An Efficient Multi-GPU Accelerated IEDG-CBFM-SMWA Method for Electromagnetic Scattering Analysis

Ziang Shen[1], Xinlei Chen[1,2], and Zhuo Li[1]

[1]Key Laboratory of Radar Imaging and Microwave Photonics
College of Electronic and Information Engineering,
Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
shenziang@nuaa.edu.cn, chenxl@nuaa.edu.cn, lizhuo@nuaa.edu.cn

[2]Key Laboratory of Inlet and Exhaust System Technology
College of Energy and Power Engineering,
Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Abstract – This paper presents a multiple graphics processing units (GPUs) accelerated direct solver for the method of moments (MoM). First, we integrate the characteristic basis function method-Sherman-Morrison-Woodbury formula-based algorithm (CBFM-SMWA) and the integral equation discontinuous Galerkin (IEDG) method. The IEDG can enforce current continuity across nonconformal meshes by employing hybrid RWG/half-RWG basis functions. Second, we develop a SMWA-centric multi-GPU parallelization strategy that fully exploits the intrinsic parallelism in compression and solution of hierarchical matrices, while CBFM serves to reduce the number of unknowns. The proposed multi-GPU IEDG-CBFM-SMWA achieves substantial computational acceleration over its CPU implementation while overcoming the memory limitations of conventional GPU-accelerated MoM. Numerical examples demonstrate that the proposed method offers excellent scalability across multiple GPUs, significant performance advantages and reliable accuracy for solving electrically large scattering problems.

Index Terms – Direct solver, method of moments (MoM), multiple graphics processing unit (multi-GPU).

I. INTRODUCTION

The method of moments (MoM) [1, 2] has been recognized as a powerful numerical method for solving electromagnetic problems due to its high computational accuracy, and various acceleration techniques have been developed to overcome its inherent high computational complexity, including the adaptive cross approximation (ACA) [3, 4] based method and the fast multipole algorithm (FMA) [5, 6] based method, which have significantly extended MoM's applicability for solving electrically large problems. Concurrently, with the rapid evolution of hardware, graphics processing units (GPUs) have become an important component in high-performance computing (HPC). GPUs offer exceptional parallel processing capabilities at a relatively low cost, making them ideal for accelerating computationally intensive algorithms. Therefore, there exists a promising research direction in developing GPU-accelerated high-performance numerical methods that leverage modern parallel architectures to achieve high efficiency for solving electromagnetic problems.

Filling and solving the impedance matrix represent the most computationally intensive stages in the MoM, typically consuming the majority of execution time. However, the inherent independence of matrix elements provides opportunities for parallel acceleration. The pioneering GPU implementations of MoM were demonstrated in [7–9], and subsequent research in [10, 11] investigated GPU's architectural characteristics to optimize program efficiency on GPUs. Despite these advances, the conventional MoM continues to present significant challenges due to the enormous memory requirements for storing the full impedance matrix, which often exceed the capacity of GPU devices. While out-of-core techniques proposed in [7, 12] addressed memory constraints by dividing the matrix into blocks and processing them sequentially, these approaches introduce data transfer overhead and require complex load balancing strategies. Consequently, the integration of fast methods and GPU acceleration represent a more practical solution approach. References [13–16] introduced GPU implementations of the FMA and multi-level FMA (MLFMA), achieving superior performance over CPU versions. Considering that iterative solvers employed in these methods may encounter convergence difficulties when facing ill-conditioned matrices, direct

Submitted On: January 14, 2026
Accepted On: June 08, 2026

1054-4887 © ACES

https://doi.org/10.13052/2026.ACES.J.410706

solvers not only circumvent convergence issues but also offer advantages for multi-excitation problems, such as monostatic radar cross-section (RCS) analysis. This has motivated GPU-accelerated implementations of direct solution methods, including the characteristic basis function method (CBFM) [17–20]. Nevertheless, for electrically large problems, the reduced matrix generated by the CBFM still consumes considerable resources which may exceed device memory capacity limitations.

This paper presents a high-performance direct solver leveraging the CBFM-Sherman-Morrison-Woodbury formula-based algorithm (CBFM-SMWA) [21], integral equation discontinuous Galerkin (IEDG) method [22, 23] and multi-GPU acceleration. The proposed method not only efficiently compresses and solves the conventional MoM matrix equation, but also can ensure current continuity at nonconformal meshes by employing a hybrid basis function scheme combining of Rao-Wilton-Glisson (RWG) and half-RWG basis functions. Furthermore, the multi-GPU acceleration can enhance the computational efficiency while maintaining high numerical accuracy.

II. THEORETICAL BACKGROUND

Conventional MoM typically requires conformal meshes, whereas the IEDG [22] relaxes this constraint by allowing nonconformal meshes between different structures. A half-RWG basis function is defined on a single triangle, as shown in Fig. 1, and current continuity is enforced through interior penalty terms rather than through mesh continuity. Specifically, RWG basis functions are retained within each conformal mesh region, while half-RWG basis functions are introduced at the boundaries of different structures where mesh are nonconformal. This hybrid RWG/half-RWG strategy preserves the modeling flexibility of independent meshing while maintaining the solution accuracy.

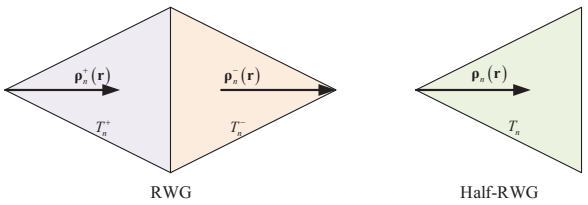


Fig. 1. Schematic diagram of RWG and half-RWG basis functions.

As a fast algorithm that can effectively reduce the number of unknowns, the CBFM [23, 24] partitions the target into several blocks and constructs characteristic basis functions (CBFs) for each block. This process transforms the original MoM matrix equation into a reduced version:

$$\mathbf{Z}^{CBF}\mathbf{I}^{CBF} = \mathbf{V}^{CBF}, \tag{1}$$

where $\mathbf{Z}^{CBF}$ represents the reduced impedance matrix, $\mathbf{V}^{CBF}$ is the reduced excitation coefficients vector, and $\mathbf{I}^{CBF}$ denotes the reduced current coefficient vector. The reduced impedance matrix between the i-th block and the j-th block can be expressed as:

$$\mathbf{Z}_{ij}^{CBF} = \mathbf{J}_i^T \mathbf{Z}_{ij}^{RWG} \mathbf{J}_j, \tag{2}$$

where $\mathbf{J}_i$ and $\mathbf{J}_j$ are the CBFs on the i-th and j-th block, respectively. $\mathbf{Z}_{ij}^{RWG}$ is the mutual impedance matrix of RWG/half-RWG between the i-th block and the j-th block. Since the number of CBFs is substantially smaller than the number of RWG basis functions, both the number of unknowns and the matrix dimension are substantially reduced.

The Sherman-Morrison-Woodbury formula-based algorithm (SMWA) provides an effective hierarchical decomposition strategy for further matrix compression and computational complexity reduction [21, 25]. Firstly, the SMWA employs a multilevel balanced binary tree to decompose the target. The reduced self-impedance matrix of leaf nodes can be directly solved and, for non-leaf nodes, its reduced matrix equation is partitioned as:

$$\begin{bmatrix} \mathbf{Z}_{11} & \mathbf{Z}_{12} \\ \mathbf{Z}_{21} & \mathbf{Z}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{I}_1 \\ \mathbf{I}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix}, \tag{3}$$

where $\mathbf{Z}_{11}$ and $\mathbf{Z}_{22}$ is the self-impedance matrix of the two child nodes, $\mathbf{Z}_{12}$ and $\mathbf{Z}_{21}$ denote the reduced mutual impedance matrices between child nodes, $\mathbf{V}$ and $\mathbf{I}$ are the excitation and current vectors of the nodes. The mutual impedance matrices $\mathbf{Z}_{12}$ and $\mathbf{Z}_{21}$ can be efficiently decomposed by ACA [3, 21] as follows:

$$\begin{cases} \mathbf{Z}_{12} \approx \mathbf{U}_{12}\mathbf{V}_{12} \\ \mathbf{Z}_{21} \approx \mathbf{U}_{21}\mathbf{V}_{21} \end{cases}. \tag{4}$$

Multiplying both sides of (3) by the inverse matrix of

$$\begin{bmatrix} \mathbf{Z}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}_{22} \end{bmatrix}, \tag{5}$$

we have

$$\begin{bmatrix} \mathbf{1} & \mathbf{Z}_{11}^{-1}\mathbf{U}_{12}\mathbf{V}_{12} \\ \mathbf{Z}_{22}^{-1}\mathbf{U}_{21}\mathbf{V}_{21} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{I}_1 \\ \mathbf{I}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{Z}_{11}^{-1}\mathbf{V}_1 \\ \mathbf{Z}_{22}^{-1}\mathbf{V}_2 \end{bmatrix}, \tag{6}$$

where $\mathbf{1}$ is identity matrix and $\mathbf{0}$ is zero matrix. The inverse matrix $\mathbf{Z}_{11}^{-1}$ and $\mathbf{Z}_{22}^{-1}$ can be obtained by direct inversion or recursively via the SMWA algorithm [21].

The solution of (6) can be efficiently obtained by using the SMW formula [21]:

$$(\mathbf{1}+\mathbf{UV})^{-1} = \mathbf{1}-\mathbf{U}(\mathbf{1}+\mathbf{VU})^{-1}\mathbf{V}. \tag{7}$$

Therefore, the computational complexity is remarkably reduced. More detailed solution steps can be found in [21].

As demonstrated throughout this section, a significant advantage of the SMWA is its independence from the underlying basis functions. This flexibility not only facilitates the incorporation of hybrid RWG/half-RWG basis functions for handling nonconformal meshes, but also enables the generation of CBFs at the leaf nodes of the SMWA's binary tree structure to reduce the size of the hierarchical matrix.

III. MULTI-GPU ACCELERATED IEDG-CBFM-SMWA IMPLEMENTATION

The GPU implementation of the IEDG-CBFM-SMWA leverages the parallelism inherent in GPU architectures to accelerate the generation of CBFs, decomposition and solution of hierarchical matrix. The compute unified device architecture (CUDA) developed by NVIDIA provides a mature GPU programming framework with highly optimized linear algebra libraries such as cuSOLVER and cuBLAS, making it the foundational platform for the GPU implementation discussed in this paper.

A. GPU-accelerated MoM functions

Before introducing the proposed multi-GPU accelerated method, we must first implement the GPU-MoM functions upon which its computation depends. As described in section I, the matrix filling process is inherently parallelizable as each element of impedance matrix or voltage vector can be computed independently. Following pre-processing on the host, the mesh data and basis function information are transferred to the device. After allocating sufficient memory space on the device, the host will launch GPU kernel functions that spawn thousands of concurrent threads, where each thread assigned to fill a single matrix element.

Although CUDA has minimized the complexity of GPU programming, effective GPU implementation requires more than a straightforward port of CPU code. It requires careful management of thousands of parallel threads and optimization strategies tailored to the GPU's architectural characteristics. For instance, thread blocks must be configured to align with 32 threads per warp to maximize hardware resource utilization, as shown in Fig. 2, and memory hierarchy optimizations such as texture memory or constant memory are used to accelerate frequent data operations. Consequently, achieving high-performance on the GPU entails considerable effort in kernel design and resource management.

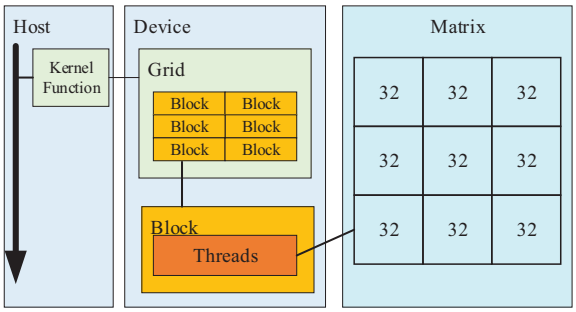


Fig. 2. Task allocation within a CUDA kernel function. A kernel function starts a grid containing multiple blocks, and each block contains multiple threads. Each thread is responsible for filling one element in the matrix. Elements are grouped into a warp of 32, with each thread warp performing the filling process concurrently for optimal efficiency.

Furthermore, once the impedance matrix and voltage vector have been completely assembled in GPU memory, the resulting dense linear system is solved directly using the highly optimized routines provided by the NVIDIA cuSOLVER library. This approach avoids the need for a hand-crafted GPU-based solver and ensures a robust, high-performance solution of the MoM system with minimal development overhead, allowing our efforts to remain focused on the multi-GPU acceleration strategies.

B. Multi-GPU IEDG-CBFM-SMWA

Having established the foundation of GPU-MoM, we now present the proposed SMWA-centric multi-GPU parallelization strategy, which restructures the solution phase to exploit hierarchical matrix parallelism across multiple devices. The steps follow.

B.1. Host-centric hierarchical matrix construction

The SMWA initiates with a recursive binary partitioning of basis functions. The CPU is responsible for constructing and managing the binary tree topology, while the computationally intensive task of filling the impedance matrix for each node is fully offloaded to the GPUs. To avoid write conflicts and to accommodate the hybrid RWG/half-RWG filling formulation, the matrix elements are computed using an edge-to-edge interaction scheme. Since the matrix filling processes for different nodes are mutually independent, they can be distributed across multiple GPUs for parallel execution. However, because the number of basis functions varies significantly across different levels of the tree, dynamic

workload distribution is required to prevent idle computing resources.

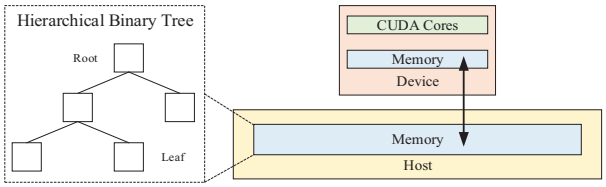


Fig. 3. SMWA's hierarchical binary tree is stored in the host memory. Matrices are only passed to the GPU memory when computation is needed, and retrieved after computation is complete.

The hierarchical matrix operations in the SMWA dictate a deliberate design choice regarding memory management. Unlike conventional GPU-accelerated MoM solvers that aim to retain the entire system matrix in device memory, our approach maintains the binary tree and all impedance submatrices in host memory, as illustrated in Fig. 3. This decision is motivated by the requirements of the SMWA. Specifically, during tree traversal, frequent manipulation of block submatrices within impedance matrices of parent nodes is required. Such operations are inefficient when implemented directly in device memory. Conversely, host memory offers the flexible random-access capabilities required for such manipulations, while also serving as a datasharing space accessible to all GPUs in a multi-GPU configuration.

Throughout the entire multi-GPU IEDG-CBFM-SMWA solution process, rather than just in this step, each GPU acts as a specialized computational accelerator. Relevant data will be transferred to GPU memory only when the host requires computationally intensive tasks, such as matrices multiplication or linear equations solution. Once the computation is complete, the results are returned to host memory, and the GPU memory is released to make room for subsequent tasks. This host-centric memory model, with GPUs dynamically invoked as co-processors, is specifically tailored to the hierarchical traversal pattern of the SMWA.

B.2. CBF generation and hierarchical matrix compression

When the hierarchical impedance matrix has been filled, the excitation-independent CBFs are generated at leaf nodes. In the present IEDG-CBFM-SMWA, a leaf node may contain both RWG and half-RWG basis functions. CBF generation therefore involves hybrid basis functions [23]. Since operations on individual leaf-nodes are strictly independent, they constitute an ideal workload for parallel dispatch. The host will allocate several threads corresponding to the number of leaf nodes in the binary tree, with each thread assigned to a GPU. Filling the impedance matrix and excitation vector leverage the same CUDA kernel functions as above, whereas matrix operations utilize routines from the cuBLAS and cuSOLVER libraries. Given that the number of leaf nodes typically exceeds the number of available GPU devices, multiple threads will be assigned to same GPU device. These threads exploit concurrent execution through dedicated CUDA streams, enabling overlapping data transfer operations and computation to minimize idle time and maximize hardware utilization, as shown in Fig. 4.

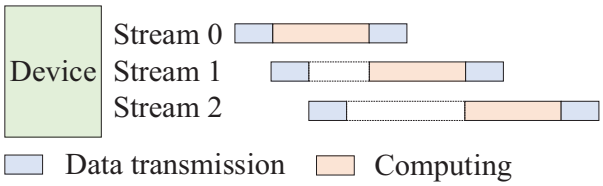


Fig. 4. Computation strategy based on multiple CUDA streams. This strategy overlaps device computation and data transmission to effectively hide data transmission time.

The generated CBF on the leaf nodes will be used to compress the hierarchical impedance matrix. Specifically, the compression phase propagates upward through the binary tree and, crucially, this process maintains node level parallelism. When multiple leaf nodes project their CBFs onto a shared parent node, their operations act on strictly disjoint impedance sub-blocks. This spatial separation inherently eliminates write-conflicts and allows concurrent compression across different branches of the tree without the need for any synchronization overhead. This conflict-free design maximizes GPU occupancy while completing the hierarchical reduction. This SMWA-driven compression strategy is absent in conventional GPU-CBFM implementations and constitutes a key of our multi-GPU acceleration framework.

B.3. SMWA-centric multi-GPU solver

The SMWA matrix factorization proceeds as a level-synchronous traversal of the hierarchical matrix, from the leaf nodes toward the root. Crucially, all nodes at a given level are mutually independent, with data dependencies existing only between different levels. This inherent level-wise independence provides the foundation for our multi-GPU parallelization strategy, as shown in Fig. 5. At the commencement of each level, the host dynamically allocates several threads corresponding to the number of nodes and each thread executes the

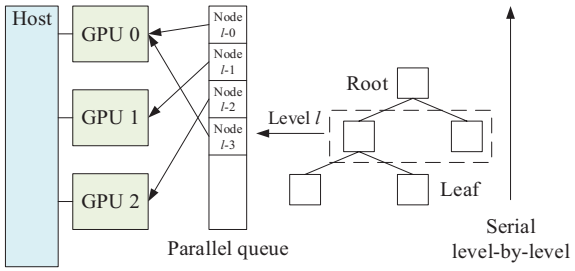


Fig. 5. Schematic diagram of parallel SMWA solution. In the hierarchical matrix, different levels are sequential, but all nodes at the same level are independent of each other.

complete SMWA computation sequence for its assigned node.

Hierarchical partitioning ensures that nodes at the same level possess a comparable number of basis functions. Consequently, the computational workload is intrinsically balanced. Moreover, to maximize hardware utilization, the mapping strategy adapts to the ratio of active nodes to available GPUs. When the number of GPU devices is insufficient relative to the node count, multiple threads will be assigned to the same GPU with task scheduling managed through different CUDA stream concurrency. Conversely, when GPU resources exceed node requirements, a one-to-one thread-to-device mapping is established to maximize parallel execution efficiency. The scenario where GPU resources exceed computational demands occurs exclusively near the root node, therefore, the resulting resource waste is negligible. This adaptive resource allocation strategy ensures optimal hardware utilization across varying problem scales while maintaining computational efficiency throughout the hierarchical solution process.

IV. NUMERICAL EXAMPLES

All examples in this paper were executed on a Dell Precision T7910 workstation. The platform comprises dual Intel Xeon E5-2690 v2 CPUs and three NVIDIA Quadro K6000 GPUs. The programs were compiled by NVIDIA nvcc 11.4 compiler. The CPU implementations utilize the Eigen for dense matrix operations, while the GPU implementations leverage linear algebra functions from the cuBLAS and cuSOLVER libraries.

To validate the accuracy of the proposed method, we computed the RCS of two canonical targets and compared the results with reference solutions obtained from the commercial solver Feko, as shown in Fig. 6. The two test cases are a cube with a side length of 1.0 m and a cylinder with a radius of 0.5 m and a height of 1.0 m. Both models are made of a perfect electric conductor (PEC) and discretized by non-conformal meshes and conformal meshes with an average edge length of $\lambda/10$ at a frequency of 2 GHz. The non-conformal mesh of the cube contains 56512 triangles, while the conformal mesh contains 54312 triangles. Similarly, for the cylinder, the non-conformal mesh contains 50944 triangles, while the conformal mesh contains 51058 triangles. The proposed GPU-accelerated IEDG-CBFM-SMWA uses non-conformal meshes and the Feko uses conformal meshes. As can be seen from Fig. 6, the results of the proposed method are in excellent agreement with those of Feko, confirming the accuracy of the proposed method.

To evaluate efficiency and memory usage, we compared the computational resources required by the single-GPU accelerated CBFM and the proposed method. Similarly, the proposed method uses non-conformal meshes and the GPU accelerated CBFM uses conformal meshes. The measured computation times are summarized in Tables 1 and 2. The “CBF Generation” step records the time required to generate CBFs, which operates on the partitioned blocks for CBFM, and on the leaf nodes of the binary tree for IEDG-CBFM-SMWA. The “Matrix Filling” step records the time to fill the reduced matrix of CBFM. The “Matrix Decomposition and Solution” step records the time for CBFM to solve the reduced matrix directly using the “cusolverDnCCgesv” routine in cuSOLVER, and for IEDG-CBFM-SMWA to decompose the hierarchical matrix via ACA and solve it. Tables 1 and 2 indicate that, although the matrix solving stage of GPU accelerated IEDG-CBFM-SMWA is slower than GPU-CBFM, it is still within an acceptable range.

For the cube model, the conventional MoM defines 85320 RWG / Half-RWG basis functions, and the IEDG-CBFM-SMWA compresses these unknowns into 11523 CBFs. Storing the reduced impedance matrix in single-precision complex format demands approximately 1 GB of GPU memory. In contrast, the hierarchical impedance matrix of the proposed IEDG-CBFM-SMWA requires a peak GPU memory of only 282 MB during the entire solution process. For the cylinder model, the corresponding memory usages are 0.8 GB versus 183 MB. These results demonstrate that the proposed method effectively exchanges a slight increase in solving hierarchical matrix time for a decisive reduction in memory usage. By keeping the peak memory usage well within the capacity of GPU memory, the method enables the direct solution of electrically large problems that would otherwise exceed the device memory limits. This substantial memory advantage combined with the parallel strategy constitutes the principal algorithmic benefit of the proposed method over GPU-CBFM approaches.

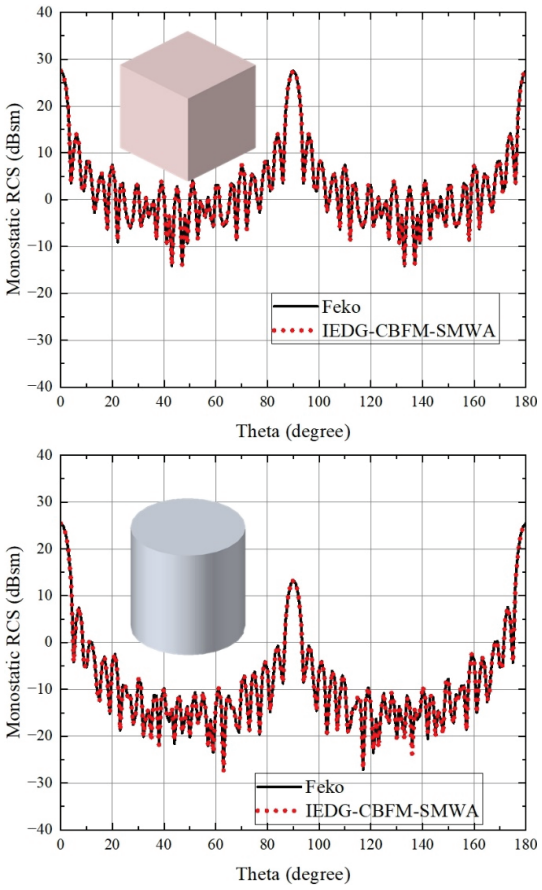


Fig. 6. Comparison of monostatic RCS results. The black curve represents the results of Feko, the red dot set represents the results of GPU-accelerated IEDG-CBFM-SMWA.

Table 1: Comparison of cube calculation times

Method	CBF Generation	Matrix Filling	Matrix Decomposition and Solution	Total
GPU-CBFM	45.8 s	1340 s	8.2 s	1390 s
GPU-IEDG-CBFM-SMWA	59.4 s	1520 s	16.5 s	1600 s

Table 2: Comparison of cylinder calculation times

Method	CBF Generation	Matrix Filling	Matrix Decomposition and Solution	Total
GPU-CBFM	56 s	1080 s	2.3 s	1140 s
GPU-IEDG-CBFM-SMWA	49.9 s	1192 s	15.4 s	1250 s

To further evaluate the proposed method on a complex, multiscale geometry, we constructed a drone model as shown in Fig. 7. The model measures 3.0 m in length, 8.3 m in wingspan, and was discretized into 38964 triangular elements. Owing to the geometric complexity of the wingtip assemblies, vertical stabilizer, and landing gear mechanisms, partial mesh refinement was applied in these regions, resulting in nonconformal mesh at the junctions with the primary airframe structures.

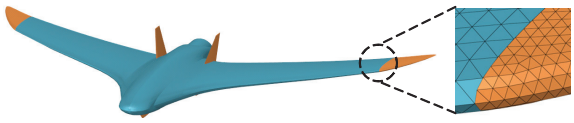


Fig. 7. Schematic of a drone model. The main structure is blue, and the detailed structure is orange. Because different structures use different mesh sizes, the meshes between the structures are nonconformal.

The monostatic RCS was computed at 750 MHz frequency. For reference, we also used Feko to calculate the results of the model being partitioned into a conformal mesh and compared the results with those of the proposed algorithm, as shown in Fig. 8. The results show that both IEDG-CBFM-SMWA achieve good agreement with Feko.

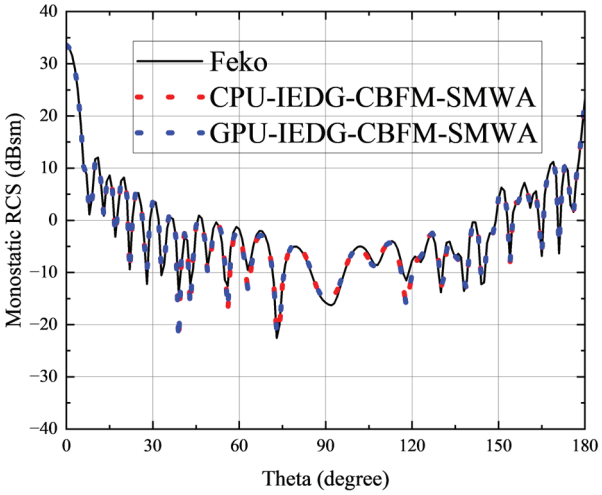


Fig. 8. Comparison of monostatic RCS results. Black curve represents the results of Feko, red dot set and blue dot set represent the results of CPU-based and GPU-based IEDG-CBFM-SMWA.

Table 3: Comparison of calculation times

Method	CBF Generation	Matrix Filling	Matrix Decomposition and Solution	Total
CPU-IEDG-CBFM-SMWA	1020 s	1401 s	21 s	2450 s
1 GPU-IEDG-CBFM-SMWA	48.8 s	662.9 s	11.6 s	724 s
3 GPU-IEDG-CBFM-SMWA	17.1 s	212.7 s	8.1 s	238 s

The computational efficiency of the IEDG-CBFM-SMWA was evaluated. The CPU version was parallelized via OpenMP with 20 CPU threads and the GPU versions were executed by single-GPU and three-GPU configurations. Table 3 shows a comparison of the solution time under different hardware. It can be seen that the multi-GPU accelerated IEDG-CBFM-SMWA achieved substantial computational speedup over the CPU version and demonstrated effective scaling from one to three GPUs.

V. CONCLUSION

This paper presents a multi-GPU accelerated IEDG-CBFM-SMWA. In our implementation, the integration of hierarchical matrix compression, direct solution techniques, and multi-GPU acceleration achieves substantially efficient electromagnetic scattering calculation. It requires less memory than the conventional GPU-accelerated CBFM, thus enabling the solution of larger electromagnetic scattering problems.

ACKNOWLEDGMENT

This work was supported in part by the National Nature Science Foundation of China under Grant 62371228, in part by Key Laboratory of Inlet and Exhaust System Technology, Ministry of Education, under Grant CEPE2024012, and in part by the Project of Key Laboratory of Radar Imaging and Microwave Photonics (Nanjing University of Aeronautics and Astronautics), Ministry of Education under Grant NJ20240005.

REFERENCES

[1] R. F. Harrington, *Field Computation by Moment Methods*. New York: MacMillan, 1968.

[2] W. C. Gibson, *The Method of Moments in Electromagnetics*. Boca Raton: CRC Press, 2007.

[3] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC," *IEEE Trans. Electromagn. Compat.*, vol. 47, no. 4, pp. 763–773, 2005.

[4] J. M. Tamayo, A. Heldring, and J. M. Rius, "Multilevel adaptive cross approximation (MLACA)," *IEEE Trans. Antennas Propag.*, vol. 59, no. 12, pp. 4600–4608, 2011.

[5] R. Coifman, V. Rokhlin, and S. Wandzura, "The fast multipole method for the wave equation: A pedestrian prescription," *IEEE Antennas Propag. Mag.*, vol. 53, no. 3, pp. 7–12, 1993.

[6] J. Song, C. C. Lu, and W. C. Chew, "Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects," *IEEE Trans. Antennas Propag.*, vol. 45, no. 10, pp. 1488–1493, 1997.

[7] S. X. Peng and Z. P. Nie, "Acceleration of the method of moments calculations by using graphics processing units," *IEEE Trans. Antennas Propag.*, vol. 56, no. 7, pp. 2130–2133, 2008.

[8] D. D. Donno, A. Esposito, G. Monti, and L. Tarricone, "Parallel efficient method of moments exploiting graphics processing units," *Microw. Opt. Technol. Lett.*, vol. 52, no. 11, pp. 2568–2572, 2010.

[9] E. Lezar and D. B. Davidson, "GPU-accelerated method of moments by example: Monostatic scattering," *IEEE Antennas Propag. Mag.*, vol. 52, no. 6, pp. 120–135, 2011.

[10] T. Topa, A. Karwowski, and A. Noga, "Using GPU with CUDA to accelerate MoM-based electromagnetic simulation of wire-grid models," *IEEE Antennas Wireless Propag. Lett.*, vol. 10, pp. 342–345, 2011.

[11] T. Topa, A. Noga, and A. Karwowski, "Adapting MoM with RWG basis functions to GPU technology using CUDA," *IEEE Antennas Wireless Propag. Lett.*, vol. 10, pp. 480–483, 2011.

[12] T. Topa, "Load-balanced Fortran-based out-of-GPU memory implementation of the method of moments," *IEEE Antennas Wireless Propag. Lett.*, vol. 16, pp. 813–816, 2017.

[13] M. Cwikla, J. Aronsson, and V. Okhmatovski, "Low-frequency MLFMA on graphics processors," *IEEE Antennas Wireless Propag. Lett.*, vol. 9, pp. 8–11, 2010.

[14] Q. M. Nguyen, V. Dang, O. Kilic, and E. El-Araby, "Parallelizing fast multipole method for large-scale electromagnetic problems using GPU clusters," *IEEE Antennas Wireless Propag. Lett.*, vol. 12, pp. 868–871, 2013.

[15] J. Guan, S. Yan, and J. M. Jin, "An OpenMP-CUDA implementation of multilevel fast multipole algorithm for electromagnetic simulation on multi-GPU computing systems," *IEEE Trans. Antennas Propag.*, vol. 61, no. 7, pp. 3607–3616, 2013.

[16] W. J. He, Z. Yang, X. W. Huang, W. Wang, M. L. Yang, and X. Q. Sheng, "Solving electromagnetic scattering problems with tens of billions of unknowns using GPU accelerated massively parallel MLFMA," *IEEE Trans. Antennas Propag.*, vol. 70, no. 7, pp. 5672–5682, 2022.

[17] J. I. Perez, E. García, J. A. de Frutos, J. R. Almagro, and M. F. Cátedra, "Application of GPU computing to the characteristic basis function method," in *2012 6th European Conference on Antennas and Propagation (EuCAP)*, Prague, Czech Republic, pp. 1003–1006, 2012.

[18] J. Zhang and Y. Sun, "Parallel SVD-CBFM based on GPU acceleration," in *2019 Cross Strait*

Quad-Regional Radio Science and Wireless Technology Conference (CSQRWC), Taiyuan, China, pp. 1–3, 2019.

[19] Y. Su and R. Mittra, "GPU-acceleration of characteristic basis function method (CBFM) for efficient analysis of complex platforms involving layered media," in *2019 USNC-URSI Radio Science Meeting*, Atlanta, GA, USA, pp. 43–44, 2019.

[20] R. Mittra, T. Marinovic, O. Ozgun, S. Liu, and R. K. Arya, "Novel strategies for efficient computational electromagnetic (CEM) simulation of microstrip circuits, antennas, arrays, and metamaterials part-II: Characteristic basis function method, perfectly matched layer, GPU acceleration," *Applied Computational Electromagnetics Society (ACES) Journal*, vol. 40, no. 06, pp. 471–498, 2025.

[21] X. Chen, C. Gu, Z. Li, and Z. Niu, "Accelerated direct solution of electromagnetic scattering via characteristic basis function method with Sherman-Morrison-Woodbury formula-based algorithm," *IEEE Trans. Antennas Propag.*, vol. 64, no. 10, pp. 4482–4486, 2016.

[22] Z. Peng, K. H. Lim, and J. F. Lee, "A discontinuous Galerkin surface integral equation method for electromagnetic wave scattering from nonpenetrable targets," *IEEE Trans. Antennas Propag.*, vol. 61, no. 7, pp. 3617–3628, 2013.

[23] Y. Su, R. Mittra, and W. Sheng, "Analysis of scattering from multi-scale and multiple targets using characteristic basis function (CBFM) and integral equation discontinuous Galerkin (IEDG) methods," *J. Electromagn. Waves Applicat.*, vol. 31, no. 10, pp. 969–980, 2017.

[24] E. Lucente, A. Monorchio, and R. Mittra, "An iteration-free MoM approach based on excitation independent characteristic basis functions for solving large multiscale electromagnetic scattering problems," *IEEE Trans. Antennas Propag.*, vol. 56, no. 4, pp. 999–1007, 2008.

[25] X. Fang, Q. Cao, Y. Zhou, and Y. Wang, "Multiscale compressed and spliced Sherman-Morrison-Woodbury algorithm with characteristic basis function method," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 3, pp. 716–724, 2018.



Ziang Shen received his Bachelor's degree in 2017 and his Master's degree in 2020 from Nanjing University of Aeronautics and Astronautics, China, and is presently pursuing the Ph.D. degree. His research focuses on high-performance computational electromagnetics, specifically frequency-domain, time-domain, and high-frequency asymptotic algorithms.



Xinlei Chen received the Ph.D. degree from Nanjing University of Aeronautics and Astronautics, China, in 2015. Currently, he is an Associate Professor at Nanjing University of Aeronautics and Astronautics. He has published over 80 papers in journals and conferences, including *IEEE Transactions on Antennas and Propagation* and *IEEE Antennas Wireless Propagation Letters*. His research interests include integral equation methods in electromagnetic problems.



Zhuo Li received the B.Sc. degree in electronic engineering and the M.E. degree in electromagnetic fields and microwave techniques from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 2001 and 2004, respectively, and the Ph.D. degree in radio engineering from the State Key Laboratory of Millimeter Waves, Southeast University, Nanjing, in 2009. From 2015 to 2016, he was a Visiting Scholar with the Department of Physics, Arizona State University, Tempe, AZ, USA. He is currently a Full Professor with the College of Electronic and Information Engineering in NUAA. He has authored or co-authored over 160 papers in refereed journals and conference proceedings, including *Advanced Science, Advanced Optical Materials, Nano Letters, Physical Review Applied, IEEE Transactions on Microwave Theory and Techniques, IEEE Transactions on Antennas and Propagation*, and *IEEE Transactions on Electromagnetic Compatibility*. He has filed more than 30 patents. His current research interests include plasmonic metamaterials, MHz-through-THz technologies and transceivers for wireless sensors and biomedical applications, electromagnetic compatibility in avionics system and auto electrical system. He is also interested in the modeling and design of microwave and terahertz photonic circuits and systems.

Li is a Senior Member of the Chinese Institute of Electronics and a Committee Member of the Antenna Branch of the Chinese Institute of Electronics. He received the Second Prize in Natural Science of the China Institute of Electronics in 2019 and the NUAA Outstanding Undergraduate Teaching Promotion Award in 2019. He is on the Advisory Board for *Journal of Physics D: Applied Physics* and on the Review Board for many international journals including *Advanced Materials, ACS Photonics, Nanotechnology, Nano-Micro Letters, IEEE T-MTT, IEEE T-AP*, and *IEEE T-EMC*. He has served as the Invited Speaker and Session Chair in many international conferences and workshops.