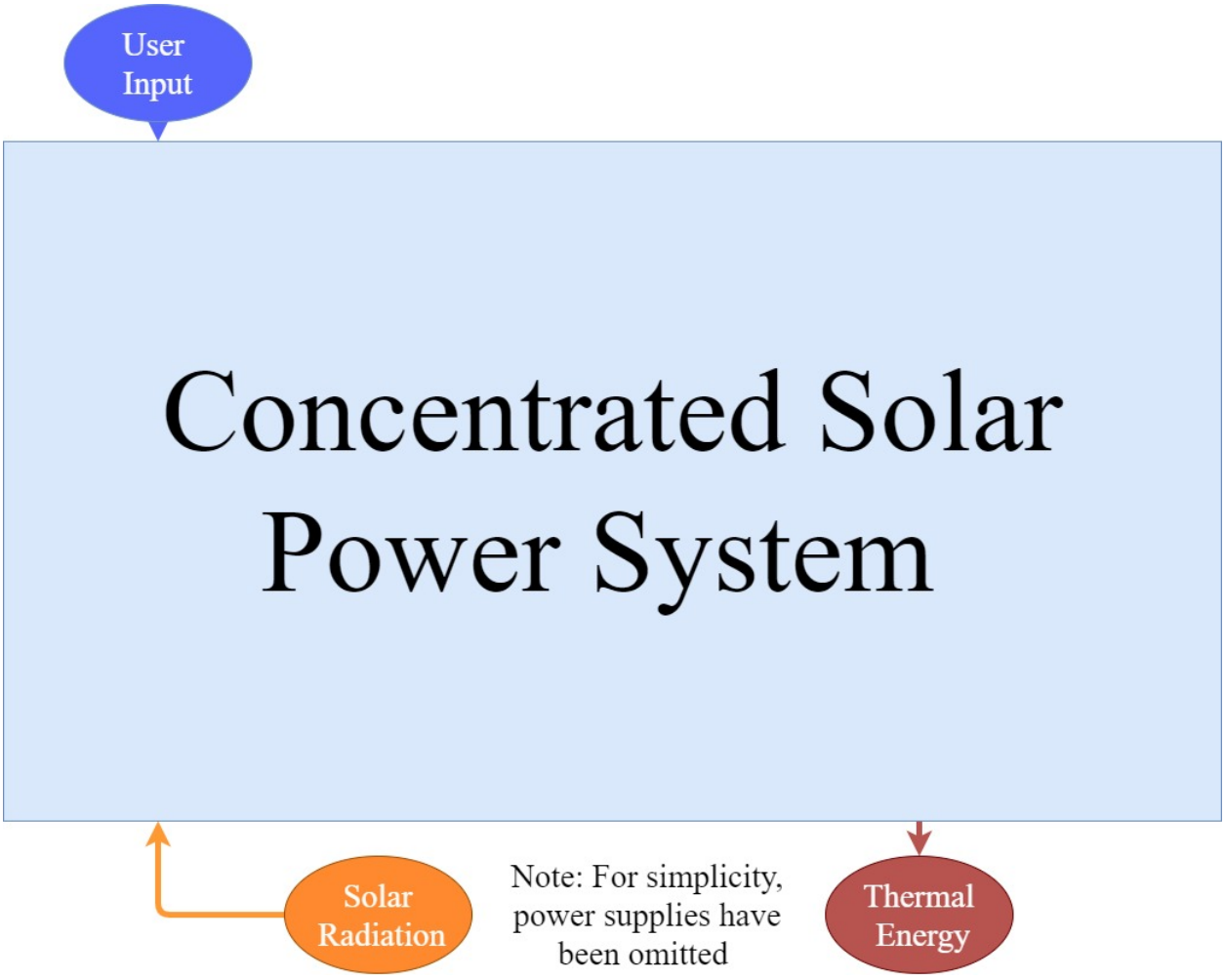

Figure 12: Level 0 system architecture

5.2 Level 1

The next level breaks the previous single block into three distinct blocks that live within the level 0 block. In addition, arrows have been added to represent the flow and communication between the blocks. The control block has four inputs, solar radiation, user input, heliostat data, and storage data. After processing, the control block outputs mechanical movements to the heliostat and storage blocks. The heliostat block has three inputs, solar radiation, mechanical movement, and HTF. This block converts these inputs to output data and HTF. Finally, the storage block takes mechanical movement and HTF as inputs and outputs data, HTF, and thermal energy which is our main system output. Below is a figure of this level 1 system architecture.

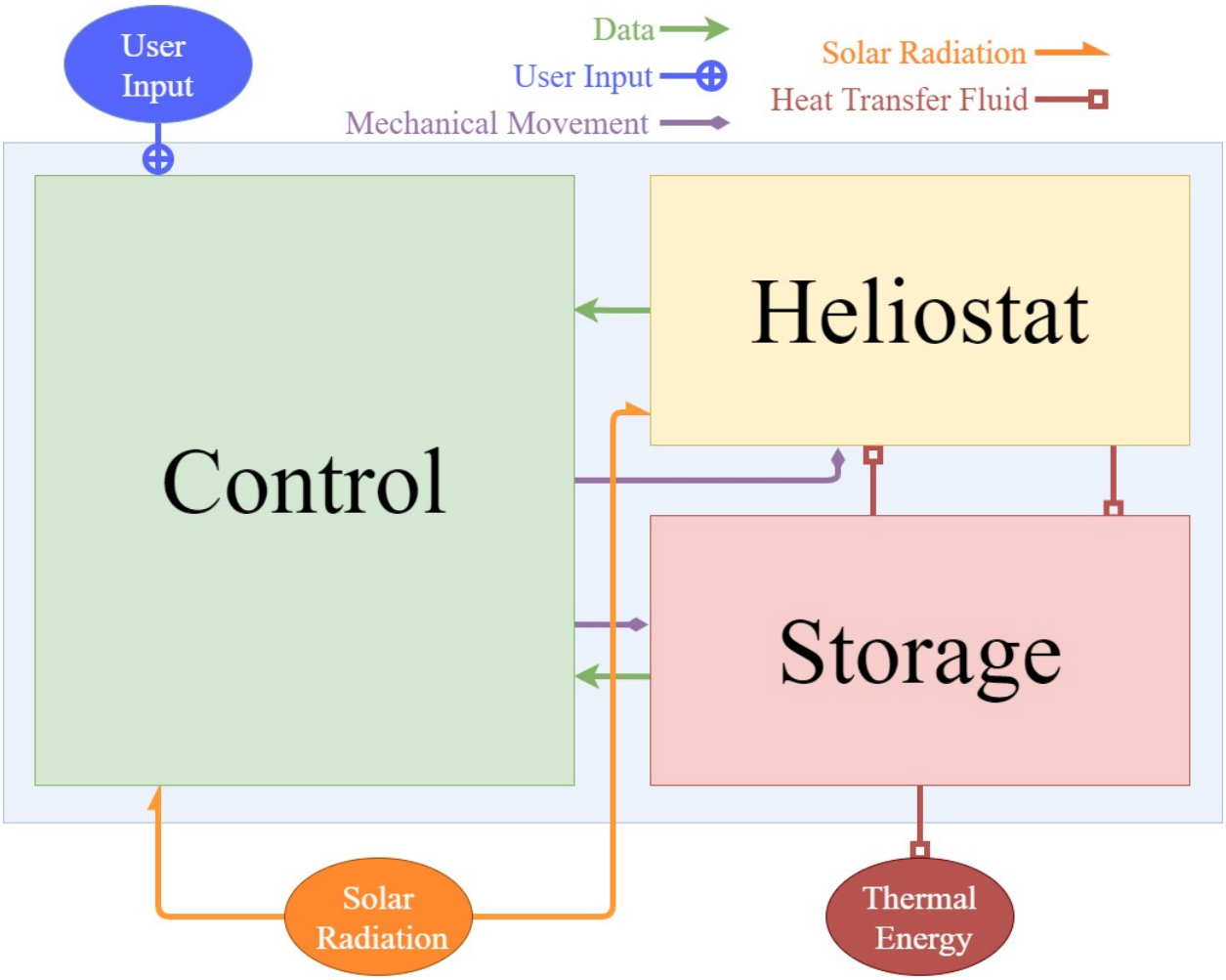

Figure 13: Level 1 system architecture

5.3 Level 2

Level 2 breaks these three blocks down even further turning the three blocks into seven. This level is when the system process become more clear. Housed in the control block is the user interface (UI) which is the physical connection between the system and the user. Attached to the UI is the controller which is the "brain" of the system. The controller is what will make all logical decisions and "tell" what parts to move when. The solar tracking block contains the sensors required to determine the position of the sun. The output of these sensors are fed into the controller. The actuator block is the last block contained in the control block. The actuator block converts data signals from the controller into mechanical movements. In the heliostat block, there is a block that represents the mirror, and another that represents the receiver. The following figure shows the level 2 architecture.

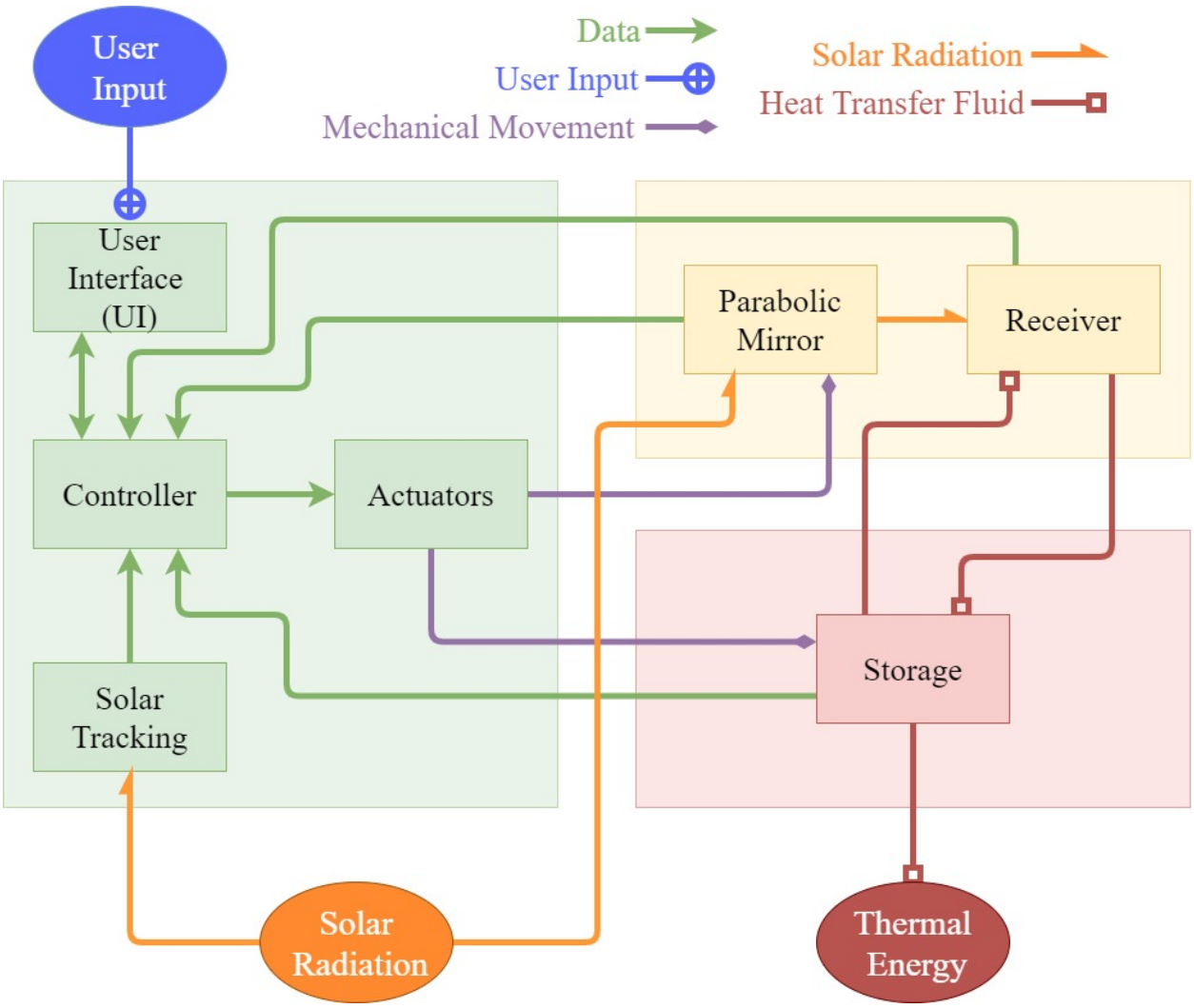

Figure 14: Level 2 system architecture

6 Material Selection

It is important to restate that the project philosophy is to use low cost and readily available materials in this design. The material selection needed to be sourced from places like home centers or places online that are commonly used for maker style projects.

Pressure treated southern yellow pine plywood was chosen for the frame and base components. This material was chosen due to its low cost, availability, rating for outdoor applications, and could be used on a CNC router. It is also sourced from renewable sources which is ideal in a renewable energy project. A white latex paint was chosen to achieve a more professional look while also adding some light reflective properties to pieces that were not supposed to absorb light. It was important that the project looked complete and finished as it is likely to be used in future years at USI.

A 10' long type K copper pipe was chosen for the receiver tube. This was chosen for copper's desirable thermal characteristics and low chemical reaction possibilities with a wide array of HTFs that may be used with this system in the future. Type K tubing falls under the American Society for testing and materials (ASTM) B88 standard for use in potable water and solar applications. This pipe was painted in a high temperature black paint to increase light absorption. High temperature Polytetrafluoroethylene (PTFE) rings were chosen to separate the pipe from the wood end caps. These rings have temperature ratings of up to 500° F.

Glass mirrors are typically used for parabolic trough applications but come at a hefty cost. They are also difficult to deal with in transportation. A thin Lexan substrate was used in conjunction with a reflective mylar film as a much more cost-effective solution. Lexan is polycarbonate resin thermoplastic that can be cold or hot formed to different shapes without cracking. It is rated for outdoor use and has UV protection. This sheet comes flat and was cold formed into the parabolic shape needed. The sheet is held in place by aluminum brackets. The reflective mylar film provides a highly spectral reflective (>92%) surface similar to that of glass mirrors that use PVD. It is much lower cost material, but likely will not have as long of a lifespan when compared to glass mirrors. This film was bonded to the Lexan substrate with 3M Super 77 spray adhesive. This adhesive is industrial strength but lacks a long working time. It also creates some bubbling caused by small partially dried pieces of adhesive being dispersed on the substrate while applying. Having a completely smooth surface, enough time to apply the adhesive, lay the film in place, and squeeze out air bubbles is crucial to creating a nice smooth surface.

Metal shafts were created by milling 1566 carbon steel into the desired dimensions. These shafts were placed into nickel plated steel bearings for support and to reduce friction while rotating. Cold rolled metal steel stock was also used to create the holding block for the shafts. The shafts were then welded to the holding blocks. This process created shafts that are adapted from 1.25" to 14mm, which is necessary to fit other components of the system.

Electrical components such as sensors, motors, embedded systems and power electronics are covered in detail in the Controls section of this report. A comprehensive Bill of Material can be found in Appendix A. The total cost of the entire system was approximately $1500, which is incredibly inexpensive when compared to just the cost of a traditional glass mirror.

7 3D Modeling

The design parameters and material selection provided a starting point for 3D modeling. Solidworks by Dassault Systems was chosen as the modeling software for a few reasons. First, Solidworks was available to use for free through the university's resources. Secondly, Solidworks is incredibly powerful software that can allow for future senior designs in anything from finite element analysis for framing, to computational fluid dynamics for HTFs, to cost estimation for the next iteration of parabolic trough design. Third, Solidworks has a built-in equation driven curve function that makes shaping a curve like a parabola much easier. Lastly, to meet the goal of producing a physical system, Solidworks can easily produce parts that can be imported into computer aided manufacturing (CAM) software. This made the manufacturing of each part much more precise and faster using a computer navigated control (CNC) router. Manufacturing is covered in detail in the next section of this report..

This project had three major iterations of design before any physical parts were manufactured. Each of these iterations is covered in detail below.

7.1 Iteration 1

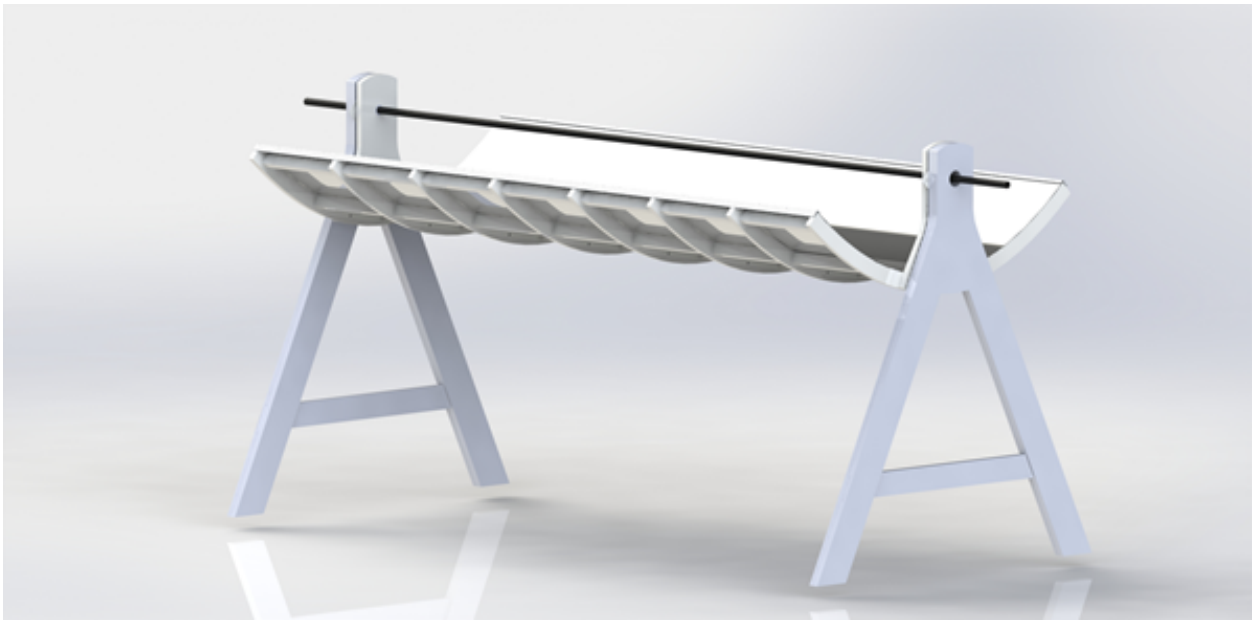

Figure 15: 1st iteration of 3D model

The first iteration of design is shown in Figure 15. Parabolic shaped ribs were created from Equation 5 as the main mirror support. Five long stiffener rails were added to stiffen the top of the trough. The ribs and stiffeners were designed to fit together with half lap joinery to avoid using screws or bolts. This was intended to make assembly easier. The Lexan sheet can be seen being held in place by aluminum brackets. These brackets were attached with bolts and threaded inserts. Threaded inserts were used in all bolted connections. This was a design choice to avoid destroying the plywood in cases were pieces needed to be bolted and unbolted multiple times. A copper pipe was used as the receiver with the high temperature bushings providing protection to the wood end caps. The pivot point is located at the receiver tube in this design. Thrust bearings were used for a rotation point. Finally, the base can be seen holding the trough off the ground.

This was a solid first attempt at a 3D model, but this design had a couple of flaws. First the base would be unstable in the long horizontal direction, especially outdoors in the wind. Secondly a large motor would have been needed to rotate the trough with the pivot point located so far away from the center of mass.

7.2 Iteration 2

Figure 16: 2nd iteration of 3D model

The second iteration of modeling improved on the previous design by lowering the pivot point closer to the center of mass. A steel pipe was used as a torsion tube to provide more support to the system, with shafts and bearings holding the top of through to the base. The base was widened for more stability. The base is held together by half lap joinery to avoid the need for bolts and screws.

This iteration was much better than the first attempt but still an issue with the center of mass not being located at the pivot point. Large torque motors and gearing are expensive and needed to be avoided.

7.3 Iteration 3

Figure 17: 3rd iteration of 3D model

The final iteration of modeling corrected the need for a large torque motor by moving the pivot point to the center of mass. Shafts adapters are seen to transition from the 1.25" bearings to the 14mm output size of the gearbox. More detail such as a stepper motor and worm gearbox were added to the model. Lastly a block representing the electrical enclosure was added. This enclosure would hold all the electronics for the control system.

This third iteration was used to start the process of manufacturing physical parts. However, before parts could be created a nested model of all the pieces needed to be created. Nesting is the process of laying out individual pieces onto a sheet good in the most useful way possible. The 3D model assembly was deconstructed, then each piece was mated to a model of 4'x8' sheet of plywood to create each nested sheet. Special attention was needed during this process to assure the router would be able to reach pockets and holes that did not go through the entire sheet. Figures 18 - 19 show each nested sheet that was created.

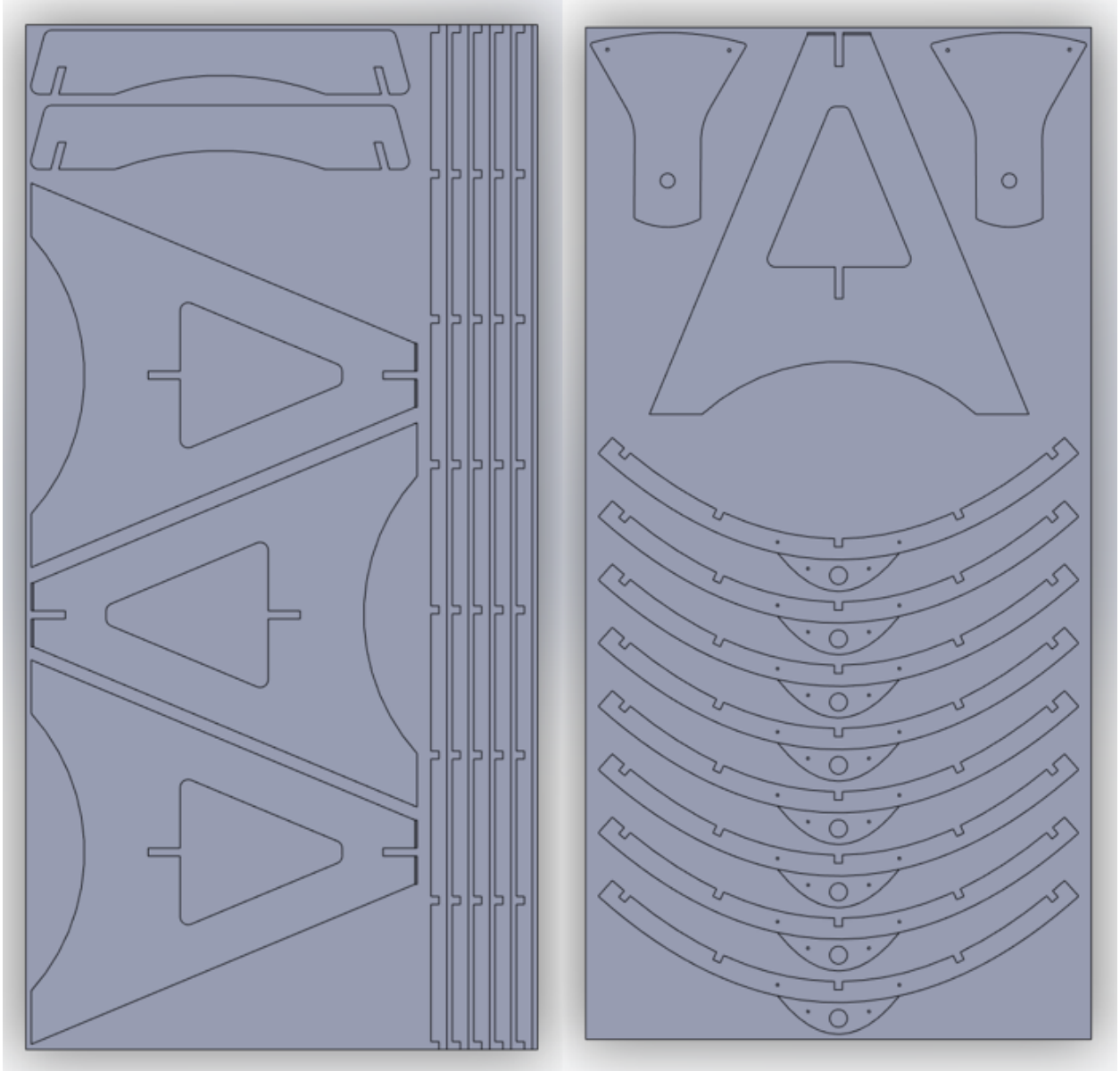

Figure 18: Nested sheet 1 and nested sheet 2

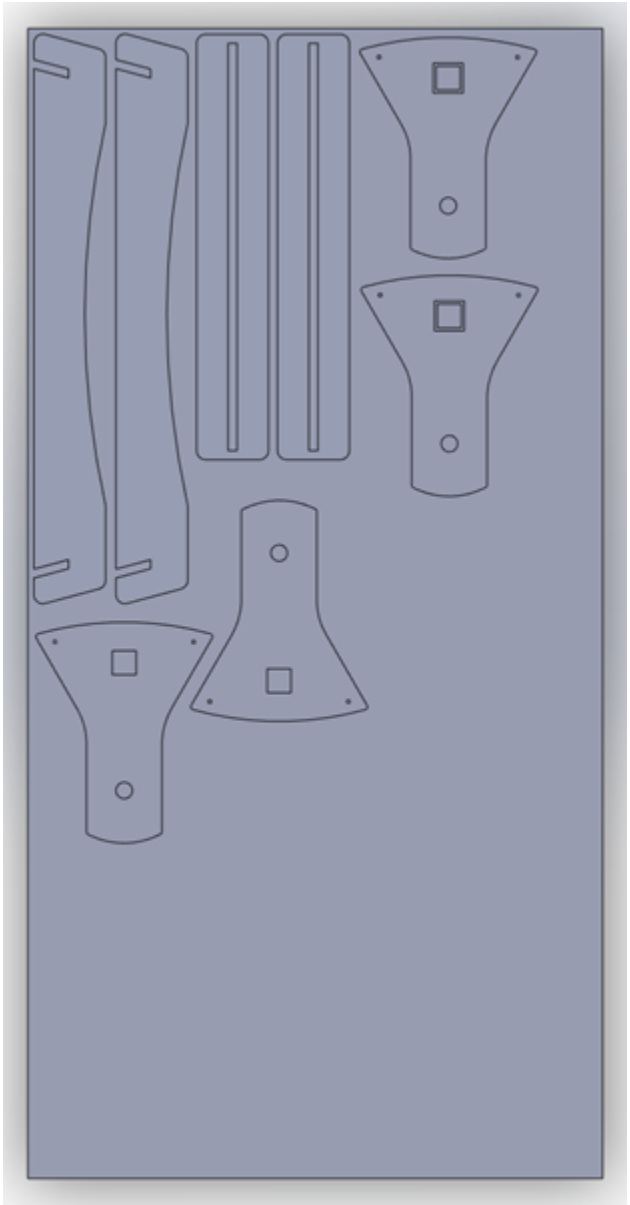

Figure 19: Nested sheet 3

These nested sheets needed to be converted into 2D .dxf files for use with CAM software. However, there is no way to directly take these nested assemblies and create the .dxf files that were needed. Solidworks does have a way to take assemblies and create 2D drawing sets though, and these 2D drawings can be converted into .dxf files. This approach worked well to create the proper .dxf file types for use in CAM software during manufacturing.

8 CNC and Assembly

The .dxf files that were created during the previous process were used as a starting point for manufacturing parts. These .dxf files were imported into MasterCam, which is a CAM software that can provide cutting instructions for the CNC router.

8.1 MasterCam

Each nested sheet was imported into MasterCam individually. Toolpaths were created using contour cuts for the outline of each piece first. More toolpaths were created for pocket cuts and holes as needed for each sheet. MasterCam has many parameters for tooling, toolpaths, cut depths, and other various items that the CNC router needs defined to cut properly. Figure 20 displays an example of these parameters inside of MasterCam.

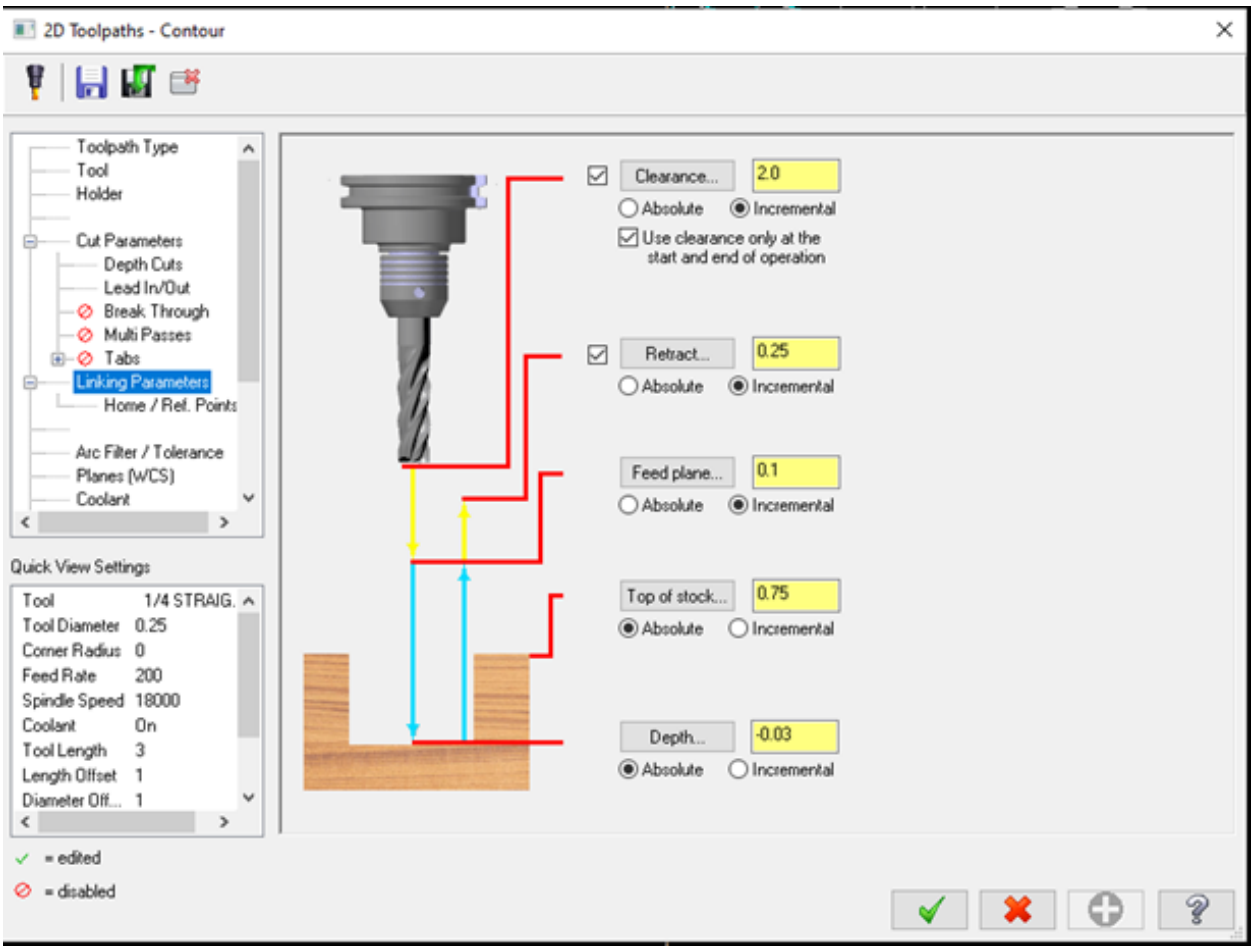

Figure 20: MasterCam toolpaths and parameters

After toolpaths and parameters were completed, a simulation was ran to verify everything looked correct. Corresponding G-code was then generated inside of MasterCam. G-code is the language that a CNC system generally uses to complete sequential movements of a machine. Figure 21 displays an example of G-code that was generated.

```
%
O0000(MIKE PLY 1)
(DATE=DD-MM-YY - 25-02-22 TIME=HH:MM - 18:55)
(MCX FILE - C:\USERS\REZPL\DOCUMENTS\_SENIORPROJECTCAD\_SENIORPROJECTCAD\MASTERCAM_FILES\PLYWOODASSEMBLY1.MCX-7)
(NC FILE - D:\MIKE PLY 1.NC)
(MATERIAL - WOOD INCH)
( T3 |   3/8 BULLNOSE | H3 )
( T1 |  1/2 STRAIGHT BIT | H1 )
( T2 |   1/4 STRAIGHT BIT | H2 )
N100 G17 G20 G90 G40 G80 G64 G49 G0 M05
N110 G8 P1
N120 G90 M05 Z2.
N130 G52 Z2.
N140 T3 M6
N150 G0 G90 G54 X20.9583 Y95.6741
N160 S18000 M3
N170 G43 H3 Z.85
N180 G1 Z-.007 F100.
N190 X27.1583 F200.
N200 Z.0305
N210 Y95.6641
N220 X20.9583
N230 Z.068
N240 Y95.654
N250 X27.1583
N260 Z.1055
N270 Y95.644
N280 X20.9583
N290 Z.143
N300 Y95.6339
N310 X27.1583
N320 Z.1805
N330 Y95.6239
N340 X20.9583
N350 Z.218
N360 Y95.6138
N370 X27.1583
N380 Z.2555
N390 Y95.6038
N400 X20.9583
N410 Z.293
N420 Y95.5937
N430 X27.1583
N440 Z.3305
N450 Y95.5837
N460 X20.9583
N470 Z.368
N480 Y95.5736
N490 X27.1583
N500 Z.4055
N510 Y95.5636
N520 X20.9583
N530 Z.443
N540 Y95.5535
N550 X27.1583
```

Figure 21: G-code generated for use with CNC router

8.2 CNC Router

Plywood sheets were loaded onto the bed of the CNC router individually. These sheets were secured with wood screws to keep each sheet flat and stationary while cutting operations ran. Feed and speeds were purposely left slower than needed since this was the first experience with MasterCam and CNC routers. Keeping the speeds low allowed for safer operation, but at the disadvantage of taking longer to complete each sheet. Figure 22 shows the CNC router processing parts.

Figure 22: CNC router with parts being cut

This process should have been straight forward, but some issues were encountered while processing parts. The plywood that was sourced was a low-quality material that was incredibly warped and even damaged in some locations. Adjustments were made to try and flatten the material with more anchoring screws. More adjustments were made to the CAM parameters as well to account for fluctuations in the material. However, most of these issues could have been avoided if higher quality hardwood plywood material was sourced from a cabinet shop. The cost of this material is higher but would allow for a much smoother CNC experience in the future.

8.3 Final Preparations and Paint

Figure 23: Rounded and sanded parts

With CNC operations completed, final preparation of each part was started. These preparations included rounding over any edges to avoid sharp points, sanding, and painting. Rounding of edges was accomplished using a hand router and a roundover bit. Sanding was done with an orbital palm sander and 120 grit sandpaper. 2 coats of white exterior latex paint was applied to each part to further protect against the elements. Figure 23-24 displays this preparation process.

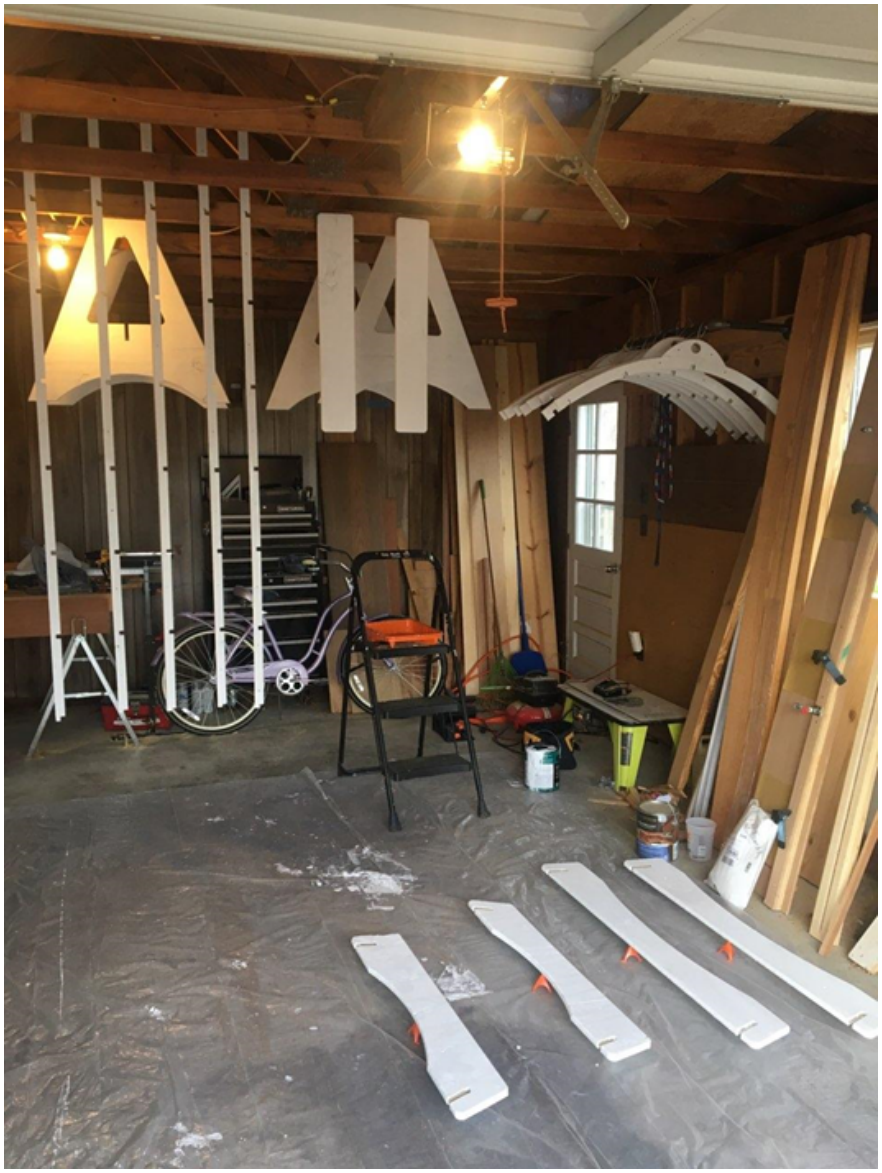

Figure 24: Painted parts

8.4 Assembly

Assembly began with constructing the top section of the trough. The parabolic ribs were spaced out and supported by wood boards and clamps. The long stiffeners were then inserted into the ribs and tapped into place with a mallet. The top assembly is designed to be connected together with half lap joinery which means no screws or bolts are needed. This is a similar method to how some furniture is assembled. Figure 25 shows the top assembly process.

Figure 25: Assembly of the top section of the parabolic trough

Next the Lexan substrate was fitted to the top section then attached in place with aluminum brackets, threaded inserts, and bolts. This part of the assembly took some time because verification of the shape of the parabola was extremely important. If the shape is not perfect, light will scatter or diffuse away from the receiver. Arc lengths of the ribs, x and y measurements, and cross measurements were all compared to the desired model and were confirmed to be correct. Figure 26 shows the Lexan substrate being fitted to the top section.

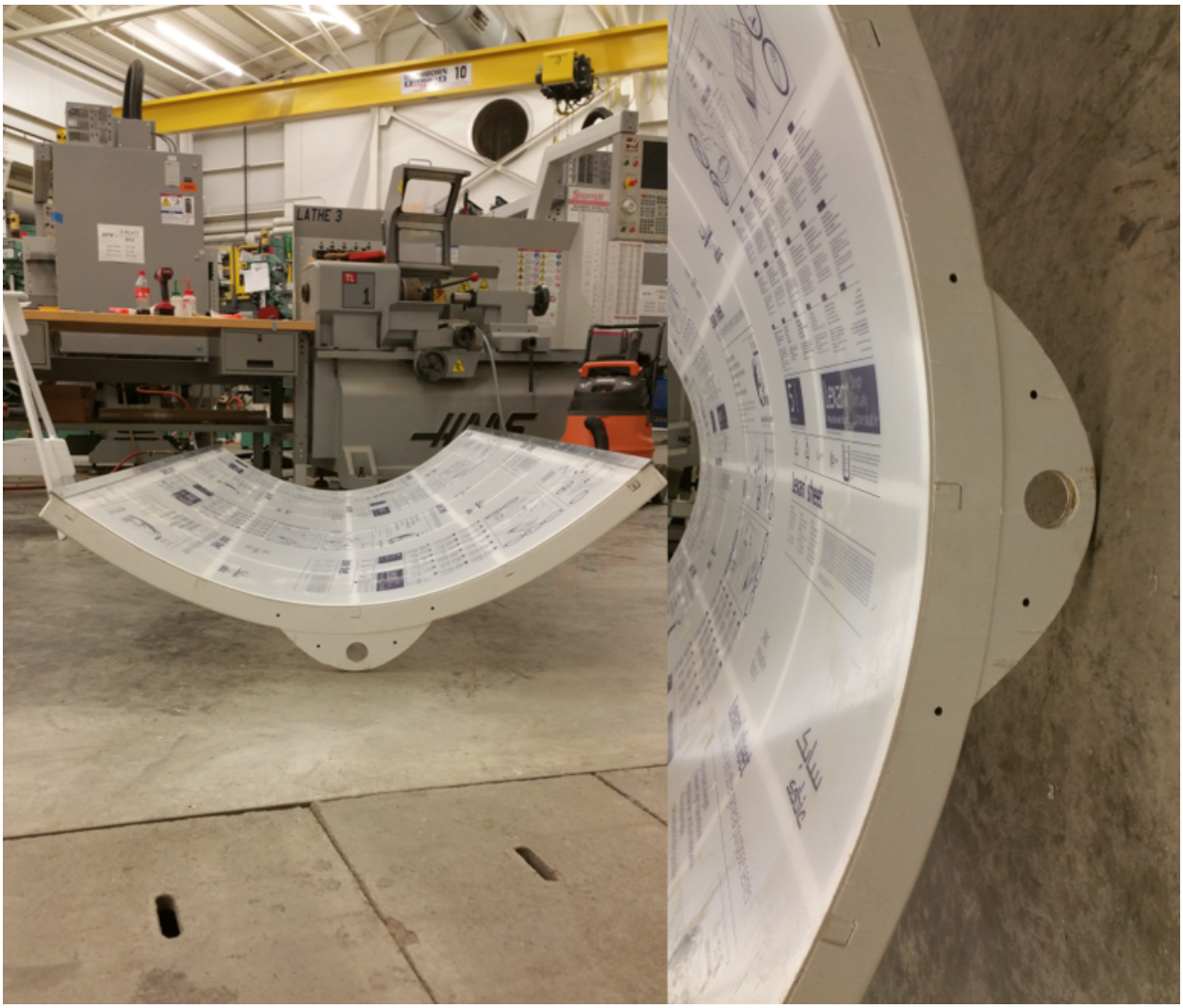

Figure 26: Fitting Lexan substrate to the top section of trough

The base pieces were then assembled together with the same half lap joinery that was used for the top section. Threaded inserts were used with bolts to provide more rigidity for the top plate of the base. An issue occurred during this process as paint and wood chips were thought to have gotten stuck in the joinery while assembly occurred. This is thought to have been a tolerance issue as everything was designed to fit together without paint. This resulted in the upper base rail to protrude higher than designed. This issue was corrected by using a hand plane to remove the excess height from the top rail. Figure 27 shows the base assembly.

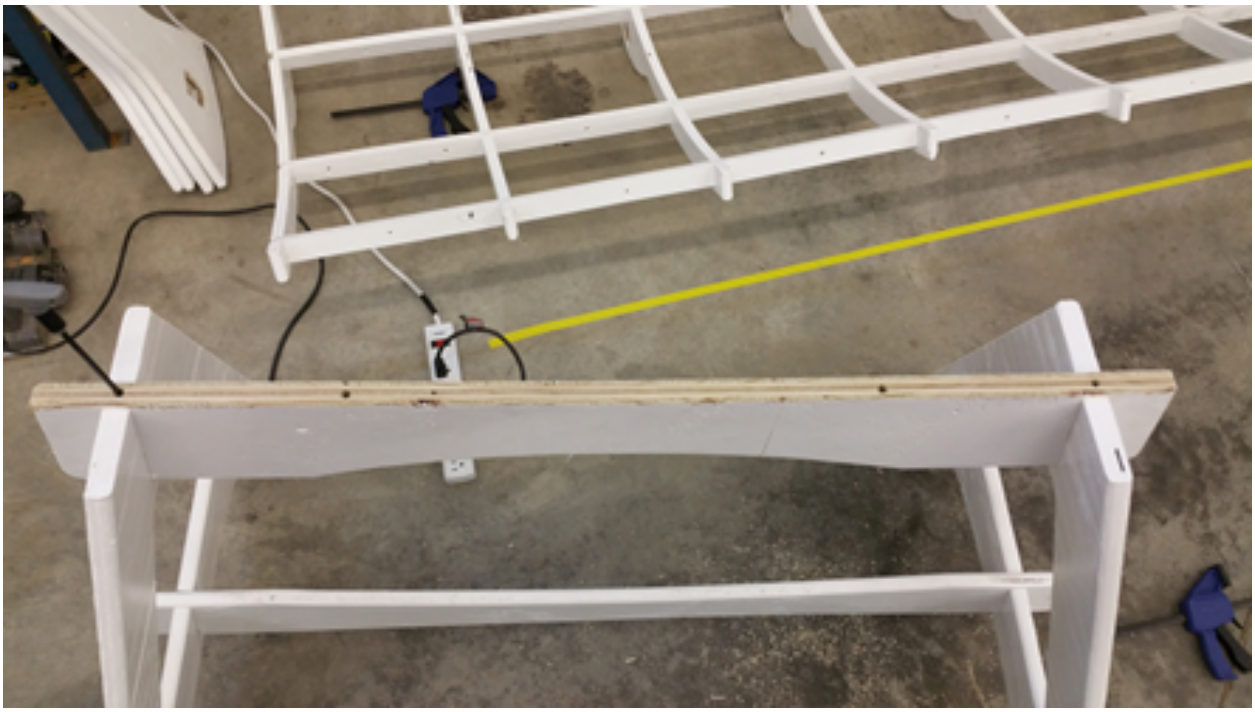

Figure 27: One of the assembled base pieces showing the upper base rail with threaded inserts

Bearings, end caps, gearbox, and shaft adapters were then secured in place. The top assembly was attached to the base assembly to form parabolic trough system. At this point in the process another issue arose. It was apparent that more support on the end caps would be needed to prevent premature failure of the plywood. Additionally, the top of the trough was undergoing torsion, so additional bracketry was added underneath the top of the trough to help stiffen the assembly. Aluminum tubular bracing was also added at this point to help stiffen the assembly, though it was not originally in the design. Figure 28-29 shows the assembled parabolic trough.

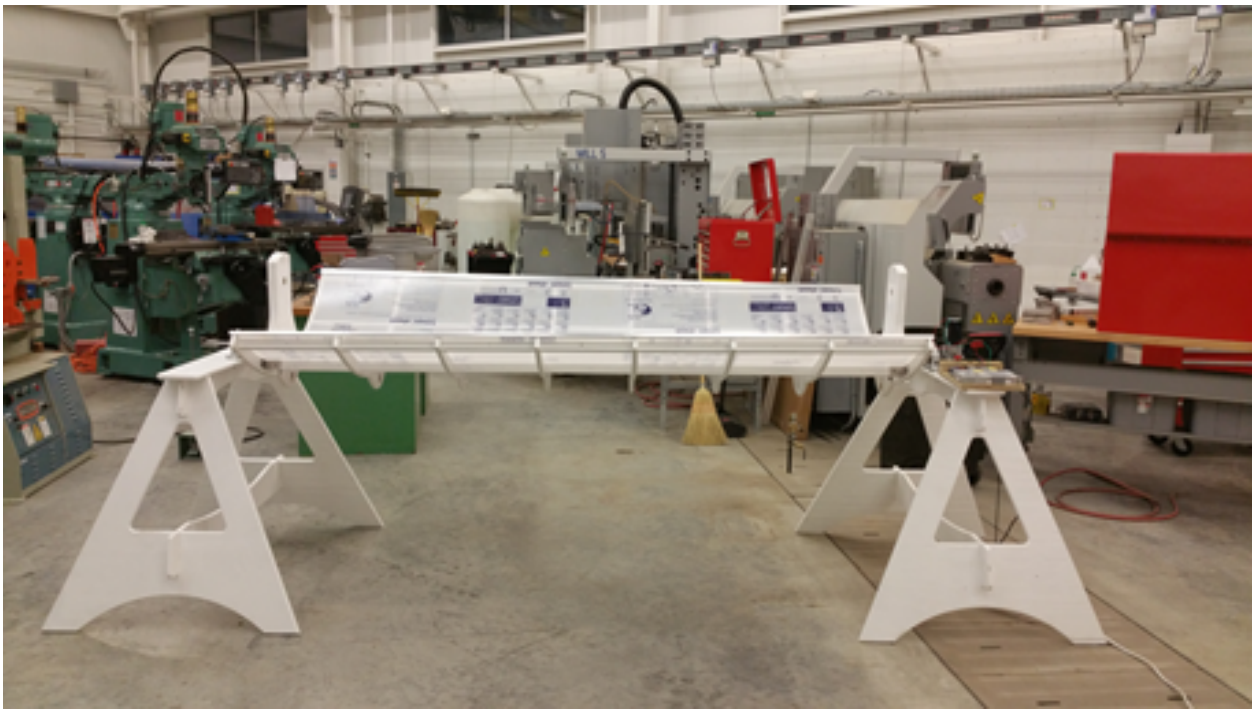

Figure 28: The partially assembled parabolic trough

Figure 29: Aluminum bracing added to the top of the trough

This assembly process continued by applying the reflective mylar film. The reflective film can be scratched fairly easily and needed to be done later in the assembly process to avoid damage. Applying the mylar film to the substrate proved to be a difficult task, mainly due to trying to find a suitable adhesive that would stick to the film appropriately. Seven different types of adhesives

were experimented with including acrylic based cement, super 77 spray adhesive, super 90 spray adhesive, 3m photo spray adhesive, double sided tape, pvc cement, and contact cement. Many of these adhesives would not bond to the mylar film at all, and only mediocre results were found using the spray adhesives. The best overall adhesive that provided a sufficient bond with minimal bubbling was the super 77 spray adhesive from 3M. The process of applying the film to the substrate was investigated as well, but there is minimal research on how a mylar film with no adhesive back is applied to a substrate. An experimental approach was developed to test which method was suitable. The best results were found by first spraying super 77 adhesive to both the back of the mylar film and the substrate, making sure to keep the adhesive from contacting the finished parts of the mirror. Next one end of the film was held up in the air while the other was placed very carefully along the substrate. A squeegee with soapy water was used to slowly press the film into the substrate. Figure 30 and 31 displays this process.

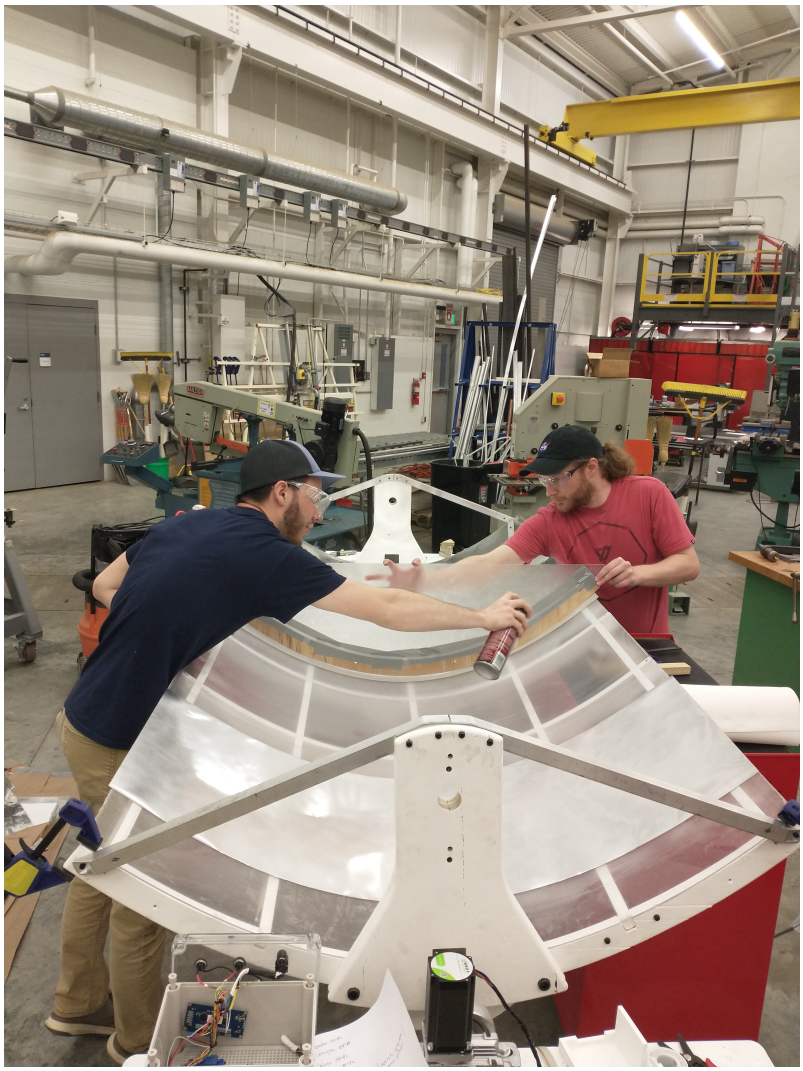

Figure 30: Applying spray adhesive to the trough and film

Figure 31: Applying mylar film to the trough with squeegee

The receiver tube was then fitted in place, using high temperature nylon bushings to keep the copper pipe from coming in contact with the plywood end caps. At this point the trough was still undergoing torsion. To make something that could hold up to the elements outside, a tensioning cable system was placed above the reflector. This tension system does correct the torsion issue well, but at the cost of creating shading on the mirror. A better solution to this issue likely involves a large redesign using a torsion tube or torsion box underneath the assembly. Another option could be to use stiffer materials like plexiglass or stamped sheet metal with a steel substrate, which would still follow the project philosophy of using low-cost materials. A final assembly showing these additions can be seen in Figure 32.

Figure 32: Fully assembled system

9 Controls

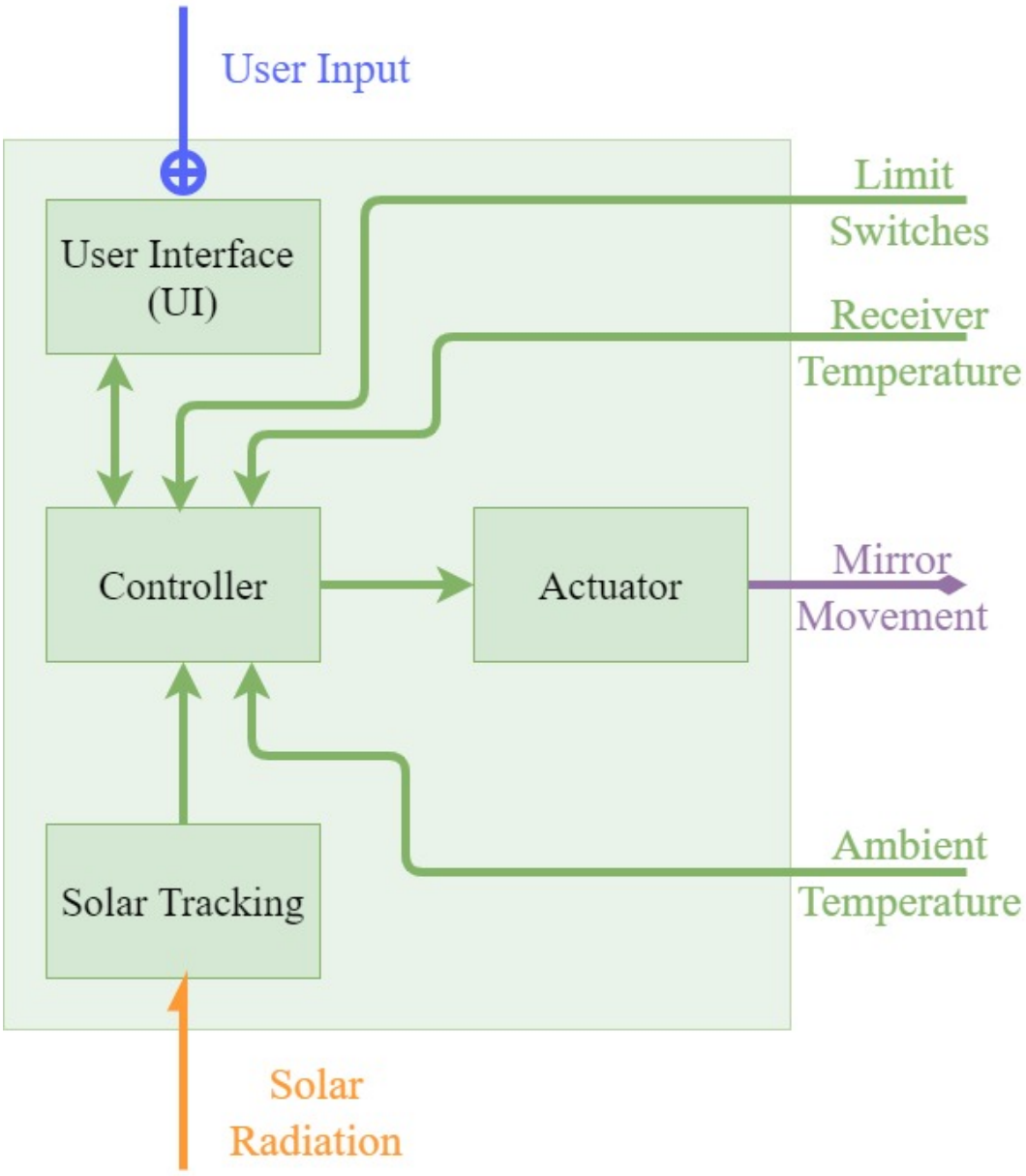

Figure 33: Control architecture

Figure 33 above is a breakdown of the control block as a part of the overall system architecture. This architecture is slightly different than the architecture seen in the previous chapter due to design changes based on time and manpower constraints. The main difference being that the actuator block is no longer controlling flow of the HTF as capture of the HTF was omitted from the scope of this project. The following sections of this chapter will discuss the hardware, programming, and testing procedure used to fully implement a complete control system.

9.1 Hardware

9.1.1 Controller

The microcontroller selected for this project was the Arduino Mega 2560. It was selected based on a few key criteria. The first being that it has a large amount of input and output pins. Sixteen pins are analog input pins, and fifty-four are digital pins that can be configured as input or output pins. For this project, analog pins are needed for each LDR used in the solar tracker. There are eight LDRs used in the solar tracker, so eight analog pins are required. When comparing the Arduino Mega to similar products, most other microcontrollers only have a total of eight analog inputs which would result in there being no room for later expansion. In addition, twenty-eight out of the total fifty-four digital pins were utilized in this project so for possible future additions the extra digital pins were required. Secondly, the Arduino Mega was selected due to its open source code and libraries that makes configuration of peripherals such as an LCD screen quick and efficient. This allows more time to be spent programming logic rather than troubleshooting communication issues with common peripherals. Lastly, the Arduino platform was selected due to the designers having prior experience using the software and IDE used to program the device. Figure 34 shows an Arduino Mega connected to a breadboard with LDRs to be used for preliminary testing.

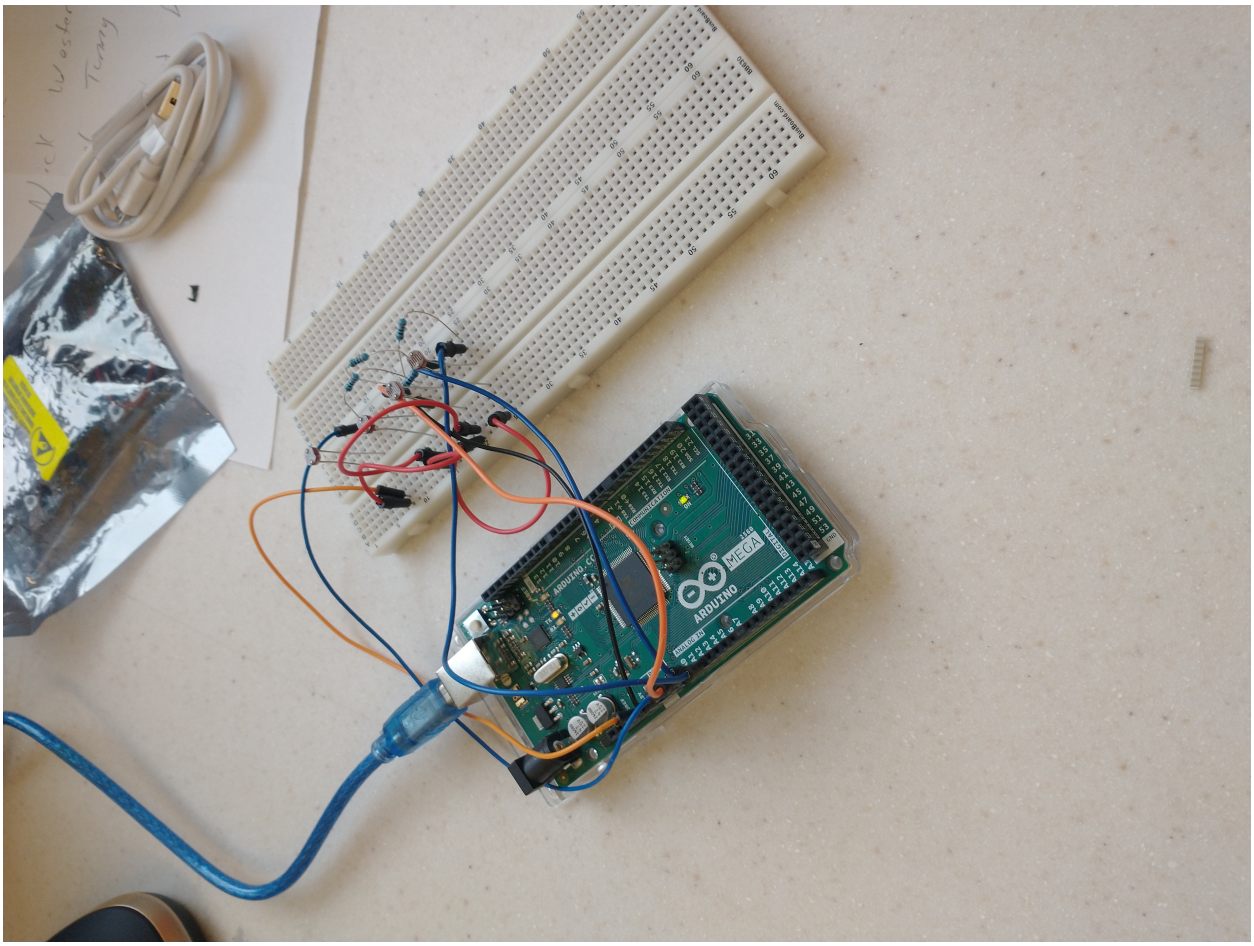

Figure 34: Arduino Mega connected to breadboard with LDRs for testing

The "Controller" block nested inside of the "Control" of the system architecture shows two main components, the Arduino Mega which is the primary controller, and a driver for the stepper motor used to orient the mirror. The arduino Mega is used to send low-power signals to the driver board so the stepper motor can operate on its required 48V, a voltage the Arduino cannot directly control. Thus, a DM542T Digital Stepper Driver was selected as it can drive a stepper motor using up to 48V and 4.2A, which is slightly less power than the NEMA 23 stepper motor that was selected can handle. This means that even if configured incorrectly and a current spike is sent to the motor, the driver will never supply a current large enough to ruin the motor. The downside of this is that if the motor needs to be driven at its limit to obtain maximum torque, a different driver would be required. Figure 35 shows the internal workings of the controller block with the Arduino and driver.

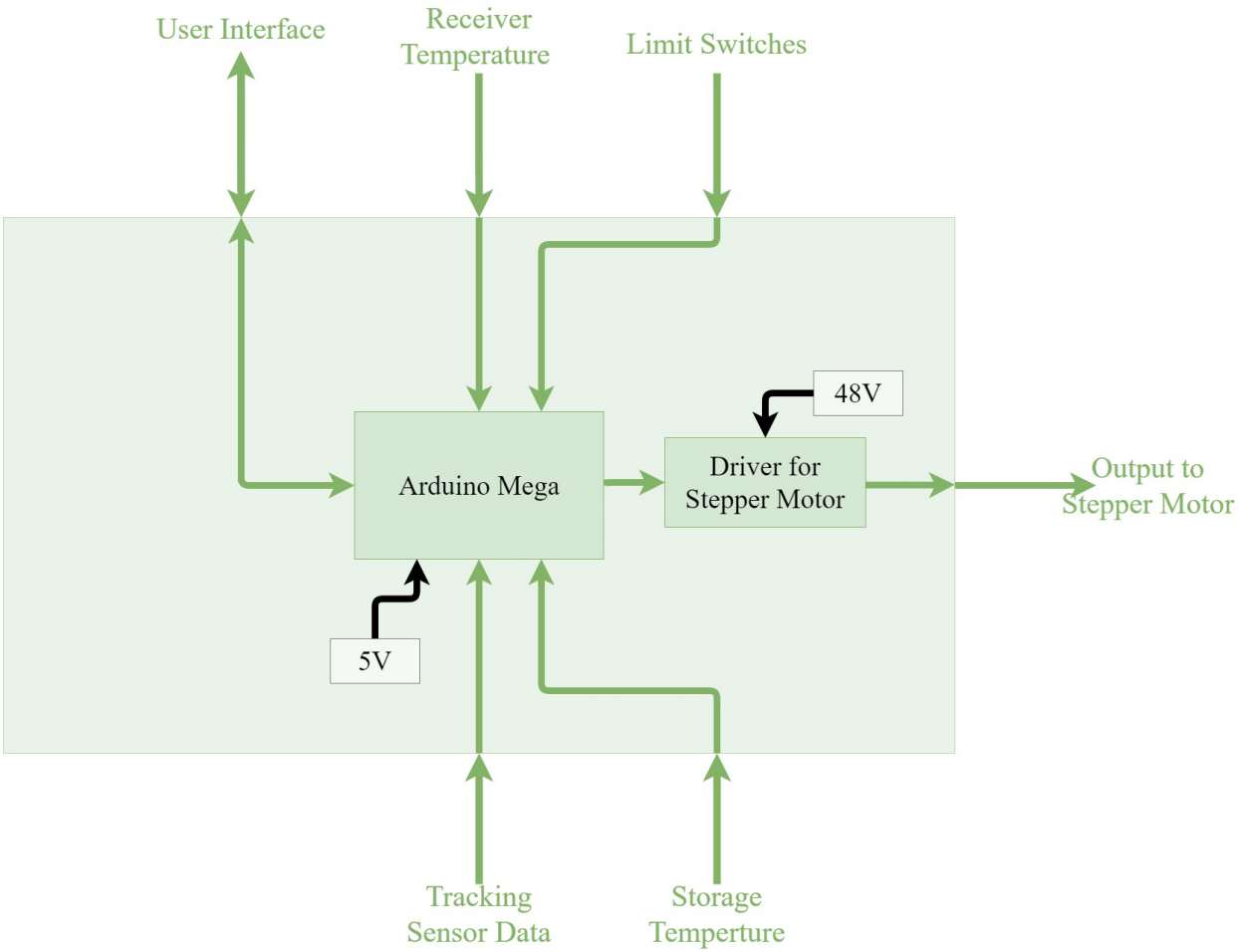

Figure 35: Controller block

Due to the large amount of pins utilized on the microcontroller in this project, and possible revisions made to wiring after initial implementation, a prototyping shield was used in conjunction with header pins to allow for all wires to easily be disconnected and reconnected to the Arduino Mega. Table 6 below shows all wires connected to this shield along with what the wire is used for and the pin it is connected to.

Connection	**Arduino Pin**	**Wire Color**	**Wire Stripe**
Encoder Button	D38	White	Orange
Encoder DT	D37	Orange	Black
Encoder CLK	D36	Black	Orange
5 Volt (UI)	5V	Red	Gray
Ground (UI)	GND	Gray	Red
TC1 CLK	D22	Yellow	NA
TC1 CS	D23	White	NA
TC1 DO	D24	Green	NA
TC2 CLK	D25	Yellow	NA
TC2 CS	D26	White	NA
TC2 DO	D27	Green	NA
LCD RS	D45	Orange	Yellow
LCD EN	D46	Yellow	Orange
LCD D4	D47	Gray	Black
LCD D5	D48	Black	Gray
LCD D6	D49	Blue	Yellow
LCD D7	D50	Yellow	Blue
CCW Limit	D3	Red	NA
CW Limit	D2	Red	NA
Reset	RESET	Yellow	Brown
Alarm	D4	Brown	Yellow
Red LED	D10	Gray	White
Blue LED	D11	White	Gray
E-Stop (ALM+)	D18	Blue	NA
LDR 1	A0	Ribbon	NA
LDR 2	A1	Ribbon	NA
LDR 3	A2	Ribbon	NA
LDR 4	A3	Ribbon	NA
LDR 5	A4	Ribbon	NA
LDR 6	A5	Ribbon	NA
LDR 7	A6	Ribbon	NA
LDR 8	A7	Ribbon	NA
Driver DIR-	D6	NA	NA
Driver PUL-	D7	NA	NA
CW Button	D15	Yellow	Green
CCW Button	D14	Green	Yellow

Table 6: Arduino pin allocation and wire color

9.1.2 Solar Tracking

The entire control system was designed with one central subsystem at the forefront, and that is the solar tracking system. The goal of this project component was to create a novel, low-cost sun tracker. Because of this goal, the design decision was made to use light dependent resistors as the sensor utilized in the tracking system. Figure 36 shows an example of an Adafruit 161 LDR which was the LDR model selected. One of these LDRs at the time of writing costs $0.95. This is why they were selected to satisfy the low-cost requirement. In addition, for large-scale production, the price of these LDRs can be greatly reduced when purchased in bulk making them an ideal choice for a solar tracker focused on cost minimization.

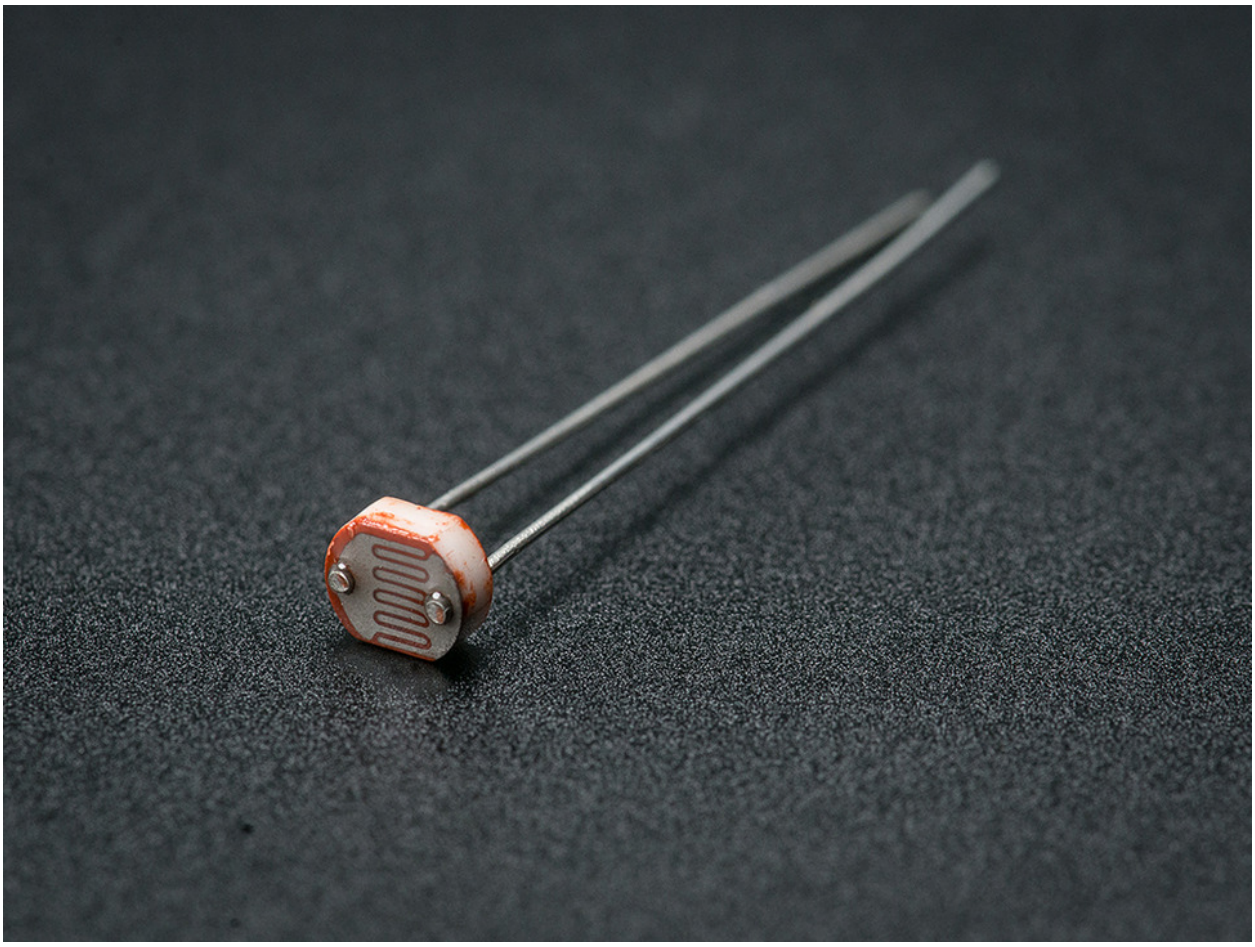

Figure 36: Example of an Adafruit 161 LDR

Another reason the LDR was selected is due to its high input resolution which provides high sensor sensitivity. This is because LDRs are analog devices (theoretically infinite input resolution) so the output resolution is only limited by the bit-width of the analog to digital converter (ADC) on their microcontroller. The Arduino Mega has a 10-bit ADC so the voltages from the sensor circuit were scaled between 0-1023. This sensor circuit had a similar topology to that of a voltage divider circuit with the resistor typically connected to the 5V source being where the LDR is placed. Figure 37 shows the schematic for connecting one LDR to a microcontroller with the label "Analog Pin" being where the ADC pin of the microcontroller would be connected.

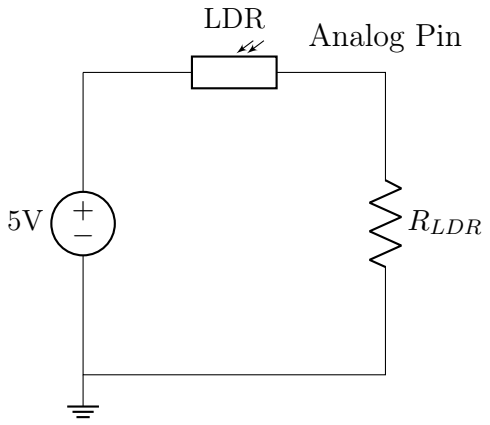

Figure 37: Circuit schematic for single LDR

One LDR alone is not enough to determine where the sun is in relation to itself; all one LDR can do is tell you the intensity of light present. For this reason, a design was made to incorporate eight LDRs in one single enclosure. This enclosure was designed in such a way that shadows would be cast on specific LDRs based on the relation of the enclosure and sun locations. To complete the task of tracking, the enclosure was designed to have three different levels of tracking. The primary level is used for coarse adjustments, being any error in position greater than approximately 25 degrees and up to the maximum error possible (i.e. sunset versus sunrise). The secondary level is responsible for positioning the mirror within a few degrees of the sun, and the tertiary level is used to determine when the mirror (and tracking assembly) is pointed directly at the sun. Figure 38 shows a block diagram of how these three levels will convert solar radiation to data which can be used by the microcontroller to make decisions of where to move the mirror.

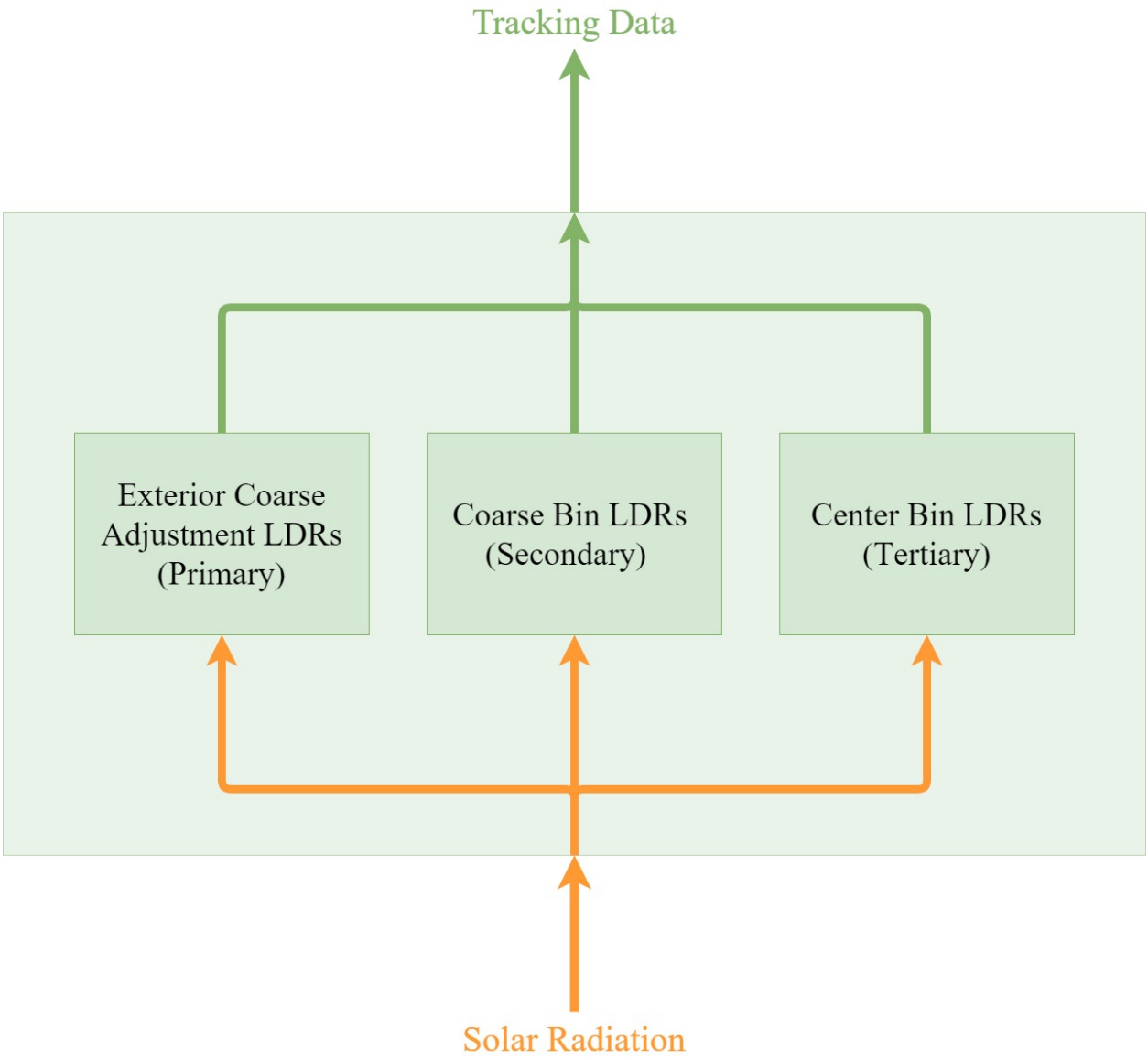

Figure 38: Solar tracking block

The layout of the LDRs were positioned so that measurement anomalies might be reduced. One LDR was used for each of the primary sensors (one for clockwise (CW) rotation and one for counterclockwise (CCW) rotation) for a total of two. Two LDRs were used in each of the secondary positions, for a toatl of four secondary sensors. Finally, two LDRs were placed in the in the single tertiary enclosure location for a total on two LDRs in this position. This means that eight total LDRs were used to form the solar tracking assembly. Figure 39 shows the location of the LDRs in relation to the enclosure from a top-down view. Each LDR location is indicated with a red circle.

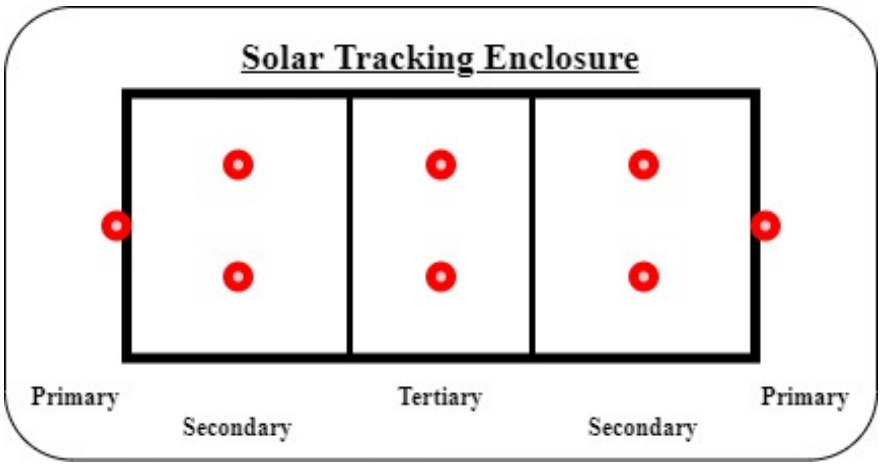

Figure 39: Top-down view of solar tracking assembly with LDR locations in red

First Iteration of Enclosure The first iteration of the enclosure was designed using an online 3D modeling software called TinkerCAD. A design choice that proved to be extremely beneficial in the prototyping phase of the solar tracker was to not print the whole enclosure as one unit, but instead make separate "caps" that adjust where shadows would be cast on the LDRs. These caps were designed to fit securely with no use of glue or fasteners and were mounted on top of the vertical walls between the secondary and tertiary bins. After modeling, the enclosure and two caps were printed on a PRUSA 3D printer with PLA filament. Figure 40 shows the modeled main enclosure and the first iteration caps along with them printed on the PRUSA printer.

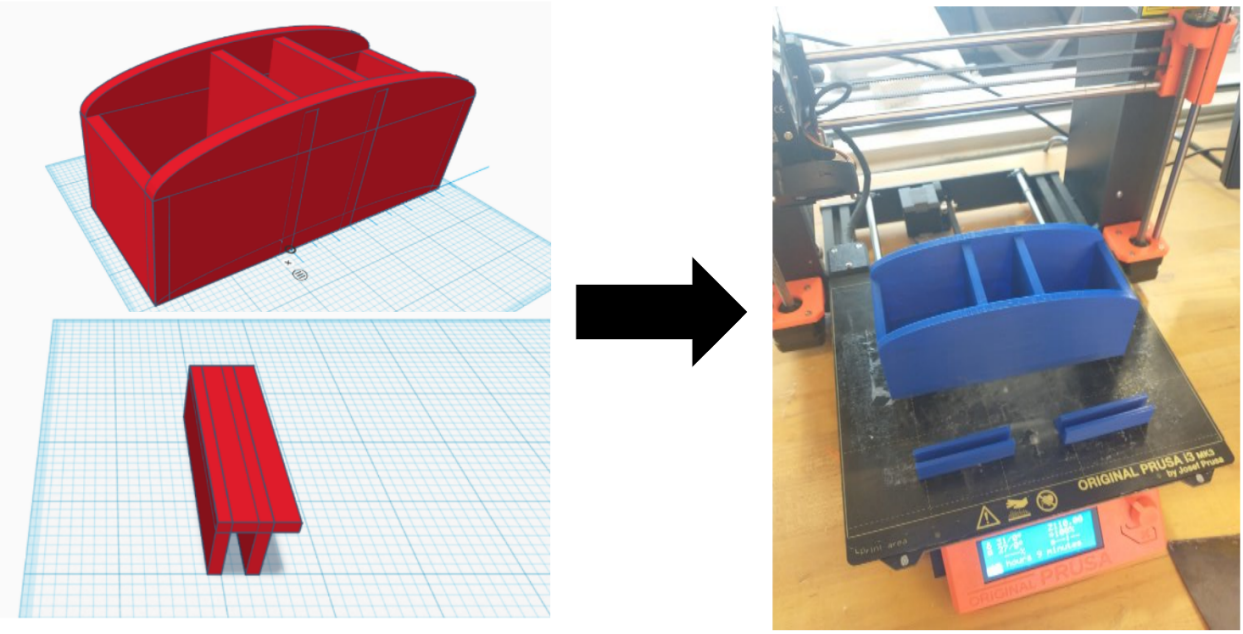

Figure 40: First iteration of tracking enclosure modeled and printed

Second Iteration of Enclosure After some testing, a second iteration of tracking enclosure could be built to increase the accuracy and precision of the tracker. In this second iteration, four major design changes occurred. The first was shortening the overall height of the tracker. This was done because the heliostat (which the tracker will be mounted to) may not always be pointed directly at the sun in regards to elevation. The first iteration had high walls that would block light entering each section if the tracker was not pointed at the right elevation. Thus, the reduced wall height allowed for a greater tolerance in the placement of the heliostat by not needing to be positioned at a proper elevation. The second design change was to the caps. The caps were widened to provide a smaller center slit over the tertiary sensors, which resulted in a higher precision and reduced maximum error in tracking. In addition, "wings" were added to the caps to provide shadows over the secondary sensors unless the tracker was pointed directly at the sun. The third design change was to add a mounting bracket that would also serve as a way to encapsulate the exposed LDR connection wires on the bottom side of the enclosure. Figure 41 is screenshots from the modeled mounting bracket and widened second iteration caps.

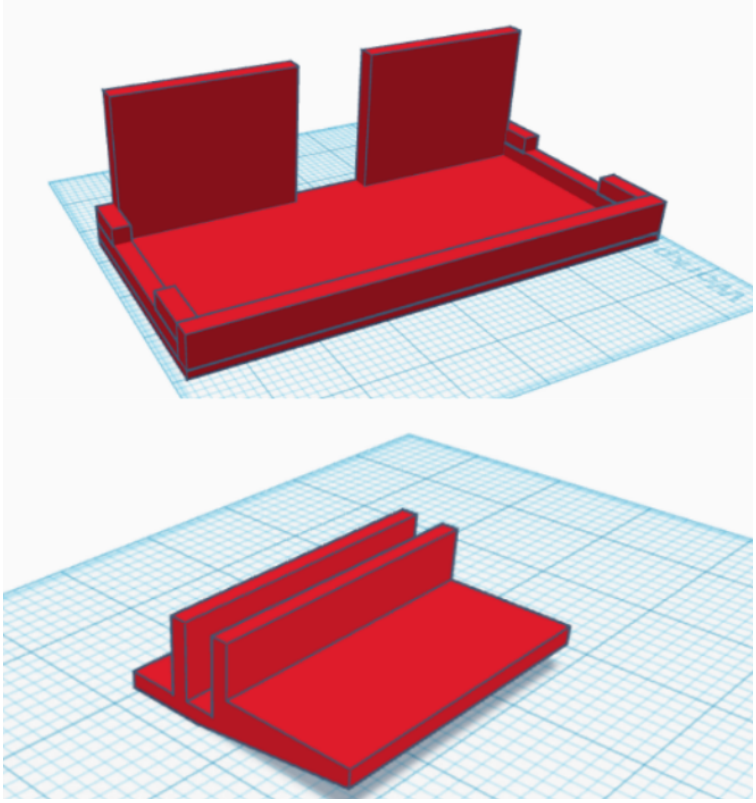

Figure 41: Models of second iteration enclosure components being a mounting bracket (top) and redesigned cap (bottom)

The final design change for this iteration was to paint the entire enclosure gloss white. This was done to better reflect solar radiation and prevent the enclosure from getting too hot and melting or deforming. Figure 42 shows the assembled and painted second iteration with caps installed (without the mounting bracket).

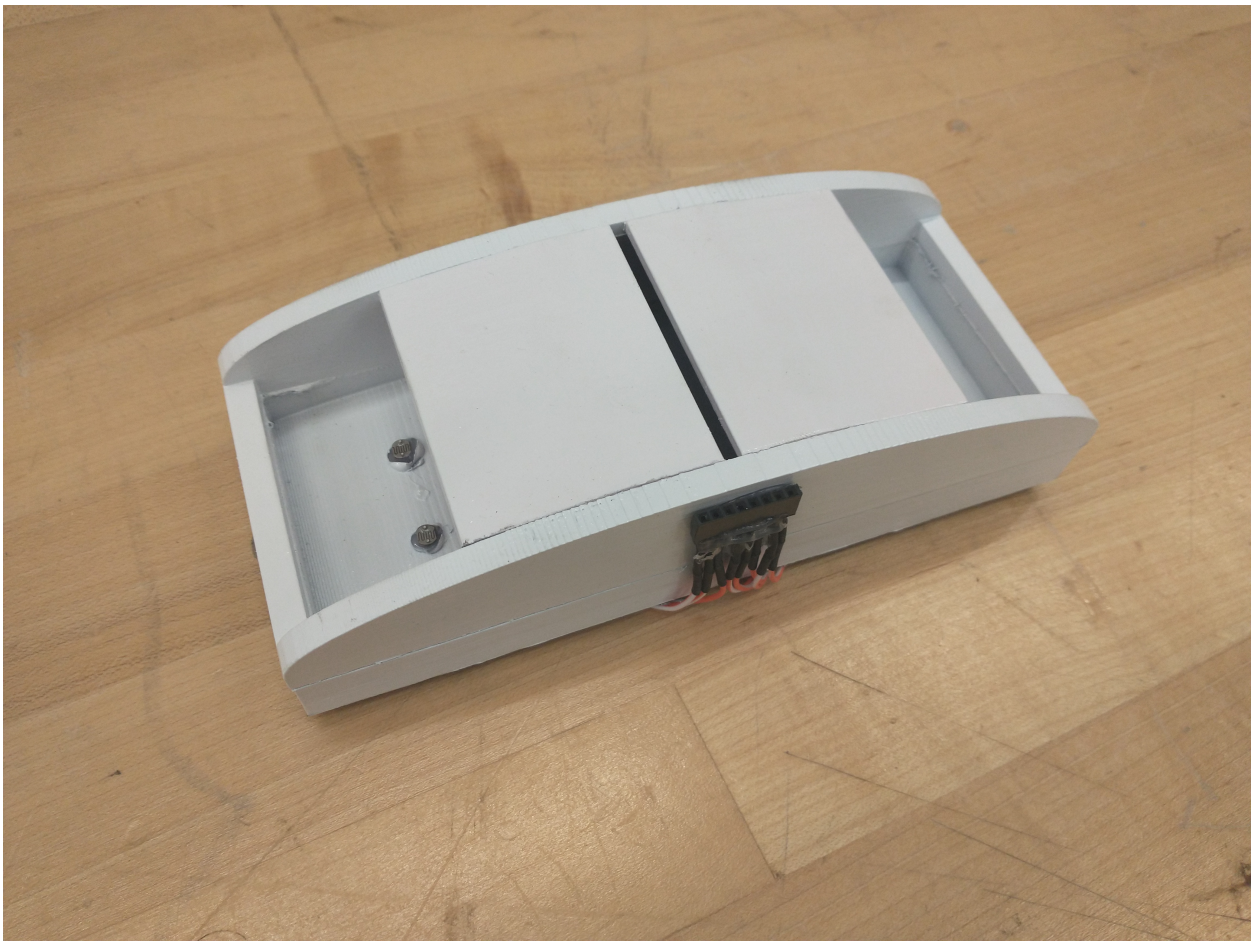

Figure 42: Assembled and painted second iteration enclosure with caps installed

Final Iteration of Enclosure When the solar tracking system was tested outside for the first time, it was clear that both the center slit over the tertiary sensors and the wings over the secondary sensors were not in ideal placements. When the second iteration of caps was designed, they were intentionally made slightly smaller than what was expected to be ideal. This was so they could be extended by adding tape on top of the caps. To find the optimal cap size, the trough was aligned with the sun and then tape was placed on each wing so that the LDRs for the secondary level of sensors was just at the edge of the shadow cast from the wings (this meant the tape overhung the wings by about an eighth of an inch). The center slit was also closed so that it was about a sixteenth of an inch wide. This ensured that the solar tracker was optimized as much as possible so the sun might be tracked accurately within the required precision necessary to achieve maximum efficiency. Figure 43 shows this final iteration of tracker mounted on the trough with the addition of tape to fine tune the size of the caps.

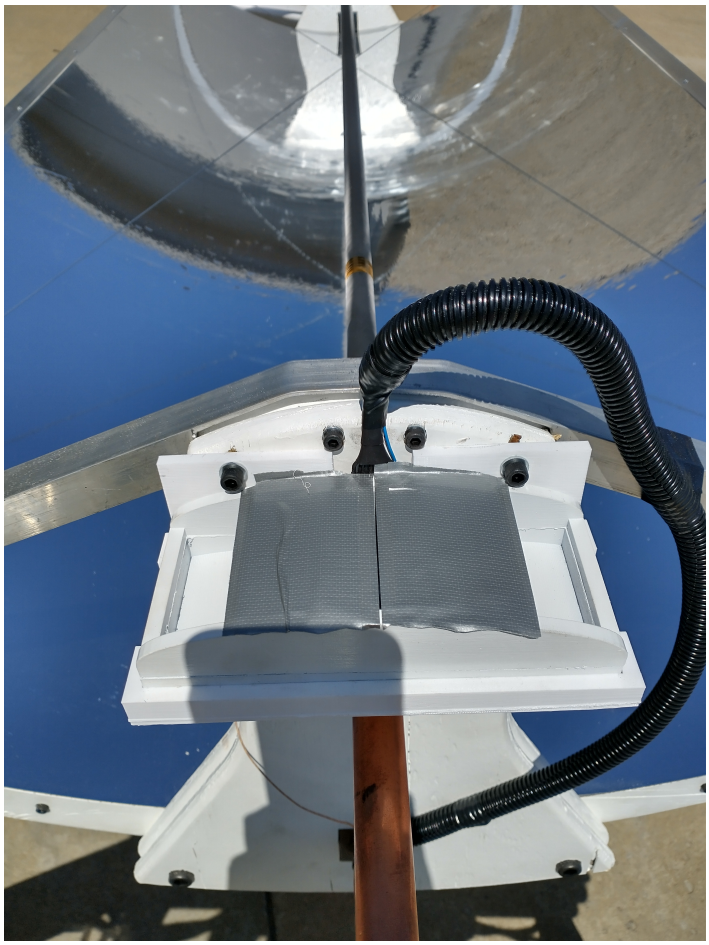

Figure 43: Final solar tracking enclosure mounted on trough

9.1.3 Actuators

To rotate the mirror an actuator must be used to convert electrical signals to mechanical movement. Since the center of gravity was designed to be on the axis of rotation, in theory, the motor required would not need hardly any torque to move the mirror, just enough to overcome the mirrors moment of inertia. However, when constructed the center of mass was not exactly at the axis of rotation and there was also friction the motor must overcome. In addition, the design decision was made to use a worm drive gear set since the gears in this configuration can only be rotated from the input side while the output stays locked in place from the output shaft prospective. This means that it is "auto locking" and no additional braking system needed to be designed. To fulfill the theoretical torque requirements while using a worm drive gear set, a 30:1 ratio NMRV gearbox paired with a 3.0 NM NEMA 23 stepper motor was selected [13]. This configuration can be described with the following block diagram seen in Figure 44 which is the actuator block housed in the control block of the system architecture. It is important to note that for a complete CSP system this block would also house pumps for control of the HTF but that is outside the scope of this project.

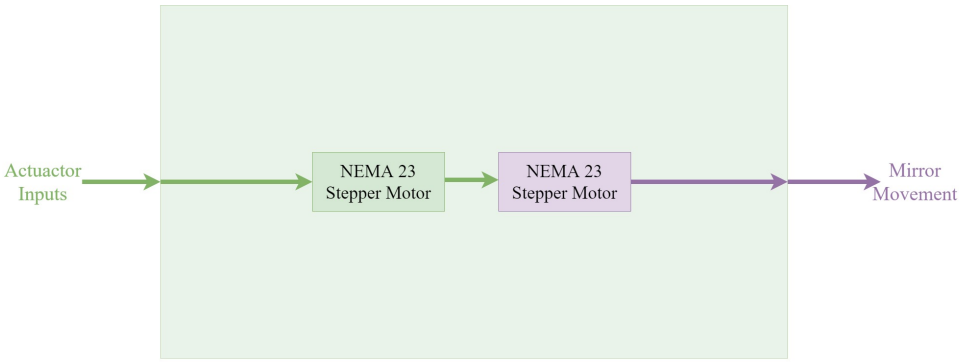

Figure 44: Actuator block

Upon testing of the actuator with the complete system, it was found that the 30:1 gearbox paired with the stepper motor did rotate the mirror. However, when the mirror reached its extreme positions (∼80° and ∼-80°) the actuator did not have enough torque to recenter the mirror. For this reason a design decision was made to switch from the 30:1 ratio gearbox to an 80:1 gearbox. This almost tripled the output torque of the actuator and after testing was found to be more than sufficient in rotating the mirror throughout all possible angles. Figure 45 shows the assembled actuator mounted to the trough.

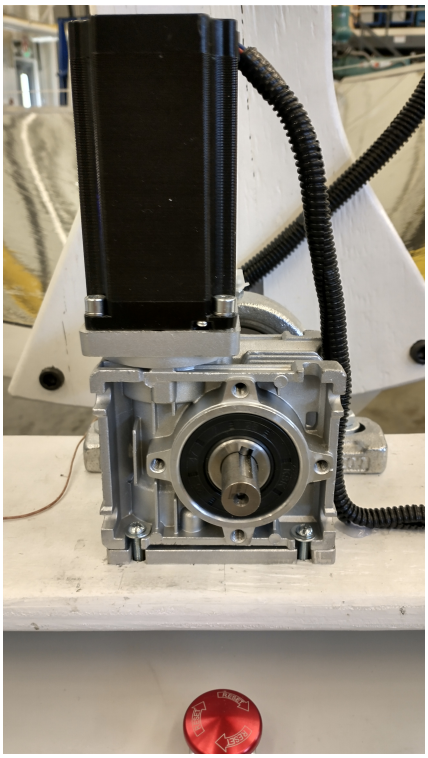

Figure 45: 3.0 Nm NEMA 23 stepper motor connected to NMRV 80:1 wormdrive gearbox mounted to trough shaft

To ensure that the mirror does not over rotate and cause the receiver pipe to hit something, limit switches were implemented on each side of the stand and would be triggered when the mirror reaches its maximum safe angle. Figure 46 shows the CCW limit switch attached to the base where the outer edge of the mirror will contact.

Figure 46: Limit switch to prevent over rotation for CCW direction

9.1.4 User Interface

Though the goal of this project is not to create the most efficient or fluid user interface (UI), this part of the system contains the majority of the electronic components and programming resources. The complexity of the UI is clear when the block diagram is compared to other block diagrams of the same level. Figure 47 shows this block diagram. Following the block diagram is a description of each component that is involved with the UI.

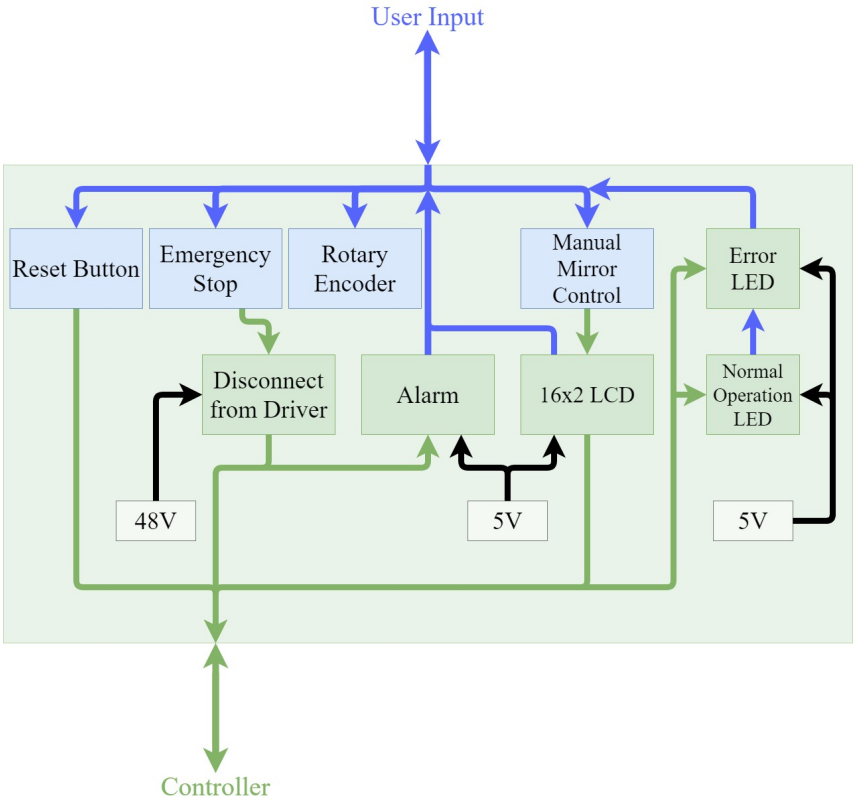

Figure 47: User interface block

Reset Button The reset button is used for a hardware reset of the Arduino. It does not require any code to operate as the Arduino already has a dedicated pin for this function. It is located in the UI as a red push button to the right of the LCD screen.

Emergency Stop The emergency stop both disconnects power to the driver for the stepper motor, as well as sending a signal to the Arduino that the emergency stop has been activated. This is so error messages can be displayed to the user to ensure that the user knows why the system is no longer operating. So, this emergency stop not only stops the code, but it disconnects the 48V supply from powering the driver so there is no possible way for the mirror to move after the emergency stop has been pressed.

Rotary Encoder The rotary encoder has two major functions, the first is to increment the degrees to move by twisting the knob, and the second to enter "manual mode" by using the built-in push button by depressing the knob. The encoder is located on the bottom right of the UI.

Manual Mirror Control Once in manual mode, the mirror can be manually rotated by pressing one of the two push buttons under the LCD screen. The left button rotates the mirror CCW and the right button rotates the mirror CW.

Status LEDs There are two sets of status LEDs, the first blue and the second red. Blue LEDs represent normal operation while red indicates there has been some error or fault.

Alarm The alarm is designed to be used with the red fault LEDs to better help the user realize that something is wrong with the system.

16x2 LCD The LCD screen is the main method of communication with the user. This screen displays what mode the system is in, degrees set to move, error messages, thermocouple readings, etc... Due to this project only being used for research and development purposes at the university, all messages on this screen are in English.

All of these components were fitted in an enclosure separate from that of the microcontroller and power supplies. This was done to move the UI to a location on top of the trough base so that the user can better interact with it. A 25-pair (50 wire) cable was used to join this UI enclosure to the main enclosure (see Figure 51). There are plenty of wires left in this cable for letter refinement and expansion of the user interface. Figure 48 shows the completed UI in its final enclosure with all components listed above.

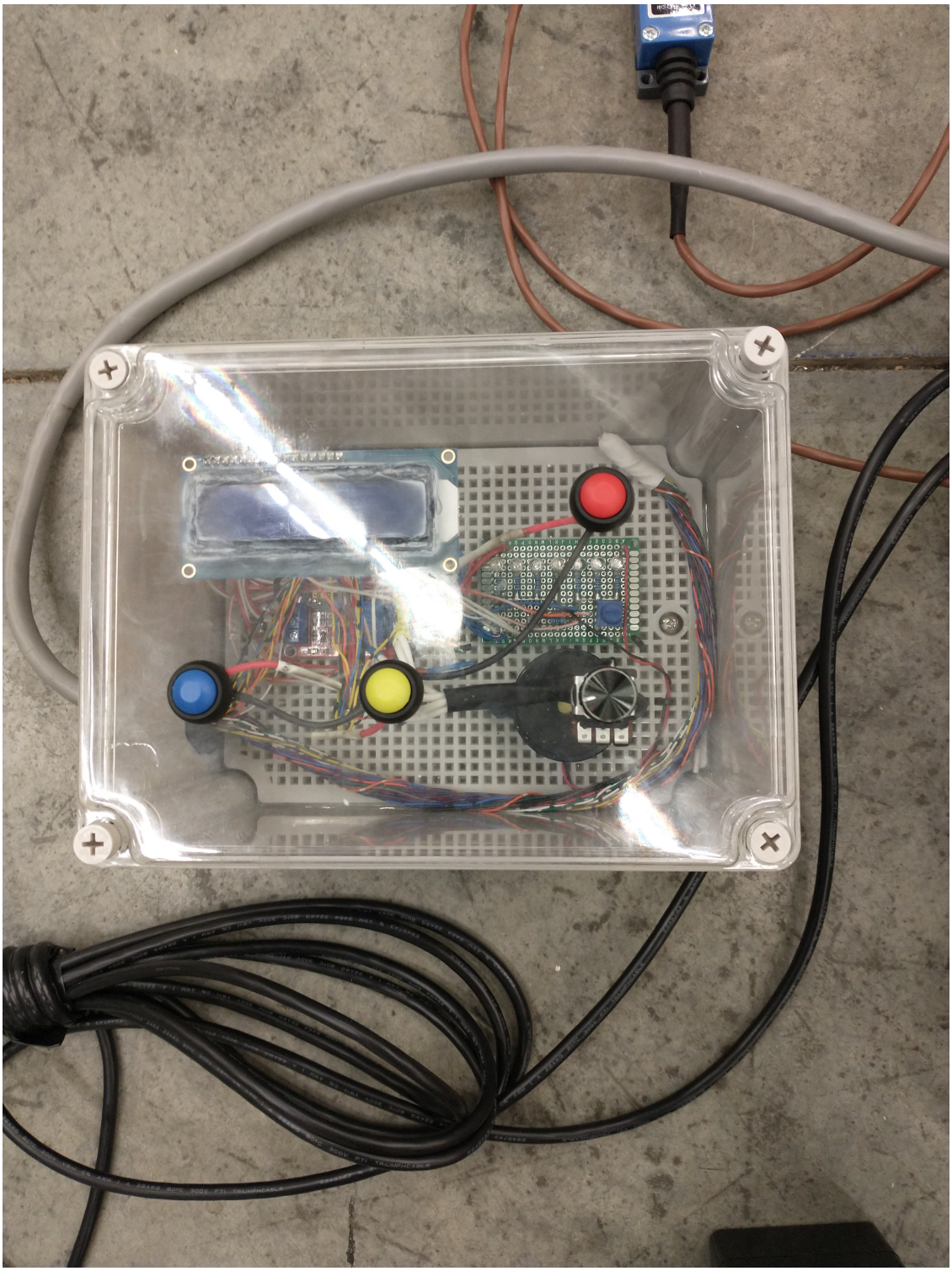

Figure 48: Completed user interface in final enclosure

9.1.5 Hardware Integration

To assemble all components of the control system together, a technical schematic was drawn using Fritzing software. This schematic was drawn after testing the components on a breadboard to ensured configurations and connections were correct. This gave confidence in cutting wires and permanently joining devices together as the proper connections were already confirmed. Figure 49 is this technical drawing.

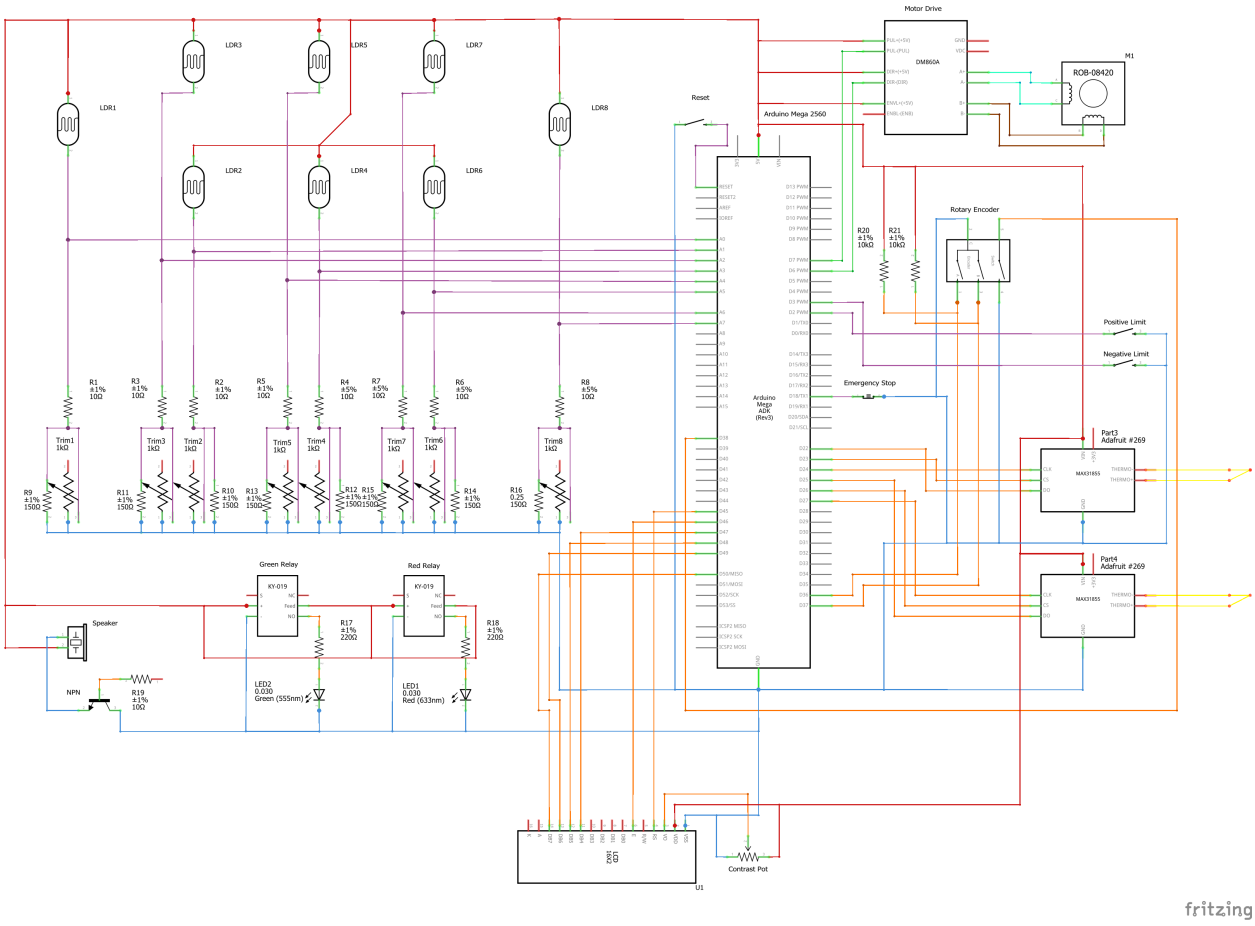

Figure 49: Technical schematic

Although this schematic is fine for those savvy in reading such diagrams, it is not very clear to the lay person. For this reason another schematic was made that describes the same information as the previous schematic, but does it using illustrated representations of each component rather than technical drawings. The main purpose of this schematic is to help bring a clear picture of how the system is connected for those that are interested in this project without a technical background. This schematic is represented in Figure 50 below.

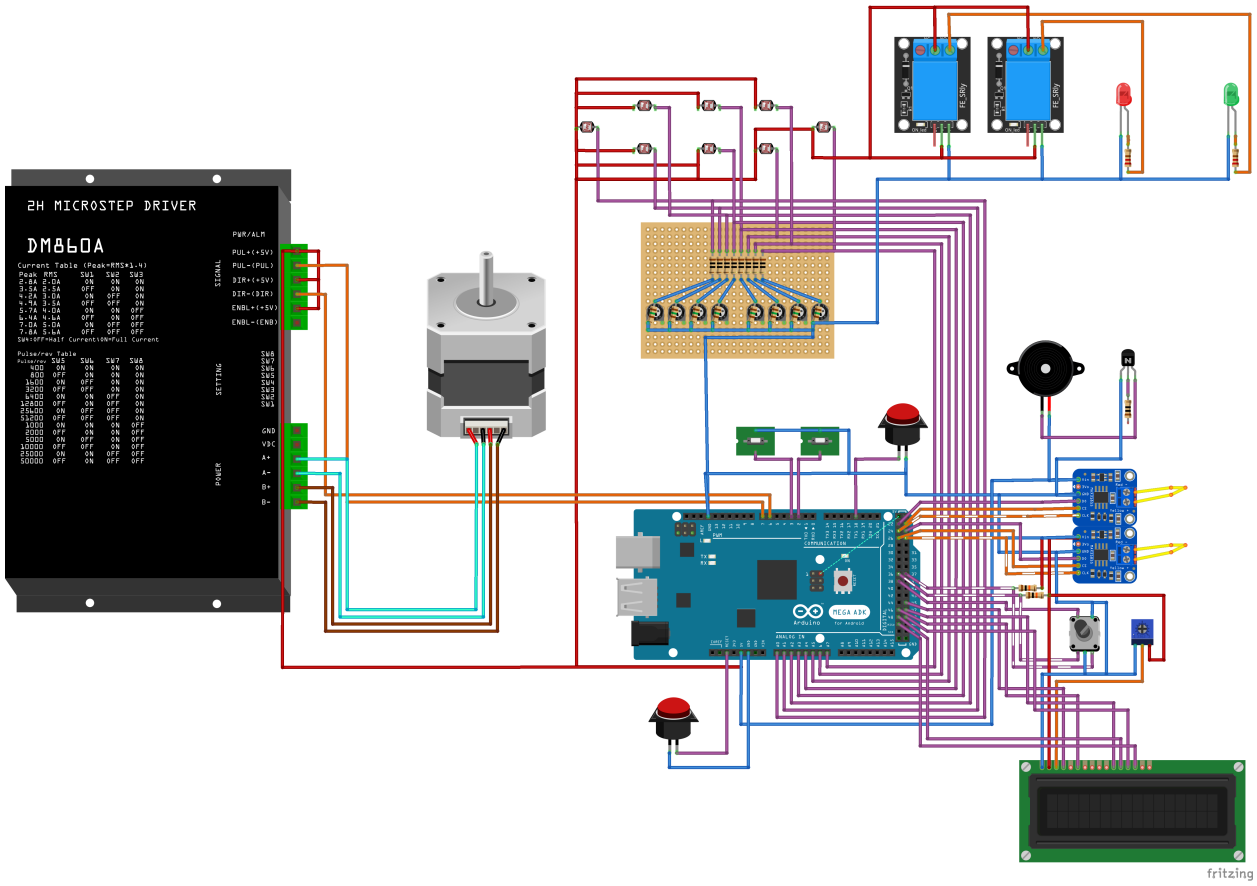

Figure 50: Symbolic schematic

The control system was assembled based on these schematics with all electronics placed in waterproof enclosures to ensure the equipment could withstand the elements when placed outside in harsh environments. Both the main enclosure and UI enclosure are IP67 rated boxes. All penetrations into the enclosures were made with cable gland waterproof cable connectors for both strain relief and waterproofing. The complete control system can be seen below in Figure 51.

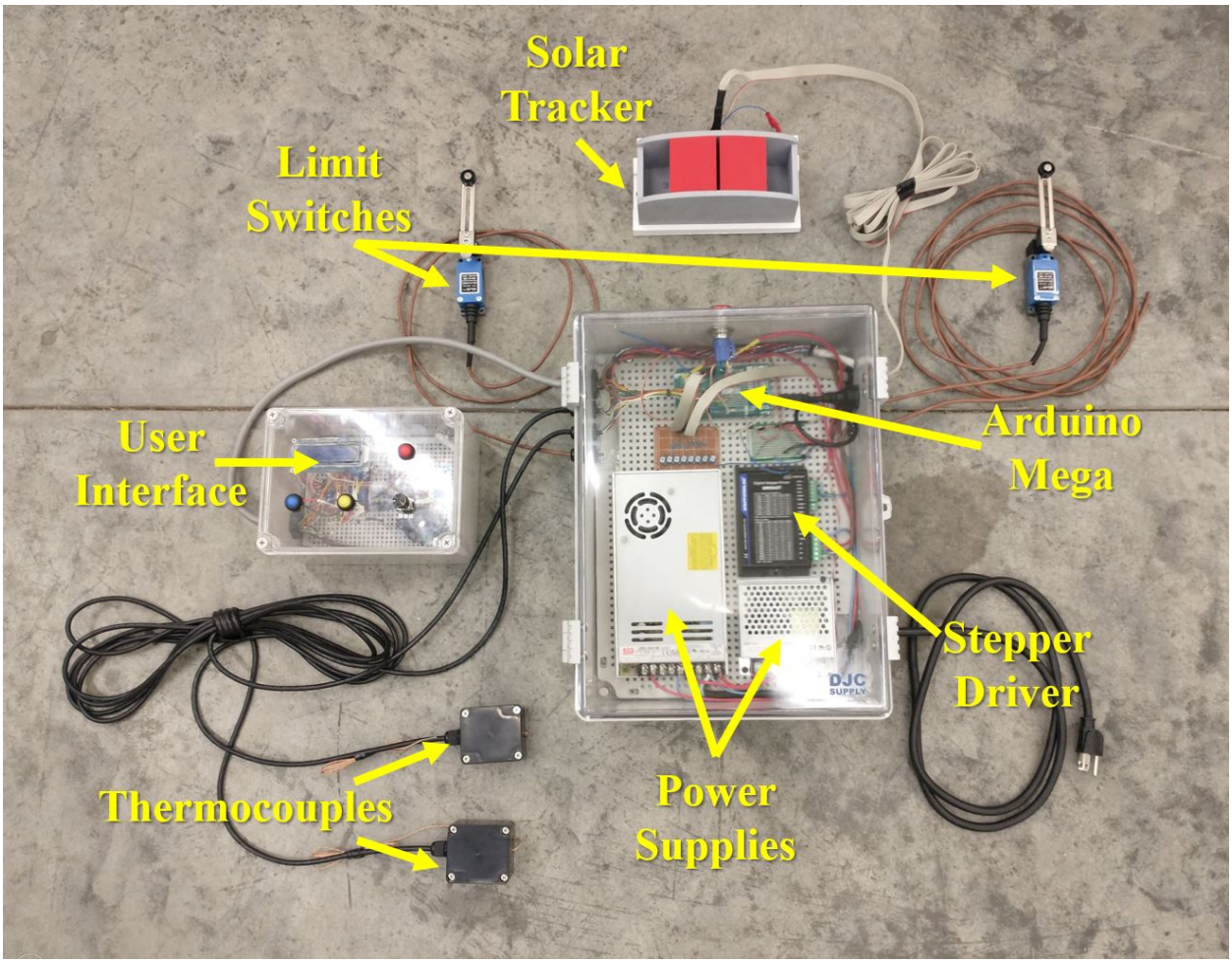

Figure 51: Entire control system before mounting on parabolic trough (missing stepper motor)

After assembly of the control system, it was mounted to the parabolic trough base. All wires and enclosures were secured and wires were shielded with automotive wire loom. This was to protect the cables from prolonged direct exposure to sunlight that might deteriorate the exterior insulation. Figure 52 shows the finished control system mounted to the parabolic trough.

Figure 52: Entire control system after mounting on parabolic trough

9.2 Programming

9.2.1 Configuration

The code to operate the system starts with many "define" and variable declarations. These are necessary to help make the code readable since "2" is not near as intuitive as "CWLimitPin." Below is a list of the various peripherals and functions that required definitions and variable initialization in order of how they appear in the code.

- LCD Screen pins
- Encoder pins
- Thermocouple 1 and 2 pins
- Tracking Buffers
- LDR pins
- LED pins
- Push button pins
- Stepper motor pins
- Stepper motor speed
- Interrupt pins
- Various global variables
- Tracking variables
- Variables for delay counters

Next, all peripherals must be properly configured. For some peripherals this requires using a preexisting library and inputting the pin locations (like for the LCD), but for other each pin much be manually assigned (life for the rotary encoder). The following is a list of the peripherals that require setup configurations in the order they appear in the code.

- Encoder
- Manual mode buttons
- LEDs
- LCD screen
- Thermocouples
- External interrupts

9.2.2 Main Loop

The main loop starts directly after the setup and configuration. The main loop is a never ending loop that continues nonstop until power is disconnected from the Arduino. All other functions are called from this loop and the code will return to this loop after the function has been executed.

1. The first step of this loop is to turn on the blue LED status lights and turn off the red LED status lights; this is to show normal operation to the user.
2. Then a comparison is made to a stored value from the *millis* function which is an internal clock. If it has been longer than 0.5 seconds since this last stored value, the eight LDRs will be read again (**note:** though not displayed in the flowchart, this same process is used for reading the thermocouple values but at a 1.5 second interval).
3. These readings will then be stored. If the LDR is in a "bin" with another LDR, the average value of these two will be stored; this is done to reduce anomalies.
4. Next, the encoder button is checked to see if it has been depressed. If it has, the code will call the *ManualMode* function. If not, the code continues and begins to check the tracking data to see if it should move the mirror.
5. The first step of this process is to determine if the primary sensors indicate that the mirror should be rotated. This is done by seeing if the one side has a higher value than the other side PLUS a predefined buffer. The buffer is used to ensure that inconsistencies in LDRs does not create a false positive. The coarse buffer is set to 100 as that is a large margin of error. It is not expected to have values of these two sensors to be further apart than 100 units unless the mirror is substantially misaligned with the sun.
6. Once the primary sensors have done their job and they are both within a value of 100 compared to each other, the secondary and tertiary sensors are compared. This comparison is not as straight forward. It starts in the same way as the primary sensors with each secondary sensor being compared to the opposite sides secondary sensor plus a buffer (this buffer is set to 30). That is not the only criteria for movement however, The center bin (or tertiary sensors) is then compared to each of the secondary sensors. For the controller to decide to move the mirror, in addition to the comparison of the secondary sensors to each other, the tertiary sensors minus a buffer of 10 must be less than *at least* one of the secondary sensors. This ensures that is the tertiary sensors are illuminated, which can only happen if directly pointed at the sun due to the enclosure design, the mirror does not move *even if* the secondary sensors think it should.
7. After these tracking comparisons if any of them are true the main will call the appropriate *Rotate* function, if all are false, the code loops back to the start of the main function.

Figure 53 is a flowchart of the main loop. This figure also shows a diagram of the solar tracker with the variable names for each of the "bins" below it.

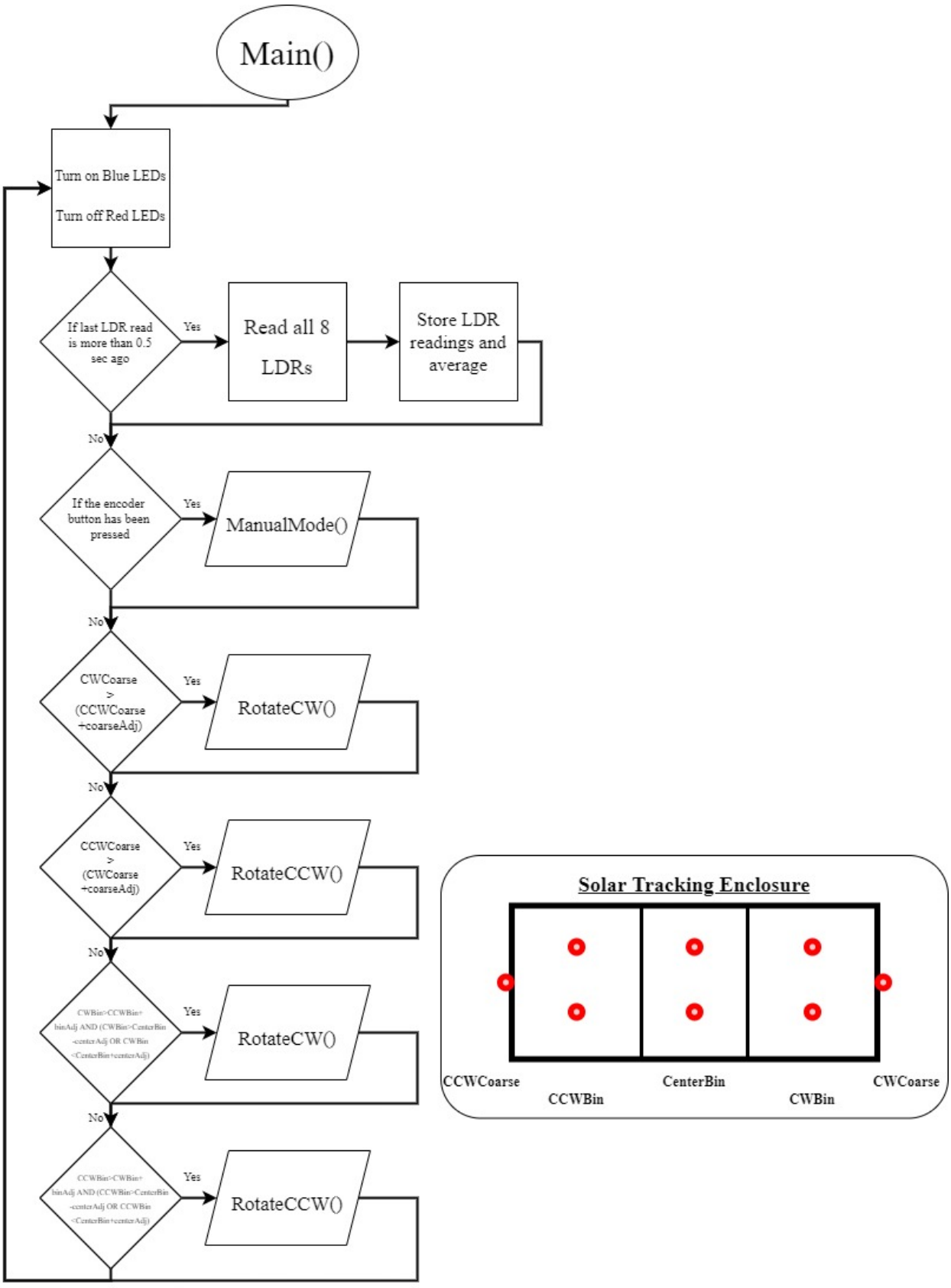

Figure 53: Main loop flowchart

9.2.3 Manual Mode

After the *ManualMode* function has been called by the main loop with the press of the encoder button it will loop until the encoder button is pressed again. On each loop of this function the encoder button is read to see if it is pressed and the direction of the encoder is also read. If the encoder has been rotated a count will be incremented and displayed on the LCD screen. This count represents the desired degrees to be moved and once the encoder button is pressed again the mirror will move these degree and then return from the function (**note:** this part of the function is not rotating the mirror as of writing this report). This function also reads the push buttons on the user interface for CW and CCW movement. If either of these buttons are pressed the appropriate *Rotate* function is called.

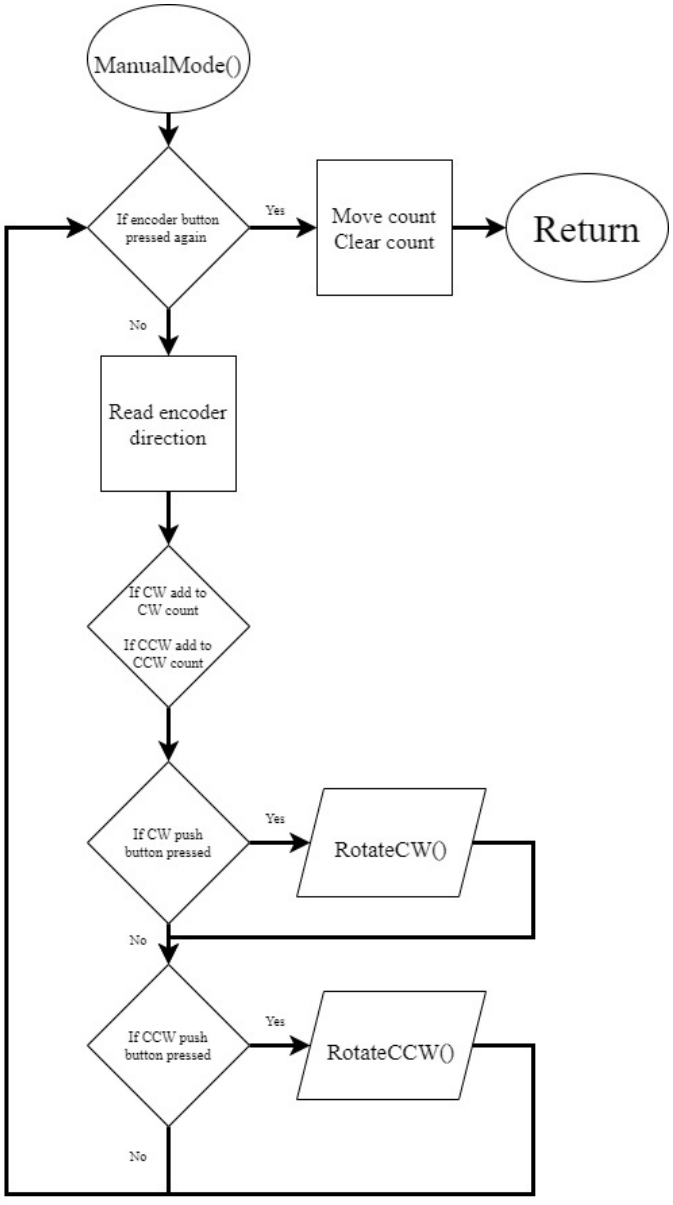

Figure 54: Manual mode function flowchart

9.2.4 Rotate

The *Rotate* function is responsible for sending commands to the stepper driver to rotate the motor and move the mirror. This function is very simple, it first configures the direction pin to drive the motor in the requested direction. It then sets the pull pin high, delays for the variable *pd* (for pulse delay) microseconds, then clears the pull pin and delays the same period of time once more. The code is then returned. The pulse delay is what sets the speed of the motor, the lower the pulse delay, the faster the motor spins. Figure 55 is a flowchart for this function.

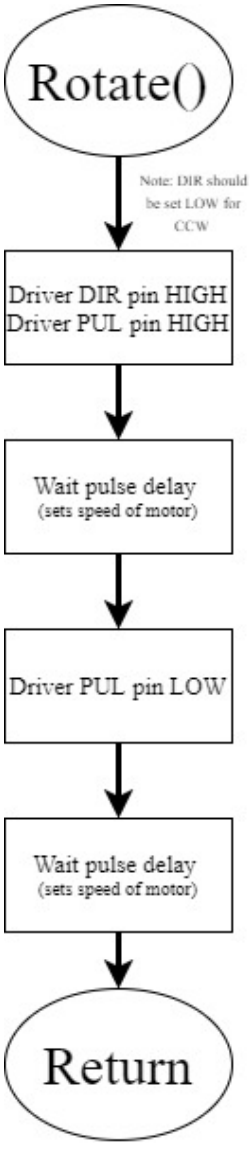

Figure 55: Rotate Function flowchart

9.2.5 E-Stop and Limits

Both the emergency stop and limit switch functions operate almost identically. They are both called by an external interrupt pin being triggered. Once called, the blue LEDs are turned off and red LEDs turned on, then the code gets stuck in a never ending loop that prints an error message to the LCD screen. To escape this the reset button must be used. The main difference between the operation of these two functions is that the emergency stop turns off interrupts so that nothing can override the error message, not even if a limit switch is hit. Figure 56 and 57 are flowcharts for both of these functions.

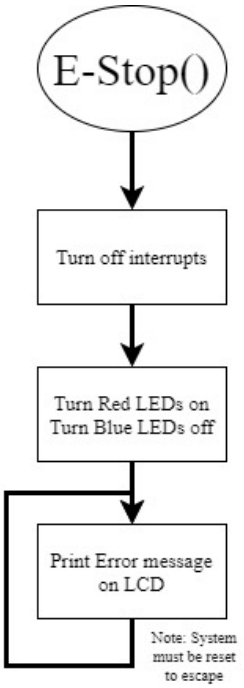

Figure 56: Emergency stop interrupt flowchart

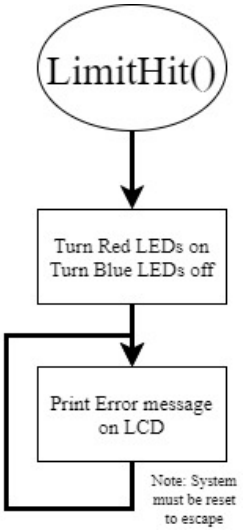

Figure 57: Rotation limit switch function flowchart

9.3 Control Testing

9.3.1 Bench Testing

The majority of testing for the control system occurred on the lab bench. This was because all peripherals needed to be tested using a breadboard to ensure that wiring was correct and that basic code operated the devices properly. An example of this was reading multiple LDR sensor measurements. This was done by wiring 4 LDRs on a breadboard and then putting tubes over each of the LDRs to make covering them easy. Then the serial plotter on the Arduino was used to plot the values obtained from the ADC connected to the LDRs. Proper operation could be confirmed by cover each tube one at a time and seeing the value on the plotter go lower than the rest. Figure 58 shows this testing taking place.

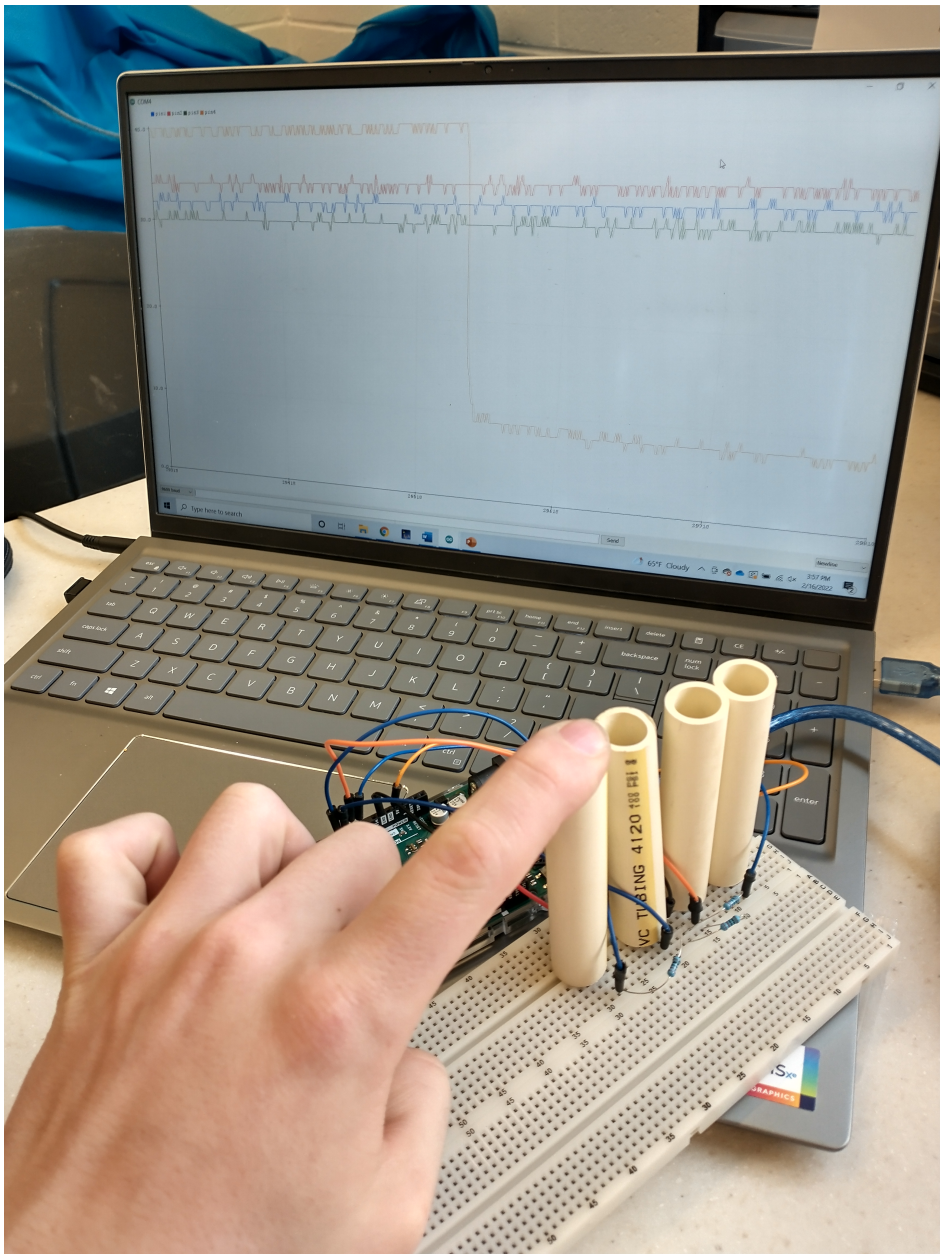

Figure 58: Four LDRs in tubes connected to Arduino with plot on computer

Since the value of R_{LDR} changes how much light will cause the ADC to saturate, it was necessary to determine the ideal value for that resistor so that even in direct sunlight the ADC does not saturate, or get close. To find the value that kept the measured value around 600-700 when pointed directly at the sun, an LDR was soldered to a small perf board and in place of an R_{LDR} resistor, a potentiometer was used. Then this test bed was taken outside and pointed at the sun. The output value was manipulated by adjusting the potentiometer until the measured value was in the 600-700 range. This resistance was then measured and found to be 180Ω. Figure 59 shows this test bed along with a graph showing what happens when the resistance is varied (the vertical axes on this plot maxes out around 550 since it is scaled).

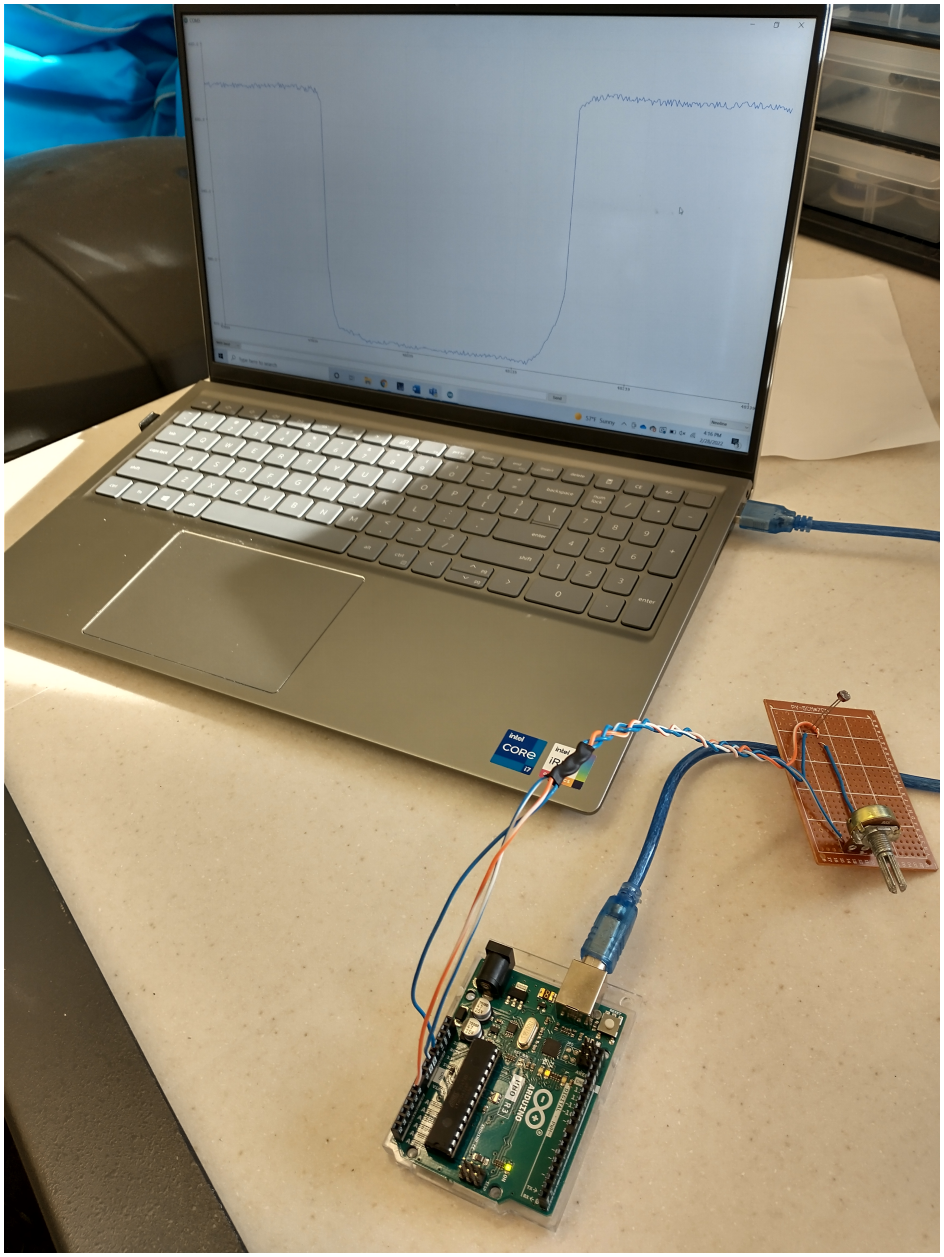

Figure 59: Test bed to determine value of R_{LDR} connected to Arduino with plot on computer

As can be seen in Figure 58, even though all four LDRs are exposed to the same amount of light with the same R_{LDR} resistors, the measured values varies between them. With this in mind, it is important to note the scale on the plot, as values are maxing out around 40 when the ADC has a max value of 1023, overall these values are still within reason. However, it was unknown when populating the perf board with the R_{LDR} resistors whether or not this would have an effect that caused measurement errors in the senors down the line. In an attempt to fix this problem, trim potentiometers were used in parallel with the R_{LDR} resistor to fine tune the resistance value if necessary. Upon final implementation it was discovered that these available adjustments were unneeded. Figure 60 shows the resistor board.

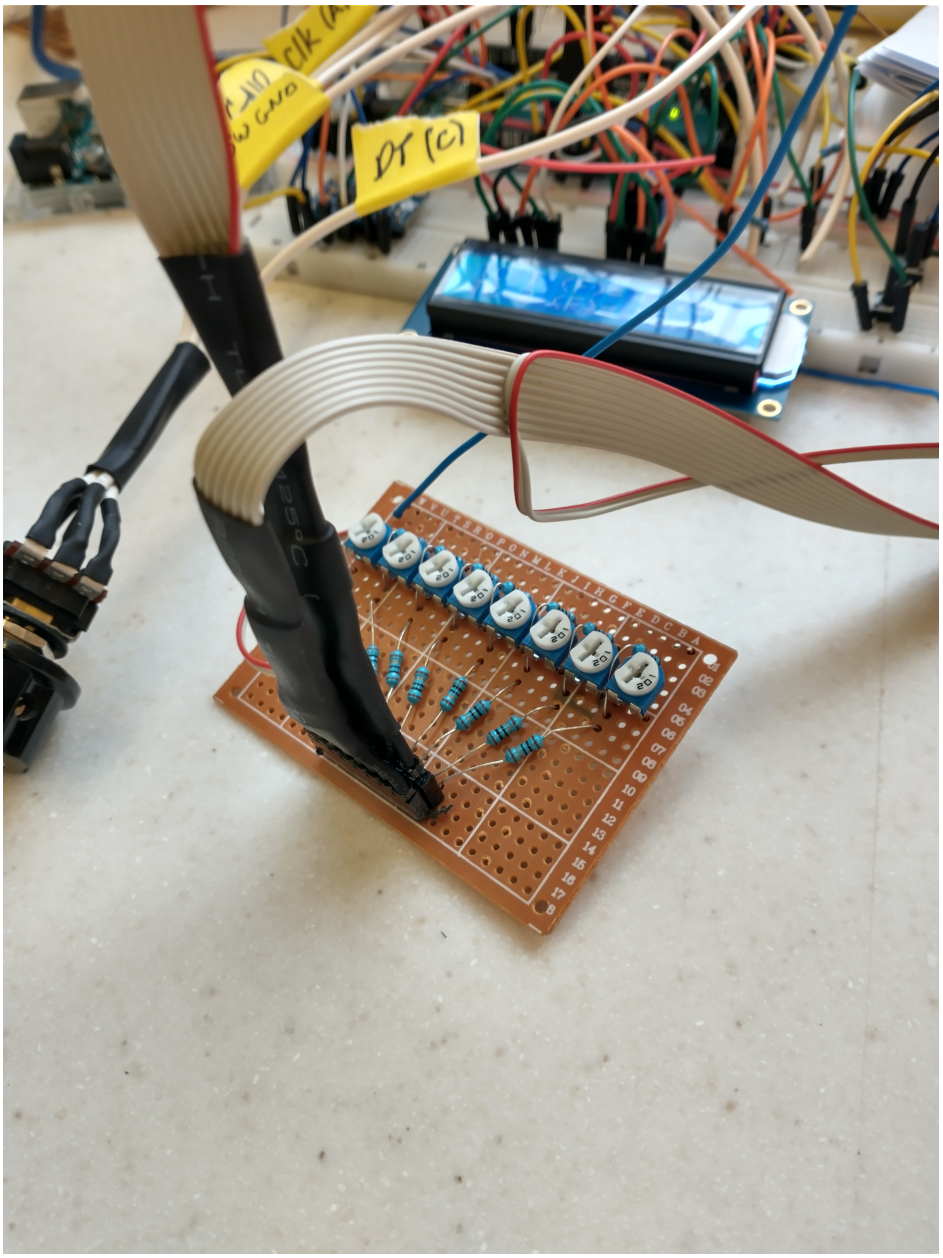

Figure 60: Board with all eight adjustible R_{LDR} resistors

Further testing with the LDRs was completed once all eight were in the tracking enclosure. Figure 61 shows an example of this testing.

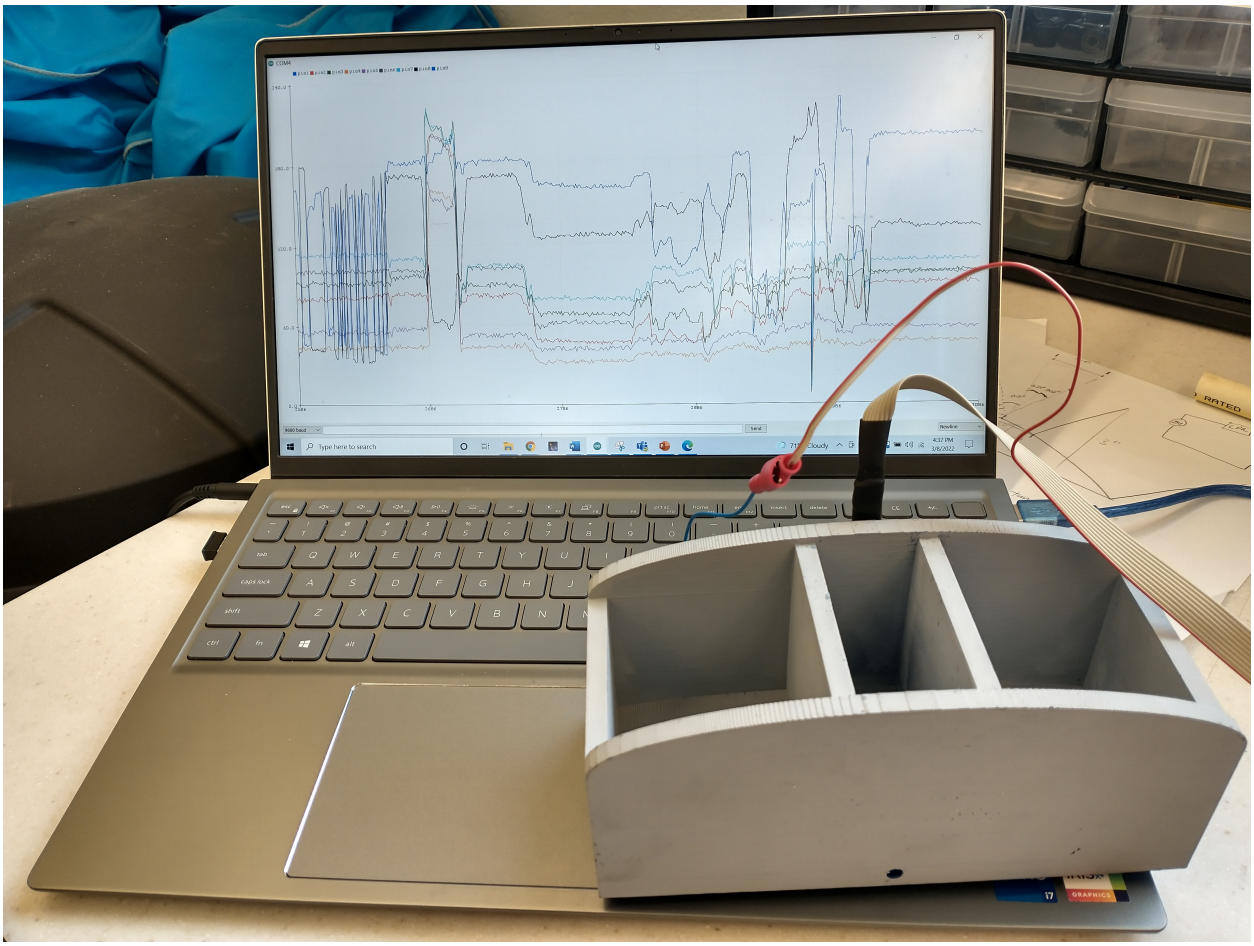

Figure 61: First iteration tracker connected to Arduino with output of all eight LDRs on plot

Besides individual components such as the LDRs being tested, a system integration test with the majority of final components was also done while still using a breadboard on a lab bench. This helped to give more confidence in creating a schematic for the complete system before implementation. Figure 62 shows what this bench testing looked like.

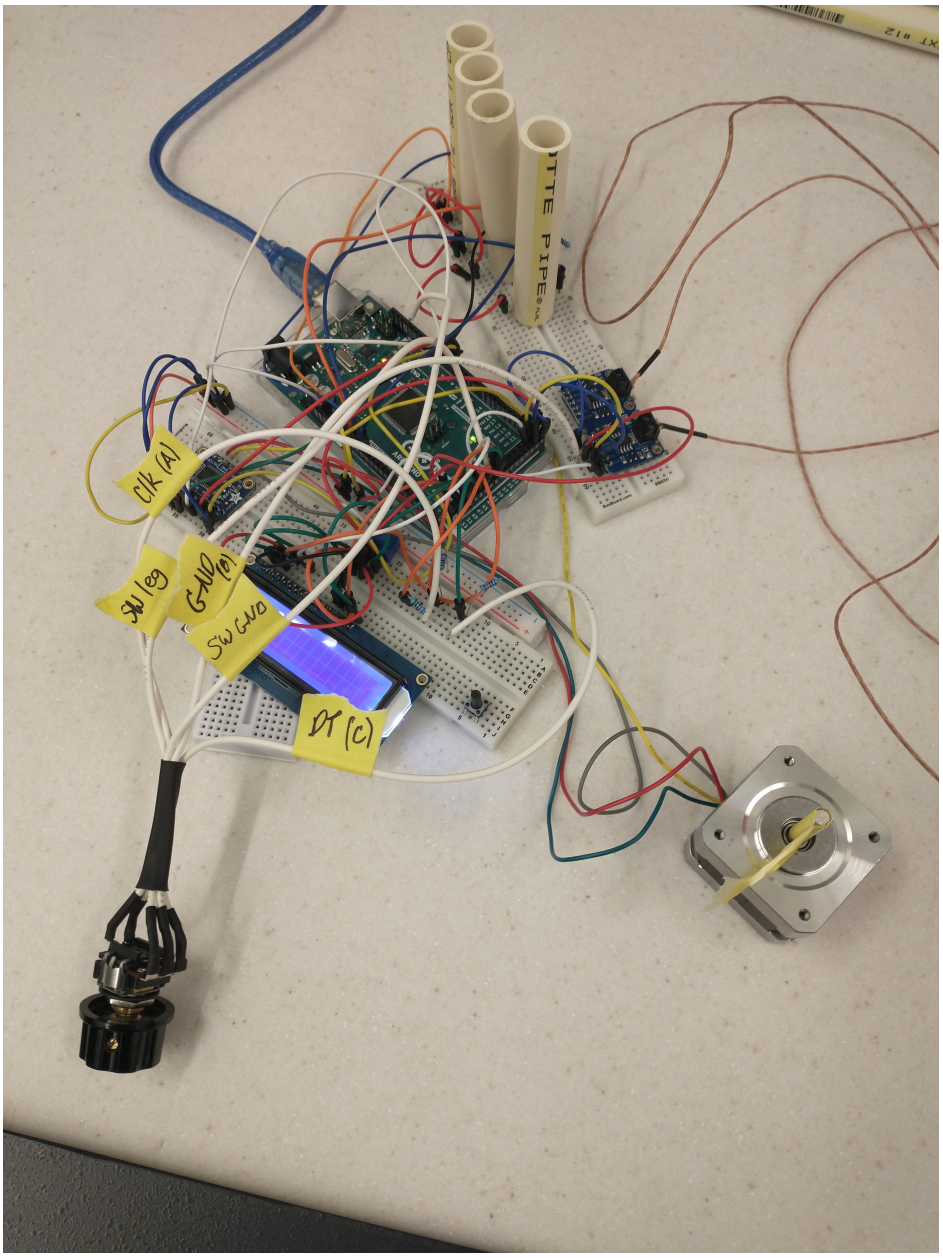

Figure 62: Majority of control system components connected on breadboard for system integration testing

9.3.2 Testing Individual Hardware Components

After all components had been permanently affixed in their respective enclosures, testing was performed on each component to validate proper connection and configuration. The test procedure for this process involved using short segments of code to only test the peripheral of interest and nothing else. This reduced troubleshooting time because all error could be attributed to the device being tested. An example of what this test would look like is in Figure 63 where the LCD is being tested by only displaying "hello world" and a counter.

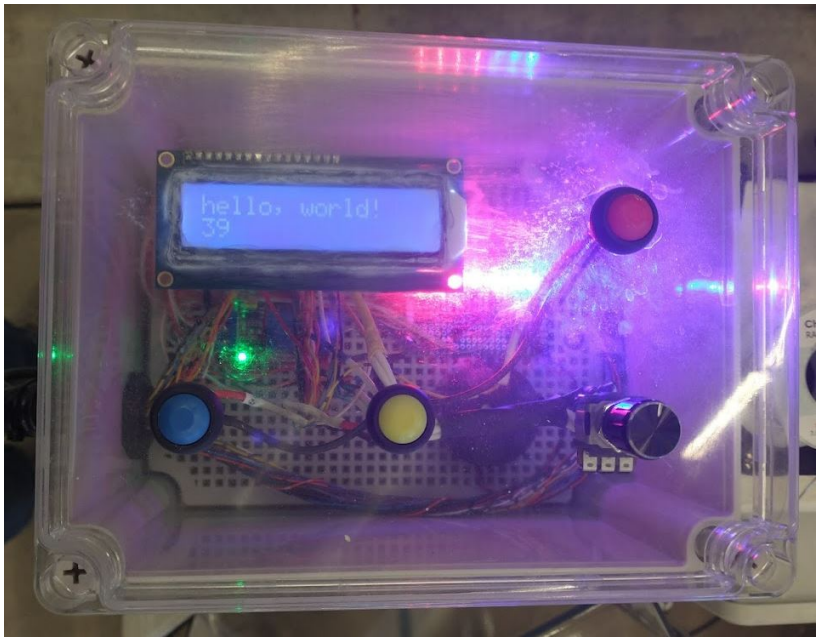

Figure 63: Individual hardware test for user interface

This process was accomplished for every control component possible and a checklist made to systematically verify the operations of the subsystems. The completed checklist is below in Table 7.

Component	**Test Result**
CW Button	Passed
CCW Button	Passed
Reset Button	Passed
Encoder Button	Passed
Encoder	Passed
Thermocouple 1	Passed
Thermocouple 2	Intermittent
LCD Screen	Passed
8 LDRs	Passed
Stepper Motor	Passed
E-Stop	Passed
Alarm	Failed
Red LEDs	Passed
Blue LEDs	Passed
CW Limit Switch	Passed
CCW Limit Switch	Passed

Table 7: Test results for individual hardware components and respective Arduino configuration

There are two devices that had issues in this testing. The first being that one of the thermocouples would only read intermittently, and the second was that the alarm would not make any noise. Since the thermocouple problem was intermittent, it was hard to troubleshoot and eventually ignored. As far as the alarm goes, this was the only device not tested on a breadboard before integration. For this reason there are too many unknowns of why it failed. It could be the device, wiring, code, etc... Since the alarm is not a critical component in the system it was left as nonoperational.

10 Results

The window for testing was fairly short due to time constraints and inclement weather, so results are noted as baseline findings only. The system was moved outside for testing via wheel dollies. The trough was aligned with the controls enclosure side of the system pointing north, and tracking occurring from east to west. Basic manual functions were tested first for safety, with a fire extinguisher readily available in case of fire. Once all manual functions were confirmed, automatic sun tracking was started. The trough was purposely aimed away from the sun and allowed to move automatically to find the position of the sun. The system completed this test without fail repeatedly. The system was then left in auto with data being gathered for temperature and tracking position every 15 minutes. This test duration was approximately 2 hours long. Table 8 shows the results for temperature. Note higher temperatures were seen at the 4:00 mark because the trough was moved out of the shadows of trees and aligned more precisely north and south.

Temperature Readings (°F)

Time	Ambient	Max	Min	Mean	Above Ambient
2:30	59	355	185	270	211
2:45	58.1	369	203	286	227.9
3:00	57.5	350	202.1	276.05	218.55
3:15	58.5	331	210.6	270.8	212.3
3:30	56	316	187.2	251.6	195.6
3:45	53.6	262	143.1	202.55	148.95
4:00	54.9	357	322	339.5	284.6
4:15	56.8	374	352	363	306.2
4:30	55.4	**394**	313	353.5	298.1

Table 8: Temperature results for parabolic trough April 19th 2022

Table 8 shows that a maximum temperature of 394 °F was recorded with an average temperature of 290 °F. These results are not a thermodynamic assessment of the system but do provide some baseline values for the system. A computational thermodynamic analysis would need to be completed to get a true idea of what the system is capable of, but that is outside the scope of this project.

On the day of testing, sun tracking proved to be both accurate and reliable. As stated before, automatic solar tracking was used for the entire two hours data was taken with zero issues that required attention from the operator. Two methods were used to confirm and quantify solar tracking. The first was by using azimuth and altitude (elevation) data from suncalc.org to mathematically determine where the sun should be relative to the ground. The measured results were within 4 degrees of the theoretical position [14] . *However,* the trough was not aligned correctly along the North to South line since this alignment was only "eyeballed." Thus, these results are not valid more than a rough estimate. The second verification method was more reliable with fewer measurement variables. It was accomplished by using shadows cast on the trough that changed based on how the trough was pointed at the sun. These shadows were measured and trigonometry used to determined the tracking error. For a period of thirty minutes and 8 degrees rotated, the maximum tracking error was 1.00 ± 0.10 degrees and the average tracking error was 0.655 ± 0.10 degrees. This was well within the goal of 2 degrees maximum error.

11 Conclusions and Recommendations

11.1 Lessons Learned

This project had many difficult tasks to overcome which required ingenuity, experimentation, and continual learning. The largest challenge was creating the reflective mirror. More specifically, finding a suitable adhesive to bond the mylar film to the Lexan substrate proved problematic. If a low cost substrate is used with a thin film in the future, then more research will be needed to find a long term bonding solution. Finding an all-in-one cost effective solution that doesn't require adhesives is advised.

Another lesson learned had to do with scalability. Research showed that using plywood for smaller troughs had been completed successfully. However, the size of this system was scaled up considerably from these examples which created mechanical issues that were not present in the smaller troughs. This project had torsion and rigidity issues that had to be solved after design was mostly complete. If a mechanical engineer was added to this project, this issue might have been avoided during design through a finite element analysis of the system.

Due to not testing the alarm on the breadboard before final integration, the troubleshooting process was much more difficult since there were so many possible variables and failure modes, none of which could be easily eliminated. Of all the mistakes made, this was one of the easiest to fix in hindsight and the only reason it was not done correctly to begin with was to save time, but that ended up costing more time. This is a key lesson learned, it is always better to do the last ten percent of testing before making anything permanent because the small amount of time you save might turn into a large amount of time lost.

11.2 Future Recommendations

There are many promising uses for this project in the future. The most obvious next step would be creating a fluid loop. A fluid loop would consist of flexible pipes that connect to the receiver, in which a heat transfer fluid would be cycled through. The fluid that is chosen will dictate the size, materials, and components that would be necessary. If air is used as a heat transfer fluid, a cycling system consisting of fans and values could be used to cycle the air at a given rate. Other fluids like water or synthetic oil would require pumps instead of fans. The cycling time of the fluid would need to be calculated with a thermodynamic analysis of the system. Careful consideration of pressure buildup in a closed loop system should not be overlooked, as a pressure vessel could turn into a safety hazard if not properly designed.

After a fluid loop is created, a basic thermal storage system could be implemented. If air is used as a heat transfer fluid a packed bed of quartz or sand may prove to be a useful option. Other storage systems may use phase change materials for better thermodynamic efficiency. The choice of heat transfer fluid will again dictate the type of storage system that is possible. A well insulated storage system is required for CSP projects. Special attention to convective losses should be a top priority.

Another idea is to redesign this project using stiffer materials like fiberglass, carbon fiber, or stamped metal. Stamped metal construction seems like a great choice of material. The same 3D model of this project could be used with the material changed. Solidworks has a feature where sheet metal can be shaped metal can be flattened easily. Regardless of material choice, a material with a higher Young's modulus is desired and would likely fix some of the torsion issues that were encountered during the assembly of this project. Another approach would be designing a torsion tube or torsion box under the trough assembly. This approach is how many commercial size installations are designed. These ideas could fix the static load issues that this project faced.

There are dynamic loads to consider as well, with wind being the largest factor. A finite element analysis and dynamic wind loading of this system would also be a great future project. This type of

project would need to be completed on a more rigid adaptation of this project using a stiffer frame however.

Another great addition to this project would be something that could harness the heat and convert it into another form of energy such as mechanical or electrical energy. A sterling engine could harness the heat generated from the parabolic trough and use it to create rotational mechanical torque. This could be coupled with a small generator to create electricity as well.

All sections of the control system can be improved on but the solar tracking system has endless possibilities to be refined since it is still a novel design. The next design iteration could focus on fewer LDRs, smaller size, being waterproof, better connections, easier assembly, or many other aspects that would make the overall design much more efficient or cost effective.

References

[1] [Online]. Available: https://www.energy.gov/eere/solar/articles/solar-two-tower-system

[2] P. Heller, "1 - introduction to csp systems and performance," pp. 1–29, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780081004470000018

[3] T. N. C, "Global review of solar tower technology," 2014. [Online]. Available: https://www.seriius.org/pdfs/global-review-solar-tower-technology.pdf

[4] C. K. Ho and B. D. Iverson, "Review of central receiver designs for high-temperature power cycles (paper)." 10 2012. [Online]. Available: https://www.osti.gov/biblio/1141055

[5] [Online]. Available: https://www.sciencedirect.com/topics/engineering/steam-rankine-cycle

[6] H. Price, E. Lu¨pfert, D. Kearney, E. Zarza, G. Cohen, R. Gee, and R. Mahoney, "Advances in Parabolic Trough Solar Power Technology ," *Journal of Solar Energy Engineering*, vol. 124, no. 2, pp. 109–125, 04 2002. [Online]. Available: https://doi.org/10.1115/1.1467922

[7] [Online]. Available: https://www.energy.gov/sites/default/files/styles/photo_gallery_509_x_678_/public/2020/09/f78/19881.JPG?itok=Chr5PygA

[8] S. E. T. Office, "Dish/engine system concentrating solar-thermal power basics." [Online]. Available: https://www.energy.gov/eere/solar/dishengine-system-concentrating-solar-thermal-power-basics

[9] D. Mancera, M. Schroedter-Homscheidt, and L. Klüser, "Aerosol estimation in the lower planet boundary layer for solar power tower plants' assessment," 06 2014.

[10] [Online]. Available: https://www.yellowlite.com/blog/post/small-scale-concentrated-solar-power-the-how-why-what/

[11] [Online]. Available: https://www.e-education.psu.edu/eme812/node/557

[12] [Online]. Available: http://boomeria.org/physicstextbook/ch13.html

[13] "Standards for motors and generators." [Online]. Available: https://www.nema.org/standards/view/motors-and-generators

[14] T. Hoffmann, "Sun position- and sun phases calculator." [Online]. Available: https://www.suncalc.org/#/37.9625,-87.669,17.231118073776123/2022.04.27/11:48/1/3

APPENDIX

APPENDIX A - Bill of Material

Bill of Material

Qty	Vendor	Cost	Description
1	Sabic	0.00	4'x8'x1/8" LEXAN sheet
4	Home Depot	65.58	4'x8'x3/4" plywood sheet for all framing, ribs, and mounting brackets
1	Amazon	42.99	4'x50' mylar reflective film
1	ePlastics	37.18	IPS 40 PINT
1	McMaster-Carr	80.51	7/8" OD x 10' copper pipe (receiver)
1	Home Depot	4.98	12 oz can hi temp black spray paint for reciever tube (option 1)
2	Home Depot	4.67	6 in. Square Lazy-Susan Turntable with 400 lb. Load Rating
1	McMaster-Carr	13.67	8-32 thread inserts
1	McMaster-Carr	6.44	8'32 x 1" countersink screws
1	McMaster-Carr	4.11	8-32 washers
2	McMaster-Carr	17.21	1/8" thick aluminum 90 degree angle
1	Mouser	40.30	Development Boards & Kits - AVR Arduino Mega 2560 Rev 3
2	Mouser	14.95	Temperature Sensor Development Tools Thermocouple Amp MAX31855 Breakout
1	Mouser	4.95	Power Management IC Development Tools TB6612 1.2A Motor Driver Breakout
2	Mouser	6.60	PCBs & Breadboards 630 tie-point Solderless Plug-in BreadBoard
2	Mouser	0.94	Encoders 12mm Rotary Incremental Encoder
2	Mouser	9.95	Temperature Sensor Development Tools Thermocouple Type-K Glass Braid Insulated Stainless Steel Tip
20	Mouser	0.95	Photoelectric Sensors Photo Cell CdS Photoresistor
1	Mouser	5.00	PCBs & Breadboards SHIELD - MEGA PROTO PCB REV3
2	Mouser	10.95	Display Development Tools Assembled Standard LCD 16x2 + extras - White on Blue
1	Mouser	2.58	Piezo Buzzers & Audio Indicators XDUCR, PIEZO, 2.9KHZ 95DB, FLANGE, 42X16
8	Mouser	0.36	Tactile Switches Thru hole 6mmx6mm SPST-NO 0.05A 50V
3	Mouser	1.84	Control Switches 1K2 BLK ACT GOLD STRT MNT
1	Mouser	5.24	Encoder with push button
1	Mouser	14.00	Adafruit Accessories Stepper Motor NEMA17 12V 350mA
25	Mouser	0.74	Flat Cables SEK CAB FLAT STD SOLD PER FOOT
1	Mouser	14.35	Switching Power Supplies AC-DC 75W LOW COST
2	McMaster-Carr	31.54	1 1/4" shaft 12" long
2	McMaster-Carr	61.83	1 1/4" pillowblock bearings
2	WestMetalSales	36.00	2"x2"x12" solid steel bar cold finished
1	Home Depot	44.98	1 gal white exterior paint
1	Home Depot	19.98	3 pack of assorted paint brushes
1	Home Depot	5.13	pack of 3 buckets
1	Home Depot	3.17	plastic drop cloth
1	Amazon	69.00	Worm Gear Gearbox NMRV-030 Speed Reducer Ratio 80:1 for Nema23 Stepper Motor
1	Amazon	28.99	CNC Stepper Motor Driver 1.0-4.2A 20-50VDC 1/128 Micro-Step Resolutions for Nema 17 and 23 Stepper Motor
1	Amazon	37.99	Bipolar Nema 23 Stepper Motor 1.8 Deg 3Nm Digital Stepping Motor
1	Amazon	7.99	Shaft Sleeve 8mm Shaft Motor to 11mm Bore Adapter for RV030 Worm Gear Speed Reduce
1	Amazon	37.61	LRS-350-48 350.4W 48V 7.3 Amp Single Output Switchable Power
1	Amazon	19.99	Waterproof Momentary Rotary Roller Lever Limit Switch Pack of 5
1	Amazon	15.99	50 Pack Cable Gland Waterproof Adjustable 3-16mm Cable Connectors
1	Amazon	10.99	10Pcs 12mm Momentary Push Button Switch
1	Amazon	9.98	Waterproof Plastic Project Box ABS IP65 Electrical Junction box
1	Amazon	6.13	3-Wire Appliance and Power Tool Cord, 6 ft, 14 AWG
1	Amazon	13.89	Emergency Push Button Swicth Latching Stianless Steel
1	Amazon	67.99	11.81" x 15.7" x 6.7" Waterproof IP67 Hinged Electronics Box with Clear PC plastic cover
1	Amazon	11.99	USB B Male to Female Flush Panel Mount Extension Cord
1	Amazon	15.97	18/4 - Brown - Solid Copper 18 Gauge, 4 Conductor
1	Amazon	25.72	Female to Female Telco Cable Assembly in 25 Pair, 5 FT
1	Amazon	17.99	22 gauge 5 conductor wire Audio Power Cable Speaker Wire (Red & Black & yellow & white & green) 25ft
1	Amazon	7.82	4pcs Relay Module DC 5V 1 Channel Relay Board with Optocoupler Isolation Support High or Low Level Trigger
1	Amazon	24.99	Electrical Junction Box with Mounting Plate and IP67 Waterproof Clear Cover (8" x 6" x 5")
	Total	**1478.08**	

APPENDIX B - Design Factor Locations

Design Factor	Page number, or reason not applicable
Public health safety, and Welfare	67
Global	N/A, Project created for USI students in America only
Cultural	48
Social	17
Environmental	1,17,52
Economic	ii,17
Professional Standards	17, 45,52

APPENDIX C - Arduino Code

```
//code used for first system test on April 19th, 2022
//Nicholas Wester and Michael Tunny

#include <Stepper.h>
#include <LiquidCrystal.h>
#include <SPI.h>
#include "Adafruit_MAX31855.h"

const int rs = 45, en = 46, d4 = 47, d5 = 48, d6 = 49, d7 = 50;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

//maximum angle
#define maxNegDeg -75
#define maxPosDeg 75

//encoder pins
#define CLK 36
#define DT 37
#define SW 38

//short thermocouple pins
#define MAXDO1   24
#define MAXCS1   23
#define MAXCLK1  22

//long thermocouple pins
#define MAXDO2   27
#define MAXCS2   26
#define MAXCLK2  25

//tracking buffers
#define coarseAdj 100
#define binAdj 30
#define centerAdj 10


// initialize the Thermocouples
Adafruit_MAX31855 thermocouple1(MAXCLK1, MAXCS1, MAXDO1);
Adafruit_MAX31855 thermocouple2(MAXCLK2, MAXCS2, MAXDO2);

//Light Dependent Resistor Pins
int ldrPin1 = A0;
int ldrPin2 = A1;
int ldrPin3 = A2;
int ldrPin4 = A3;
int ldrPin5 = A4;
int ldrPin6 = A5;
int ldrPin7 = A6;
int ldrPin8 = A7;

//LED pins
int redLED = 10;
int blueLED = 11;

//push button pins
int SW1 = 14;
int SW2 = 15;

//Stepper Driver pins
int driverPUL = 7;    // PUL- pin
int driverDIR = 6;    // DIR- pin
// Pulse Delay period from 50 to 2000, 2000 is slowest
int pd = 50;

//interrupt pins
//interrupt pins possible = 2, 3, 18, 19, 20, 21
```

```
int emergencyStopPin = 18;
int CWLimitPin = 2;
int CCWLimitPin = 3;

// various variables
int counter = 0;
int currentStateCLK;
int lastStateCLK;
int buttonFlag= 0;
String currentDir ="";

//tracking variables
int CWCoarse=0;
int CCWCoarse=0;
int CWBin=0;
int CCWBin=0;
int CenterBin=0;


//delay definitions
unsigned long lastButtonPress = 0;
unsigned long lastTCRead = 0;
unsigned long lastLDRRead = 0;









//SETUP (runs once when connected to power or reset)
void setup() {

  // Set encoder pins as inputs
  pinMode(CLK,INPUT_PULLUP);
  pinMode(DT,INPUT_PULLUP);
  pinMode(SW, INPUT_PULLUP);

  //manual mode buttons
  pinMode(SW1, INPUT_PULLUP);
  pinMode(SW2, INPUT_PULLUP);

  //LEDs
  pinMode(redLED, OUTPUT);
  pinMode(blueLED, OUTPUT);

  //Stepper Driver Board
  pinMode(driverPUL, OUTPUT);
  pinMode(driverDIR, OUTPUT);

  //setup LCD
  lcd.begin(16,2);

  // Setup Serial Monitor
  Serial.begin(9600);

  //initialize thermocouples
  while (!Serial) delay(1); // wait for Serial on Leonardo/Zero, etc

  Serial.println("MAX31855 test");
  // wait for MAX chip to stabilize
  delay(500);
  Serial.print("Initializing sensor...");
  if (!thermocouple1.begin()||!thermocouple2.begin()) {
    Serial.println("ERROR.");
    while (1) delay(10);
```

```
135   }
137   // Read the initial state of CLK (for encoder)
      lastStateCLK = digitalRead(CLK);
139
141   //initialize interrupts
      pinMode(emergencyStopPin, INPUT_PULLUP);
143   attachInterrupt(digitalPinToInterrupt(emergencyStopPin), emergencyStop, RISING);
145   pinMode(CWLimitPin, INPUT_PULLUP);
      attachInterrupt(digitalPinToInterrupt(CWLimitPin), positiveLimit, RISING); //comment if
        noise triggering
147
      pinMode(CCWLimitPin, INPUT_PULLUP);
149   attachInterrupt(digitalPinToInterrupt(CCWLimitPin), negativeLimit, RISING); //comment if
        noise triggering
151 }//end setup
153
155
157
159
    void loop() { //main loop
161
163   // set staus lights
      digitalWrite(redLED,LOW); //red LEDs off
165   digitalWrite(blueLED,HIGH); //blue LEDs
167   if (millis() - lastLDRRead > 500) { // 500 milisecond delay between readings
        int ldrRead1 = analogRead(ldrPin1);
169     Serial.print("pin1:");
        Serial.print(ldrRead1);
171     Serial.print(",");
        int ldrRead2 = analogRead(ldrPin2);
173     Serial.print("   pin2:");
        Serial.print(ldrRead2);
175     Serial.print(",");
        int ldrRead3 = analogRead(ldrPin3);
177     Serial.print("   pin3:");
        Serial.print(ldrRead3);
179     Serial.print(",");
        int ldrRead4 = analogRead(ldrPin4);
181     Serial.print("   pin4:");
        Serial.print(ldrRead4);
183     Serial.print(",");
        int ldrRead5 = analogRead(ldrPin5);
185     Serial.print("   pin5:");
        Serial.print(ldrRead5);
187     Serial.print(",");
        int ldrRead6 = analogRead(ldrPin6);
189     Serial.print("   pin6:");
        Serial.print(ldrRead6);
191     Serial.print(",");
        int ldrRead7 = analogRead(ldrPin7);
193     Serial.print("   pin7:");
        Serial.print(ldrRead7);
195     Serial.print(",");
        int ldrRead8 = analogRead(ldrPin8);
197     Serial.print("   pin 8:");
        Serial.println(ldrRead8);
199     lastLDRRead= millis();
```

```
201     // external coarse reads
        CWCoarse=ldrRead8;
203     CCWCoarse=ldrRead1;

205     //center coarse bin reads
        CWBin=(ldrRead7+ldrRead6)/2;
207     CCWBin=(ldrRead2+ldrRead3)/2;

209     //center bin read
        CenterBin=(ldrRead4+ldrRead5)/2;
211
    }
213

215 //If we detect LOW signal, button is pressed
    int btnState = digitalRead(SW);
217 if (btnState == LOW) {
      //if 50ms have passed since last LOW pulse, it means that the
219   //button has been pressed, released and pressed again
      if (millis() - lastButtonPress > 50) {
221     Serial.println("Button pressed!");
        buttonFlag=1;
223   }

225   // Remember last button press event
      lastButtonPress = millis();
227 }
    if(buttonFlag==1){
229   lcd.clear();
      manualMode();
231 }

233 //tracking logic
    //coarse adjustments
235 if(CWCoarse>CCWCoarse+coarseAdj){
      rotateCW();
237 }

239 if(CCWCoarse>CWCoarse+coarseAdj){
      rotateCCW();
241 }

243 //bin/center adjustments
    if(CWBin>CCWBin+binAdj && (CWBin>CenterBin-centerAdj||CWBin<CenterBin+centerAdj)){
245   rotateCW();
    }
247
    if(CCWBin>CWBin+binAdj && (CCWBin>CenterBin-centerAdj||CCWBin<CenterBin+centerAdj)){
249   rotateCCW();
    }
251



253


255 if (millis() - lastTCRead > 1500) {
      double F1 = thermocouple1.readFahrenheit();
257   if (isnan(F1)) {
        Serial.println("Something wrong with thermocouple 1 !");
259     //lcd.print("TC 1 (F1)= ERROR");
        lcd.setCursor(0,0);
261   } else {
        lcd.print("TC 1 (F)= ");
263     lcd.print(F1);
        lcd.setCursor(0,1);
265   }

267   double F2 = thermocouple2.readFahrenheit();
      if (isnan(F2)) {
```

```
269       Serial.println("Something wrong with thermocouple 2 !");
          lcd.print("TC 2 (F)= ERROR");
271       lcd.setCursor(0,0);
        } else {
273       lcd.print("TC 2 (F)= ");
          lcd.print(F2);
275       lcd.setCursor(0,0);
        }
277
        lastTCRead= millis();
279
    }
281 }//end main loop
283
285
287
289
291 void manualMode() {
293
    while(buttonFlag){
295
        int btnState = digitalRead(SW);
297
        if (btnState == LOW) {
299       //if 150ms have passed since last LOW pulse, it means that the
          //button has been pressed, released and pressed again
301       if (millis() - lastButtonPress > 150) {
            Serial.println("Button pressed!");
303         buttonFlag=0;
          }
305
          // Remember last button press event
307       lastButtonPress = millis();
        }
309
        // Read the current state of CLK
311     currentStateCLK = digitalRead(CLK);
313     // If last and current state of CLK are different, then pulse occurred
        // React to only 1 state change to avoid double count
315     if (currentStateCLK != lastStateCLK  && currentStateCLK == 1){
317       // If the DT state is different than the CLK state then
          // the encoder is rotating CCW so decrement
319       if (digitalRead(DT) != currentStateCLK) {
            if (counter>maxNegDeg){
321           counter =counter-5;
            }
323         currentDir ="CCW";
          } else {
325         // Encoder is rotating CW so increment
            if (counter<maxPosDeg){
327           counter =counter+5;
            }
329         currentDir ="CW";
          }
331
          Serial.print("Direction: ");
333       Serial.print(currentDir);
          Serial.print(" | Counter: ");
335       Serial.println(counter);
        }
```

```
337
      // Remember last CLK state
339     lastStateCLK = currentStateCLK;

341     // Put in a slight delay to help debounce the reading
        delay(1);
343     lcd.print("Manual Mode");
        lcd.setCursor(0,1);
345     lcd.print("Degrees: ");
        lcd.print(int(counter));
347
        lcd.setCursor(0,0);
349
        //using buttons
351
        int SW1State = digitalRead(SW1);
353     int SW2State = digitalRead(SW2);

355     if(SW1State==LOW){
          Serial.print("CCW");
357       rotateCCW();

359     }

361     if(SW2State==LOW){
          Serial.print("CW");
363       rotateCW();

365     }

367   }//end while

369   moveSetDegrees();

371 } //end manualMode

373



375
    void rotateCCW() {
377   Serial.println("Rotate CCW");
      digitalWrite(driverDIR,LOW);
379   digitalWrite(driverPUL,HIGH);
      delayMicroseconds(pd);
381   digitalWrite(driverPUL,LOW);
      delayMicroseconds(pd);
383 }

385


387 void rotateCW() {
      Serial.println("Rotate CW");
389   digitalWrite(driverDIR,HIGH);
      digitalWrite(driverPUL,HIGH);
391   delayMicroseconds(pd);
      digitalWrite(driverPUL,LOW);
393   delayMicroseconds(pd);
    }
395


397



399


401 void moveSetDegrees() {
      //code here to move mirror
403   counter=0;
      buttonFlag=0;
```

```
405   lcd.clear();
}
407
409
411
413
//Interrupt Function Definitions
415 void emergencyStop() {
  noInterrupts();
417   digitalWrite(redLED,HIGH);
  digitalWrite(blueLED,LOW);
419   while(1){
    lcd.clear();
421     lcd.print("EMERGENCY STOP");
    lcd.setCursor(0,1);
423     lcd.print("Reset Switch");
    delay(500000);
425     lcd.clear();
    lcd.print("Then Press");
427     lcd.setCursor(0,1);
    lcd.print("Reset Button");
429     Serial.println("Emergency Stop");
    delay(500000);
431   }
}
433
435
437
439
void positiveLimit() {
441   digitalWrite(redLED,HIGH);
  digitalWrite(blueLED,LOW);
443   Serial.println("Positive Limit Hit");
  lcd.clear();
445   lcd.print("CW Limit Hit");
  lcd.setCursor(0,1);
447   lcd.print("Use Reset");
  while(1){
449   }
}
451
453
455
void negativeLimit() {
457   digitalWrite(redLED,HIGH);
  digitalWrite(blueLED,LOW);
459   Serial.println("Negative Limit Hit");
  lcd.clear();
461   lcd.print("CCW Limit Hit");
  lcd.setCursor(0,1);
463   lcd.print("Use Reset");
  while(1){
465   }
}
```

APPENDIX D - Resources

- Contact Info for Michael Tunny: mtunny@eagles.usi.edu
- Contact Info for Nicholas Wester: nlwester@eagles.usi.edu
- Contact Info for Dr. Jenna Kloosterman: jkloosterm@usi.edu
- Link to SharePoint:
  https://pilotusi.sharepoint.com/sites/GRP_Senior_Project/SitePages/ProjectHome.aspx