Shooting-Bouncing-Rays Technique to Model Mine Tunnels: Algorithm Acceleration

Stephen Kasdorf, Blake Troksa, Jake Harmon, Cam Key, and Branislav M. Notaroš
Colorado State University, Electrical & Computer Engineering Department, Fort Collins, CO
skasdorf@rams.colostate.edu, blake.troksa@gmail.com, J.Harmon@colostate.edu, camkey@rams.colostate.edu, notaros@colostate.edu

***Abstract*—We present and discuss acceleration of a shooting and bouncing rays (SBR) algorithm for ray-tracing electromagnetic analysis of electrically very large structures such as underground mine tunnels at modern wireless communication frequencies. The acceleration is based on the parallelization of the SBR technique on NVIDIA GPUs using the OptiX application programming interface. The results show dramatic speedups of the parallel SBR algorithm compared with serial implementation.**

Keywords—wireless communications, computational electromagnetics, high-frequency techniques, ray tracing, high performance computing, parallelization, graphics processing units, GPU acceleration, signal propagation, waveguides.

I. Introduction

In the field of computational electromagnetics (CEM), the necessity of acceleration of simulation techniques is becoming more and more apparent. At high frequencies especially, acceleration of CEM algorithms becomes vital for practicality of CEM solutions. Ray tracing (RT) [1]–[4] is an asymptotic high-frequency CEM methodology that demonstrates significant potential to efficiently characterize extremely large structures with computation times that are orders of magnitude shorter when compared to traditional full-wave CEM techniques, such as the finite element method and method of moments. When coupled with high-performance computing (HPC) strategies such as general-purpose computing on graphics processing units (GPGPU), simulations which would otherwise requires days or weeks can be condensed into minutes or hours. Moreover, with proper memory allocation and management, even low-end computing hardware can be leveraged for extremely rapid electromagnetic simulations and post-processing.

This paper presents and discusses acceleration of a CEM algorithm based on shooting and bouncing rays (SBR) method for RT analysis of electrically very large structures such as underground mine tunnels at modern wireless communication frequencies. It highlights the parallelization of the SBR technique on NVIDIA GPUs using the OptiX application programming interface.

II. Acceleration of SBR Ray-Tracing Algorithm for CEM Modeling

The SBR ray-tracing algorithm [3], [4] involves spawning, and propagating millions of rays. These rays are traced geometrically through the environment, and the electric field is tracked as this propagation occurs. The memory and time requirements of tracking the information of each ray can be substantial on a CPU when implemented in series. The field contribution from each ray is linear, and thus ray tracing algorithms are perfect candidates for parallel computations on a GPU. Geometric ray propagation calculations are independent from one another. This means that rays can be traced through the environment in parallel without the need for synchronization. Additionally, the electric field at observation points is linear, meaning that the electric field calculations can be carried out independently per ray.

We perform parallelization and acceleration of our SBR RT algorithm using the compute unified device architecture (CUDA) parallel computing platform as the primary language for parallel programming on NVIDIA GPUs. The GPUs are used as a tool to parallelize the core ray-tracing algorithm and also to provide access to the NVIDIA OptiX ray-tracing application programming interface (API) [5]. The OptiX API efficiently traces rays within complex structures. Using OptiX, we are able to generate information about the closest hit point of a ray with the geometric environment and use this information to reflect rays and adjust their power according to Fresnel's coefficients. The combination of OptiX and CUDA enables quick and efficient simulations to optimize the placement of communication nodes within the structures under consideration.

The memory requirements of the ray tracing can still be cumbersome due to the large number of rays and information associated with the simulation. The problem is alleviated by breaking up the ray spawning into batches. Rays are launched by inscribing an icosahedron into a unit sphere. The faces of the icosahedron are subdivided according to the *n*th triangular number so that the desired number of rays is achieved. The *n*th triangular subdivision of a face of the icosahedron is shown in Figure 1.

This helps to ensure an even distribution of rays used to cover the radiation pattern. The icosahedron also proves useful for batching. The subdivided rays on each face of the icosahedron make up each batch of the algorithm. Again the geometric and electric field ray contributions are independent, so splitting the algorithm into batches is not problematic. The algorithm is performed on each batch individually, and the electric field contribution of this batch at the observation

This work was supported by the National Science Foundation under grant ECCS-1646562.

978-1-7335096-0-2 ©2020 ACES

points is recorded. Once the batch has finished, the memory from these rays can be cleared before moving to the next. This alleviates the memory bottleneck associated with launching millions of rays simultaneously. If enough rays are launched such that these batches also experience bottlenecks, they can be subdivided into sub-batches and the memory bottleneck is again alleviated. This process can be used to launch and trace extremely large numbers of rays quickly and efficiently.

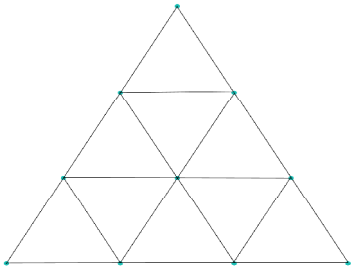

Fig. 1. Subdivision of the icosahedron face into the *n*th triangular number *(n* = 4). The face is used as a batch of rays, where $N = n(n+1)/2$.

This helps to ensure an even distribution of rays used to cover the radiation pattern. The icosahedron also proves useful for batching. The subdivided rays on each face of the icosahedron make up each batch of the algorithm. Again the geometric and electric field ray contributions are independent, so splitting the algorithm into batches is not problematic. The algorithm is performed on each batch individually, and the electric field contribution of this batch at the observation points is recorded. Once the batch has finished, the memory from these rays can be cleared before moving to the next. This alleviates the memory bottleneck associated with launching millions of rays simultaneously. If enough rays are launched such that these batches also experience bottlenecks, they can be subdivided into sub-batches and the memory bottleneck is again alleviated. This process can be used to launch and trace extremely large numbers of rays quickly and efficiently.

III. Results and Discussion

The acceleration of this method is tested by comparing run time of the accelerated parallel algorithm against a serial CPU implementation. The algorithm steps remain the same, so the comparison gives a true comparison of the speedup. Table 1 shows the results of this comparison. The results are generated from a tunnel environment detailed in [6].

When launching 10,000 rays we see a speedup of 12.36, which is not as substantial. Due to the fairly significant fixed cost associated with GPU usage, the performance of the parallelized code scales rapidly with increasing number of rays relative to the serial realization of the code. With a high ray count, we expect to see extremely high speedup because the algorithm can fully take advantage of the threads in the GPU. Indeed, at one million rays and 100 million rays, respectively, we see a speedup of over 130 times.

Table 1: Speedup of the parallel SBR algorithm compared with serial implementation for a RT simulation of a tunnel environment.

Number of rays	Number of reflections	Sequential code	Parallelized code	Speedup
10,000	20	4.251 s	0.344 s	12.36 x
1,000,000	80	24 m 40.096 s	11.125 s	133.04 x
100,000,000	40	21 h 14 m 17 s	9 m 46 s	130.47 x

IV. Conclusions

This paper has presented acceleration of an SBR RT algorithm for CEM analysis based on the parallelization of the SBR technique on NVIDIA GPUs using the OptiX application programming interface. Accurate simulations in electrically large and complex geometric environments, such as underground mine tunnels and galleries in wireless communication applications, will certainly necessitate the use of extremely large numbers of rays to adequately sample the environment. Based on the results presented in this paper, the presented algorithm proves to be very beneficial for these types of simulations.

References

[1] M. F. Cátedra and J. Perez, Cell Planning for Wireless Communications. Norwood, MA, USA: Artech House, 1999.

[2] Z. Yun, M. F. Iskander and Z. Zhang, "Development of a new shooting-and-bouncing ray (SBR) tracing method that avoids ray double counting," *IEEE Antennas and Propagation Society International Symposium. 2001 Digest.* Boston, MA, USA, 2001, pp. 464-467.

[3] B. Troksa, C. Key, F. Kunkel, S. V. Savic, M. M. Ilic, and B. M. Notaros, "Ray Tracing Using Shooting-Bouncing Technique to Model Mine Tunnels: Theory and Verification for a PEC Waveguide," *Proceedings of the 2018 International Applied Computational Electromagnetics Society (ACES) Symposium – ACES2018*, March 25–29, 2018, Denver, Colorado, USA.

[4] C. Key, B. Troksa, F. Kunkel, S. V. Savić, M. M. Ilic and B. M. Notaroš, "Comparison of Three Sampling Methods for Shooting-Bouncing Ray Tracing Using a simple Waveguide Model," *2018 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting*, Boston, MA, 2018, pp. 2273-2274.

[5] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich, "OptiX: a general purpose ray tracing engine," *ACM Trans. Graph.*, Vol. 29, No. 4, Article 66, July 2010.

[6] D. Didascalou, "Ray Optical Wave Propagation Modelling in Arbitrarily Shaped Tunnels", *Forschungsberichte aus dem Institut für Höchstfrequenztechnik und Elektronik der Universität Karlsruhe*, Vol. 24, Institut für Höchstfrequenztechnik und Elektronik, Universität Karlsruhe, 2000.