Efiicient and scalable high-order FDTD modeling on GPGPU using DiamondTile algorithms

Andrey Zakirov[1] (zakirovandrey@gmail.com), Vadim Levchenko[2] (vadimlevchenko@mail.ru)
[1]Moscow Institute of Physics and Technology, Russia
[2]Keldysh Institute of Applied Mathematics, Russia

Introduction

Finite-Difference Time-Domain (FDTD) is the main method for solving vector wave equations in space-time region. Its base and most popular formulation is developed for Maxwell equations. Now FDTD is used in modelling of numerous electromagnetic problems and devices, from radio antennas to nano-optics and meta-materials.
Despite on simple formulation and many advantages FDTD is very computational costly for many actual 3D problems, so developing of fast and efficient algorithms and programs is quite important task.
Here is we present CUDA realization of high-efficient algorithms (named LRnLA DiamondTile) of 3D FDTD method
for Maxwell equations, which includes following capabilities:

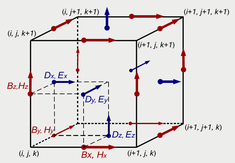
- Perfectly Matched Layer (PML) as boundary conditions;
- 4th space order which allows to use more coarse discretization;
- Different materials, including dispersive and nonlinear;
- Special sources, such as TF/SF (Total Field / Scattered Field).

Maxwell equations and FDTD method

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t}, \qquad \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}.$$

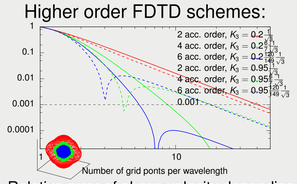
+ Material equations $\mathbf{E} = f(\mathbf{D}), \quad \mathbf{H} = g(\mathbf{B}).$

The speed of light in vacuum is set as 1.



Yee cell

Higher order FDTD schemes:



Relative error of phase velocity depending on number of yee cells per wavelength (for constant Courant number K)

For 3D problems space discretization can be significantly decreased for 4th-order scheme.

DiamondTile algorithm

Traditional FDTD realization implies layer-by-layer synchronization and domain-decomposition method for any parallelization.
But the computational rate will be very low in this case. To reach high efficiency as in CPU as in GPGPU one need to take in account locality properties of numerical scheme's graph.
The distance between two acts of this graph:

$$\rho(A,B) = \sqrt[p]{\sum_{i=1}^{dim} \left| x_i^A - x_i^B \right|^p}.$$

For cross stencil $p = 1$, so
diamond($dim = 2$) or octahedron($dim = 3$) is the projection of dependency/influence domain of one point.
— Locality parameter for traditional layer-by-layer algorithm $\sim 1/3$;
Locality parameter for DiamondTile algorithm $\sim \sqrt[3]{D}$;
— Tiling allows flexible alteration numerical scheme at different zones of computational area;
— No actual limits for problem size;

LRnLA (Local Recursive non-Local Asynchronous) algorithms

— Divide et impera idea;
The whole large task is decomposed recursively on multiple small problems
It corresponds to hierarchical memory model of modern computer.

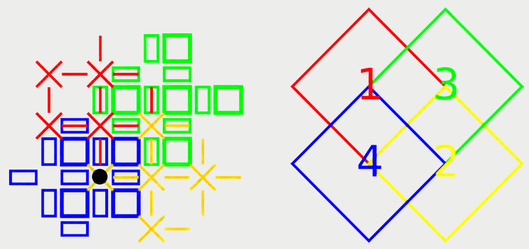
DiamondTorre algorithm

Cuda threads Vectorization — along Z axis;
Cuda blocks — along Y axis;

Notation of six field values:

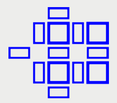
Hy	Hz	Hx	Ex	Ez	Ey

Data structure



RagRug structure and base point position

— $\mathrm{DiamondH}_0$

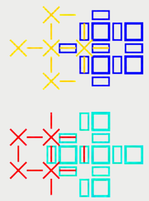


— $\mathrm{DiamondE}_0$

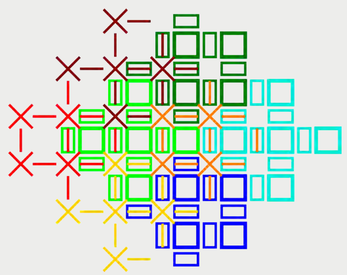
$\mathrm{DiamondH}_0$ + $\mathrm{DiamondE}_0$ = **DiamondFold**

$\mathrm{DiamondH}_0$ + $\mathrm{DiamondE}_0$ + $\mathrm{DiamondH}_1$ + $\mathrm{DiamondE}_1$ +... = **DiamondTorre**

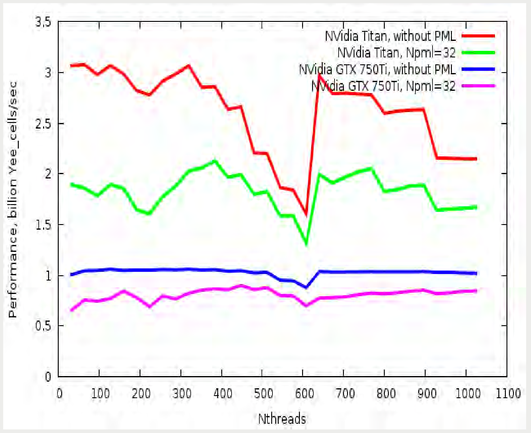
DiamondFold (two cases):



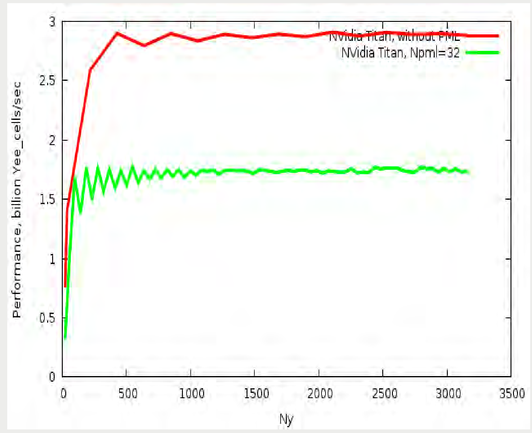
Dependecies of one DiamondFold:



Performance tests



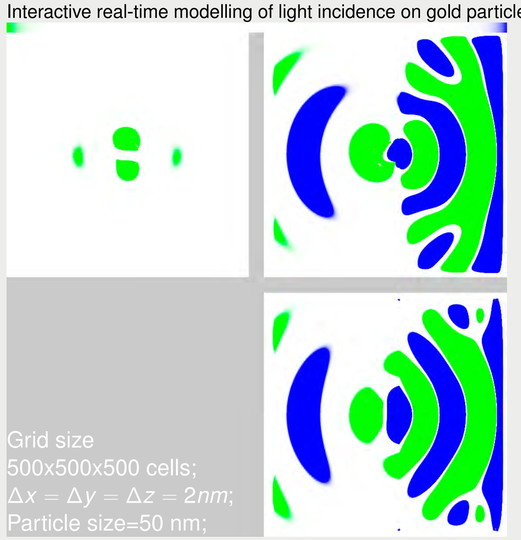
Performance of 4th-order FDTD depending on Cuda-threads parallelization



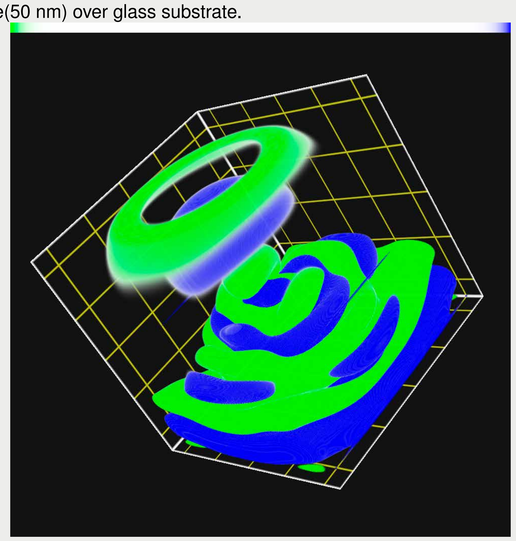
Performance of 4th-order FDTD depending on Cuda-blocks parallelization

Example of plasmonic nanoparticle resonance

Interactive real-time modelling of light incidence on gold particle(50 nm) over glass substrate.



Grid size
500x500x500 cells;
$\Delta x = \Delta y = \Delta z = 2nm$;
Particle size=50 nm;



Results

1. Based on DiamondTile algorithms FDTD(4,2) programm code with high performance (up to $3 \cdot 10^9 Yee_cells/sec$ on one GPGPU card) is presented. The code has PML boundary conditions and different material types which includes dispersion.
2. The code can be used both for interactive modelling of electromagnetic problems and large-scale high-performance computing.

Levchenko V.D. // Information technology and computing systems. 2005. N1. p.68