Efficient parallel LOD-FDTD method for Debye-dispersive media

Tadashi Hemmi, Fumie Costen, *Senior Member, IEEE*, Salvador Garcia, *Member, IEEE*, Ryutaro Himeno, Hideo Yokota, Mehshan Mustafa

***Index Terms*—Electromagnetic propagation in dispersive media, Electromagnetic fields, Finite difference methods, Time domain analysis, Numerical analysis, Distributed memory systems, Parallel programming**

***Abstract*—The Locally One-Dimensional Finite Difference Time Domain (LOD-FDTD) method is a promising implicit technique for solving Maxwell's Equations in numerical electromagnetics. This paper describes an efficient Message Passing Interface (MPI)-parallel implementation of the LOD-FDTD method for Debye-dispersive media. Its computational efficiency is demonstrated to be superior to that of the parallel ADI-FDTD method. We demonstrate the effectiveness of the proposed parallel algorithm in the simulation of a bio-electromagnetic problem: the deep brain stimulation (DBS) in human body.**

I. Introduction

Computational electromagnetic simulators have become an invaluable tool with applications ranging from telecommunications to radar systems and design of high speed electronic circuit boards as well as health-care device in biomedical engineering. There exist several approaches to solve Maxwell equations numerically. Among them, the Finite Difference Time Domain (FDTD) method has become the most widely used [1].

Even though the FDTD method is flexible and robust and easy to parallelise, its computational efficiency is limited by the Courant-Friedrich-Levy (CFL) stability condition [1]. This criterion imposes an upper limit on the maximum time-step Δt_{CFL} depending on the minimum space-step, which may lead to large numbers of FDTD iterations. We face such a situation, for instance, in highly resonant problems or in complex problems requiring very fine spatial discretization.

There are two major approaches to improve the computational efficiency of the FDTD method, developing either hardware or algorithmic methodology. The hardware approach involves the parallelisation of the computation by 1) using multiple cores in shared and/or distributed memory architectures (or Graphics Processing Units [2]) and 2) taking advantage of modern processors' features such as the register level parallelisation, *i.e.*, the Streaming SIMD Extensions (SSE)[3] or the Advanced Vector eXtensions (AVX).

Following the algorithmic approach, the development of implicit formulations of the FDTD method, overcoming the CFL limit, has attracted great attention in the recent literature. Most of them are based on some variation of the Crank-Nicolson fully implicit FDTD (CN-FDTD) method [4], [5]. The most usual ones are the Alternating Direction Implicit FDTD (ADI-FDTD) method [6], [7], and the Locally One-Dimensional FDTD (LOD-FDTD) method [8], [9], [10], [11], [12], [13], [14], [15]. Implicit CN-FDTD-based methods are not constrained by the CFL stability condition and permit time-steps Δt only constrained by an accuracy criterion [16], over the CFL limit (*i.e.,* $N_{\mathrm{CFL}} \triangleq \dfrac{\Delta t}{\Delta t_{\mathrm{CFL}}} > 1$), with potential computational gains. When they are combined with hardware acceleration techniques, they provide an efficient alternative to the classical FDTD method.

However, unlike parallel implementations of the FDTD method [17], CN-FDTD-based algorithms require data, which are not just one cell neighbours as is used for the explicit FDTD method, at each time-step. Thus parallel implementation of these methods using techniques such as Message Passing Interface (MPI) [18] results in a huge amount of data communication, hindering the scalability of the implementation.

There are efficient implementations of the parallel ADI-FDTD method [19], [20]. Nevertheless their efficiency is limited by the fact that these methods require the alternative use of data along two-space directions at each time-step. On the other hand, the LOD-FDTD method only requires data along one direction, making it more attractive for parallelisation.

In this paper we present a new parallel MPI algorithm for the LOD-FDTD method, including Debye-dispersive media, and demonstrate its efficacy with a biomedical problem: the simulation of a Deep Brain Stimulation (DBS) scenario, where the choice of precise positions of the transmitters and the stimulation waveforms, which are the key for the successful non-invasive stimulation, may be properly tuned with numerical simulations.

The rest of this paper is organized as follows. Section II shows the mathematical procedure for including Debye media in the LOD-FDTD method. Section III presents the approach to parallelise the LOD-FDTD method with Debye media on distributed memory architectures. The results of performance and scalability tests are given in Section IV and are compared with the parallel ADI-FDTD and classical FDTD methods. Section V demonstrates the applicability of the parallel LOD-

T. Hemmi, F. Costen, and M. Mustafa are with the School of Electrical and Electronic Engineering, The University of Manchester, U.K. (email: fumie.costen@manchester.ac.uk)

S. Garcia is with Faculty of Sciences, University of Granada, Granada, Spain

R. Himeno is Advanced Center for Computing and Communication, RIKEN, Saitama, Japan

H. Yokota and F. Costen are with the Image Processing Research Team, Center for Advanced Photonics, RIKEN, Saitama, Japan.

Color versions of the figures in this paper are available online at http://ieeexplore.ieee.org.

FDTD method to a real problem, more specifically, the DBS problem mentioned above. Section VI provides concluding remarks.

II. LOD-FDTD method for Debye media

The time dependent Maxwell curl equations can be expressed for Debye media in a material independent form as

$$\nabla \times \boldsymbol{H} = \frac{\partial \boldsymbol{D}}{\partial t}, \tag{1}$$

and

$$\nabla \times \boldsymbol{E} = -\mu \frac{\partial \boldsymbol{H}}{\partial t}, \tag{2}$$

together with the convolute constitutive relationship

$$\boldsymbol{D} = \epsilon \otimes \boldsymbol{E} \tag{3}$$

where $\boldsymbol{E}$, $\boldsymbol{H}$ and $\boldsymbol{D}$ are the electric and magnetic fields and the electric flux density; ϵ, and μ are dispersive permittivity and permeability, respectively. The symbol $\otimes$ signifies convolution.

The permittivity of a one-pole Debye medium in the frequency-domain can be expressed as

$$\epsilon = \epsilon_0(\epsilon_\infty + \frac{\epsilon_\mathrm{S} - \epsilon_\infty}{1 + \jmath\omega\tau_\mathrm{D}} + \frac{\sigma}{\jmath\omega\epsilon_0}) \tag{4}$$

where ϵ_∞ is the optical permittivity, ϵ_S is the static permittivity, τ_D is the characteristic relaxation time, σ is the static conductivity, and ω is the angular frequency.

The convolute relationship (3) can also be expressed in an Auxiliary Differential Equation (ADE) form [21] as

$$\begin{aligned}\frac{\partial^2(\tau_\mathrm{D}\boldsymbol{D})}{\partial t^2} + \frac{\partial \boldsymbol{D}}{\partial t} &= \\ \frac{\partial^2(\epsilon_0\epsilon_\infty\tau_\mathrm{D}\boldsymbol{E})}{\partial t^2} &+ \frac{\partial(\epsilon_0\epsilon_\mathrm{S} + \sigma\tau_\mathrm{D})\boldsymbol{E}}{\partial t} + \sigma\boldsymbol{E}\end{aligned} \tag{5}$$

As in the classical CN-FDTD algorithm, the LOD-FDTD method is developed by first using central differences for the time and space derivatives and averaging the fields affected by the curl operators in time. For instance, for the x component of (2) we find (similarly for the y and the z components of (2))

$$\begin{aligned}&\frac{1}{2}\Bigg\{\frac{E_z^{n+1}(i,j+1,k) - E_z^{n+1}(i,j,k)}{\Delta y} \\ &-\frac{E_y^{n+1}(i,j,k+1) - E_y^{n+1}(i,j,k)}{\Delta z} + \frac{E_z^{n}(i,j+1,k) - E_z^{n}(i,j,k)}{\Delta y} \\ &-\frac{E_y^{n}(i,j,k+1) - E_y^{n}(i,j,k)}{\Delta z}\Bigg\} = -\mu\frac{H_x^{n+1}(i,j,k) - H_x^{n}(i,j,k)}{\Delta t}\end{aligned} \tag{6}$$

where Δx, Δy, and Δz are the spatial discretization in the x, y and z directions, respectively. The basis of the LOD-FDTD procedure consists of splitting the curl operator into each space direction and building a split-step time-marching algorithm [22]. For example, (6) advances in the y direction using

$$\begin{aligned}&\frac{1}{2}\Bigg\{\frac{E_z^{n+\frac{2}{3}}(i,j+1,k) - E_z^{n+\frac{2}{3}}(i,j,k)}{\Delta y} \\ &+\frac{E_z^{n+\frac{1}{3}}(i,j+1,k) - E_z^{n+\frac{1}{3}}(i,j,k)}{\Delta y}\Bigg\} \\ &= -\mu\frac{H_x^{n+\frac{2}{3}}(i,j,k) - H_x^{n+\frac{1}{3}}(i,j,k)}{\Delta t} \\ &\therefore H_x^{n+\frac{2}{3}}(i,j,k) = H_x^{n+\frac{1}{3}}(i,j,k) - \\ &\frac{\Delta t}{2\mu\Delta y}\Bigg\{E_z^{n+\frac{2}{3}}(i,j+1,k) - E_z^{n+\frac{2}{3}}(i,j,k) \\ &+E_z^{n+\frac{1}{3}}(i,j+1,k) - E_z^{n+\frac{1}{3}}(i,j,k)\Bigg\}\end{aligned} \tag{7}$$

and in the z direction as

$$\begin{aligned}&\frac{1}{2}\Bigg\{-\frac{E_y^{n+1}(i,j,k+1) - E_y^{n+1}(i,j,k)}{\Delta z} \\ &-\frac{E_y^{n+\frac{2}{3}}(i,j,k+1) - E_y^{n+\frac{2}{3}}(i,j,k)}{\Delta z}\Bigg\} \\ &= -\mu\frac{H_x^{n+1}(i,j,k) - H_x^{n+\frac{2}{3}}(i,j,k)}{\Delta t} \\ &\therefore H_x^{n+1}(i,j,k) = H_x^{n+\frac{2}{3}}(i,j,k) + \\ &\frac{\Delta t}{2\mu\Delta z}\Bigg\{E_y^{n+1}(i,j,k+1) - E_y^{n+1}(i,j,k) \\ &+E_y^{n+\frac{2}{3}}(i,j,k+1) - E_y^{n+\frac{2}{3}}(i,j,k)\Bigg\}.\end{aligned} \tag{8}$$

In the same way, spatial and temporal discretization is applied to (1) and (5) and the term of $\sigma\boldsymbol{E}$ in (5) is averaged over time. After the discretization, the z component of (1) is expressed as

$$\begin{aligned}&\frac{1}{2}\Bigg\{\frac{H_y^{n+1}(i,j,k) - H_y^{n+1}(i-1,j,k)}{\Delta x} \\ &-\frac{H_x^{n+1}(i,j,k) - H_x^{n+1}(i,j-1,k)}{\Delta y} + \frac{H_y^{n}(i,j,k) - H_y^{n}(i-1,j,k)}{\Delta x} \\ &-\frac{H_x^{n}(i,j,k) - H_x^{n}(i,j-1,k)}{\Delta y}\Bigg\} = \frac{D_z^{n+1}(i,j,k) - D_z^{n}(i,j,k)}{\Delta t}\end{aligned} \tag{9}$$

and the z component of (5) is described as

$$\begin{aligned}&\tau_{\mathrm{D}(i,j,k)}\frac{D_z^{n+1}(i,j,k) - 2D_z^{n}(i,j,k) + D_z^{n-1}(i,j,k)}{(\Delta t)^2} \\ &+\frac{D_z^{n+1}(i,j,k) - D_z^{n}(i,j,k)}{\Delta t} \\ &= a_{1(i,j,k)}\frac{E_z^{n+1}(i,j,k) - 2E_z^{n}(i,j,k) + E_z^{n-1}(i,j,k)}{(\Delta t)^2} \\ &+a_{2(i,j,k)}\frac{E_z^{n+1}(i,j,k) - E_z^{n}(i,j,k)}{\Delta t} \\ &+\sigma_{(i,j,k)}\frac{E_z^{n+1}(i,j,k) + E_z^{n}(i,j,k)}{2}\end{aligned} \tag{10}$$

where $a_{1(i,j,k)} = \epsilon_0\epsilon_{\infty(i,j,k)}\tau_{\mathrm{D}(i,j,k)}$ and $a_{2(i,j,k)} = (\epsilon_0\epsilon_{\mathrm{S}(i,j,k)} + \sigma_{(i,j,k)}\tau_{\mathrm{D}(i,j,k)})$.

As before, (9) and (10) are split into the three different directions. For instance, the y part of (9) is

$$\begin{aligned}
&\frac{1}{2}\Big\{-\frac{H_x^{n+\frac{2}{3}}{}_{(i,j,k)} - H_x^{n+\frac{2}{3}}{}_{(i,j-1,k)}}{\Delta y} \\
&\quad -\frac{H_x^{n+\frac{1}{3}}{}_{(i,j,k)} - H_x^{n+\frac{1}{3}}{}_{(i,j-1,k)}}{\Delta y}\Big\} \\
&\quad = \frac{D_z^{n+\frac{2}{3}}{}_{(i,j,k)} - D_z^{n+\frac{1}{3}}{}_{(i,j,k)}}{\Delta t} \\
&\therefore D_z^{n+\frac{2}{3}}{}_{(i,j,k)} = D_z^{n+\frac{1}{3}}{}_{(i,j,k)} - \\
&\frac{\Delta t}{2\Delta y}\Big\{H_x^{n+\frac{2}{3}}{}_{(i,j,k)} - H_x^{n+\frac{2}{3}}{}_{(i,j-1,k)} \\
&\quad +H_x^{n+\frac{1}{3}}{}_{(i,j,k)} - H_x^{n+\frac{1}{3}}{}_{(i,j-1,k)}\Big\}.
\end{aligned} \tag{11}$$

and the y part of (10) is

$$\begin{aligned}
&\tau_{\mathrm{D}(i,j,k)}\frac{D_z^{n+\frac{2}{3}}{}_{(i,j,k)} - 2D_z^{n+\frac{1}{3}}{}_{(i,j,k)} + D_z^{n}{}_{(i,j,k)}}{(\frac{1}{2}\Delta t)^2} \\
&\quad +\frac{D_z^{n+\frac{2}{3}}{}_{(i,j,k)} - D_z^{n+\frac{1}{3}}{}_{(i,j,k)}}{\frac{1}{2}\Delta t} = \\
&a_{1(i,j,k)}\frac{E_z^{n+\frac{2}{3}}{}_{(i,j,k)} - 2E_z^{n+\frac{1}{3}}{}_{(i,j,k)} + E_z^{n}{}_{(i,j,k)}}{(\frac{1}{2}\Delta t)^2} \\
&\quad +a_{2(i,j,k)}\frac{E_z^{n+\frac{2}{3}}{}_{(i,j,k)} - E_z^{n+\frac{1}{3}}{}_{(i,j,k)}}{\frac{1}{2}\Delta t} \\
&\quad +\sigma_{(i,j,k)}\frac{E_z^{n+\frac{2}{3}}{}_{(i,j,k)} + E_z^{n+\frac{1}{3}}{}_{(i,j,k)}}{2} \\
&\therefore E_z^{n+\frac{2}{3}}{}_{(i,j,k)} = c_{1(i,j,k)}D_z^{n+\frac{2}{3}}{}_{(i,j,k)} \\
&\quad -c_{2(i,j,k)}D_z^{n+\frac{1}{3}}{}_{(i,j,k)} + c_{3(i,j,k)}D_z^{n}{}_{(i,j,k)} \\
&\quad +c_{4(i,j,k)}E_z^{n+\frac{1}{3}}{}_{(i,j,k)} - c_{5(i,j,k)}E_z^{n}{}_{(i,j,k)}
\end{aligned} \tag{12}$$

where

$$c_{1(i,j,k)} = \frac{\tau_{\mathrm{D}(i,j,k)} + \frac{1}{2}\Delta t}{a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) + \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}},$$

$$c_{2(i,j,k)} = \frac{2\tau_{\mathrm{D}(i,j,k)} + \frac{1}{2}\Delta t}{a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) + \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}},$$

$$c_{3(i,j,k)} = \frac{\tau_{\mathrm{D}(i,j,k)}}{a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) + \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}},$$

$$c_{4(i,j,k)} = \frac{2a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) - \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}}{a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) + \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}},$$

$$c_{5(i,j,k)} = \frac{\epsilon_0\epsilon_{\infty(i,j,k)}\tau_{\mathrm{D}(i,j,k)}}{a_{1(i,j,k)} + \frac{1}{2}\Delta t(a_{2(i,j,k)}) + \dfrac{(\frac{1}{2}\Delta t)^2\sigma_{(i,j,k)}}{2}}.$$

Substitution of (12) into (7) so as to remove $E_z^{n+\frac{2}{3}}{}_{(i+1,j,k)}$ and $E_z^{n+\frac{2}{3}}{}_{(i,j,k)}$ yields

$$\begin{aligned}
H_x^{n+\frac{2}{3}}{}_{(i,j,k)} &= -d_2c_{1(i,j+1,k)}D_z^{n+\frac{2}{3}}{}_{(i,j+1,k)} \\
&+d_2c_{2(i,j+1,k)}D_z^{n+\frac{1}{3}}{}_{(i,j+1,k)} - d_2c_{3(i,j+1,k)}D_z^{n}{}_{(i,j+1,k)} \\
&+d_2c_{1(i,j,k)}D_z^{n+\frac{2}{3}}{}_{(i,j,k)} - d_2c_{2(i,j,k)}D_z^{n+\frac{1}{3}}{}_{(i,j,k)} \\
&+d_2c_{3(i,j,k)}D_z^{n}{}_{(i,j,k)} + d_2c_{5(i,j+1,k)}E_z^{n}{}_{(i,j+1,k)} \\
&-(d_2c_{4(i,j+1,k)} + d_2)E_z^{n+\frac{1}{3}}{}_{(i,j+1,k)} \\
&+(d_2c_{4(i,j,k)} + d_2)E_z^{n+\frac{1}{3}}{}_{(i,j,k)} \\
&-d_2c_{5(i,j,k)}E_z^{n}{}_{(i,j,k)} + H_x^{n+\frac{1}{3}}{}_{(i,j,k)}.
\end{aligned} \tag{13}$$

where $d_2 = \frac{\frac{1}{2}\Delta t}{\mu\Delta y}$.

Inserting (13) into (11) to remove $H_x^{n+\frac{2}{3}}{}_{(i,j,k)}$ and $H_x^{n+\frac{2}{3}}{}_{(i,j-1,k)}$ yields

$$\begin{aligned}
&d_2c_{1(i,j+1,k)}D_z^{n+\frac{2}{3}}{}_{(i,j+1,k)} + d_2c_{1(i,j-1,k)}D_z^{n+\frac{2}{3}}{}_{(i,j-1,k)} \\
&\quad -(2d_2c_{2(i,j,k)} + \frac{\Delta y}{\frac{1}{2}\Delta t})D_z^{n+\frac{2}{3}}{}_{(i,j,k)} \\
&= d_2c_{2(i,j+1,k)}D_z^{n+\frac{1}{3}}{}_{(i,j+1,k)} - d_2c_{3(i,j+1,k)}D_z^{n}{}_{(i,j+1,k)} \\
&-(2d_2c_{2(i,j,k)} + \frac{\Delta y}{\frac{1}{2}\Delta t})D_z^{n+\frac{1}{3}}{}_{(i,j,k)} + 2d_2c_{3(i,j,k)}D_z^{n}{}_{(i,j,k)} \\
&+d_2c_{2(i,j-1,k)}D_z^{n+\frac{1}{3}}{}_{(i,j-1,k)} - d_2c_{3(i,j-1,k)}D_z^{n}{}_{(i,j-1,k)} \\
&\quad -(d_2c_{4(i,j+1,k)} + d_2)E_z^{n+\frac{1}{3}}{}_{(i,j+1,k)} \\
&\quad -(d_2c_{4(i,j-1,k)} + d_2)E_z^{n+\frac{1}{3}}{}_{(i,j-1,k)} \\
&+d_2c_{5(i,j+1,k)}E_z^{n}{}_{(i,j+1,k)} + d_2c_{5(i,j-1,k)}E_z^{n}{}_{(i,j-1,k)} \\
&+2(d_2c_{4(i,j,k)} + d_2)E_z^{n+\frac{1}{3}}{}_{(i,j,k)} - 2d_2c_{5(i,j,k)}E_z^{n}{}_{(i,j,k)} \\
&\quad +2H_x^{n+\frac{1}{3}}{}_{(i,j,k)} - 2H_x^{n+\frac{1}{3}}{}_{(i,j-1,k)}.
\end{aligned} \tag{14}$$

(14) forms a set of simultaneous equations. Thus $D_z^{n+\frac{2}{3}}{}_{(i,j,k)}$ is obtained by solving a tri-diagonal matrix structured using (14). $D_z^{n+\frac{2}{3}}{}_{(i,j,k)}$ and (12) produce $E_z^{n+\frac{2}{3}}{}_{(i,j,k)}$. This newly updated $E_z^{n+\frac{2}{3}}{}_{(i,j,k)}$ and (7) generate $H_x^{n+\frac{2}{3}}{}_{(i,j,k)}$. The remainder of the components in each direction are derived using the same approach. The following algorithm is the complete procedure for the LOD-FDTD method with Debye media:

1) x direction part
   a) implicitly calculate ${D_y}^{n+\frac{1}{3}}$ and ${D_z}^{n+\frac{1}{3}}$
   b) explicitly calculate ${E_y}^{n+\frac{1}{3}}$, ${E_z}^{n+\frac{1}{3}}$, ${H_z}^{n+\frac{1}{3}}$ and ${H_y}^{n+\frac{1}{3}}$
2) y direction part
   a) implicitly calculate ${D_z}^{n+\frac{2}{3}}$ and ${D_x}^{n+\frac{2}{3}}$
   b) explicitly calculate ${E_z}^{n+\frac{2}{3}}$, ${E_x}^{n+\frac{2}{3}}$, ${H_x}^{n+\frac{2}{3}}$ and ${H_z}^{n+\frac{2}{3}}$
3) z direction part
   a) implicitly calculate ${D_x}^{n+1}$ and ${D_y}^{n+1}$
   b) explicitly calculate ${E_x}^{n+1}$, ${E_y}^{n+1}$, ${H_y}^{n+1}$ and ${H_x}^{n+1}$

III. PARALLELISATION STRATEGY

A. Data partitioning approach

Typical parallelisation starts by dividing the computational domain between cores; for example, in slices along a single space direction as is shown in Fig.1. In the case of the explicit FDTD parallelisation, each core updates both $\boldsymbol{E}$ and $\boldsymbol{H}$ inside its slab at each time-step, afterwards sharing $\boldsymbol{H}$ at the interface between the slabs with the cores in charge of the adjacent slices, thus requiring only a one-to-one sending/receiving communication [17] on the interface planes.

When it comes to the LOD-FDTD method with Debye media more data-communication and synchronization are involved:

1) $\boldsymbol{D}$ and $\boldsymbol{E}$ partitioning: Partitioning D_y and E_y along the y axis works for the parallelisation of the computation of D_y and E_y in procedures 1a and 3a (described in the previous section). Similarly D_x and E_x are partitioned along the x axis. D_z and E_z are partitioned along the z axis.

2) $\boldsymbol{H}$ partitioning: There are two partitioning directions for each of H_x, H_y and H_z. For instance, H_z is calculated according to procedures 1b and 2b. At procedure 1b H_z is obtained using D_y and E_y. Since D_y and E_y are partitioned along y axis, H_z also needs to be partitioned along y axis. The data partitioning of H_z in procedure 1b is depicted in Fig. 1. At procedure 2b, H_z is obtained using D_x and E_x. Since D_x and E_x are partitioned along x axis, H_z also needs to be partitioned along x axis. Fig. 2 depicts the data partitioning of H_z at procedure 2b. By comparing Fig. 1 and Fig. 2, it is clearly seen that the H_z space allocated to the core 1 at procedure 1b does not include the entire H_z space allocated to the core 1 at procedure 2b. Thus H_z needs to be communicated between cores to carry out computations in procedure 2b.

In summary, in this data partitioning scheme, no data communication is required for $\boldsymbol{D}$ and $\boldsymbol{E}$ but $\left(1 - \dfrac{1}{m}\right) \times 100$ % of $\boldsymbol{H}$

of the entire FDTD space needs to be exchanged between the cores, where m is the number of cores involved in the computation.

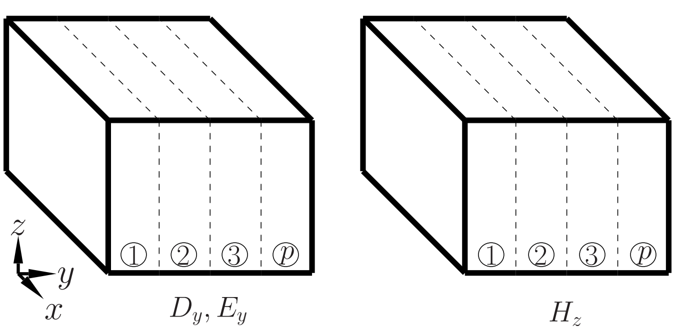


Fig. 1. Data partitioning scheme for D_y, E_y and H_z. D_y, E_y and H_z are all partitioned along the y axis. Core numbers are shown in circles.

B. Data communication

For simplicity, only the data communication required for H_z, just after procedure 1b, is discussed here. At the end of procedure 1b, each core communicates with all the other cores in order to acquire updated values of H_z to update D_x and E_x in procedure 2a.

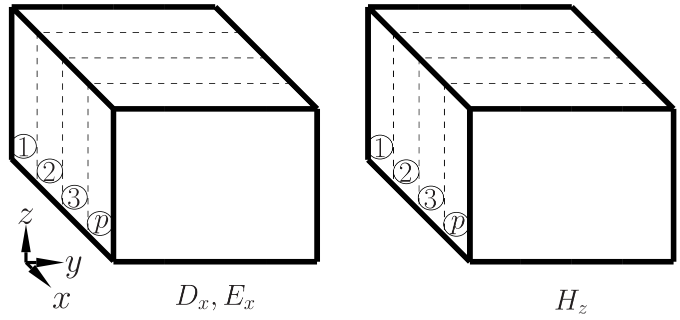


Fig. 2. D_x and E_x are partitioned along the x axis.

The data transfer required to acquire all the H_z blocks needed by core 1 is depicted in Fig. 3. Fig. 4 illustrates the H_z blocks sent from core 1 to all the other cores.

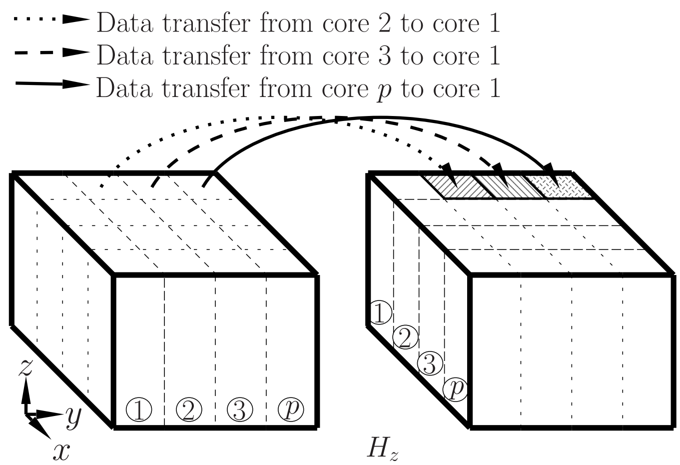


Fig. 3. Data transfers from all other cores to core 1. Arrows depict the direction of data transfer.

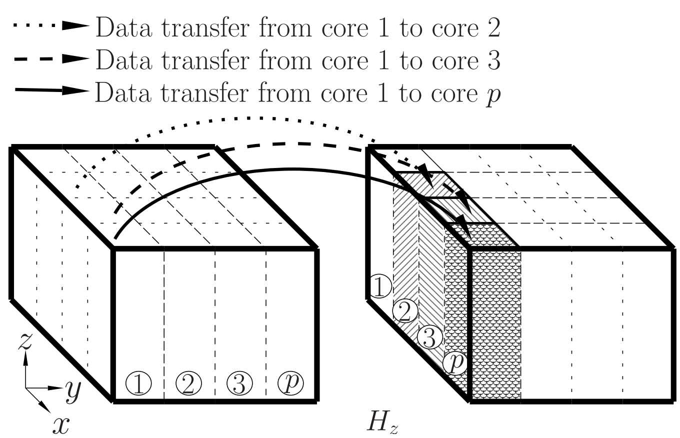


Fig. 4. Data transfers from core 1 to all other cores.

The major performance driver in this data partitioning strategy is the parallelisation of $\boldsymbol{D}$ and $\boldsymbol{E}$ calculations in all three direction parts. Furthermore, no memory rotation is required when data is transferred from one core to another. The

Stage	C1	C2	C3	C4	C5
1st	C5	C4		C2	C1
2nd		C5	C4	C3	C2
3rd	C2	C1	C5		C3
4th	C3		C1	C5	C4
5th	C4	C3	C2	C1	

TABLE I
A LOOK-UP TABLE FOR SCHEDULING INTER-CORE COMMUNICATION IN CASE OF 5 CORES INVOLVED.

drawback of this scheme is that all the cores in the computation have to communicate with each other in order to exchange recently updated values of $\boldsymbol{H}$, not only those on the interface plane but also those in the FDTD space, unlike the transfer in the classical FDTD method.

C. Implementation

MPI is the most commonly used method of implementing parallel algorithms on distributed memory architectures[18]. The parallel LOD-FDTD algorithm for Debye media was implemented using the Fortran programming language and the MPICH 2.0 library [23].

Instead of using the library `MPI_Alltoall` routine, our implementation involves a custom routine based on `MPI_Send` and `MPI_Recv` routines to order communication and send data between all the cores.

The custom routine involves a look-up table which is automatically generated based on the number of cores involved in the simulation. For example when there are 5 cores for computation, TABLE I is automatically generated before the FDTD iteration at each core. Each core communicates with the others at the stage suggested by TABLE I. TABLE I states that there are 5 stages of communication at the end of procedures 1b, 2b and 3b and the communication is depicted in Fig. 5.

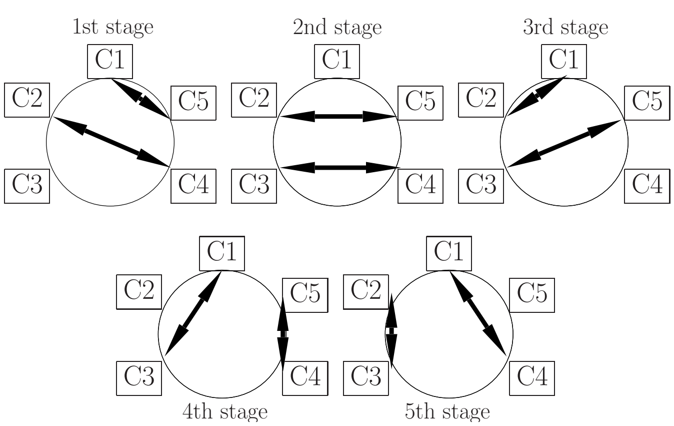


Fig. 5. One-to-one communication between cores at each stage of TABLE I in the case of 5 cores involved. C1 means core 1. The arrow between C1 and C5 means the data communication between C1 and C5. A set of these 5 stages is carried out at the end of procedures 1b, 2b and 3b.

IV. COMPUTATIONAL EFFICIENCY

A scalability test of the parallel LOD-FDTD method with Debye media was carried out on RIKEN's Massively Parallel Cluster (MPC), which is a part of the RIKEN Integrated Cluster of Clusters (RICC) facility. The MPC cluster consists of a total of 1048 PRIMERGY RX200S5 [1], and each node consists of two quad-core Intel Xeon processors and contains 12GB RAM. The code was compiled using the Fujitsu Fortran compiler (configured to use the maximum level of optimization).

Benchmarking tests on the parallel LOD-FDTD code were conducted in both single and double precision. A computational domain of 140^3 cells per core was used to keep the computational load per core constant, independent of m. m is varied from 2 to 64.

In order to measure the performance of the MPI implementation correctly, we avoided communication between cores within a node since such communication is faster than the communication between two different nodes. Thus only one core was used within each node. In other words, job submission was carefully tailored so that any cores, which participated in the parallel computation, could not share a motherboard.

A total of one hundred FDTD time-steps were calculated in each simulation. Each simulation was repeated four times to average the elapsed time. Fig. 6 plots the number of processed FDTD cells per second as a function of the number of cores. A computational efficiency figure-of-merit $\mathcal{R}$ was defined by normalizing the computational speed to that found where only one core was used for computation.

$$\mathcal{R} = \frac{\text{cells per second using } m\text{-nodes}}{\text{(cells per second in 2 nodes)/2}}.$$

Fig. 7 shows $\mathcal{R}$ of the parallel LOD-FDTD method with Debye media.

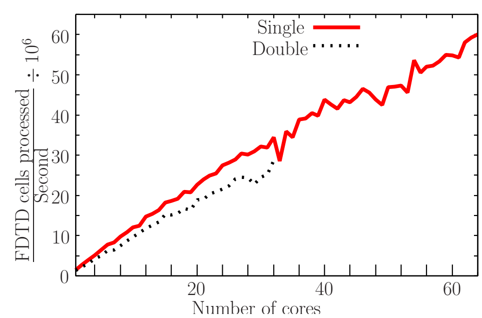


Fig. 6. Number of cells per second processed in single and double precision.

In single precision $\mathcal{R}$ steadily and gradually deteriorates to 75 % of the ideal case when the number of cores is 40 and, in double precision, 66 % of the ideal case when the number of cores is 32.

We also studied how our implementation of the parallel LOD-FDTD method with Debye media compares to another implicit scheme, the Alternating Direction Implicit Finite Difference Time Domain (ADI-FDTD) method [20]. Its computational efficiency had been measured by fixing the FDTD space size to 500^3 cells, recording the run time for 8 to 32

[1] http://www.pcpro.co.uk/reviews/servers/354196/fujitsu-primergy-rx200-s5/specifications

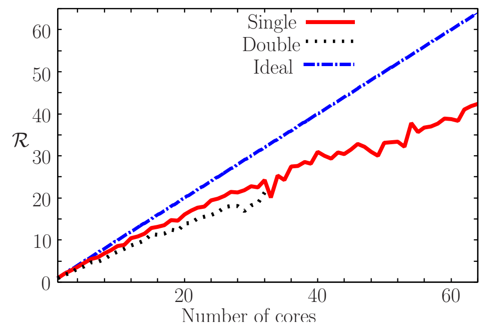


Fig. 7. $\mathcal{R}$ of the parallel LOD-FDTD method with Debye media. The "Ideal" line represents linear increase in efficiency with respect to increase in number of cores. The "Single" and "Double" lines present $\mathcal{R}$ from the single and double precision computation, respectively.

cores. In order to allow comparison between the the ADI-FDTD and LOD-FDTD methods using published data [20], we defined a speedup factor as

$$\mathcal{S} = \frac{\text{Run time with 8 cores}}{\text{Run time}}.$$

Using our parallel LOD-FDTD code, we carried out the same simulations as [20]. Fig. 8 presents $\mathcal{S}$ for the LOD-FDTD and the ADI-FDTD methods. As the number of cores is increased the parallel LOD-FDTD method performs better than the parallel ADI-FDTD method. This is mainly due to the fact that only $\boldsymbol{H}$ needs to be communicated between the cores. The rest of the computations, in particular the $\boldsymbol{D}$ calculations which require solution of a linear system are performed in parallel in the LOD-FDTD method. The fact that the LOD-FDTD method has a lower communication overhead than the ADI-FDTD method also contributes to low $\mathcal{S}$ value of the LOD-FDTD method.

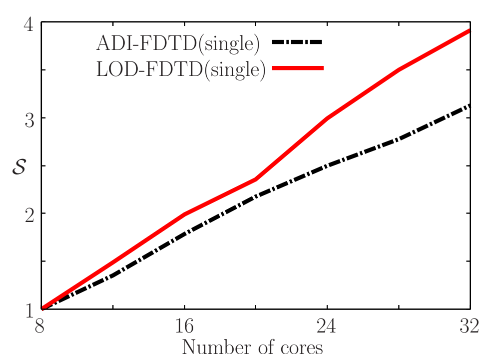


Fig. 8. S obtained for FDTD space size of 500^3 and up to 32 cores in case of the parallel LOD-FDTD method with Debye media and the parallel ADI-FDTD method in single precision.

We also compared the parallel LOD-FDTD method with an MPI parallel FDTD method we have developed, also including Debye-dispersive media. The parallel LOD-FDTD implementation is roughly 8.4 times slower than the parallel FDTD simulator (tested using 64 cores for a 560^3 cell problem). As a consequence, the LOD-FDTD method allows gains over the classical FDTD method, as far as we can take a CFL number $N_{\rm CFL}$ higher than 8.4 while maintaining accuracy under control.

Regarding the accuracy, [15] involved an extensive study on the numerical dispersion of the 3D LOD-FDTD method. [15] showed that the error of the LOD-FDTD method rises to 5 % at $N_{\rm CFL} = 10$ and 10 % at $N_{\rm CFL} = 20$ when a wavelength is sampled by 100 points. Keeping $N_{\rm CFL}$ constant, the error decreases with increases in the spatial sampling resolution. Thus we can set $N_{\rm CFL}$ as high as 20 as long as the acceptable error is above the error predicted by [15].

V. Applications

A Deep Brain Stimulation (DBS) scenario has been simulated to demonstrate a practical application of the LOD-FDTD method. DBS is a surgical operation which involves implantation of an electrode in the brain to deliver electrical stimulation to a precisely targeted area. In the treatment of Parkinson's disease the SubThalamic Nucleus (STN) of a patient is stimulated by electromagnetic field [24].

Although DBS can provide therapeutic benefits for Parkinson's disease, it has a number of risks, such as infection, skin erosion, electrode fracture, electrode dislocation, hardware failure and associated difficulties due to the invasive electrode implantation [25]. Therefore the application of electromagnetic wave, which may be able to provide non-invasive STN stimulation, could be an alternative method of treatment. In order to focus the electromagnetic energy on the targeted location inside the head, the waveforms on the skull, which originated from the invasive stimulation of STN, have to be known. Thus numerical simulation of wave propagation from inside the brain to the skull is performed.

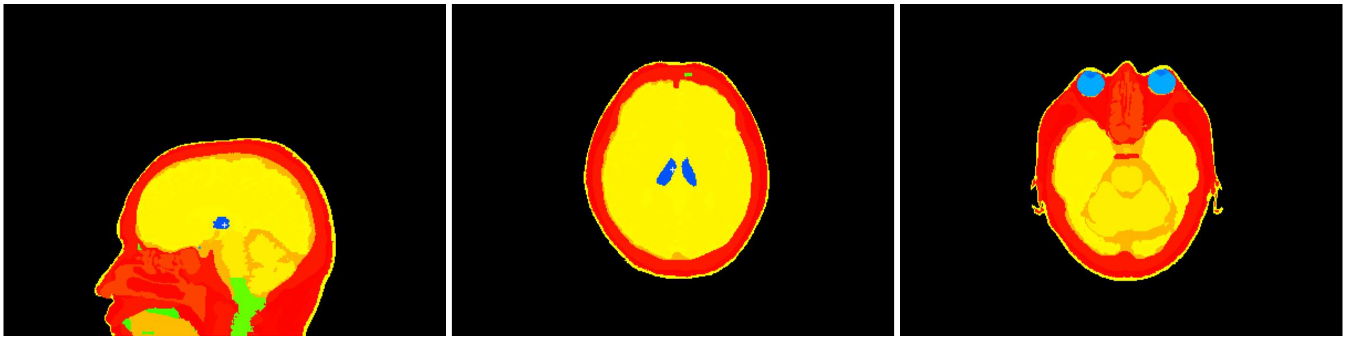
The radio environment of this practical numerical simulation was set using the Digital Human Phantom (DHP), as in [26], provided by RIKEN (Saitama, Japan), whose usage was approved by the RIKEN ethical committee. The spatial resolution of the DHP was 1mm in all three directions. The DHP consisted of $265 \times 490 \times 1682$ voxels and 53 distinct tissues. We fitted the one-pole Debye media parameters of human tissues [27] using the measurements provided by the U.S. Air force. TABLE II lists some of this data. The one-pole media parameters for all 53 human tissues are available in [28].

Tissue	Code	σ [S/m]	$\epsilon_{\rm S}$	ϵ_∞	$\tau_{\rm D}$ [ps]
white matter	2	0.35	41.28	24.37	33.59
midbrain	4	0.35	41.28	24.37	33.59
eyeball	5	1.45	67.71	10.31	8.27
thalamus	10	0.60	56.44	33.06	35.20
tongue	13	0.69	56.52	28.26	20.45

TABLE II
Example of one-pole Debye media parameters of the human tissues around the head.

The head part above the shoulders was placed in free-space, in a total domain $900 \times 900 \times 300$ cells (*i.e.,* 90cm $\times$90cm $\times$30 cm), meshed with a constant space-step of 1mm.

We placed a z-directed hard source [29] at $(i_{\rm src}, j_{\rm src}, k_{\rm src}) = (450, 450, 550)$ which corresponds to the center of the thala-



(a) $x = i_{\mathrm{src}}$ plane. $+$ shows the excitation point which is in the middle of STN.

(b) $z = k_{\mathrm{src}}$ plane. $+$ is situated in the left STN.

(c) $z = k_{\mathrm{src}} - 28$ plane which is 28 mm below the excitation location in z direction.

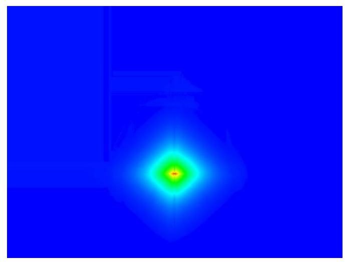
Fig. 9. Radio environment setting using the DHP and the excitation point in the FDTD space. An individual colour, available in the on-line version, is allocated to each human tissue.

mus in the DHP as shown in TABLE II, excited with a Gaussian pulse with spectral content up to 3.82 GHz (according to the definition given in [29]), which corresponds to a free–space wavelength of 79 mm.

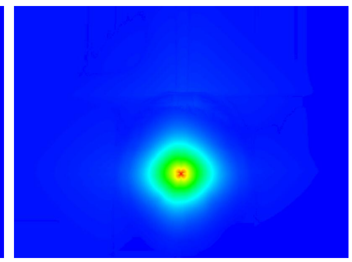
Fig. 9a and Fig. 9b show the excitation point with $+$ mark on the $x = i_{\mathrm{src}}$ plane and $z = k_{\mathrm{src}}$ plane, respectively. The eye balls exists $10 \sim 35$ mm below the excitation point. Fig. 9c is 28 mm below Fig. 9b. The DHP closes its eyes with the eyelids.

The E_z distribution on the $z = k_{\mathrm{src}}$ plane obtained from the LOD-FDTD computation with $N_{\mathrm{CFL}} = 20$ is visualized in Fig. 10. Its orientation is the same as Fig. 9c but the cropped area differs from Fig. 9c. The signal comes out of eyes first and second from the left ear (due to the excitation of the left STN) and reach the $z = k_{\mathrm{src}}$ observation-plane at about $300\Delta t_{\mathrm{CFL}}$ and $400\ \Delta t_{\mathrm{CFL}}$, respectively. An animation of the detailed movement of the electromagnetic wave propagation from this simulation is presented in colour at http://personalpages.manchester.ac.uk/staff/fumie.costen/LODFDTDpropagation.html These results were obtained about 2.4 times faster than the in-house parallel explicit FDTD code.

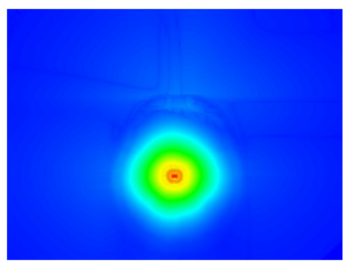
We have performed the same computation, varying the N_{CFL} parameter between 1 and 20, and we have calculated the error of the LOD-FDTD method with respect to the usual explicit FDTD method (Fig. 11). For $N_{\mathrm{CFL}} = 8.4$, the error of the parallel LOD-FDTD method is found to be around 6 % , requiring the same computational time than the parallel explicit FDTD method to reach a given physical time. For values over 8.4, the parallel LOD-FDTD method presents gains in the computational time over FDTD, linearly increasing with N_{CFL}. For instance, for $N_{\mathrm{CFL}} = 2.4 \cdot 8.4 = 20$, the parallel LOD-FDTD takes 2.4 times less CPU time than FDTD to reach a solution, while the error becomes, as expected, 14.2 % .



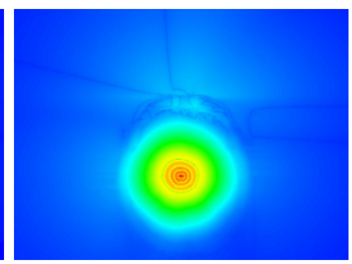
(a) $100 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 1.



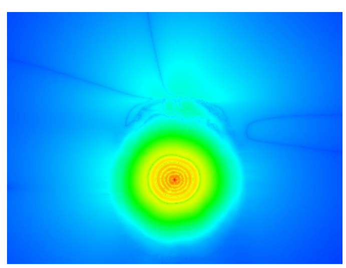
(b) $200 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 15.



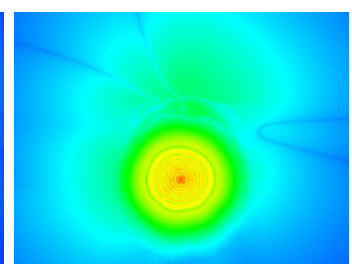
(c) $300 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 3.5.



(d) $400 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 2.4.



(e) $500 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 2.



(f) $600 \times \frac{10^{-11}}{3\sqrt{3}}$ seconds. Maximum E_z on the plane is 1.6.

Fig. 10. Contour plot of E_z on $z = k_{\mathrm{src}}$ plane, the computational results from the LOD-FDTD method with $N_{\mathrm{CFL}} = 20$ at $100 \cdot n \cdot \frac{10^{-11}}{3\sqrt{3}}$ seconds where $n = 1 \sim 6$. The orientation is the same as in Fig. 9b. They are obtained 2.4 times faster than the parallel explicit FDTD method. In the on-line version, the area in red has the maximum field values and the one in blue has the minimum field values.

VI. Conclusion

The Locally One Dimensional FDTD method is an alternative to the classical FDTD method permitting us to model electrically small details with a temporal sampling larger than that governed by the CFL stability condition. In this paper we

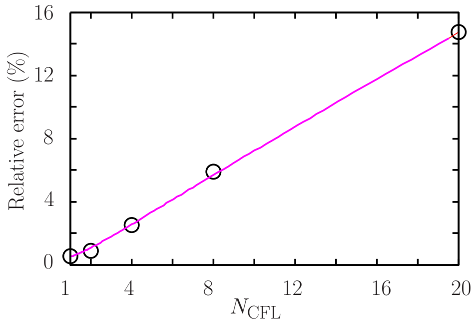


Fig. 11. Error of LOD-FDTD method relative to the explicit FDTD method when $N_{\rm CFL}$ changes from 1 to 20.

presented an efficient parallel implementation of the LOD-FDTD method, including Debye dispersion treatment. Since it is only one order of magnitude slower than the classical parallel FDTD method, it becomes advantageous for problems where the time-step can be increased ten times or more above the FDTD stability limit ($N_{\rm CFL} > 10$), assuming that space and time sampling are set appropriately.

The performance of the parallel LOD-FDTD code was examined showing an overall good scalability, better than the parallel ADI-FDTD method, thanks to the fact that the LOD-FDTD method requires lower communications between cores than the ADI-FDTD method. The LOD-FDTD method achieves a consistent rise in performance using up to 40 cores. However, some level of saturation in efficiency is observed when more than 40 cores are utilized.

We have demonstrated the utility of this tool in a complex bio-electromagnetic problem requiring the simulation of the deep brain stimulation in the human body, densely meshed in space. The results were obtained 2.4 times faster than the parallel FDTD method using an identical computational environment.

Acknowledgments

The work described in this paper and the research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013, under grant agreement no 205294 (HIRF SE project), and from the Spanish National Projects TEC2010-20841-C04-04, CSD2008-00068, and the Junta de Andalucia Project P09-TIC-5327.

References

[1] A. Taflove and S. Hagness, *Computational Electrodynamics: The Finite–Difference Time–Domain Method, third ed.* Boston, MA: Artech House, 2005.

[2] M. Livesey, J. Stack, F. Costen, T. Nanri, N. Nakashima, and S. Fujino, "Development of a CUDA implementation of the 3D FDTD method," *IEEE*, vol. 54, no. 5, pp. 186–195, October 2012.

[3] M. Livesey, F. Costen, and X. Yang, "Performance of Streaming SIMD Extensions instructions for the FDTD computation," *IEEE Antennas and Propagation Magazine*, vol. 54, no. 3, pp. 160–168, June 2012.

[4] Y. Yang, R. Chen, and E. Yung, "The unconditionally stable Crank Nicolson FDTD method for three-dimensional Maxwell's equations," *Microwave and Optical Technology Letters*, vol. 48, pp. 1619–1622, 2006.

[5] H. K. Rouf, F. Costen, S. G. Garcia, and S. Fujino, "On the solution of 3-D frequency dependent Crank–Nicolson FDTD scheme," *Journal of Electromagnetic Waves and Applications*, vol. 23, pp. 2163–2175, 2009.

[6] F. Zheng, Z. Chen, and J. Zhang, "A finite difference time domain method without the Courant stability conditions," *IEEE Microwave and Guided Wave Letters*, vol. 9, pp. 441–443, 1999.

[7] T. Namiki, "3-D ADI-FDTD method-unconditionally stable time-domain algorithm for solving full vector Maxwell's equations," *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 1743–1748, 2000.

[8] J. Shibayama, M. Muraki, J. Yamauchi, and H. Nakano, "Efficient implicit FDTD algorithm based on locally one-dimensional scheme," *Electronics Letters*, vol. 41(19), p. 10461047, September 2005.

[9] J. Shibayama, R. Takahashi, J. Yamauchi, and H. Nakano, "Frequency-dependent LOD-FDTD implementations for dispersive media," *Electronics Letters*, vol. 42(19), pp. 1084–1086, September 2006.

[10] J. Shibayama, R. Ando, J. Yamauchi, and H. Nakano, "A 3-D LOD-FDTD method for the wideband analysis of optical devices," *Journal of Lightwave Technology*, vol. 29(11), pp. 1652–1658, June 2011.

[11] E. L. Tan, "Unconditionally stable LOD-FDTD method for 3-D Maxwell's equations," *IEEE Microwave Theory and Wireless Component Letters*, vol. 17(2), p. 8587, February 2007.

[12] ——, "Fundamental schemes for efficient unconditionally stable implicit finite difference time domain methods," *IEEE Transactions on Antennas and Propagation*, vol. 56(1), pp. 170–177, January, 2008.

[13] E. Li, I. Ahmed, and R. Vahldieck, "Numerical dispersion analysis with an improved LOD-FDTD method," *IEEE Microwave and Wireless Components Letters*, vol. 17(5), pp. 319–321, May 2007.

[14] I. Ahmed, E.-K. Chua, E.-P. Li, and Z. Chen, "Development of the three-dimensional unconditionally stable LOD-FDTD method," *IEEE Transactions on Antennas and Propagation*, vol. 56(11), pp. 3596–3600, November 2008.

[15] I. Ahmed, E.-K. Chua, and E. Li, "Numerical dispersion analysis of the unconditionally stable three-dimensional LOD-FDTD method," *IEEE Antennas and Propagation*, vol. 58(12), pp. 3983–3989, December 2010.

[16] S. G. Garcia, T. Lee, and S. Hagness, "On the accuracy of the ADI-FDTD method," *IEEE Antennas and Wireless Propagation Letters*, vol. 1, pp. 31–34, 2002.

[17] W. Yu, R. Mittra, T. Su, Y. Liu, and X. Yang, *Parallel Finite-Difference Time-Domain Method*. Artech House Publishers, 2006.

[18] M. P. I. Forum, "MPI: A message-passing interface standard," Tech. Rep., www-unix.mcs.anl.gov/mpi/.

[19] H. Jordan, S. Bokhari, S. Staker, J. Sauer, M. ElHelbawy, and M. Piket-May, "Experience with ADI-FDTD techniques on the Cray MTA supercomputer," *Proc. SPIE*, vol. 4528, pp. 68–76, 2001.

[20] T. Stefaski and T. D. Drysdale, "Parallel implementation of the adi-fdtd method," *Microwave and Optical Technology Letters*, vol. 51, no. 5, pp. 1298–1304, 2009.

[21] R. Joseph, S. Hagness, and A. Taflove, "Direct time integration of Maxwell's equations in linear dispersive media with absorption for scattering and propagation of femtosecond electromagnetic pulses," *Optics Letters*, vol. 16(18), pp. 1412–1414, 1991.

[22] J. Lee and B. Fornberg, "A split step approach for the 3-D Maxwell's equations," *Journal of Computations and Applied Mathematics*, vol. 158(2), p. 485505, September 2003.

[23] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A high-performance, portable implementation of the MPI message passing interface standard," *Parallel Computing*, vol. 22, pp. 789–828, 1996.

[24] C. McIntyre, M. Savasta, L. K. Goff, and J. L. Vitek, "Uncovering the mechanism(s) of action of deep brain stimulation: activation, inhibition, or both," *Clinical Neurophysiology*, vol. 115, no. 6, pp. 1239 – 1248, 2004.

[25] M. I. Hariz, "Complication of deep brain stimulation surgery," *Movement Disorders*, vol. 17, pp. 162 – 166, 2002.

[26] M. Abalenkovs, F. Costen, J.-P. Bérenger, R. Himeno, H. Yokota, and M. Fujii, "Huygens subgridding for 3D frequency-dependent finite-difference time-domain method," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 9, pp. 4336–4344, 2012.

[27] T. Wuren, T. Takai, M. Fujii, and I. Sakagami, "Effective 2-Debye-pole FDTD model of electromagnetic interaction between whole human body and UWB radiation," *IEEE Microwave and Wireless Components Letters*, vol. 17(7), pp. 483–485, 2007.

[28] The RIKEN webpage, media parameters for the Debye relaxation model. Accessed: Jul. 3, 2013. [Online]. Available: http://cfd-duo.riken.jp/cbms-mp/

[29] F. Costen, J.-P. Berenger, and A. Brown, "Comparison of FDTD hard source with FDTD soft source and accuracy assessment in Debye

media," *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 7, pp. 2014 –2022, July 2009.

Tadashi Hemmi was born in Tochigi, Japan in 1970. He received the B.E. and M.Phil. degrees in Electrical Engineering from the University of Manchester and is currently pursuing the Ph.D. degree. His current research interests include computational electromagnetics.

Fumie Costen (M'07) received the B.Sc. degree, the M.Sc. degree in electrical engineering and the Ph.D. degree in Informatics, all from Kyoto University, Japan. From 1993 to 1997 she was with Advanced Telecommunication Research International, Kyoto, where she was engaged in research on direction-of-arrival estimation based on Multiple SIgnal Classification algorithm for 3-D laser microvision. She filed three patents from the research in 1999 in Japan. She was invited to give 5 talks in Sweden and Japan during 1996-2014. From 1998 to 2000, she was with Manchester Computing in the University of Manchester, U.K., where she was engaged in research on metacomputing and has been a Lecturer since 2000. Her research interests include computational electromagnetics in such topics as a variety of the finite difference time domain methods for microwave frequency range and high spatial resolution and FDTD subgridding and boundary conditions. She filed a patent from the research on the boundary conditions in 2012 in the U.S.A. Her work extends to the hardware acceleration of the computation using general-purpose computating on graphics processing units, Streaming Single Instruction Multiple Data Extension and Advanced Vector eXtentions instructions. Dr. Costen received an ATR Excellence in Research Award in 1996 and a best paper award from 8th International Conference on High Performance Computing and Networking Europe in 2000.

Salvador Garcia was born in 1966 in Baeza, Spain. He received the M.S. and Ph.D. degrees (with extraordinary award) in physics from the University of Granada, Granada, Spain, in 1989 and 1994, respectively. In 1999, he joined the Department of Electromagnetism and Matter Physics, University of Granada as an Assistant Professor (qualified for Full Professor since 2012). He has published over 50 refereed journal articles and book chapters, and over 80 conference papers and technical reports, and participated in several national and international projects with public and private funding. He has received grants to stay as a Visiting Scholar at the University of Duisburg (1997), the Institute of Mobile and Satellite Communication Techniques (1998), the University of Wisconsin-Madison (2001), and the University of Kentucky (2005). His current research interests include computational electromagnetics, electromagnetic compatibility, terahertz technologies, microwave imaging and sensing (GPR), bioelectromagnetics, and antenna design.

Ryutaro Himeno received his Doctor of Engineering degree from the University of Tokyo in 1988. In 1979, he joined Central Research Laboratories, Nissan Motor Co., Ltd., Yokosuka, Japan, where he has been engaged in the research of applying Computational Fluid Dynamics analysis to the car aerodynamic development. In 1998, he joined RIKEN and is the director of Advanced Center for Computing and Communication and had been the deputy program director of the Next Generation Computational Science Research Program at RIKEN till April, 2013. He is also a visiting professor at Hokkaido University, Kobe University and Tokyo Denki University. He currently studies Computational Bioengineering, High Performance Computing and blood flows of human bodies. He was a winner of Nikkei Science, Computer Visualization Contest in 2000 and Scientific Visualization Contest in 1996, and received JSME Computational Mechanics Division Award in 1997 and JSME Youth Engineer Award in 1988. He has also received the Paper Award by NICOGRAPH in 1993, Giga FLOPS Award by CRAY Research Inc. in 1990 and other awards.

Hideo Yokota received his Doctor of Engineering degree from the University of Tokyo in 1999. In 1993, he joined Higuchi Ultimate Mechatronics Project, Kanagawa Academy of Science and Technology, Kawasaki, Japan. In 1999, he joined RIKEN and is the contract researcher of Computational biomechanics unit. 2004-2012 Bio-research Infrastructure Construction Team Leader, VCAD sytem research program, RIKEN. 2007-2012 Cell-scale Research and Development Team Leader,Research Program for Computational Science, RIKEN. 2013-present Image Processing Research Team Leader, Center for Advanced Photonics,RIKEN. He is also a visiting Professor at Hokkaido University, Kobe University, Tokai University and the Tokyo University of Agriculture and Technology. He currently studies biomedical imaging and image processing to the biomedical simulation. Bioimaging Society, Best Image Award, 2005. The Commendation for Science and Technology by the Minister of Education, Culture, Sports, Science and Technology , The Young Scientists Prize, 2008.