TREVOR DAVID BLACK

408-218-8770 • trevordblack@trevord.black
trevord.black • linkedin.com/in/trevordblack • github.com/trevordblack

RELEVANT EXPERIENCE

Google - Android GPU, *Software Engineer* May 2020 – Present

- One of the Representative Members for Android on the Khronos Vulkan WG Board
- Maintainer Android Vulkan loader
- Maintainer Android Vulkan instance and swapchain
- Maintainer Android's Vulkan and OpenGL ES documentation on SAC (source.android.com/docs/core/graphics/arch-vulkan)
- Maintainer Android GPU code interaction with Hardware Buffers and AIDL
- Simplified and consolidated Android Graphics stack interaction with Hardware Buffers and AIDL
- Championed and created the Android Baseline Vulkan profile to simplify Vk development across Android
- Updated the Android Baseline Vulkan profile for 2022
- Designed and implemented Cache reference counting and eviction schema for ANGLE Vulkan objects
- Wrote, tested, and pushed proof of concept Multi-Threading for D3D11 backend in ANGLE
- Created and hardened Android Hardware Buffer functionality for SwiftShader and ANGLE
- Owned and implemented multiple features within Android's Game Dashboard feature
- Wrote and distributed many internal design documents that are being implemented across the GPU team

Intel Corporation, *Graphics Engineer* December 2018 – May 2020

- Optimization of Intel's DirectX12 Driver
- Graphics driver development DX12 and Metal
- Organized a weekly meeting to cover and teach Luna DX12 book to new DX12 employees
- Rewrote existing thin Graphics API shimming layer for portability and simplicity, rewrote build system
- Engineered Clang-based tool to scrape c headers (obj-c, c++) for global scope function signatures to YAML
- Intel OSPRay work on OpenVKL, OSPRay studio, Embree
- Compute liaison and maintainer of Geekbench benchmarking suite for Intel Mac platform

UCLA Electrical Engineering, *Graduate Student Researcher* December 2015 – June 2018

- Wrote LLVM compiler passes (C++) for a more generalized programming of academic ASIC digital chips
- LLVM based general programming for DARPA program DSP Chips (C++ and MATLAB)
- Aided in draft proposal for hardware DARPA project

PROJECTS AND RESEARCH

Co-Author, raytracing.github.io

- Open Source rerelease of Peter Shirley's *Ray Tracing* series
- Responsible for *significant* rewrites of all three books
- Responsible for much of the overall direction and philosophy of the rewrites
- Hosted a SIGGRAPH 2023 BoF (alongside co-authors) to announce v4.0.0 went alpha
- Bug fixes, code improvements, language adjustments, Math notation in LaTex, repo maintaining
- 4.0.0-alpha is a hellscape that we'll never escape from
- please send us more issues to perpetuate the nightmare: github.com/RayTracing/raytracing.github.io

Ray Tracing Gems Volume 2 Chapter

- *A Breakneck Summary of Photographic Terms and their Utility to Ray Tracing*
- Free access at link.springer.com/book/10.1007/978-1-4842-7185-8
- Created, pitched, wrote, edited, shipped

Homebrewed C++ PBR Rendering Engine

- Renders in either OpenGL 4.5 or DirectX12
- Wrote own asset libraries including Models, Meshes, Materials, Textures, and Lights
- Designed own geo libraries with own Model, Mesh, and Texture structures
- Model loading of .obj via tinyobjloader and .fbx via Assimp loader
- Wrote own math libraries including Vectors, Matrices, and Quaternions
- Wrote own Ubershader in HLSL and GLSL
- Added support for Alpha Blending, Transparency, MSAA, Anisotropic Filtering, and Stenciling

Multithreaded C++ Monte Carlo Path Tracer

- Multi-threaded a classical Whitted style Monte Carlo path tracer
- Support for the typical materials: Lambertian, Metals, Dielectrics, Volumetrics, Ubershader
- Stratified (jittered) sampling and Gaussian filtering for supersampled anti-aliasing
- Model loading of .obj via tinyobjloader and .fbx via Assimp loader
- Rendered to either a DX12 or OGL window for "interactive" sample rates. This buffer accumulated over time
- Drafted up code to track variance per pixel over time and prioritize samples to highly variant pixels
- Hooked up ImGui to OGL window. I never ended up doing anything particularly interesting with this

Experiments with PBRT

- Skeleton of ASSIMP to .pbrt conversion, finished by Matt Pharr groups.google.com/g/pbrt/c/A7PZg2dqjyY
- Overhauled Spectral code to enable runtime spectrum resolution
- Massive investigation of differing spectral techniques: Vector, Gaussian, Single Wavelength, Hero Wavelength
- Integrated Open Image Denoise into PBRT to run denoiser over beauty shot

Experiments with OptiX

- Completed OptiX7 tutorial from SIGGRAPH 2019
- Ported Peter Shirley's *Ray Tracing* series to OptiX 6.0
- Rewrote CMake build system (multiple times)
- Wrote source and outline for own text: An Introduction to GPU Ray Tracing in OptiX 6.0 | *shelved due to 7.0*

EDUCATION

Master of Science | ***Electrical Engineering*** Sep. 2015 – Dec 2017
University of California, Los Angeles Los Angeles, CA

- MS Thesis Accepted 18th Dec, 2017: *Toward a Generalized Model of Hardware Parallelism and Reconfigurability*

Bachelor of Sciences | ***Major: Electrical Engineering, Graduated with Honors*** Sep. 2010 – June 2014
University of California, Davis Davis, CA

SKILLS

Programming Languages: C, C++, Java, Python, Rust, Julia
Compute/GFX: Vulkan, DirectX12, OpenGL, CUDA, ISPC, SIMD, Multithreading
Tools: Perf, VTune, LLVM, lldb, Clang, CMake, vim, neovim, gvim (I like vim), Git, Visual Studio, XCode
Content Creation: Unity3D, ShaderToy, Blender, Krita, Davinci Resolve, Photoshop