15-418 Final Project Milestone Report

Ray Tracing Parallelization

Jonathan Ke & Kavya Tummalapalli

[Project Web Page]

Initial Schedule

Week 1: 11/7 - 11/13

- ☑ ~~Research existing parallelization methods for path tracing using SIMD, CUDA, and OpenMP.~~
- ☑ ~~Test initial sequential ray tracing algorithm for spheres and create some initial benchmark tests that will work with the initial algorithm.~~
- ☑ ~~Begin parallelization implementation using CUDA.~~

Week 2: 11/14 - 11/20

- ☐ Finish parallelization implementation using CUDA.
- ☑ ~~Begin parallelization implementation using OpenMP and SIMD intrinsics.~~

Week 3: 11/21 - 11/27

- ☐ Finish parallelization implementation using OpenMP and SIMD intrinsics.
- ☐ Measure speedup and compare performance for two parallel implementations against sequential and each other.
- ☐ Implement more features and functionality to sequential ray tracing algorithms noted in the **Hope to Achieve** section.
- ☑ ~~Begin working on the milestone report.~~

Week 4: 11/28 - 12/4

- ☑ ~~Finish milestone report. (*due* **Wednesday, November 30th, 9:00am**)~~
- ☐ Parallelize code using CUDA and OpenMP + SIMD intrinsics on new features.

Week 5: 12/5 - 12/11

- ☐ Buffer time for weeks 1-4.
- ☐ Begin working on the poster, website, and writeup.

Week 6: 12/12 - 12/18

- ☐ Clean up code.
- ☐ Finish poster, website, and writeup. (*due* ***Saturday, December 17th, 11:59pm***)
- ☐ Present poster. (***Sunday, December 18th, 1:00-4:00pm***)

Revised Schedule

Week 4.2: 11/30 - 12/3

- ☐ Jonathan - Finish parallelization implementation using CUDA.
- ☐ Kavya - Finish parallelization implementation using OpenMP and SIMD intrinsics.
- ☐ Jonathan & Kavya - Finish implementing testing suite, including correctness checks and timing code.

Week 5.1: 12/4 - 12/6

- ☐ Jonathan & Kavya - Measure speedup and compare performance for two parallel implementations against sequential and each other.
- ☐ Jonathan - Further parallelize code using CUDA based on initial performance analysis.
- ☐ Kavya - Further parallelize code using OpenMP + SIMD intrinsics based on initial performance analysis.

Week 5.2: 12/7 - 12/10

- ☐ Jonathan & Kavya - Make sure all features we desire are supported and parallelized.
- ☐ Jonathan & Kavya - Complete further performance analysis and render live ray-traced animations.
- ☐ Jonathan & Kavya - Begin working on the poster, website, and writeup.

Week 6.1: 12/10 - 12/12

- ☐ Jonathan & Kavya - Clean up code.
- ☐ Jonathan & Kavya - Continue working on the poster, website, and writeup.
- ☐ Buffer time for weeks 1-5.

Week 6.2: 12/13 - 12/17

- ☐ Jonathan & Kavya - Finish poster, website, and writeup. (*due* ***Saturday, December 17th, 11:59pm***)
- ☐ Jonathan & Kavya - Present poster. (***Sunday, December 18th, 1:00-4:00pm***)

Work Completed

We have begun implementations for parallelizing using CUDA and using OMP + SIMD intrinsics. We also have all of our test cases and testing rig setup that our implementations can hook into for testing and benchmarking. This includes tuning the test cases to be easy to display for demonstration purposes. Now we can just focus on the parallel implementations and any auxiliary testing and benchmarking should be easy to insert.

Goals and Deliverables

We are slightly behind schedule from our original goals but still anticipate finishing the primary goals in our schedule. This will include detailed performance measurements on the two parallel implementations against the sequential implementation. However, we likely cannot reoptimize the parallel implementations for any additional features we add to the system. For example, we likely cannot go through a second iteration of parallel optimizations over the “nice to haves” listed which includes more complicated objects, surfaces, scenes, and moving scenes. Also, we may also not hit the hard 20 FPS performance target listed in the “nice to have” section and will only be aiming to gain a strong scaling speedup on our parallel implementations instead. Some difficulties impacting our delays include not having access to GPU drivers over the past two weeks on the GHC machines and difficulty modifying the starter code to run our test cases.

Our revised listed goals:

- Implement a CUDA GPU parallel implementation
- Implement a OMP + SIMD parallel implementation
- Support 3D rendering from all angles
- Support reflection and refraction
- Speedup and performance analysis of parallel implementations
- Demonstrate live ray-traced animation of circular movement

Poster Demo

- Performance results (graphs + tables)
- Images of some of the renderings
- 360° live video rendering with all three implementations

Issues and Concerns

We primarily need to finish the coding work and get to testing and benchmarking as soon as possible.