Process & Decision Documentation

Side Quest Week 5 — "Drift: A Meditative Camera Experience" Name: Jimin Kim **WatID:** j243kim **Student Number:** 21062367

Project/Assignment Decisions

I decided to transform the Week 5 tutorial side-scroller into a meditative, non-interactive camera experience rather than building from scratch. The original codebase used a Camera2D class and parallax-ready structure, so repurposing it for an auto-scrolling contemplative scene was a natural extension. I chose a twilight color palette and slow pacing to evoke the quiet, reflective moments in Super Mario 64 where the player simply exists in the world rather than acting on it.

Role-Based Process Evidence

Name: Jimin Kim **Role(s):** Sole developer (individual Side Quest) **Primary responsibility for this work:** All design, coding, and iteration

Goal of Work Session

Build a reflective camera experience that scrolls through a world larger than the screen, using pacing and motion to evoke emotion. Additionally, implement the bonus feature of hidden interactive symbols for the camera to discover.

Tools, Resources, or Inputs Used

- Claude Code (Claude Opus 4.6) — used for code generation, debugging, and iteration
- p5.js reference documentation [1]
- p5.js noise() reference [2]
- Week 4 side-scroller codebase as starting point
- Lecture notes on camera systems and parallax

GenAI Documentation

Date Used: February 21, 2026

Tool Disclosure: Claude Code (Claude Opus 4.6, via Anthropic CLI)

Purpose of Use: Code generation for the meditative camera sketch, debugging camera boundary logic, fixing parallax math for hidden symbols, and iterating on visual feedback (cursor, click effects).

Summary of Interaction: Claude generated the initial sketch.js rewrite with parallax layers, procedural world generation, and hidden symbol mechanics. Across multiple iterations, it fixed bugs including: camera getting stuck at boundaries (direction flag logic), unreachable symbols due to parallax offset miscalculation, invisible mouse cursor, and trees floating above the ground plane.

Human Decision Point(s):

- I directed Claude to keep the original modular files (BlobPlayer.js, Camera2D.js, etc.) rather than deleting them, to maintain project structure consistency.
- I requested specific speed adjustments (0.4 to 0.8 to 1.2) after testing each increment myself.
- I identified that only 6 of 9 symbols were reachable and reported the bug for Claude to fix.
- I noticed the trees were floating above the grass and requested repositioning.
- I chose Super Mario 64 as the inspiration game over Skull.

Integrity & Verification Note: I tested the sketch in the browser after each change, verifying that the camera scrolled correctly in both directions, all 9 symbols were discoverable, the custom cursor appeared and responded to symbol proximity, and click effects were visible. I cross-checked the final code against the Week 5 rubric requirements.

Scope of GenAI Use: Claude wrote the core p5.js code (sketch.js, style.css, index.html updates, README.md). I made all creative direction decisions (theme, pacing, color palette choice, speed tuning), identified bugs through manual playtesting, and verified the final output met assignment requirements. This Process & Decision Documentation was drafted with Claude's assistance but reflects my actual development process.

Limitations or Misfires:

- Claude initially deleted all the modular files from Week 4 without considering the assignment's project structure expectations. I had to ask it to restore them.
- The first camera boundary logic used `camBaseSpeed *= -1` which caused the camera to flicker at edges instead of reversing smoothly. Required a second fix with a direction flag.
- The parallax math for symbol placement was incorrect on the first attempt, making 3 of 9 symbols unreachable. Claude fixed this after I reported the issue from testing.

Summary of Process (Human + Tool)

1. Started with the Week 5 tutorial side-scroller codebase.
2. Used Claude to rewrite sketch.js into a meditative auto-scrolling camera experience with parallax layers.
3. Tested in browser — found trees floating above ground, requested fix.
4. Tested camera controls — found left arrow did not change direction, reported and fixed.
5. Found camera getting stuck at right boundary — reported and fixed with direction flag.
6. Discovered only 6 of 9 symbols were reachable — reported parallax math bug, fixed.
7. Adjusted scroll speed through two iterations (0.4 to 0.8 to 1.2) based on feel.
8. Added custom glowing cursor and click burst effects after noticing the hidden cursor made symbols unclickable.
9. Final verification against rubric and assignment requirements.

Decision Points & Trade-offs

Decision 1: I chose to make the camera auto-scroll with gentle player influence (arrow keys nudge direction) rather than giving full manual camera control. This preserved the meditative, passive quality of the experience while still allowing the player to explore at their own pace. Full manual control felt too active for the reflective theme.

Decision 2: For the bonus hidden symbols, I chose to make them proximity-based (only glowing when the camera drifts near) rather than always visible. This rewards patience and attention, aligning with the meditative concept. The trade-off was that symbols could be missed on a single pass, but the auto-reversing camera ensures multiple opportunities to discover them.

Verification & Judgement

- Playtested the sketch multiple times in Chrome to confirm all features worked.
- Verified all 9 symbols were discoverable by playing through the full camera cycle.
- Re-read the Week 5 assignment requirements and rubric to confirm all criteria were met.
- Checked project folder structure against the required template (index.html, sketch.js, jsconfig.json, style.css, libraries/).

Limitations, Dead Ends, or Open Questions

- The sketch has no audio. Adding ambient sound (wind, chimes) would strengthen the meditative quality but was not attempted due to time constraints.
- The sky gradient is redrawn every frame using line-by-line rendering, which is not optimal but works within the 800x480 canvas size.
- Symbol positions use `random()` so they change on each page refresh, making the experience slightly different each time. This could be a feature or a limitation depending on perspective.

Appendix — GenAI Transcript

Below is a condensed transcript of the Claude Code (Claude Opus 4.6) conversation used during development. Minor formatting has been adjusted for readability.

User Prompt 1: "based on this file week 05, Feb 12, please modify the code that satisfies these requirements: Create a reflective or meditative camera experience that scrolls through a world larger than the screen. Use pacing and motion to evoke emotion. Bonus: Hide small interactive symbols or objects for the camera to 'discover.'"

Claude Response 1: Explored the existing project structure (sketch.js, BlobPlayer.js, Camera2D.js, WorldLevel.js, etc.). Rewrote sketch.js entirely into a meditative auto-scrolling camera experience with: twilight gradient sky, 120 twinkling stars, procedural parallax mountains (far at 0.2x, near at 0.45x), 80 swaying trees (0.7x), ground details (0.8x), 50 floating particles (0.6x), shimmering water layer, and 9 hidden symbols (bonus). Updated index.html, style.css. Created README.md. Deleted old platformer files (BlobPlayer.js, Camera2D.js, etc.).

User Prompt 2: "why did you remove these? will these allow to remove it based on the pdf I provided."

Claude Response 2: Acknowledged the assignment requires modular file structure. Asked user preference: restore modular structure, keep single file, or restore original files too.

User chose: "Restore original files too"

Claude Response 2 (continued): Ran `git checkout HEAD -- BlobPlayer.js Camera2D.js LevelLoader.js Platform.js WorldLevel.js levels.json` to restore all deleted files. Updated index.html to include all script tags.

User Prompt 3: "when I try to click the glowing symbols, I cannot see my mouse clicker, and should be there any changes when I click the glowing symbols?"

Claude Response 3: Identified `noCursor()` was hiding the mouse. Added `drawCustomCursor()` function — a soft glowing dot that grows larger with a pulsing ring when near a clickable symbol. Added `drawBursts()` and `spawnBurst()` functions for an expanding ring + spark particle effect on click. Increased click hitbox from 25px to 40px. Changed discovery text format to `"~ moon discovered ~"`.

User Prompt 4: "can you make that trees should be located on the green grass?"

Claude Response 4: Changed tree `baseY` from `VIEW_H * 0.72` to `VIEW_H * 0.82` to match the ground line. Updated canopy alpha mapping range accordingly.

User Prompt 5: "can you increase the speed little bit?"

Claude Response 5: Changed `camBaseSpeed` from 0.4 to 0.8.

User Prompt 6: "little bit more quicker"

Claude Response 6: Changed `camBaseSpeed` from 0.8 to 1.2.

User Prompt 7: "when I found 6 glowing symbols out of 9, I cannot go to right side no more, and 3 are missing. please fix this"

Claude Response 7: Identified the camera reversal logic `camBaseSpeed *= -1` was toggling every frame at the boundary, causing the camera to get stuck. Replaced with a `camDirection` flag (1 or -1) that only flips once when hitting each boundary edge.

User Prompt 8: "and also while it's moving the left arrow does not work. What I mean is I can not adjust the direction."

Claude Response 8: Changed keyboard controls so arrow keys set `camDirection` directly instead of just adding a small nudge offset. Left arrow now sets `camDirection = -1`, right arrow sets `camDirection = 1`, giving full directional control.

User Prompt 9: "only 6 glowing are existed? but in the game at the right corner of the bottom, it said out of 9"

Claude Response 9: Identified that with parallax 0.65, symbols placed beyond x=3400 in the 4800px world would never scroll into view. Calculated max reachable x: `VIEW_W + (WORLD_W - VIEW_W) * 0.65 ≈ 3400`. Changed symbol spacing to distribute all 9 within the reachable range instead of across the full world width.

End of transcript.

References

[1] p5.js Foundation. 2024. p5.js Reference. Retrieved February 12, 2026 from https://p5js.org/reference/

[2] p5.js Foundation. 2024. noise() Reference. Retrieved February 12, 2026 from https://p5js.org/reference/p5/noise/