Arjun Krishnan

Email: ajn.krishnan@gmail.com | GitHub: github.com/AKris0090 | Website: ajnkrishnan.me | LinkedIn: @ajnkrishnan

Software engineer specializing in graphics and gameplay programming

Relevant Work Experience

Sony Interactive Entertainment - Software Engineering Intern — Summer 2025

- Implemented gameplay features into an Unreal Engine game targeted at PC and console platforms.
- Conceptualized, designed, and shipped a 2D Match-3 mobile game in Godot, experiencing the full development cycle from concept to product; self-taught React Native to integrate the game into the PlayStation App.
- Playtested, gathered feedback, and iterated. Built a particle state machine, save system, tutorial card system, and a flash banner. Sourced official astro bot assets from designers on the mobile development team.
- Explored graphics programming using PS5 SDK. Wrote shaders and a .OBJ loader to render on PS5 devkit.

Projects

Orchid Game Engine - C++, Vulkan, PhysX — June 2024 - Current

Conceptualized and developed a 3D game engine, blending elements of photorealistic and non-photorealistic rendering.

- Implemented graphics techniques including compute skinning, PBR texture processing, cascaded shadow mapping, physically-based bloom, inverse hull outlines, and a custom toon shader.
- Developed a Dynamic Diffuse Global Illumination (DDGI) system by ray tracing from a grid of probes to calculate indirect illumination.
- Created singleton managers for graphics, physics, time, and input, making it easy to script and animate objects and quickly prototype games with unique mechanics.
- Utilized frame analysis tools such as RenderDoc and NVIDIA Nsight to locate bottlenecks in application performance. Achieved a **32% reduction** in frame render time, from **8.1 ms/frame** to **6.15 ms/frame**, through compute frustum culling and a depth prepass.

Stingray Raytracer - C++, CUDA — June 2023 - Sept 2023

Designed an iterative ray tracing application, accelerated using the CUDA Runtime API.

- Incorporated features such as iterative anti-aliasing, physically-based materials with variable parameters, and direct light sampling for specular lights and soft shadows.
- Optimized frame time from **350 ms** to **15 ms (~23x speedup)** by implementing Bounding Volume Hierarchies and parallelizing ray calculations on the GPU.

3D Model Renderer - Java — Apr 2022 - June 2022

Developed an interactive GUI to render 3D models using the Processing API

- Implemented a perspective camera, light & shadow vector arithmetic, back-face culling, and polygon by polygon rendering to display and interact with complex 3D models.
- Wrote vector and matrix arithmetic classes, alongside various linear algebra formulae, from scratch.
- Designed and integrated an .OBJ file parser, enabling users to upload and view custom models.

Education

University of California: Santa Cruz — Graduated December 2025

Bachelor of Science in Computer Science: Game Design, Minor in Computer Science

Technical Skills

Languages: C++, C, Python, Java, C#, GLSL

Software Tools & APIs: Vulkan, CUDA, Visual Studio, Nvidia Nsight, Git, Blender, Unreal Engine, Unity