Tony Yiding Tian

Philadelphia, PA | (267)249-1202 | tonytg@seas.upenn.edu | github.com/tonytgrt

Education

University of Pennsylvania - School of Engineering and Applied Sciences Philadelphia, PA

B.S.E. in Computer Engineering, Accelerated M.S.E. in Computer Graphics and Game Technology *May 2027*

- GPA: 3.74 — Relevant Courses: GPU Programming, Advanced Rendering, Interactive Computer Graphics, Computer Animation, Operating Systems, Data Structures & Algorithms, Computer Architecture

Projects

CUDA Path Tracer | *CUDA, C++* Sep 2025

- Monte Carlo path tracer capable of rendering complex 3D scenes with custom 3D models and environment maps
- Implemented shading BSDF kernel supporting global illumination, multiple importance sampling, anti-aliasing, depth of field, sub-surface scattering and various material types including diffuse, specular, and refractive surfaces
- Designed efficient stream compaction kernel to remove terminated rays, improving performance by 30% for highly complex scenes. A standalone project with detailed performance analysis is available here: Github Link
- Utilized NSight Compute and NSight Graphics for performance profiling and optimization
- Project Repo and Demo link: CUDA-Path-Tracer. A previous GLSL path tracer project demo: Render-Demo

Mini Minecraft - Voxel-based 3D Game | *C++, OpenGL, Qt, GLSL* Oct 2024 – Dec 2024

- Collaborated in team of 3 to develop fully-featured voxel game engine in C++ using OpenGL, generating infinite worlds with 1M+ blocks and maintaining 60+ FPS performance
- Engineered procedural terrain generation system using layered 2D/3D Perlin noise algorithms, creating 5 distinct biomes (Grassland, Mountain, Desert, Islands, Caves) with biome-specific block distributions and procedurally placed vegetation assets
- Implemented post-processing rendering pipeline with custom GLSL fragment shaders, featuring dynamic underwater/lava distortion effects using UV coordinate manipulation and real-time crosshair overlay rendering
- Developed dual physics simulation system: gravity-based collision detection with terrain for ground movement, and buoyancy calculations for water/lava interaction, plus creative fly-mode with 6-DOF movement
- Built efficient chunk-based world management system with frustum culling and LOD optimization, reducing draw calls by 80% through face culling of adjacent blocks
- Implemented real-time block manipulation (mining/placing) with ray-casting intersection testing and immediate mesh updates, supporting 16 different block types with unique textures and properties
- Project demo showcasing all features: Game-Demo

PennOS - UNIX-like Operating System | *C, Shell, Kernel* Mar 2025 – May 2025

- Architected and implemented a complete user-level operating system in C with team of 4, featuring 8000+ lines of systems code with full process lifecycle management
- Designed Process Control Block (PCB) data structure managing 50+ concurrent processes with metadata including PID allocation, priority levels, parent-child relationships, signal handling, and user/kernel stack management
- Implemented preemptive multi-level priority scheduler supporting 3 priority levels with Round Robin time-slicing (10ms quantum), preventing starvation through priority aging and achieving 95% CPU utilization
- Built POSIX-compliant interactive shell supporting 15+ built-in commands (ps, kill, jobs, fg/bg), I/O redirection, pipeline chaining, and batch script execution with robust error handling

Experience

Linux Kernel Policies Research Assistant - PURM Scholar May 2025 – Aug 2025

Learning Directed Operating System (LDOS), Prof. Sebastian Angel *Philadelphia, PA*

- Developed eBPF-based kernel monitoring infrastructure collecting real-time TCP networking metrics retrieved from 5 crucial kernel tcp functions of tcp_v4_rcv, v4_connect, state_process, congestion_control, and cubic
- Engineered high-performance data analysis pipeline processing 10,000+ TCP state transitions per second, identifying critical performance bottlenecks in kernel networking policies and congestion control algorithms
- Architected and contributed 2000+ lines of C and Python code to open-source KernMLOps repository, implementing kernel probing infrastructure used by 15+ researchers

Technical Skills

Graphics/Rendering: NSight Profiling, Path Tracing, Deferred Rendering, Rasterization, Animation systems, PBR
Programming: CUDA, C/C++, GLSL, Parallel algorithms, Memory management, Rendering pipeline
Tools/APIs: Nvidia NSight, Visual Studio, Qt, OpenGL, WebGPU, Git, CMake, MakeFile, Clang, GDB, GCC