GPU Programming Final Project Pitch

Xuecheng Sun, Xuanyi Zhou, Jiarui Yan

Project Name: Vulkan Implementation of ReSTIR

Project Goals and outcomes:

For this project, we are going to implement a recent research paper, '*Spatiotemporal Reservoir Resampling for Real-Time Ray Tracing with Dynamic Direct Lighting*', that enables the usage of a large number of point and surface lights by combining various sampling techniques. Our implementation will use the Vulkan API. Since the algorithm uses ray tracing for visibility testing, we will explore using the Vulkan extension VK_KHR_ray_tracing for ray tracing. In addition, since this extension leverages hardware raytracing, we will also use Vulkan's compute shader to implement a software ray tracer, and compare the differences between them to evaluate the benefits of hardware ray tracing. We will also be implementing support for various material types.

(Changed Part)

Why this project matters:

This project gives us a good opportunity to explore the cutting-edge APIs, hardware ray tracing and recent researches in real-time rendering. This project will also demonstrate the speedup that hardware ray tracing can bring.

Platforms and APIs:

Platforms: Windows

APIs: Vulkan, GLM, GLTF.

Schedule:

(The delay is caused by the fact that we find out [2], [3] and [4] are not good materials and code base for us. Therefore, we need to revisit [6] for our development. [2], [3] and [4] can give us raytracer + denoiser for free, but they contain too many functions that we don't need and we are not able to exclude. Therefore, we decided to visit them in another way.)

Milestone 1 – Nov 18:

- Basic hardware and software Vulkan path tracer.
- Basic diffusive and specular material.
- GLTF scene loader.

(Changed Part)

GBuffer/Rasterization renderer.

GLTF scene loader (Basic Geometry, Node transformation – pos, normal, indices)

GLTF PBR support for rasterization rendering (If everything works well)

Milestone 2 – Nov 30:

- Biased ReSTIR algorithm.

(Changed Part)

GLTF PBR support for rasterization rendering

(I believe this can be done here. It will support Raytracing as needed)

Milestone 3 – Dec 7:

- Unbiased ReSTIR algorithm.
- More materials such as the Disney Principled BRDF.

(Changed Part)

Final – Dec 13:

- Testing, performance analysis and debugging.
- Preparing for the final presentation.

(Changed Part)

Reference and code base:

[1]: https://cs.dartmouth.edu/~wjarosz/publications/bitterli20spatiotemporal.html

[2]: https://nvpro-samples.github.io/vk_raytracing_tutorial_KHR/

[3]: https://github.com/nvpro-samples/shared_sources

[4]: https://github.com/nvpro-samples/shared_external

[5]: https://github.com/nvpro-samples/build_all

[6]: https://vulkan-tutorial.com/ (Base Code)