MTEC-498 Capstone — PanGu — Spatial Audio, Live Phase Controller & Composition Assistant

Li Shi (Charlie) | Berklee College of Music | October 22, 2025

Instructor: Akito van Troyer | Repos: https://github.com/CharlieSL1/MTEC-498-Capstone | https://github.com/CharlieSL1/PanGu-Gesture-and-ML-Based-Control-for-Human-Centered-3D-Audio

Course Journey Overview

- Project Introduction: A next-gen spatial audio performance platform with a wearable, arm-mounted Circle MIDI controller.
- Semester Arc: From proposal → sensor bring-up → 3D printing & tech dev → OS/UX design → iteration → paper & demo (NIME-ready).
- Key Evolution: Consolidated multi-camera ML gesture recognition (RGB + depth) and robust OSC → Ableton/Max routing.
- Journey Highlight: Treating spatial sound as a playable instrument—position then perform in 3D, live.

Project Overview

- Problem / Opportunity: Spatial audio authoring is either too technical or too experimental; performers need an intuitive, real-time way to place and play sound in space.
- Audience & Use Cases: Musicians, producers, sound artists; studio exploration, live immersive shows, interactive exhibitions.
- Core Goals: (1) Embodied spatial control (2) Low-latency real-time mapping (3) Seamless DAW/Max integration (4) Portable setup.
- Repositories: Main (capstone) — https://github.com/CharlieSL1/MTEC-498-Capstone | Gesture/GUI — https://github.com/CharlieSL1/PanGu-Gesture-and-ML-Based-Control-for-Human-Centered-3D-Audio

System Architecture & Tech Stack

- Modules: Wearable MIDI (arm-mounted hex controller) • Gesture/GUI (MediaPipe) • OSC bridge • Max for Live (receiver).
- Tech Stack: Python, C/C++, Max/MSP; PyTorch/NumPy for ML; Arduino ecosystem for hardware; TorchAudio/DSP for audio.
- Data Flow: Cameras → ML gesture classify → OSC (/pangu/*) → DAW/Max → spatial audio engine.
- Testing/Tooling: Launchers, unit/integration tests (OSC, camera), packaged build PANGU_OS_V.1.3.0.

Technical Achievements & Quantified Accomplishments

- Real-time Gesture Control: MediaPipe-based recognition; multi-camera (Angle One/Two) with GUI feedback.
- OSC Protocol: Namespaced addresses (/pangu/gesture, /audio, /landmarks, /timestamp, /system/status) on port 7400.
- MaxForLive Integration: PanGu.maxpat receives OSC in Ableton Live for spatial parameter control.
- Packaging & Launch: Cross-platform scripts; build v1.3.0; quick-start entry points for GUI and main handler.

Metrics at a Glance

Metric	Value / Evidence	Notes
Main Repo Commits	15	As listed on GitHub history
Gesture/GUI Repo Commits	2	As listed on GitHub history
Supported Gestures	9	Fist, Peace, Point, Three, Four, Open, ThumbsUp, ThumbsDown, OK
OSC Address Patterns	7+	gesture/{name}, gesture/info, audio/{control}, landmarks/{i}, timestamp, system/status
Platforms	macOS / Windows / Linux	Cross-platform documented

Learning Outcomes Assessment

- Programming Proficiency: Built multi-module system (GUI, ML, OSC, Max) with clean separation and repeatable launch paths.
- Computational Thinking: Designed gesture-to-control mappings; modular OSC addressing; robust state handling.
- Industry Practices: Version control; tests; documentation/readme; packaging and platform considerations.
- Communication Growth: Public README and quick-start guide; structured risk/mitigation and roadmap.
- Peer Collaboration: Open to feedback; MaxForLive bridge for collaborators’ DAW setups.
- Professional Reflection: Aligns with creative-tech career in spatial audio and human-centered instrument design.

Challenge Analysis & Problem-Solving Growth

- Latency: Tuned I/O paths and control smoothing to keep gesture → audio response perceptually immediate.
- Hardware Endurance & Fit: Iterated on battery and wearability; refined hex-controller ergonomics for performance.
- Scope Control: Modularized roadmap; staged delivery (sensing → mapping → DAW routing → performance presets).
- Debugging: Added OSC/dev tools (osc_monitor, tests) to isolate camera/gesture vs DAW integration issues.
- Learning from Setbacks: Reduced fragile features; prioritized stable control set and clear OSC schema.

Skills Development & Integration

- Technical Growth: From single-camera experiments to multi-camera ML with GUI and packaged builds.
- Creative Development: Evolved from ‘engineering task’ into a playable spatial instrument and performance method.
- Professional Skills: Project planning, documentation, demo readiness, and integration with music production tools.
- Cross-Course Integration: EDI prototyping; Programming in C; Max/MSP; Creative Coding minor courses.
- Transferable Skills: Real-time systems, HCI for music, ML-to-DSP pipelines, cross-app protocol design.

Timeline & Milestones (per course plan)

Phase	Weeks	Focus / Outcome
1	1–3	Component checks; sensor bring-up; circuit design
2	4–7	3D modeling & printing; core tech development
3	8–10	OS/UX design; continued tech development
4	11–13	Iteration and improvements; stability
5	14–15	Paper, function checks, and mini-performance (NIME)

Future Planning & Professional Development

- Short-Term: Harden gesture models; add presets; extend spatial mappings (reverb trajectories, height cues).
- Career Alignment: Core artifact for spatial-audio and music-tech roles; live demo piece.
- Educational Pathways: Continue ML/DSP studies; instrument design; HCI for music.
- Portfolio Strategy: Video demo + README + Max device; clear quick-start for collaborators.
- Project Evolution: Standalone instrument (no computer), pre-install stage instruments, expand team roles.

Acknowledgments & Contact

- Instructor: Akito van Troyer
- Credits: Li Shi (concept & design), Xinyu Li (sketch & design), Special thanks to Akito van Troyer

- Contact: cshi@berklee.edu

- Links: Main Repo — https://github.com/CharlieSL1/MTEC-498-Capstone | Gesture/GUI — https://github.com/CharlieSL1/PanGu-Gesture-and-ML-Based-Control-for-Human-Centered-3D-Audio