Congpei Zhou

7372287622

Portfolio: https://bobhansky.github.io/portfolio/

Education

University of California San Diego Sep.2025 - Dec 2026

Computer Science (Master)

University of Minnesota Twin Cities Sep. 2021 - Aug. 2025

Computer Science (Bachelors) GPA *3.92/4*

Professional Experience

NetEase Games (Hangzhou, China) Sept. 2024 - Nov. 2024

Game Engine Development Engineer Intern

- Writing extension for **Python 3.12** (Cpython) with **C++ 17** to dump and load data with Msgpack format (**json-like**).
- Using shared resources mechanism to reduce memory usage when loading object from binary msgpack data to realtime game engine. Enable custom edit for the object whose data is shared while preserving the shared data unchanged.
- Using **RAII** mechanism wrapper class for PyObject* to prevent memory leaks.

University of Minnesota (Minneapolis) May 2023 - Sept. 2024

Research Assistant

- Building **Unity3D VR** projects, setting up gameplay logic (**C#**) with **Meta Oculus Quest2,** writing custom post-processing effects with built-in rendering pipeline. Debug experience in **Unity Frame Debugger, Unity OpenXR.**
- Experience with writing **shader** in **HLSL** to create a fixed fovea restrictor with peripheral post-processing effect to mitigate the cybersickness using **VR** headset. Reducing the algorithm time complexity from O(N^2) to **O(2N)** by optimization in shaders.
- Conducting in-person VR experiment on 36 participants to collect real-time experiment data such as 3D motion tracking. Data collected by scripts **(C#)** written from scratch.

Projects (check my portfolio for more)

- **Monte Carlo Ray Tracing Renderer (C++, CMake, Git)** Jul. 2023 - Present

  Link: bobhansky/TutuRenderer: A CPU based Renderer for personal interest and education. (github.com)

  A CPU based offline **ray tracer** built for personal interest and education purpose. **C++** code completely written from scratch. The implemented state of the art integrator is **Bidirectional Path Tracing**. Features includes Microfacet Model, Multiple Importance Sampling, BVH, etc. Use **Multithreading (std::thread)** to facilitate rendering, speeding up the rendering by **12x**. Use Git as version control.

- **Vulkan Grass Rendering (C++, Vulkan)** Dec. 2025 - Dec 2025

  Link: bobhansky/VulkanGrass

  A real time grass field rendering demo. GPU programming. Used **C++**, **Vulkan**, and GLSL to set up Compute Pipeline, Render Pipeline, Descriptor sets etc. Implement GPU side Culling, tessellation (Level of Details), and whole pipeline for grass physics animation, shadings. A thorough Performance analysis is written in readme. Used Git as version control.

Skills

- **C++, Python, Java, C#, Rendering, Vulkan, OpenGL, GLSL/HLSL, RenderDoc, Unity3D**
- Knowledge in Machine Learning, Multithread programming.
- Specialized in Computer Graphics, **Rendering**, GLSL, HLSL, **Ray Tracing,** real-time rendering performance bottleneck analysis and improvement.