GP GPU Acceleration of a PO Based RF Simulation Software Dedicated to Radar Simulation in Large Scale and Complex Environments

A. Boudet, **N. Douchin**, **and P. Pitot**
OKTAL Synthetic Environment, 11, avenue du Lac, Vigoulet-Auzil 31 320, France

Abstract— SE-RAY-EM software is based on a combination of Shooting and Bouncing Rays (SBR) technique, that has been optimized to calculate efficiently the intersections between rays from a transmitter towards a 3D database and back to a receiving point, and EM models for computing propagation, reflection and diffraction. These models are the formulations of Geometrical Optics (GO), Physical Optics (PO) and Equivalent Current Method (ECM). An operating strategy enables unified calculation for the near or far EM scattered fields from the scenes. The "forward scattering" approach based on the equivalence principle is also used to compute EM fields in the shadow region. Since it relies on asymptotic methods SE RAY-EM is well suited for computing the EM interactions of an incident wave with complex 3D models of large scale environments and objects at high frequencies typically in the 1–100 GHz range.

Existing works have shown that GP GPU acceleration is well suited for ray tracing applications. EM asymptotic simulations based on ray tracing simulation can benefit from that technology, with the following difficulties.

Computation of phases requires double precision: though GPU double precision capabilities are improved at each GPU generation, double precision numbers cost also twice as much registers than float numbers and twice as much memory bandwidth. So avoiding double computations while preserving phase precision is necessary.

For physical optics simulations, high frequency computation needs very precise surface sampling. To be efficient, it is necessary to reduce the number of samples to compute. As on existing SE-RAY-EM CPU renderer, we choose to implement an adaptive anti-aliasing process that reduces the samples number while achieving the precision requirements. Adaptive anti-aliasing process on a highly parallel architecture is not obvious as the number of rays to trace can actually explode. It must be controlled without consuming too much memory and keeping a high level of parallelism.

In this paper, we discuss on the way to solve such problems. We also present the SE-RAY-EM computation scheme on its existing CPU version and compare it to the new GP GPU version. Results and comparison with the CPU version of SE-RAY-EM are presented.

1. INTRODUCTION

In this paper, we first present the SE-RAY-EM software tool and its principle to handle electromagnetic computations. Then we discuss its current implementation and present the issues involved in its optimisation using Graphic Processor Unit (GPU) in the frame of EM simulation. Then we present a novel solution using cones for beam tracing and an adaptive anti-aliasing algorithm suited for massively parallel computation. Finally, we present some results to illustrate the efficiency of the new implementation.

2. STATE OF THE ART

2.1. Principle

SE-RAY-EM is an electromagnetic simulation code based on ray tracing [1] and asymptotic methods [2]. These methods are less physically rigorous than "exact" methods that are strictly based on the resolution of Maxwell equations. However, asymptotic methods as they are used in SE-RAY-EM enable to handle complex scenes that are very large compared to the wavelength with enough accuracy. That is when the main contributions to the electromagnetic field are correctly represented.

In the case of complex targets, SE-RAY-EM gives results very similar to the "exact" method for a much lower computation time. Where 600 MHz is a high limit for "exact" solutions used on this type of objects, it is almost a low limit in terms of physical validity for asymptotic methods. The standard computation range addressed by SE-RAY-EM is more between 1 to 100 GHz on far more complex scenes (natural terrain on several kilometres with several complex targets).

In SE-RAY-EM, rays are traced from transmitters towards reception points. These rays are grouped four by four in beams. Rays are traced from transmitters through a grid (Figure 1). The intersections of theses beams are computed. They are two types of interactions (Figure 2):

- Geometrical Optics (GO) when the beam is reflected by a metallic or dielectric surface.
- Physical Optics (PO) towards the reception points at each interaction.

Using beams has three main interests. They enable to:

- Know the interaction contour of the incident wave with the surface, which is necessary for using PO.
- Detect edges.
- Detect the main aliasing cases: geometry, material, curvature of the surface.

As shown in Figure 3, the adaptive anti-aliasing mechanism of SE-RAY-EM consists in subdividing the computation grid as a "quatree". So when aliasing is detected on the path of a beam, it is subdivided in four new beams.

Adaptive anti-aliasing is very interesting for EM computations as:

- The size of the primary grid can be coarse as the result of computation is a signal and not an image. The only constraint is that the grid is thin enough to detect important elements of the scene.
- The accuracy needed for EM computation is linked to the wavelength and object boundaries must be sampled at a fraction of the wavelength. For example, at 100 GHz, sampled element size can be less than 1 mm for a target of several meters. Without adaptive anti-aliasing, the computation should use a 100000×100000 pixel grid!
- Adaptive anti-aliasing enables to reduce the number of contributors for which EM models are applied. These complex computations have a cost that cannot be neglected and can be done for a lot of wavelengths. Reducing the number of contributors is then inevitable to reduce computing times.

2.2. SE-RAY-EM Discussion

The current anti-aliasing approach can miss some details or some small objects (e.g., mast of a boat). Actually, beams are built from a punctual sampling. On one hand, this forces the user to

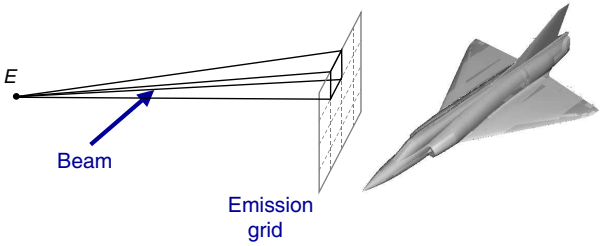

Figure 1: SE-RAY-EM emission grid.

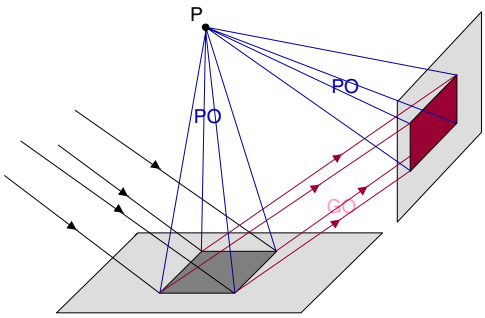

Figure 2: Principle of beam interactions.

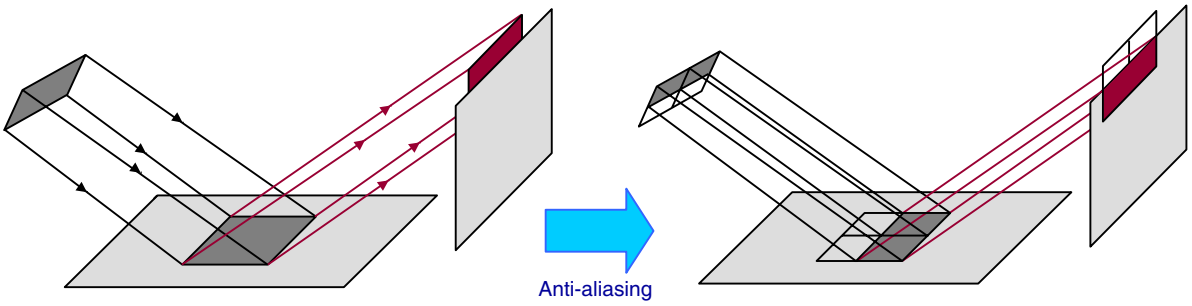

Figure 3: SE-RAY-EM adaptive anti-aliasing principle.

define a grid with a minimal resolution. On the other hand, the user has to use transparent faces (or wings) around small objects and edges in order to ensure their detection.

Another problem of the current anti-aliasing algorithm is that if the beam is aliased after several reflections, it is subdivided from the beginning. Even though the previous reflection levels are not aliased. The number of contributors can then be raised even if not necessary, and, more importantly, this can lead to unwanted artefacts when dealing with radar imagery cases.

3. MASSIVELY PARALLEL APPROACH

Nowadays, Graphic Processors Unit (GPU) have proven to be very efficient for optimising General Purpose (GP) computations, and particularly ray tracing applications [3] as SE-RAY-EM. However, the transition to GPU is not straightforward and several issues have to be taken into account.

3.1. Double versus Float

Historically, SE-RAY-EM performs all computations using double precision floating point operations (*double*). This permits to avoid naturally some problems of accuracy without any specific work. However, GPU are very efficient for single precision floating point (*float*) computation. Even though, *double* computations are now handled and tends to become faster, they should be used with parsimony. Also *double* data storage and transfer are more costly. Also *double* operations need twice as much registers as *float* ones. And registers are rare and precious when using GPUs. Our new implementation only uses *double* when it is absolutely necessary, mostly in the computations that involve the phase of the electromagnetic signal.

3.2. Object Instances and Moving Objects

SE-RAY-EM uses instances of objects in order to share geometries. Such objects can also reference underlying instances, which results in a hierarchical graph. A matrix defines the position of each instance. During raytracing, rays are transformed using these matrices in local space to perform intersection computation with the geometry. The hierarchical definition implies several transformations when a ray traverses the scene, which results in a lot of computation. This process has proven itself inefficient on the GPU. Then, we flattened the instance hierarchy before computation so that a ray is only transformed one time during ray tracing. This enables to handle both instances and moving object on the GPU with a small impact on performances.

3.3. Cone tracing

Instead of tracing individual rays, we use cones to trace beams. A cone will detect any geometric element in its volume. This way anti-aliasing is more reliable and small object and edges are detected directly without the need of any transparent wings. Besides the initial resolution can be really coarse. The only reason to increase the resolution will then be imposed if the GPU cannot be loaded enough.

Another advantage of using cones is that they can be processed independently of their neighbours, which is very interesting for parallelization. It is also simple to handle the generated contributors independently from one reflection level to another.

3.4. Adaptive Anti-aliasing

The difficulty when implementing adaptive anti-aliasing on GPUs consists essentially in managing the number of beams. Actually, when performing anti-aliasing, beams are cut in four sub-beams at each level, which can lead to an explosion of the number of beams to achieve the necessary accuracy. Unfortunately, these beams need to be stored in buffers to be handled on the GPU, which represents a lot of data.

In order to solve this issue, we use serialisation of the beams. Which means that when a beam is aliased, instead of generating the four beams at once, only the first child is generated, and when it is finished, it will launch his brother. With this approach, we ensure that the number of beams to treat never rises. However, if there is enough space left in the beam buffer, some beams may be de-serialised, in order to fill the buffer completely. Indeed, the GPU needs to be loaded enough to be efficient.

3.5. Multi-frequencies

Multi-frequencies computations consist in computing several (a lot, e.g., one to ten thousands) frequencies at the same time. Frequencies are independent in terms of electromagnetic computations, but share all the geometric computations (if we consider that anti-aliasing parameters given for the highest frequency are also valid for the smaller ones). In this new version, we postpone the

frequency computations as long as possible. The anti-aliasing process generates all geometrical contributors. These contributors are then used multiple times for applying EM models, one time per frequency.

4. RESULTS

We tested our new implementation of SE-RAY-EM versus the old one, in terms of accuracy and performances.

4.1. Accuracy

The first test consists in comparing qualitatively the results between the new implementation and both the standard version of SE-RAY-EM and the reference from the ELSEM3D software [4] which is based on Integral Equation EM approach and Moment Method numerical implementation. The test consists in the computation of the Radar Cross Section (RCS) computation of an aircraft for a frequency of 600 MHz (as represented in Figure 4). Figure 5 shows the results that are obtained. Some differences exist because signal of a complex object is noisy, but globally results computed by the three codes are very similar.

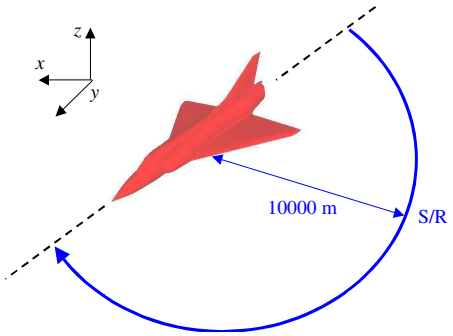

Figure 4: Aicraft RCS computation setup.

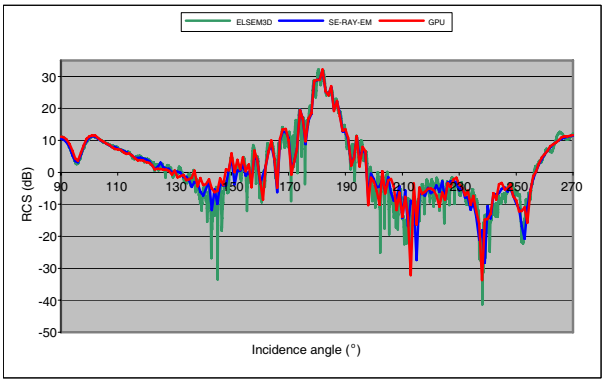

Figure 5: RCS of an aircraft computed with ELSEM3D (green), SE-RAY-EM (blue) and GPU version (red).

4.2. Performances

The test set up consists in the computation of the RCS at 10000 m of a tank with one or 151 frequencies from 8 to 11 GHz. Figure 6 shows the complex tank used for the simulation and an ISAR image generated from computed multi-frequencies and multiple angles results.

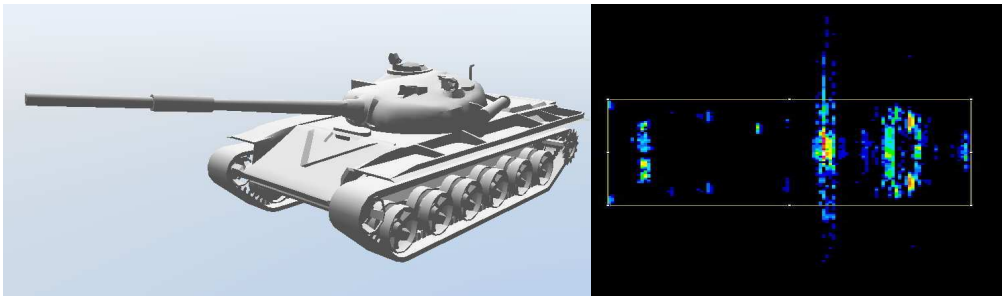

Figure 6: The tank used for test simulation and a computed ISAR image.

To compare the performances, the computations have been performed using the standard version of SE-RAY-EM and using the new parallel implementation with both a CPU and a GPU version. We performed the tests on two distinct computers:

- One laptop: Intel Core I7 2630QM and Nvidia GeForce 540M.
- One desktop: Intel Core I5 3470 and Nvidia GeForce GTX Titan.

Table 1: Computation times for one incidence in seconds with SE-RAY-EM standard version and with new parallel version in both CPU and GPU implementations.

	Standard Core I7	**Standard Core I5**	**CPU Core I7**	**CPU Core I5**	**GPU GeForce 540M**	**GPU GeForce Titan**
Mono-frequency	16.28	12.17	7.49	5.60	0.86	0.111
151 frequencies	109.00	80.40	97.85	70.4	7.62	0.735

Comparing the times of the standard version and the CPU new version, we can see that the new algorithm is more efficient than the previous one. The GPU implementation is 10 to 100 times faster than the standard SE-RAY-EM version.

5. CONCLUSION

Our new implementation of SE-RAY-EM is still on the road. It is now capable of handling multi-frequency RCS computations on complex targets with metallic or dielectric materials. This new version take benefits of a brand new algorithm that is massively parallel and takes advantage of the wide computing power available on GPUs, providing the same results as the previous version but in much shorter computation times (from ten to one hundred times faster computation).

Compared to SE-RAY-EM standard version, functionalities are still missing in the new version. They are planned for development in the forthcoming year: speckle materials, multi-texture, transparency, diffraction, optimisation of multiple reception points...

REFERENCES

1. Whitted, T., "An improved illumination model for shaded display," *Communications of the ACM*, Vol. 23, 343–349, 1980.
2. Mametsa, H. J., S. Laybros, T. Volpert, P. F. Combes, P. N. N'Guyen, and P. Pitot, "FERMAT: A high frequency EM scattering code from complex scenes including objects and environment," *PIERS Proceedings*, 1–4, Pisa, Italy, March 28–31, 2004.
3. Aila, T. and S. Laine, "Understanding the efficiency of ray traversal on GPUs," *Proc. High-Performance Graphics*, 145–149, 2009.
4. Soudais, P., P. Leca, J. Simon, and T. Volpert, "Computation of the scattering from inhomogeneous objects with a discrete rotational symmetry and a nonsymmetric part," *IEEE Transactions on Antennas and Propagation*, Vol. 50, No. 2, February 2002.