# Fast Electromagnetic Simulation by Parallel MoM Implemented on CUDA

Ming Fang[a], Kai-Hong Song[a], Zhi-Xiang Huang[a],Xian-Liang Wu[a,b]
[a]Key Laboratory of Intelligent Computing and Signal Processing, Anhui University,Hefei 230039,China
[b]School of Electronic and Information Engineering, Hefei Normal University, Hefei 230061,China
Email: sk_hong@sina.com, zxhuang@ahu.edu.cn

***Abstract-* The keys to electromagnetic simulation of arbitrary configuration of conducting surfaces by method of moments (MoM) are filling of impedance matrix elements and solving of linear equations. A CUDA (Compute Unified Device Architecture) enabled graphics processing unit (GPU) launched by NVIDIA company to accelerate implementation of the fast filling of impedance matrix in MoM based on Rao-Wilton-Glisson (RWG) basis functions was presented. Parallel LU decomposition method is applied for the solving of linear equations, and a parallel method looping over squares is proposed in CUDA parallel platform, furthermore. The GPU numerical results for a user-created benchmark structures are checked with comparison to CPU results. A noticeable speedup of the filling of impedance matrix (hundreds times) and solving of linear equations (about 10 times) is achieved due to the employing of GPU.**

***Keywords-* Method of Moments; GPU; LU Decomposition; CUDA**

## I. INTRODUCTION

The MoM method was introduced to electromagnetic simulation by R.F.Harrington[1]. As a numerical method for strictly computing electromagnetic problems, it is widely applied and its precision of computed results is high. However, as the dimension of object to be computed increases, the dimensionality of the impedance matrix to be filled is higher, and gives rise to computation and memory capacity, which is the bottleneck of computation by MoM[2]. It has been difficult for the traditional computing element CPU competent for massive data processing.

Fortunately, the MoM method is by its nature a massively parallelizable algorithm and thus can benefit greatly from most advances in multi-core and parallel computing. In this paper, the acceleration of MoM method is implemented with parallel programming of GPUs using Nvidia's CUDA programming platform. We extend MoM's electromagnetic scattering problem from two aspects. Firstly, the filling computation process of each element in impedance matrix is the same, meeting the feature of GPU parallel computation. Each thread of the multi threads of GPU can fill an element, and then tens of thousands of elements are computed parallelly, so it has obvious computation advantages compared with traditional computing element in CPU. Secondly, as the computation method of linear equations, LU decomposition, conjugate gradient (CG) method and Gaussian elimination can be used. These methods may change matrix element during iteration process, which make these methods unsuited for parallel processing. We will use parallel LU decomposition method. It is proposed for parallel solution of linear equations based on CUDA. In this method, traditional LU decomposition method is partitioned in blocks. Following from the two points, filling time of the impedance matrix and optimizing computation of linear equations become the key concerns and are the subjects of this paper.

## II. THEORIES

### *A. Acceleration Techniques in CUDA*

In general, applying more effective algorithms and using powerful computers are carried out for speeding up an application. It's a naturally rising possibility to increase the operational speed of the traditional application processor-CPU. However, the computing capacity is under the impact of the manufacturing restrictions. The other possibility is to increase the number of CPUs or the number of cores. Due to the complexity of CPU cores junction and the architectural design problems, it's an expensive attempt.

Taking into account the above factors, GPU launched by NVIDIA supports CUDA techniques which initially designed for graphic processing, and the new hardware design, which is well suited for general parallel computation, and consisting of hundreds of processor cores is capable to execute thousands of threads parallelly at the same time. As we can see in Fig. 1, GPU has more ALUs(Arithmetic Logical Units) than CPU[3].

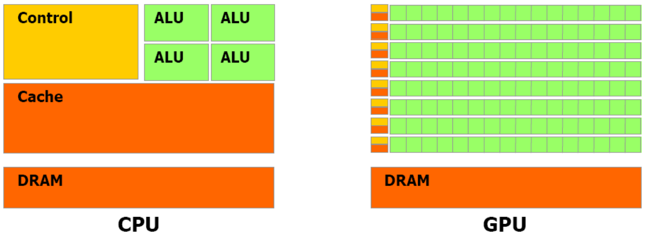


Figure 1. GPU has more transistors for arithmetic logical processing

### *B. MoM with RWG Basis Functions*

Solving an electromagnetic scattering problem involving perfect electric conduction (PEC), the well-known electric field integral equation (EFIE) is to be solved, which can be expressed as

$$\mathbf{E}^{inc}(\mathbf{r})\,|_{\tan}= jk_0\eta\int_{\mathbf{s}}\left[\mathbf{J}(\mathbf{r'})+\frac{1}{k^2}\nabla\left(\nabla'\cdot\mathbf{J}(\mathbf{r'})\right)\right]G(\mathbf{r},\mathbf{r}')ds'\Bigg|_{\tan} \qquad (1)$$

and $k_0$ is wave number in the air medium, $\eta$ is intrinsic impedance of the free space, $G(\mathbf{r},\mathbf{r'})$ is the free space Green's function.

With respect to computation of scattering characteristics of three-dimensional PEC body, RWG basis functions adopt triangle pairs to conduct modeling mesh generation on the conductions surface, and the application of triangle pairs can well simulate three-dimensional conductions with arbitrary shapes [4]. RWG basis functions adopt two adjacent triangles as surface element to define the current, the current vector to common edge by the triangle flowing through is as shown in Fig. 2.

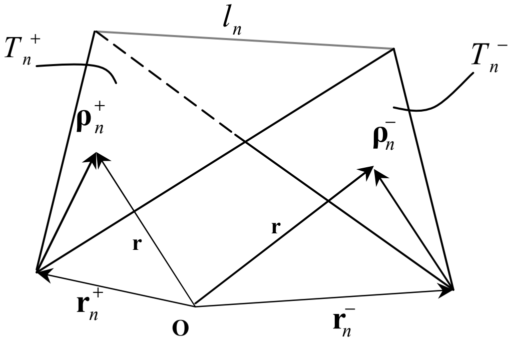


Figure 2. RWG basis function

Assuming $A_n^+$ and $A_n^-$ are areas of triangles $T_n^+$ and $T_n^-$, $l_n^+$ is the common edge, the vector draw from the vertex triangle $T_n^+$ to the observation point is $\boldsymbol{\rho}_n^+$, and $\boldsymbol{\rho}_n^-$ is the vector draw from observation point to the vertex of triangle $T_n^-$. So the expression of basis function can be defined as

$$\mathbf{f}_n(\mathbf{r})=\begin{cases}\dfrac{l_n}{2A_n^+}\boldsymbol{\rho}_n^+ & \mathbf{r}\in T_n^+\\[2ex] -\dfrac{l_n}{2A_n^-}\boldsymbol{\rho}_n^- & \mathbf{r}\in T_n^-\end{cases} \qquad (2)$$

Expand the scattered surface by RWG basis functions. The following linear equation system can be obtained by testing the EFIE by the RWG basis functions

$$\sum_{n=1}^{N} Z_{mn} I_n = V_m \quad m = 1,2,\cdots,N \qquad (3)$$

where the computational formulas of EFIE-MoM impedance matrix can be gotten

$$Z_{mn} = jk_0\eta\Bigg(\int_{\mathbf{S}}\int_{\mathbf{S'-r}} f_m(\mathbf{r})\cdot f_n(\mathbf{r'})G(\mathbf{r},\mathbf{r'})d\mathbf{S'}d\mathbf{S}$$
$$-\int_{\mathbf{S}}\int_{\mathbf{S'-r}}\frac{1}{k^2}\nabla_S\cdot f_m(\mathbf{r'})\nabla'_S\cdot f_n(\mathbf{r'})G(\mathbf{r},\mathbf{r'})d\mathbf{S'}d\mathbf{S}\Bigg) \qquad (4)$$

### *C. Doolittle LU Decomposition*

The acquisition of impedance matrix and excitation-vectors is followed by a key step which is the consideration of solving linear equations. For *LU* decomposition method, as a solution of linear equation $Ax=b$, if *A* is invertible matrix, then matrix *A* can be resolved to a product of lower triangular matrix *L* and upper triangular matrix *U*[5].

$$\begin{bmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{bmatrix}=\begin{bmatrix}1 & & & \\ l_{21} & 1 & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & 1\end{bmatrix}\begin{bmatrix}u_{11} & u_{12} & \cdots & u_{1n}\\ & u_{22} & \cdots & u_{2n}\\ & & \ddots & \vdots\\ & & & u_{nn}\end{bmatrix} \qquad (5)$$

Elements of matrix *L* and matrix *U* are easily solved

$$u_{1j}=a_{1j}, j=1,2,\cdots,n \qquad (6)$$

$$l_{i1}=a_{i1}/a_{11}, i=1,2,\cdots,n \qquad (7)$$

$$u_{ij}=a_{ij}-\sum_{k=1}^{i-1}l_{ik}u_{kj}, j=i,i+1,\cdots,n \qquad (8)$$

$$l_{ij}=\left(a_{ij}-\sum_{k=1}^{i-1}l_{ik}u_{kj}\right)\Big/u_{ii}, i=j,j+1,\cdots,n \qquad (9)$$

Substituting *L, U* for *A*, let $UX=Y$, and then we two back substitution problems *Ax=b* are the things to be resolved.

$$\left.\begin{aligned}Ly=b\\ Uy=y\end{aligned}\right\} \qquad (10)$$

Solving $Ly=b$ can get $y$,

$$\left.\begin{aligned}&y_1=b_1\\ &y_1=b_i-\sum_{j=1}^{i-1}l_{ij}y_j \quad i=2,\cdots,n\end{aligned}\right\} \qquad (11)$$

Taking the values for *y* and solving the equation *Ux=y*. And the solution to the system *Ax=b* can be acquired.

$$\left.\begin{aligned}&x_n=y_n/u_{nn}\\ &x_i=\left[y_i-\sum_{j=i+1}^{n}u_{ij}x_j\right]\Big/u_{ii}, \quad i=n-1,\cdots,1\end{aligned}\right\} \qquad (12)$$

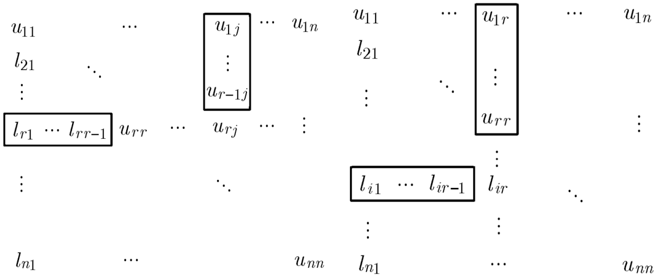
## III. Parallel Computation Implementation CUDA

3D Studio Max software is used to conduct modeling and mesh generation on three-dimensional PECs. In the CUDA program, the vertexes and common edges data got by 3D Studio Max are required to conduct numbering and matching treatment. GPU reads post-processing information of the mesh from the internal memory, the needed threads are allocated and computed on GPU according to the data volume, and the mesh data read from host-CPU need to be placed on shared memory in device-GPU. Each thread realizes the filling of an impedance matrix element [6].

After filling impedance matrix and excitation vertex, when adopting LU decomposition subprogram to compute induced current coefficient, it requires to share the data above and the left side of diagonal elements to other threads, then we can read the data to complete a circulation according to right-angle circulation. These computations are conducted successively in line with number sequence, and multi-threads may not read shared data at the same time. Therefore, the shared data can be stored in shared memory. To prevent multi-threads reading shared memory at the same time, the *"__syncthreads()"*

CUDA order can be used to avoid race conditions when loading shared memory[7]. This is quite dangerous in CUDA platform. According to the induced current coefficient, the scattering features of scattered field, induced current at conductor surface and radar cross section (*RCS*) etc. can be achieved.

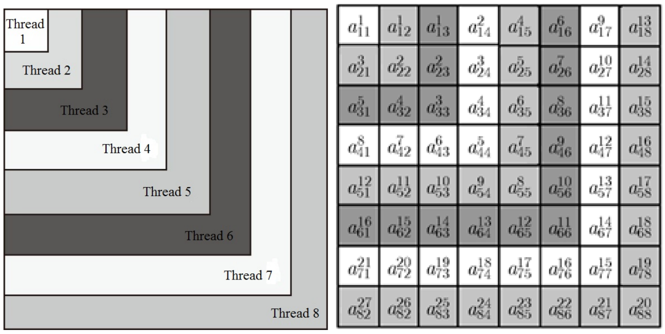
It can be seen from Eq. (8) and Eq. (9) that computation of $u_{ij}$ needs $a_{ij}$, $l_{ik}$ and $u_{kj}$, while computation of $l_{ij}$ needs $a_{ij}$, $l_{ik}$ and, i.e. computation of $u_{ij}$ needs all data above the $j^{th}$ line, and computation of $l_{ij}$ needs all data at the left of the $i^{th}$ row, as shown in the Fig. 3. Therefore, if the traditional *LU* decomposition method is used, the computations of $u_{ij}$ and $l_{ij}$ can be done only after data relied on by computation are computed, so it is unable to realize parallel computation. Doolittle *LU* decomposition adopts right-angle circulation pattern to distribute the computation process, thus making *LU* decomposition process able to parallelly compute.



(a) (b)

Figure 3. Traditional LU decomposition: (a)Elements required for calculation of $u_{rj}$ ,(b) Elements required for calculation of $l_{ir}$

What shown in Fig. 4 are three processes of distribution according to right-angle circulation. Allocate the elements at the left side of and above the diagonal elements into a process, through loop computation, till the whole matrix is worked out. The method features that when computing $u_{mm}$, ..., $u_{mn}l_{m+1}$, ..., $l_{nm}$, other processes can be computed through sharing all data at the left side of and above umm to them after finishing computing $u_{mm}$. The whole computation process can be completed through data sharing for n times. Data volume needed to be shared at $m^{th}$ step is 2 × (*m* - 1) data, while the data needed to be transmitted at $m^{th}$ step of traditional *LU* decomposition method is (*m* - 1) ×2, while greatly reduces the data volume of data transmission.



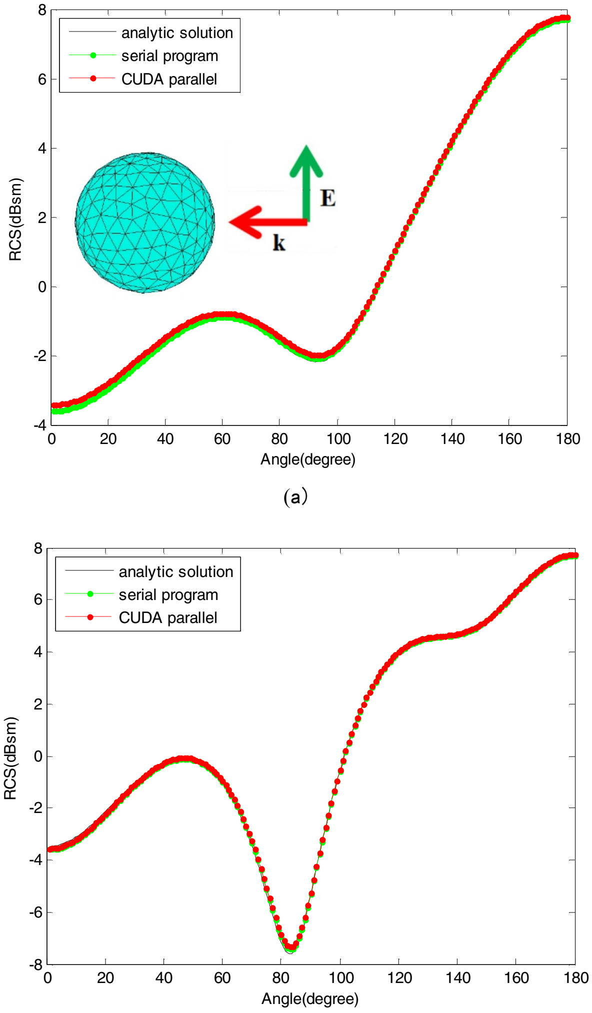
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $a^{1}_{11}$ | $a^{1}_{12}$ | $a^{1}_{13}$ | $a^{2}_{14}$ | $a^{4}_{15}$ | $a^{6}_{16}$ | $a^{9}_{17}$ | $a^{13}_{18}$ |
| $a^{3}_{21}$ | $a^{2}_{22}$ | $a^{2}_{23}$ | $a^{3}_{24}$ | $a^{5}_{25}$ | $a^{7}_{26}$ | $a^{10}_{27}$ | $a^{14}_{28}$ |
| $a^{5}_{31}$ | $a^{4}_{32}$ | $a^{3}_{33}$ | $a^{4}_{34}$ | $a^{6}_{35}$ | $a^{8}_{36}$ | $a^{11}_{37}$ | $a^{15}_{38}$ |
| $a^{8}_{41}$ | $a^{7}_{42}$ | $a^{6}_{43}$ | $a^{5}_{44}$ | $a^{7}_{45}$ | $a^{9}_{46}$ | $a^{12}_{47}$ | $a^{16}_{48}$ |
| $a^{12}_{51}$ | $a^{11}_{52}$ | $a^{10}_{53}$ | $a^{9}_{54}$ | $a^{8}_{55}$ | $a^{10}_{56}$ | $a^{13}_{57}$ | $a^{17}_{58}$ |
| $a^{16}_{61}$ | $a^{15}_{62}$ | $a^{14}_{63}$ | $a^{13}_{64}$ | $a^{12}_{65}$ | $a^{11}_{66}$ | $a^{14}_{67}$ | $a^{18}_{68}$ |
| $a^{21}_{71}$ | $a^{20}_{72}$ | $a^{19}_{73}$ | $a^{18}_{74}$ | $a^{17}_{75}$ | $a^{16}_{76}$ | $a^{15}_{77}$ | $a^{19}_{78}$ |
| $a^{27}_{82}$ | $a^{26}_{82}$ | $a^{25}_{83}$ | $a^{24}_{84}$ | $a^{23}_{85}$ | $a^{22}_{86}$ | $a^{21}_{87}$ | $a^{20}_{88}$ |

Figure 4. Distribution and numbering according to right-angle circulation of simple eight process in parallel LU decomposition

## IV. Results

With the same hardware platform and solution method, the solutions to scattering features of three-dimensional conductor are realized respectively on CPU and GPU. The configuration of the computer used in the text is: Intel Core i5-2410M dual core CPU main frequency 2.3GHz, host memory 4Gbyte, GPU is NVIDIA GeForce GT 550M, there are totally 1,480,000 thread blocks, and each thread block has 1024 threads, and the video memory is 2Gbyte.

Taking normal plane wave incidence with frequency $3.0\times10^{8}$Hz to PEC sphere of the radius as 0.45 times of wavelength, and compute the bistatic RCS of the PEC sphere. Using C++ serial program and GPU parallel programs for computation, the result through comparison validation between GPU computed result, C++ serial program and analytic solution [8] is shown in Fig. 5.



(a)

(b)

Figure 5. Comparisons among the bistatic RCS of PEC sphere illuminated by a normal incident vertically polarized plane wave obtained by analytic solution, serical program and parallel program, respectively. (a) Bistatic RCS of E-plane, (b) bistatic RCS of H-plane.

From Fig.5, it can be perceived that, the results solved by parallel program based on CUDA and serial MoM program almost match the analytic solution perfectly. So that the accuracy of CUDA parallel program is verified. To compare the serial program based on CPU and GPU parallel program, the speedup ratio is introduced, i.e. the ratio of times needed for computing same issues by means of GPU parallel program and CPU serial program.

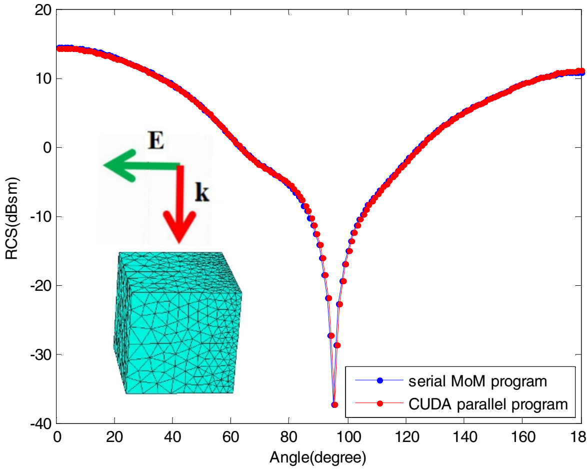


Figure 6. H-plane RCS for PEC cube illuminated by a plane wave (depicted in the figure).Excellent agreements between analytic solution and CUDA parallel program are observed.

The speedup ratios are shown in the Table. I and Table. II for the impedance matrix with different dimensions for the cube structure shown in the insert of Fig. 6. It can be seen that as impedance matrix dimensionality increases, the speed-up ratio may also increase, and the speedup effect is more obvious [9].

Table I:
The speedup ratios of parallel matrix filling

| Matrix dimensions | CUDA parallel program (s) | CPU serial program (s) | Speed-up ratio |
|---|---|---|---|
| 1440 | 0.0373 | 23.98 | 328.49 |
| 2280 | 0.1192 | 63.55 | 533.14 |
| 5220 | 0.5852 | 358.23 | 612.15 |
| 9360 | 1.7430 | 1257.48 | 748.71 |

Table II:
The speedup ratios of parallel matrix filling

| Matrix dimensions | Parallel LU decomposition(s) | Traditional LU decomposition(s) | Speed-up ratio |
|---|---|---|---|
| 1440 | 0.8912 | 8.98 | 10.08 |
| 2280 | 1.2351 | 13.28 | 10.75 |
| 5220 | 3.0729 | 38.34 | 12.48 |
| 9360 | 8.1984 | 127.27 | 15.52 |

## V. Conclusion

A computational method was presented for facilitating the fast solution of scattering problems due to PEC surfaces. The proposed method was aiming at solving of the two difficulties i.e. the filling of impedance matrix and solving of linear equations in MoM method. The RWG basis functions were used to expand currents on the conductor surfaces, and GPU was used as parallel programming computing platform to realize parallel filling of impedance matrix elements and parallel computation of LU decomposition, respectively. The proposed method for parallel filling of impedance matrix is found hundreds times faster than CPU serial program, and parallel LU decomposition method can decrease communication quantity on CUDA platform and accelerate computing speed efficiently.

## Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant (Nos. 60931002, 61101064, 51277001, 61201122), DFMEC (No. 20123401110009) and NCET (NCET-12-0596) of China, Distinguished Natural Science Foundation (No. 1108085J01), and Universities Natural Science Foundation of Anhui Province (No. KJ2011A002), Graduate Academic Innovate Research Foundation (No.60931002, 61101064), and the 211 Project of Anhui University.

## References

[1] R. F. Harrington, “Field Computation by Moment Methods”. *Macmillan New York*, pp. 30-36, 1968.

[2] Walton C. Gibson, “The Method of Moments in Electromagnetics”, Chapman & Hall/CRC, pp.270-281, March 2007.

[3] [Online], Link: http://www.nvidia.cn/object/cuda_home_new_cn.html

[4] D. R. Wilton, S. M. Rao, A. W. Glisson, “Electromagnetic scattering by surface of arbitrary shape”, *IEEE Transactions on Antennas and propagation*, vol. 30, pp. 409-418, 1982.

[5] Gilbert Strang, “Introduction to Linear Algebra, 3th edition”, Massachutts Institute of Technology, vol. 2, pp. 21-96, 2003.

[6] D. David and E.Lezar, GPU acceleration of method of moments matrix assembly using RWG basis functions, *International Conference On ICEIE*, Amsterdam, pp. 56-60, 2010.

[7] Z. Badics, I. Kiss, Parallel realization of the element-by-element fem technique by CUDA, *IEEE Transactions on Magnetics*, vol. 48, pp. 507-510, 2012.

[8] S.L. Grand, M. Garland, and J. Hardwick, Parallel Computing Experriences With CUDA, *IEEE Transactions on Antennas and propagation*, vol.28, pp. 13-27, 2008.

[9] C. Leat, N. Shuley and G. Stickley, Triangular-patch model of bowtie antennas, *IEE Proceeding Microwaves Antennas and Prop.*, vol. 145, pp. 465-470, 1998.