DANIEL ZHONG

☎ 628-222-1000 ✉ danielzhong2000@gmail.com
www.linkedin.com/in/danielzhong-/ github.com/DanielZhong https://danielzhong.github.io/

EDUCATION

University of Pennsylvania **Aug 2023 – Dec 2024**
Master of Science (MS) in Computer Graphics and Game Technology *Philadelphia, PA*

- **Teaching Assistant**: CIS 565: GPU Programming and Architecture

University of California-Santa Cruz **Sep 2019 – Jun 2023**
Bachelor of Science (BS) in Computer Science: Game Design *Santa Cruz, CA*

- **Award**: Highest Honors award in the School of Engineering (2023), Dean's Honor (2020-2021)
- **Certificate**: Tencent Unreal Engine Game Client Course Certificate (2022)

TECHNICAL SKILLS

- **Programming:** C++, C#, CUDA, TypeScript, Python
- **Graphics:** OpenGL, WebGPU, Vulkan, GLSL, HLSL
- **Software & tools:** Houdini VEX, MAYA MEL, Qt, GitHub
- **Game Engine:** Unity, Unreal Engine
- **VR/AR/MR Frameworks:** Oculus SDK, OpenXR, ARKit
- **Profiling & Debugging Tools:** NVIDIA Nsight, RenderDoc

EXPERIENCE

Graphics Software Engineer **May 2024 – Present**
Bentley System *Philadelphia PA, US*

- **Rendering Library:** Maintained and revamped iTwin.js project's **real-time rendering** library and upgraded from WebGL to WebGPU, leveraging **GPGPU** capabilities to elevate user visual experience and optimize performance by at least 30 %
- **Features implementation:** Designed and developed multiple WebGPU-based features, including **ray tracing** using NVIDIA RTX pipeline, real-time dynamic Level of Detail (LOD) using **compute shaders**, Point Cloud Rendering, WebGPU Anime 4K upscaling, and temporal anti-aliasing (TAA) & ambient occlusion (TAO), enhancing **GPU rendering optimization** and **GPU parallel computing** capabilities
- **Research and Presentation**: Solved complex graphical challenges by conducting extensive research, analyzing, and implementing multiple recent SIGGRAPH papers. Presented the advancements and newest techniques in graphics rendering during sprint reviews

Freelance Game Developer **Apr 2023 – Aug 2023**
Upwork *San Francisco CA, US*

- **Game Development:** Led the development of multiple diverse and innovative games utilizing both **Unity** and **Unreal** Engine, encompassing genres such as a dating simulation, a platform fighter, and a score-chasing arcade game, etc
- **Client Communication:** Successfully communicated with clients and teams, translating visions into game features

Unity Developer Internship **Apr 2022 – Sep 2022**
RCT.AI (Mirror World) *Beijing, China*

- **NFT Mobile Game Development:** Successfully developed and published two NFT mobile video games: "Mirrama," an Action RPG, and "Bom," a Platform Drop Battle Royale, demonstrating expertise in cutting-edge gaming trends and blockchain integration
- **Collaboration:** Enhanced team productivity by working closely with backend developers to integrate WebSocket APIs, implementing real-time data communication features such as the NFT wallet transaction function
- **Performance Optimization:** Collaborated with the Art team on Atlas to enhance game performance by **reducing draw calls**. Implemented techniques such as object pooling, Level of Detail (LOD), culling, texture atlasing, etc
- **Game Mechanics Implementation:** Coordinated with the operations and design teams to implement the battle system mechanics, refined Unity UGUI for skill cool-down displays, and also implemented key game features including reward systems, inventory management, etc
- **Tool Development:** Created an intuitive **debug** panel and tools for Art and QA team, improving overall team productivity by 70 %

PROJECTS

Mini Minecraft Game Engine | *C++, OpenGL, QT, GLSL* **April 2024**

- **Led Team Development:** Directed a 3-person team to design and develop a **real-time deferred rendering pipeline** for a **rasterization**-based Minecraft game engine with advanced player movement precise collision detection using grid marching
- **Procedural Generation:** Infinite biomes generation using Perlin noise, and procedural texture, shape, sky box using Ray March and SDF
- **Dynamic Texture Mapping:** Utilized texture atlas and displacement mapping to create realistic environments with animated textures
- **Rendering Features:** Achieved high-quality material effects with **physically-based rendering** (PBR), incorporating screen space reflection (SSR), subsurface scattering (SSS), and Cook-Torrance BRDF for realistic plastic and metal appearances, shadow maps, etc

CUDA GPU Accelerated Path Tracer | *CUDA, C++, GLSL* **October 2023**

- **Global Illumination:** Demonstrated expertise in rendering mathematics for **BSDF models**, **importance sampling**, and **Monte Carlo** methods
- **Enhanced Visual:** Employed super sampling (SS), Depth of Field (DOF), and Atrous Denoiser to achieve smoother visuals and cleaner images.
- **Optimized Performance:** Utilized Stream Compaction to efficiently remove inactive rays, optimizing CUDA performance and achieving a 45% performance improvement based on profiling **Nvida Nsight**.

MMORPG Game with Toon Shading Style (Zera: Reborn) | *Unity, C#, Universal Render Pipeline (URP), HLSL* **June 2023**

- **Leadership**: Directed a team of 8 as Technical Art Lead, successfully launching the game on itch.io with key features such as a battle system, NPC interaction and narrative system, teleportation mechanics, navigation map, and comprehensive inventory and reward systems
- **Shader Programming:** Developed a 360° character showcase scene, dynamic lighting and weather systems with particle system and post-processing effects
- **Toon Shading:** Created custom toon shaders for a consistent stylized aesthetic, optimized for performance across hardware

Battlefield Scavenger | *Unreal Engine, C++, Vulkan* **July 2022**

- **Game System:** Created an interactive 3D environment with weapon collection, real-time combat, score saving and loading, and multiple levels
- **Enemy AI:** Developed a complex **animation system** utilizing animation layer blending and designed diverse enemy AI using Behavior Tree
- **Grass Simulation:** Implemented GPU parallel computing grass simulation using **tessellation** and **compute shaders** within the **Vulkan** API