

Project Proposal

Living Worlds: Multi-Scale 2.5D GPU Cellular Automata for Dynamic Terrain Generation

Student: Mohammad Alkhalifah - 182822

Instructor: Prof. Markus Hadwiger — **TA:** Peter Rautek

Motivation and Background

Procedural terrain generation in games and simulations typically relies on static noise functions that produce unchanging worlds. While efficient, these approaches lack the dynamic evolution and emergent complexity found in natural systems. Cellular automata (CA) offer a compelling alternative, capable of generating complex patterns from simple local rules, as demonstrated by Conway's Game of Life. Recent work in GPU-accelerated CA has shown impressive performance gains (bryanoliveira, 2021: 729 gen/s on 13500×13500 grids), yet applications to terrain generation remain largely unexplored, particularly using modern APIs like Vulkan.

This project addresses a significant gap in applying multi-scale, coupled CA systems to geological simulation. While hierarchical CA has been studied in other fields, its application to terrain generation with geological feedback loops represents a novel contribution. By leveraging Vulkan's compute shaders and explicit synchronization control, this project aims to achieve real-time performance while maintaining visual quality comparable to modern 2.5D games like *Hades* and *Hollow Knight*.

Objectives

- **Implement a high-performance 2D cellular automaton system** using Vulkan compute shaders with ping-pong buffering for race-free parallel updates on 1024×1024 grids at 60+ FPS.
- **Develop a physically-based erosion CA** following Jako & Toth (2011) that operates on continuous heightmap values, simulating hydraulic and thermal erosion processes.
- **Create a coupled multi-layer CA system** where geological processes (erosion/deposition) and ecological dynamics (biome spreading) interact through bidirectional feedback loops.
- **Design a 2.5D rendering pipeline** using height-based vertex displacement, normal mapping from derivatives, and atmospheric perspective to create depth perception without full 3D complexity.
- **Benchmark performance against existing implementations**, targeting 500+ generations/second based on CUDA/OpenGL baselines, while analyzing Vulkan-specific optimizations.

- **(Optional) Implement GPU tessellation** for dynamic level-of-detail and explore reaction-diffusion systems (Gray-Scott model) as an alternative to noise-based terrain initialization.

Methodology

The implementation will utilize **Vulkan with C++**, leveraging compute shaders written in GLSL. The core CA simulation will employ a ping-pong buffer pattern using `VkImage` storage with appropriate pipeline barriers for synchronization. The architecture consists of three main components:

1. **Compute Pipeline:** Separate compute shaders for geological CA (erosion/deposition), biome CA (ecological spreading), and fractal noise generation (Simplex/Perlin fBm).
2. **Memory Management:** Heightmaps stored as R32F textures (4MB per buffer), biome maps as R8_UINT (1MB per buffer), with total memory footprint under 20MB for 1024×1024 grids.
3. **Rendering Pipeline:** Isometric projection with height-based shading, using fragment shaders for biome coloring and atmospheric effects.

Testing methodology will include correctness validation using known CA patterns (gliders, oscillators), performance profiling via Vulkan timestamp queries and Nsight Graphics, and visual quality assessment against reference 2.5D games.

Expected Results and Deliverables

The system is expected to achieve **60+ FPS on 1024×1024 grids** with 8 distinct biome types exhibiting realistic spreading patterns. The erosion simulation should produce natural-looking valley networks and drainage patterns. Performance is targeted at 500+ CA generations per second, with memory usage under 200MB. The optional tessellation extension could enable 2048×2048 grids while maintaining real-time performance.

Deliverables include:

- **Vulkan application** with complete CA simulation and 2.5D rendering pipeline, including source code and build instructions.
- **Performance analysis** comparing Vulkan compute shaders against published CUDA/OpenGL benchmarks, with detailed profiling data.
- **Interactive demo** showcasing real-time terrain evolution, user-controlled parameter modification, and pattern spawning.
- **Technical report** documenting implementation details, optimization strategies, and analysis of emergent behaviors.
- **Video demonstration** (2-3 minutes) showing time-lapse terrain evolution and interactive features.
- **Final presentation** summarizing the multi-scale CA approach, Vulkan implementation challenges, and achieved performance metrics.

Timeline

Week	Milestone	Deliverable
1	Vulkan initialization, compute pipeline setup	Working Vulkan application
2	Conway's Game of Life with ping-pong buffers	60+ FPS on 512×512 grid
3	Heightmap CA with erosion rules	Visible erosion patterns
4	Multi-state biome system	8 biome types with spreading
5	2.5D rendering implementation	Isometric view with depth cues
6	Performance optimization, UI, polish	Final application

References

1. B. Jako and B. Toth, "Interactive Hydraulic and Thermal Erosion on the GPU," *GPU Pro 2*, 2011.
2. B. Chan, "Lenia and Expanded Lenia: Continuous Cellular Automata," *Artificial Life Conference*, 2020.
3. bryanoliveira, "High-Performance Cellular Automata GPU Implementation," GitHub Repository, 2021. <https://github.com/bryanoliveira/cellular-automata>
4. F. Losasso and H. Hoppe, "Geometry Clipmaps: Terrain Rendering Using Nested Regular Grids," *ACM SIGGRAPH*, 2004.