Augmented Instrumental Reality

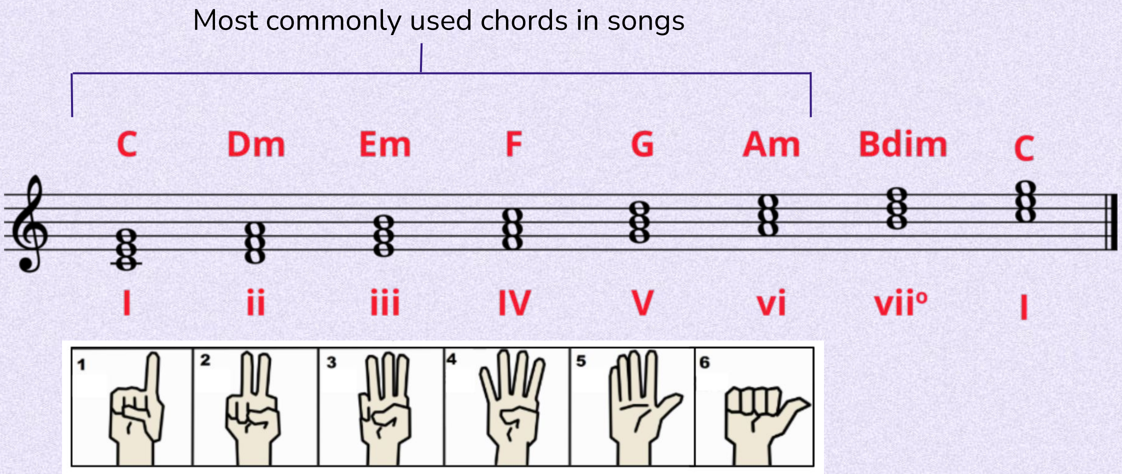
Team 3 - Taj Abdin, Alexa Lowe, Lucy Wang

Use-case and Design Requirements

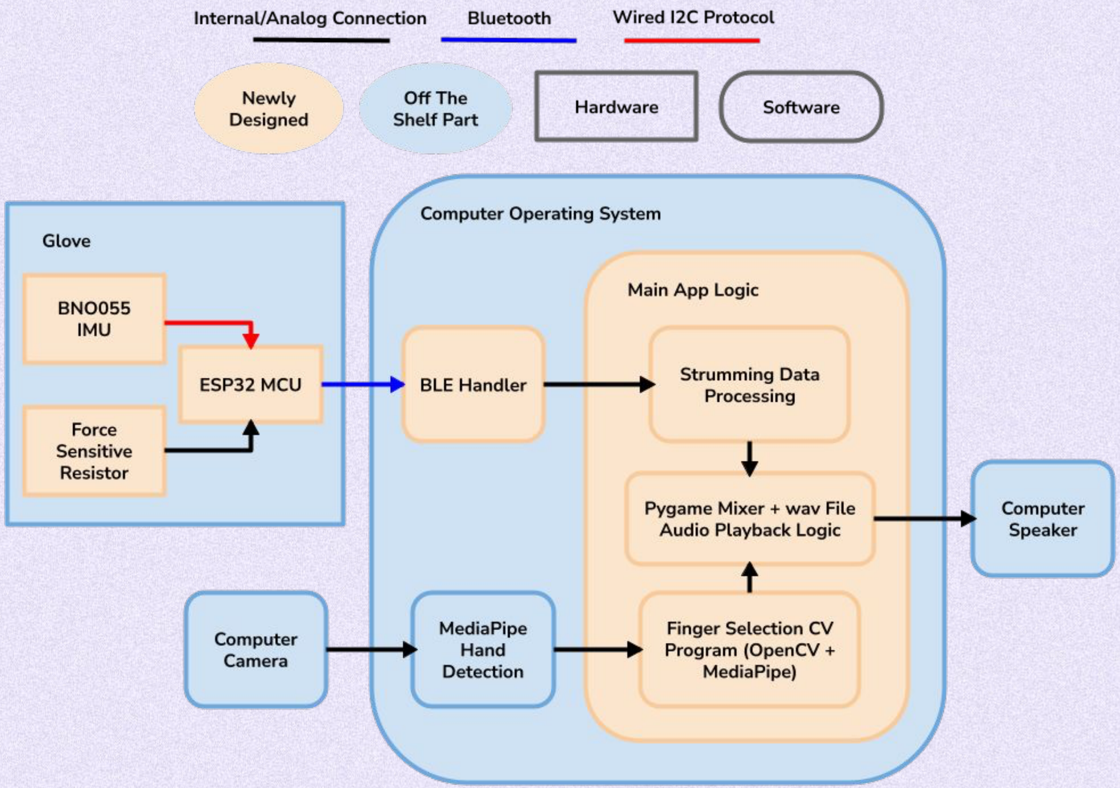
Use Case Requirements:	*Design Requirements:*
Overall user latency below 100 milliseconds	Latencies: Computer vision and analysis ≤ 45 ms, Bluetooth signal ≤ 35 ms, Audio rendering ≤ 20 ms
95 percent accuracy for both chord selections and strumming motions	Left hand detection, individual landmark detection, and individual finger detection IMU Strum Detection: Angular velocity and acceleration based algorithm
Accessible, lightweight hardware weighing less than 70 grams in total and requiring < 2N of force for reliable activation	Battery: 32 g Breadboard: 13 g Others (e.g. tape, glove, wires): 10 g
Detects strumming rates ranging from 60 to 280 beats per minute (around 1-5 strums per second) and different strumming speeds	200 ms refractory period between strums with accuracy (previously mentioned). Can detect between 0.2m/s - 1m/ms speed
Works with 2 meters of distance, easy setup and up to 2 hours of use	Complete Bluetooth pairing within 25 seconds ~ 15 ms message transmission latency. Power draw of < 0.9 W with battery of capacity > 600 mAh

Use-case and Design Requirements

Use Case Requirements:	Design Requirements:
Supports 6 different chords within a scale to allow easy playing of chord progressions, and within modes such as piano, guitar, and voice sounds	Switch hand gestures with ≤ 20 ms Automatically adjust landmarks based on preset patterns Delay between notes based on velocity

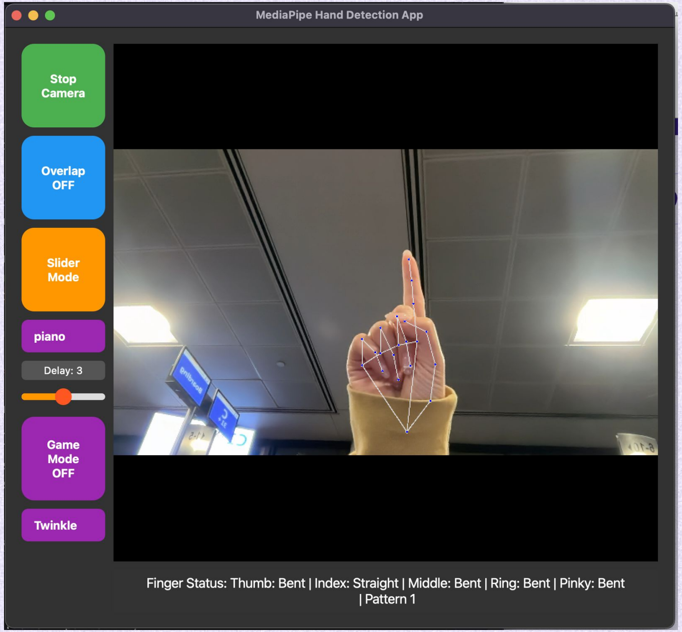


Complete Solution

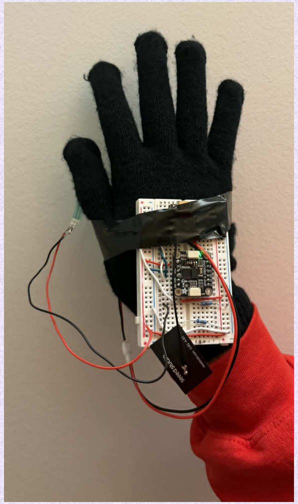


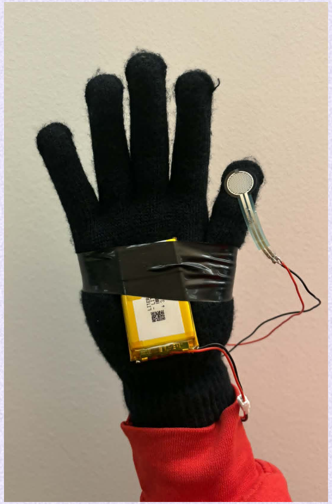
Solution Approach & MVP Review

Left Hand User Interface



Front/Back of Right Hand Glove





Complete Solution/DEMO

Left Hand Chord Selection

- Detect up to 6 customizable chords based on finger bends
- Target ≥ 95% accuracy within 0.3-2 m camera range

Right Hand Strumming (IMU Wristband + Force-Sensitive Resistor)

- IMU measures strumming motion speed and direction
- Force-Sensitive Resistor register strum initiation events
- Target strumming speed range: 60-280 bpm (around 1-5 strums / sec)

Verification - Design Requirements

Design Requirements:	Test:
Latencies: Computer vision and analysis ≤ 45 ms, Bluetooth signal transmission ≤ 35 ms, Audio rendering ≤ 20 ms	Time latency for each component
Left hand detection, individual landmark detection, and individual finger detection; IMU Strum Detection: Angular velocity and acceleration based algorithm, sampled every 10 ms	N/A – (set all parameters as requirements)
Battery: 32 g; Breadboard: 13 g; Others (e.g. tape, glove, wire): 10 g	Weigh all components separately on a scale
200 ms refractory period between strums with accuracy (previously mentioned). Can detect between 0.2m/s - 1m/ms speed	N/A – (set refractory period as requirements) Graph program delay given different hand speeds
Complete Bluetooth pairing within 25 seconds ~ 15 ms message transmission latency. Power draw of < 0.9 W with battery of capacity > 600 mAh	Time latency of initial bluetooth pairing Time latency between strum and transmission
Switch hand gestures with ≤ 20 ms Automatically adjust landmarks based on preset patterns Delay between notes based on velocity	Measure time between hand gestures Check in code whether the landmarks are changing Graph correlation between velocity recorded and delay time

Result - Verification

Latency Breakdown:
- Computer vision: **3.572 ms**
- Bluetooth signaling: **25.39 ms**
- Audio rendering: **79.19 ms**

Weight for Components:
- Battery: **32 g**
- Breadboard: **13 g**
- Others (e.g. tape, glove, wires): **20 g**
- Activation force: **0.47 N**

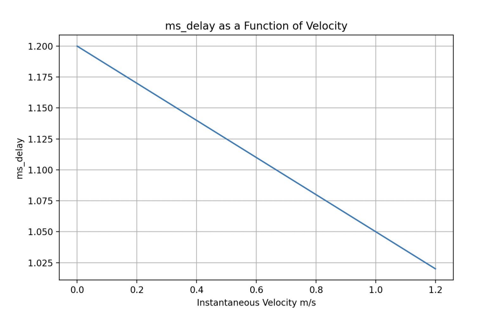
Time between Hand Gestures: **11.9 ms**

Landmarks Adjusting? YES

Strumming
- Angular velocity: **>1.2** rad/s, and resets when **<0.8** rad/s
- Acceleration: acceleration spike of **5 m/s^2** above baseline

Latency for (taken between 50 samples):
- Initial Bluetooth pairing: **7.763 s**
- Between strum and transmission: **25.386 ms**

Velocity vs. Delay:

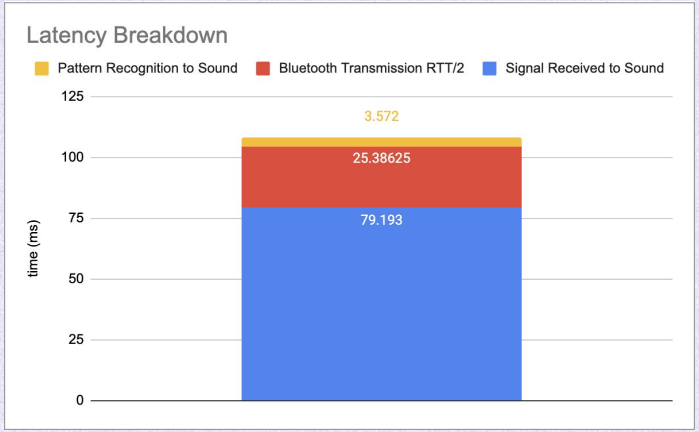


Validation - Use-case Requirements

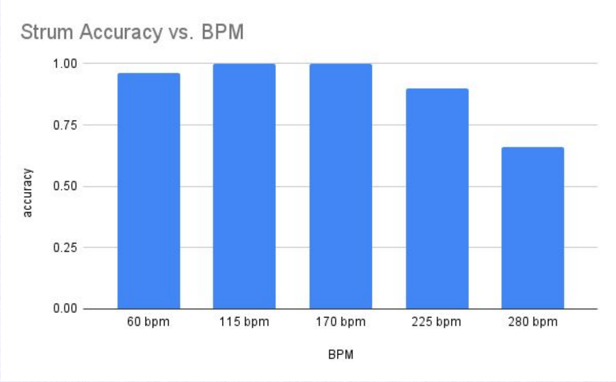
Use Case Requirements:	Test:
Overall user latency below 100 milliseconds	Add up individual segment latencies to obtain times for each segment of the system
95 percent accuracy for both chord selections and strumming motions	Left hand detection and strumming design tested with user studies from musicians in the school of music
Accessible, lightweight hardware less than 70 grams in total; requires < 2N of force for reliable activation	Measure weight of components and force required for activation
Detects strumming rates ranging from 60 to 280 beats per minute (around 1-5 strums per second) and different strumming speeds	Test efficacy of different strumming rates with experiments of consistent bpm strumming across our desired range
Works with 2 meters of distance and up to 2 hours	Startup and robustness tests with different distances and orientations
Supports 6 different chords within a scale for easy playing of chord progressions, and within modes such as piano, guitar, and voice sounds	Ensure correct chord selections for the 6 chords and accurate chord play for different timbres

Result - Validation

Total Latency: 108.15 ms



Strum Accuracy Tests:



100 samples

latency issues with 5 strums/sec

Overall Weight: 65 g (Meets goal of 70 g)

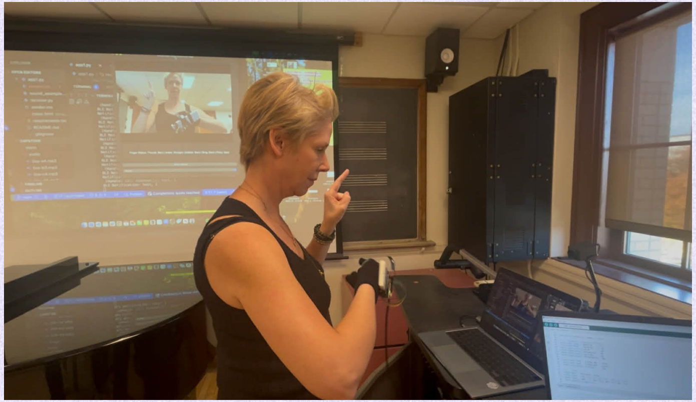
Accuracy up to 2 meters:

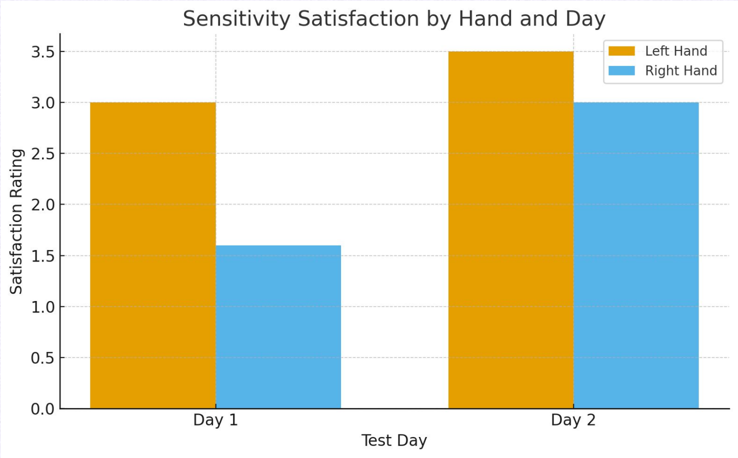
100 %* from ~95% accuracy of MediaPipe landmark tracking

Result - Validation

User Studies for Chord Selection, Strumming, and Game

- As part of the Engineering Creative Interaction: Music + AI course by Prof. Jocelyn Dueck of the School of music, discussed design choices and participated in user studies
- Iterated across 3 class periods using feedback from user study forms





Trade-Offs

- **C++ vs. Python**
  - 2-5x faster, harder deployment vs. fast prototyping, ML integration
- **MediaPipe vs. OpenPOSE**
  - lightweight (~50-100 MB), hands-only vs. full body tracking, GPU-intensive
- **IMU vs. Ball Tilt**
  - 5× user preference (100% vs 20%), continuous 6–9 DOF motion data at 100–200 Hz vs. simple binary switch with no angle or rate information
- **BNO055 vs. MPU6050**
  - orientation built-in, easier integration vs. raw data
    - No built-in magnetometer + fusion in the chip of MPU6050
- **STM32 vs. RPi vs. ESP32**
  - low power, slow development vs. powerful but 40+ g vs. lightweight, low power, fast development
- **Bluetooth vs. ESP-NOW vs. Wi-Fi UDP**
  - lowest power, good latency vs. lowest latency, device-to-device vs. high throughput, high power, calibration issues when switching Wi-Fi

Project Management