**Propeller Hat**

Andrew Kats (VC1B) [andrew.kats42@gmail.com](mailto:andrew.kats42@gmail.com)

**CISC 4900**

Fall 2022

**Technical Supervisor**

Gezim Hakramaj

Professor, Brooklyn College

[gezimhakramaj@gmail.com](mailto:gezimhakramaj@gmail.com)

**Abstract**

The purpose of this project is to create a motorized, remote controlled, and battery powered propeller hat.

This project will require electrical engineering, programming, CAD, soldering, and immense amounts of research to create a device that is fun for all ages. All eyes in the room will immediately be on you when you enter with this miracle of aviation on your head.

**Project Details**

My motivation for making this project is that I enjoy making people laugh. If for even a moment, I can get them to forget about whatever they might have going on in their life and just laugh, it is worth it.

In my mind, this hat is going to be a great way to get your friends laughing at a get-together, or just a fun gift to give to a child.

This hat will have a motorized propeller. It will be built in such a way that it is not obvious that the propeller is motorized until the user turns it on.

Instead of it just being one speed, which is what would happen if the motor was just wired directly to the battery, it will have a processor and a motor board which will allow for a very fine adjustment of speed, plus it gives the option to have preprogrammed tricks and perhaps upgraded features in future revisions.

The main reason for implementing it this way is that I find it more fun to just have it spin slowly than at full speed all of the time. The processor will have a Bluetooth module which will allow the hat to be controlled remotely via an app on your phone.

I am going to use CAD software like Fusion 360 to design most of the 3D printed materials and I will use Arduino hardware and software to program the controls. I will also use Cinema 4D for other parts of the project.

I will also be creating my own PCB to hold my components to limit the amount of wires needed to construct this project.

The project will be good enough when the hat can be controlled at all without needing to plug it in and upload code to it. It will be done when the main functions can be controlled remotely and easily using the app.

**Background**

I have never worked with an Arduino board or a Raspberry Pi. I am a novice with CAD and Cinema 4D, which I will have to get good with if I want to make a product that looks good and not just a ball of wire that will get me arrested if I try to go through the airport with it. On the topic of wires, I have never soldered any wires before so I will have to learn how to do that too. Essentially, I will have to teach myself engineering and design for this project to even be presentable, regardless of whatever code I will also have to learn how to write.

**Literature Review**

I started of by researching the core components I am going to need for this project, those being the processor and the motor board. I spent a lot of time on forums and Reddit, learning about which components people have had success with. Naively, I thought that I could just google “smallest processor” and “smallest motor board” and it would magically work. I was wrong. Often I would find that people were having problems with components, whether they couldn’t connect to them through Bluetooth, or Arduino IDE just simply wouldn’t work with them when it said it was supported.

I’ve also spent a lot of time watching YouTube videos and reading articles. I’ve never made a circuit in my life, apart from those snap kits I had as a kid, so I had to learn what all the pins on all of these Arduino boards are for if I didn’t want magic smoke. There is an abundance of CAD tutorials on YouTube and they have been incredibly helpful in my design process for the enclosure that will contain my parts and the PCB that I am designing with CircuitWorks. Interestingly enough, I have been learning Cinema4D (Maxon provides a very cheap license for students) so I can create a captivating video for my presentation and a lot of the concepts are similar to traditional CAD software like Fusion360 or Solidworks, but the videos are usually put together much better, which has greatly improved the efficiency with which I can realize this project.

**Obstacles**

As I am learning how to solder, and working on the prototype, I have discovered that it is very hard to connect as many wires as I have had to, in such a confined space as they will have to be in, by hand. This led me to explore creating a PCB that I would have manufactured by a company such as JLCPCB. This is of course, a much more refined and better looking solution than soldering a ball of wires together, but now I have to learn how to design a PCB, and it introduces new and expensive obstacles, such as, what if there is a mistake in the design? I can’t just rewire it. I now have to redesign and have it manufactured again, which apart from costing money, takes time that I have little to spare.

Another obstacle is accidentally shorting a lithium battery, which I definitely didn’t do.

Here’s a big one. The day before the Final Project is due, the motor board on my PCB died. Given that I had only a small amount of footage of it actually working, I had to find another way to showcase my project. So I took it upon myself to recreate the entire hat in Cinema 4D, not just the small bits I was using before. After many hours, I was able to create an animation that shows the hat in action and I added a few fun elements to it.

**Architecture**



**Software Construction**

Luckily for me, Android Studio does a lot of the file organization for you.

Unfortunately, the Bluetooth Library provided for the Bluno Beetle BLE development board used an old version of Java and Android Studio. It took a lot of trial and error to find the version that would work for me.

I only had a handful of revisions, simply named v1, v2, etc. I would save these files on my computer and externally because I learned the hard way to back up my files.

**Milestones**

- v1 of enclosure design finished and 3D printed.

- Working-ish prototype done, using cheaper and not final parts (useful to discover obstacles early).

- script for video started.

- Full board built.

- Solder necessary wires to the board.

-v2 of enclosure done.

-FINISH THE HAT

**Project Management**

I updated my supervisor on my progress every couple of weeks.

I used a calendar to set deadlines for myself in regards to when I wanted certain features to be finished.

I routinely backed up my files because the first time I started writing the script for my presentation I had a crash and lost hours worth of work.

**Expected Outcomes**

I expected the app to have a ton of features such as sound and notifications, etc.

I expected the implementation of the Arduino code to be much harder than I thought.

**Actual Outcomes**

The app did not require lots of unnecessary features to work. In fact some of them made the app lag so much it would become useless after running for a minute because of all the data being updated.

I didn’t expect the motor board to die. What I learned from this is to always have spares when developing hardware.

I didn’t expect to cram hours of animation into the last day of the semester. Truly one of the hardest things I’ve ever had to do.

**Timeline**

- The PCB was done by November 21st

- The Arduino IDE code for control using the potentiometer was done by the end of November.

- The app was done by December 9th with some minor visual changes on the 13th.

- The Arduino code for both potentiometer control and app control was done by December 13th.

**Deployment**

If I was to put this project into production. I would add some more capacitors to my board and then do rigorous quality testing before final assembly to ensure the motor boards are reliable.

I believe this would have to become an “economies of scale” type thing in order to be profitable, because designing and manufacturing your own motherboard is not easy or cheap.

**Testing**

One of my first test cases was when I set up a simple loop to send a PWM signal to my motor to check if the board was functioning correctly, because from there it’s all software.

Another test case is when I had to use the serial monitor that I had set up to figure out why adjusting the potentiometer did not have an accurate control of the speed. To fix this I implemented the map() function to my code.

Another test case is when I connected a real phone to my computer to test if the app works instead of just emulating it in the IDE and I discovered a bug where it would connect to the board and crash immediately (fixed by removing unnecessary function from BlunoLibrary). If I hadn’t done this I would have kept working on the app until a metric ton of debugging would have been dropped on my lap.

**Changelog**

**-**  “I will also be creating my own PCB to hold my components to limit the amount of wires needed to construct this project.” - added to project details

- update Obstacles section

- update Milestones section

- added Expected Outcomes

- added Actual Outcomes

-added Architecture

-added software construction.

-added Timeline