A CUDA Implementation of the Finite Element-Boundary Integral Method for Electromagnetic Scattering Simulation

Jian Guan*, Su Yan, and Jian-Ming Jin
Center for Computational Electromagnetics
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801-2991, USA
E-mail: jguan4@illinois.edu

Numerical analysis of electromagnetic scattering has been widely studied in the past. The method of moments (MoM) is suitable to simulate scattering from an object in an open region. In order to solve problems with complicated geometries and materials, the finite element method (FEM) is preferred. However, when solving unbounded problems with FEM, the use of absorbing boundary conditions or perfectly matched layers will result in truncation errors. To combine the advantages of both MoM and FEM, the finite element-boundary integral (FE-BI) method was developed and widely used (J. M. Jin, *The Finite Element Method in Electromagnetics, Second Edition*. New York: Wiley, 2002). This algorithm is not only able to deal with highly complicated geometry and material, but also able to eliminate the truncation error.

However, since the MoM has $O(N^2)$ computational and storage complexities, the computational cost of the BI-related algorithm is still prohibitively high when solving large-scale scattering problems. To alleviate this problem, the multilevel fast multipole algorithm (MLFMA) has been incorporated into the FE-BI method (J. Liu and J. M. Jin, *IEEE Trans. Antennas Propag*., vol. 51, pp.1157–1167, Jun. 2003). Another important technique to accelerate the computation is to resort to parallel computers. Recently, the graphics processing unit (GPU), as one of many-core computing systems, has received intensive attention from the scientific computing community due to its low price, a massive number of processors, and the resulting high computational efficiency.

In this work, we discuss the GPU implementation of the FE-BI algorithm for solving electromagnetic scattering problems. First, the assembly of the BI-related matrix is parallelized on GPU by NVIDIA's compute unified device architecture (CUDA). A massive number of threads are assigned to carry out the assembly, and each thread computes one element in the BI-related matrix. Second, both the direct and iterative solvers are implemented on GPU. For the direct solver, the Intel's math kernel library (MKL) is used for LU decomposition, and the CUSPARSE (a set of basic linear algebra subroutines used for handling sparse matrices) is employed to speedup forward and backward substitutions. For the iterative solver, the high performance CUDA Basic Linear Algebra Subprograms (BLAS) is utilized for the matrix-vector multiplication and the vector-vector multiplication. Extensive numerical experiments are conducted to validate the implementation and demonstrate the achievable computational speedup.