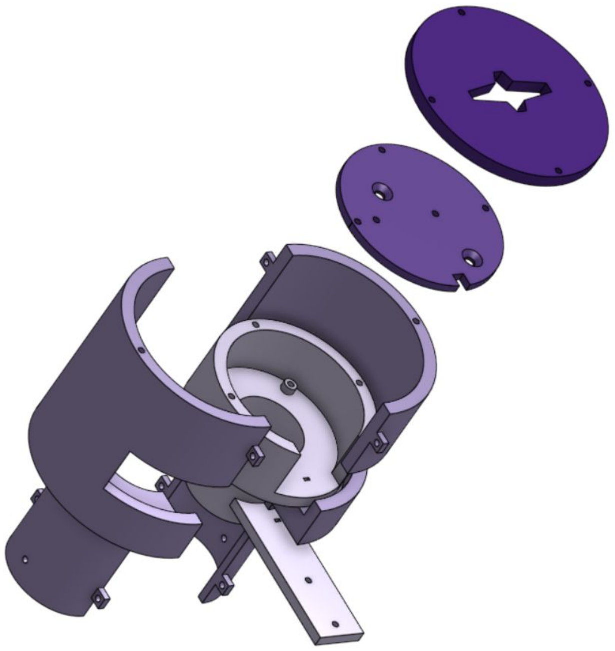
Northwestern | McCORMICK SCHOOL OF ENGINEERING

Department of Electrical Engineering and Computer Science
EECS 395/495 – Engineering System Design II
Spring 2017



Belobog Studios

Pan-Tilt-Zoom Webcam
Final Report

Team: Caleb Burton, Alex Gangwish
Submitted to: Professor Ilya Mikhelson
Report Due: 6/9/2017

Table Of Contents

Executive Summary

This report details the design and implementation of a pan-tilt-zoom webcam by Belobog Studios, performed by founding members Caleb Burton and Alex Gangwish. Figure 1 below shows the final result of this endeavor.

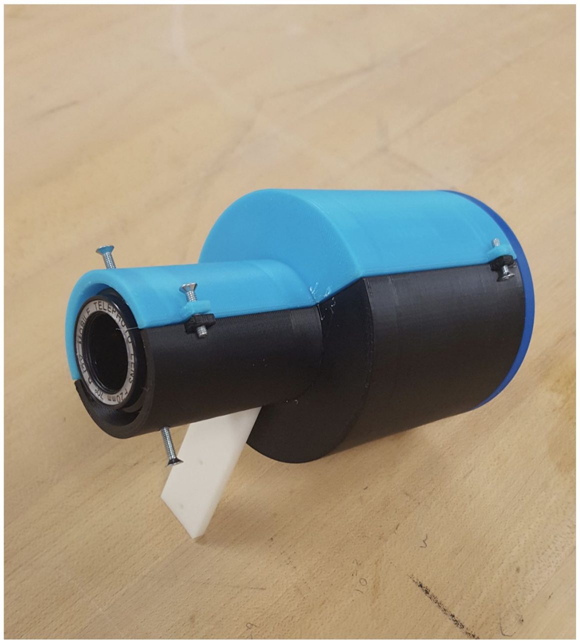


Figure 1: Final Result

There are many components to the pan-tilt-zoom webcam design: an external smartphone zoom lens; a pre-built pan-tilt base; a custom-built 3D zoom enclosure; a PCB containing a camera, a WiFi chip, a microcontroller, and two voltage regulators; and a remote Raspberry Pi server to host the web application that controls everything.

Through the course of this project we improved on and expanded on the skills we obtained in the first term of this course. For example, we gained a great deal of experience in 3D modelling and printing when designing our custom zoom enclosure--it was a long journey to finally produce a print that met all of our needs, but many lessons were learned. Additionally, our troubleshooting and debugging skills were put to the test by the various (sometimes quite strange) issues we encountered as we developed our design.

Introduction

In this course we designed and implemented a pan-tilt-zoom webcam with a web application for remote motor control and image viewing. To accomplish this we utilized a webcam similar to the one designed in the last term of this course along with a pre-built pan-tilt stand (Figure 2) and a custom-built zoom enclosure (Figure 1).

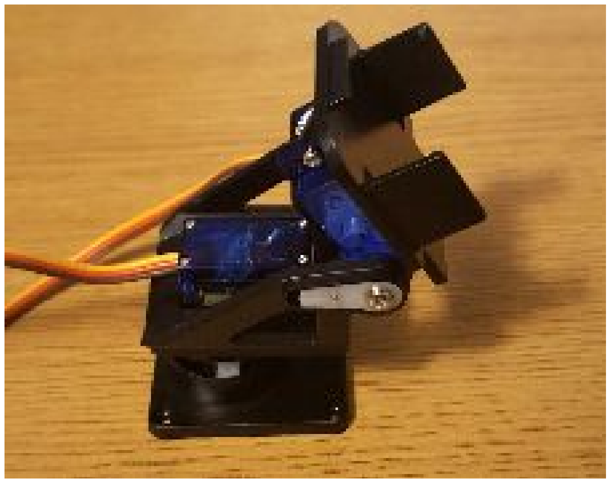


Figure 2: Pan/Tilt Base

Design Description

System Overview

The physical webcam system involves an embedded board that combines a camera, microcontroller unit (MCU), WiFi module, and servo motor headers containing power, ground, and pulse-width modulation (PWM) pins. Once the board is assembled into a 3D-printed enclosure with the three servo motors, it communicates with an off-board server using HTTP requests and websockets. The off-board server also listens for HTTP and websocket requests from web browsers directed to the correct IP address, and serves up a website that allows users to interact with the webcam.

Block Diagram

A summary of this system is shown in Figure 3.

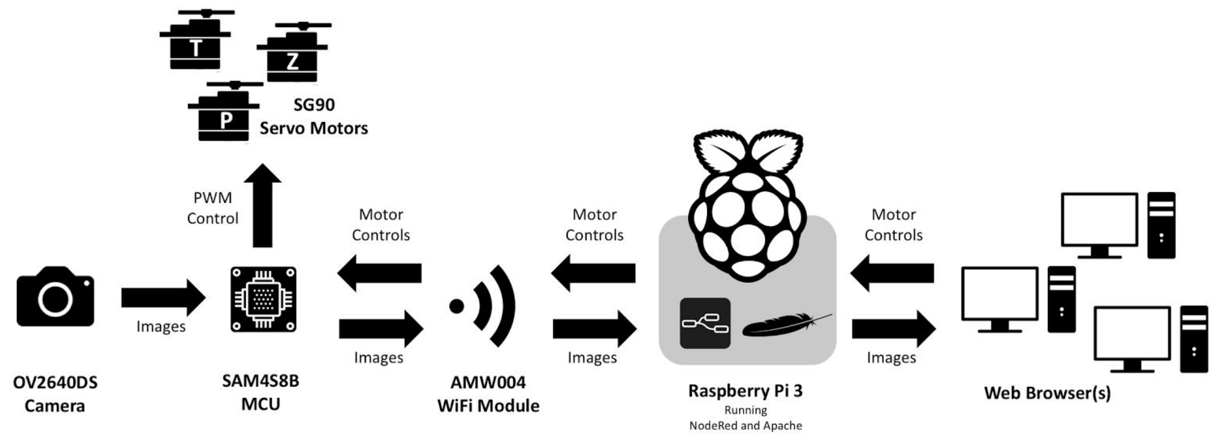


Figure 3: Block Diagram of Entire System

Algorithms and Code

In the complete system, there are three separate programs running simultaneously:

- MCU code
- NodeRED website back-end
- HTML/JS/CSS website front-end hosted with Apache

MCU

The microcontroller code was written in C using the Atmel framework. After initializing the AMW004 module, the program cycles endlessly through a loop where it checks the OV2640DS camera for an image, sends an HTTP POST request to the Raspberry Pi server, and checks the server for any P/T/Z requests that came in since the last image was posted. These requests are parsed into PWM signals, which are then sent to the respective motors.

NodeRED

The NodeRED back-end provides a graphical drag-and-drop interface for NodeJS. The "flow" that we created for this project had three inputs: a websocket for accepting P/T/Z values from the browser(s), a websocket for sending those values to the embedded webcam, and a URL for accepting HTTP POST requests that are automatically written to a JPEG file in the Raspberry Pi's filesystem. Every time the file is updated, one byte of data is sent to the browser(s) that causes it to refresh the image being displayed to the user. The full flow can be seen in Figure 4.

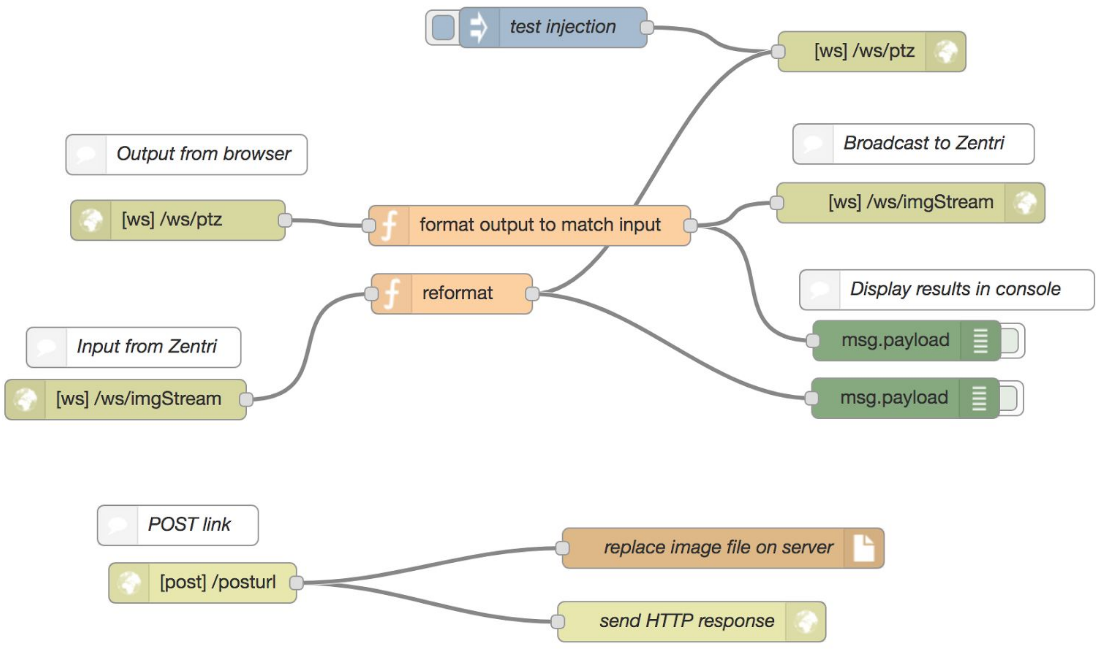


Figure 4: NodeRED Flow

Website

The website displays the most recent image written to the server, as well as the timestamp for that image. Two sliders allow the user to control the webcam's pan and tilt, while a rotary knob at the bottom controls the webcam's zoom. Figure 5 shows a screenshot of the site before any images were captured. P/T/Z values are encoded into a 6-byte ASCII string, which is relayed to the MCU via the Raspberry Pi before being decoded on-board. The use of ASCII to transmit numerical information is unintuitive at first, but is useful for debugging purposes since the output of the AMW004 module is most easily interpreted as a string.

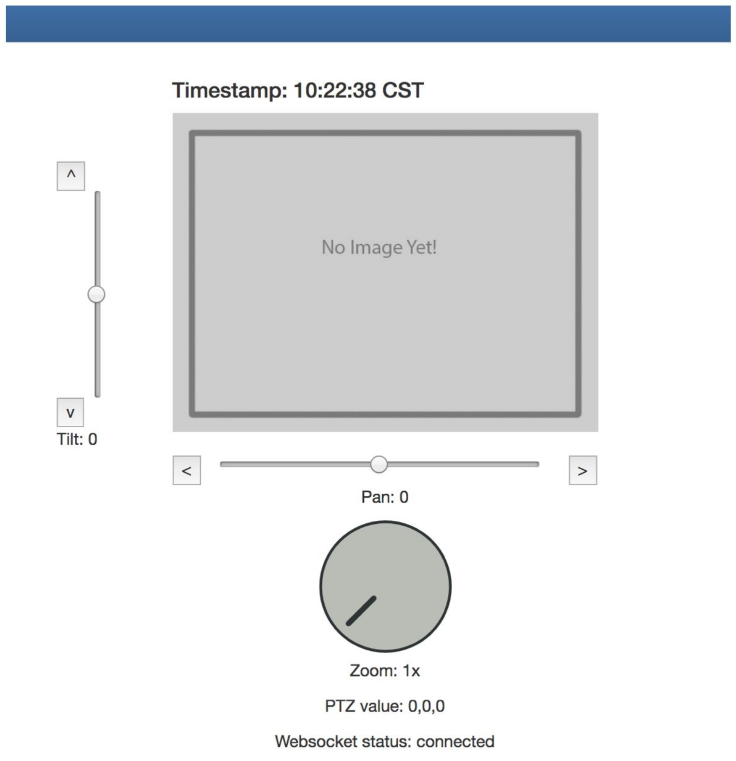


Figure 5: User-facing Website

PCBs

We had two different PCBs printed for this project – the first was a breakout board to test the capabilities of our voltage regulator, while the second was our final circular PCB intended for use in our zoom enclosure.

Voltage Regulator Breakout

We created a breakout board for the Texas Instruments LM317 voltage regulator to ensure that one 12V power supply could be used to simultaneously create a 5V and a 3V power supply for the servos and the webcam, respectively. To accomplish this our board has a 2-pin header input that allows us to use a power supply in the lab to test a variety of different power inputs. The board also has a 3-pin pan-tilt-zoom header that allows for control signals to be passed through the board.

The board also contains 4 potentiometers to allow for adjustment of the output voltages to a variety of levels. When properly tuned, the board outputs to three 3-pin servo headers for the pan, tilt, and zoom servos, each containing a pin for the PWM control signal, the 5V power signal, and a ground signal. The board also produces two 2-pin header power outputs for 3V and 5V, to run power lines to a microcontroller breakout system. This breakout can be seen in

Figures 6 and 7. Note that in Figure 7 we slightly modified our design by soldering in two through-hole resistors in place of two of the originally intended potentiometers. This was done to hold those resistances constant while we adjusted the other two potentiometers to reduce variability.

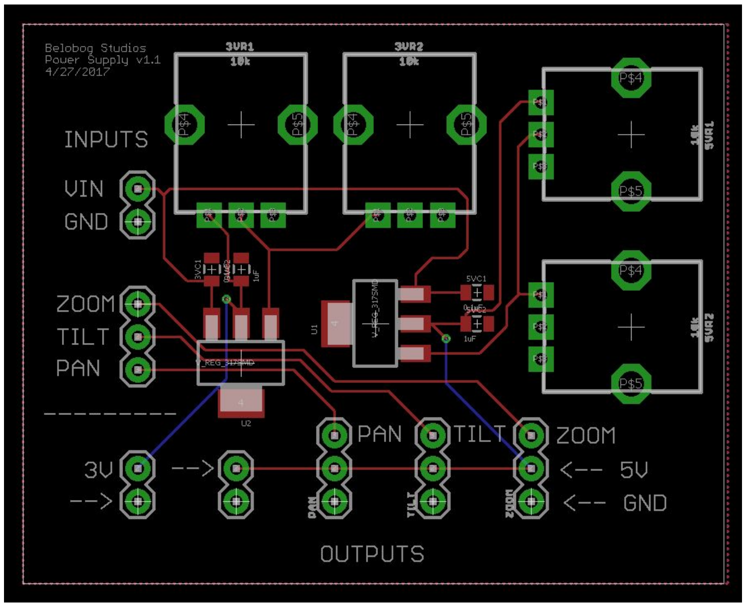


Figure 6: Voltage Regulator Breakout PCB Design

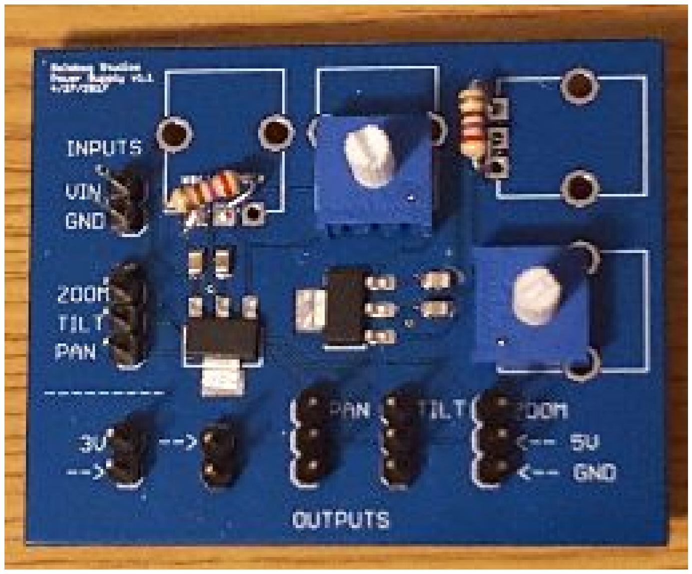


Figure 7: Voltage Regulator Breakout PCB, Assembled

Final PCB Design

The next PCB we designed was our final complete embedded system board, intended for use inside our zoom enclosure. To improve the efficiency of our design we decided to implement this PCB as a circular board (Figure 11 shows one of our earlier enclosure designs intended for a square board, where the wasted space is quite apparent). The board was designed with the camera as the centerpiece so that when placed in the enclosure it would be lined up with the center of the lens.

The board also includes the microcontroller, WiFi chip, 2 LM317 voltage regulators to distribute 5V and 3V power, a barrel jack to supply a 12V input, headers for the servo controls, and 3mm grounded vias for mounting in the enclosure. The final design has a diameter of 53mm--this was slightly larger than we had originally hoped for, but the overall space savings from switching to a circular board more than made up for it. The final design can be seen in Figures 8 and 9.

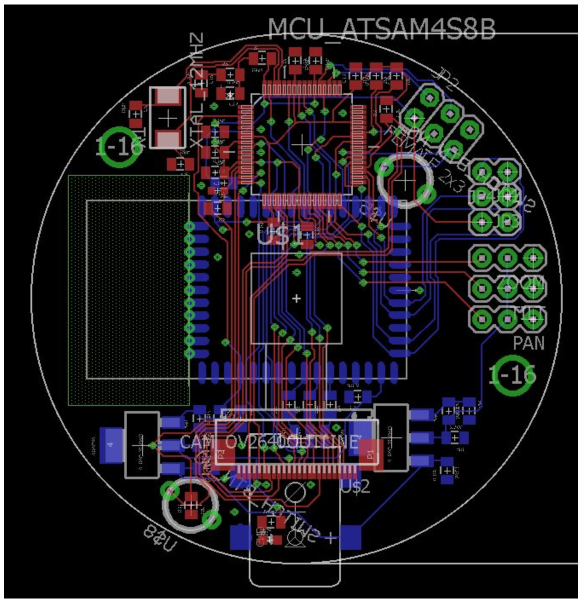


Figure 8: Final PCB Design

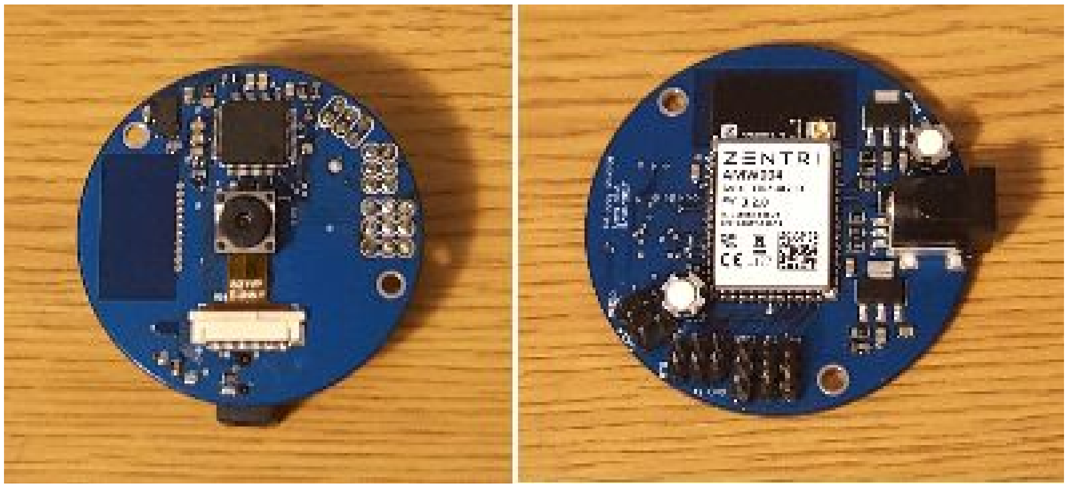


Figure 9: Final PCB, Assembled

Issues

A few issues with our design became apparent after testing our board. Most notably, the voltage regulators become extremely hot after what should be considered relatively light usage–and fail altogether when called upon to provide enough current for running multiple servo motors at once. This led to several problems in our development of our final prototype, as it was difficult to run our webcam for long periods of time (especially in the enclosure, where the heat problems couldn't as easily be countered from convection into the air) and nearly impossible to test multiple servos together.

After attempting many different debugging options to try to pinpoint the source (e.g. replacing the regulators, checking for floating connections, changing the adjustment resistors to different values, etc.) we were still unable to produce a worthwhile solution to the issue. Our belief is that the source of the problem is most likely the traces carrying power from the voltage regulators, which may be too thin to carry the requisite current.

A more minor issue is that there is some unused space on our board. It should be possible with some tweaking to create a smaller design in the future that utilizes this space better, or at least spaces the components in a way that allows them to be soldered more easily.

Potential Improvements

The first change we would be interested in making in a future printing would be to thicken the traces carrying the larger power loads in our design–this would be a relatively straightforward change that would require us to change very little about our design. We believe that many of the problems we encountered during prototyping could well be solved by this simple change to our PCB.

Additionally, in future prints we would like to experiment with creating a smaller design. The first, most obvious way to do this would be to make better utilization of the space on our circular board by repositioning components and perhaps rotating them to less-standard angles. Another idea we had was to move the power supply components (the barrel jack, LM317 regulators, and associated resistors and capacitors) onto a separate power supply board. This two-board design may allow for a smaller enclosure by keeping the motor headers and power regulation in a separate unit, and would make it easier to troubleshoot and fix the current issues we mention above.

3D Printing

3D printing was an extensive, continuous process throughout the quarter that adapted frequently alongside our overall design. Much of the time spend in this arena involved designing models in Onshape, waiting for them to print, and then deciding to do something completely different and starting all over again.

Initial Design

We brainstormed a number of different approaches during the first week of the project, but for the majority of the quarter we focused on creating a fully enclosed tube design that would house both the webcam and the zoom servo motor inside with no exposed pieces. Figure 10 shows the original sketch of this concept. It features an inside tube that grasps the back of the zoom lens and holds the webcam and servo motor, and an outside tube that attaches to the focus knob of the zoom lens and the motor arms of the servo.

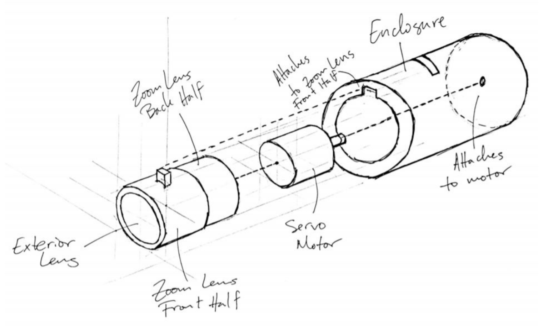


Figure 10: First Concept Sketch

First Prints

The next major step was translating this concept sketch into a 3D model that could feasibly work for our real design. The first model we created can be seen in Figure 11. It features a square PCB mount designed to fit a board approximately the size of the designs we produced in the first term of this course. A very large outer tube was required to rotate around this square enclosure, creating a somewhat unwieldy design.

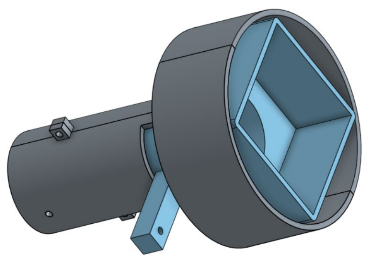


Figure 11: First 3D Model

Printing this iteration allowed us to confirm some of our theories about how the design might work. Specifically, it allowed us to test whether a small servo would even be capable of moving such a large enclosure (even without the internal friction of the lens). By putting our zoom lens in the enclosure and fastening everything together we were able to prove that it would in fact take very little force to operate, provided everything fit together properly. Based off of what we learned from this design, we moved toward our final enclosure.

Final Prints

An exploded model of our final enclosure can be seen in Figure 12. This design features a circular PCB mount, designed to fit our final circular PCB design without the wasted space that of our earlier square version. The inner tube contains a long, flat mounting post designed to be screwed onto the pan-tilt base and has a lid with two large holes for accessing the MCU reset and WiFi setup buttons and two small holes for mounting the servo motor, as well as a notch for feeding out wires. The outer tube fits snugly around the inner tube and has a small slot for the mounting post and a larger slot for the power and servo control lines. The outer tube has a lid with a notch that fits the arms of the servo motor, so that when the servo moves it rotates the outer tube around the inner tube.

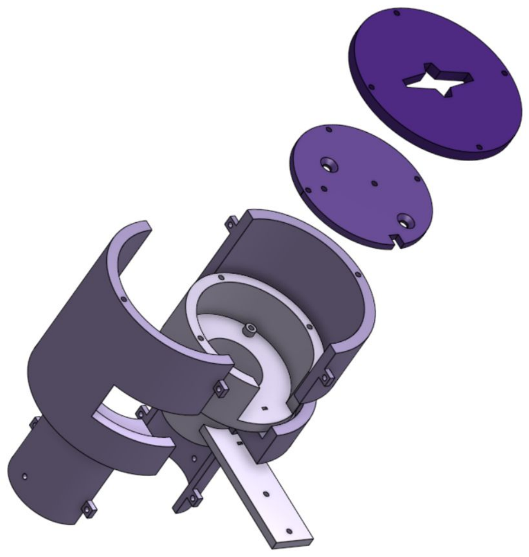


Figure 12: Final 3D Model

Overall we were very happy with how this final design turned out. Although we would be interested in testing new designs with smaller PCBs and different motors, we were satisfied with the performance of this particular design. It was sturdy enough to support itself while still remaining lightweight and was smooth enough to allow easy operation by a small motor.

Final Product

Initial Goal vs. Final Result

Our final product was similar to our initial conceptualization, but did not attain all the goals we had set at the beginning of the quarter. In many cases, some of which are described below, we achieved part of the goal but in doing so we encountered new obstacles that limited our results. One example of this was our goal of moving our website to an off-board server to increase the frame rate of the stream. We succeeded at moving the site off-board, but in the process we discovered an issue with HTTP request timeouts that actually worsened our frame rate. All-in-all, our final result clearly expresses the spirit of our initial goals even if it did not necessarily meet them.

Performance and Limitations

There were three main limitations that caused the disparity between our initial goals and our final product. The first of these involved the zoom lens itself, which did not actually have an adjustable zoom but rather had a fixed zoom with an adjustable focus. We later acquired a lens that had a true adjustable zoom as well as focus, but the new lens' bulk, high turning torque, and oddly-placed zoom adjustment knob led us to remain with our initial lens despite the fact that its zoom was fixed. Both lenses are shown in Figure 13, with an example of their capabilities shown in Figures 14 and 15.

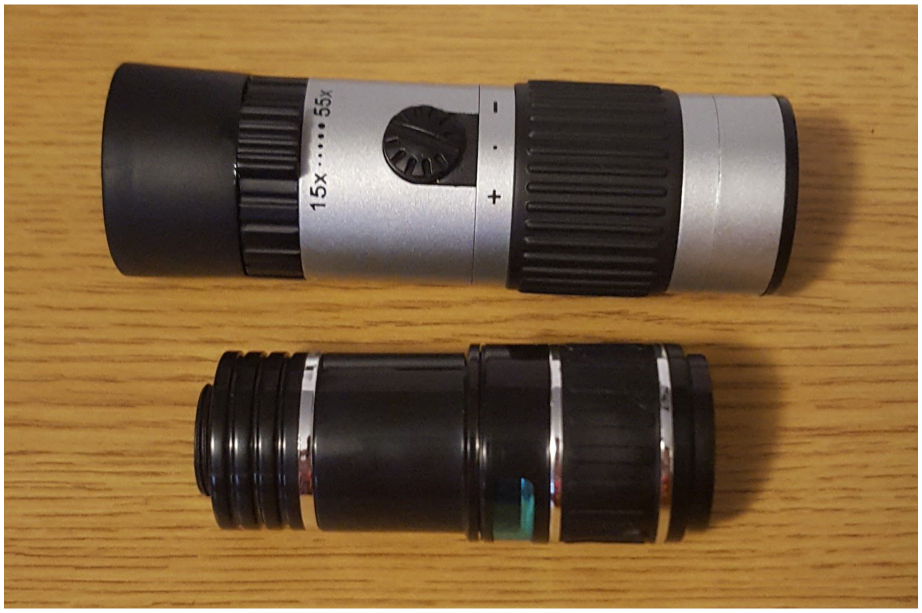


Figure 13: Fixed Zoom Smartphone Lens (bottom) and Adjustable Zoom Monocular (top)

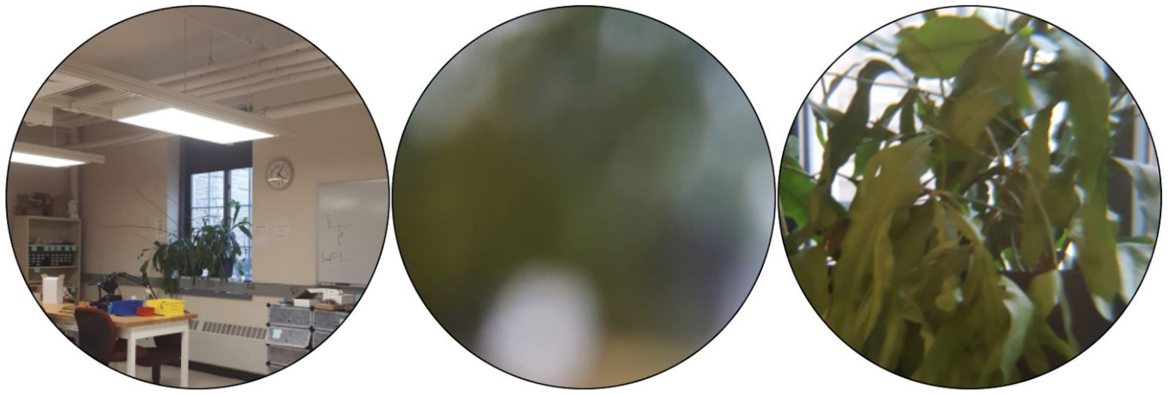


Figure 14: Fixed-Zoom Lens. From left to right: No lens, Focused, and Unfocused

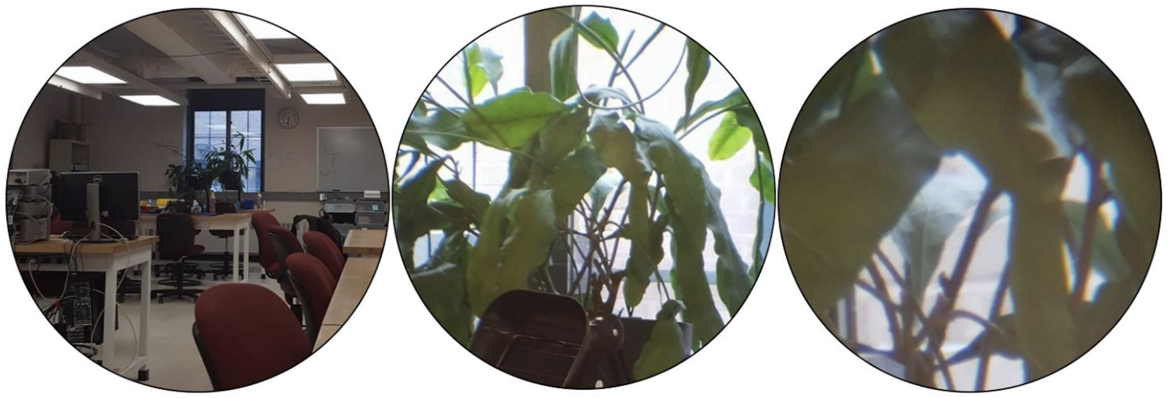


Figure 15: Adjustable-Zoom Lens. From left to right: No lens, Minimum (15x), and Maximum (55x)

The second involved our motor control. A misinterpretation of the SAM4S datasheet led us to design the final PCB with only two functional PWM pins rather than three, since we anticipated using a PWMFI0 pin as an output to complement the PWMH0 and PWMH1 pins. Unfortunately, the PWMFI0 pin is an input-only PWM Fault pin rather than a standard PWM output, and we were therefore unable to drive all three servo controllers at once. We attempted to jump the line to a PWMH2 pin, but this was unsuccessful. Furthermore, our power supply could not source enough current to drive more than a single servo motor at a time, as described earlier.

The final difference involved the frame rate for our feed, which was much slower than our goal. Our main rationale for using an external server to host the image was that it would allow us to increase the frame rate substantially, but at the conclusion of this quarter we were unable to make that goal a reality. The frame rate hovered around 0.25 to 0.5 Hz when the site was hosted directly on the AMW004 module, but that rate plummeted to around 0.1 Hz when the site was hosted on the Raspberry Pi. We never definitively determined the cause of this slowdown, but we were able to pinpoint the point in the code where it occurred. After talking with our classmates who were encountering a similar issue, we came to believe that the root cause was the timeout length on our HTTP requests. About 25% of the POST requests failed, but the AMW004 module would not accept these failures for about 5 seconds. We attempted to adjust the timeout settings, but it wasn't clear that this was ever successful. In the intervals between the failed HTTP requests the frame rate was closer to the 0.5 Hz that we expected, but the failed requests would often occur back-to-back and lead to a full 10 (or more) second delay between frames.

Challenges Encountered

In addition to the challenges listed above, which we did not surmount by the time the course ended, we encountered many challenges that we did indeed solve.

A large issue we faced early on was designing a good enclosure for our final PCB design that wouldn't be too massive to print or too heavy to stay upright. As discussed earlier in the report, planning for a square board led to some wasted space in our enclosure and a generally ugly design. To combat this issue, we pivoted to using a circular PCB (with some inspiration from our fellow classmates). This allowed us to drastically reduce the size of the exterior of our enclosure without reducing the amount of surface area we had to work with.

There were also many, many lessons learned when it came to 3D printing. Some of them were simple, like learning to double check that ZSuite actually put supports in places that critically need supports (which it does not do a surprising amount of times) or making sure that there's enough filament to finish a print (or at least making sure to be there to hotswap the filament). Figure 16 gives an example of what happens when you forget those steps.

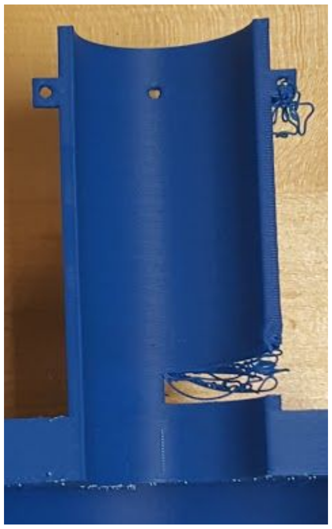


Figure 16: Unsupported Print

Others were not so simple, like learning how to to adjust for warping (changing wall thicknesses, fine tuning the gaps between tubes, etc.) or learning how to find and reinforce stress points (preferably before waiting 6 hours for a print just to snap it in half when you try to put it together).

One of the more bizarre issues we faced in our code was that the Zentri module was unable to open the NodeRED websocket unless there was already some other device connected to that websocket. We never understood why this happened, but after a lot of head-scratching we at least discovered the nature of the problem and a feasible workaround. We wrote a basic dummy webpage that has the singular purpose of connecting to the "imgStream" websocket on the Raspberry Pi. As long as that webpage was open and running, the webcam could also connect to the Raspberry Pi. After the board had connected to the websocket we could close the dummy page without consequence, which only added to the mystery.

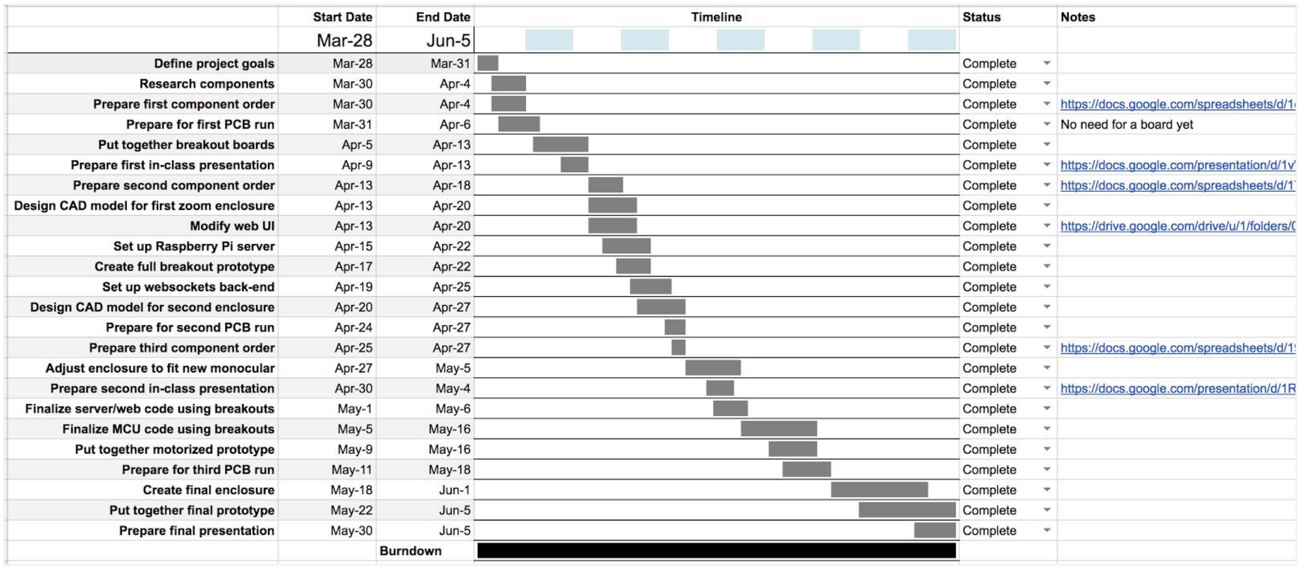
Other challenges with coding involved learning how to produce PWM signals from the MCU. There was a very helpful PWM example project, but it included quite a lot of unnecessary code and was not written with the SAM4S8B in mind. This example only included two PWM channels, rather than the three that we needed, which made it essential for us to actually understand the example code instead of just blindly copying it into our project.

Unlike last quarter, we didn't run into any challenges in terms of soldering our embedded board together. At one point we did attempt to jump out a pin from the MCU and that didn't work, but our camera, MCU, and WiFi module all worked on the first try.

Planning and Organization

Gantt Charts

The main way we planned this project and ensured that it was on track for completion was through the use of a Gantt chart. The completed sheet is shown below in Figure 17. At least once a week, we would update the sheet with our progress and add rows for any new tasks that we had not initially anticipated.



	Start Date	End Date	Timeline	Status	Notes
	Mar-28	Jun-5			
Define project goals	Mar-28	Mar-31		Complete	
Research components	Mar-30	Apr-4		Complete	
Prepare first component order	Mar-30	Apr-4		Complete	https://docs.google.com/spreadsheets/d/1
Prepare for first PCB run	Mar-31	Apr-6		Complete	No need for a board yet
Put together breakout boards	Apr-5	Apr-13		Complete	
Prepare first in-class presentation	Apr-9	Apr-13		Complete	https://docs.google.com/presentation/d/1v
Prepare second component order	Apr-13	Apr-18		Complete	https://docs.google.com/spreadsheets/d/1
Design CAD model for first zoom enclosure	Apr-13	Apr-20		Complete	
Modify web UI	Apr-13	Apr-20		Complete	https://drive.google.com/drive/u/1/folders/
Set up Raspberry Pi server	Apr-15	Apr-22		Complete	
Create full breakout prototype	Apr-17	Apr-22		Complete	
Set up websockets back-end	Apr-19	Apr-25		Complete	
Design CAD model for second enclosure	Apr-20	Apr-27		Complete	
Prepare for second PCB run	Apr-24	Apr-27		Complete	
Prepare third component order	Apr-25	Apr-27		Complete	https://docs.google.com/spreadsheets/d/1
Adjust enclosure to fit new monocular	Apr-27	May-5		Complete	
Prepare second in-class presentation	Apr-30	May-4		Complete	https://docs.google.com/presentation/d/1R
Finalize server/web code using breakouts	May-1	May-6		Complete	
Finalize MCU code using breakouts	May-5	May-16		Complete	
Put together motorized prototype	May-9	May-16		Complete	
Prepare for third PCB run	May-11	May-18		Complete	
Create final enclosure	May-18	Jun-1		Complete	
Put together final prototype	May-22	Jun-5		Complete	
Prepare final presentation	May-30	Jun-5		Complete	
		Burndown			

Figure 17: Completed Gantt Chart

We utilized the same template as the rest of the class, which gave visual feedback by changing colors depending on whether the project was upcoming, active, completed, or behind schedule.

Communication Among Team Members

Much of our communication for the quarter was assisted by Google Drive. We used the service to host all of our documents including reports, presentations, and sketches, giving both team members access to read and edit at any time. Additionally, Facebook Messenger was used for general team communication such as planning meetings.

Splitting Tasks Among Team Members

Although every step of the design process was planned and agreed upon by both team members, we did specialize in certain tasks throughout the quarter in order to increase efficiency. Specifically, Alex took lead on the PCB design and the 3D printing and Caleb took lead on the web development and MCU programming.

Individual sections

Conclusion

- *What You Learned*
- *What You Would Do Differently*
- *Possible Next Steps*

Class Feedback

- *Did you learn as much as you hoped to in this class?*
- *Do you have any suggestions for improvement of the class format or structure to increase learning?*
- *Sum up your thoughts to this project, the class, and your overall experience.*