Iterative Physical Optics GPU Accelerated

J.-P. Adam
IEEA, Courbevoie, France
jean-pierre.adam@ieea.fr

G. Kubické, R.Hémon
DGA-MI, Bruz, France

***Abstract*— An Iterative Physical Optics solver has been ported to CUDA in order to use the computation power of a GPU. The obtained speedup is outstanding, even for mid-range graphic cards.**

Keywords : Physical Optics ; IPO ; GPU acceleration

I. Introduction

Physical Optics (PO) is particularly well suited to compute the diffraction of electrically large objects. Iterative Physical Optics (IPO) is an improvement of PO that takes into account higher order interactions. Developed to simulate objects with many internal reflections, like cavities, IPO is now used for a wide range of application, as described in ref. [1] and ref. [2].

PO and IPO are usually classified as asymptotic methods. However, unlike other asymptotic techniques, like Geometrical Optics (GO), they avoid the implementation problems coming with the ray representation. For example, no ray caustics need to be considered or no complex ray tracing algorithm is used. Instead of rays, PO is based on the integration of surface currents directly deduced from the incident electromagnetic field. This also provides more accurate results. Unfortunately, these advantages require a much higher CPU usage.

II. GPU Acceleration

The aim of this paper is to show the hardware acceleration of IPO by using the processor embedded in the graphic cards, called GPU. These processors provide high computation power at lower cost than the classical CPU. In recent years, software packages like CUDA from NVIDIA (ref. [3]) made this computation power usable for non-graphic applications. This explains why more and more scientific applications are ported on GPU.

Only the most time consuming part of the IPO has been implemented on a graphic card : the integration loop over the currents to evaluate their radiation. Such a summation is now a common problem on a GPU and it was possible to benefit from the experience of other fields to use as efficiently as possible the GPU. The obtained performances are outstanding, even for the mid-range graphic card (i.e. Geforce), as shown in Fig. 2. Moreover, the high number of cores of the high-end card (i.e. Tesla) does not affect the parallelization efficiency.

III. Application Examples

The presented method was originally developed for Radar Cross Section (RCS) computations. For the example in Fig. 1, up to 6 IPO iterations are required, depending on the direction of incidence. Metal cavities require even more iterations. This ability to handle large numbers of internal reflections is very interesting for Electromagnetic Compatibility (EMC) applications, like reverberation chambers for example.

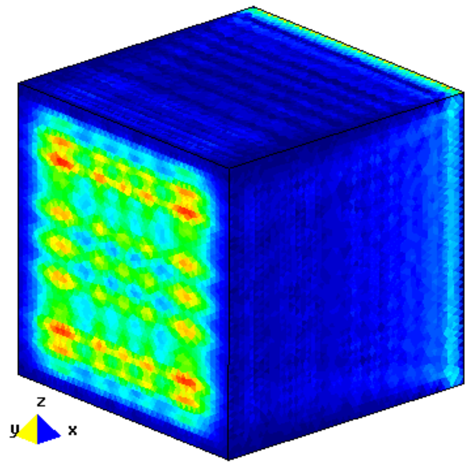

Figure 1. IPO electric surface currents inside a dielectric cube illuminated by a plane wave coming from the –x direction (the illuminated face is not visible on the picture). The size of the cube is 12 cm and its dielectric constant is 2.7–j0.01. The operating frequency is 30 GHz.

Description	Cores	Speed (GHz)	Parallelization	Time (s)	Speedup
CPU Phenom 9650	4	2.3	OpenMP	41700	1 X
GPU Geforce GT440	96	1.62	CUDA	890	47 X
GPU Tesla M2050	448	1.15	CUDA	250	167 X

Figure 2. Speedup obtained with the GPU for the IPO simulation presented in figure 1. The time corresponds to the computation of a monostatic RCS pattern of 91 directions. Each direction requires about 6 IPO iterations.

IV. Conclusion

In addition to its versatility and precision, the IPO algorithm has proven itself an efficient GPU executable, making it even more an adequate tool for real world electrically large structures simulation.

References

[1] R.J. Burkholder, C. Tokgoz, C.J. Reddy, P.H. Pathak, “Iterative physical optics: its not just for cavities anymore”, Antennas and Propagation Society International Symposium, vol. 1A, pp. 18- 21, July 2005

[2] P. Pouliguen, R. Hémon, J.-F. Damiens, "Physical optics for large scale electromagnetic scattering problems", EuCAP 2010 Proceedings, April 2010

[3] http://www.nvidia.com/object/cuda_home_new.html