Assignment 2D: Project Reflection

Xinyi Han, Student ID: 25751470, Xinyi.Han@student.uts.edu.au
32027 31080 Interactive Media, Interactive Media — Spring 2025

Project Overview

Assignment 2 created an **Interactive Hand-Tracking Virtual Drum Kit** that transforms hand gestures into music and visual art. Using MediaPipe computer vision and `p5.js`, the project tracks hand movements through a webcam, allowing users to play four virtual drums (kick, snare, tom, hi-hat) by moving their index fingers through drum circles with sufficient velocity.

The most novel aspects combine gesture-controlled music creation with real-time shader-based video processing and audio-reactive particle effects. The webcam feed transforms into living artwork through GLSL shaders that apply pixelation, color posterization, and dynamic palette remapping that progresses from cool teal tones to warm magenta–orange hues based on audio energy. This fusion of computer vision, audio engineering, and graphics programming creates an accessible instrument requiring no physical equipment—just hands, a webcam, and creativity.

Feature 1: Hand Tracking and Hit Detection System

Recount

The hand tracking and hit detection system formed the foundation of the entire drumming experience. I integrated the **MediaPipe Hands** library to track up to two hands in real time, extracting 21 landmark points per hand representing finger joints and palm positions. The critical design decision involved determining when a gesture should trigger a drum sound. Rather than simple position-based collision detection, I implemented **velocity-sensitive triggering** that required the index fingertip (landmark 8) to both enter a drum circle's radius and exceed a minimum speed threshold of 500 pixels per second.

The implementation required careful modular organization. I created separate files for distinct responsibilities: `tracker.js` initialized MediaPipe and managed landmark data, `hitmap.js` provided geometric utilities for distance calculations and circular collision detection, and the main `sketch.js` calculated frame-to-frame velocity vectors and maintained hit cooldown timers to prevent double-triggering. Each drum tracked its previous frame's fingertip position, enabling velocity calculation through position delta divided by frame time. When a hit registered, the system captured the impact velocity to scale visual feedback intensity—faster hits produced more particles and stronger visual responses.

React

I felt genuinely proud and satisfied with how this system turned out. The drumming interaction feels natural and responsive in a way that exceeded my expectations. When testing the final version, there's a satisfying tactile quality to the virtual drumming—the velocity threshold prevents accidental triggers while still allowing expressive dynamics. Watching others try the instrument and immediately start creating rhythms without instruction validated that the interaction design succeeded in being intuitive. The pride comes not just from technical achievement but from creating something that feels good to use.

Analyse

The success of this feature stemmed from two key approaches. First, breaking the complex problem into smaller, manageable modules allowed me to tackle hand tracking, geometric calculations, velocity detection, and visual feedback as separate concerns that cleanly interfaced together. This modular structure made debugging straightforward—when hits weren't registering properly, I could isolate whether the issue lay in landmark detection, distance calculation, or velocity thresholds.

Second, iterative refinement through extensive testing proved essential. My initial velocity threshold was too high (800 px/s), making drumming exhausting and unresponsive. Through repeated testing, I discovered 500 px/s provided the sweet spot where deliberate hits registered reliably while casual movements didn't trigger false positives. I also experimented with different landmark points before settling on the index fingertip, finding it provided the most intuitive "drumstick" metaphor. This iterative testing cycle transformed the system from technically functional to genuinely playable.

Improve

For future projects, I would maintain this modular approach and commitment to iterative testing, as they proved invaluable. However, I would enhance the system with user-facing improvements. Adding calibration options would allow users to adjust velocity sensitivity for their specific setup, camera angle, and playing style. I would also explore expanding beyond fingertip position to recognize different hand poses—perhaps closed fist for muted hits versus open palm for accents. Finally, better documentation of optimal threshold values and design rationale would help me (and others) reuse this knowledge for future gesture-based interfaces.

Feature 2: Particle System with Multiple Effect Modes

Recount

The particle system provides visual feedback for drum hits through colorful burst effects. I implemented five distinct effect modes: *burst* (radial explosion), *spray* (directional cone), *cluster* (dense slow-moving clouds), *ring* (expanding wave), and *random* (automatic cycling). Each particle has properties including position, velocity, lifespan, color (from a six-color warm palette), and shape (square, diamond, or cross). The system scales particle count based on hit velocity—gentle taps generate 14 particles while forceful hits spawn up to 40—creating visual intensity that matches playing dynamics.

Performance optimization required careful attention. I implemented a maximum cap of 160 active particles per drum to prevent frame rate degradation during rapid drumming sequences. The particle update loop uses efficient in-place array filtering to remove expired particles, and velocity damping creates natural deceleration over each particle's lifetime. Optional swirl physics adds rotational motion, and occasional sparkle particles (white/gold) provide visual accents.

React

While I'm satisfied with the final result, this feature represented significant learning through challenge. The particle system works well and adds essential visual energy to the experience, but the path to achieving meaningful visual feedback involved considerable struggle and experimentation. The learning process, though frustrating at times, deepened my understanding of balancing aesthetics with technical performance constraints.

Analyse

The primary challenge emerged from my project timeline decisions. After completing the audio system with drum samples and microphone FFT analysis, I spent substantial time late in the project figuring out how to make visual feedback meaningful and connected to the musical experience. This revealed a critical pattern: I had treated visual effects as a separate afterthought rather than designing them in parallel with audio functionality.

The complexity was compounded by the interconnected nature of particle system decisions. Visual appeal depends on particle count, colors, physics behavior, and lifespan working harmoniously together, but adjusting one parameter affected the others. Finding the right balance required extensive trial-and-error with velocity damping, spawn counts, color choices, and effect timing. Performance constraints added another dimension—beautiful effects that dropped frame rates to 20 fps were unusable regardless of aesthetics.

Improve

The key lesson learned centers on early integration of visual design with other systems. Next time, I would plan performance considerations from the beginning by establishing particle budget limits and profiling early prototypes to understand performance characteristics before building complex features. I would also create simple visual prototypes alongside audio development rather than sequentially, allowing the systems to inform each other's design from the start.

Specifically, I would build a debug control panel early in development with sliders for particle parameters (count, velocity, lifespan, colors) enabling real-time experimentation without code changes. This would accelerate the iterative refinement process that proved so valuable for the hand tracking system. Most importantly, I've learned that meaningful audio-visual connections require simultaneous consideration—designing how sounds and visuals relate should happen together, not as separate sequential phases.