Team 8 aka Team Old vs New PRESENTS

Snowboard Support System (aka The Triple S)

By Allen Keng, John Mamola, Bran Zhang

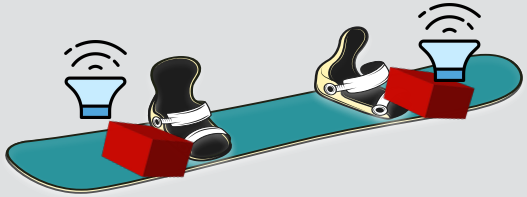
06/03/2024 ECE-196

Problem Definition

Many snowboarders, particularly beginners, encounter injuries and struggle to identify areas for improvement on their own.

Self-assessment or outside feedback can be very limited due to lack of perspective.

Proposed Idea



Device

- Rectangular boxes attached to the snowboard
- Next to their feet
- Inside will be our sensor system

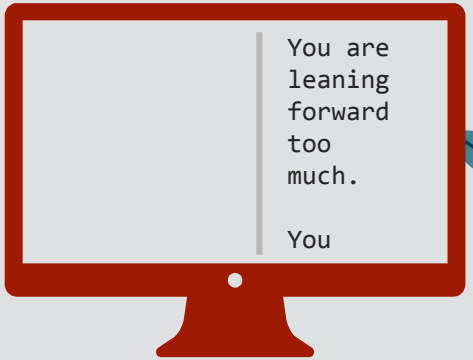
Features

Alert System:

- Buzzer that activates when detecting unstable movement

Personalized Tips:

- Shifting their weight (Balance)
- Slowing down or speeding up (Velocity)



Application

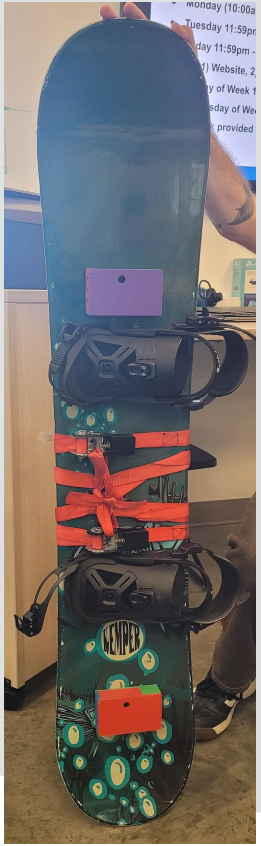
- Displays the tips to the user
- Calibration of thresholds
- Maintains list of all tips provided

Testable Hypothesis

- **The user responds to the alerts from the buzzers to carve and maintain a relatively controllable speed.**
- **Fall percentage with and without the “Triple S”.**

Milestone Achievements

Milestones	Tasks	Percent Complete	Projected/ Completion Date
(1) Planning	PCB Hub Design, Outline for Website and Poster, BLE code setup	100%	Week 6
(2) Assembly	Prototyping, Software for displaying data, Enclosure Design	100%	Week 7
(3) Testing/Printing	Using data to solve the problem. How interpret? Printing enclosure	100%	Week 8/9
(4) Debugging	Ensuring buzzers go off when they are supposed to, correct tips documented	100%	Week 9
(5) Production	Recording our video demo, adding more details/images to poster/website	100%	Week 9



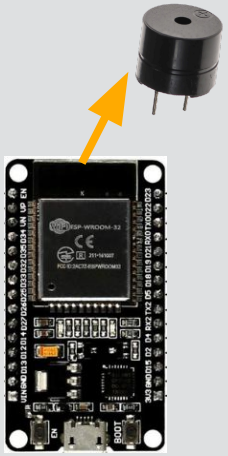
If We Had More Time

Mobile or Web App

More portable to snowboarders

The boxes would have access to the potentiometer and a switch

Redesign



Software Upgrade

Have the ESP32s activate the buzzers themselves, rather than from UI

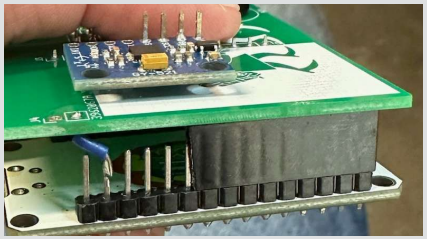
Actually use the device while snowboarding

Test it on the snow

Lessons Learned

Dimensions/Positioning

- The PCB hub we designed had a **GND pin off by 1 pin.**
- Our Enclosure was too tight and needed more tolerance

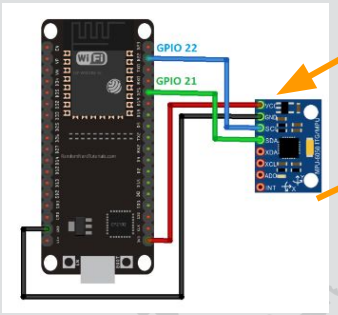


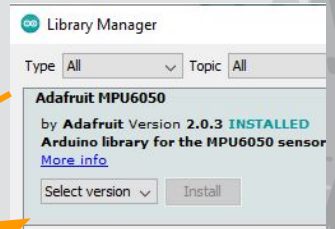
Managing Latency

- Python Bleak Connection is asynchronous which led to **delays**
  - Moved calculations to ESP32

Software Barriers

- Communication with Accelerometer module
  - **Existing MPU6050 libraries did not work**

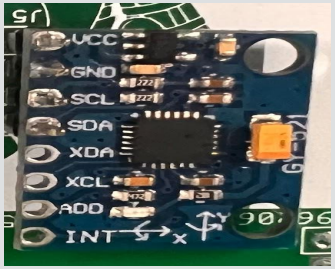




Lessons Learned

Accelerometer Sensitivity

- Matching coding with how quick and sensitive the accelerometer responds to actual movement.

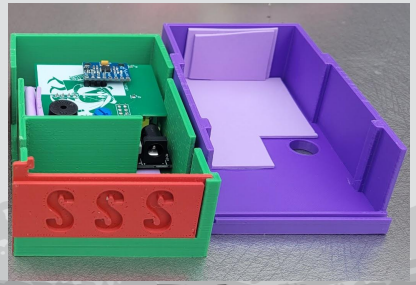


How to test without snow

- Snowboard strapped to a skateboard.

Last minute ideas

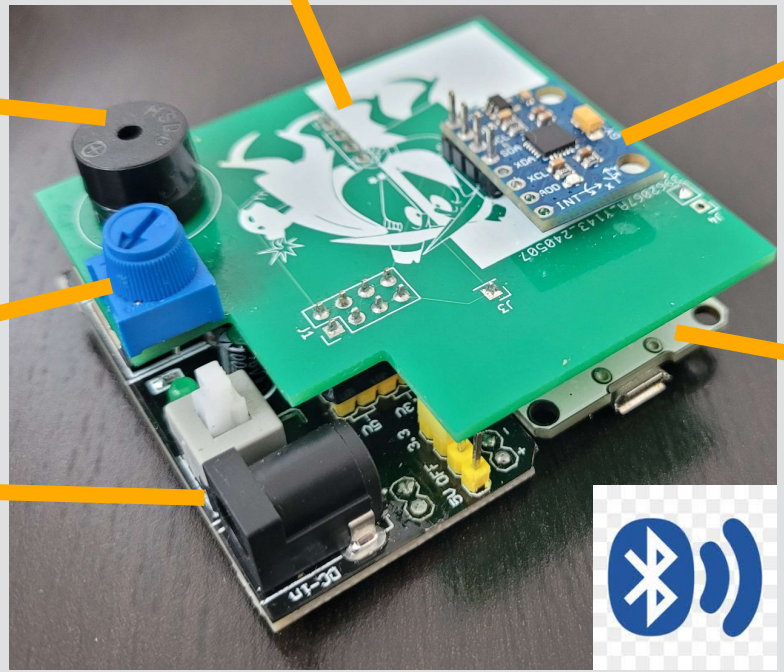
- Adding foam padding to prevent sliding of main assembly so we did not get inaccurate readings.



The Breakdown

PCB Hub

Through Holes so we can align pin slots to match the module placement.



Buzzer

Our mighty alert system

Potentiometer

Power Supply

Module that Uses 9V batteries and step downs to 3.3V

MPU-6050

I2C Communication
Read sensor readings

ESP32

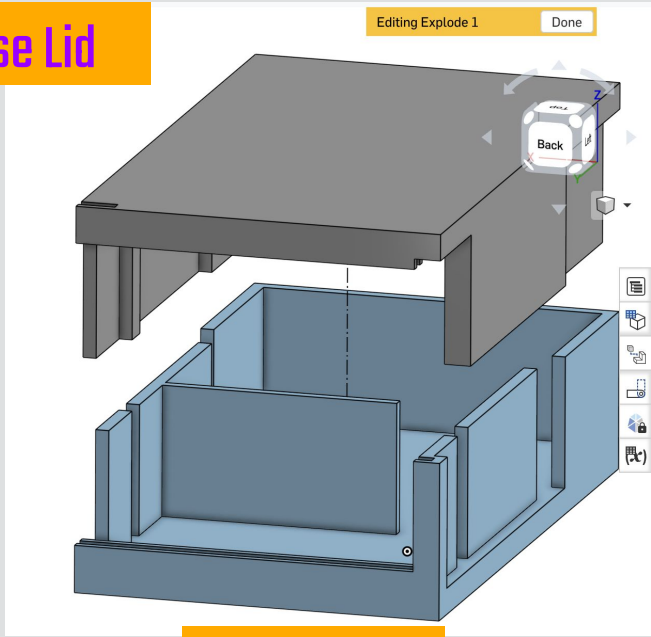
Arduino BLE
Writes out the sensor data
Receive buzzer signal

BLE Connection

Creating a service, characteristics, and then advertising the connection.

Enclosure Design

Case Lid

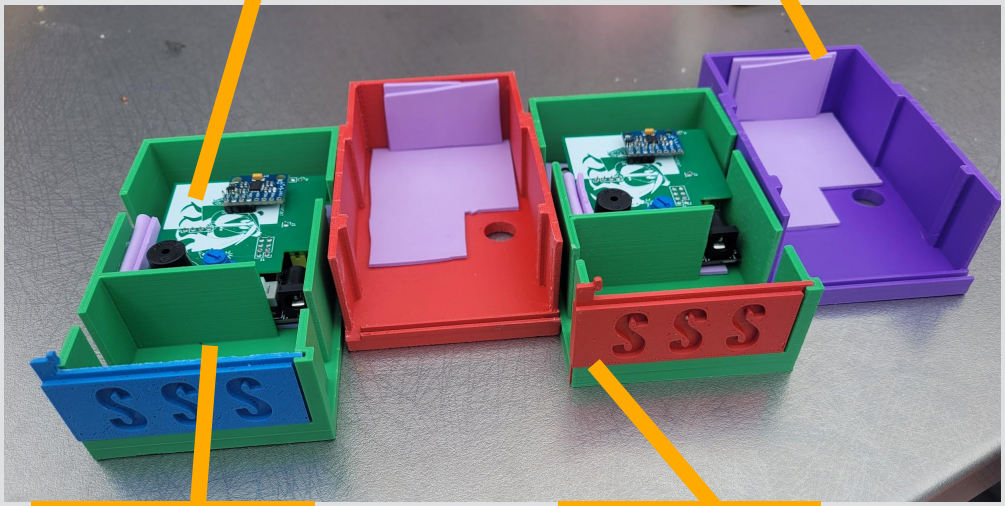


Case Base

Master Board

Foam Padding

To keep the prototype stable inside.

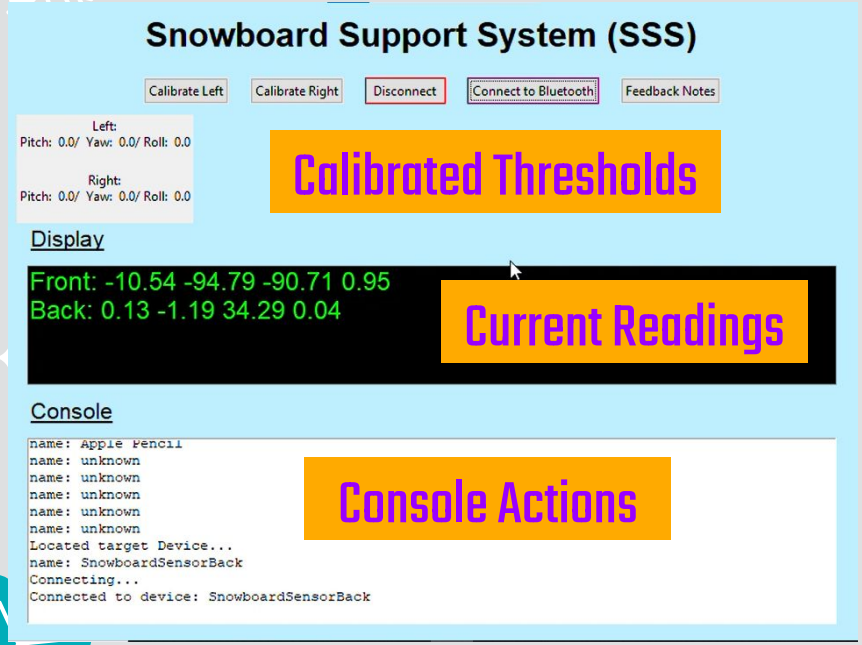


Battery Spot

Slide Door

To lock the lid and base together. Also for battery access.

Final Prototype (UI + Device Setup)



Python + Bleak

- **Asynchronous connection to BLE**
  - Connect/Disconnect
- **Logic to Read Sensor Bytes**
- **UI and Calibration**

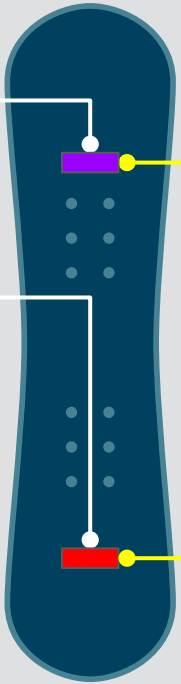
Tutorial:

Front SSS

Alerts the snowboarder to transition carving

Back SSS

Alerts the snowboard to excessive speed



1. Install Batteries

Verify power up beep

2. Connect BT

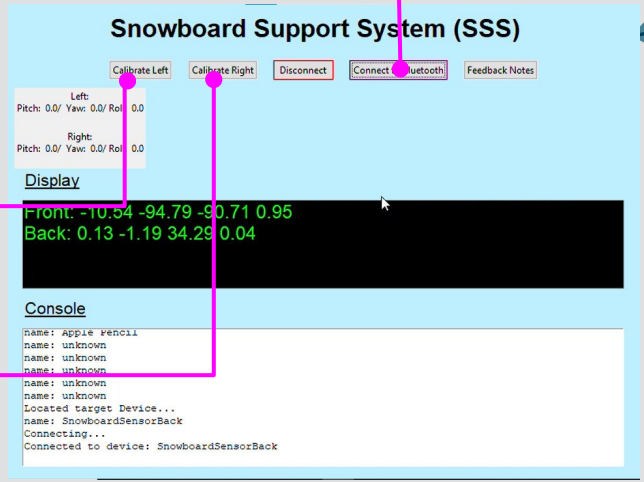
3. Calibrate Left

Lean Goofy(toes) Reg(heels)

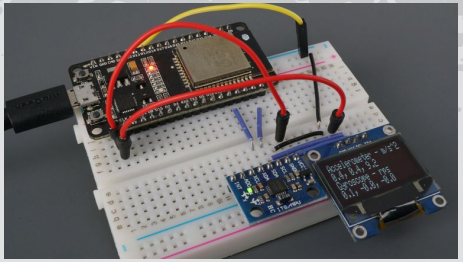
4. Calibrate Right

Lean Goofy(heels) Reg(toes)

5. Go RIDE!!!



Resources/References



Connections

- ESP32 with MPU-6050
  - Figuring out the pins we were going to connect to while designing PCB
- https://bleak.readthedocs.io/en/latest/api/client.html
  - Making python act as a BLE receiver.

```
async def main():
    devices = await BleakScanner.discover()
    for device in devices:
        print(device)
```

I2C Communication

- ESP32 I2C Communication
  - Understanding how to read values from an I2C device via “wire” transmission

Velocity Calculations

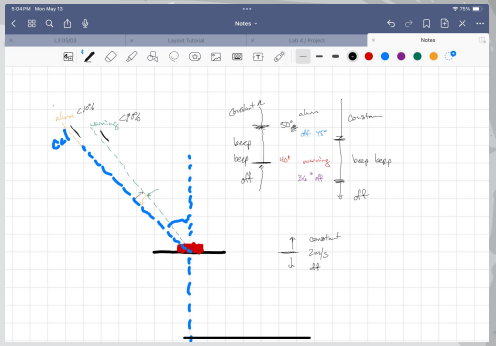
- Can I measure velocity from an accelerometer? How accurately?

$$V_f = V_i + at$$

Snowboarding Knowledge

- How to Snowboard | REI Expert Advice | REI

Thanks!



- What challenges do you think could occur if trying to implement this system on a broader scale?
- Are there any better ways to calibrate the accelerometer?
- If you have never snowboarded before would you be willing to try?

Do you have any questions?