Speed efficiency optimization for GPU accelerated Rigorous Coupled-Wave Analysis program

Jingxiao Xu[a], Martin D. B. Charlton[a]

[a]University of Southampton, University Rd, Southampton SO17 1BJ, United Kingdom

ABSTRACT

Rigorous Coupled Wave Analysis (RCWA) method is highly efficient for the simulation of diffraction efficiency and field distribution patterns in periodic structures and textured optoelectronic devices such as VCSELs, LEDs, and DOEs. RCWA provides exact solutions provided the Fourier expansion has infinite order. In practice, the Fourier expansion must be truncated due to computer memory limitations. Researchers are trying to utilize fast convergence algorithms such as the 'normal vector method' and 'Li's rule' which could obtain accurate TM mode results with fewer Fourier orders. However, to thoroughly investigating the behavior of a structure usually requires thousand and even millions of RCWA simulations which may last hours and days. GPU is highly suitable for solutions of complex systems allowing large-scale multi-threaded parallel programming (> 1000 / low-end GPU, >5k / high-end GPU) to speed up matrix computations significantly. In this paper, we present a high-speed RCWA program utilizing optimized CUDA-GPU code and MAGMA libraries. It achieves 2-6 X speedup compared to conventional multithreaded CPU-based code utilizing the Intel MKL library running on IRIDIS 5 super-computer (1 NVIDIA v100 GPU, 40 Intel Xeon Gold 6138 2.0GHz cores CPU)

Keywords: GPU computing, RCWA, periodic structure, MKL library, supercomputer

1. INTRODUCTION

Photonic crystal is a periodic structure that can control light propagation inside a material [1]. It has many applications such as improving waveguide transmission [2][3], high efficiency LED [4], solar cell [5] and VCSEL [6]. A fast and accurate simulation tool is necessary for researchers to produce advanced and high efficiency photonic structures. Many computational electromagnetism (CEM) tools can be used to simulate photonic crystal. the finite-difference time-domain (FDTD) method [7], finite-difference frequency-domain (FDFD) method [8], and the finite element method (FEM) [9], RCWA [10][11]. Each method has advantages and disadvantages. RCWA was firstly invented to solve diffraction of a 1D photonic crystal (grating) device [10]. And soon 2D cross grating and arbitrary device profile [12] are covered by RCWA. 3D problems can also be solved by slicing into multilayer structure which is then connected with scattering matrix [13] or enhanced transfer matrix methods [14]. The Fourier expansion technique of RCWA produces convenient and fast mode spectrum simulation of periodic structures [15].

RCWA gives exact solutions provided the Fourier expansion has infinite order. In reality, the Fourier expansion must be truncated due to computer memory limitations. To have an accurate RCWA simulation with less Fourier expansion, many extension methods are implemented to obtain faster convergence rate, such as normal vector method [16] and Li's rule [17].

Besides improving convergence rate of RCWA algorithm, the computational time of a RCWA process can also be improved with computer techniques. RCWA algorithm composes many matrix operations such as matrix multiplication/inversion, 2D matrix Fast Fourier Transform, eigen-system etc. Large matrix sizes are required if high accuracy is required which means large number of Fourier harmonics are included. Parallel computing is an obvious way of improving speed efficiency of matrix operations.

The General-Purpose Graphics Processing Units (GPGPU) has been increasingly used for many scientific problems such as the COVID-19 spread model and AI training [18][19]. The parallel architecture of hardware allows GPU to speed up matrix operation significantly. Initially, GPUs are used for manipulating graphics and image processing. The design was focusing on single precision floating point operations including NVIDIA GTX/RTX series. As high accuracy is required

Physics and Simulation of Optoelectronic Devices XXXI, edited by Bernd Witzigmann, Marek Osiński, Yasuhiko Arakawa, Proc. of SPIE Vol. 12415, 124150J
 · 0277-786X · doi: 10.1117/12.2650022

for scientific problems, NVIDIA developed Data Center series, such as Tesla V100 and A100, which has much more double precision floating point processing speed [20][21].

In this work, we conduct our experiments on the iridis 5 supercomputer. 1 NVIDIA Tesla V100 GPU and 40-cores Intel Xeon Gold 6138 CPU resources are used. Time efficiency of RCWA programs implemented with CPU library and optimized GPU libraries is investigated. We achieved at least 2X speedup using GPU computing technique.

2. RCWA ALGORITHM

The basic idea of RCWA algorithm is to analyze Maxwell's equations in Fourier space. The schematic of a photonic crystal is shown in figure 1. Only the unit cell in transverse plane is necessary to be constructed for RCWA simulation. The device is divided into layers based on the homogeneity in vertical direction. Generally, a device has at least three layers: reflection region, device layer and transmission region. In each layer, the material is homogeneous in z axis and periodic in x-y plane. The electromagnetic wave propagation property in one layer is firstly deduced. Solutions form layers are then connected with boundary conditions. Together with the incident field profile, RCWA calculates the reflection and transmission information of the device.

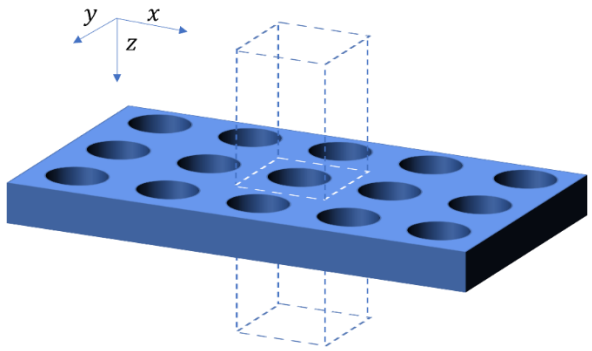

Figure 1. Schematic of a photonic crystal. The rectangle cuboid region surrounded by dotted line is the structure constructed in RCWA algorithm.

2.2 RCWA Layer solution

The formulation of RCWA start with Maxwell's equation. The Maxwell's equations describe the relation between the magnetic and electric fields with their relative permittivity ε_r and permeability μ_r.

$$\nabla \times \mathrm{E} = k_0 \mu_r \tilde{H} \tag{1}$$

$$\nabla \times \tilde{H} = k_0 \varepsilon_r E \tag{2}$$

Fourier transform is then applied to E and H fields resolved in x-y Cartesian coordinates in real space, representing them as a sum of frequency components in Fourier (frequency domain) space.

$$E_i(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} S_i e^{-j(k_x x + k_y y)} \tag{3}$$

$$\tilde{H}_i(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} U_i e^{-j(k_x x + k_y y)} \tag{4}$$

$$\varepsilon_r(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} a_{m,n} e^{j(mT_x x + nT_y y)} \tag{5}$$

$$\mu_r(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} b_{m,n} e^{j(mT_x x + nT_y y)} \tag{6}$$

Where $k_i = k_i^{inc} - mT_i$. k_i is wave vectors. T_i is reciprocal space (Fourier space) lattice unit length. $T_i = \frac{2\pi}{\Lambda_i}$. Λ_i is periodicity in i direction. i represents direction x or y. m is any integer.

Expanding the equation (1) and (2).

$$\frac{\partial \widetilde{H_z}}{\partial y} - \frac{\partial \widetilde{H_y}}{\partial z} = k_0 \varepsilon_r E_x \tag{7}$$

$$\frac{\partial \widetilde{H_x}}{\partial z} - \frac{\partial \widetilde{H_z}}{\partial x} = k_0 \varepsilon_r E_y \tag{8}$$

$$\frac{\partial \widetilde{H_y}}{\partial x} - \frac{\partial \widetilde{H_x}}{\partial y} = k_0 \varepsilon_r E_z \tag{9}$$

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = k_0 \mu_r \widetilde{H}_x \tag{10}$$

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = k_0 \mu_r \widetilde{H}_y \tag{11}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = k_0 \mu_r \widetilde{H}_z \tag{12}$$

Two first differential equations are obtained by substitute (3)-(6) into (7)-(12) and eliminate z filed components.

$$\frac{\mathrm{d}}{\mathrm{d}\tilde{z}} \begin{bmatrix} \boldsymbol{u_x} \\ \boldsymbol{u_y} \end{bmatrix} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{s_x} \\ \boldsymbol{s_y} \end{bmatrix} \tag{13}$$

$$\frac{\mathrm{d}}{\mathrm{d}\tilde{z}} \begin{bmatrix} \boldsymbol{s_x} \\ \boldsymbol{s_y} \end{bmatrix} = \boldsymbol{P} \begin{bmatrix} \boldsymbol{u_x} \\ \boldsymbol{u_y} \end{bmatrix} \tag{14}$$

Then a second differential is deduced.

$$\frac{\mathrm{d}^2}{\mathrm{d}\tilde{z}^2} \begin{bmatrix} \boldsymbol{s_x} \\ \boldsymbol{s_y} \end{bmatrix} = \boldsymbol{\Omega^2} \begin{bmatrix} \boldsymbol{s_x} \\ \boldsymbol{s_y} \end{bmatrix} \tag{15}$$

Where $\Omega^2 = \mathrm{PQ}$.

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{K}_x [[\mu_r]]^{-1} \boldsymbol{K}_y & [[\varepsilon_r]] - \boldsymbol{K}_x [[\mu_r]]^{-1} \boldsymbol{K}_x \\ \boldsymbol{K}_y [[\mu_r]]^{-1} \boldsymbol{K}_y - [[\varepsilon_r]] & -\boldsymbol{K}_y [[\mu_r]]^{-1} \boldsymbol{K}_x \end{bmatrix} \tag{16}$$

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{K}_x [[\varepsilon_r]]^{-1} \boldsymbol{K}_y & [[\mu_r]] - \boldsymbol{K}_x [[\varepsilon_r]]^{-1} \boldsymbol{K}_x \\ \boldsymbol{K}_y [[\varepsilon_r]]^{-1} \boldsymbol{K}_y - [[\mu_r]] & -\boldsymbol{K}_y [[\varepsilon_r]]^{-1} \boldsymbol{K}_x \end{bmatrix} \tag{17}$$

Where $\boldsymbol{K_x}$ and $\boldsymbol{K_y}$ are diagonal matrices with elements are normalized wave vectors $k_x(m,n)/k_0$ and $k_y(m,n)/k_0$. k_0 is free space wave vector. $k_0 = \frac{2\pi}{\lambda_0}$. $[[\mu_r]]$ and $[[\varepsilon_r]]$ are 2D convolution matrices of permeability and permittivity in Fourier space. The field solution of this layer can be written as

$$\begin{bmatrix} \boldsymbol{s_x} \\ \boldsymbol{s_y} \end{bmatrix} = \boldsymbol{W} e^{-\lambda \tilde{z}} \boldsymbol{c}^+ + \boldsymbol{W} e^{\lambda \tilde{z}} \boldsymbol{c}^- \tag{18}$$

$$\begin{bmatrix} \boldsymbol{u_x} \\ \boldsymbol{u_y} \end{bmatrix} = -\boldsymbol{V} e^{-\lambda \tilde{z}} \boldsymbol{c}^+ + \boldsymbol{V} e^{\lambda \tilde{z}} \boldsymbol{c}^- \tag{19}$$

Where $\boldsymbol{W}$ and $\boldsymbol{\lambda^2}$ are eigenvectors and eigenvalues of matrix Ω^2. V is calculated based on the electric field solution by substitute equation () into (). $\boldsymbol{V} = \boldsymbol{Q}\boldsymbol{W}\boldsymbol{\lambda}^{-1}$. $\boldsymbol{c}^+ = \boldsymbol{W}^{-1}\boldsymbol{s}^+$ and $\boldsymbol{c}^- = \boldsymbol{W}^{-1}\boldsymbol{s}^-$. $\boldsymbol{s}^+$ and $\boldsymbol{s}^-$ are Fourier coefficients of initial electric field amplitudes in forward and backward direction.

2.3 Scattering matrix

To connect solutions of multi layers, scattering matrix method is used for high numerical stability [13]. Scatter matrix is generally used to connect input and output ports of a system. Each layer i has its own scattering matrix $\boldsymbol{S_i} = \begin{bmatrix} \boldsymbol{S_{11}^i} & \boldsymbol{S_{12}^i} \\ \boldsymbol{S_{21}^i} & \boldsymbol{S_{22}^i} \end{bmatrix}$. In this section, the scattering matrix uses capital $\boldsymbol{S}$ notation to distinguish from the electric field Fourier coefficient $\boldsymbol{s}$.

$$\mathrm{S_{11}^i} = \mathrm{S_{22}^i} = \left(\mathrm{A_i\text{-}X_iB_iA_i^{-1}X_iB_i}\right)^{-1}\left(\mathrm{X_iB_iA_i^{-1}X_iA_i\text{-}B_i}\right) \tag{20}$$

$$\boldsymbol{S_{12}^i} = \boldsymbol{S_{21}^i} = \left(\boldsymbol{A_i} - \boldsymbol{X_iB_iA_i^{-1}X_iB_i}\right)^{-1}\boldsymbol{X_i}\left(\boldsymbol{A_i} - \boldsymbol{B_iA_i^{-1}B_i}\right) \tag{21}$$

Where $\boldsymbol{A_i} = \boldsymbol{W_i^{-1}W_0} + \boldsymbol{V_i^{-1}V_0}, \boldsymbol{B_i} = \boldsymbol{W_i^{-1}W_0} - \boldsymbol{V_i^{-1}V_0}$ and $\boldsymbol{X_i} = e^{-\lambda_i k_0 L_i}$

A global S matrix $[\boldsymbol{S}]$ is used to connect incident, reflected and transmitted waves.

$$\begin{bmatrix} c_{ref}^{-} \\ c_{trn}^{+} \end{bmatrix} = [S] \begin{bmatrix} c_{inc}^{+} \\ 0 \end{bmatrix} \tag{22}$$

$$[\boldsymbol{S}] = \left[\boldsymbol{S_{ref}}\right] \otimes [\boldsymbol{S_1}] \otimes [\cdots] \otimes [\boldsymbol{S_i}] \otimes [\boldsymbol{S_{trn}}] \tag{23}$$

Then the electric field can be reconstructed using Eqn. (3) with an incident field and connecting multilayer S-matrix with Redheffer Star Product [13]. The Redheffer Star product is defined as

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \otimes \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \tag{24}$$

$$\boldsymbol{C_{11}} = \boldsymbol{A_{11}} + \boldsymbol{A_{12}}[\boldsymbol{I} - \boldsymbol{B_{11}A_{22}}]^{-1}\boldsymbol{B_{11}A_{21}} \tag{25}$$

$$\boldsymbol{C_{12}} = \boldsymbol{A_{12}}[\boldsymbol{I} - \boldsymbol{B_{11}A_{22}}]^{-1}\boldsymbol{B_{12}} \tag{26}$$

$$\boldsymbol{C_{21}} = \boldsymbol{B_{21}}[\boldsymbol{I} - \boldsymbol{A_{22}B_{11}}]^{-1}\boldsymbol{A_{21}} \tag{27}$$

$$\boldsymbol{C_{22}} = \boldsymbol{B_{22}} + \boldsymbol{B_{212}}[\boldsymbol{I} - \boldsymbol{A_{22}B_{11}}]^{-1}\boldsymbol{A_{22}B_{12}} \tag{28}$$

The incident field profile is known. Reflected and transmitted waves are easily obtained.

$$\boldsymbol{c_{ref}^{-}} = \boldsymbol{S_{11}c_{inc}^{+}} \tag{29}$$

$$\boldsymbol{c_{trn}^{+}} = \boldsymbol{S_{21}c_{inc}^{+}} \tag{30}$$

$$\boldsymbol{s_{ref}} = \boldsymbol{W_{ref}c_{ref}^{-}} \tag{31}$$

$$\boldsymbol{s_{trn}} = \boldsymbol{W_{trn}c_{trn}^{+}} \tag{32}$$

One of important concepts in RCWA is number of harmonics *h*, it defines the number of Fourier coefficients used in Fourier transform, and effects accuracy of the simulation results. In this paper and later experiment, number of harmonics *h* equals to 2N+1. Where N is the maximum value of m or n used in equations (3) to (6).

3. METHOD

3.1 GPU computing

The hardware design of GPU allows thousands of threads to operate concurrently. To utilize the power of GPU, our work chooses CUDA (Compute Unified Device Architecture) C++ as programming platform. There are primarily two ways to program in GPU. One is using prebuilt GPU libraries. CUDA provides a series of math libraries including CUBLAS, CUFFT and CUSOLVER. There are also other libraries that is compatible with NVIDIA GPUs. MAGMA is a linear algebra library built for heterogeneous GPU-based architectures and developed by the same team that developed LAPACK. The other way is using self-defined kernels which has more flexibility. Self-defined kernels allow user to design their own functions. Libraries may not be able to provide all the functions that a particular algorithm required. CUDA programming model allow programmer to define the number of threads and blocks used in a function. A typical CUDA kernel, for example, a vector add function, starts from defining matrices in CPU memory. Then the data is transferred to GPU memory to be added simultaneously. Finally, the data is transferred back to CPU for further calculation.

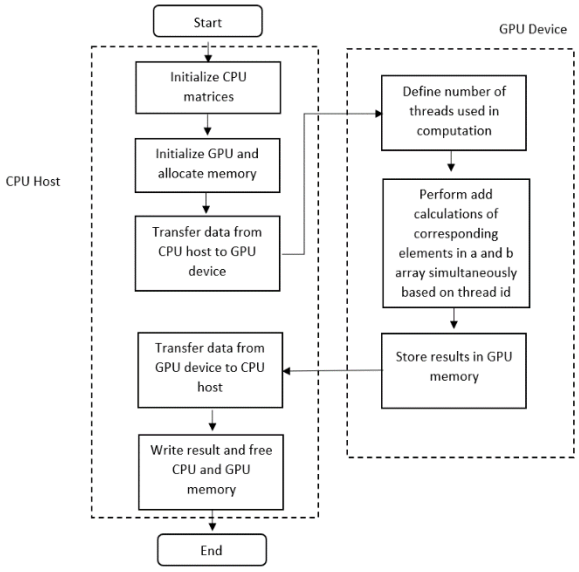

Figure 2. Vector-add GPU kernel example which can be applied to other GPU programs.

3.2 Matrix operations

Matrix operations are dominant in the RCWA algorithm. Various types of matrix operations are in the RCWA algorithm. Equation (18) solves eigenvalues and eigenvectors. Equations (16)-(17), (20)-(21) and (25) to (28) involves matrix inversions. Fast Fourier transform is employed in equations (3) to (6). And matrix multiplications are used all over the algorithm. There are few other elementwise operations such as matrix addition in equations (16) (17) and diagonal matrix exponential in equations (18) (19). Most operations solver can be found in GPU libraries. We employ MAGMA library for eigen system (eigenvalues and eigenvectors) calculation and matrix inversion. CUBLAS and CUFFT are used for matrix multiplication and fast Fourier transform respectively. Other matrix operations including matrix elementwise addition, exponential, and other special functions are solved using our own designed CUDA kernels.

Size of matrix is an important factor for computational speed. It is determined by the number of harmonics. For Fourier transform matrix size is h x h. Eigenvalue problem, most of matrix multiplications and inversions and most of other operations solve matrices with size h^2 x h^2. Therefore, if 70 harmonics are used which may be required by complex devices, the matrices size are around 10k x10k. And for simple structure designs, 7 harmonics are sufficient with matrix size 100x100.

4. PERFORMANCE

A RCWA algorithm utilizing Intel MKL library is implemented in C++ as a reference. Intel MKL is a multithreaded version of LAPACK which have a good performance for multicore CPU system. All the tests including the CPU reference program are conducted on iridis 5 supercomputer. CPU program utilizes only the CPU resources. However, due to that eigenvalue problem solver is only partially accelerated on GPU, the GPU program utilizes both GPU and CPU resources.

Table 1. The hardware configuration of this experiment [22][23].

CPU	2X Intel Xeon Gold 6138 Processors
CPU cores	40
CPU base clock	2.00GHz
CPU max clock	3.70GHz
Main memory	188GB
GPU	NVIDIA Tesla V100
GPU CUDA cores	5120
GPU base clock	1230MHz
GPU boost clock	1380MHz
GPU memory	32GB

The example structure of the following simulations is a single layer photonic crystal design with square robs. The rob has relative permittivity $\epsilon_r = 3.0$. Other spaces are air with $\epsilon_r = 1.0$. Periodicity in x and y directions are $0.6\mu m$.

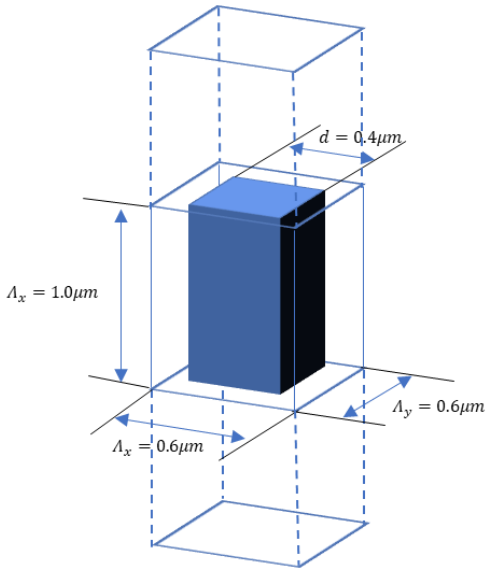

Figure 3. Total run time comparison between GPU and CPU RCWA programs

GPU computing is efficient. However, the data transfer between CPU and GPU memories costs extra time. A pair of memory transfer operations, including both CPU to GPU and GPU to CPU, takes up to 0.7 secs for a 10k * 10k matrix (fig 4) on one NVIDIA Tesla V100 GPU. However, in the single RCWA algorithm, there will be hundreds of such operations which will account for a large proportion of total runtime.

There are two methods to solve this issue. First, matrices could be stored only in GPU memory and freed when the program exit. Alternatively, the data transfer process can be hidden behind a GPU calculation kernel by asynchronous memory copy function. The second method can be used when the problem size is so large that GPU is out of memory. Under the context of this test, we choose the first method to avoid CPU-GPU data transfers.

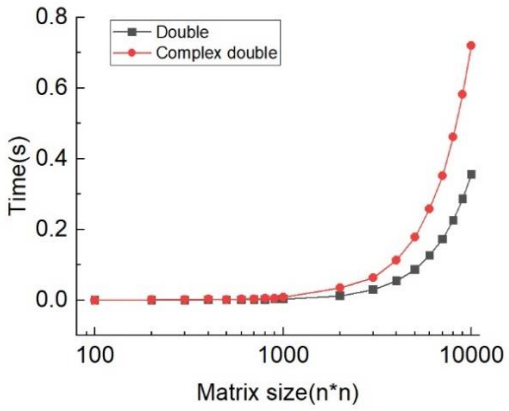

Figure 4. Time cost of transferring data between GPU memory and main memory. Gray line represents matrix with double precision. Red line also represents double precision but with complex value format.

A set of RCWA simulation with GPU and CPU implementations is performed. In figure 5, runtime of both versions is compared as a function of number of harmonics ranging from 3 to 49. The step size between each simulation is 2 which is the minimum number that harmonic can be incremented. The right y-axis of fig 5 shows the speedups of GPU over CPU version which is defined as T_{CPU}/T_{GPU}.

For very small number of Fourier harmonics (3 and 5), due to the small size of matrix, highly paralleled calculation in GPU may not gain enough benefit and is slower than CPU. Then, GPU RCWA performs around 3-6x faster when number of Fourier harmonics is 6-20. As number of harmonics increases further, speedup converges to a steady 2x. The initial increasing trend is possibly due to that the execution time in CPU is at milliseconds level which could be largely affected by overhead of allocating threads and non-computational operations. And GPU performs inefficiently with few threads. As matrix size gradually increasing, the overhead may still affect the CPU performance while occupancy of GPU keeps growing. Then, a steady trend is seen. Overhead involved in allocating threads become insignificant. Both CPU and GPU are processing efficiently.

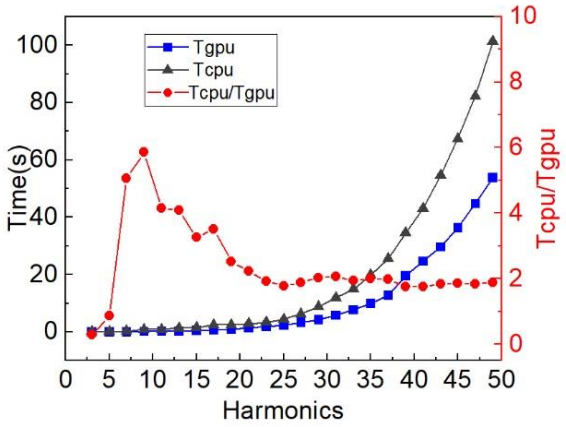

Figure 5. Total run time comparison between GPU and CPU RCWA programs

The selection of harmonics depends on the complexity of the specific structure. Large number of harmonics is required to describe small feature. Fig 6 plots reflection as function of number of harmonics of two structures: square rod model

in fig 3 and grating structure in fig 7. Both plots perform simulations with harmonics from 3 to 29 with step size of 2. Fig 6a shows convergence for rod structure at 15 harmonics. For simpler structure such as grating in fig 7, the results may converge at 9 or lower harmonics shown in fig 6b. Circle hole model have similar convergence harmonics as square rod. However, the smaller the size of sector which can be a hole, a unit cell subtracting a hole or a square or any other shape in the layer, the higher the harmonics required.

The required accuracy and the value of permittivity or permeability also affect the choice of harmonics. The higher relative permittivity contrast between different materials in a unit cell results in high number of Fourier coefficients to represent the grid. Therefore, testing for convergence is the only way to determine the proper harmonics of a structure.

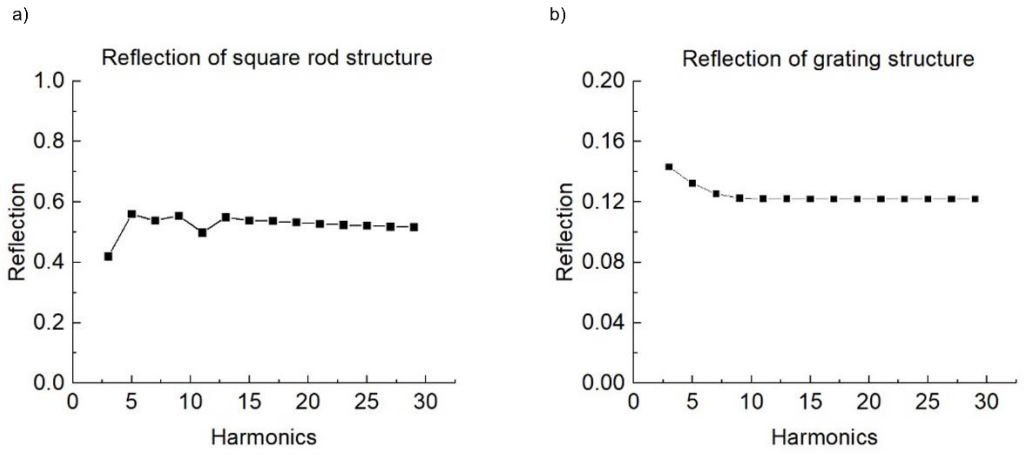

Figure 6. a) left side is the reflection of the square rod structure with TE mode plane wave incident at 10^0 angles. b) shows convergence of solution for reflection from the grating structure depicted in figure 7

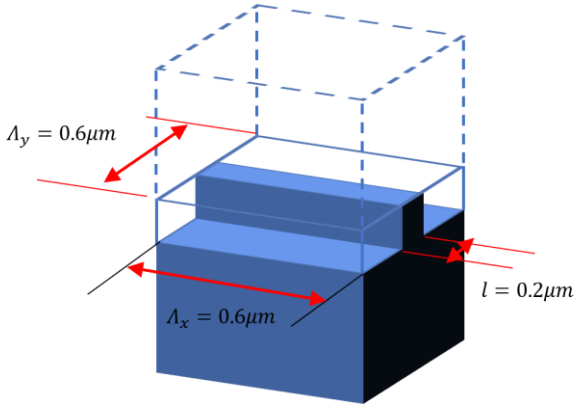

Figure 7. Schematic of a grating design. The relative permittivity of the blue space equals to 4.1616. Other area is air with $\epsilon_r = 1$

5. CONCLUSION

In this paper, we have described how a GPU RCWA program is built. And we have shown that the RCWA program with GPU accelerated routines achieved 2-6x speedup against 40-cores CPU version. At 3 or 5 harmonics, CPU performs faster than GPU due to loading GPU context and too short runtime. In practice, we see that for different structure the convergence rate of diffraction efficiency varies. Square rob needs at least 15 harmonics. While grating structure reach convergence around 7 harmonics. It is only necessary to use harmonics higher than 15 for more complex structure such as a stricture whose unit cell has more than one circle and other designs that have many features in one unit cell. For most of classic photonic crystal designs such as circle hole cubic, circle hole hexagonal, rectangle hole and grating structures, our GPU RCWA program have at least 3x speedup.

REFERENCES

[1] Joannopoulos, J. D.; Johnson, S. G.; Winn, J. N. & Meade, R. D. (2008), Photonic Crystals: Molding the Flow of Light (Second Edition), Princeton University Press.

[2] Torrijos-Morán, L., Griol, A. & García-Rupérez, J. Slow light bimodal interferometry in one-dimensional photonic crystal waveguides. Light Sci Appl 10, 16 (2021). https://doi.org/10.1038/s41377-020-00460-y

[3] Sharee J. McNab, Nikolaj Moll, and Yurii A. Vlasov, "Ultra-low loss photonic integrated circuit with membrane-type photonic crystal waveguides," Opt. Express 11, 2927-2939 (2003)

[4] Wierer, J., David, A. & Megens, M. III-nitride photonic-crystal light-emitting diodes with high extraction efficiency. Nature Photon 3, 163–169 (2009). https://doi.org/10.1038/nphoton.2009.21

[5] Peter Bermel, Chiyan Luo, Lirong Zeng, Lionel C. Kimerling, and John D. Joannopoulos, "Improving thin-film crystalline silicon solar cell efficiencies with photonic crystals," Opt. Express 15, 16986-17000 (2007)

[6] Dae-Sung Song, Se-Heon Kim, Hong-Gyu Park, Chang-Kyu Kim, and Yong-Hee Lee , "Single-fundamental-mode photonic-crystal vertical-cavity surface-emitting lasers", Appl. Phys. Lett. 80, 3901-3903 (2002) https://doi.org/10.1063/1.1481984

[7] A. Lavrinenko, P. I. Borel, L. H. Frandsen, M. Thorhauge, A. Harpøth, M. Kristensen, T. Niemi, and H. M. H. Chong, "Comprehensive FDTD modelling of photonic crystal waveguide components," Opt. Express 12, 234-248 (2004)

[8] Chin-ping Yu and Hung-chun Chang, "Compact finite-difference frequency-domain method for the analysis of two-dimensional photonic crystals," Opt. Express 12, 1397-1408 (2004)

[9] F. Brechet, J. Marcou, D. Pagnoux, P. Roy, Complete Analysis of the Characteristics of Propagation into Photonic Crystal Fibers, by the Finite Element Method, Optical Fiber Technology, Volume 6, Issue 2, 2000, Pages 181-191, ISSN 1068-5200, https://doi.org/10.1006/ofte.1999.0320. https://www.sciencedirect.com/science/article/pii/S1068520099903206

[10] M. G. Moharam and T. K. Gaylord, "Rigorous coupled-wave analysis of grating diffraction— E-mode polarization and losses," J. Opt. Soc. Am. 73, 451-455 (1983)

[11] Shi, J., Pollard, M.E., Angeles, C.A. et al. Photonic crystal and quasi-crystals providing simultaneous light coupling and beam splitting within a low refractive-index slab waveguide. Sci Rep 7, 1812 (2017). https://doi.org/10.1038/s41598-017-01842-w

[12] Thomas Schuster, Johannes Ruoff, Norbert Kerwien, Stephan Rafler, and Wolfgang Osten, "Normal vector method for convergence improvement using the RCWA for crossed gratings," J. Opt. Soc. Am. A 24, 2880-2890 (2007)

[13] Raymond C. Rumpf, "Improved Formulation of Scattering Matrices for Semi-Analytical Methods That Is Consistent with Convention," Progress In Electromagnetics Research B, Vol. 35, 241-261, 2011. doi:10.2528/PIERB11083107, http://www.jpier.org/PIERB/pier.php?paper=11083107

[14] M. G. Moharam, Drew A. Pommet, Eric B. Grann, and T. K. Gaylord, "Stable implementation of the rigorous coupled-wave analysis for surface-relief gratings: enhanced transmittance matrix approach," J. Opt. Soc. Am. A 12, 1077-1086 (1995)

[15] Paulsen, M., Neustock, L.T., Jahns, S. et al. Simulation methods for multiperiodic and aperiodic nanostructured dielectric waveguides. Opt Quant Electron 49, 107 (2017). https://doi.org/10.1007/s11082-017-0918-6

[16] Evgeni Popov and Michel Nevière, "Grating theory: new equations in Fourier space leading to fast converging results for TM polarization," J. Opt. Soc. Am. A 17, 1773-1784 (2000)

[17] Lifeng Li, "Use of Fourier series in the analysis of discontinuous periodic structures," J. Opt. Soc. Am. A 13, 1870-1876 (1996)

[18] Athanasios Voulodimos, Eftychios Protopapadakis, Iason Katsamenis, Anastasios Doulamis, and Nikolaos Doulamis. 2021. Deep learning models for COVID-19 infected area segmentation in CT images. In The 14th PErvasive Technologies Related to Assistive Environments Conference (PETRA 2021). Association for Computing Machinery, New York, NY, USA, 404–411. https://doi.org/10.1145/3453892.3461322

[19] Y. Wang et al., "Benchmarking the Performance and Energy Efficiency of AI Accelerators for AI Training," 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), Melbourne, VIC, Australia, 2020, pp. 744-751, doi: 10.1109/CCGrid49817.2020.00-15.

[20] CUDA-Enabled Datacenter Products (2022) Your GPU Compute Capability. Available at: https://developer.nvidia.com/cuda-gpus (Accessed: January 24, 2023).

[21] Features and Technical Specifications (2023) CUDA C++ Programming Guide. Available at: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html (Accessed: January 24, 2023).

[22] Nvidia Tesla V100 (no date) NVIDIA. Available at: https://www.nvidia.com/en-gb/data-center/tesla-v100/ (Accessed: January 24, 2023).

[23] Intel® Xeon® Gold 6138 processor (no date) Intel. Available at: https://www.intel.co.uk/content/www/uk/en/products/sku/120476/intel-xeon-gold-6138-processor-27-5m-cache-2-00-ghz/specifications.html (Accessed: January 24, 2023).