ECE451/566 Final Project Report

Electrical and Computer Engineering Department
Rutgers University, Piscataway, NJ 08854

Gravity Simulator

CUDA-BASED SIMULATION AND RAY TRACING RENDERING

Submitted by:
Shizhe Yang , Olisadebe Ojukwu

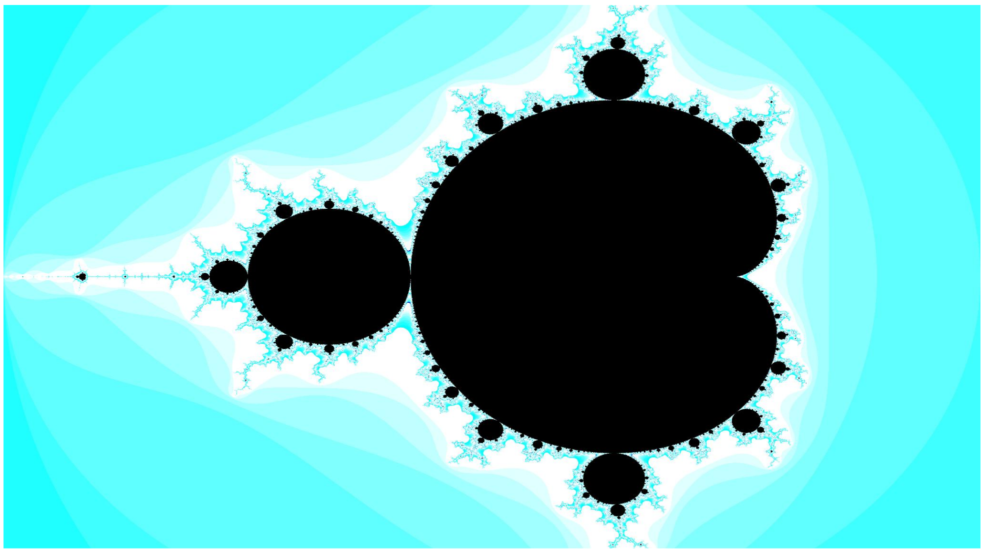
Advisor[s]:
Dov Kruger

Dec 10th,2025

Electrical and Computer Engineering Department
Rutgers University, Piscataway, NJ 08854

1. Introduction

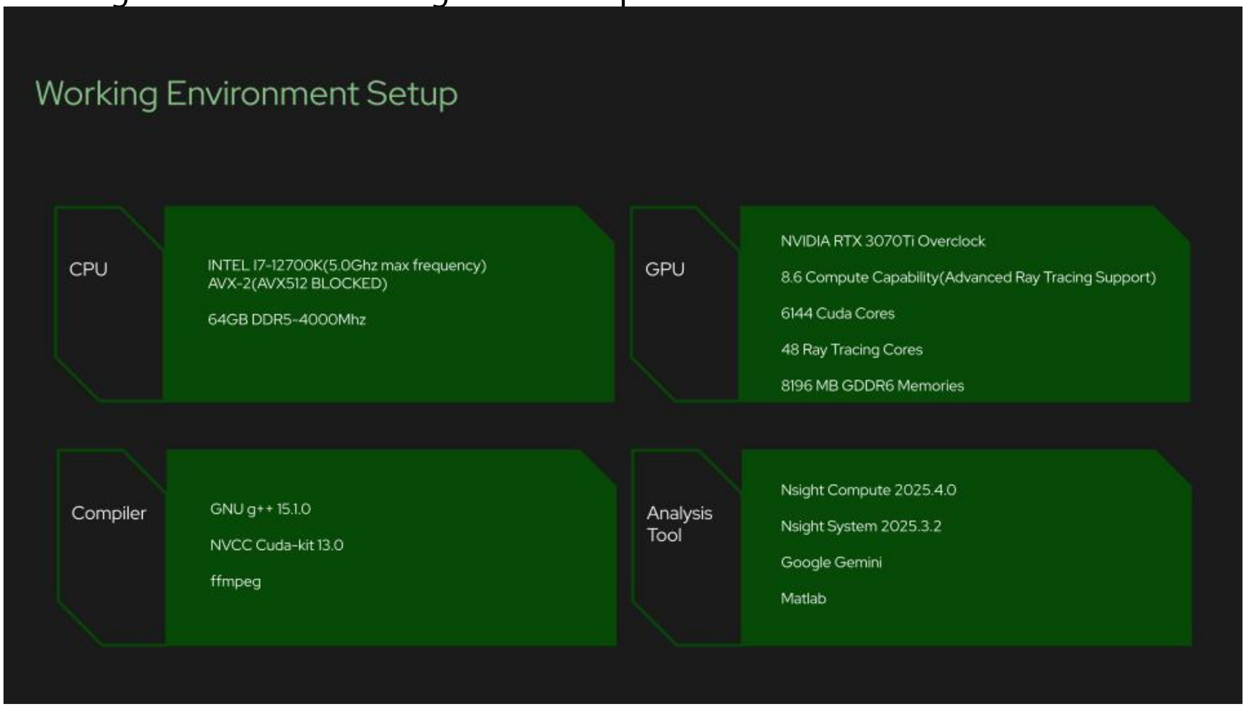
We had the idea of completing this project using CUDA during the process of finishing Mandelbrot homework. Mandelbrot is a program that immensely use the potential of GPU architecture as GPU allows massive parallel computing.



By using GPU acceleration, we are able to achieve much better performance for mandelbrot iterations than using CPU. Hence, it gives us the idea of using CUDA on gravity simulation. Our initial target is to built a naive brute-force simulator using cpp, and then convert it to CUDA codes. After the simulator compute a file containing all position data, we will use another CUDA file to generate frames using the data and render those frames using Ray Tracing techniques.

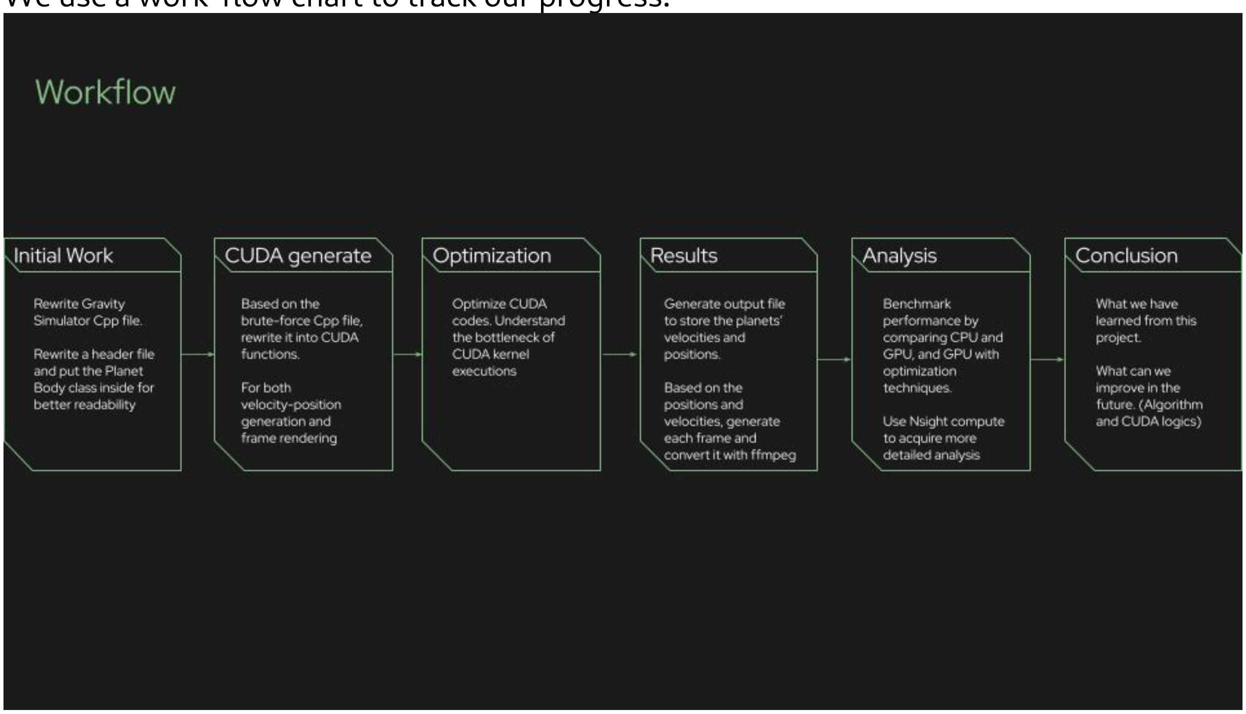
2. Methods / Optimization Process / Results

we setup our working environment with specific tools,compilers and use high end consumer grade computers for simulations.



2.1. Methods

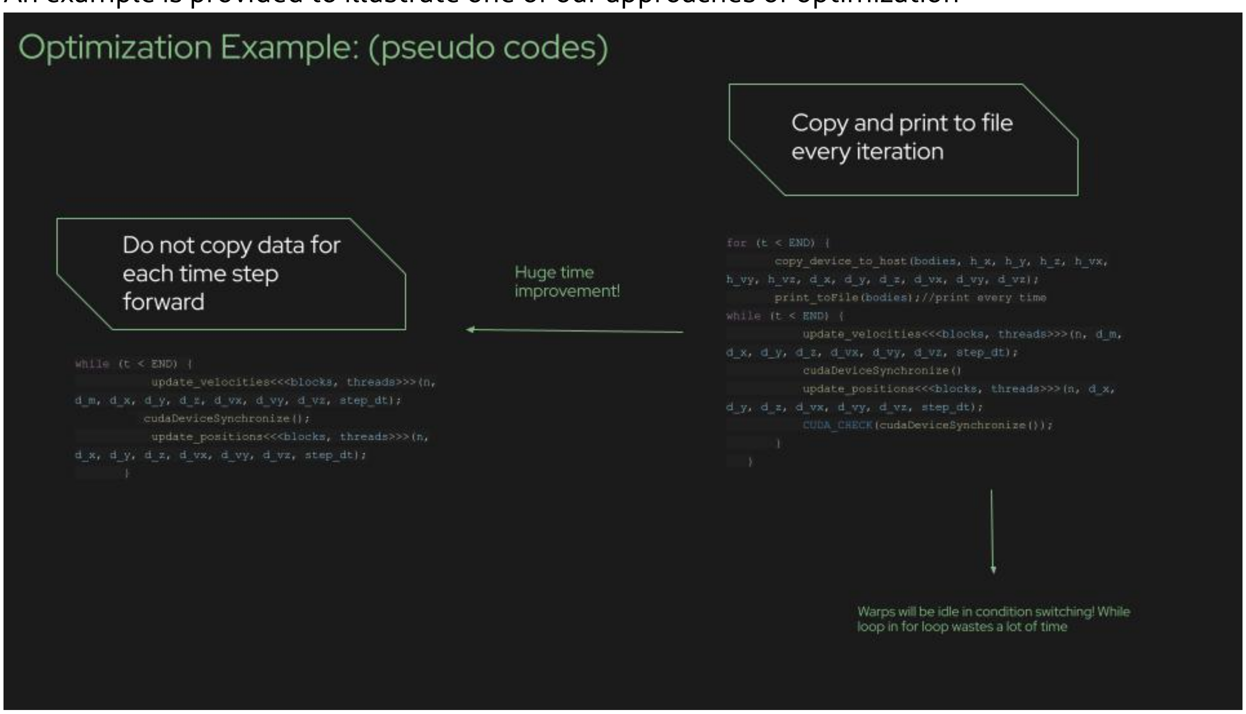
We use a work-flow chart to track our progress.



2.2. Optimization Process

While doing our project, we follow a guideline to optimize our codes.

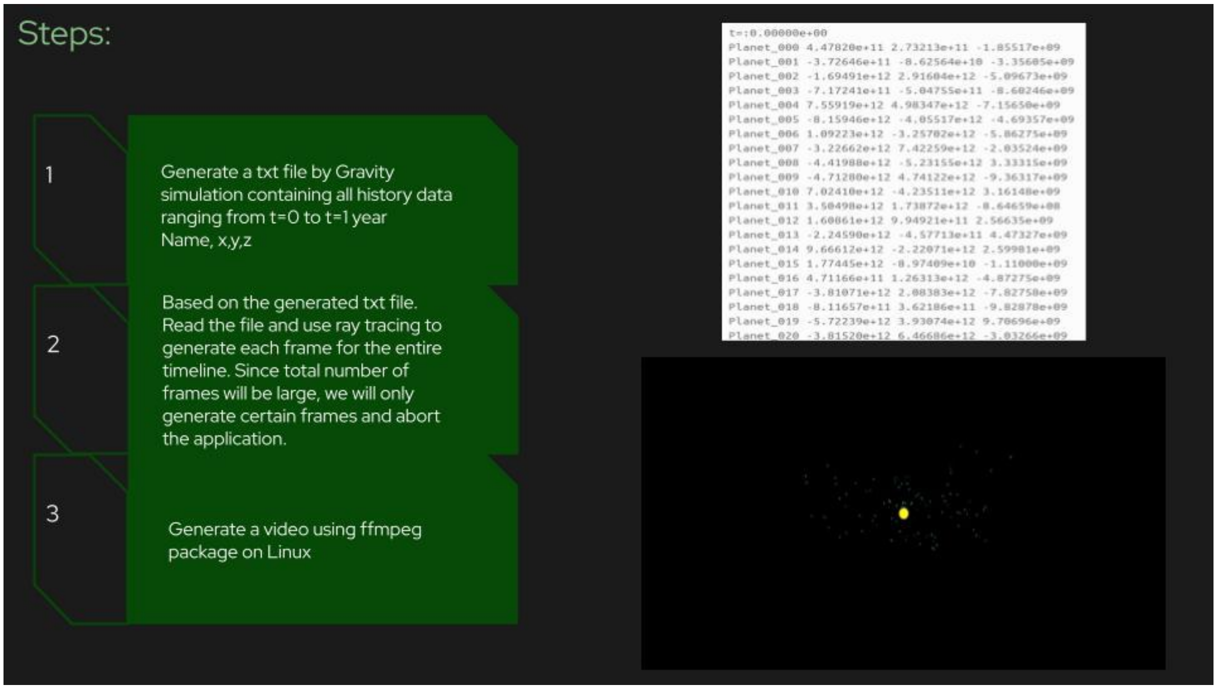
An example is provided to illustrate one of our approaches of optimization



2.3. Results

Our program separately compute a txt file containing all position data w.r.t each planet.

By using those data, the rendering program use ray tracing to render multiple frames, and we use ffmpeg package in Linux to convert all the frames into a video



A link is provided for readers to see the final video:

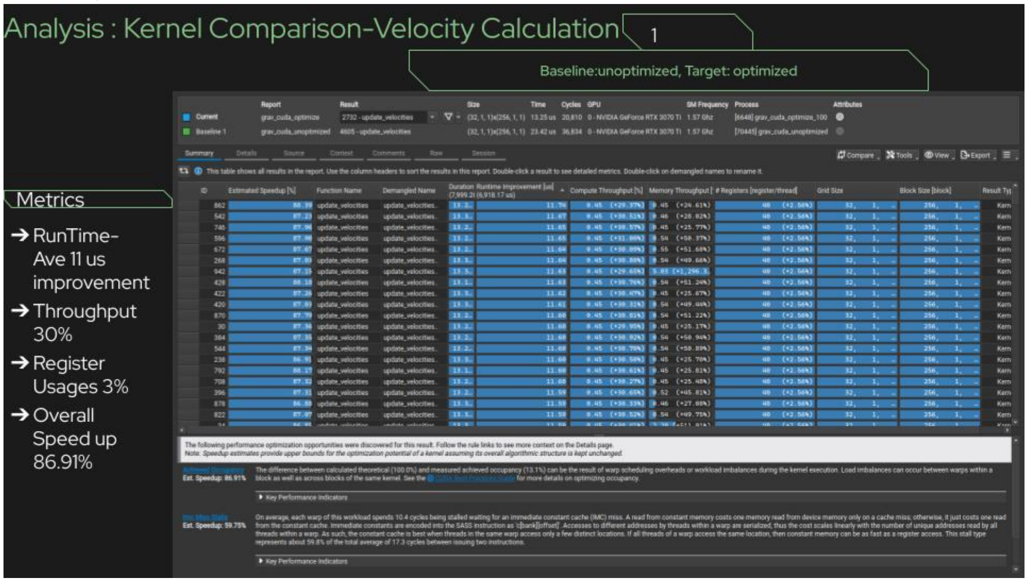
https://drive.google.com/file/d/150FLoCXtmvFSEA8-obTG3o-hkJqxsGqG/view?usp=sharing

3. Analysis

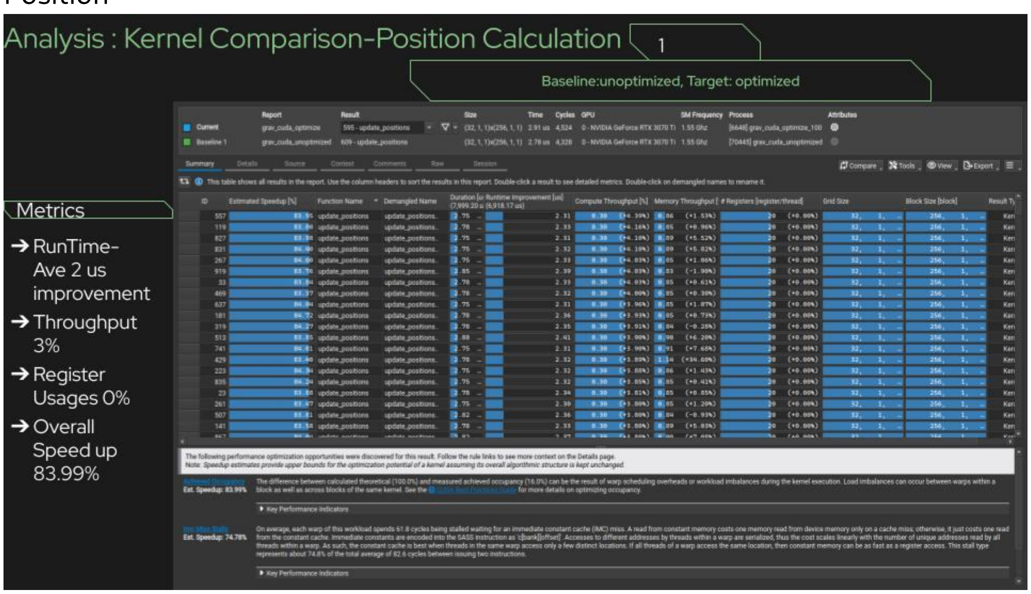
We use Nvidia Nsight Compute tool to analyze our program performance.
We compare the optimized version to unoptimized one to check the program speed up.

3.1 Gravity Simulator Version Comparison

Velocity



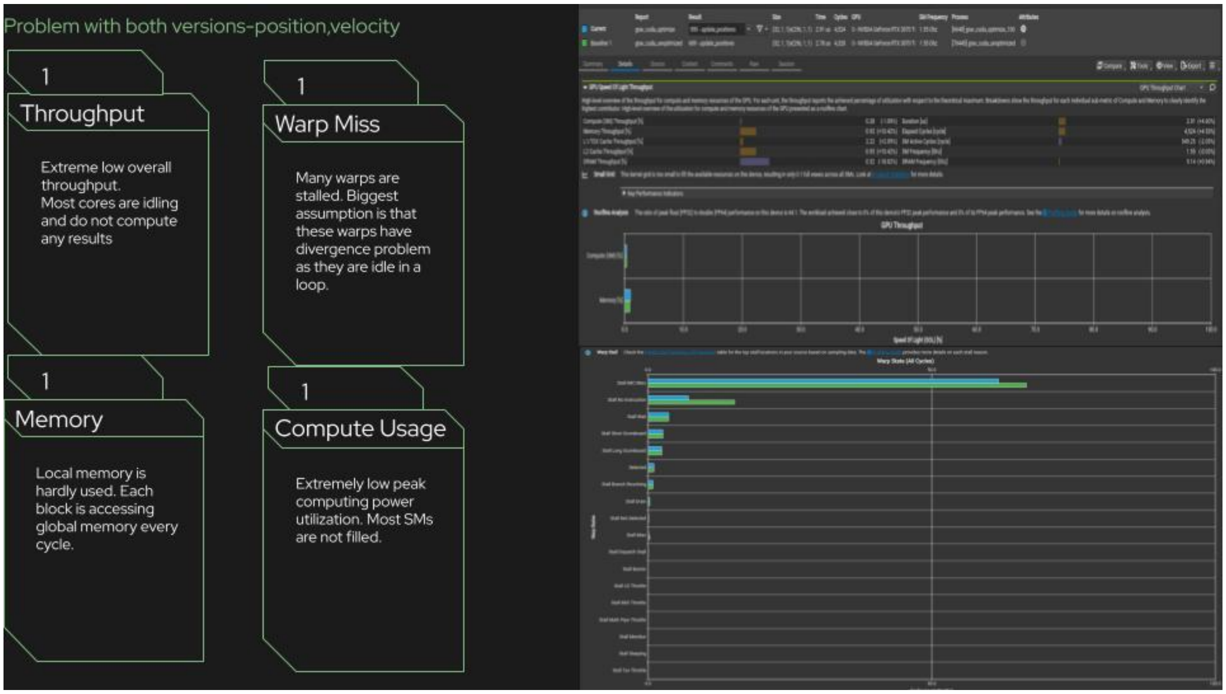
Position



We find out that by optimizing the data writing algorithm, we achieved phenomenon speed ups.

3.1.1 Problems

a)We discover that we are not using the full potential of SMs, and our memory latency is slowing down the program. The warp divergence problem also causes latency.



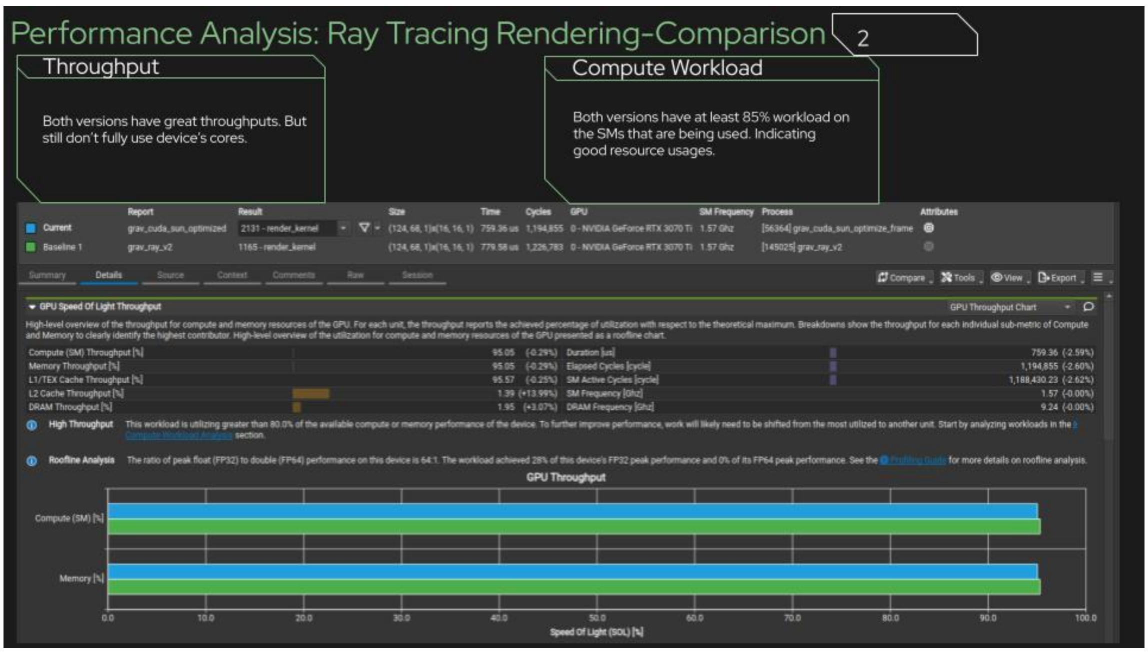
b)Therefore, we decide to saturate the SMs and memories. Instead of iterating through 100 planets, we planet to iterate 100,000 planets and maximimze the memory allocation to be around 6000MB, which is the maximum that the GPU can allocate without halting the system.

We discovered that when memory is saturated and we have enough data to feed the kernels, the over GPU throughput increased enourmously! Although it still does not reach the full throughput potential, but now we are confident that our program is heading to the right direction.

Furthermore,stats show that the nearly 100% warps are active. This indicates that our program does not have divergence problem, instead, warps and blocks are just waiting for previous instructions to finish in order to execute next instructions. Our guess is that it's because of the data dependecy between 2 kernels.

3.2 Ray Tracing Rendering

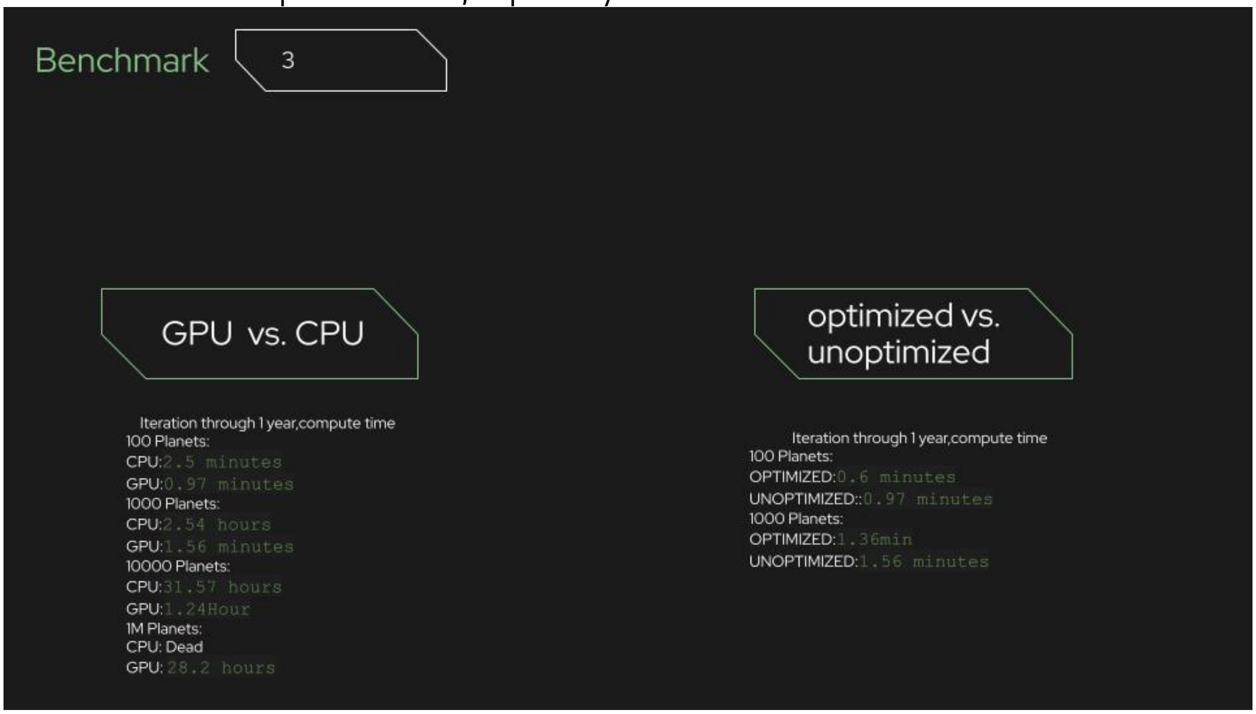
Two versions evidently perform well creating enough throughput



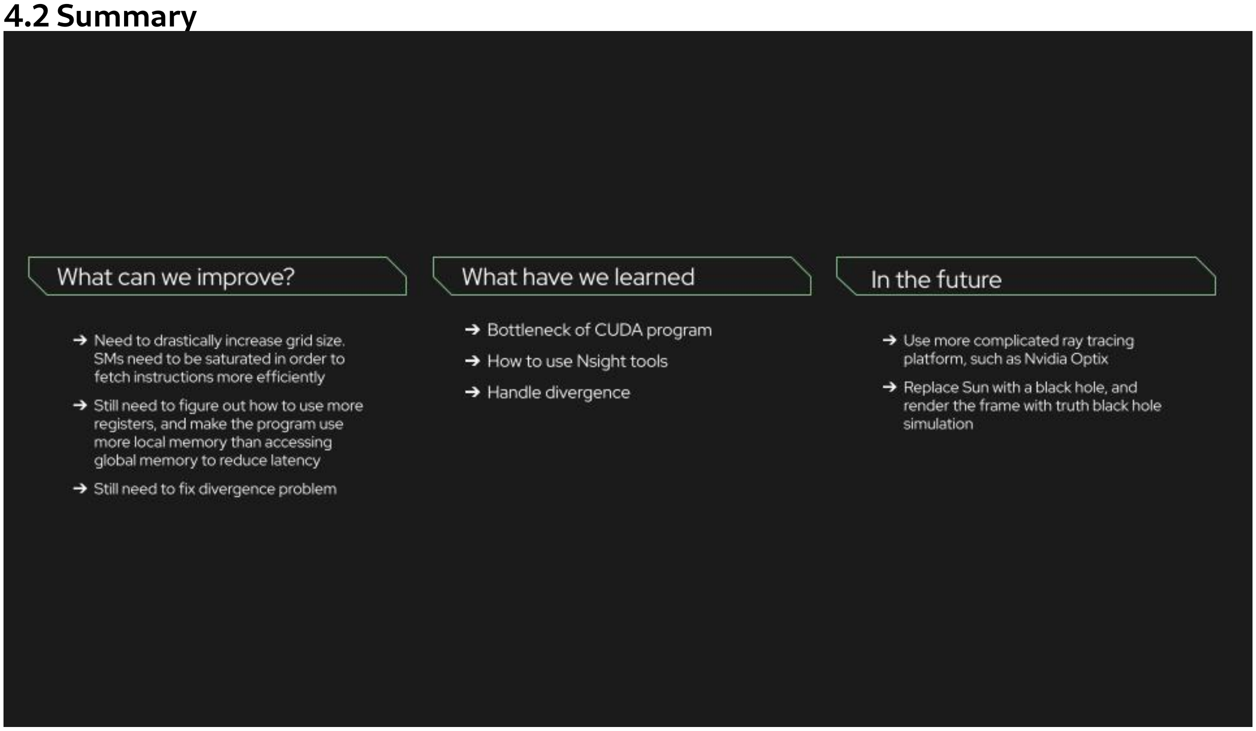
4. Conclusions / Summary

4.1 Conclusion

We benchmark the performance, especially between GPU and CPU



4.2 Summary



We learned that in order to fully utilize the potential of GPU parallel compute capabilities, we need to feed enough data into GPU and keep all GPU units busy. It is better for GPU units to wait for next cycle to begin processing than idling.

5. Acknowledgments

We appreciate the guidance by Professor Kruger.

6. Github Repo

A link to github repo is attached here:
https://github.com/Sz-Yang-rutgers/ECE451_PROJECT_TEAM_CUDA

7. REFERENCES