Progress In Electromagnetics Research, Vol. 125, 55–77, 2012

GPU IMPLEMENTATION OF SPLIT-FIELD FINITE-DIFFERENCE TIME-DOMAIN METHOD FOR DRUDE-LORENTZ DISPERSIVE MEDIA

A. Shahmansouri[1] and B. Rashidian[1, 2, *]

[1]Institute for Nanoscience & NanoTechnology, Sharif University of Technology, Azadi Avenue, Tehran 14588-89694, Iran

[2]Department of Electrical Engineering, Sharif University of Technology, Azadi Avenue, Tehran 11155-9363, Iran

Abstract—Split-field finite-difference time-domain (SF-FDTD) method can overcome the limitation of ordinary FDTD in analyzing periodic structures under oblique incidence. On the other hand, huge run times of 3D SF-FDTD, is practically a major burden in its usage for analysis and design of nanostructures, particularly when having dispersive media. Here, details of parallel implementation of 3D SF-FDTD method for dispersive media, combined with total-field/scattered-field (TF/SF) method for injecting oblique plane wave, are discussed. Graphics processing unit (GPU) has been used for this purpose, and very large speed up factors have been achieved. Also a previously reported formulation of SF-FDTD based on the Drude model for dispersive media, is extended to cover Drude-Lorentz model, which is usually needed for materials such as gold. The resulting reduction in the number of variables in this formulation, not only helps in reducing the computational time, but also makes it possible to be implemented in GPU, where its memory limitation is a major concern. As an example for demonstrating the importance of this method in optimization of nanophotonics structures, improvement in the performance of a refractive index sensor, made of an array of nanodisks, using suitable angle of incidence is reported. To the best of our knowledge this is the first report of GPU implementation of SF-FDTD method, capable of analyzing periodic dispersive media under oblique incidence.

Received 5 January 2012, Accepted 9 February 2012, Scheduled 18 February 2012

* Corresponding author: Bizhan Rashidian (rashidia@sharif.ir).

1. INTRODUCTION

Periodic plasmonic nanostructures have found many applications in nanophotonics. The response of these structures usually depends on the incident angle. In order to optimize structures, having a time efficient analysis tool is essential. Split-field finite-difference-time-domain (SF-FDTD) method [1] is a powerful method for analyzing periodic structures under oblique incidence, a case that ordinary FDTD is facing difficulty. SF-FDTD method has the advantage of wideband analysis of periodic structures by considering only one unit cell. This method has been used for analyzing dielectric or lossy [1–7], anisotropic [8] and dispersive [9, 10] periodic structures. We have recently reported an improved formulation of 3D SF-FDTD [11] for analyzing dispersive media. In deriving those formulations,Drude model was assumed for dispersive media. Although Drude model can be applied to metals such as silver or aluminum in visible or infrared spectral range, for some other materials such as gold, it might be necessary to use Drude-Lorentz model. In this paper the extended formulation for implementing Lorentz model in SF-FDTD is reported. This implementation of Drude-Lorentz model has considerably reduced number of variables compared to the previously reported works [9, 10].

Using SF-FDTD, especially in the case of dispersive media, faces the problem of huge runtimes needed on ordinary computational platforms. This challenge originates from the large number of variables present in the SF-FDTD formulation, much more than that in ordinary FDTD.

One logical solution for reducing the computational load of FDTD could be increasing the time step. However, it cannot be used in fully explicit FDTD method, due to Courant stability condition [12, Chap. 4]. Some methods have been reported for ordinary FDTD based on implicit FDTD to overcome the Courant stability restraint on the time step of fully explicit FDTD. ADI-FDTD [13–16], and LOD-FDTD [17–21] have been developed for this purpose. In this way, up to 5 times reduction in computation times have been reported [21]. These methods require larger memory than fully explicit FDTD. Also the implementation of these methods for 3D problems faces more complexity compared to 2D problems.

Parallel processing can efficiently reduce the runtime. Among different approaches, such as using clusters [22] or field programmable gate arrays (FPGA) [23], using graphics processing unit (GPU) is more efficient, and cheaper in reducing runtimes. In spite of large efforts on GPU parallel programming for ordinary FDTD [24–31], GPU parallel programming for SF-FDTD has not been reported yet, to the best of

our knowledge. Even though cluster programming by using Massage Passing Interface (MPI) has been reported for SF-FDTD [32, 33], it has not achieved good efficiency, compared to the GPU implementation reported here.

In order to show the practical importance of this method in designing nanostructures, a plasmonic refractive index sensor made of an array of nanoparticles is investigated. It is shown that the refractive index sensing performance can become much better at oblique incidence, compared to normal incidence.

The extended formulation for applying Lorentz model in SF-FDTD is described first. After a brief review on GPU programming, implementation of SF-FDTD combined with Drude-Lorentz model for dispersive media and total-field/scattered-field (TF/SF) method (for plane wave injection) on GPU will be discussed, and the achieved speed up factors will be reported. Refractive index sensors are introduced next, and the results of simulations done for a plasmonic refractive index sensor will be discussed.

2. FORMULATION OF LORENTZ MODEL IN SF-FDTD

To analyze a structure with periodicity in two dimensions (y and z), only one unit cell can be considered (Fig. 1), by taking advantage of the periodic boundary condition in these directions. In the third direction (x) the structure should be truncated, by using an appropriate absorbing boundary condition. Convolutional perfectly matched layer (CPML) [34] is chosen for this purpose.

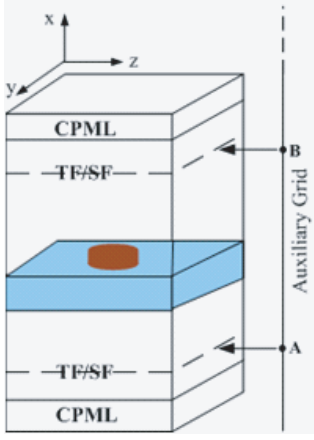


Figure 1. One unit cell of a periodic array.

In the SF-FDTD method, the field transformation is applied to the electric and magnetic fields in order to eliminate the phase shift between periodic boundaries in oblique incidence [12, Chap. 13]:

$$\breve{\mathbf{P}} = \breve{\mathbf{E}} \exp[j(k_y y + k_z z)] \tag{1a}$$

$$\breve{\mathbf{Q}} = \eta_0 \breve{\mathbf{H}} \exp[j(k_y y + k_z z)] \tag{1b}$$

where $\breve{\mathbf{E}}$ and $\breve{\mathbf{H}}$ are (phasor domain) electric and magnetic field vectors, $\breve{\mathbf{P}}$ and $\breve{\mathbf{Q}}$ are (phasor domain) field transformed vectors, k_y and k_z are the wave numbers of incident wave in the periodicity directions y and z, and η_0 is the intrinsic impedance of free space. By inserting the field transformed variables in Maxwell's equations, extra time-derivative terms appear in the equations. To overcome this problem, the transformed field variables (P_x, P_y, P_z, Q_x, Q_y, Q_z) become split and new variables (P_{xa}, P_{ya}, P_{za}, Q_{xa}, Q_{ya}, Q_{za}) are defined. So the method is named split field FDTD (SF-FDTD).

The so called field transformations can be also applied to current $\breve{\mathbf{J}}$ by introducing a new field transformed vector variable $\breve{\mathbf{G}}$ [11]:

$$\breve{\mathbf{G}} = \breve{\mathbf{J}} \exp[j(k_y y + k_z z)] \tag{2}$$

The relative permittivity of a dispersive medium introduced by Lorentz model in the frequency domain is given by [12, Chap. 9]:

$$\varepsilon(\omega) = \varepsilon_\infty + \sum_{p=1}^{P} \frac{\Delta\varepsilon_p \omega_p^2}{\omega_p^2 + 2j\omega\delta_p - \omega^2} \tag{3}$$

Considering one pole (p) of the Lorentz model and $\breve{\mathbf{J}}_p$ as the polarization current vector associated with it, $\breve{\mathbf{J}}_p$ and $\breve{\mathbf{E}}$ are related through: [12, Chap. 9]

$$\breve{\mathbf{J}}_p = j\omega\varepsilon_0 \left(\frac{\Delta\varepsilon_p \omega_p^2}{\omega_p^2 + 2j\omega\delta_p - \omega^2} \right) \breve{\mathbf{E}} \tag{4}$$

Replacing $\breve{\mathbf{J}}_p$ and $\breve{\mathbf{E}}$ in the above equation by the field transformed variables $\breve{\mathbf{G}}_p$ and $\breve{\mathbf{P}}$ results:

$$\omega_p^2 \breve{\mathbf{G}}_p + 2j\omega\delta_p \breve{\mathbf{G}}_p - \omega^2 \breve{\mathbf{G}}_p = j\omega\varepsilon_0 \Delta\varepsilon_p \omega_p^2 \breve{\mathbf{P}} \tag{5}$$

As in auxiliary differential equation (ADE) method, performing inverse Fourier transformation on each term, and integrating, the time domain equation for G_p is achieved:

$$\frac{\partial^2 G_p}{\partial t^2} + 2\delta_p \frac{\partial G_p}{\partial t} + \omega_p^2 G_p = \varepsilon_0 \Delta\varepsilon_p \omega_p^2 \frac{\partial P}{\partial t} \tag{6}$$

The equations of SF-FDTD with dispersion are as follows [11]:

$$\frac{\varepsilon_\infty}{c}\frac{\partial P_{xa}}{\partial t}+G_x=\frac{\partial Q_z}{\partial y}-\frac{\partial Q_y}{\partial z},\quad \frac{\mu_r}{c}\frac{\partial Q_{xa}}{\partial t}=-\frac{\partial P_z}{\partial y}+\frac{\partial P_y}{\partial z} \tag{7a}$$

$$\frac{\varepsilon_\infty}{c}\frac{\partial P_{ya}}{\partial t}+G_y=-\frac{\partial Q_z}{\partial x}+\frac{\partial Q_x}{\partial z},\quad \frac{\mu_r}{c}\frac{\partial Q_{ya}}{\partial t}=\frac{\partial P_z}{\partial x}-\frac{\partial P_x}{\partial z} \tag{7b}$$

$$\frac{\varepsilon_\infty}{c}\frac{\partial P_{za}}{\partial t}+G_z=\frac{\partial Q_y}{\partial x}-\frac{\partial Q_x}{\partial y},\quad \frac{\mu_r}{c}\frac{\partial Q_{za}}{\partial t}=-\frac{\partial P_y}{\partial x}+\frac{\partial P_x}{\partial y} \tag{7c}$$

$$\begin{aligned}&\left(1-\frac{\bar{k}_y^2}{\mu_r\varepsilon_\infty}-\frac{\bar{k}_z^2}{\mu_r\varepsilon_\infty}\right)P_x=P_{xa}+\frac{\bar{k}_z}{\varepsilon_\infty}Q_{ya}-\frac{\bar{k}_y}{\varepsilon_\infty}Q_{za},\\&\left(1-\frac{\bar{k}_y^2}{\mu_r\varepsilon_\infty}-\frac{\bar{k}_z^2}{\mu_r\varepsilon_\infty}\right)Q_x=Q_{xa}-\frac{\bar{k}_z}{\mu_r}P_{ya}+\frac{\bar{k}_y}{\mu_r}P_{za}\end{aligned} \tag{7d}$$

$$P_y=P_{ya}-\frac{\bar{k}_z}{\varepsilon_\infty}Q_x,\quad Q_y=Q_{ya}+\frac{\bar{k}_z}{\mu_r}P_x \tag{7e}$$

$$P_z=P_{za}+\frac{\bar{k}_y}{\varepsilon_\infty}Q_x,\quad Q_z=Q_{za}-\frac{\bar{k}_y}{\mu_r}P_x \tag{7f}$$

In the above equations (P_{xa}, P_{ya}, P_{za}, Q_{xa}, Q_{ya}, Q_{za}) are time domain split variables ("a" components), (P_x, P_y, P_z, Q_x, Q_y, Q_z) are time domain field transformed components, and (G_x, G_y, G_z) are time domain transformed current components (G_x, G_y, G_z are computed from the superposition of the results obtained from all poles, i.e., all G_p values). $\bar{k}_y=k_y/k_0$, $\bar{k}_z=k_z/k_0$, $k_0=2\pi f/c$, f is the frequency, and c is the speed of light in the free space.

In the SF-FDTD method, each field component is updated in each half time step ($\Delta t/2$). So the difference formula for G_p can be written as:

$$\begin{aligned}&\frac{G_p^{n+1/2}-2G_p^n+G_p^{n-1/2}}{\left(\frac{\Delta t}{2}\right)^2}+2\delta_p\frac{G_p^{n+1/2}-G_p^{n-1/2}}{\Delta t}+\omega_p^2G_p^n\\&=\varepsilon_0\Delta\varepsilon_p\,\omega_p^2\frac{P^{n+1/2}-P^{n-1/2}}{\Delta t}\end{aligned} \tag{8}$$

And update formula is:

$$\begin{aligned}G_p^{n+1/2}\;=\;&\frac{\delta_p\,\Delta t/2-1}{\delta_p\,\Delta t/2+1}G_p^{n-1/2}+\frac{2-\omega_p^2\,(\Delta t/2)^2}{\delta_p\,\Delta t/2+1}G_p^n\\&-\frac{\varepsilon_0\Delta\varepsilon_p\,\omega_p^2\Delta t/4}{\delta_p\,\Delta t/2+1}P^{n-1/2}+\frac{\varepsilon_0\Delta\varepsilon_p\,\omega_p^2\Delta t/4}{\delta_p\,\Delta t/2+1}P^{n+1/2}\end{aligned} \tag{9}$$

As can be seen from Eq. (7), G components only affect the equations of P_{xa}, P_{ya}, P_{za}. Their update formulas are:

$$P_{xa}^{n+1/2}(i,j,k) = P_{xa}^{n-1/2}(i,j,k) + \frac{1}{\varepsilon_\infty}\begin{bmatrix} Ty\left(Q_z^n(i,j,k) - Q_z^n(i,j-1,k)\right) \\ -Tz\left(Q_y^n(i,j,k) - Q_y^n(i,j,k-1)\right) \\ -c\Delta t G_x^n(i,j,k) \end{bmatrix} \quad (10a)$$

$$P_{ya}^{n+1/2}(i,j,k) = P_{ya}^{n-1/2}(i,j,k) + \frac{1}{\varepsilon_\infty}\begin{bmatrix} Tz\left(Q_x^n(i,j,k) - Q_x^n(i,j,k-1)\right) \\ -Tx\left(Q_z^n(i,j,k) - Q_z^n(i-1,j,k)\right) \\ -c\Delta t G_y^n(i,j,k) \end{bmatrix} \quad (10b)$$

$$P_{za}^{n+1/2}(i,j,k) = P_{za}^{n-1/2}(i,j,k) + \frac{1}{\varepsilon_\infty}\begin{bmatrix} Tx\left(Q_y^n(i,j,k) - Q_y^n(i-1,j,k)\right) \\ -Ty\left(Q_x^n(i,j,k) - Q_x^n(i,j-1,k)\right) \\ -c\Delta t G_z^n(i,j,k) \end{bmatrix} \quad (10c)$$

where $T_x = c\Delta t/\Delta x$, $T_y = c\Delta t/\Delta y$, $T_z = c\Delta t/\Delta z$.

The update equations for other variables can be written in a similar way. The update equation of G_p (Eq. (9)) show that for updating at time $(n+1/2)\Delta t$, the components $G_p^{n-1/2}$, G_p^n, $P^{n-1/2}$, and $P^{n+1/2}$ are needed. In order to minimize the number of variables that have to be saved, it is better to update G_p in two stages, as described below.

In each half time step, at first (P_{xa}, P_{ya}, P_{za}, Q_{xa}, Q_{ya}, Q_{za}) components are updated. Then G_p components are updated based on the Eq. (9), without the last term which contains $P^{n+1/2}$. After that, (P_x, Q_x), (P_y, Q_y), and (P_z, Q_z) are updated, and then updating G components is completed by adding the term containing $P^{n+1/2}$. In this way implementation of one pole Lorentz model in SF-FDTD adds only 9 extra variables, and 3 extra equations (3 variables for saving $G_p^{n+1/2}$, 3 for $G_p^{n-1/2}$, and 3 for $P^{n-1/2}$). Adding each pole to the dispersion model increases the number of variables by 6, and the number of equations by 3. As discussed in [11], the ordinary SF-FDTD method needs 18 variables for implementation. Using CPML absorbing boundary condition adds 8 additional variables with the size of CPML layer. Memory storage and consumption time for updating all of the CPML variables are even less than that of one ordinary variable. So the total number of variables for ordinary SF-FDTD with CPML is about 19, and by inclusion of Lorentz model, it increases to 28. Implementing

Drude model (Eq. (11)) adds 6 extra variables to SF-FDTD [11], so SF-FDTD with Drude-Lorentz model (Eq. (12)) needs 34 variables.

The other method of SF-FDTD for dispersive media [9, 10] uses 112 variables for implementation of Drude-Lorentz dispersion model. The large number of variables not only increases the runtime, but also may prohibit the implementation of the algorithm on a GPU card, due to its limited memory.

$$\varepsilon(\omega) = \varepsilon_\infty - \sum_{p=1}^{P} \frac{\omega_p^2}{\omega^2 - j\omega\gamma_p} \tag{11}$$

$$\varepsilon(\omega) = \varepsilon_\infty - \sum_{P_D=1}^{P_D} \frac{\omega_{P_D}^2}{\omega^2 - j\omega\gamma_{P_D}} + \sum_{P_L=1}^{P_L} \frac{\Delta\varepsilon_{P_L}\omega_{P_L}^2}{\omega_{P_L}^2 + 2j\omega\delta_{P_L} - \omega^2} \tag{12}$$

3. PARALLEL PROCESSING FOR SF-FDTD

3.1. Overview of Parallel Processing by GPU

GPU devices are designed for compute-intensive, highly parallel computations. Each GPU device contains streaming multiprocessors (for example GeForce GTX 480 has 15 multiprocessors), and each multiprocessor consists of scalar processors (32 for GeForce GTX 480), and each scalar processor can execute 32 parallel threads (a thread is the smallest unit of processing); thus 15360 parallel threads can be executed in parallel by the GeForce GTX 480. Data-parallel processing maps data elements to the parallel processing threads [35, 36].

Compute unified device architecture (CUDA) by NVIDIA [36], provides a general purpose parallel computing architecture. CUDA comes with a software environment that allows programmers to use high-level programming language or application programming interfaces such as C, FORTRAN, OpenCL, and Direct Compute. CUDA allows programmers to program the GPU device, compute parallel functions (called kernels) that when called are executed N times in parallel by N different CUDA threads, and exchanging data between the host (CPU) and the device (GPU). Threads can be arranged into one dimensional, two dimensional or three dimensional thread blocks. These blocks are organized into a one dimensional or two dimensional grid. Data should be also partitioned into blocks that can be handled in parallel by blocks of threads.

CUDA threads can access data from different memory spaces on device, such as global, shared, constant, texture memory, and registers. All threads have access to the global, constant, and texture memories. These memories are persistent for all kernel launches during

an application. Shared memory is visible only for the threads in each block, and has the same life time of that block. Registers can be accessed only by one thread. It should be mentioned that constant and texture memory spaces are read-only. Registers, shared, and constant memory, are the fastest memory spaces, followed by texture and global memories. Global memory is the largest memory space. Accessing the global memory takes 400 to 600 clock cycles of memory latency, because this memory is off-chip and uncashed. Shared memory is also uncashed, but is on chip, it is about 100x faster than the global memory. Texture memory is off-chip but it is cashed, hence its latency is lower than the global memory.

3.2. Implementation of SF-FDTD on GPU

Host (CPU) and device (GPU) are programmed by Visual Studio C++ linked by CUDA. At first, static data, variables, and outputs are initialized in the host. Static data (dimensions, time and spatial steps, k_y and k_z, and CPML parameters) are copied to GPU constant memory, constant arrays (like permittivity array) are bind to the texture memory. The variables (P_a & Q_a, P & Q, G, and CPML variables) are initialized on the GPU global memory. The intermediate variables (P_a & Q_a, G, and CPML variables) are only initialized on the GPU memory.

In FDTD method, for injecting a perfect plane wave at an arbitrary angle to the 3D structure, TF/SF method is used [12, Chap. 5]. In this method a virtual closed surface is considered around the structure and the incident field components are added to the field components on the virtual surface. The region inside the virtual closed surface is called total-field (TF) region,which contains both incident and scattered fields. The region outside the virtual surface is called scattered-field (SF) region, which contains only the scattered fields. Using an auxiliary one-dimensional grid and calculating the incident field components on this grid, followed by applying the field components to the 3D virtual surface, effectively reduce the computational time. The implementation of TF/SF method for SF-FDTD was discussed in [11]. Formulation was derived for calculating the incident transformed fields on an auxiliary one-dimensional grid. The calculated fields on this grid are applied to the planes of TF/SF (Fig. 1). The same field values are imposed on all points of each TF/SF plane. Implementation of TF/SF in SF-FDTD is simpler than ordinary FDTD, where for injection of an oblique plane wave a Look-Up table should be generated, based on the field components of auxiliary grid, and data should be interpolated to obtain the incident field values on TF/SF surface [12, Chap. 5].

The equations for the auxiliary grid in SF-FDTD are as follows [11]:

$$\frac{\varepsilon_r}{c}\frac{\partial P_{ya}}{\partial t} = -\frac{\partial Q_z}{\partial x}, \quad \frac{\mu_r}{c}\frac{\partial Q_{ya}}{\partial t} = \frac{\partial P_z}{\partial x} \tag{13a}$$

$$\frac{\varepsilon_r}{c}\frac{\partial P_{za}}{\partial t} = \frac{\partial Q_y}{\partial x}, \quad \frac{\mu_r}{c}\frac{\partial Q_{za}}{\partial t} = -\frac{\partial P_y}{\partial x} \tag{13b}$$

$$\begin{aligned}\left(1 - \frac{\bar{k}_y^2}{\mu_r\varepsilon_r} - \frac{\bar{k}_z^2}{\mu_r\varepsilon_r}\right) P_x = P_{xa} + \frac{\bar{k}_z}{\varepsilon_r} Q_{ya} - \frac{\bar{k}_y}{\varepsilon_r} Q_{za} \\ \left(1 - \frac{\bar{k}_y^2}{\mu_r\varepsilon_r} - \frac{\bar{k}_z^2}{\mu_r\varepsilon_r}\right) Q_x = Q_{xa} - \frac{\bar{k}_z}{\mu_r} P_{ya} + \frac{\bar{k}_y}{\mu_r} P_{za}\end{aligned} \tag{13c}$$

$$P_y = P_{ya} - \frac{\bar{k}_z}{\varepsilon_r} Q_x, \quad Q_y = Q_{ya} + \frac{\bar{k}_z}{\mu_r} P \tag{13d}$$

$$P_z = P_{za} + \frac{\bar{k}_y}{\varepsilon_r} Q_x, \quad Q_z = Q_{za} - \frac{\bar{k}_y}{\mu_r} P_x \tag{13e}$$

Implementation of the above equations needs 14 variables. If the auxiliary grid is truncated by an absorbing boundary condition, the reflection error is unavoidable. The auxiliary grid can be made long enough to avoid fields from reaching its end, even at the last time steps, simply by choosing the number of spatial steps on this grid more than the total number of time steps (T). To avoid large memory usage of these variables during the calculations of the structure, the calculation of the auxiliary grid can be done before initializing the structure on the GPU memory. After initializing the variables of the auxiliary grid, the fields are calculated in parallel by launching appropriate kernels based on Eq. (13) for all of the time steps (T). In each half time step ($\Delta t/2$), the field values which should be applied to the TF/SF interfaces (P_y, P_z, Q_y, Q_z of points A and B on the auxiliary grid of Fig. 1) are saved in a Source matrix on the GPU global memory. These values are then added to (P_{ya}, P_{za}, Q_{ya}, Q_{za}) components of TF/SF planes while running the structure. This is the advantage of SF-FDTD, over ordinary FDTD, for oblique incidence. In the implementation of the TF/SF for ordinary FDTD, the points on each TF/SF plane get different values, and in each time step the fields of every points located on TF/SF planes should be saved. By finishing the auxiliary grid calculations, all its variables are cleaned from the global memory, and only the Source matrix is kept. This strategy helps to keep free the maximum amount of device memory during the analysis. After that, the structure variables (P_a & Q_a, P & Q, G, and CPML variables) are initialized in the GPU global memory and the kernels for computing equations of SF-FDTD are launched one by one. Variables are arranged

into data blocks. In each kernel launch, data blocks are loaded from the global memory to the shared memory (being on chip, thus very fast), update equation is run, and the results are sent to the global memory. By finishing the calculations at each half time step, the field variables (P & Q) are sent to the host for post processing. The flowchart of algorithm is shown in Fig. 2.

As an example, kernel for calculating P_{ya} is shown in Fig. 3. P_{ya} is selected here due to the fact that this is one of those variables that Source matrix appears in its calculation. The field component saved in the global memory is defined by $d_$variable. Each variable is arranged into a 3D block, and in each kernel launch data block is copied to a shared memory variable which is defined by $sh_$variable. The 3D blocks are arranged such that the blocks in y-z plane compose a two dimensional grid. In calculating each variable at point (i, j, k) its neighboring field components are also needed. Considering that the shared memory is visible only for threads in each block, the extra data needed from neighboring cells, have to be copied in each block of shared memory. At each half time step, the needed value of Source matrix (defined as $d_$source are copied to the registers (QzHuyg1 and QzHuyg2), and added appropriately to the TF/SF planes (marked as $d_x_$HuygensDown and $d_x_$Huygensup). The static data which are copied to the constant memory are defined by adding "d" before the corresponding name (for example "dc", instead of "c" for the speed of light). The structure permittivity matrix is bind to the texture memory and defined as Tx_Structure.

In GPU parallel processing, all the variables are kept in the global memory of the device during the process, so their communication is fast. As GPU devices has limited amount of memory (for example 1.5 GB for GeForce GTX 480), and SF-FDTD uses large number of variables, reduction in the number of variables is very important here.

3.3. Speed up Factor

Transmission of a 35 nm thick Au thin film calculated by using parallel SF-FDTD method, using Drude-Lorentz model for Au dispersion [37], is shown in Fig. 4 for both TE and TM polarizations, in agreement with the analytical results [38]. To investigate the efficiency of GPU parallel processing, problems with different sizes are considered by changing transverse dimensions of the so called Au thin film. Different numbers of cells in the periodicity (y-z) plane are considered, while the number of cells in the third direction (x) is kept constant (160 number of cells). The number of time steps is set to 15000. In Table 1 the runtimes by using parallel algorithm on a GPU device (GeForce GTX 480) are compared with the runtimes of a nonparallel SF-FDTD (Visual Studio

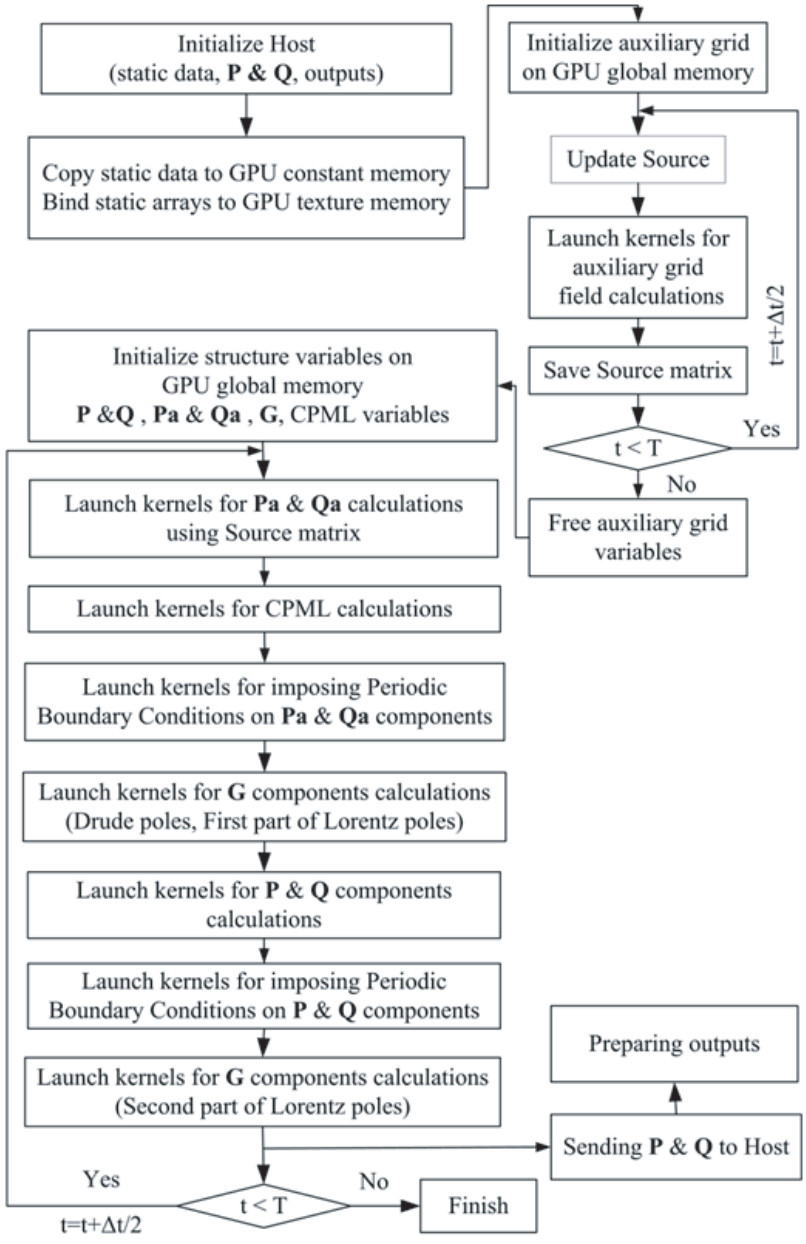


Figure 2. Flowchart of GPU parallel programming for SF-FDTD.

```
__global__ void __cuda_Pya ( double *d_Qx,  double *d_Qz,
                             double *d_Pya, double *d_Pya_prev,
                             double *d_GyD, double *d_GyL,
                  double *d_Source, int i_Time, int numBlocksX )
{
   double f,  QzHuyg1, QzHuyg2;
   int ix, iy, iz, i, j, k, ii, jj, kk, n, iBlocks_x ;

   const int stride_x = blockDimX ;
   const int stride_y=xdim;
   const int stride_z=xdim*ydim;

   __shared__ double  sh_Qx    [blockDimY] [blockDimX] [blockDimZ];
   __shared__ double  sh_Qz    [blockDimY] [blockDimX] [blockDimZ];
   __shared__ double  sh_prev  [blockDimY] [blockDimX] [blockDimZ];
   __shared__ double  sh_GyD   [blockDimY] [blockDimX] [blockDimZ];
   __shared__ double  sh_GyL   [blockDimY] [blockDimX] [blockDimZ];

for(iBlocks_x=0;   iBlocks_x < numBlocksX;    iBlocks_x++)
       {
          ix = threadIdx.x ;
          iy = threadIdx.y ;
          iz = threadIdx.z ;

          i = iBlocks_x * stride_x +  threadIdx.x ;
          j = blockIdx.x * blockDim.y + threadIdx.y ;
          k = blockIdx.y * blockDim.z + threadIdx.z ;

          ii = i - iBlocks_x ;
          jj = j - blockIdx.x ;
          kk = k - blockIdx.y ;

          n=ii+ jj*stride_y+ kk*stride_z ;

          sh_Qx    [ iy ][ ix ][ iz ] = d_Qx [ n ] ;
          sh_Qz    [ iy ][ ix ][ iz ] = d_Qz [ n ] ;
          sh_prev [ iy ][ ix ][ iz ] = d_Pya_prev [ n ] ;
          sh_GyD  [ iy ][ ix ][ iz ] = d_GyD [ n ] ;
          sh_GyL [ iy ][ ix ][ iz ] = d_GyL [ n ] ;
           __syncthreads();
```

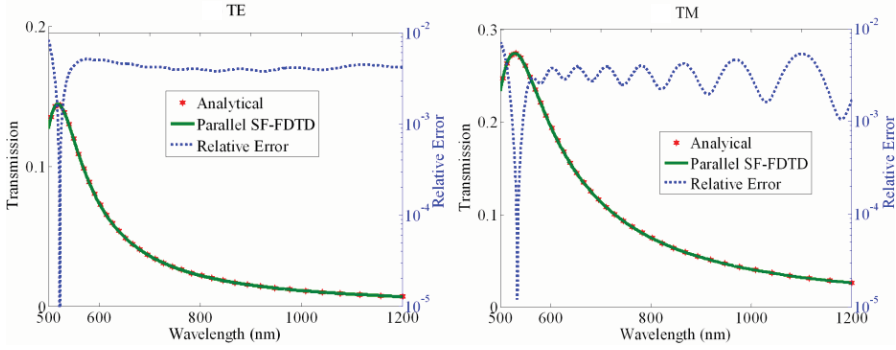
```
if (ii==d_x_HuygensDown)
                        { QzHuyg1= d_Source[8*i_Time+3];}else
                        { QzHuyg1= 0;    }
if (ii==d_x_HuygensUp+1)
                        { QzHuyg2= - d_Source[8*i_Time+7];}else
                        { QzHuyg2= 0;}

        if( ix > 0  &&  iy > 0  &&  iz > 0  )
        {
          const float st=tex1Dfetch(  Tx_Structure,float ( n));

          f = sh_prev [iy][ix][iz]+1/st*(-dTx*(sh_Qz [iy][ix][iz]-
                        (sh_Qz[iy][ix-1][iz] + QzHuyg1+QzHuyg2))
                        + dTz *(sh_Qx[iy][ix][iz]-sh_Qx [iy][ix][iz-1])-
                dc*ddt*(sh_GyD[iy][ix][iz]+sh_GyL[iy][ix][iz]) ) ;

          d_Pya_prev[n] = d_Pya[n] ;
          d_Pya[n] = f ;
        }
        __syncthreads();
        }
}
```

Figure 3. Kernel for calculating P_{ya} component.



(a) (b)

Figure 4. Transmission through 35 nm thick Au thin film, comparison between parallel SF-FDTD and analytical results. Both TE and TM polarizations are shown.

Table 1. Comparing the CPU and GPU runtimes for different problem size in the y and z directions, the size in x direction is kept constant (160 cells), the number of time steps is set to 15000.

Size of Transverse Plane	CPU Runtime	GPU Runtime	Speed Up Factor
8×8	497 sec	129 sec	3.8
16×16	1314 sec	144 sec	9.1
32×32	1.67 hours	219 sec	27.5
64×64	7.5 hours	609 sec	44.3
128×128	49.3 hours	2165 sec	82
160×160	84.4 hours	3345 sec	90.8

C++) on a Quad core CPU (Intel®Core(TM) i7 930 @ 2.80 GHz). The speed up factor is also calculated by dividing the runtime of CPU by that of GPU. It can be seen that for small problem sizes, the speed up is low, but it increases efficiently for larger problem sizes up to 90. Ordinary FDTD method combined with Drude-Lorentz model on GPU has been reported by Lee et al. [31]. Speed up factor of 21 was reported in comparison with the serial C++ version.

4. REFRACTIVE INDEX SENSOR

4.1. Refractive Index Sensor Overview

Plasmon resonances are observed in the interaction of noble metals, such as gold or silver, with incident electromagnetic waves. The Surface Plasmon Resonance (SPR) occurs on the surface of thin metallic films [39]. Localized Surface Plasmon Resonance (LSPR) occurs for metallic nanoparticles [40]. The plasmon behavior strongly depends on the structural geometry (size, shape), and also material properties of metal and surrounding medium [41]. Shift of plasmon resonant wavelength by the change in the refractive index of surrounding medium is the basis of plasmonic refractive index sensors. These sensors have attracted great attention in chemical, biological and gas sensing [42–47]. Two major kinds of these sensors have been developed: (1) Excitation of SPR on a metallic thin film, and measuring the shift of SPR wavelength, or excitation angle, caused by changing the refractive index of surrounding medium [48]. (2) Using

metallic nanoparticles in solution [49, 50], or in a periodic array [51–58], and measuring the shift of LSPR resulting from the change in the refractive index of surrounding medium. The SPR sensors mainly use the attenuated total internal refraction method for plasmon excitation. Kretschman configuration, by using prism coupling, is a usual excitation method [48]. These sensors have better sensitivity than LSPR sensors, and are commercially available. On the other hand, LSPR sensors are based on using transmission/extinction measurements, not requiring bulky prism excitations used in SPR excitation. That's why the LSPR sensors have attracted so much attention for nanoscale applications. Large efforts have been done to improve the sensitivity $S = \Delta\lambda/\Delta n$ (where, Δn is the change in the refractive index, expressed in Refractive Index Unit (RIU), and $\Delta\lambda$ is the corresponding wavelength shift), and figure of merit FOM = S/FWHM (FWHM is the full width at half maximum) of plasmonic refractive index sensors. NanoDisk arrays are investigated in [51, 52], the sensitivity and FOM were reported as $S \sim 178\,\text{nm/RIU}$, FOM $\sim$ 2 in [51], and $S \sim 84\,\text{nm/RIU}$, FOM $\sim$ 0.19 in [52]. Symmetric structures could not show high sensitivity and FOM. To improve sensitivity, arrays of nonsymetric nanoparticles have been considered. Nanowire array ($S \sim 300\,\text{nm/RIU}$) [53], nanorod array ($S \sim 884\,\text{nm/RIU}$, FOM $\sim$ 1), norod array combined with cavity ($S \sim 354\,\text{nm/RIU}$, FOM $\sim$ 7.1) [54], and gold nanobar array placing close to a thin gold film ($S \sim 600\,\text{nm/RIU}$, FOM $\sim$ 4.68) [55] are few examples. More complicated structures, such as nanocrescent array ($S \sim 332\,\text{nm/RIU}$, FOM $\sim$ 0.4) [56], nanocrosses array ($S \sim$ 500–740 nm/RIU, FOM $\sim$ 2–2.2) [57], and double nanopillar array with nanogap ($S \sim 1056\,\text{nm/RIU}$, FOM $\sim$ 12.2) [58] have also been investigated.

It has been shown that dipolar couplings of neighboring plasmons in a nanoparticle array tend to very narrow plasmon resonances [59–61]. In [62], by changing the nanoparticle size and lattice constant of periodic array, narrow and tuneable plasmon peaks have been achieved under normal incidence. That leads to high FOM in refractive index sensing. Narrow plasmon peaks and high phase sensitivity, has been reported experimentally for array of gold nanodots at large angles of oblique incidence [63]. We have reported numerically, by using parallel SF-FDTD, the appearance of narrow plasmon peaks in Au nanodisk array under oblique incidence [11, Fig. 8]. In this paper, we calculate and compare the refractive index sensitivity of nanodisk array under normal and oblique incidence. We will show that high sensitivity and FOM can be easily achieved under oblique incidence, without getting involved in more complicated structures.

4.2. Simulation Results

A square array (with array constant (Λ) of 540 nm) of Au nanodisks (with diameter of 180 nm and height of 40 nm) is considered. The substrate is a 20 nm thick layer of indium tin oxide (ITO) on glass (Fig. 5). This structure has been investigated in [11]. We have shown that under oblique incidence by using TE polarization (electric field along one axis of the square array) the plasmon peak has become narrower by increasing angle. Nanodisks were considered in water having refractive index of $n = 1.327$. FWHM was 68 nm at normal incidence, and 14 nm at incident angle of $\theta = 22°$. Here the extinction cross section of the array is plotted by changing the refractive index of the medium surrounding the nanodisks from $n = 1.2$ up to $n = 1.5$. The results for normal incidence are shown in Fig. 6(a), and for $\theta = 22°$ are shown in Fig. 6(b). In calculating the extinction cross section of the array, Drude-Lorentz model was considered for Au dispersion. The sensitivity and FOM are calculated from the results of Fig. 6. For normal incidence, $S \sim 300$–380 nm/RIU, and FOM $\sim$ 4.85–5.2 are achieved. While for $\theta = 22°$, $S \sim 370$–500 nm/RIU, and FOM $\sim$ 14.8–33 are achieved. It can be seen that using oblique incidence efficiently increases the FOM, thus the performance in the refractive index sensing. This example demonstrates the applicability of the SF-FDTD in optimization of plasmonic nanostructures.

By changing the refractive index, it can be seen that for normal incidence the extinction cross section changes gradually, but for oblique incidence at $\theta = 22°$ the extinction cross section is larger for $n = 1.327$. The nanodisks in the array experience far field (dipolar) coupling. The plasmon spectra are strongly affected by the excitation of diffraction grating orders into the medium surrounding the nanodisks and also the substrate. This effect was fully discussed in [11, Sec. 3B]. The cut off

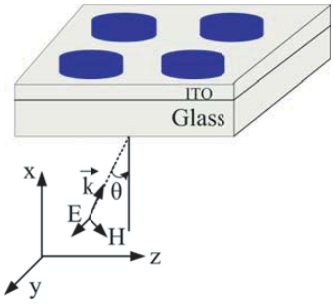


Figure 5. Au nanodisks array under oblique incidence with TE polarization.

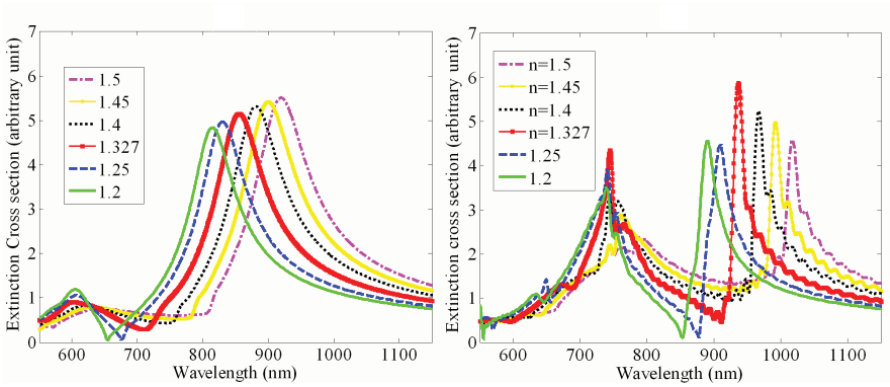


Figure 6. Extinction cross section of nanodisk array by changing refractive index. (a) Normal incidence. (b) Oblique incidence, $\theta = 22°$.

wavelength of the first grating order into each layer, L (ITO: $n_{\text{ITO}} = 1.45$, and medium surrounding the nanodisks: n_m) are calculated by: $\lambda_{1stL} = \Lambda \times n_{\text{air}} \times \sin(\theta) + \Lambda \times n_{\text{L}}$.

At normal incidence $\lambda_{1\text{stITO}}$ and $\lambda_{1\text{stm}}$ occurs far from the plasmon peak and have less effect on the shape of the curve. At $\theta = 22°$, the $\lambda_{1\text{stITO}}$ and $\lambda_{1\text{stm}}$ occurs near the plasmon peak, strongly affect the shape. The interaction of diffraction grating order and plasmon resonance is maximum at $\theta = 22°$ for $n_m = 1.327$. For other amount of n_m, the maximum interaction may occur at different angles.

5. CONCLUSION

In conclusion, a formulation was derived for applying Lorentz model for dispersive media (using ADE method) in 3D SF-FDTD method. It has been discussed that Drude-Lorentz model can be implemented in SF-FDTD by considerably reduced number of variables compared to other reported methods. GPU parallel programming by using CUDA was explained for SF-FDTD combined with dispersive media and TF/SF method. The efficiency of GPU parallel programming has been reported by comparing runtimes of parallel algorithm with nonparallel algorithm (Visual Studio C++). Speed up factor up to 90 has been achieved.

We chose ADE method as our first step, based on comparison made in [12, Chap.9, page 361] between ADE and recursive convolution methods (RC, PLRC), and the fact that availability of fields at each half time step in SF-FDTD simplifies implementation of ADE

(alleviating the need for time averaging used in ADE for ordinary FDTD). Considering more recently reported works [21] on recursive convolution methods, adapting them to the method presented in this paper, might result in more improvements.

Refractive index sensing for nanodisks array under oblique incidence was compared with normal incidence. Much better performance was achieved under oblique incidence. This demonstrates the importance of having an efficient numerical tool for analyzing periodic structures under oblique incidence, where the ordinary FDTD cannot be used.

REFERENCES

1. Roden, J. A., S. D. Gedney, M. P. Kesler, J. G. Maloney, and P. H. Harms, "Time-domain analysis of periodic structures at oblique incidence: Orthogonal and nonorthogonal FDTD implementations," *IEEE Trans. Microwave Theory Tech.*, Vol. 46, No. 4, 420–427, 1998.
2. Maloney, J. G. and M. P. Kesler, "Analysis of antenna arrays using the split-field update FDTD method," *Antennas and Propagation Society International Symposium, IEEE*, 2036–2039, 1998.
3. Wu, B., E. Yang, J. A. Kong, J. A. Oswald, K. A. McIntosh, L. Mahoney, and S. Verghese, "Analysis of photonic crystal filters by the finite-difference time-domain technique," *Microwave Opt. Technol. Lett.*, Vol. 27, No. 2, 81–87, 2000.
4. Mosallaei, H. and Y. Rahmat-Samii, "Grand challenges in analyzing EM band-gap structures: An FDTD/Prony technique based on the split-field approach," *Antennas and Propagation Society International Symposium, IEEE*, 47–50, 2001.
5. Farahat, N. and R. Mittra, "Analysis of frequency selective surfaces using the finite difference time domain (FDTD) method," *Antennas and Propagation Society International Symposium, IEEE*, 568–571, 2002.
6. Amjadi, S. M. and M. Soleimani, "Design of band-pass waveguide filter using frequency selective surfaces loaded with surface mount capacitors based on split-field update FDTD method," *Progress In Electromagnetics Research B*, Vol. 3, 271–281, 2008.
7. Belkhir, A. and F. I. Baida, "Three-dimensional finite-difference time-domain algorithm for oblique incidence with adaptation of perfectly matched layers and nonuniform meshing: Application to the study of a radar dome," *Phys. Rev. E*, Vol. 77, No. 5, 056701, 2008.

8. Oh, C. and M. J. Escuti, "Time-domain analysis of periodic anisotropic media at oblique incidence: An efficient FDTD implementation," *Opt. Express*, Vol. 14, No. 24, 11870–11884, 2006.
9. Baida, F. I. and A. Belkhir, "Split-field FDTD method for oblique incidence study of periodic dispersive metallic structures," *Opt. Lett.*, Vol. 34, No. 16, 2453–2455, 2009.
10. Belkhir, A., O. Arar, S. S. Benabbes, O. Lamrous, and F. I. Baida, "Implementation of dispersion models in the split-field-finite-difference-time-domain algorithm for the study of metallic periodic structures at oblique incidence," *Phys. Rev. E*, Vol. 81, 046705, 2010.
11. Shahmansouri, A. and B. Rashidian, "Comprehensive three-dimensional split-field finite difference time-domain method for analysis of periodic plasmonic nanostructures: Near- and far-field formulation," *J. Opt. Soc. Am. B*, Vol. 28, No. 11, 2690–2700, 2011.
12. Taflove, A. and S. C. Hagness, *Computational Electrodynamics: The Finite-Difference-Time-Domain Method*, Artech House, 2005.
13. Zheng, F., Z. Chen, and J. Zhang, "A finite-difference time-domain method without the courant stability conditions," *IEEE Microw. Guided Wave Lett.*, Vol. 9, No. 11, 441–443, 1999.
14. Namiki, T., "A new FDTD algorithm based on alternating-direction implicit method," *IEEE Trans. Microwave Theory Tech.*, Vol. 47, No. 10, 2003–2007, 1999.
15. Wang, S., F. L. Teixeira, and J. Chen, "An iterative ADI-FDTD with reduced splitting error," *IEEE Microw. Wireless Compon. Lett.*, Vol. 15, No. 2, 92–94, 2005.
16. Ahmed, I. and Z. Chen, "Error reduced ADI-FDTD methods," *IEEE Antennas Wireless Propag. Lett.*, Vol. 4, 323–325, 2005.
17. Shibayama, J., M. Muraki, J. Yamauchi, and H. Nakano, "Efficient implicit FDTD algorithm based on locally one-dimensional scheme," *Electron. Lett.*, Vol. 41, No. 19, 1046–1047, 2005.
18. Li, E., I. Ahmed, and R. Vahldieck, "Numerical dispersion analysis with an improved LOD-FDTD method," *IEEE Microw. Wireless Compon. Lett.*, Vol. 17, No. 5, 319–321, 2007.
19. Tan, E. L., "Unconditionally stable LOD-FDTD method for 3-DMaxwel's equations," *IEEE Microw. Wireless Compon. Lett.*, Vol. 17, No. 2, 85–87, 2007.

20. Ahmed, I., E. K. Chua, E. P. Li, and Z. Chen, "Development of the three-dimensional unconditionally stable LOD-FDTD method," *IEEE Trans. Antennas Propagat.*, Vol. 56, No. 11, 3596–3600, 2008.
21. Shibayama, J., A. Nomura, R. Ando, J. Yamauchi, and H. Nakano, "A frequency-dependent LOD-FDTD method and its application to the analyses of plasmonic waveguide devices," *IEEE J. Quantum Electron.*, Vol. 46, No. 1, 40–49, 2010.
22. Yu, W., R. Mittra, T. Su, Y. Liu, and X. Yang, *Parallel Finite-Difference Time-Domain Method*, Artech House, 2006.
23. Schneider, R. N., L. E. Turner, and M. M. Okoniewski, "Application of FPGA technology to accelerate the finite difference time-domain (FDTD) method," *FPGA'02 Proc. of the ACM/SIGDA Tenth Int. Symp. on Field-programmable Gate Arrays*, ACM, 2002.
24. Inman, M. J., A. Z. Elsherbeni, and C. E. Smith, "FDTD calculations using graphical processing units," *IEEE/ACES Int. Conf. on Wireless Communications and Applied Computational Electromagnetics*, 728–731, 2005.
25. Adams, S., J. Payne, and R. Boppana, "Finite difference time domain (FDTD) simulations using graphics processors," *DOD High Performance Computing Modernization Program Users Group Conf., IEEE*, 334–338, 2007.
26. Sypek, P., A. Dziekonski, and M. Mrozowski, "How to render FDTD computations more effective using a graphics accelerator," *IEEE Trans. Magn.*, Vol. 45, No. 3, 1324–1327, 2009.
27. Nagaoka, T. and S. Watanabe, "A GPU-based calculation using the three-dimensional FDTD method for electromagnetic field analysis," *Engineering in Medicine and Biology Society on Annu. Int. Conf. of the IEEE*, 327–330, 2010.
28. Zunoubi, M. R., J. Payne, and W. P. Roach, "CUDA implementation of TE^z-FDTD solution of Maxwell's equations in dispersive media," *IEEE Antennas Wireless Propag. Lett.*, Vol. 9, 756–759, 2010.
29. Tay, W. C., D. Y. Heh, and E. L. Ta, "GPU-accelerated fundamental ADI-FDTD with complex frequency shifted convolutional perfectly matched layer," *Progress In Electromagnetics Research M*, Vol. 14, 177–192, 2010.
30. Toivanen, J. I., T. P. Stefanski, N. Kuster, and N. Chavannes, "Comparison of CPML implementations for the GPU-accelerated FDTD solver," *Progress In Electromagnetics Research M*, Vol. 19, 61–75, 2011.

31. Lee, K. H., I. Ahmed, R. S. M. Goh, E. H. Khoo, E. P. Li, and T. G. G. Hung, "Implementation of the FDTD method based on lorentz-drude dispersive model on GPU for plasmonics applications," *Progress In Electromagnetics Research*, Vol. 116, 441–456, 2011.
32. Ma, L. C. and R. Mittra, "Parallel implementation of the periodic boundary condition (PBC) in the FDTD for the investigation of spatial filters," *Antennas and Propagation Society International Symposium, IEEE*, 1–4, 2008.
33. Mao, Y., B. Chen, B. Zhou, L. Cheng, and Q. Wu, "Parallel implementation of the split-field FDTD method for the analysis of periodic structure," *8'th International Symp. Antennas, Propagation and EM Theory, IEEE*, 875–878, 2008.
34. Roden, J. A., J. P. Skinner, and S. L. Johns, "Shielding effectiveness of three dimensional gratings using the periodic FDTD technique and CPML absorbing boundary condition," *IEEE/ACES Int. Conf. on Wireless Communications and Applied Computational Electromagnetics*, 128–131, 2005.
35. http://www.nvidia.com.
36. *NVIDIA CUDA C Programming Guide*, Version 3.2, NVIDIA, 2010.
37. Vial, A., A. Grimault, D. Macías, D. Barchiesi, and M. L. de la Chapelle, "Improved analytical fit of gold dispersion: Application to the modeling of extinction spectra with a finite-difference time-domain method," *Phys. Rev. B*, Vol. 71, No. 8, 085416, 2005.
38. Born, M. and E. Wolf, *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*, Pergamon, 1980.
39. Maier, S. A., *Plasmonics: Fundamentals and Applications*, Springer, New York, 2007.
40. Bohren, C. F. and D. R. Huffman, *Absorption and Scattering of Light by Small Particles*, Wiley Interscience, 1983.
41. Lee, K. S. and M. A. El-Sayed, "Gold and silver nanoparticles in sensing and imaging: Sensitivity of plasmon response to size, shape, and metal composition," *J. Phys. Chem. B*, Vol. 110, No. 39, 19220–19225, 2006.
42. Larsson, E. M., C. Langhammer, I. Zorić, and B. Kasemo, "Nanoplasmonic probes of catalytic reactions," *Science*, Vol. 326, 1091–1094, 2009.

43. Bera, M. and M. Ray, "Precise detection and signature of biological/chemical samples based on surface plasmon resonance (SPR)," *J. Opt.*, Vol. 38, No. 4, 232–248, 2009.
44. Shankaran, D. R., K. V. Gobi, and N. Miura, "Recent advancements in surface plasmon resonance immunosensors for detection of small molecules of biomedical, food and environmental interest," *Sensors and Actuators B*, Vol. 121, 158–177, 2007.
45. Ankerm, J. N., W. P. Hall, O. Lyandres, N. C. Shah, J. Zhao, and R. P. Van Duyne, "Biosensing with plasmonicnanosensors," *Nature Materials*, Vol. 7, 442–453, 2008.
46. Bingham, J. M., J. N. Anker, L. E. Kreno, and R. P. Van Duyne, "Gas sensing with high-resolution localized surface plasmon resonance spectroscopy," *J. Am. Chem. Soc.*, Vol. 132, 17358–17359, 2010.
47. Nau, D., A. Seidel, R. B. Orzekowsky, S. H. Lee, S. Deb, and H. Giessen, "Hydrogen sensor based on metallic photonic crystal slabs," *Opt. Lett.*, Vol. 35, No. 18, 3150–3152, 2010.
48. Homola, J., *Surface Plasmon Resonance Based Sensors*, Springer, Berlin, 2006.
49. Charles, D. E., M. Gara, D. Aherne, D. M. Ledwith, J. M. Kelly, W. J. Blau, and M. E. Brennan-Fournet, "Scaling of surface plasmon resonances in triangular silver nanoplate sols for enhanced refractive index sensing," *Plasmonics*, Vol. 6, 351–362, 2011.
50. Steinbrück, A., O. Stranik, A. Csaki, and W. Fritzsche, "Sensoric potential of gold-silver core-shell nanoparticles," *Anal. Bioanal. Chem.*, Vol. 401, 1241–1249, 2011.
51. Svedendahl, M., S. Chen, A. Dmitriev, and M. Käll, "Refractometric sensing using propagating versus localized surface plasmons: A direct comparison," *Nano Lett.*, Vol. 9, No. 12, 4428–4433, 2009.
52. Rodríguez-Cantó, P. J., M. Martínez-Marco, F. J. Rodríguez-Fortuño, B. Tomás-Navarro, R. Ortuño, S. Peransí-Llopis, and A. Martínez, "Demonstration of near infrared gas sensing using gold nanodisks on functionalized silicon," *Opt. Express*, Vol. 19, No. 8, 7664–7672, 2011.
53. Sosnova, M. V., N. L. Dmitruk, A. V. Korovin, S. V. Mamykin, V. I. Mynko, and O. S. Lytvyn, "Local plasmon excitations in one-dimensional array of metal nanowires for sensor applications," *Appl. Phys. B*, Vol. 99, 493–497, 2010.

54. Ameling, R., L. Langguth, M. Hentschel, M. Mesch, P. V. Braun, and H. Giessen, "Cavity-enhanced localized plasmon resonance sensing," *Appl. Phys. Lett.*, Vol. 97, 253116, 2010.
55. Ye, J. and P. Van Dorpe, "Improvement of figure of merit for gold nanobar array plasmonic sensors," *Plasmonics*, Vol. 6, 665–671, 2011.
56. Jiang, H., J. Markowski, and J. Sabarinathan, "Near-infrared optical response of thin film pH-sensitive hydrogel coated on a gold nanocrescent array," *Opt. Express*, Vol. 17, No. 24, 21802–21807, 2009.
57. Rodríguez-Fortuño, F. J., M. Martínez-Marco, B. Tomás-Navarro, R. Ortuño, J. Martí, A. Martínez, and P. J. Rodríguez-Cantó, "Highly-sensitive chemical detection in the infrared regime using plasmonic gold nanocrosses," *Appl. Phys. Lett.*, Vol. 98, 133118, 2011.
58. Kubo, W. and S. Fujikawa, "Au double nanopillars with nanogap for plasmonic sensor," *Nano Lett.*, Vol. 11, 8–15, 2011.
59. Lamprecht, B., G. Schider, R. T. Lechner, H. Ditlbacher, J. R. Krenn, A. Leitner, and F. R. Aussenegg, "Metal nanoparticle gratings: Influence of dipolar particle interaction on the plasmon resonance," *Phys. Rev. Lett.*, Vol. 84, No. 20, 4721–4724, 2000.
60. Zou, S. and G. C. Schatz, "Narrow plasmonic/photonic extinction and scattering line shapes for one and two dimensional silver nanoparticle arrays," *J. Chem. Phys.*, Vol. 121, No. 24, 2004.
61. Chu,Y., E. Schonbrun, T. Yang, and K. B. Crozier, "Experimental observation of narrow surface plasmon resonances in gold nanoparticle arrays," *Appl. Phys. Lett.*, Vol. 93, 181108, 2008.
62. Offermans, P., M. C. Schaafsma, S. R. K. Rodriguez, Y. Zhang, M. Crego-Calama, S. H. Brongersma, and J. G. Rivas, "Universal scaling of the figure of merit of plasmonic sensors," *ACS Nano*, Vol. 5, No. 6, 5151–5157, 2011.
63. Kravets, V. G., F. Schedin, A. V. Kabashin, and A. N. Grigorenko, "Sensitivity of collective plasmon modes of gold nanoresonators to local environment," *Opt. Lett.*, Vol. 35, No. 7, 956–958, 2010.