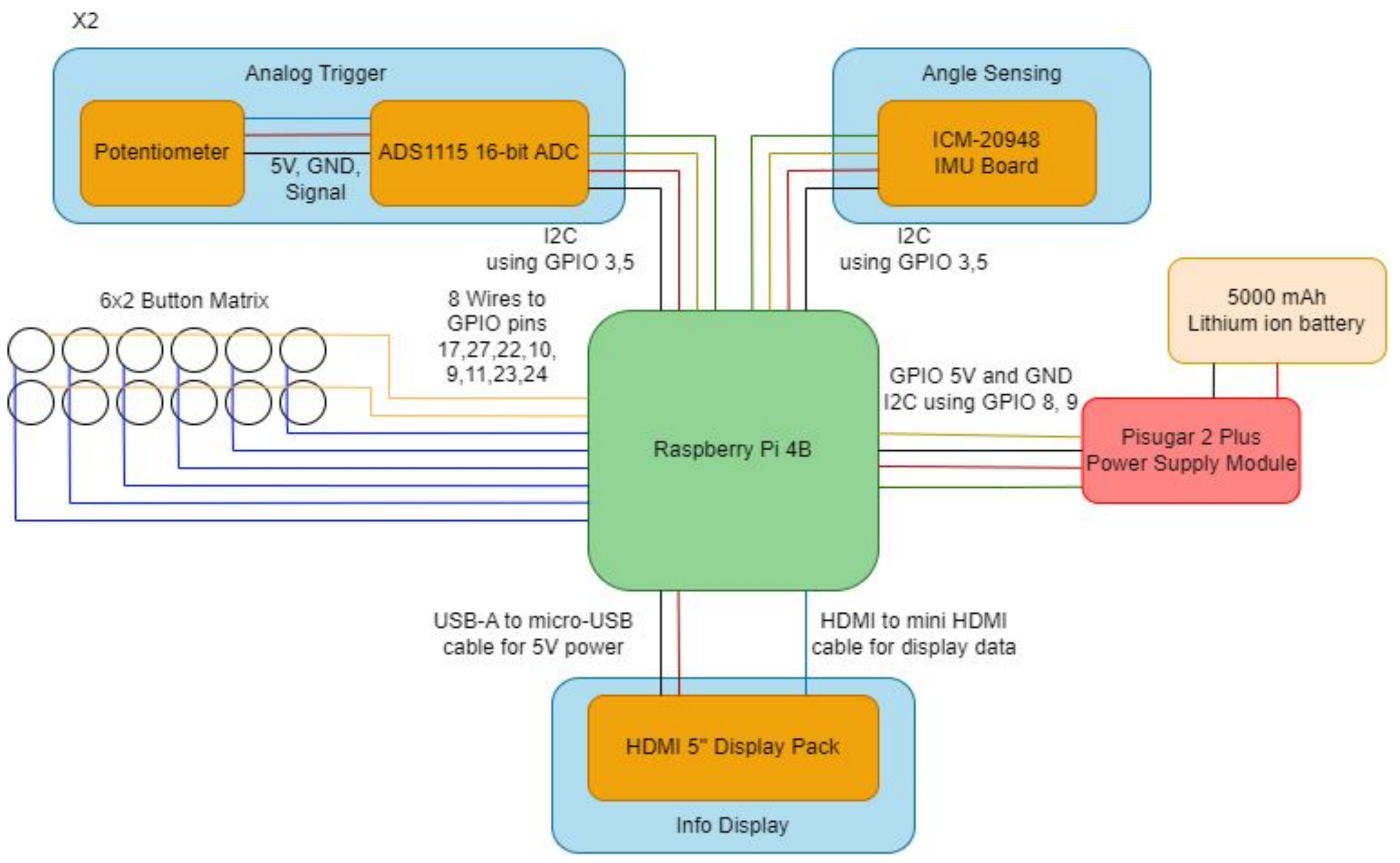
Mobile Steering Wheel

Team B3: Qiaoan Shen, Yuxuan Zhu, Xiao Jin
18-500 Capstone Design, Spring 2023
Electrical and Computer Engineering Department
Carnegie Mellon University

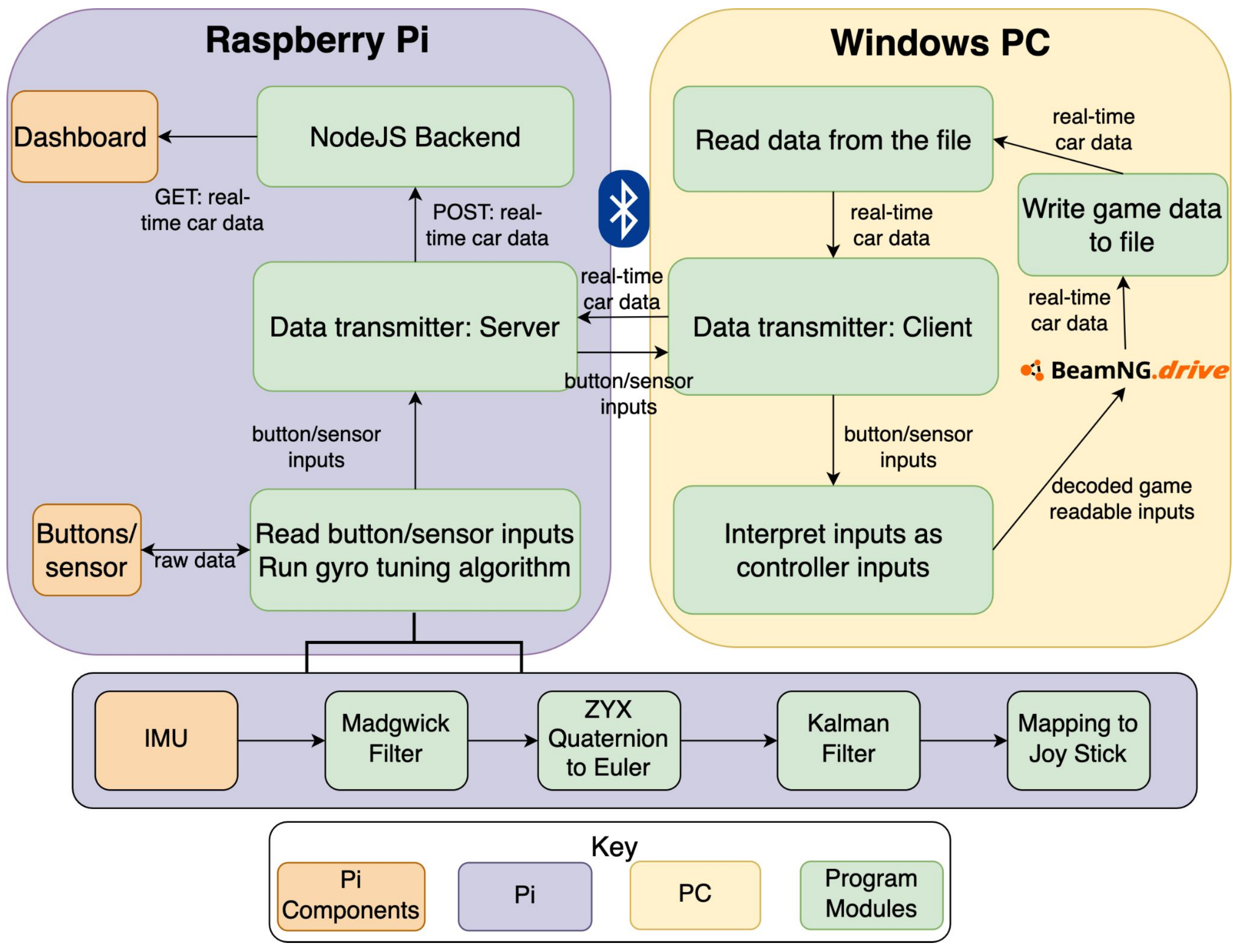
Product Pitch

Do you play racing games (RAC) to feel the thrill of **Fast** and **Furious**? Do you envy the professional-grade racing simulators? Come try out our **Mobile Steering Wheel**! Instead of pushing buttons and sticks, you turn the car by turning our controller! It is **intuitive** and **precise**, providing the ultimate level of **immersion**, as well as every edge in competitive scenarios.

System Architecture


X2
Analog Trigger
Potentiometer
5V, GND, Signal
ADS1115 16-bit ADC
Angle Sensing
ICM-20948 IMU Board
I2C using GPIO 3,5
I2C using GPIO 3,5
6x2 Button Matrix
8 Wires to GPIO pins 17,27,22,10, 9,11,23,24
Raspberry Pi 4B
5000 mAh Lithium ion battery
GPIO 5V and GND I2C using GPIO 8, 9
Pisugar 2 Plus Power Supply Module
USB-A to micro-USB cable for 5V power
HDMI to mini HDMI cable for display data
HDMI 5" Display Pack
Info Display

The hardware system (picture above) consists of all off the shelf components, with custom wiring.



Controller data is transmitted through Bluetooth to PC to control the game. Game data is sent back from PC to Pi to display car information on dashboard.

Conclusions & Additional Information

The final product is more complete than we first envisioned. Most of the tech were new to us, but through learning we got the hang of them. Integration is definitely the most difficult part, and we should always leave ample time for integration and testing in future projects.

We have two main plans for the future. As we realize a smartphone has all hardware components we have, we can possibly incorporate it into our design. We also want to redesign the shell of the steering wheel to be more ergonomic.

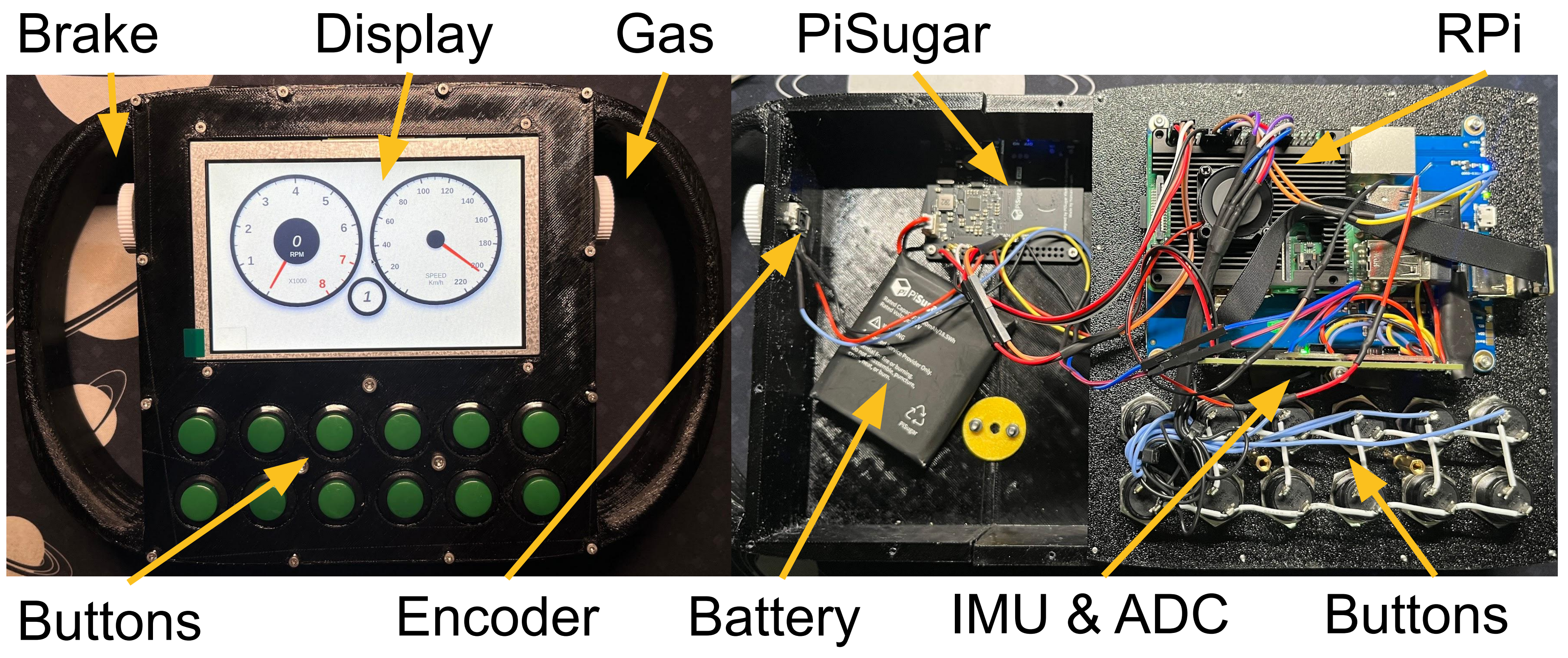
System Description

Gyroscope
- Use Madgwick filter to calculate orientation from IMU data
- Use ZYX sequence to transform from quaternion to euler to mitigate gimbal lock
- Use Kalman filter to reduce noise and achieve smooth control

Software
- Bluetooth communication is achieved using RFCOMM protocol with Socket library through JSON packets
- A NodeJS backend is implemented for Python program to post car data and for frontend to fetch the data concurrently

Hardware
- Button Matrix to achieve efficient wiring
- Encoder with ADC for analog input
- HDMI Display as game dashboard
- PiSugar HAT for power management

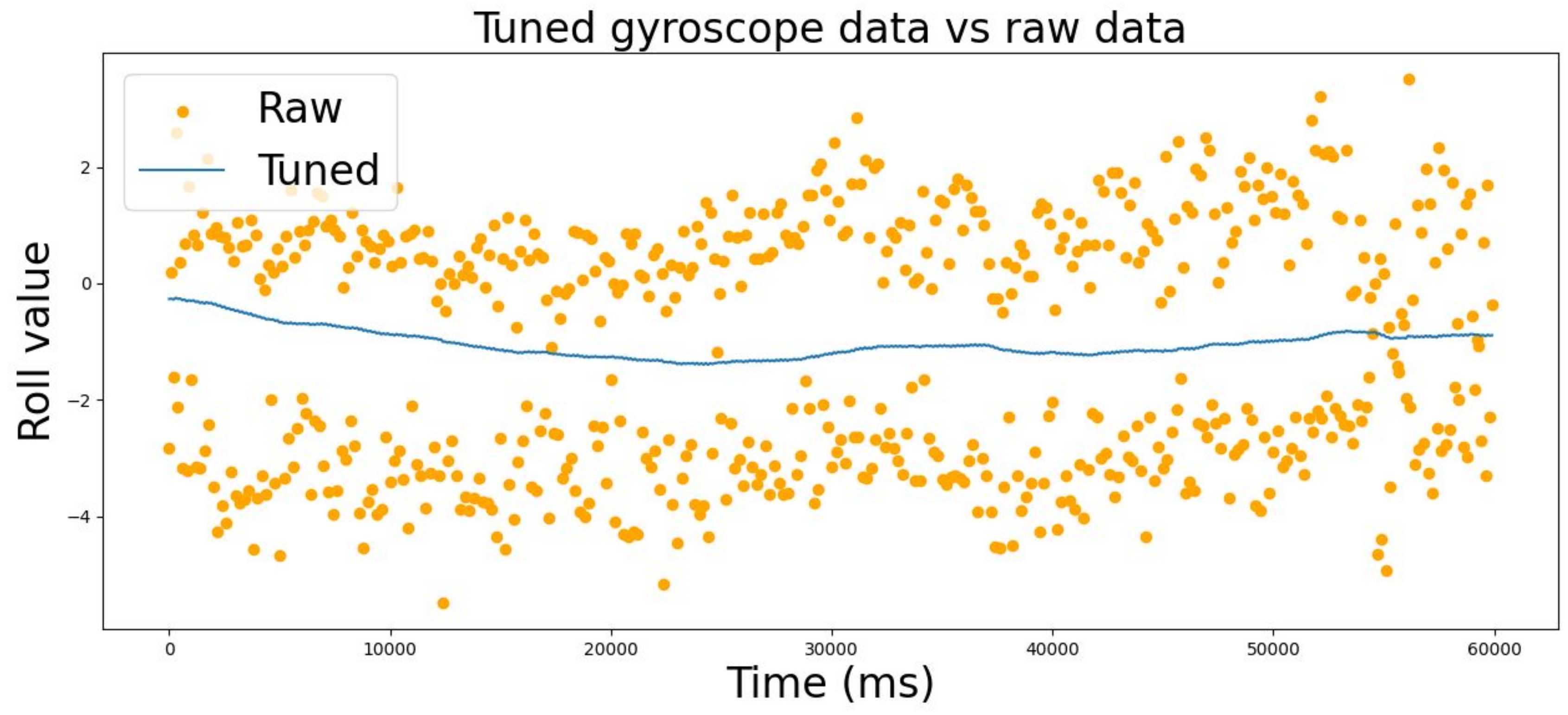
Controller Overview (Left) & Component View (Right)



System Evaluation

Metrics	Testing Approach	Result
Weight	Put controller on a scale	660g > 400g (goal).
Gyroscope	Place controller on a flat surface. Read raw Roll angle and the converted analog axis value	Drifting within 0.8 degree after 60 seconds (goal: 1 degree). Oscillating within range of +/-0.5 degree (goal: 0.5 degree).
Battery Life	Measure the RPi's power consumption and calculate	4.6h > 4h (goal). 18.5Wh (battery capacity)/4W (power) = 4.6h.
Input Latency	Use time.time() to measure avg round trip time/2	Avg round trip time ~= 33 ms Input latency <= 20ms (goal).

Tuned gyroscope data vs raw data


Raw
Tuned
Roll value
Time (ms)