Camera-Controlled Snake Game

ECE 554 Project Proposal · University of Wisconsin-Madison · Spring 2025

1. Team Introduction

Member	Major	Key Skills	Primary Responsibility
Chance Howarth	CE	FPGA, hardware design	Set up hardware infrastructure (VGA, SDRAM etc) Color detection and gesture recognition from the camera feed
Munasib Ilham	CE	C/C++, Verilog	Snake engine - Display rendering and snake logic
Anirudh Kompella	CS	C/C++, React+JS	Set up hardware infrastructure (VGA, SDRAM etc), testing and project report
Amer Salem	CE	FPGA, verilog	Top-level wiring, direction-detection, Hardware testing

2. Project Overview

Our project is a real-time Snake game running on an FPGA, controlled by hand gestures. The player holds a colored object in front of a camera and moves it into one of four zones — up, down, left, or right — to steer the snake. The entire game, from reading the camera to drawing on the screen, runs directly in hardware with no processor involved.

We will be building a hardware framework that handles low-level camera input, display output, and memory management. Additionally, our three contributions are the functional modules that sit on top of that foundation: gesture detection, game logic, and rendering.

Motivation: Snake is a great fit for an FPGA because it requires real-time responsiveness that is hard to guarantee in software. Every frame, the system has to read the camera, figure out where the player's hand is, update the game state, and draw the result — all within about 16 milliseconds. Custom hardware handles this kind of tightly-timed parallel work far more reliably than a processor, and produces a system that is genuinely fun to demonstrate.

3. Technical Approach

The system is divided into four layers. The bottom layer — camera input, memory buffering, and display output — is provided to us as a starting point. We will design and implement the three layers above it:

- **Gesture detection:** Reads each camera frame and determines which direction the player is signaling. It looks for a distinctly colored object in one of four screen zones and filters out rapid or accidental changes before passing the direction signal to the game logic.
- **Game logic:** Handles everything about Snake: moving the snake each game tick, checking for wall and self-collisions, placing food randomly in an empty cell, and tracking the score. The score is shown both on screen and on the board's built-in number displays.
- **Display rendering:** Draws the current game state onto the screen pixel by pixel — snake head and body, food, the play-field border, and score. It also supports the board's push-buttons as a fallback control method for testing without the camera.

Why this fits ECE 554: The project requires designing hardware modules that run in parallel, building custom state machines and memory structures in hardware, and carefully integrating multiple blocks with different timing requirements — all core skills for this course.

4. Execution Strategy — Risk Management

Risk	Mitigation
Gesture detection is unreliable under different lighting	Make the color sensitivity threshold adjustable via switches on the board; push-button controls serve as a backup so the demo can still run

Risk	Mitigation
Game logic is hard to synthesize correctly in hardware	Write and fully test all game logic in software simulation before attempting to compile to hardware
Timing closure fails after full integration	Set up timing constraints early and verify them incrementally
Running out of on-chip memory for the snake body	Limit the maximum snake length; use external memory as a fallback if needed
Modules don't work together when integrated	Weekly check-ins with a shared code repo; all individual modules must pass simulation before integration begins

5. Project Timeline & Evaluation

Midterm goal (Week 6): All three modules pass simulation. Snake is playable on the FPGA using push-buttons, displaying correctly on screen. The camera is live but gesture steering is not yet required.

Final goal (Week 10): Fully playable via hand gestures. Snake moves, eats, grows, and ends correctly on collision. The score is displayed on screen. Project report submitted.

Week	Task / Milestone	Owner	Description
Wk 1-2	Setup & hardware bring-up	All	Verify camera, display, and memory work on the board; divide module ownership
Wk 2-4	Gesture detection	M1	Detect the colored object from camera and output a direction signal
Wk 3-5	Snake game logic	M2	Movement state machine, collision detection, and food placement
Wk 4-5	Display rendering	M3	Draw the snake, food, borders, and score onto the screen
Wk 6	**Midterm checkpoint**	**All**	**All modules verified in simulation; Snake playable via push-buttons on FPGA**
Wk 6-8	Synthesis & wiring	M1-M4	Compile all modules to hardware and wire everything together
Wk 8-9	Hardware testing & debug	All	Test on the board; fix timing issues; tune gesture detection
Wk 10	**Final demo + report**	**All**	**Fully playable camera-controlled Snake on FPGA; report submitted**

6. Team Responsibilities

Each member owns one module end-to-end, from design through hardware verification. All members will work on different assignments that will be subject to change:

- **Chance Howarth — Set up hardware infrastructure (VGA, SDRAM etc) Color detection and gesture recognition from the camera feed**
- **Anirudh Kompella — Set up hardware infrastructure (VGA, SDRAM etc), testing and project report**
- **Munasib Ilham — Snake engine - Display rendering and snake logic**
- **Amer Salem — Top-level wiring, direction-detection, Hardware testing**

All four members will attend weekly check-ins, participate in hardware debugging, and present together at the final demonstration.