Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 4 (2011) 2337–2343

Procedia Computer Science

International Conference on Computational Science, ICCS 2011

A Multi-Scale Electromagnetic Particle Code with Adaptive Mesh Refinement and Its Parallelization

Hideyuki Usui[a]*, Masanori Nunami[b], Toseo Moritaka[a], Tatsuki Matsui[a], Yohei Yagi[a]

[a]*Graduate School of System Informatics, Kobe University, 1-1 Rokko-dai Nada-ku, Kobe 657-8501, JAPAN*
[b]*National Institute for Fusion Science, 322-6, Oroshi-cho Toki, Gifu 509-5292, JAPAN*

Abstract

Space plasma phenomena occur in multi-scale processes from the electron scale to the magnetohydrodynamic scale. In order to investigate such multi-scale phenomena including plasma kinetic effects, we started to develop a new electromagnetic Particle-In-Cell (PIC) code with Adaptive Mesh Refinement (AMR) technique. AMR can realize high-resolution calculation saving computer resources by generating and removing hierarchical cells dynamically. In the parallelization, we adopt domain decomposition method and for good locality preserving and dynamical load balancing, we will use the Morton ordered curve. In the PIC method, particle calculation occupies most of the total calculation time. In our AMR-PIC code, time step intervals are also refined. To realize the load balancing between processes in the domain decomposition scheme, it is the most essential to consider the number of particle calculation loops for each cell among all hierarchical levels as a work weight for each processor. Therefore, we calculate the work weights based on the cost of particle calculation and hierarchical levels of each cell. Then we decompose the domain according to the Morton curve and the work weight, so that each processor has approximately the same amount of work. By performing a simple one-dimensional simulation, we confirmed that the dynamic load balancing is achieved and the computation time is reduced by introducing the dynamic domain decomposition scheme.

"Keywords: Particle-In-Cell, Adaptive Mesh Refinement, Domain Decomposition, Space plasma"

1. Introduction

Full Particle-In-Cell (PIC) simulation [1] is a powerful simulation method to investigate microscopic phenomena occurring in space and laboratory plasmas because it can treat electron as well as ion kinetics with no fluid approximation. Spatial grid size and time step interval are basically determined by the Debye length, which implies a characteristic spatial length for the electrostatic shielding of a charged particle, and the Courant condition which gives the upper limit of the time step interval for the calculation to avoid the numerical instability. To handle multi-scale phenomena with the conventional PIC simulations with uniform grid system, we need to use the minimum spatial grid size and time step interval which correspond to the micro-scale phenomena existing in the simulation system. For example, when a simulation domain includes a localized high density region, the local Debye length at

* Corresponding author. Tel.: +81-78-803-6140
E-mail address: h-usui@port.kobe-u.ac.jp.

Selection and/or peer-review under responsibility of Prof. Mitsuhisa Sato and Prof. Satoshi Matsuoka
doi:10.1016/j.procs.2011.04.254

the highest density becomes very small and the corresponding grid size also should be small enough. In such a situation, we need a huge number of grid points to model the region to guarantee the numerical stability. In the conventional codes which hire uniform grid system, the smallest spatial grid size as used in the high density region has to be assigned to the other regions which have relatively low density. This treatment is not efficient at all in terms of the usage of limited computer resources such as the amount of memory and calculation time.

To achieve an efficient simulation with reasonable cost of computer resources, the spatial and temporal resolutions can be adjusted locally and dynamically depending on the local scales of phenomena. To realize this efficiency in PIC simulations, we decided to incorporate the Adaptive Mesh Refinement (AMR) technique which has been used in the field of computational fluid dynamics as a useful method to investigate such multi-scale phenomena. In the AMR simulation, grids with different spacing are dynamically created in hierarchical layers according to the local conditions of phenomena. Fine grids suitable to the local high density region are applied only there and other regions are simulated by using moderate size grids. Therefore, increment of the numerical cost due to the localized region is not serious by adopting the AMR technique. The AMR technique has been rarely employed in PIC codes except for a few examples [e.g. 2].

In the parallelization, we usually adopt the domain decomposition method. Each sub-domain is assigned to each processor. In PIC simulations, particles located in each sub-domain move and eventually the number of particles in each sub-domain becomes non-uniform. In addition, in the AMR scheme, hierarchical grid systems are dynamically created or deleted in each sub-domain. Then the load balancing between processors cannot be guaranteed through the simulation run. To achieve the load balancing, we need to monitor the load in each processor and adjust the domain decomposition. For this purpose, we will use the Morton ordered curve to number all the cells in the system. In the PIC method, particle calculation occupies most of the total calculation time. In AMR, time intervals are also refined. Therefore, it is the most essential to consider the number of particle calculation loops for each cell among all hierarchical levels as a work weight for each processor. We calculate the work weights based on the cost of particle calculation and hierarchical levels of each cell. Then we decompose the domain according to the Morton curve and the work weight, so that each processor has approximately the same amount of work. We will show you a preliminary result on the dynamic load balancing by using a simple one-dimensional model in which a dense plasma cloud is initially loaded at the central region of the simulation space.

In the present paper, the fundamental equations to be solved in each hierarchy domain and the data structure are given in Sec.2. The concept of the domain decomposition using the Morton curve as well as a preliminary result on the dynamic load balancing in one-dimensional model are described in Sec. 3. The conclusion is given in Sec.4.

2. Simulation scheme and data structure

In the development of a new code we hire a standard three-dimensional electromagnetic explicit PIC simulation

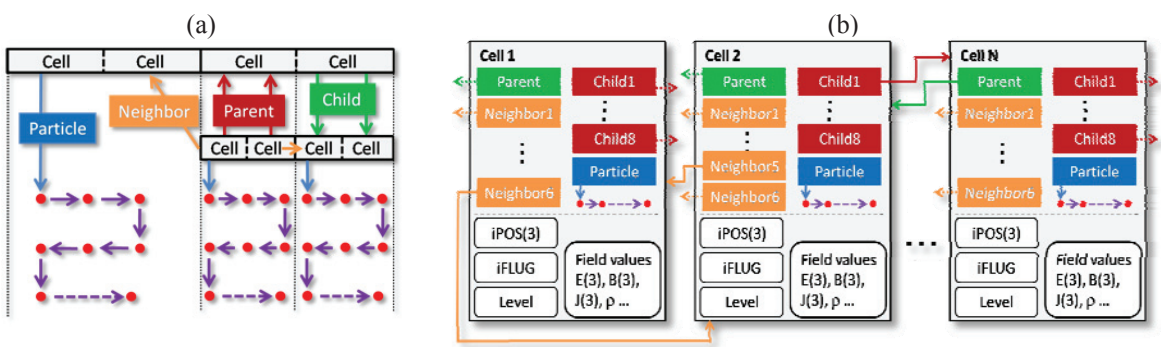

Fig.1. A schematic illustration of the FTT data structure adopted in the AMR-PIC code. Arrows indicate the pointers. Four types of pointers, particle, neighbour, parent, and child, are used to specify the relationship between two cells. For simplicity, a case of two hierarchical levels is depicted in panel (a). Particles belong to a certain cell with a chain of pointers. Panel (b) shows the detail of the Cell structure. Physical quantities such as fields and some flags are also contained in each Cell.

method as described in [1] and [3]. At each time step we update fields values defined at grid points by solving Maxwell's equations with the standard FDTD (Finite-Difference Time-Domain) method [4]. The obtained fields are then applied to the equation of motion for updating the particles' velocities. In the PIC method [1], the field values at each particle position are interpolated from those assigned at the grid points adjacent to the particle. The Buneman-Boris method [1] is used in updating the particle's velocity. The particle position is then updated with the new velocity for one time step. With the updated particles we calculate current density at each cell by gathering the velocity values of particles located in the corresponding cell. The current densities are then used in updating the fields in Maxwell's equations for the next time step. We repeat the above steps until the plasma phenomena of interest reach the steady state. By introducing the PIC method, variation of fields defined at grid points and dynamics of plasma particles located at arbitrary positions are updated in the self-consistent manner.

The AMR simulation is structured hierarchically. Grid size and time step intervals are defined according to the hierarchy levels, where high and low levels correspond to the fine and coarse grid systems, respectively. As the simulation system evolves, some complex microscale phenomena can locally and intermittently occur in a hierarchical domain (Level L). If the grid size in Level L is too coarse to simulate the local complex phenomena, A higher hierarchical domain (Level L+1) is adaptively created in which the grid spacing size and the time step interval become half of those used in the domain of (Level L). In the Level L+1 domain, to synchronize the temporal advancement to Level L, the number of particle loops for updating the velocities and positions becomes double compared to that in the Level L domain because the time step interval is half in the Level L+1. This increase of computational cost caused by the introduction of AMR to particle simulation should be considered in keeping the load balancing in the process parallelization.

The dynamical grid system is organized by using fully threaded tree(FTT) structure [5] as shown in Figure 1. Instead of three dimensional data matrix, `Cells` defined by structure in Fortran 90 contain the data of the physical quantities such as fields and particles defined at the corresponding grids. A cell also contains the level of hierarchical system where the cell itself is located, some flags and pointers which represent the relationship to the other grids, such as `neighbor` for the surrounding grids, `parent` for the original coarse grids in the lower hierarchy and `child` for the refined grids in the higher hierarchy. Particles located in the corresponding cell are also connected with a pointer called `particle`. By using the FTT structure, it becomes easy to handle the creation or removal of each hierarchical domain during a simulation run.

In Figure 2, we show one result obtained with a test simulation using our developing AMR-PIC code. In a three

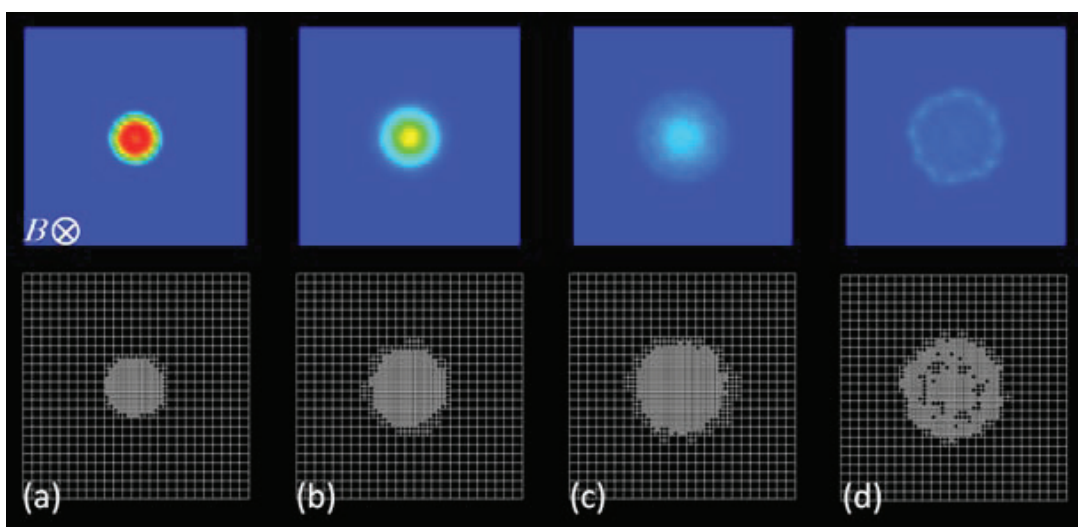

Fig. 2. A test simulation on a laser-produced plasma expansion across the external magnetic field. The upper panel shows temporal variations of electron density in a plane including the center of the plasma cloud. The lower panels display the variation o f spatial grids using AMR. Fine grids are adaptively created at the center region where the density is relatively high.

dimensional space, we put a plasma cloud with a certain temperature in the center as the initial condition and observed the plasma expansion in the radial direction. At the plasma cloud, the density is high and the corresponding Debye length is small. In the current AMR-PIC simulation, we set a criterion that a higher layer is locally created with the half grid size and time step interval when the local Debye length becomes smaller than the current grid spacing. As shown in panel (a) of Figure 2, fine grids are created at the location of the plasma cloud where the local density is high. As the plasma expands in time, the region of fine grids layer also expands as shown in panel (b) and (c). When the density starts to decrease at the center at the final stage, the fine grids correspondingly disappear as shown in panel (d). Although not shown here, the temporal evolution of plasma energy is almost the same as that obtained with the conventional simulation using uniform fine grids which needs much more computational resources.

3. Parallelization using dynamic domain decomposition with particle weight

To speed up the current AMR-PIC simulation, we need to parallelize the code. To simulate a large-scale domain which needs a large amount of memory space, we basically adopt the domain decomposition method and assign each divided sub-domain to each processor for the parallel calculation. The difficulty is the load balancing between processors because the number of spatial grids and particles belonging to each sub-domain is not always constant due to the AMR procedure. In each sub-domain, the hierarchical grid layers are adaptively created or deleted. In addition to the variation of the number of spatial grids, we need to care about the load balancing of particles between processors. This consideration is very important in parallelizing a PIC code because the cost of the particle calculation becomes dominant in PIC simulation.

To determine the manner of decomposing the domain into sub-domains so that load balancing is achieved between processors, we first need to number all the cells in the simulation domain in such a manner that neighboring cells are closely ordered in a series with a space-filling curve. To realize this type of numbering the cells, we hire the Morton ordered curve [6]. In this method, from a three-dimensional grid (*i, j, k*), we extract each binary index with the order as (*k, j, i, k, j, i*, ...), and a new binary number is generated. According to the newly generated number, cells are numbered or ordered along one dimension. Then we decompose the cells by dividing the order by the number of processors.

In dividing the Morton ordered curve, we additionally need to consider the cost of particle calculation in each cell. In hierarchical system, calculation cost of particles increases twice of the parent level as approaching deep hierarchical level because the time step interval becomes half of the parent level, $\Delta t \rightarrow \Delta t/2$, according to the grid spacing becomes $\Delta x \rightarrow \Delta x/2$. To synchronize to the parent level, the calculation loops for particles located in the child level has to be doubled. As stated earlier, the cost of particle calculation is dominant in particle codes, which is about 70-80% of the total cost. Therefore the number of particle calculation loops should be balanced between each processor. The number of particle loops distributed to each processor should be

$$\frac{\Sigma_{cell} 2^L N_{particle}}{N_{process}} \qquad (1)$$

where $N_{particle}$ is the number of particles located in a cell belonging to the Level L domain, and $N_{process}$ is the number of processes used in the parallel simulation. The base level corresponds to L=0 and L increases in the higher hierarchical level with fine grids. Note that the time step interval Δt at the base level (L=0) is divided by 2^L on the level L. Then the numerator of the above quantity implies the total loop number of particle calculation in the whole simulation system required for updating Δt.

In order to achieve the load balancing, the whole domain is decomposed with the Morton ordered curve so that the above quantity (1) becomes the same in each sub-domain. Figure 3 displays a demonstration of the domain decomposition by applying the above idea to a simulation data of an arbitrary time step obtained in a two-dimensional AMR-PIC simulation of plasma cluster model. As shown in panel (c) of Figure 3, the manner of dividing the whole domain into sub-domains is modified from that of the conventional domain decomposition shown in panel (b).

We will show you a preliminary result on the dynamic load balancing by performing a simple one-dimensional simulation in which a local dense plasma cloud is initially loaded in the central of the simulation space. The number of cells is 5120 and we initially put 400 particles in each cell from the cell number 2,433 to 2,688 which are located at the center region. In the other cells, we put 20 particles per cell. Note that we used 32 processors in the parallelization and there is no hierarchical grid system introduced in the simulation for simplicity. In such a situation, 2^L shown in the numerator of the quantity (1) becomes the unity because of L=0 and the whole domain should be divided into sub-domains so that the number of particles in each sub-domain becomes uniform. Figure 4 shows the calculation time of each processor for the cases of the dynamic domain decomposition and the conventional one. It is obviously shown that the calculation time for the dynamic domain decomposition becomes almost uniform among

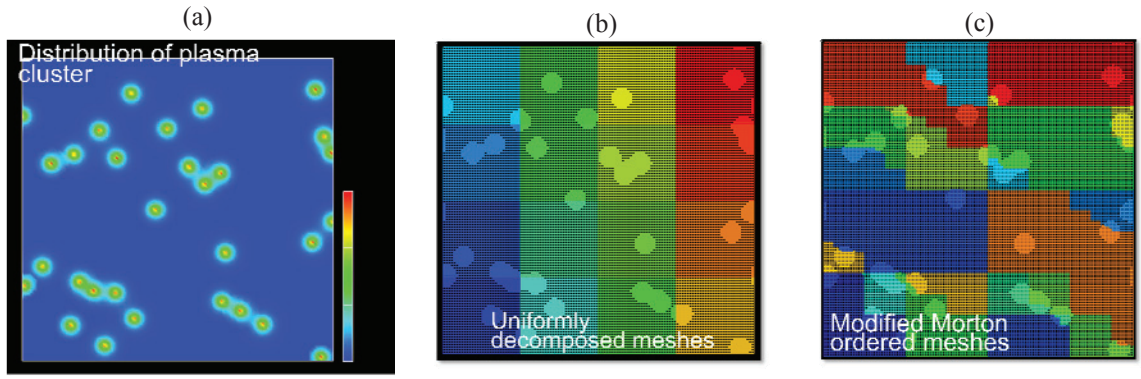

Fig. 3. A demonstration of domain decomposition method by using two-dimensional data obtained in a plasma cluster simulation. In panel (a), many plasma clouds are randomly distributed. In panel (b), fine grids are adaptively created at each cloud. A case of conventional decomposition with uniform and fixed sub-domains is shown in panel (b). Each sub-domain is shown in different color. To achieve the load balancing we determine the sub-domains by using the Morton ordering so that the number of cells and the particle loads becomes uniform between processors as shown in panel (c).

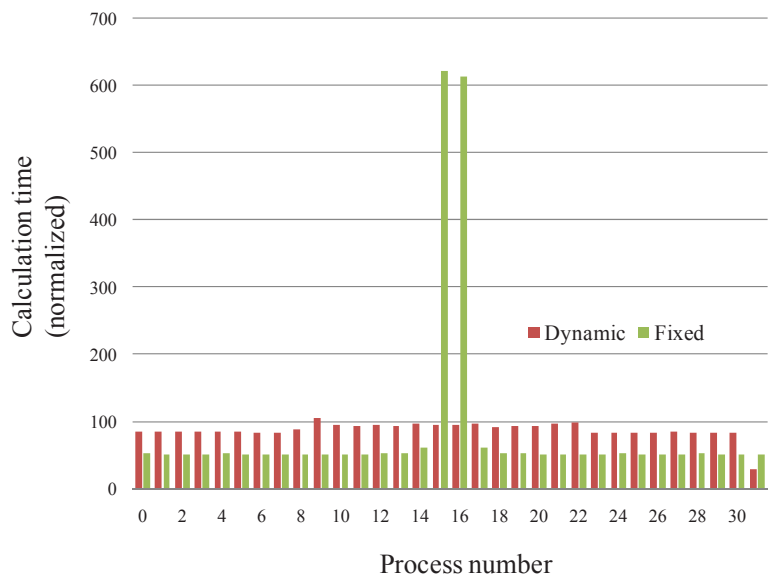

Fig. 4. One example of calculation time in each process for cases using the dynamic domain decomposition scheme and the conventional fixed one. The results in green are for the conventional domain decomposition with fixed uniform sub-domains. The results in red shows a case of the dynamic domain decomposition in which the particle loops evaluated with the quantity shown in (1) is considered when the system is decomposed into sub-domains.

processors while inbalancing of calculation time is seen for the conventional fixed decomposition case. This difference implies that synchronization of calculation time among processors has been achieved for the dynamic domain decomposition case in which uniform number of particles is assigned in each processor. Note that the calculation time is relatively small for the number 31 processor for the dynamic decomposition case because the leftover particles from those already assigned to the other processors are gathered to the last processor. Due to the achievement of the dynamic load balancing, the total calculation is shortened and the calculation becomes approximately six times faster than the conventional method for the current simple model.

We are currently incorporating this scheme in our AMR-PIC code and are trying to perform a test simulation to examine the parallelization efficiency as well as the scalability with respect to the number of processors.

4. Conclusion

We have been developing a new electromagnetic Particle-In-Cell (PIC) code by incorporating the Adaptive Mesh Refinement (AMR) technique in order to simulate multi-scale phenomena of space plasma including electron and ion kinetics. In the AMR scheme, each hierarchy level has its own grid size and time step interval. Fine grids in the hierarchical domains are dynamically removed or produced depending on local physical conditions. A test simulation of a laser-produced plasma expansion shows good agreement with the results obtained with the conventional PIC code, which can validate our developing AMR-PIC code.

To speed up the simulation, we started the parallelization of our developing AMR-PIC code by adopting the domain decomposition scheme using MPI. To achieve the load balancing among sub-domains distributed to processors, we number all the cells with the Morton ordered curve and divide the order into the number of the processors in such a manner that each sub-domain has approximately the same amount of the particle calculation loops. Since the particle calculation occupies most of the total simulation time, this modification in the usage of the Morton curve is very important and can cause the efficient parallelization. By performing a simple one-dimensional

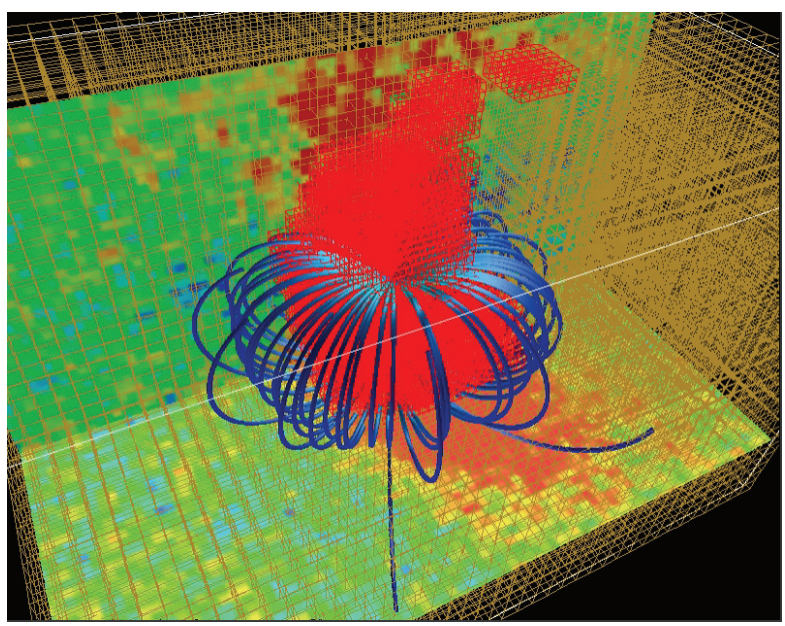

Fig. 5. A bird's eye view of a three-dimensional ARM-PIC simulation on interaction between a dipole magnetic fields and background plasma flowing from the left to the right direction. The blue curves indicate some dipole magnetic field lines. The contour maps show density variations of plasma on different planes. Red color on the contour maps implies high density. Two hierarchical systems which have fine and coarse grids are created with the AMR scheme as shown in red and yellow, respectively. The fine grids shown in red are particularly created at the central region where the magnetic field density is relatively high. At the high density region of plasma shown in red on the contour maps fine grids are also created adaptively.

simulation, we could confirm that the dynamic load balancing has been successfully achieved and the calculation time is much reduced in comparison with a case with the conventional fixed decomposition scheme. We are currently incorporating this dynamic domain decomposition scheme using the Morton curve to our developing AMR-PIC simulation code and trying to examine the parallelization efficiency of the code.

Finally another preliminary result is shown in Figure 5 which is a bird's eye view on interaction between a dipole magnetic field and a background plasma flow. In the simulation we could confirm that two hierarchical systems with fine and coarse grids are adaptively and locally created or deleted with the AMR scheme depending on the local variation of plasma condition. In Figure 5, red and yellow grids correspond to the fine and coarse ones, respectively. It is shown that fine grids are particularly created at the region where the magnetic field density as well as the plasma density is intense around the center of the magnetic dipole fields.

When we complete the parallelization of the AMR-PIC code with the dynamic domain decomposition scheme described above, we will intensively perform simulations in order to analyze multi-scale phenomena in association with the interactions between the dipole magnetic fields and a fast plasma flow such as the solar wind in space. We are particularly interested in the plasma phenomena occurring at the boundary region such as magnetic field reconnection and shock structure as well as plasma turbulence. We will analyze such phenomena from a view point of multi-scale coupling between electrons and ions. By using the AMR-PIC code, we can simulate such wave-particle interactions with computer resources much less than ever used, which is one of the advantages of adopting the AMR scheme in a PIC code.

Acknowledgements

This work is funded by JST (Japan Science and Technology Agency)/CREST (Core Research for Evolutional Science and Technology). The computation in the present study was performed with the KDK system of Research Institute for Sustainable Human sphere (RISH) at Kyoto University.

References

1. C.K. Birsall, and A.B. Langdon, Plasma Physics via Computer Simulation, Adam Hilger, New York, (1991)

2. K. Fujimoto, Physics of Plasmas 13, 072904 (2006).

3. Y. Omura and H. Matsumoto, Computer Space Plasma Physics: Simulation Techniques and Softwares, H. Matsumoto and Y. Omura, Ed. Tokyo: Terra Scientific, 21(1993).

4. K. S. Yee, IEEE Trans. Antennas Propagat., 14, 302(1966).

5. A.M. Khokhlov, Journal of Computational Physics 143, 519 (1998).

6. M. S. Warren, and J. K. Salmon, Proceedings of Supercomputing , IEEE, 12,21,(1993).