Multi-Disciplinary Senior Design Conference
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York 14623

Project Number: P11543

VARIABLE LED HEMISPHERICAL IMAGER

Alexander Usachev - CE
Nicholas Liotta - EE

Sigitas Rimkus - ME
Michael Miranda - EE
Azamat Boranbayev - ME

ABSTRACT

The objective of this project was to design an illumination cavity consisting of a 1-meter diameter hemisphere and several LED clusters with repeatable positioning. Each cluster has seven individually addressable LEDs, with six of them having an analog control while the seventh is digital. The clusters are controlled using a numerical computing environment known as MATLAB, which can access any and all clusters simultaneously. Dr. Dave Wyble and Dr. Jinwei Gu of the Color Science department of Rochester Institute of Technology provided the specific requirements addressed in this project.

INTRODUCTION (OR BACKGROUND)

The many forms of LEDs (light-emitting diodes) are the emerging technology for modern illumination. This project takes an advantageous step forward by providing an adjustable level of illumination, which has been missing in previous designs. Fundamental evaluation and quantification of image appearance as a function of the illumination spectrum, intensity, and incident angles require a tunable light source. A camera positioned at the top of the hemisphere will take images of the sample area at which the clusters are pointing, and will be used for research conducted by the Color Science department of RIT.

DESIGN PROCESS

The design of this project was separated into three sections, the LED cluster, the LED control, and the hemisphere. As preliminary prototypes were built, different sections of the projects began to mend together to provide a clearer view of the final objective.

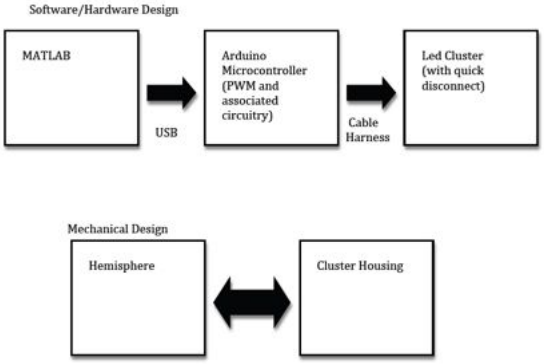
Software/Hardware Design
MATLAB
USB
Arduino Microcontroller (PWM and associated circuitry)
Cable Harness
Led Cluster (with quick disconnect)
Mechanical Design
Hemisphere
Cluster Housing

Figure 1. Block Diagram of Entire System

LED Cluster

The design of the cluster began with a simple prototype of how the LEDs would be positioned. Seven LEDs were placed on a small board and connected to a 4-AA battery pack.

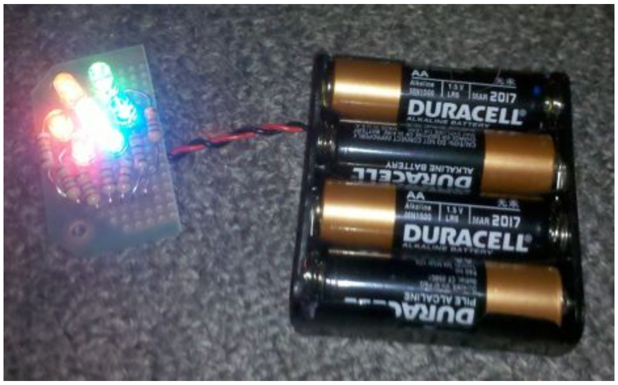

Figure 2. LED Cluster Prototype 1

While the prototype was a good first step, the light coming from the cluster was not concentric and needed revising. In the process of developing a second prototype, the physical characteristics of the cluster were being designed. A cluster housing would be machined that would hold the LEDs in place. The second cluster prototype contained a reflector in the back, which helped mix the light as it was projected onto the sample area. This prototype was also connected to a variable control, which will be discussed later.

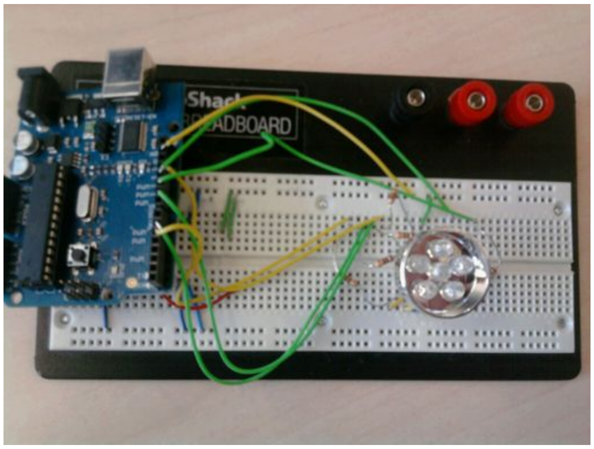

Figure 3. LED Cluster Prototype 2

The cluster housing, which comprised the majority of the cluster, is made out of aluminum. It contains one hole on the bottom for positioning onto the dome vertices, and one hole on the side for the cable harness input. The cluster is attached to the dome using two magnets on opposite sides of the positioning hole.

The LEDs are connected to the cluster using 14 press-fit pins, two for each LED, which are soldered to an originally designed PCB (printed circuit board). The PCB is connected to the cluster housing using three 4-40 bolts. A machined reflector was added to the design after a similar reflector taken from a flashlight helped center the light coming from the LEDs. The LEDs wavelength can be seen below. The reflector is attached to the cluster using three more 4-40 bolts. The entire cluster, comprised of the cluster housing, PCB, reflector and LEDs, is connected to the control using a 12-foot cable harness. The harness connects to both the cluster and the control using several plastic socket connectors.

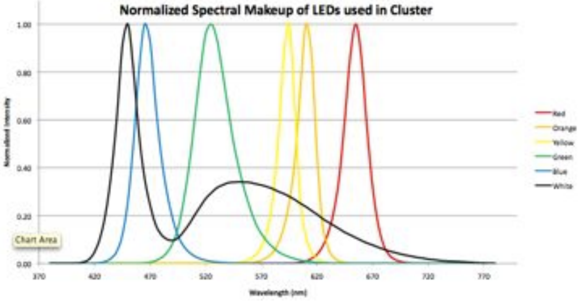

Figure 4 Individual LED Wavelengths

Several issues were encountered while manufacturing the cluster. The LED pins which sit in the PCB connectors were not large enough to be tightly secured within the PCB, and it was required that they be soldered in place. This created additional possibilities of short circuits within the cluster. This problem revealed itself after several short circuits were discovered in a cluster as a side-effect of grinding down the solder to allow for a better fit of the reflector. The PCB in question was repaired and a layer of electrical tape was also added between the cluster housing and the reflector to act as an insulator and further prevent short circuits.

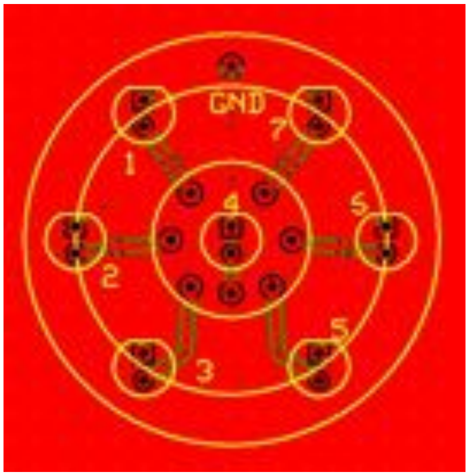

Figure 5. Original Cluster PCB Design (with Silk Screen)

After a couple of revisions, the PCB the shorting problem was resolved. By removing the excess tin and creating larger leads for soldering on the connectors, solder would no longer overflow to the entire plane. This made it easier to manufacture and solved the shorting problem encountered on previous designs.

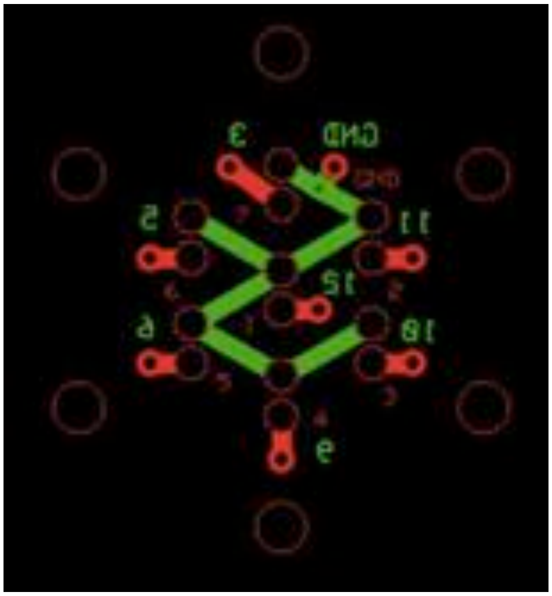

Figure 6 Final Cluster PCB Layout

The entire cluster is shown in the Figure below.

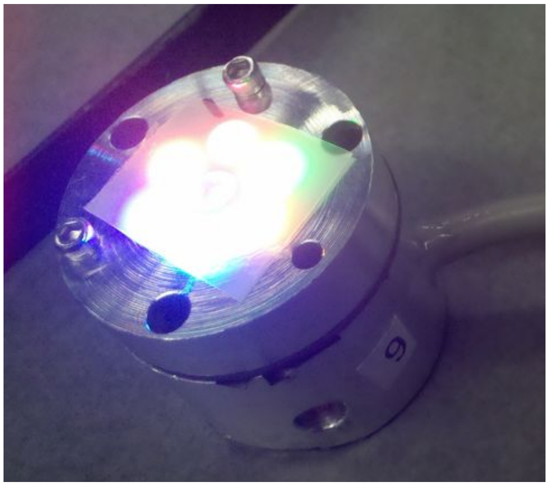

Figure 7 Full Cluster (on) with Diffusor

LED Control

Since the main priority of the control was to be adaptable to MATLAB, the choices were limited. The second priority was that the design was easily scalable. After researching multiple microcontrollers, it appeared only one was able to interface with MATLAB in the manner we desired, the Arduino Duemilanove.

Six PWM (pulse-width-modulation) outputs on the Arduino are used to control the individually addressable LEDs. Each output has 8-bit control over one LED, and has the ability to vary its intensity between a binary count of 0-255. However, since the cluster has seven LEDs, the middle LED, which is almost always chosen to be white, has to be controlled digitally. Due to the fact that there are only 6 PWM outputs per microcontroller, each cluster has its own Arduino associated with it. This makes for very easy scaling, one cluster per controller.

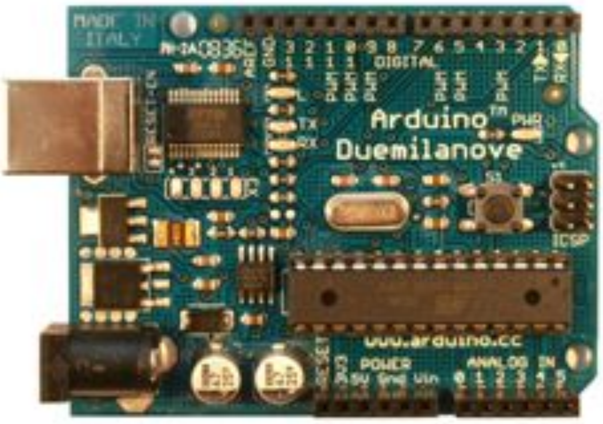

Figure 8 Arduino Duemilanove Microcontroller

In order for a computer to be able to transmit and receive information from an Arduino, drivers must be installed after it is connected via USB. The process for this step is different between computers with the Windows and Mac OSX operating systems. On Windows 7, the drivers are automatically installed once the Arduino is connected, and this only needs to be done for the first Arduino, as the drivers installed are compatible with any identical Arduinos that are connected afterword. The Mac OSX operating system needs FTDI drivers to be manually installed on each Arduino that is going to be controlled by the computer. Once the drivers are installed, the Arduino software is able to communicate with the microcontroller, and is used to upload the MATLAB package onto the Arduino. Once the Arduino contains the MATLAB package, it is able to be controlled through MATLAB as long as the file 'arduino.m' is in the working directory of the program.

MATLAB uses the functions *arduino('portname')* and *pinMode('pin','OUTPUT')* to associate the Arduino with a given port and let it know which pins will be used for either an input or output signal. The functions *analogWrite('pin','value')* and *digitalWrite('pin','value')* are used to control the analog and digital pins, respectively. The analog values are 0-255 while the digital ones are 0 and 1. The *delete(a)* function is used to close the connection to the Arduino, where 'a' is the variable used to instantiate it.

The generic functions described above were used to create the original functions *DomeInit()*, *DomeSet(),* and *DomeClose()* to help the customer control the clusters more effectively. The *DomeInit()* and *DomeClose()* functions require an array parameter that allows MATLAB to connect to or disconnect from multiple clusters with a single command. The *DomeSet()* command requires two parameters: the cluster number, and an array of the intensities desired. The code snippet below shows how clusters numbered 1 and 3 are instantiated, and disconnected.

```
>> DomeInit([1,3]);
>> DomeSet(3, [1, 255, 255, 255, 255, 255, 255]
)
>> DomeClose([1,3,4]);
```

The first value of the intensities array is used to control the digital pin, and only alternates between 0 and 1. The order of the pins associated with *DomeSet()* reflects the order in which they are connected to the PCB, with the digital pin being first followed by the analog pins in what the cluster shows as a clockwise motion. The code is almost identical for Windows and MacOSX operating systems except for the initial connection performed by the *arduino('portname')* function. In Windows, the port name is shown as COM1, COM2, COM3, etc. whereas MacOSX shows them as A400fxG5, A8007mqe, etc. In Windows, the number of the cluster does not necessarily reflect the number of the port.

In order to connect multiple Arduinos to a single USB port, a 7-port USB hub was used, allowing the user to control up to 7 clusters simultaneously. Each pin that was used by the Arduino is connected to a 100 ohm resistor prior to reaching the desired LED. This resistor limits the current flowing to each LED, as they are only rated for 25mA and the Arduino can provide a maximum of 50mA per output. This is shown in the schematic below.

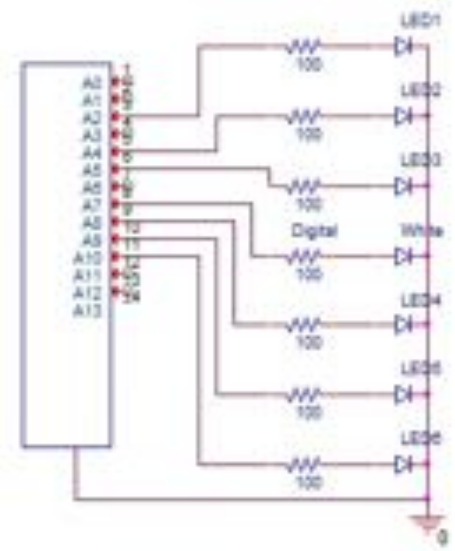

Figure 9 Electrical Connection From Arduino to LED

Geodesic Dome

The dome was constructed using a 4V Icosahedral geodesic pattern. The original design called for either a 2V/L2 Icosahedron Dome (shown below) or the 3V Octahedral Dome (Figure 10).

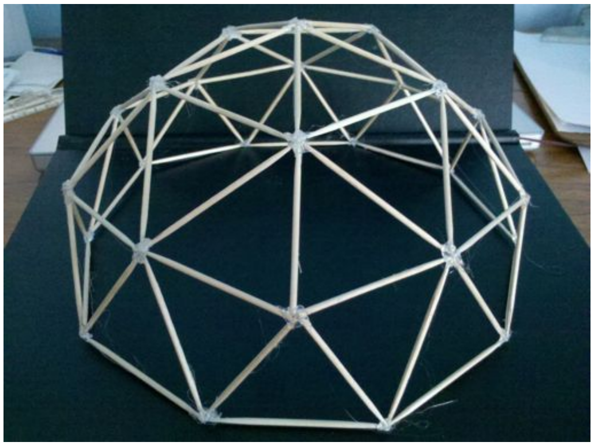

Figure 10 2V/L2 Icosahedral Dome Prototype

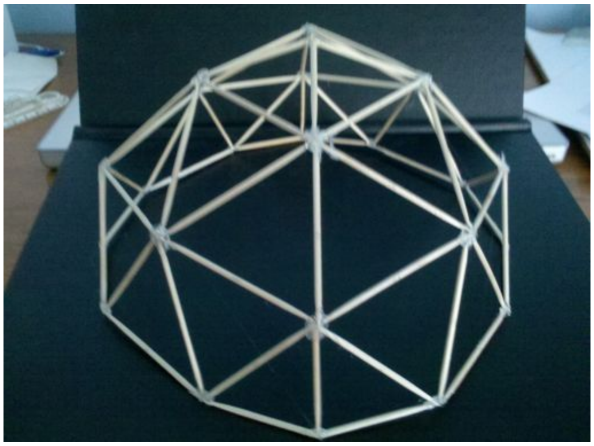

Figure 11 3V Octahedral Dome Prototype.

However, these designs did not have enough vertices on which to place the clusters, so a denser dome design was chosen. The completed dome is shown in Figure 9.

The dome was made of 1/2” wide by 1/16” thick steel flat stock and was bolted together at each vertex. Design of the vertices is such that the normal from the plane of the vertex is oriented towards the sample area. The cluster attaches to the dome using two magnets. The ideal cluster locations are the vertices themselves, as they point directly toward the sample area, but any point on the dome can be used as a mounting point, although not as accurately. Basic machining techniques were used to fabricate the dome and the clusters. After the dome was completed it was painted matte black to reduce any undesired optical aberrations.

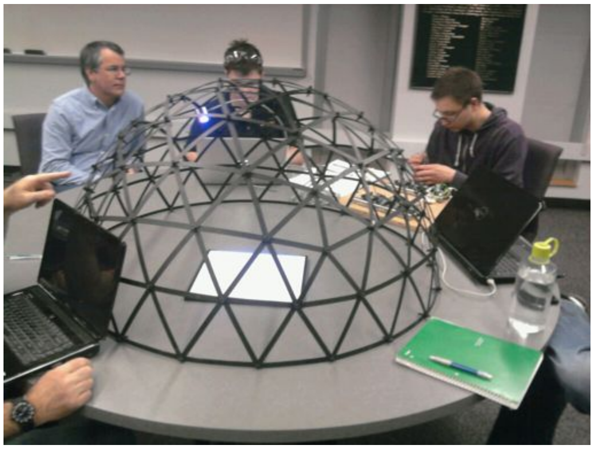

Figure 12 Completed 4V Icosahedron Dome

EXPERIMENTAL PROCEDURE/DATA ANALYSIS

Several techniques were used to verify the functionality of the dome, including the testing of LEDs and reflector designs.

LED Testing

Three LEDs of each color were evaluated for their exact wavelength and luminous intensity using a radial spectrograph in a dark room located in the Color Science building. The results of each LED were compared to the results for other LEDs of the same color for consistency. It turned out that the LEDs were almost identical in their luminous intensity and wavelength, per color. The combined results of all LEDs showed that the entire visible spectrum is adequately covered by the LEDs located in the cluster (red, orange, yellow, green/aqua, blue, violet, white). There is a small gap in the spectrum of the green LED, which was expected. The plot below shows the scaled spectral data of all tested LEDs.

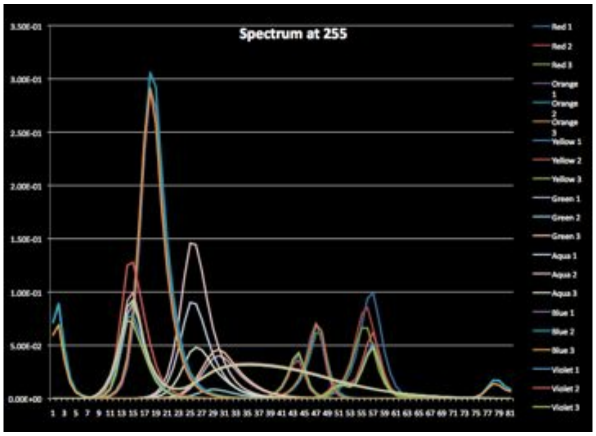

Figure 13 LED Spectral Data

Reflector Designs

There were 6 different reflector designs were tested by photographing a sample area illuminated by the cluster. (See Figure 13 for different reflector designs. The image was interpreted by MATLAB to see which reflector provided the highest light intensity and the highest qualitative light overlap on the sample area. There were several original designs of the reflector, including a simple 6-hole pattern and a honeycomb design. The final design used was the simplest, but different thicknesses were also tested for the most desirable results. After looking at several thicknesses, it appeared clear that although a thicker reflector condensed the light into a smaller area, it did not increase the light intensity. Therefore, the size used was the smallest that would cover the LEDs.

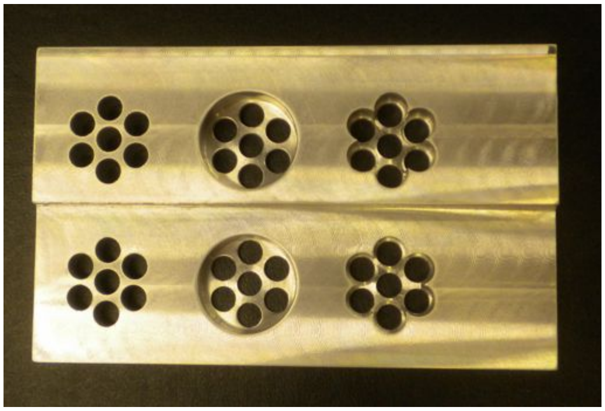

Figure 14 Different Reflector Designs

The final reflector was chosen because it focused the light the best with out focusing it too much. The final reflector was .375 inches high (seen below).

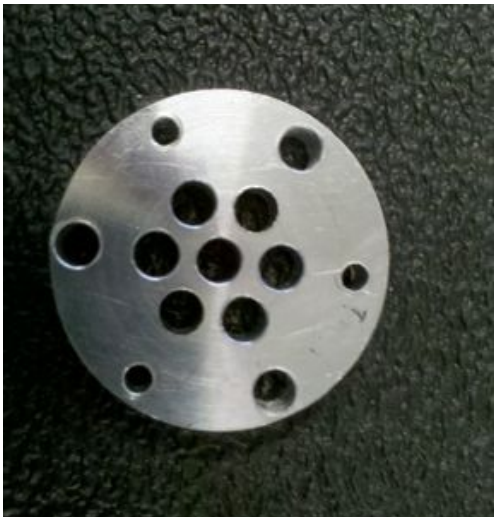

Figure 15 Final Reflector Design

Another problem that was encountered with the reflector was that it would not sit flat on the cluster housing because of the step on the led, connectors and soldered wire ends. This problem was solved by counter boring a .075 inch deep and 1 inch diameter recess. This allowed the reflector to sit flush onto the cluster housing. The most importantly it focused the LEDs onto the sample area. Before the recess the LEDs would not always sit the same way.

CONCLUSIONS AND RECOMMENDATIONS

The conclusions that can be drawn thus far do not reflect the ultimate intended purpose for this project. One thing that has been shown is that the LED cluster is capable of reproducing a color to a reasonable degree of qualitative accuracy if given the correct input parameters for the intensities of the LEDs. This was confirmed during a technical demo where the RIT orange color was acquired via an imaging instrument and recreated using the LED clusters. This also shows that MATLAB can be successfully used to control the LED clusters such that a desired light spectrum output is achieved.

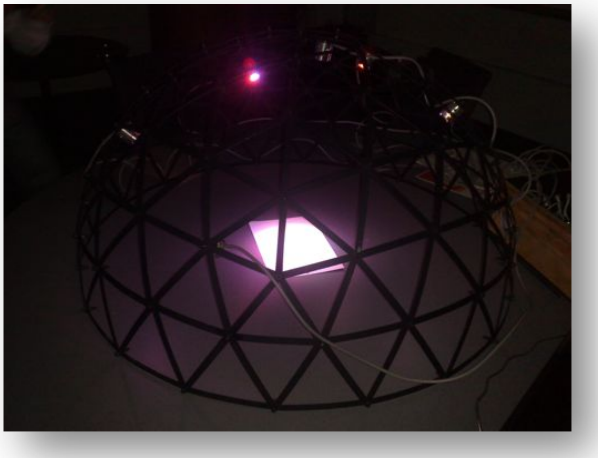

Figure 16 Illuminated Dome

There are several recommendations for future work that can improve the usability, maintainability, and accuracy of the design. High-power LEDs can be introduced to increase the luminous intensity of the cluster. However, a change like this would require additional hardware such as an amplifier because the Arduino only outputs 50 mA. High-power LEDs are usually made around 75 mA.

A more robust and user-friendly cluster design, specifically the PCB, would increase the maintainability of the cluster and reduce risk associated with the electrical connections. A good step forward would be to have a silk screen on the PCB and tightly fitting LED pins to minimize the amount of solder.

The dome itself can also use a denser network of mounting points, as there can never be too many locations from which to illuminate the sample area. The more mounting points there are, the more accurate results can be extracted from the research. .

REFERENCES

[1] Campa, Giampiero. Learning Basic Mechatronics Concepts Using the Arduino Board and MATLAB. El Segundo, CA: The MathWorks, Inc., 2009. PDF.
[2] Mueller, Rene K. "Geodesic Dome Notes." *SimplyDifferently.org*. 15 Jan. 2009. Web. Dec. 2010. <http://simplydifferently.org/Geodesic_Dome_Notes?page=3>.
[3] "Arduino - MacOSX." *Arduino - HomePage*. Arduino. Web. Dec. 2010. <http://www.arduino.cc/en/Guide/MacOSX>.
[4] "Arduino Tutorial - Learn Electronics and Microcontrollers Using Arduino!" *Ladyada*. Web. Dec. 2010. <http://www.ladyada.net/learn/arduino/index.html>.

ACKNOWLEDGMENTS

David R. Wyble, Associate Scientist, RIT Munsell Color Science Laboratory
Jinwei Gu, Assistant Professor, RIT Munsell Color Science Laboratory
Ed Hanzlik, Adjunct Instructor, RIT Department of Mechanical Engineering
Val Hemink, Administrative Assistant, RIT Munsell Color Science Laboratory
Staff of The RIT Brinkman Machine Tools and Manufacturing Laboratory
Staff of The RIT Machine Tool Laboratory
Staff of The RIT Department of Mechanical Engineering Machine Shop