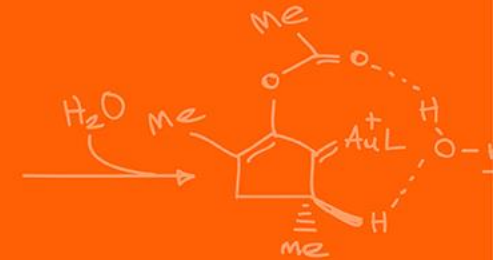
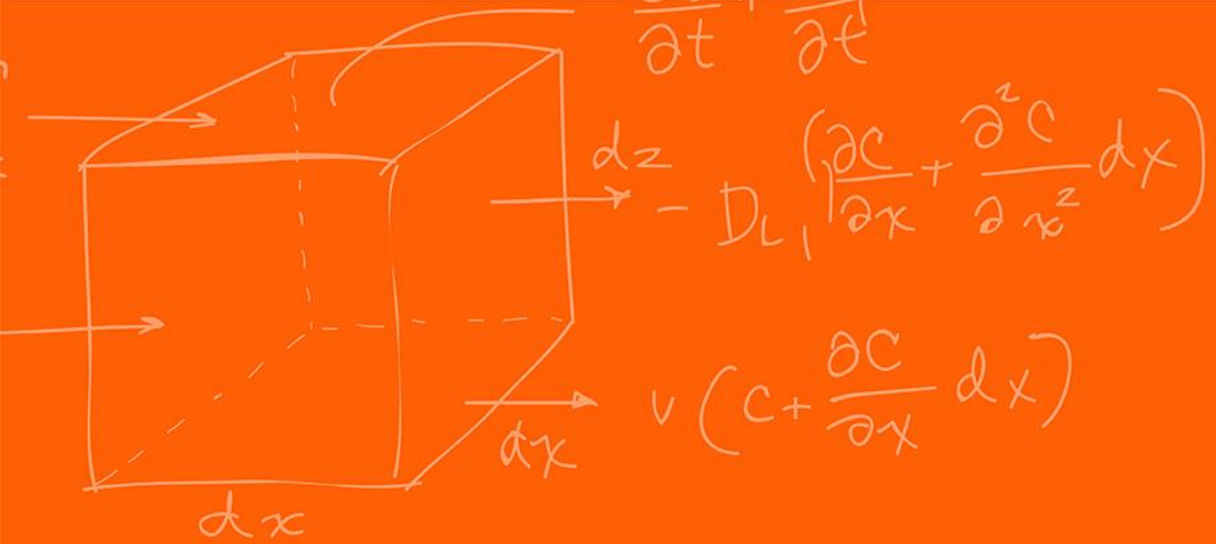


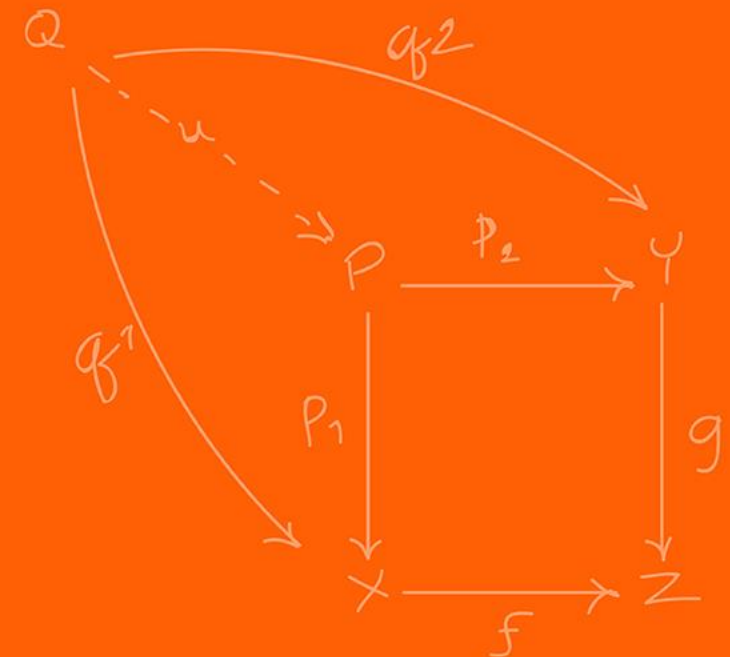


UNIVERSITY OF
ILLINOIS
URBANA-CHAMPAIGN

Persistence of Vision Globe – Team #17



Introduction



Problem

Professor Kwiat of lab escape introduced his latest endeavor to include a persistence of vision globe to the numerous science related puzzles in the escape room. However, the device they acquired did not perform as intended

Issues it had:

- Unregulated rotational speed
- Single color
- Little documentation to go along with it
- No easy way to program

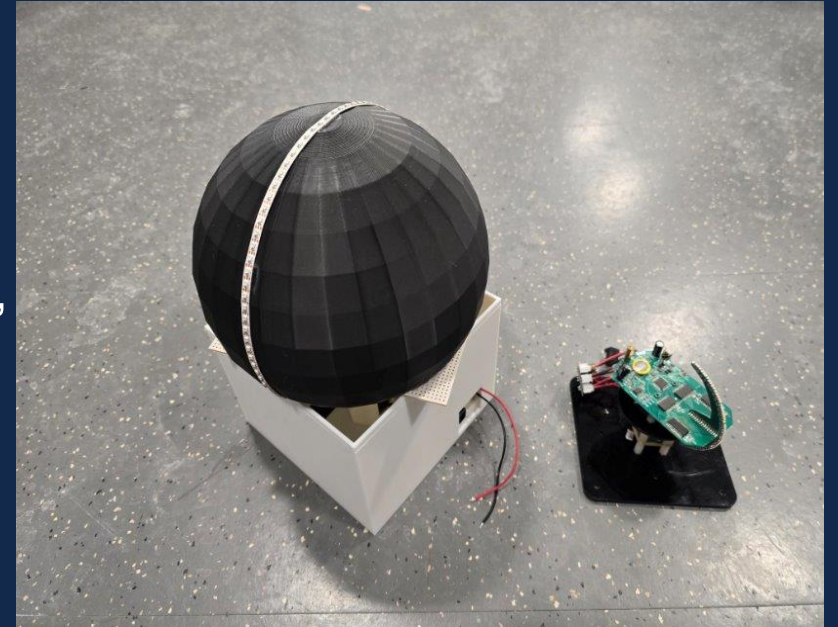


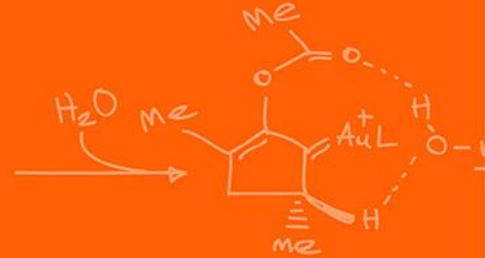
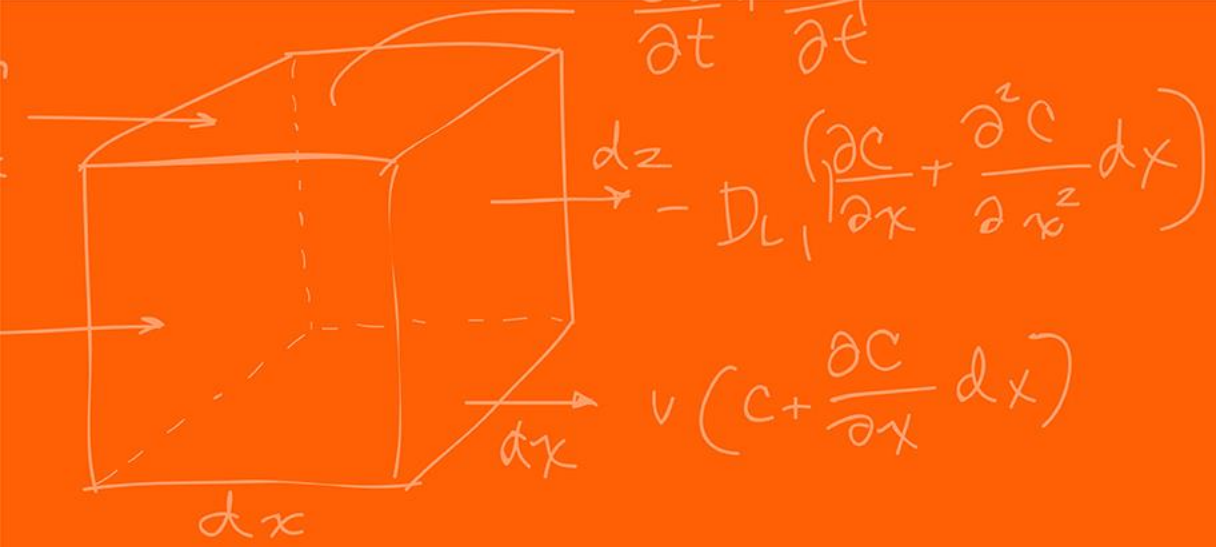
Solution

To address the limitations of the existing POV globe, we designed and built a fully programmable Persistence of Vision system with improved color capability, stable rotation control, and a user-friendly interface.

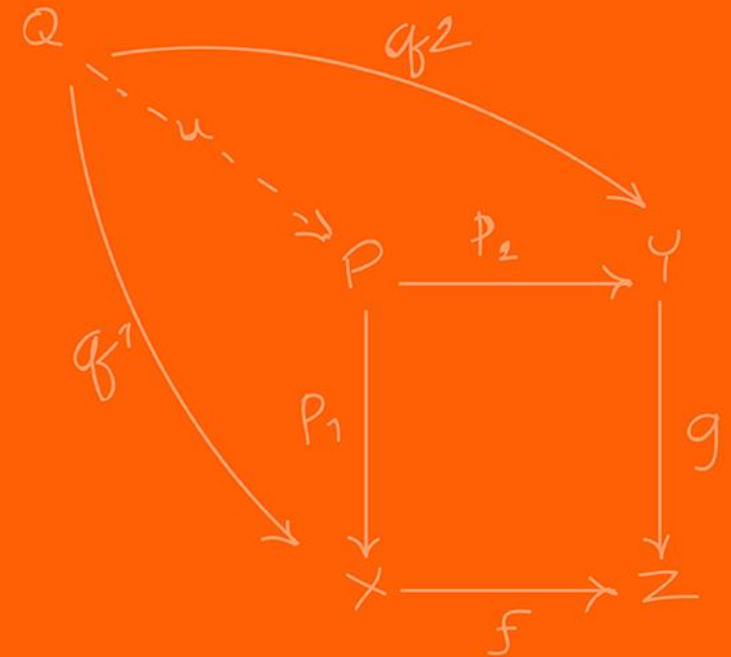
Our solution uses an RGB LED strip to support full-color images, and an ESP32 microcontroller to provide integrated Wi-Fi and handle real-time LED timing, rotation sensing, and image rendering.

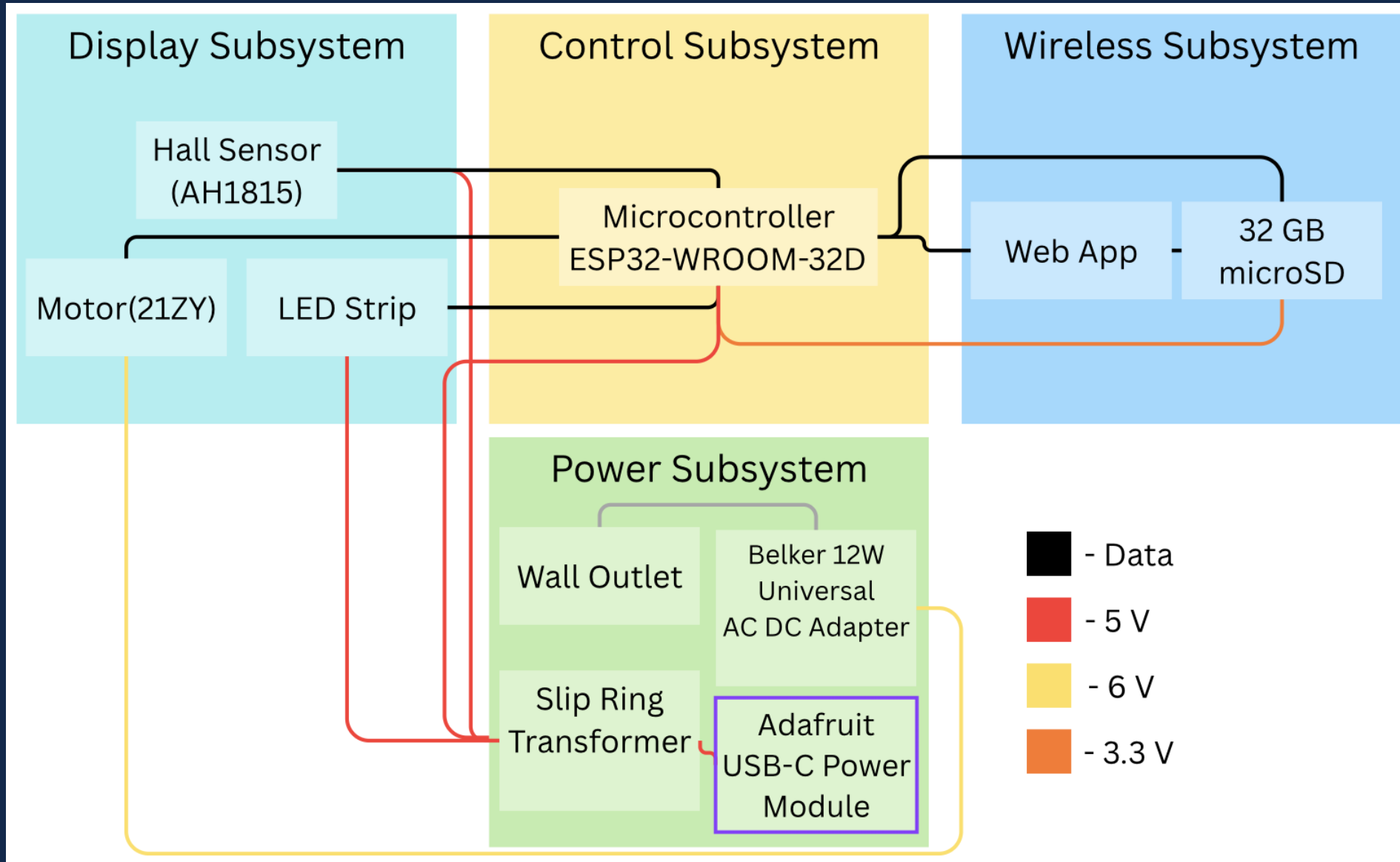
We developed a custom web application that allows users to upload compatible image files, manage stored graphics, and select which images the globe displays.





Design

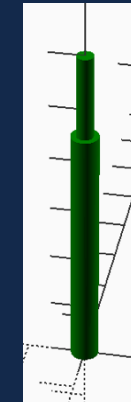
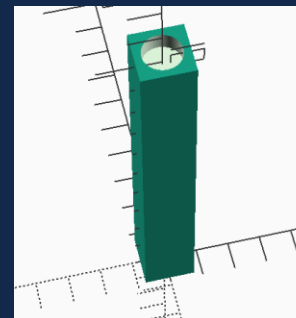
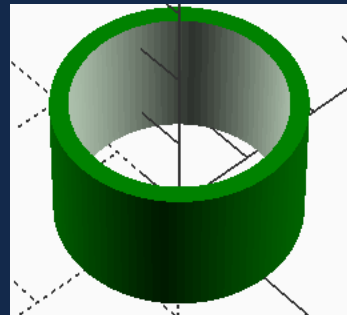
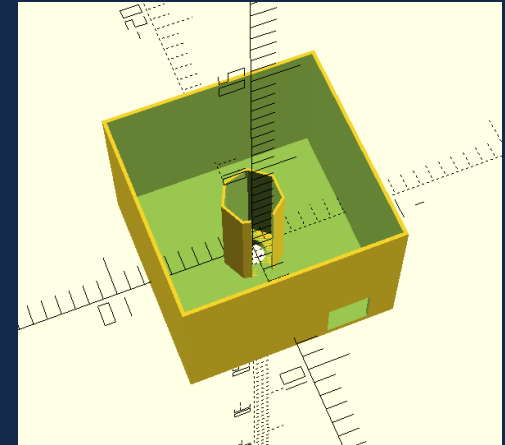
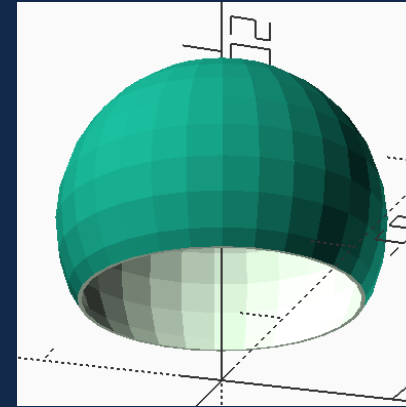


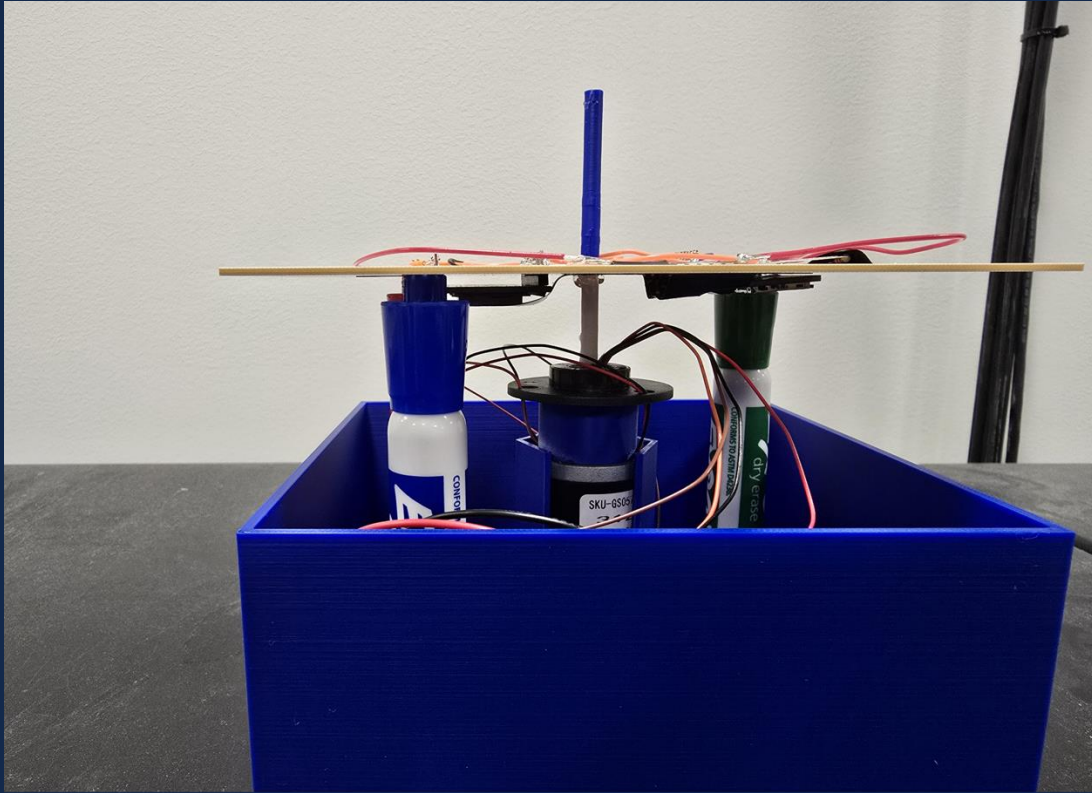


Components and Construction: How did we build it?

Main Components:

- Component Connections Board
- LED Globe Structure
- Project Mount
- Motor Assembly
- Power Supply Unit





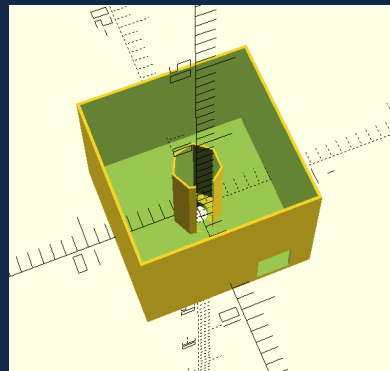
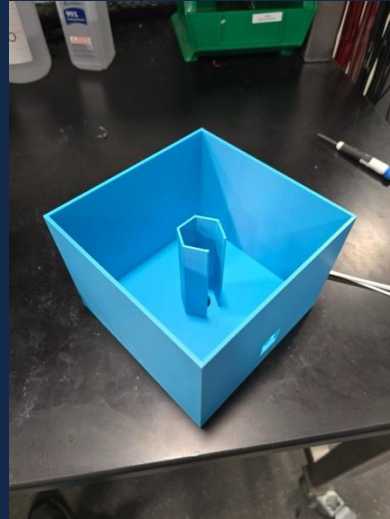
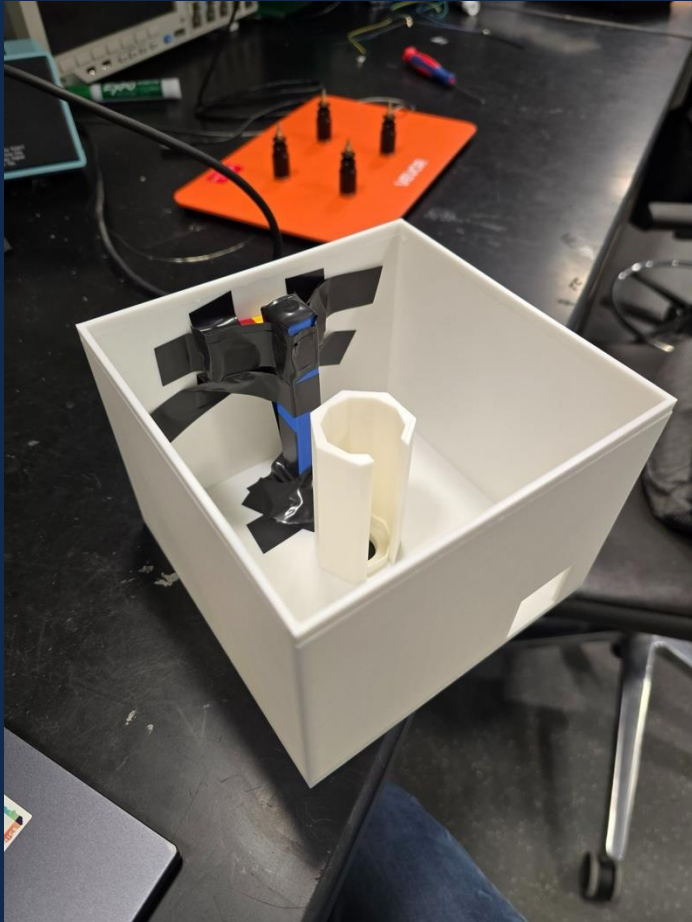
1. Component Connections Board

On this board we connected the wiring for the SD Card, ESP32, and Hall Sensor. Additionally, it serves the the connector to the Motor assembly and base for the LED Globe.

2. LED Globe Structure

This structure was 3D printed with a radius of 9.25cm to accommodate 62 LEDs. It is printed hollow with a minimal supporting wall to contribute to lightweightness.



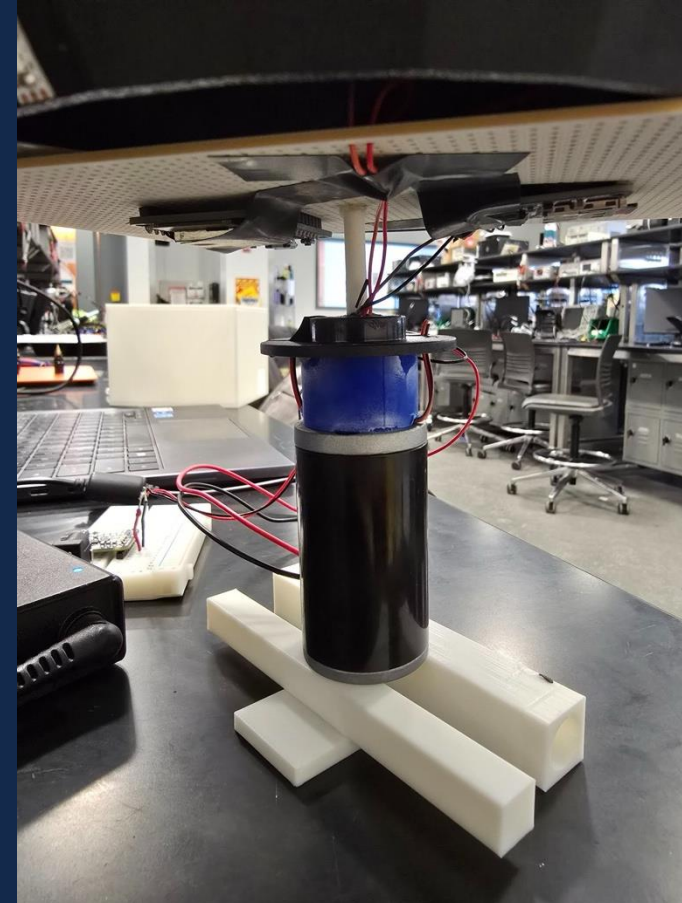
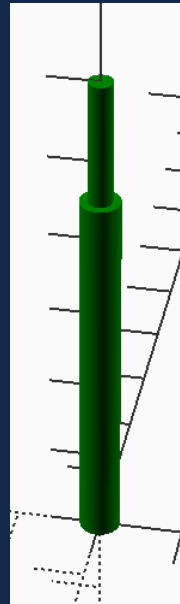
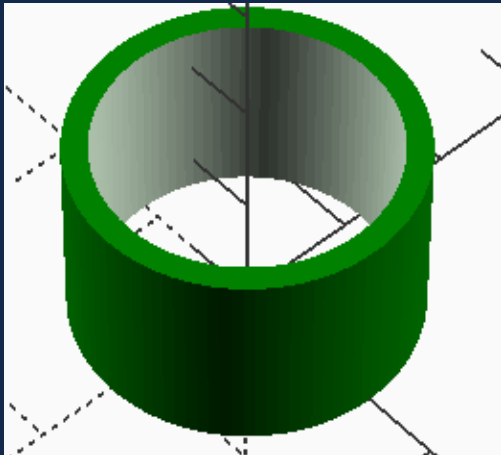


3. Project Mount

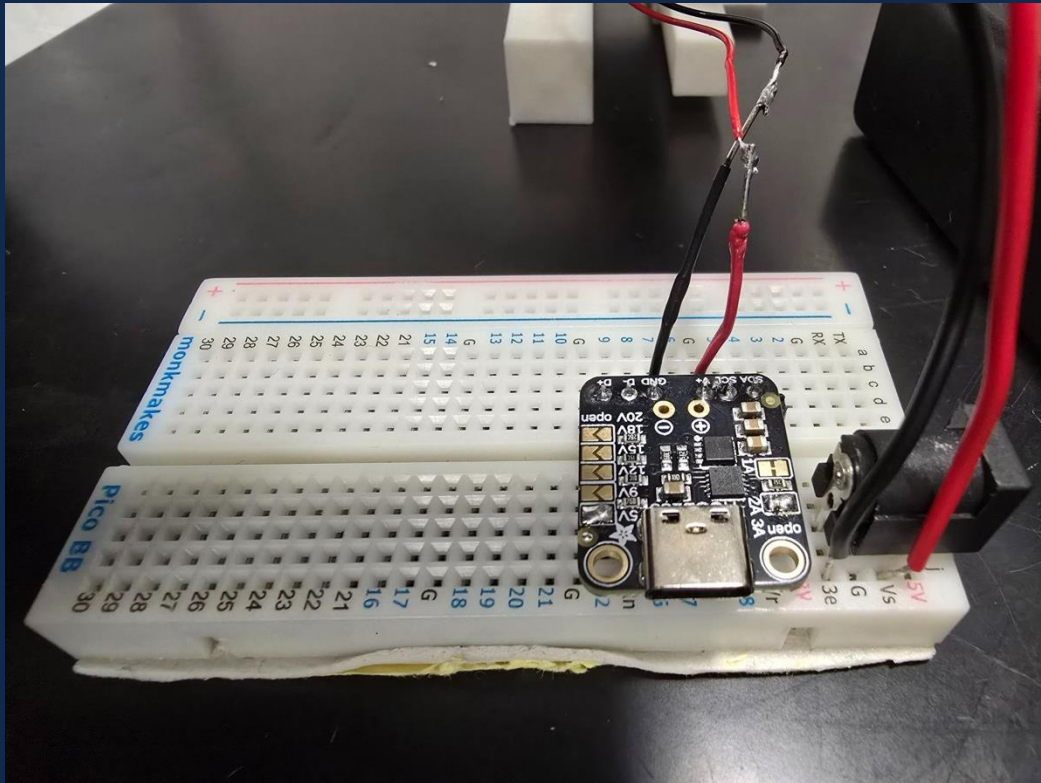
This Structure was 3D printed to provide support for our motor assembly while also encapsulating the Power Unit. Its final dimensions were 15x15x11.4cm. This also served to mount the magnet which is used to synchronize the RGB display.

4. Motor Assembly

The Motor Assembly Consists of the Motor, Slip Ring, and other 3D printed Support/Alignment Structures.



5. Power Supply Unit



This Unit consisted of a port for wall power dedicated to running our motor at 3, 4.5, 5, 6, 7.5, 9, or 12 volts for adjustable speed and a USBC power converter to provide 5 volts to all other components through the slip ring.

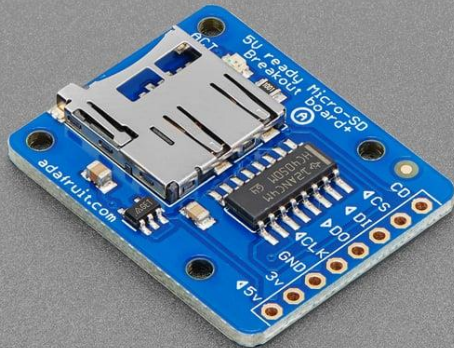
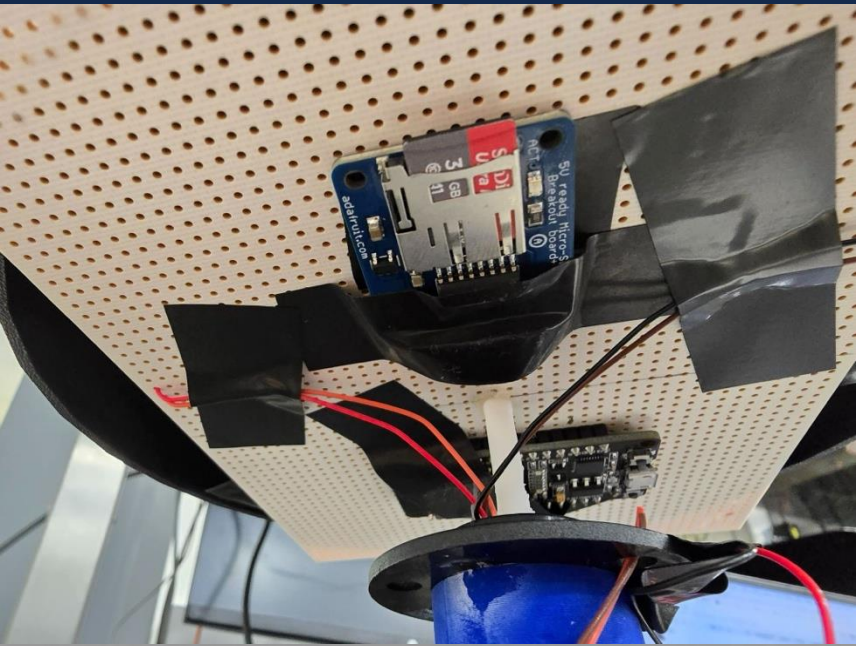
6. Power Transfer Unit

A Slip Ring was used to supply power to rotating portion of the globe such as the LEDs, MCU, and microSD card reader. It consisted of 4 wires rated for 2 A but only 2 were used. Provided constant 5V to these components.



7. SD Card Reader

We utilized SD Card Reader and an arbitrary storage sized SD card. The SD card serves as a repository of all images that could possibly be retrieved by the ESP32 for display.



8. Hall Sensor

We utilized a AH1815 Hall Sensor which is a non-latching hall sensor. This component serves to detect a magnet and synchronize our LED Display timings.



9. Motor



We utilized a 31ZY motor. While this motor has little torque, it is perfect for this design due to its high amperage rating (2 Amps), high voltage maximum (12 Volts), and maximum spin speed (6000 RPM).

10. Slip Ring

We utilized a Taidacent Hollow Slip Ring to accomplish the challenge of transfer of power from a stationary reference frame to a spinning one. Its voltage and amperage specifications are 240 V @ 2A allowing for proper power transfer.



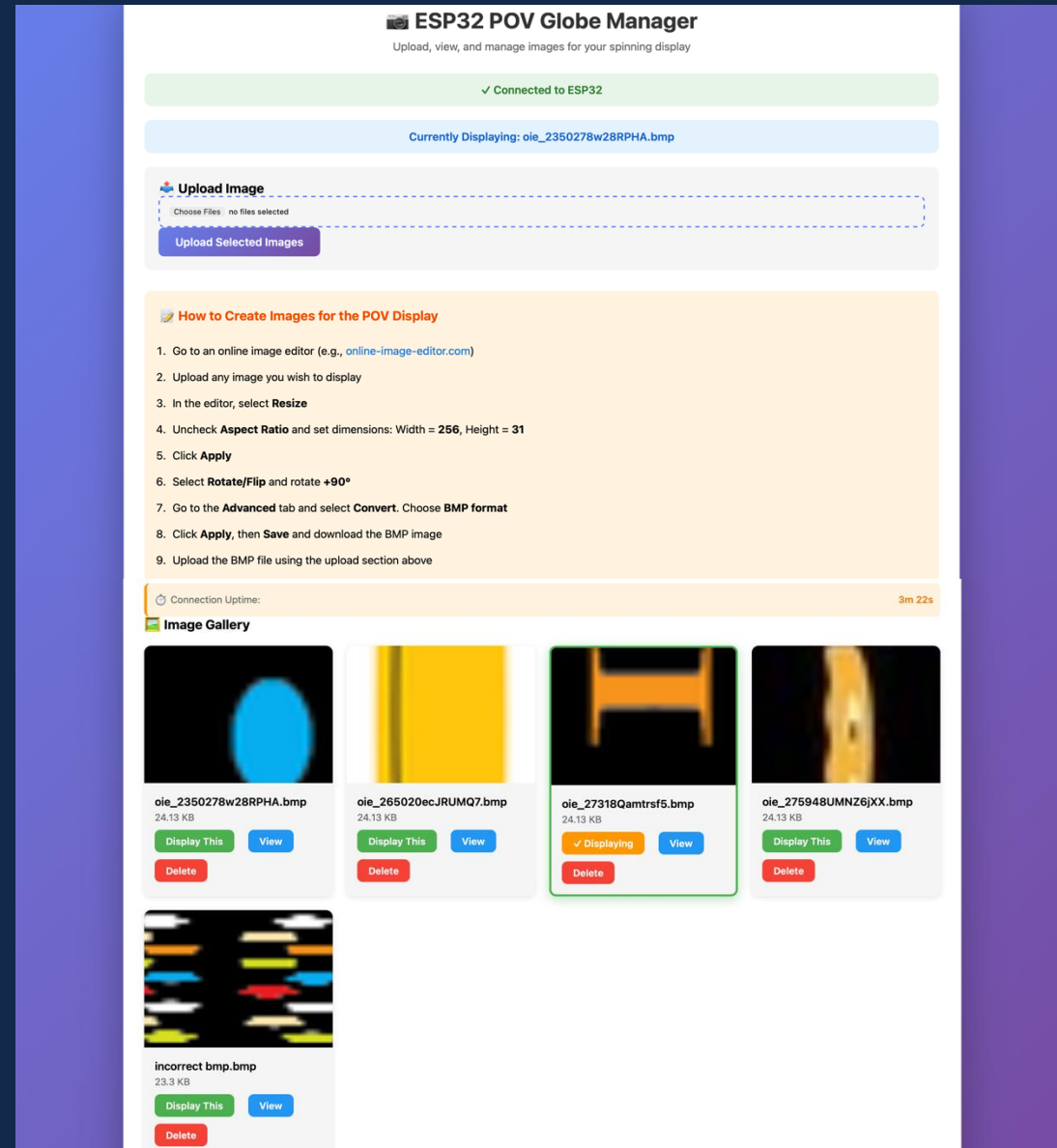
11. LED Strip

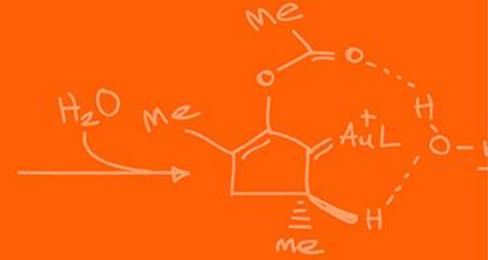
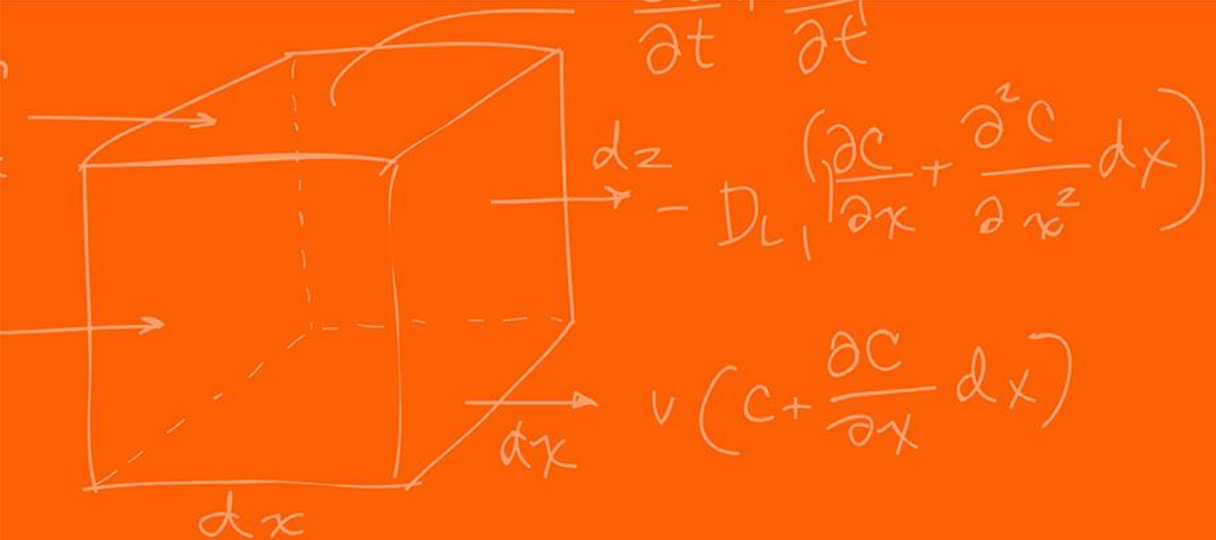
We utilized a BTF-LIGHTING WS2812B IC RGB LED Strip as it allowed for individual LED addressability, customizable length, and a $256 \times 256 \times 256$ RGB color range.



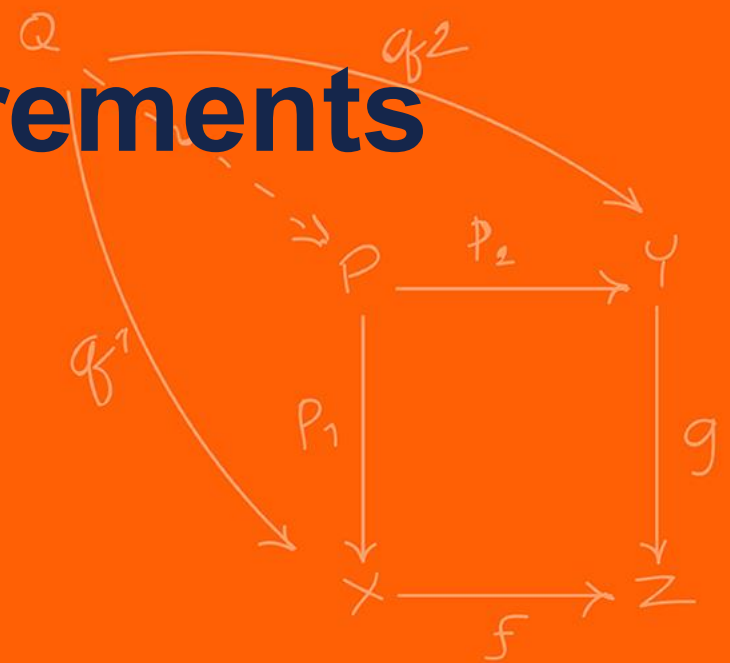
12. Web App

Our Web App was programmed to accommodate the customizable sequencing of image display utilizing the live uploading and deletion of custom images.





Verifications and Requirements



Verifications and Requirements: Did we accomplish our goals?

Basics:

- A spinning array of powered LEDS from external power source.
- RGB Image Display
- Synchronized Image
- Fast Wireless I/O

Externally Powered Spinning LED's

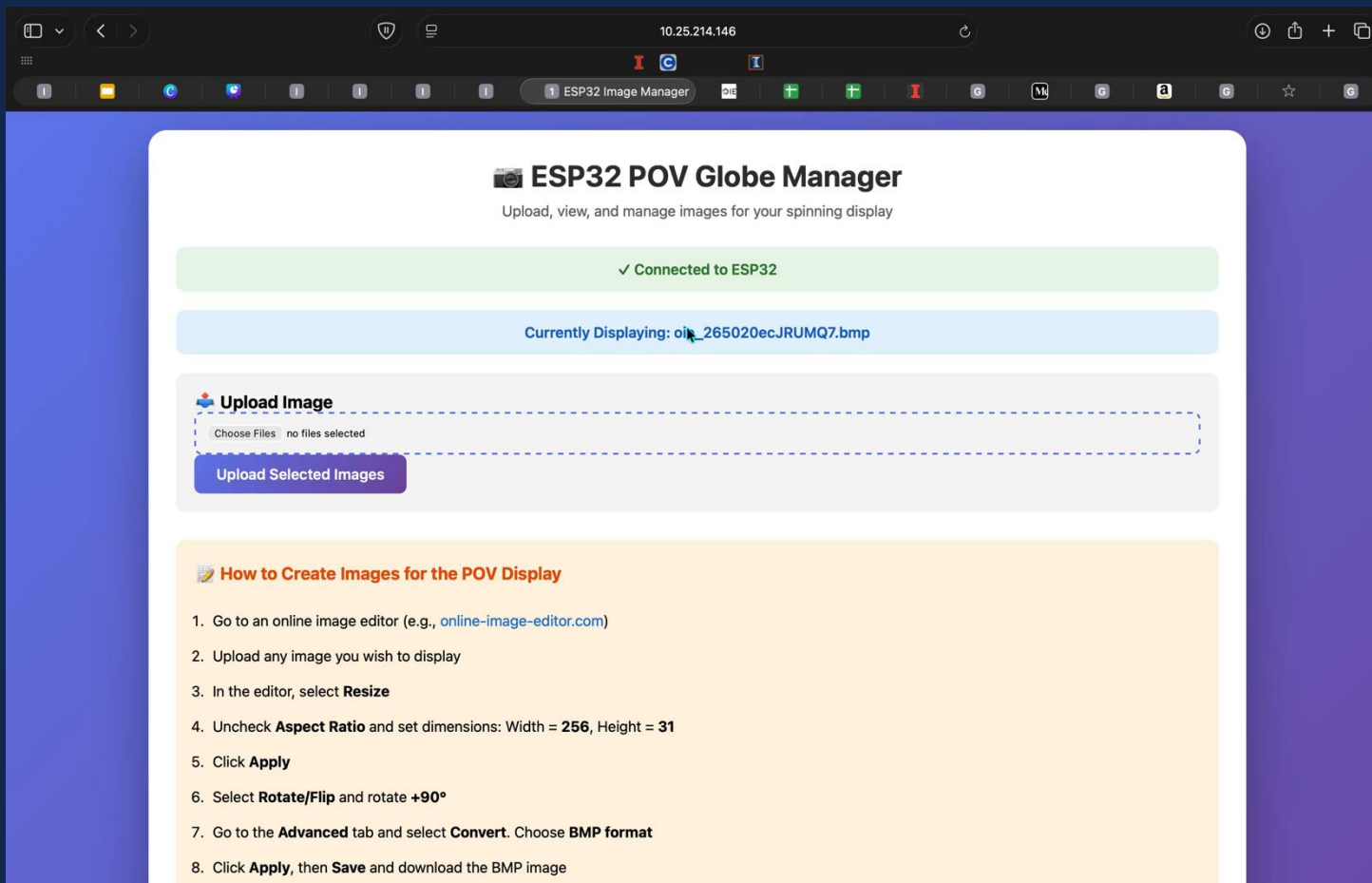


As shown, our LED array is powered externally while spinning. This is indicative of properly transferring power from a stationary to spinning reference frame.

Synchronized Image Display

As will be further expanded on later in this presentation, issues with the LED strip bit rate, and mechanical mishaps in design, disallowed the proper display of synchronized imagery.

Fast Wireless Upload



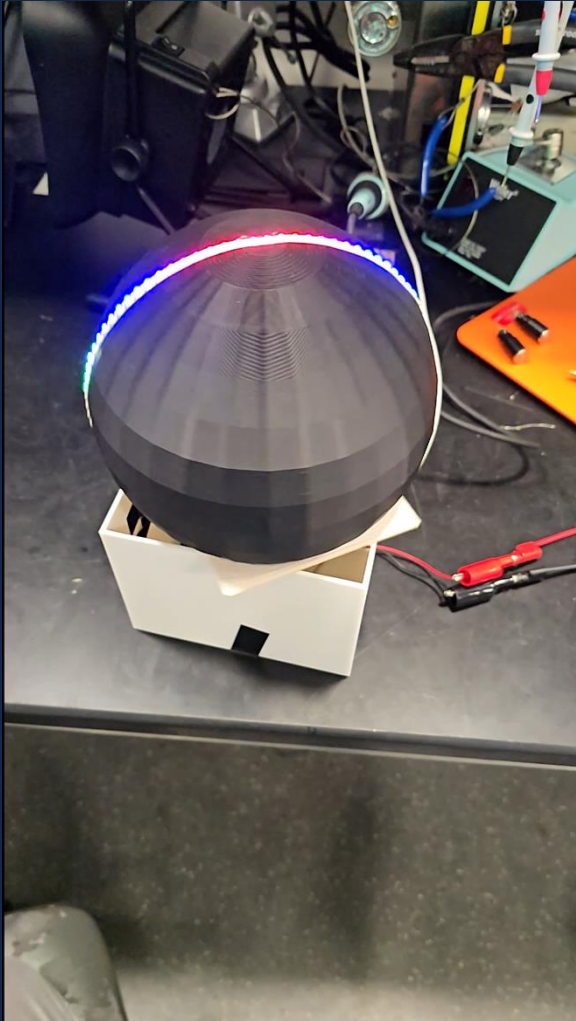
As shown in the accompanying video our upload time for images is fast and effective, far exceeding our desired goal of 3 seconds.

Verifications and Requirements: Continued

Granular:

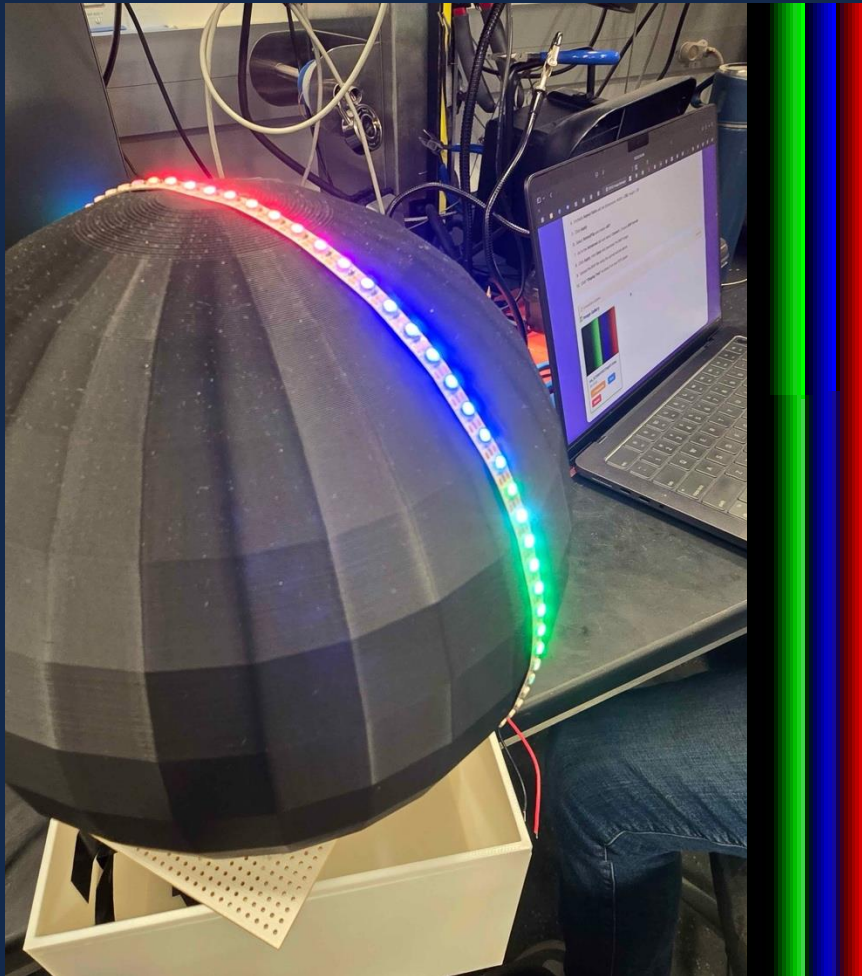
- 600 RPM
- At least 256 Colors
- Pixel Updates
- Adaptable to differing RPMS besides 600
- Reliable Wireless Connection
- Full IO and Image Control

600 RPM



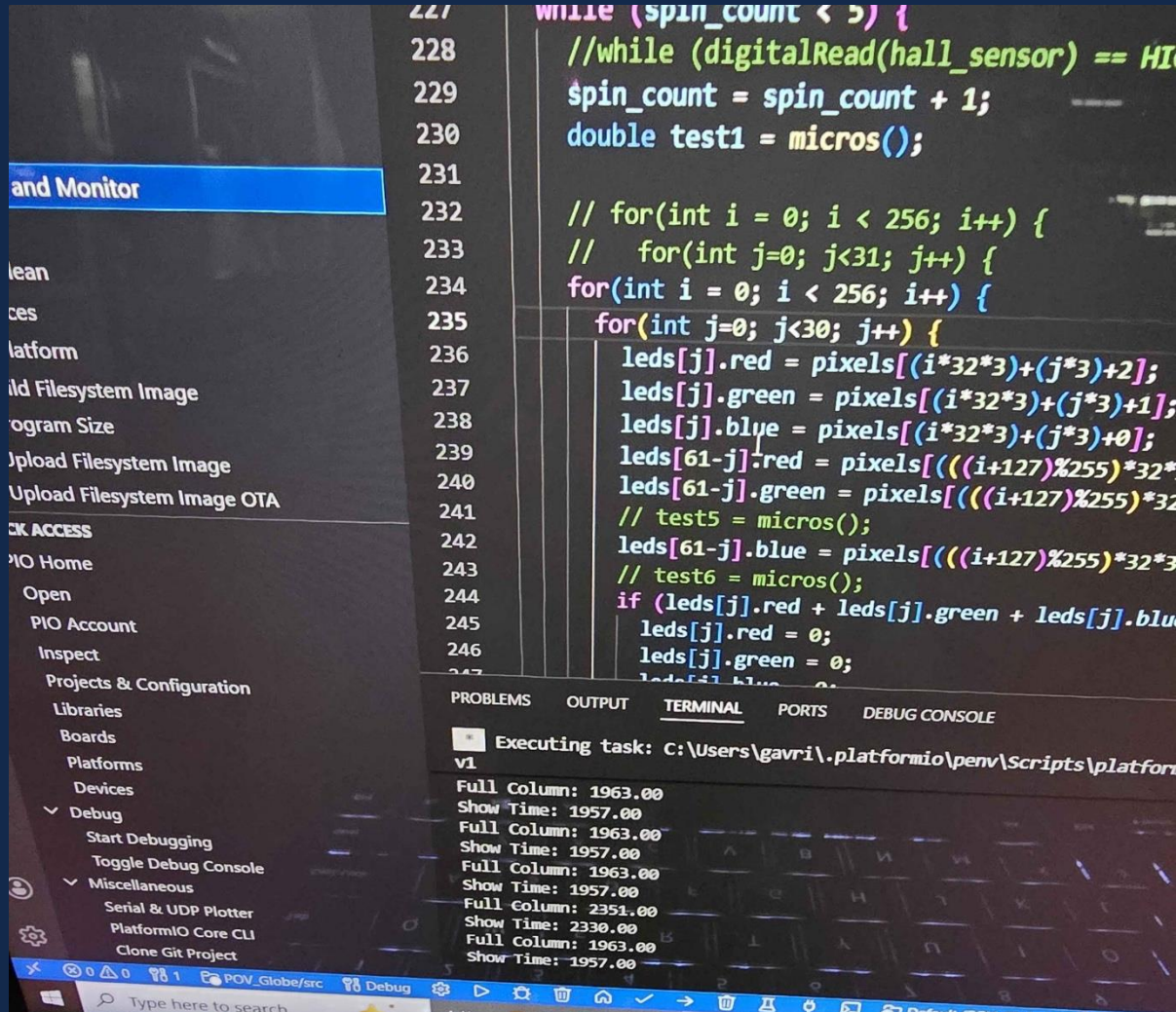
Due to the mechanical mishaps with our current design, we are unable to spin our device up to the desired speed of 600 RPM. However, the following video demonstrates our designs speed ability of 120 RPM.

At least 256 Colors



As a base test of our globes display capabilities, we displayed 8 different levels of brightness for each of the RGB values. This yields at least 512 colors, doubling our initial desire of 256 colors.

Pixel Updates



The screenshot shows an IDE with a C++ code file. The code is a loop that updates LED pixels. The terminal window at the bottom shows the execution of the task, displaying the full column number and the show time for each update.

```

227 while (spin_count < 5) {
228     //while (digitalRead(hall_sensor) == HI
229     spin_count = spin_count + 1;
230     double test1 = micros();
231
232     // for(int i = 0; i < 256; i++) {
233     //     for(int j=0; j<31; j++) {
234     for(int i = 0; i < 256; i++) {
235         for(int j=0; j<30; j++) {
236             leds[j].red = pixels[(i*32*3)+(j*3)+2];
237             leds[j].green = pixels[(i*32*3)+(j*3)+1];
238             leds[j].blue = pixels[(i*32*3)+(j*3)+0];
239             leds[61-j].red = pixels[(((i+127)%255)*32*3
240             leds[61-j].green = pixels[(((i+127)%255)*32
241             // test5 = micros();
242             leds[61-j].blue = pixels[(((i+127)%255)*32*3
243             // test6 = micros();
244             if (leds[j].red + leds[j].green + leds[j].blue
245                 leds[j].red = 0;
246                 leds[j].green = 0;
247                 leds[j].blue = 0;

```

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

Executing task: C:\Users\gavri\.platformio\penv\Scripts\platformio.exe

Full Column: 1963.00
Show Time: 1957.00
Full Column: 1963.00
Show Time: 1957.00
Full Column: 1963.00
Show Time: 1957.00
Full Column: 1963.00
Show Time: 1957.00
Full Column: 2351.00
Show Time: 2330.00
Full Column: 1963.00
Show Time: 1957.00

At 1200 RPM, our upper limit, we would need to write information to our pixels faster than 0.2 milliseconds.

At 480 RPM, faster than .48 milliseconds.

As shown our LED's columns were being updated every 2 milliseconds, 10 times slower than needed.

Pixel Updates Continued

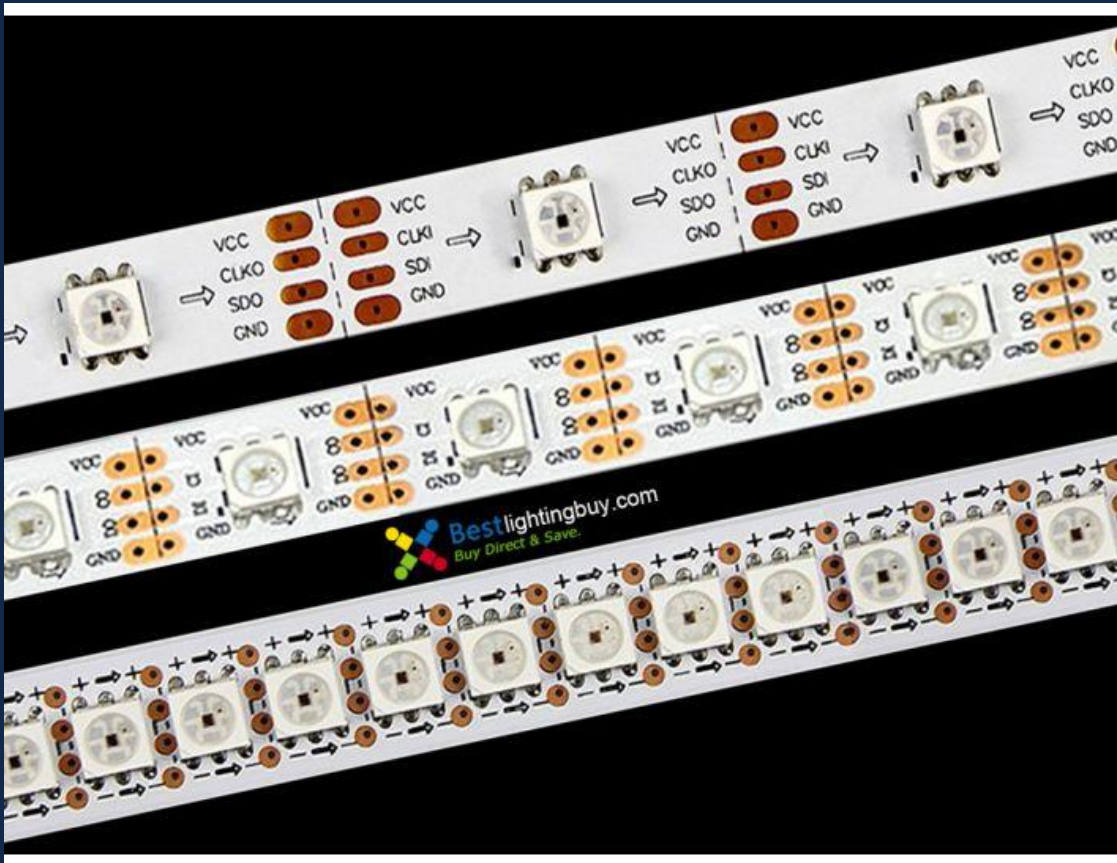


- The WS2812B LED strip requires 24 bits per LED.
- Bit Rate of 800 kHz.
- Each bit needs 1.25 μ s.
- Each LED needs 30 μ s.
- Total time = $62 * 30 = 1860 \mu$ s.

Source:

https://www.digikey.com/htmldatasheets/production/2371852/0/0/1/ws2812b-led.html?gclsrc=aw.ds&gad_source=1&gad_campaignid=120565755&gbraid=0AAAAADrbLliPi33eoHc7gYS7GQQx_jXZT&gclid=Cj0KCQiA6NTJBhDEARIsAB7QHD2DYsHnz5aWtXo1SixjOg0YdCfDQq3N6JQqI5yBwSGTzfz95ru-wukaAkeXEALw_wcB

Pixel Updates Continued



How could this have been fixed?

- SK9822 LED strip.
- Full RGB
- Bit Rate of 20 MHz.
- Each bit needs 50 ns.
- Each LED needs 1.6 μ s.
- Total time = $62 * 1.6 = 99.2 \mu$ s.
- Accommodate up to 100 LED's comfortably.

Source:

https://www.pololu.com/file/0J1234/sk9822_datasheet.pdf

RPM Adaptability



Limiting Factor:

- Our issues with LED Strip data rate do not allow our synchronization to properly take effect.

Successes:

- Our system is able to detect and begin our image display upon detection of our magnet.
- Therefore, with proper LED strip, hall sensor/magnet synchronization is likely workable.

Reliable Wireless Connection

ESP32 Image Manager

Upload, view, and manage images on your ESP32

Connected to ESP32

Upload Image

Choose Files no files selected

Upload Selected Images

How-To: Create a New Image for Display

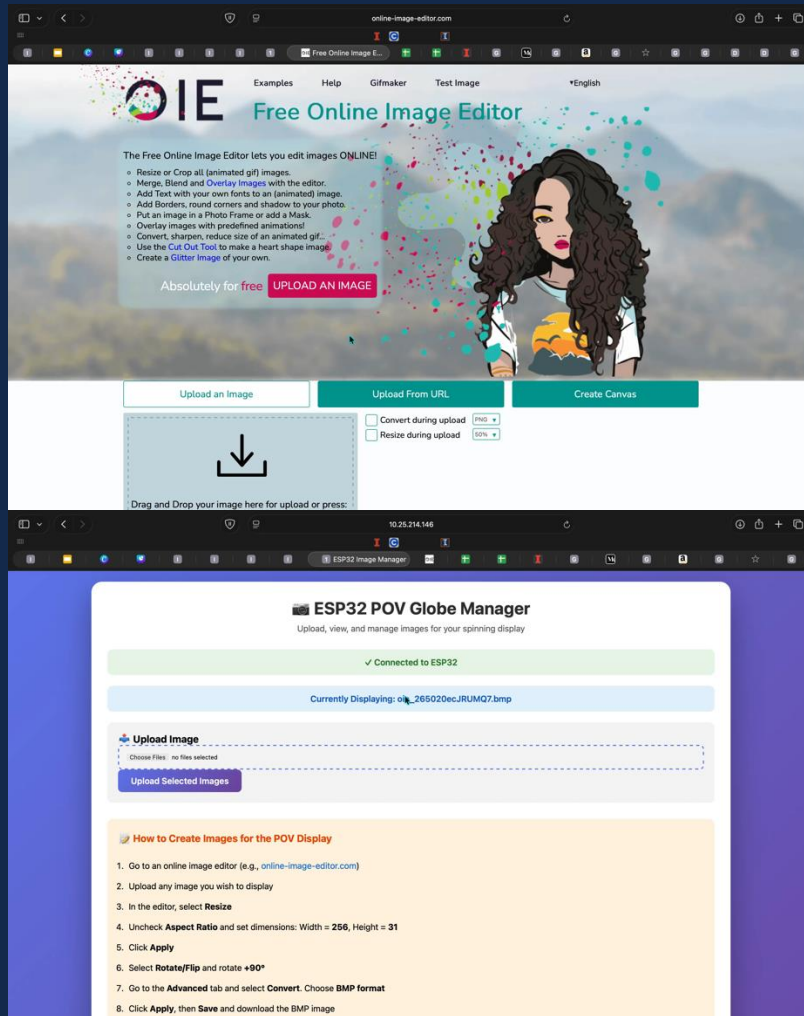
1. Go to an online image editor (e.g., online-image-editor.com).
2. Upload any image you wish. This should open a new editing window.
3. In the new window, select **Resize**.
4. Uncheck **Aspect Ratio** and set Width = 256, Height = 31.
5. Click **Apply**.
6. Select **Rotate/Flip** and rotate **+90°**.
7. Go to the **Advanced** tab and select **Convert**. Choose BMP format.
8. Click **Apply**, then **Save** and download the BMP image to your computer.
9. Upload the saved BMP image using the upload section above.
10. Wait for it to appear on the globe display!

Connection Uptime:

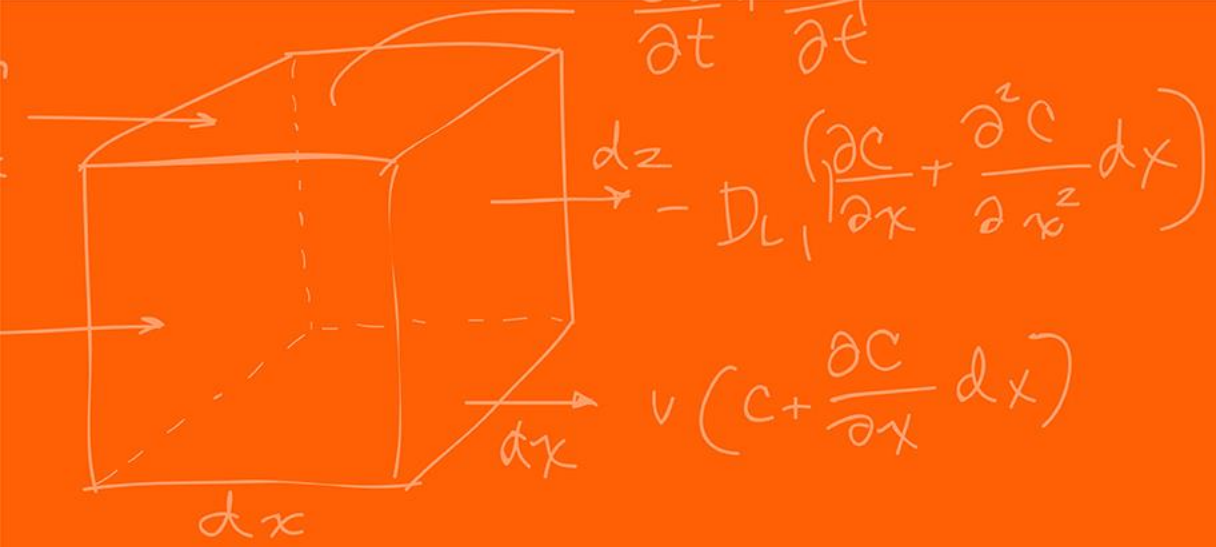
2h 27m 32s

Our testing showed that our wifi connection to the ESP32 was continuous and effective over a period of at least 2 hours. The only reason for disconnect we saw resulted from other code malfunctions which were addressed.

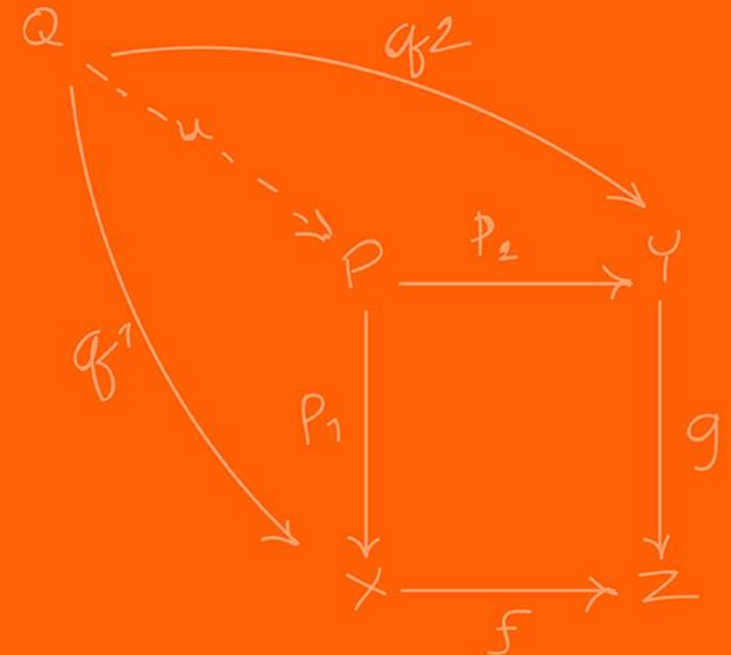
Full I/O and Image Control



Using a combination of a customized web app and free online resources, we created a streamlined process for taking any image, uploading it, and displaying it all remotely from the globe.



Conclusion



What did we learn?

Hall Sensor Issues

Hall sensor positioning is crucial to allow for proper detection and synchronization.

Center of Mass

Proper Center of Mass positioning is crucial to allow for reliable high speed spinning with minimal external supports.

Support

More streamlined spin axis support supplements errors in mass centering without invasive obstruction.

LED Strip Issue

Data line of LED strip does not support fast enough LED update speeds.

What are our next steps?

Update Web App

Adding more functionality to the POV globe would be great to be able to do more things such as displaying text and real time.

Replace LED Strip

Acquire and install an LED strip with a bit rate that is compatible with our design

Fix Support

Redesign 3D-printed components or outsource it to the Machine Shop.

The background is a solid orange color with a collage of white line drawings. In the top left is a 3D cube with axes labeled dx, dy, and dz, and a differential operator D_L with partial derivatives. To its right is a sphere with points A and B and angles α and β . In the top right is a chemical reaction showing a five-membered ring with an oxygen atom and a gold atom (AuL) reacting with H2O. In the bottom left are three cyclohexane rings with different hydrogen atom orientations. In the bottom right is a thermodynamic cycle diagram with states Q, P, X, Z, Y and various energy and pressure labels.

Thank You! Any Questions?



The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN