29th January 2021

Project Proposal – Using GPU Computing to model Mars subsurface ice detection with polarimetric synthetic aperture radar

Author: Jack Hanlon, *University of Victoria*
Supervisors: Bill Bird, *University of Victoria*. Etienne Boulais, *Canadian Space Agency*

1. Introduction

This project concept was derived from ongoing research at the Canadian Space Agency by Dr. Etienne Boulais in which discovering Martian subsurface ice has been deemed valuable in assessing the landing sites for future Human colonization.

"Numerous studies have suggested that polarimetric orbital synthetic aperture radar (SAR) platforms would be capable of identifying accessible water resources necessary for the onset of Mars colonization" (Boulais. E., 1). A preemptive understanding of these orbital SAR systems could be simulated in addition to providing tools to visualize the collected images. Recently, the CSA developed a prototype Monte-Carlo computational model that shows promising results for sufficiently describing electromagnetic wave propagation under Mars' surface. "However, using conventional CPU-based parallel computing approaches, the current code performance is still prohibitive in regard of the large parameter space associated to Mars soil and large illuminated area, limiting radar image interpretation and instrument development" (Boulais. E., 1).

Therefore, this project aims to explore the use of GPU-based computing to increase the computational speed of the CSA radar propagation model. "Based on recent literature, similar problems have shown improvements of orders of magnitude in computational speed, which could be of great interest" (Boulais. E., 1) to the Canadian Space Agency if transferrable into their current model. As a proof of concept, specific bottleneck portions of the radar propagation algorithm will be ported to GPU using various GPU computing wrappers and their runtimes compared to their CPU counterparts. The end of the project will entail an estimation on potential computational speedup, and a recommendation will be formulated that may induce action on specific hardware selection in an operational setting.

2. Statement of Problem

Analyzing Mars' surface requires vast amounts of data, thus optimizing the radar propagation algorithm, through software and hardware speedups, is fundamental in detecting subsurface ice. The overarching problem is how to effectively speedup an existing CPU-based radar propagation model using GPGPU computing acceleration. The specific problem that will be tackled is how to test ported bottleneck subsections of the radar propagation algorithm in a GPU to prove a speedup will occur.

3. Review of Relevant Literature

This stands as a guide to what the current relevant understanding is on GPU-based Monte Carlo Algorithms that utilize CUDA. "FullMonteCuda: a fast, flexible, and accurate GPU-accelerated Monte Carlo Simulator for light propagation in turbid media" [2] by Young-Schultz, T., et al. is an algorithm written by members of the University of Toronto Department of Electrical & Computer Engineering, that found a speedup of 288-936x through various GPU optimizations, over a single threaded, non-vectorized version of their Monte Carlo Simulator (FullMonteSW). It is a promising result as it shows that a similar algorithm may be created that could follow the same fundamentals and be transferred to the Canadian Space Agency model.

The paper discusses the application of using light propagation in tissue for the purpose of photodynamic therapy (PDT) and bioluminescent imaging (BLI). To accomplish this goal, they were in a very similar position in that the best first step would be to simulate the light propagation in a "tissue-like" media. This could be done analytically by solving the Radiative Transport Equation, however at the boundaries of materials this becomes significantly more difficult. Thus, a numerical approximation using the Monte-Carlo method was adopted. The initially developed simulation encapsulated multi-threading to reduce computational cost when attempting high resolution simulations. They further enhanced this by manually hand-vectorizing instructions, but this leads to a drawback in reduction of readability and portability of the code to other projects. After this was noted, they realized further optimizations led to diminishing returns in performance gains, therefore GPU's should be explored. FullMonteCUDA is the open source project that they validated and benchmarked for NVIDIA GPU's.

The FullMonteCUDA implementation overview is illustrated in Figure 1, showing the CPU host logic of the algorithm:

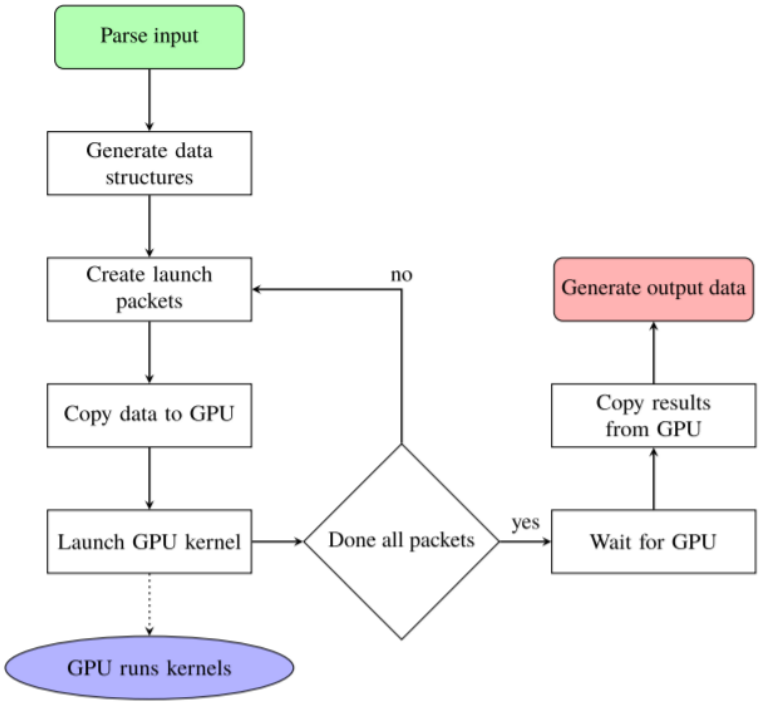

Figure 1: The CPU host logic of FullMonteCuda (Young-Schultz, T., et.al., 4717)

Once the kernels have finished executing in the GPU, the CPU host copies the output data from the memory of the GPU into its own memory and generates the output files. A brief outline of speedups they were able to achieve with various hardware manipulations is in the Table below:

Optimization	Incremental Speedup
Naive	2x
CUDA vector datatypes and math operations	2.5x
Materials constant cache	1.6x
Thread local accumulation cache	1.3x

Table 1: Performance increase for each FullMonteCUDA optimization over FullMonteSW (Young-Schultz, T., et.al., 4718)

An important item to note, is that NVIDIA Visual Profiler was used to identify bottlenecks in the code and to accurately measure the impact of optimizations. It is an item that will be investigated, for this project as it seems to fit the specific objective.

To conclude, the results of "FullMonteCuda: …" showed 11x speedup over MCtet, 2x over CUDAMCML, 12x over MMCM, 19x over TIM-OS, and 288 – 936x over the initial single-threaded non-vectorized version of FullMonteSW. "FullMonteCUDA's performance improvement over the highly optimized software code significantly improves its ability to be used in solving the inverse problem for biophotonic procedures like PDT and BLI." (Young-Schultz, T., et.al., 4724). The time

scale difference in operational runtime reduces hours to matters of minutes or allowing for significantly more accuracy in the same time scale. It is stated that FullMonteCUDA could be extended to applications outside of biophotonics as it has been written with the intention of making it simple to use and extend.

Another relevant paper is “GPU accelerated electric field Monte Carlo simulation of light propagation in turbid media using a finite beam size model”, by Wang, Y., et. al. which discusses how their algorithm was sped up by CUDA on a GPU by up to 370x. This will be relevant to this project as it also pertains to light propagation in turbid media.

In this paper they focus on Electric Field Monte Carlo simulations (EMC), which directly trace complex electric field vectors in multiple scattering and estimate the preferred direction of the electric field. They use this method specifically to simulate the coherence and polarization of the light. By launching large quantities of photons, the numerical method becomes time-consuming. So, to counteract this, GPU computing was used to increase efficiency on a Fermi architecture.

4. Statement of Solution

To effectively speedup the CSA radar Monte Carlo Model, specific bottleneck portions of the code/pseudocode are to be extracted and evaluated in a GPU-based context. Various parameters will be tested using pyCUDA to assess the validity and methodology to transferring these components to this GPU optimization in an operational setting.

5. References

[1] Wang, Y., et al. (2012). "GPU accelerated electric field Monte Carlo simulation of light propagation in turbid media using a finite-size beam model." Optics express **20**(15): 16618-16630.

[2] Young-Schultz, T., et al. (2019). "FullMonteCUDA: a fast, flexible, and accurate GPU-accelerated Monte Carlo simulator for light propagation in turbid media." Biomedical optics express **10**(9): 4711-4726.

[3] Boulais, Etienne. *Interdisciplinary Research Project*, *Project_description_Jack_Hanlon.docx*, 18 Jan. 2021.