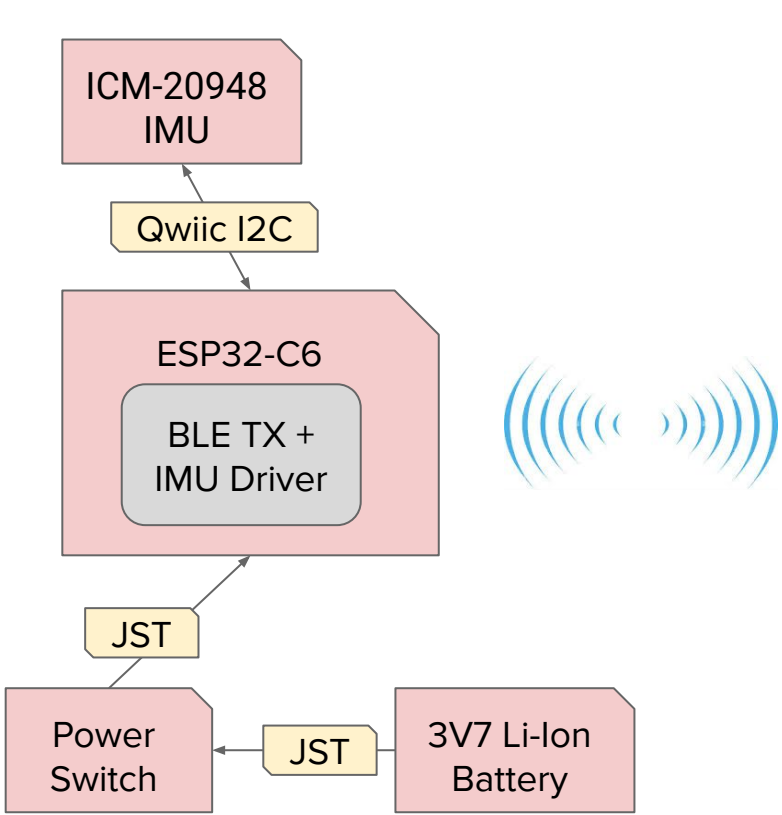
Not Wii Sports

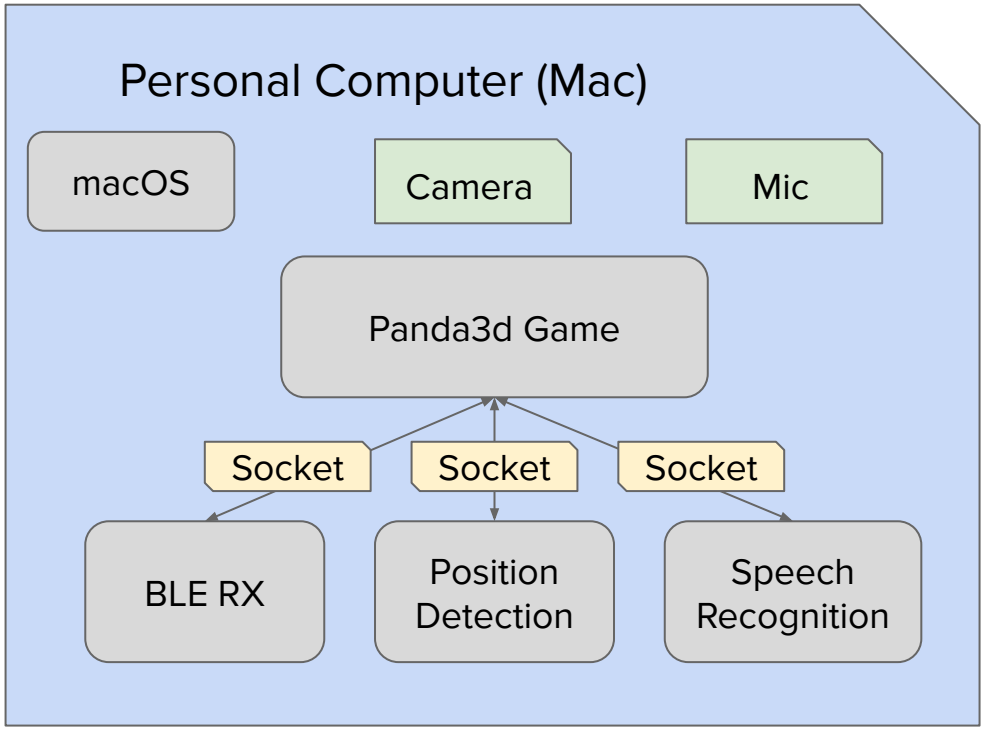
Team 7
Brandon Sutton, Kimon Anagnostopoulos, Anthony Guerrera, Benjamin Delgado

Motivation

- Mobile games are leading the technology & entertainment sectors
- Teenagers & college students are constantly on the lookout for new fun ways to play games.
- Most people are afraid of commitment to more expensive gaming consoles such as Nintendo Switch, Xbox, or Playstation
- 2 player games such as bowling are incredibly popular and has a simple user interface and fun game mechanics.

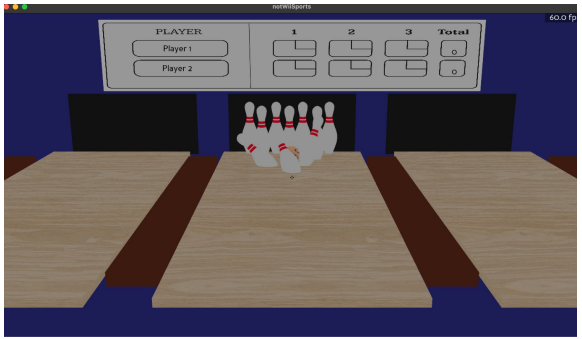
High Level Design





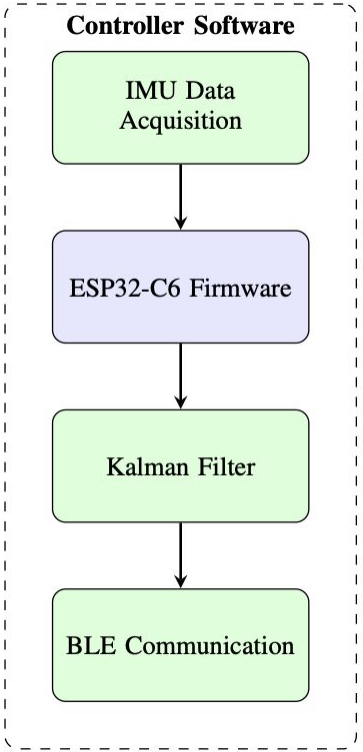
Game Implementation

- Panda 3D event loop listens for events via multiple processes communicating to threads on the main event loop via sockets. These include Bluetooth Low Energy Data & OpenCV Data
- Feeds data into main driver for app that includes classes that maintain pin state and player score across multiple different rounds
- Scoreboard running total stored as well as on a per round basis.
- Model & Collisions are handled via custom collision algorithms based on vector calculus.
- UI updates are performed in real time



Remote Control Software and Bluetooth

- ESP32-C6:
  - Data acquisition, filtering, and bluetooth communication
- Kalman Filtering:
  - Smooths out data from accelerometer and gyroscope readings
- BLE Server:
  - Peripheral: Defines custom BLE services for motion data
  - Host: Struct unpacking and processing for filtered data
- Error Handling and Disconnection:
  - Auto-reconnect if BLE connection drops
  - Use of buffered data to tackle motion lag

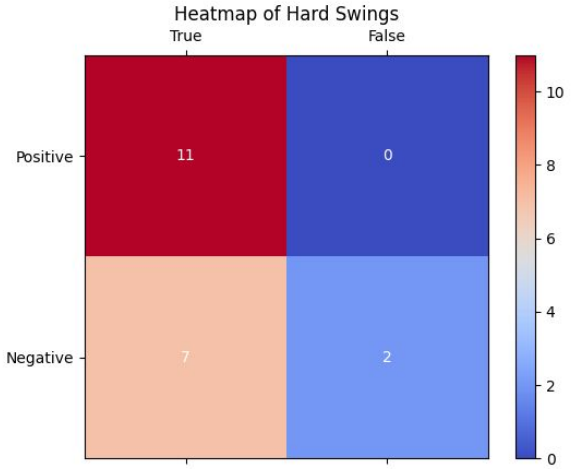


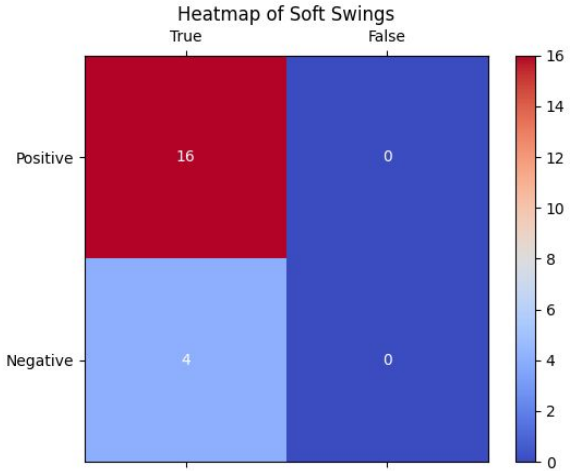
Motion Detection

1. Collect current value of angular velocity about y-axis
2. Circular buffer to track historical values (key value: size)
3. Count consecutive values above threshold (key values: threshold, consecutive count)
4. “Stroke” power = average of consecutive values
5. Exponential scaling (key value: exponent)
6. Map to “roll time”

Testing Motion Detection

- Goal: Evaluate “hard” vs “soft” swings
  - Detection = Has swing occured
  - Classification = Does ball speed match swing power?
- Result: Successful detection and classification in 67.5% of trials



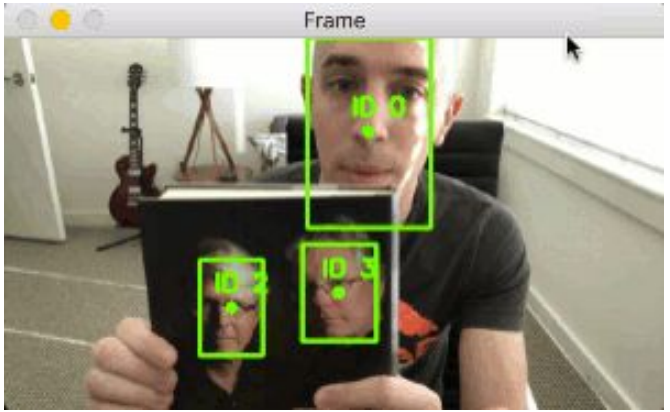


Speech Recognition

- PyAudio library
- Voice commands for player name recognition
- Detected names displayed on the game scene

Computer Vision

- Use OpenCV to track player movement to move the ball
- Takes in information from head / body movement to dictate control of ball on screen
- Ensure that data received from OpenCV is not noisy and scaled down considerably in order to receive smooth real time data



Deployment

- Initial Attempts:
  - Raspberry Pi 4: Too slow
  - Docker: GUI could not run in docker
- Final Solution: Scripts Downloadable from Github
- Setup Script
  - Ensure using macOS
  - Download Python 3.8.10
  - Download dependencies
  - Set up environment
- Play Script
  - Run game from any directory!

Testing and User Experience

Test Area	Purpose	Key metrics
Motion Tracking and Kalman Filtering Optimization	Optimize IMU readings and reduce sensor noise	Accelerometer drift, filter response to varying speeds
BLE Performance Testing	Evaluate BLE latency, reliability, and throughput	Latency measurement, packet loss rate, RSSI values
Motion Detection Testing	Assess speed detection accuracy and speed classification	True/False Positive rates, precision, and recall
Usability Testing	Evaluate setup ease, responsiveness, and user experience	Setup time, motion tracking accuracy, GUI usability

Timeline - Goal Overview

Weeks 1-5 (Q1)	Initial development and design
Week 6-8 (Q1)	Individual component development (voice control, 3D scene, motion tracking etc.)
Week 8-10 (Q1)	Testing, full integration, and skeleton version of the game
Week 1-6 (Q2)	Refactored hardware setup, UI updates, advanced filtering and mechanics
Week 6-10 (Q2)	Component validation, motion tracking improvements, and mechanics simulations

Project Contributions

Component	Details	Contributors
BLE Communication and Data Handling	BLE server setup, reliability and connectivity testing	Brandon, Kimon
IMU Filtering & Motion Tracking	Implemented Kalman Filtering for IMU readings, optimized sensor fusion, ball speed modulation	Brandon, Kimon
Hardware Configuration	Set up ESP32-C6 and IMU, integrated with BLE and game software.	Brandon
Game Engine and Mechanics	Built game engine in Panda3D, developed physics, UI, and scoring system.	Anthony
Peripheral Assistance	Usability Testing, Quality Assurance,, Remote Case	Ben

Limitations & Future Prospects

- What We Think Works
  - Affordable and Accessible
  - Easy to set up & use
- Future Development and Potential
  - More Advanced Sensor Fusion
  - Online Play
  - More Accurate Gesture Recognition
  - Cool Animations for Strikes/Spares

Live Demo - Backup Slide