B-CALM: An Open-Source GPU-based 3D-FDTD with Multi-Pole Dispersion for Plasmonics

Pierre Wahl[1,2], Dany-Sebastien Ly-Gagnon[2], Christof Debaes[1], David A.B. Miller[2] , Hugo Thienpont[1]
[1] Brussels Photonics Team, Applied Physics and Photonics Department, Pleinlaan 2, Vrije Universiteit Brussels
[2] Department of Electrical Engineering, Stanford University, CA 940305, USA

***Abstract*—Numerical calculations with finite-difference time-domain (FDTD) on metallic nanostructures in a broad optical spectrum require an accurate approximation of the permittivity of dispersive materials. Here, we present the algorithms behind B-CALM (Belgium-California Light Machine), an open-source 3D-FDTD solver operating on Graphical Processing Units (GPU's) with multi-pole dispersion models. Our modified architecture shows a reduction in computational times for multi-pole dispersion models for a broad spectral range. We benchmark B-CALM by computing the absorption efficiency of a metallic nanosphere with a one-pole and a three-poles Drude-Lorentz model and compare it with Mie theory.**

***Index Terms* – GPU, dispersive materials, multi-pole Lorentz model, plasmonics, NVIDIA, finite-difference time-domain(FDTD) methods.**

I. Introduction

Finite-Difference Time-Domain (FDTD) simulations play a prominent role in numerical electromagnetic calculations[1]. Many problems in nanophotonics require three dimensional full-field simulations, and FDTD simulations are often limited by the available computational power. While GPU's can be used as a very low cost hardware solution to accelerate FDTD, dispersive materials have been implemented for microwave frequencies, where the material dispersion can be simply modeled as a one-pole Drude medium [2]. However, metals at optical frequencies can have a permittivity with more complex features, which requires including multiple resonances to obtain an accurate model [3]. We present an algorithm to simulate multi-pole Drude-Lorentz materials and minimize thread divergence, enabling fast simulations of complex materials. As an example, we use B-CALM to simulate the absorption cross section of a gold nanosphere and compare the results with Mie theory. We find that a multi-pole Drude-Lorentz model significantly improves numerical accuracy.

II. Drude-Lorentz Model for Dispersive Media

The complex permittivity $\epsilon(\omega) = \epsilon^{'}(\omega) + i\epsilon^{''}(\omega)$ over a broad range of wavelengths can be modeled as

$$\epsilon_{DL}(\omega) = \epsilon_\infty + \sum_{m=0}^{P} \frac{{\omega_{pm}}^2}{\omega_m^2 - \omega^2 + i\omega\Gamma_m}. \qquad (1)$$

We fit the permittivity of gold to experimentally measured values [3] using a one-pole ($P = 1$) and a three-poles ($P = 3$) model for wavelengths between 650nm and 1200nm. For each pole, ω_{pm} and ω_m are used to fit the real part of $\epsilon_{DL}(\omega)$ and Γ_m to fit the imaginary part of $\epsilon_{DL}(\omega)$. The *nonlinear least squares method* [4] is used as the fitting method using uniform weigths and functions readily implemented in *Matlab*$^{\text{TM}}$. The fits are shown in Fig. 1. The three-poles model shows an error of less than 3% in absolute value over the entire band. In comparison, the one-pole model leads to an error greater than 35% at lower wavelengths, where the absolute value of permittivity is small.

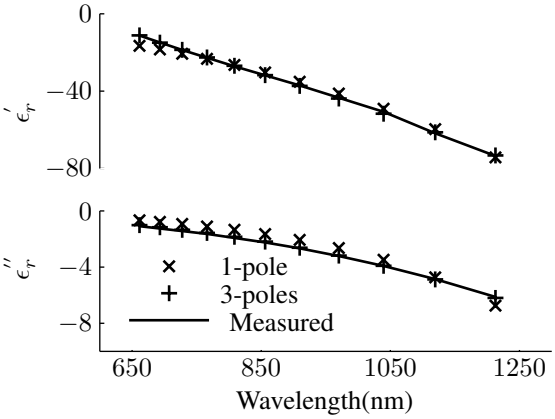

Figure 1. A three-poles model fits more accurately to the experimentally measured permittivity of gold [3] than a one-pole model.

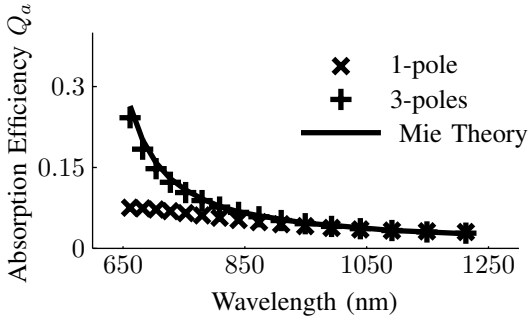

Figure 2. The absorption efficiency of a 200nm gold nanosphere calculated in FDTD as compared to Mie theory is significantly closer for a three-poles model than a one-pole model.

III. FDTD Implementation on GPU

The FDTD algorithm allows the calculation of propagating electromagnetic waves by alternately calculating the discretized electric and magnetic fields using a first-order spatial and temporal difference equation of Maxwell's equations[1]. For non-magnetic materials, the electric field update equation

in a cell with the permittivity described by Eq. 1 is

$$\begin{aligned}\mathbf{E}^{n+1} = &\underbrace{\mathbf{E}_s + C_1\mathbf{E}^n + C_2\nabla\times\mathbf{H}^{\mathbf{n+1/2}}}_{<<<\text{Update}E_A>>>} \\ &+ C_{pml}\Big(\underbrace{\mathbf{\Psi^n}_{\mathbf{E}_{\perp 1}} + \mathbf{\Psi^n}_{\mathbf{E}_{\perp 2}}}_{<<<\text{Update}E_A>>>}\Big) \\ &\underbrace{- C_3\mathbf{E}^{n-1} + \frac{1}{2}\sum_{p=1}^{P}\alpha_p\mathbf{J_p^n} + \xi_p\mathbf{J_p^{n-1}}}_{<<<\text{Update}E_B>>>},\end{aligned} \tag{2}$$

where n denotes the timestep, $\mathbf{E}$ the electric field, $\mathbf{E_s}$ the sourcing term and $\mathbf{H}$ the magnetic field. C_{pml} is a scaling constant specific to the *Perfectly Matched Layer (PML)* while $\mathbf{\Psi^n}_{\mathbf{E}_{\perp 1}}$ and $\mathbf{\Psi^n}_{\mathbf{E}_{\perp 2}}$ are recursive accumulators only stored in the PML regions. C_1, C_2, C_3, α_p, ξ_p are material specific parameters and C_3, α_p, ξ_p are only used in dispersive materials. Finally, $\mathbf{J_p^n}$ and $\mathbf{J_p^{n-1}}$ denote recursive accumulators only stored for the electric field of dispersive materials. The calculation of $\mathbf{\Psi^{n+1}}_{\mathbf{E}_{\perp 1,2}}$ requires $\mathbf{\Psi^n}_{\mathbf{E}_{\perp 1,2}}$ and $\mathbf{H}$ fields of neighboring cells. In constrast, the calculation of $\mathbf{J_p^{n+1}}$ requires $\mathbf{J_p^n}$, $\mathbf{J_p^{n-1}}$, $\mathbf{E}^n$ and $\mathbf{E^{n-1}}$, which are associated to the calculated cell only.

GPU's typically exploit the *Same Instruction Multiple Data (SIMD)* computer architecture [5]. This implies that all computation cores execute identical commands on each clock cycle, but on data contained in different memory addresses. To keep track of this process, groups of *threads-blocks* containing a constant amount of *threads* are allocated to a *kernel* which contains the instruction set for each thread. The execution speed of FDTD on GPU's is often limited by *memory bandwidth and latency* and *thread divergence*. The latter occurs when within a kernel, different threads have to perform different instructions, e.g. through an *if*-statement. Both paths are then performed *sequentially*, causing significant slowdown [5].

Smart use of memory to minimize bandwidth and latency

To minimize memory transfers, the constant parameters C_1, C_2, C_3, α_p, ξ_p are stored in a fast read-only *texture memory*. Also, fast *shared memory* is used as described in [6], [2], so that the electromagnetic fields in neighboring cells are only loaded once per update. The slower and large *device memory* is only used to store the fields $\mathbf{E}$, $\mathbf{H}$ and the recursive accumulators for the CPML's and the dispersive materials.

An additional kernel to alleviate thread divergence

In previous GPU implementations of FDTD [2], the update equations are split into two kernels, one for the electric and one for the magnetic field update equation. However, the electric field update creates a diverging path that leads to slow calculations. The situation is exacerbated for materials with a higher number of poles, as the calculation is done sequentially for the diverging paths. To alleviate this issue, B-CALM separates the electric field update equation into two separate kernels (labeled <<<UpdateE_A>>> and <<<UpdateE_B>>> in Eq. 2). The cost per field is one extra read and write operation to device memory to store the intermediate result as both kernels are called sequentially. As the underbraced area is responsible for $2(P+1)$ read-write operations per field[1], a split kernel is faster as soon as the number of poles $P > 1$.

IV. Results

We used B-CALM to calculate the absorption efficiency of a metallic sphere of $200\mathrm{nm}$ diameter under plane wave illumination and compare it with Mie theory[7] using the measured values of permittivity[3]. We used a uniform mesh of $1\mathrm{nm}$ and $208X208X400$ cells with two symmetry planes and a 15-cell wide PML. We monitor the absorbed power by substracting the integrated Poynting vector on a closed box surrounding the sphere from the integrated input flux. The simulation computed at $1.4e10 cells/min$ for a three-poles dispersion model on a NVIDIA C-1060 GPU, which is 30 times faster than with Meep[8] on a AMD Phenom$^{\mathrm{TM}}$ II X4 945 processor. As shown in figure 2, the single-pole model diverges significantly (error of >50%) while remaining below 5% for the the 3-pole model, greatly enhancing numerical accuracy.

V. Conclusion

We show that dispersive materials with complex wavelength dependences can be accurately simulated in a GPU-based FDTD while preserving the speed and low-cost advantage of GPUs. B-CALM, our GPU-accelerated open-source 3D-FDTD simulator allows us to quickly simulate complex metal structures for a broad wavelength range. B-CALM currently supports any user-defined sources, variable grid size, PML and is structured to easily allow the implementation of nonlinearity and anisotropy. B-CALM and its user-friendly interface can be freely downloaded at http://b-calm.sf.net.

Acknowledgements

The authors acknowledge the support of the Belgian American Education Foundation, the Methusalem and Hercules Foundation and the Interconnect Focus Center, one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity.

References

[1] A. Taflove and S. C. Hagness, *Computational electrodynamics: the finite-difference time-domain method*. Artech House Norwood, MA, 1995.

[2] M. Zunoubi and J. Payne, "Analysis of 3-Dimensional Electromagnetic Fields in Dispersive Media Using Cuda," *Progress In Electromagnetics Research*, vol. 16, pp. 185–196, 2010.

[3] E. D. Palik and G. Ghosh, *Handbook of optical constants of solids*. Academic press, 1998.

[4] C. T. Kelley, *Iterative methods for optimization*. Society for Industrial Mathematics, 1999.

[5] Nvidia, "Compute Unified Device Architecture-Programming Guide Version 2.3," *NVIDIA Corporation, July*, 2009.

[6] P. Micikevicius, "3D finite difference computation on GPUs using CUDA," in *Proceedings of 2nd Workshop on General Purpose Processing on Graphics Processing Units*. ACM, 2009, pp. 79–84.

[7] A. Ishimaru, *Wave propagation and scattering in random media*. Wiley-IEEE Press, 1999.

[8] A. F. Oskooi, D. Roundy, M. Ibanescu, P. Bermel, J. D. Joannopoulos, and S. G. Johnson, "Meep: A flexible free-software package for electromagnetic simulations by the FDTD method," *Computer Physics Communications*, vol. 181, no. 3, pp. 687–702, Mar. 2010.

[1] This corresponds to $\mathbf{E}^n$, $\mathbf{E^{n-1}}$ for each field and $\mathbf{J_p^n}$, $\mathbf{J_p^{n-1}}$ per pole per field.