GPU 컴퓨팅에 의한 고속 Double Random Phase Encoding

사이플라흐, 문인규*
조선대학교 컴퓨터공학부
*corresponding author: inkyu.moon@chosun.ac.kr

Fast Double Random Phase Encoding by Using Graphics Processing Unit

Saifullah, Inkyu Moon
School of Computer Engineering, Chosun University

Abstract

With the increase of sensitive data and their secure transmission and storage, the use of encryption techniques has become widespread. The performance of encoding majorly depends on the computational time, so a system with less computational time suits more appropriate as compared to its contrary part. Double Random Phase Encoding (DRPE) is an algorithm with many sub functions which consumes more time when executed serially; the computation time can be significantly reduced by implementing important functions in a parallel fashion on Graphics Processing Unit (GPU). Computing convolution using Fast Fourier transform in DRPE is the most important part of the algorithm and it is shown in the paper that by performing this portion in GPU reduced the execution time of the process by substantial amount and can be compared with MATALB for performance analysis. NVIDIA graphic card GeForce 310 is used with CUDA C as a programming language.

1. Introduction

Reliable encoding methods for documents and images are great need of time to ensure the information security of individuals, organizations and governments by protecting against potential threats. As the amount of sensitive data being collected increases, a new interesting dilemma is being face by authorities; how to store data securely while still being able to access it quickly. While the standard Digital Holographic Encoding is both secure and efficient in terms of time and memory requirements, the amount of information that must encrypted simultaneously leads response times that are inadequate. Several performance increases for these algorithms have been [1] found but these too are not always quick enough, so GPU is the next logical step.

With the release of new generation of general purpose GPUs (GPGPUs) and Computer Unified Device architecture (CUDA) extension to the C Programming language, current single-threaded algorithms can be converted into parallel GPU programs which results in immense expansion of their possible applications area.

2. Previous Work

Over the past few decades, the inherent parallelism of optics has prompted a great deal of research on optical information processing. Among all the possible applications, a very promising direction is optical encryption [2]. However, directly recorded digital holograms do not constitute a secure tool for encryption. Digital holography with some modifications is a proper means for encrypting 2-D or 3-D optical information. With their pioneering work on double random phase encoding, Refregier and Javidi [3] paved the way for many following proposals of optical security and encryption system. Double random phase encoding (DRPE), broadly adopted method for optical encryption, uses two independent random phase masks in the (i) input plane and (ii) Fourier plane to convert a primary image into stationary white noise data.

3. Parallelization of DRPE Algorithm

In DRPE a reference wave with known amplitude and phase is modulated. A random key equal to the size of input image ranging from 0-1 will be used to produce stationary noise in the input image. The input image will be multiplied with a phase mask and then resulted image will be convolved with an impulse response of phase only transfers function. In the whole procedure of DRPE there are many processes which can be executed in the parallel fashion and resulted in the decrease of computational time. Out of these Fast Fourier

transform, performing convolution between the input image and an impulse response of phase only transfers function, gives very attracting results when executed parallel on GPU using CUFFT [4] library.

4. GPU Implementation

Implementing the FFT on the graphics card is relatively straight forward. The FFT is inherently an image-based algorithm [5]. The input image is read in the C code on Visual studio and its pixels are extracted. In FFT the lower bottom rows are the real values and the upper rows will be the imaginary values. The real values in the bottom rows are coupled with the imaginary values in the upper rows. The fragment programs used for real values and imaginary values are similar, but differ in their indexing and the value computed. We therefore load a separate program for the real and imaginary values. By grouping all the real values together and likewise all the imaginary values rather than interlace them. After this interlacing an appropriate fragment program will be run on all the entries by rendering only two quadrilateral primitives. Each fragment program performs single iteration of the FFT algorithm. Next, Fourier transform will be performed on each column. At this point, there are only two columns that contain real values. Performing the FFT on these two columns in the same way as performed on all the rows earlier. The rest of the entries in the array contain one part of a true complex number. The inverse FFT is performed by reversing the order of operations of the forward FFT. When the output vector is divided into even and odd sums, the location of the output vector elements can get scrambled in memory. Fortunately, it is usually in such a way that you can reverse bits that are in adjacent locations in memory and work your way back to ordered output. The Convolution process carried out on GPU using CUFFT library is found to be 22 × faster than MATLAB code and more faster if the size of the input matrix is in order of radix-2.

5. Platform and Language

We implemented the FFT algorithm on NVIDIA GeForce-310 graphics card. This graphics card features fully programmable vertex and fragment units, 32 bit floating point frame buffers and textures, and full 32 bit floating point enabled throughout the entire pipeline. Total amount of shared memory per block of graphics card is 16KB and global memory is 512MB. The software used is Visual studio 2010 with CUDA C as programming language for interfacing with GPU and C/C++ for executing CPU codes.

6. Conclusion and Future Work

We designed a simple system to increase the performance of the double random phase encoding by decreasing computational time usually consumes for executing the FFT. The process can be extended using Fractional Fourier transform or window Fourier transforms for better results.

References

[1] S. J. Orfandis, Introduction to Signal Processing. Upper Saddle River, New Jersey: Prentice Hall, 1996.

[2] B. Javidi, "Optical Information Processing for Encryption and Security Systems," Optics & Photonics News 8(3), 28, 1997.

[3] P. Refregier and B. Javidi, "Optical image encryption based on input plane and Fourier plane random encoding," Opt. Lett. 20, 767-769, 1995.

[4] CUDA CUFFT Library, NVIDIA Corp., 2007.

[5]J. W. Cooley, P. A. W. Lewis, and P. D. Welch, "Application of the fast fourier transform to computation of fourier integrals, fourier series, and convolution integrals," IEEE Transactions on Audio and Electroacoustics, AU 15(2), 79-84, 1967.