

Progress Report: Boo-Bot

Daniel Garcia and Devin Pohl

CS 370

4/1/2021

Project Overview

The goal of this project is to create a small, remote controlled robot. This robot, nicknamed the “Boo-Bot” by the authors, is capable of sending real-time video/audio and taking real-time controls over the internet, allowing it to be controlled from anywhere with a network connection. A specific goal is to allow this bot to be controlled via Twitch live-stream chat, yielding more participation during demos.

Throughout construction, this project has been made as from-scratch as possible. This has included designing of circuitry, PCBs, and 3D printed parts, as well as bare-metal coding for drivers, communication, and interaction.

This progress report will go over work done, project direction, and next steps.

Software Development

Software work done so far can be sectioned into two areas: OS/Systemd and Twitch. While this progress report will only give a brief overview, specifics will be provided in a final report.

OS and Systemd



In order to link external hardware (camera, microphone, speaker, etc) with other software being written, we needed both an OS and drivers. We chose stock Raspbian light as a starting point, due to its exhaustive support for our Raspberry Pi 3 A+, then wrote some systemd modules to interface with our external components. The camera was dead-simple to get working great, as the `raspivid` package is built in and very fast. The speaker amplifier and microphone were a bit harder to set up. Interfacing over I2C, these parts should theoretically work with minor setup, but we ran into some UDEV issues. At this point, I2S devices are working great, with the caveat of needing to be ran with root privileges – for some reason.

Twitch Bot

With the environment set up, work could begin on interfacing with Twitch. It was decided that a Twitch bot would be written from scratch in Rust – the twitchchat crate was eventually decided on. The source code for the current iteration of the bot can be found on GitHub. After a significant amount of work was completed, the bot went online with no issues.

During development, the authors discovered Flite, an open-source offline-only terminal-based text to speech engine. Surprisingly, it could be dropped right in to the twitch bot with zero issues, taking in `!say` commands from Twitch chat and playing synthesized speech straight out of the I2S speaker.

Below are two screenshots: one of Twitch chat and the other of terminal output running on the Pi. This shows the bot joining a Twitch chat and accepting commands from another user. While the movement commands are yet to be implemented, `!say` and various information commands are complete.

```
1:23  shizcow: BooBot is online
1:23 retrodistort: !info
1:23  shizcow: Uptime: 6.99s, all users can
control, other info coming soon!
1:23 retrodistort: !forward
1:23 retrodistort: !b
1:23 retrodistort: !say this is a test
1:26 retrodistort: !quit
```

```
> cargo run -- -c src/twitch_config.toml
Finished dev [unoptimized + debuginfo] target(s) in 0.11s
Running `target/debug/twitch -c src/twitch_config.toml`
connecting, we are: shizcow
joining: shizcow
Connected!
moving: Forward
moving: Backward
saying phrase 'this is a test'
Bot exited gracefully
```

Next Steps

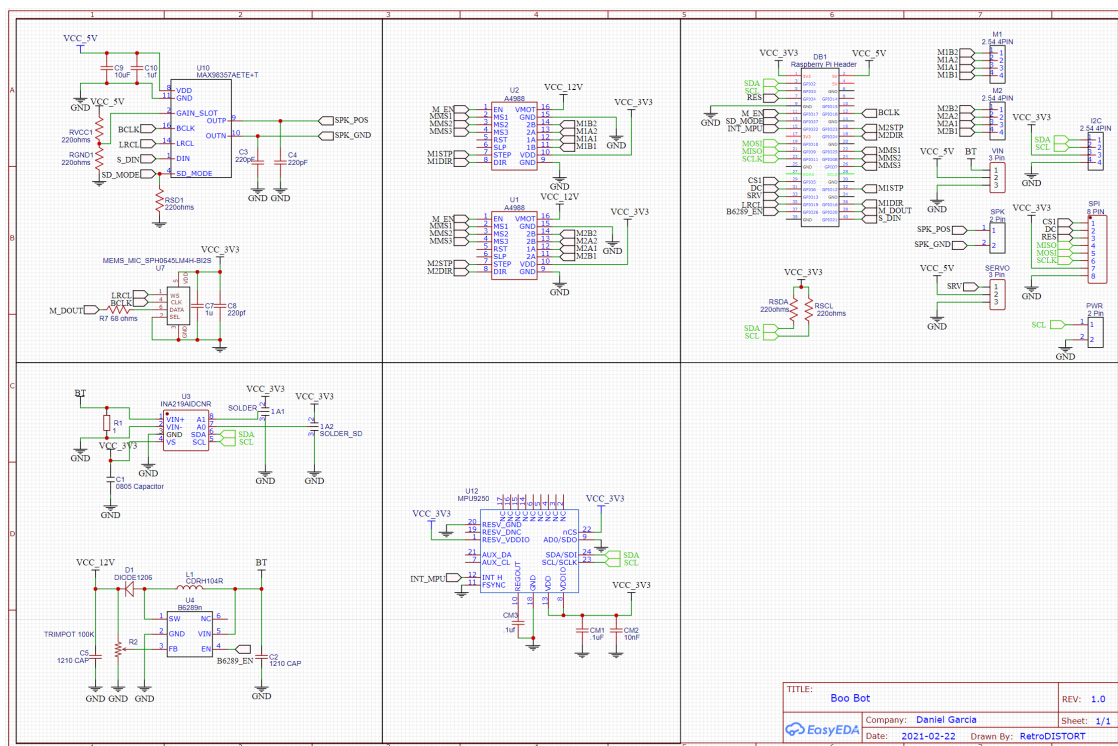
Once hardware development is complete, the movement commands can be implemented. After this, we aim to address the previously mentioned UDEV issues. At that point, auxiliary peripherals may be programmed for – as much as time may allow.

Hardware Development

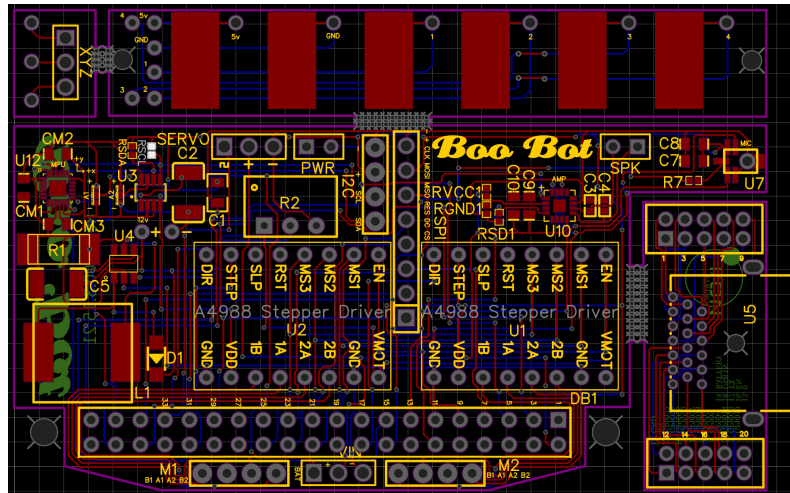
Hardware work done so far can be sectioned into two areas: Electronics/PCB and Case. While this report progress will only give a brief overview, specifics will be provided in the final report.

Electronics and PCB

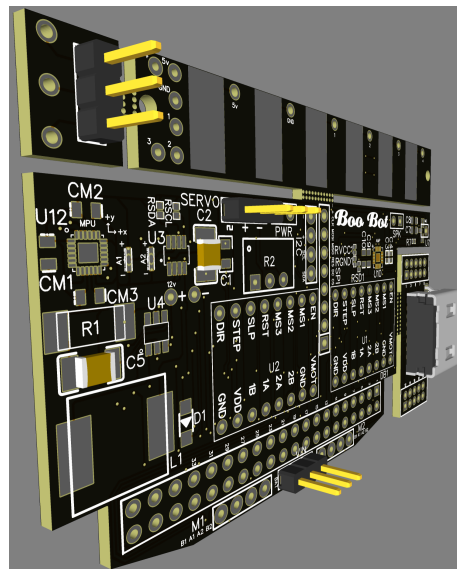
To connect all the sensors and actuators effectively, a PCB was designed to fit all the electronic components and reduce wiring. Designing a custom PCB makes the circuitry manageable, prevents wiring errors, makes the design compact and manageable. We decided to use the EasyEDA PCB design software due to previous experience with the software. Making the connections was very simple since most of the diagrams were provided in their respective datasheets. Below, a screenshot provides the connection diagram.



Soon after completing the design, the PCB was digitally wired. This process took longer than expected due to issues confirming I2S pins. After the I2S pins were confirmed, we proceeded to complete the missing connections and starting searching for design flaws. A few minor flaws were encountered and resolved. The following image is a screenshot of the final PCB layout.



Once satisfied with the design, the file was sent to a manufacturer. We decided to go for JLCPCB for manufacturing due to previous positive experiences with the company and low prices. As of right now, the PCBs have been approved, manufactured, and shipped. We are currently waiting for delivery. The following screenshot is a preview of the PCBs.



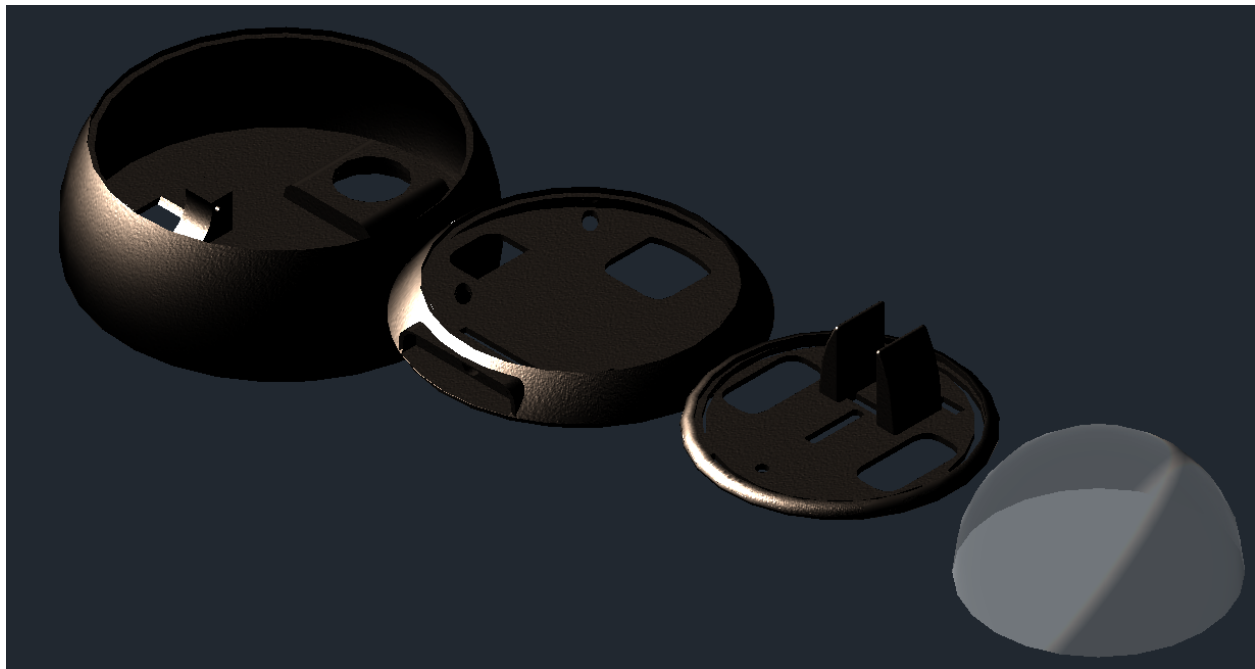
We are expecting the PCBs to arrive within the next week. Once delivered, we will proceed to solder and test the final results.

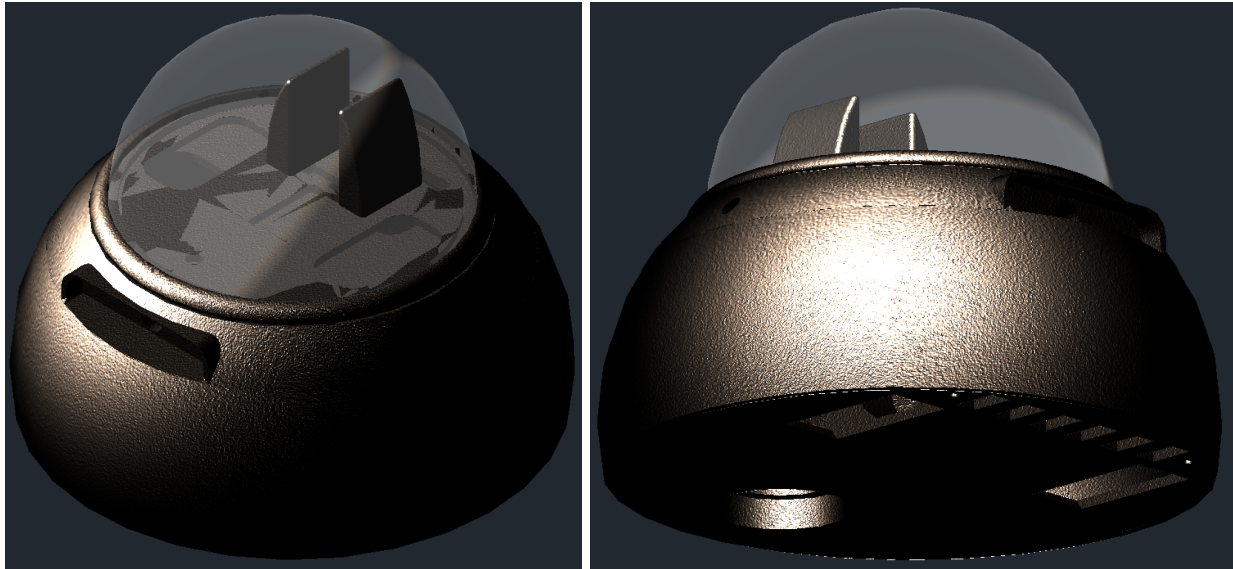
Case

The project requires an enclosure to fit all the components. Because of this, we designed a case using AutoCAD. For this, all components such as motors, batteries, and buttons were measured and replicated into this software. In the meantime, the PCB was also under design. Because of this, the case was being designed at the same time to fit new spacing needs. The robot case was designed into three levels which are further discussed below.

- Electromechanical, and power supply (Bottom layer): This level holds the stepper motors, batteries, caster, charging port, and wheels. The batteries were also placed at this level to lower its center of mass and increase stability.
- Electronics (Middle Layer): This layer holds all the fragile electronics. This layer is separated from the motors and the batteries. This was done as a protection EMF and increase the distance from the motors. The microphone and speakers are also in this layer; this also is an attempt to reduce the noise from the motors.
- Camera (Top Layer): This layer contains the camera, the display, and the servo only. This was done to avoid wire tangling while also providing a cleaner look. This layer also holds the clear plastic dome that encapsulates the device while also making it easy to view around.

The following screenshots show a preview of the design.





Once the design was done, some of the parts were printed to test the dimensions. Currently, we are still printing a few pieces to accommodate updates in the PCB and designs in issues with measurements. Overall, the current renders show a close resemblance to the final product.

Attributes to Evaluate

For this project, we have chosen to evaluate the *Limitations* and *Cost* attributes. This section will give an overview and analysis of each in turn.

Limitations

Both the current test stages and the envisioned final product of this project will have significant limitations. The first is in processing power. As this device is meant to be small, relatively low-cost, and passively cooled, the Raspberry Pi 3 A+ is one of a few great choices here. And while it is more than able to meet the base requirements (video processing, interfacing with peripherals, network communication) expandability is rather low. As such, far-future stretch goals such as fully autonomous driving, peer-to-peer swarm communication, and full server hosting may not be attainable without significant engineering.

A second limitation is in the choice of hosting. The current iteration of the project uses Twitch for controlling the bot over long distances. While this is a relatively easy solution to execute, Twitch struggles with low-latency streaming. As such, although the bot is able to push video feed with very low latency, Twitch is bottlenecking the stream delay. During testing, we measured this delay to be between 3 and 7 seconds.

Another limitation of Twitch streaming is resolution and bit-rate. While the bot can just reach 60fps 1080p streams, Twitch throttles incoming connections to specific resolutions, bitrates, and framerates. After some testing, Twitch would only display what is (after bitrate compression) approximately 720p at a locked 24fps. Future iterations of this design using custom streaming servers may be able to overcome all the limitations we have been experiencing with Twitch.

Cost

After looking at the total price, the device may seem expensive at around 110 USD but can get reduced. If decided to be done on a larger scale, the final total could be well under 70 dollars.

This lower price can be explained through the following points:

- Components: Many of these components were purchased individually. Purchasing in smaller quantities increases the price drastically. If components are purchased in higher quantities, prices can be decreased.
- Different components: Some components are also good candidates to be replaced with cheaper and higher quality ones. Because of time constraints, we used popular components. Due to the popularity these components had high demand, there were quite pricey.
- Lack of time: Because of this, some devices like the Raspberry Pi could have been integrated directly into the PCB, and thus reduce its price significantly.

Another component of the device is marketability. Due to the modular port underneath the device, it can serve a wide range of people. The modular port can be connected to other devices and increase its functionality. Some examples can be a robot sweeper, moppper, surveillance, long-distance calls, etc.

Bill of Materials

Below is a bill of materials for this project, including all hardware used. At this point, all components except the custom PCBs are in hand. We expect the PCB to arrive in 1-2 weeks.

Amount	Component	Price Ea (\$)	Description	Cost (\$)
2	Drivers	3.16	A4988 Stepper Motor Driver Module	6.32
2	Motors	3.98	MINEBEA NMB 2-phase 4-Wire 18° Stepper Motor	7.96
1	Amp	4.99	MAX98357A I2S Class D amplifier	4.99
1	Speaker	0.99	8 ohm speaker	0.99
1	Lipo	15.05	Lipo battery pack	15.05
1	Microphone	7.51	I2S MEMS Microphone SPH0645LM4H	7.51
1	Voltage Regulator	0.79	B628 3-24V to 12V 2A Adjustable Boost Step-Up Converter	0.79
10	Capacitors	0.466	16v 1000UF Electrolytic SMD	4.66
1	Raspberry pi	29.99	Raspberry Pi 3 Model A+ 2018 model	29.99
1	9-axis MPU	4.60	MPU9250 (Gyro, Accelerometer, Compass)	4.60
1	ADC	1.69	INA219 DC current and voltage sensor	1.69
1	Servo	1.79	SG90 9G Micro Servo Motor	1.79
1	Display	2.95	0.96" I2C OLED Display	2.95
1	SD Card	5.00	32 GB Class 10 Micro SD Card	5.00
1	PLA Filament	5.00	100g Black PLA filament 1.75 mm	5.00
1	PCB	8.00	5 Custom PCBs from EASY EDA	8.00
			Total:	107.29