Tongwei Dai

408-590-9830 tongweid@gmail.com LinkedIn Santa Clara, CA
Portfolio: https://dw218192.github.io/portfolio/

EDUCATION

University of Pennsylvania (GPA: 4.0/4.0) **PA, USA**
MSE in Computer Graphics and Game Technology Aug.2022 - Dec.2023
Core courses**:** *Physically based Rendering, GPU Programming, Procedural Generation, Computer Animation*

University of Southern California (GPA: 3.8/4.0) **CA, USA**
BS in Computer Science Aug.2017 - Jun.2021

SKILLS

- **Programming:** C/C++, Python, C#, CUDA, GLSL, HLSL, Maya MEL
- **Tools:** Unity, Unreal Engine 3/4/5, Git, Maya, Houdini
- **Tech:** OpenUSD, NVIDIA Warp, Vulkan, OpenGL, WebAssembly, LLVM

WORK EXPERIENCE

System Software Engineer | NVIDIA ***Jan.2024 – Present***

- Developed various **synthetic data generation (SDG)** tools focused on character AI and crowd simulation in NVIDIA's Isaac Sim for training object detection and multi-object tracking models.
- Helped prototype and optimize the **MEGA Omniverse blueprint**, a comprehensive framework designed to simulate, test, and optimize robotic fleets in industrial digital twins.
- Implemented CI/CD pipelines using **GitLab CI/CD** for new synthetic data generation projects, enabling automated builds, tests, and deployments for the first time, replacing manual and error-prone release processes.
- Implemented an **end-to-end testing framework** for SDG projects, ensuring data integrity, reproducibility, and robustness across multiple datasets and use cases.

Research Assistant, Software Developer Intern | Penn Computational Intelligence Lab ***May.2023 – Sep.2023***

- Incorporated Microsoft's **mimalloc** allocator into **WASI-SDK**'s C standard library.
- Implemented a multi-way segment tree to achieve O(log(n)) address-tag pair lookup for a custom kernel, achieving a **5x** speed boost with only **1.3x** worst-case memory overhead.
- Built a specialized memory allocator based on mimalloc for disaggregated computing applications.

Unity Engine Tool Programmer | NetEase Games ***May.2020 – Aug.2020***

- Used **Unity Editor API** to develop pipeline automation and scene object placement tools using **C#**, greatly improving the working efficiency of level designers.
- Refactored existing animation debugging tools, achieving **~50%** performance gain and reducing memory usage by **~100%.**
- Leveraged **compute shaders** to optimize terrain brush tools by transferring some workloads to GPU.
- Held regular meetings with the art team to gauge tool usage and collect performance data.

PROJECTS

Differentiable Rendering for Material Optimization

- Developed an **OpenUSD**-native differentiable path tracing renderer using **NVIDIA Warp**.
- Implemented path-replay backpropagation for differentiable path tracing, computing gradients with respect to PBR metallic–roughness parameters through Monte Carlo light transport.
- Implemented **latent-space inverse rendering** with **Adam + LR scheduling** to recover metallic–roughness material parameters from image-space losses.

Lightweight 3D Scene Editor and Rendering Playground

- Developed PTStudio, a lightweight **C++17** 3D scene editor + renderer with **OpenGL** and **Vulkan** rendering backends.
- Implemented various low-level engine primitives, including a static reflection system using **C++ template metaprogramming** to register types/properties at compile time and drive editor-facing introspection (e.g., generic inspectors/serialization hooks)

Maya Plugin for Procedural Flower Modeling

- Implemented Blossom Pro, a Maya Plugin using C++ and **MEL** to procedurally generate tree-like structures and flowers based on extended **L-System** production rules and grammar with positional information.
- Created a user-friendly GUI using the **Qt** framework, enabling convenient configuration and visualization of the parameters.

GPU Sound Propagation Simulator

- Built GPUVerb, a Unity plugin for real-time acoustics in dynamic scenes.
- Implemented a **GPU Finite-Difference Time-Domain (FDTD)** solver using **Unity compute shaders**, simulating 2D wave propagation on a listener-aligned slice of scene geometry as an efficient approximation of 3D acoustics.
- Optimized GPU scheduling by spreading FDTD work across simulation frames (e.g., 10 Hz sim step) to reduce periodic GPU utilization spikes and avoid starving rendering workloads.

CUDA Path Tracer and Denoiser

- Implemented a Monte-Carlo Path Tracer using **CUDA C++** with .obj Mesh loading and texture mapping.
- Implemented an edge avoiding denoiser for high-quality, low-SPP denoising using G-Buffer data.
- Achieved a 10x speed-up for triangle intersection tests in complex scenes using GPU-based octree and stream compaction.
- Utilizing **NVIDIA Nsight** kernel profiler, conducted a thorough performance analysis to catch significant factors that impact the running time of the path tracer.