Application of Photon Transport Monte Carlo Module with GPU-based Parallel System

Chang Je Park[a]*, Heejeong Shon[b], Donghak Lee[c]
[a]*Sejong University, Nuclear Engineering Department, 209 Neungdong-ro, Gwangjin-gu, Seoul 143-747, Korea*
[b]*Golden Eng. Co. LTD, 501 Moklime B/D 30 Guuigangbyeon-ro, Gwangjin-gu, Seoul 143-747, Korea*
[c]*CoCo Link Inc., 1596-6 Bongcheon-dong, Kwanak-gu, Seoul, 305-353, Korea*
**Corresponding author: parkcj@sejong.ac.kr*

1. Introduction

A simple photon transport Monte Carlo module has been developed in order to estimate an effective dose rate and to specify proper shielding thickness connecting with a detection system. In general, it takes lots of computing time to get reliable results in Monte Carlo simulations especially in deep penetration problems with a thick shielding medium. To mitigate such a weakness of Monte Carlo methods, lots of variance reduction algorithms are proposed including geometry splitting and Russian roulette, weight windows, exponential transform, and forced collision, etc[1]. Simultaneously, advanced computing hardware systems such as GPU(Graphics Processing Units)-based parallel machines are used to get a better performance of the Monte Carlo simulation.[2][3] The GPU is much easier to access and to manage when comparing a CPU cluster system. It also becomes less expensive these days due to enhanced computer technology. There, lots of engineering areas adapt GPU-bases massive parallel computation technique.

In this paper, a photon transport Monte Carlo module which is rewritten in C language, is changed into CUDA (Compute Unified Device Architecture) platform.[4] And several parallel performance tests are carried out and provide analysis results, too.

2. A Simplified Photon Transport Monte Carlo Module

To implement photon transport simply, coherent (or Thomson) scattering is ignored and electron behavior is not taken into consideration. Thus, three main events are considered such as photoelectric effect, pair production, and Compton (or incoherent) scattering. The photoelectric effect is considered as a pure absorption of photon. And pair production is also an absorption case but the heat deposition happens instead as E-1.022 MeV. Scattering controls direction and energy of photon. Fig.1 shows the simple algorithm of photon transport Monte Carlo module. XCOM library is used and 92 elements and 182 mixtures can be used. Text-based interaction coefficients are produced to utilize further. The interaction coefficients of lead based on XCOM data are depicted in Fig. 2.

After collision, the flight distance (l) is obtained the following equation

$$l = (-\ln \xi) \times \lambda,$$

where ξ is a random number, $\lambda = 1/(\rho \times \mu_t)$ is the mean flight distance (cm), ρ is a medium density (g/cm^3), and μ_t is total attenuation coefficient (cm^2/g). When a photon passes through medium boundary with a given flight distance, it restarts from the boundary due to different material properties. After collision, the direction and energy is obtained from the Klein-Nishina formula.[5] For the test problem with GPU performance, two region sphere with concrete and lead is used. The photon source is assumed to be positioned at the center of sphere and its energy is 1 MeV. The problem depicts in Fig. 3 and the relative dose rates are provided in Fig. 4. The results are compared with those of MCNP, which provide accurate results.

3. Application with GPU System

The GPU system specification is provided in Table I. In order to compare fairly, CPU based photon transport Monte Carlo code is migrated into GPU based system with CUDA code without any optimization. 1 GPU processor is used in this performance test. The results are tabulated in Table II including latency, memory allocation and upload, kernel execution. The performance of GPU system approaches almost 30, which is expected to be 150 times when GPU is loaded up to the possible maximum of 18. And optimization is done for this system, the performance will be increased around 200. For the random number generator, the following typical 48 bit congruential method is used.

```
#define RAND() xrand(rnd)
__host__ __device__ long xrand(long rnd)
{
        long m48=0x7fffffffffff;
        return(((long)25214903917*rnd+11)&(long)m
48);
}
```

4. Conclusions

This study is focused on the performance of GPU-based photon transport Monte Carlo method. It provides almost 30 times speedup without any optimization and it is expected almost 200 times with fully supported GPU system. It is expected that GPU system with advanced parallelization algorithm will contribute successfully for development of the Monte Carlo module which requires quick and accurate simulations.

And the GPU system will be developed with a new technology according to the corresponding hardware development. However it still remains some difficulties to access GPU system due to unfamiliar softwares and architectures. However, its potential is very high and promising to apply heavy burdened Monte Carlo application fields.

.

Table I. GPU System Specification

Processor	Intel Xeon E5620(quod-core, 2.4GHz, 12MB L3 Cache) x 2
Memory	128 GB, DDR3
GPU	Nvidia Geforce GTX TITAN(GK-110, 2,688 core)), 837MHz, GDDR5(6000MHz), 6GB, 384bit
OS	CentOS 6.4
Compiler	
CPU	C(optimized) on gcc 4.4.7
GPU	CUDA(not optimized) on CUDA 6.5

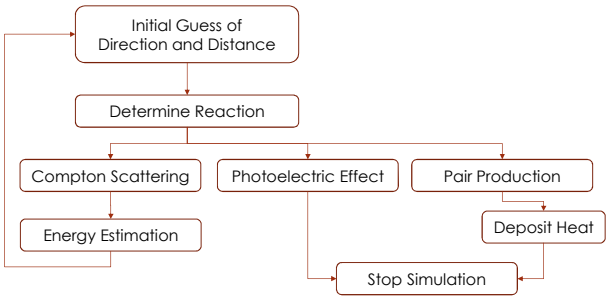

Figure 1. Simplified Calculation Procedure of the Photon Transport Monte Carlo Module

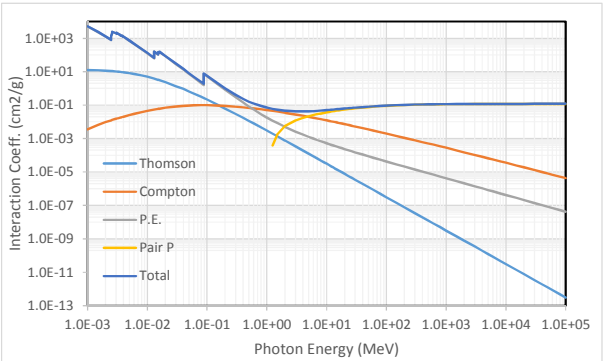

Figure 2. XCOM data of Pb (http://atom.kaeri.re.kr)

Table II. GPU Test Performance of Photon Monte Carlo Transport for the Two Region Problem

Loop Thread	CPU (sec)	GPU (sec)					
		Init Latency	Memory Allocate	Memory Upload	Kernel Execute	Total	FOM
1E6	9.51e-1	3.19e-04	3.19e-04	6.44e-04	3.09e-02	3.19e-02	29.7
1E7	9.50e-0	3.58e-04	3.58e-04	6.26e-04	3.00e-01	3.01e-01	31.5
1E8	9.48e+1	4.99e-04	4.99e-04	1.88e-03	3.00e+00	3.00e-00	31.5
1E9	9.45e+2	4.16e-04	4.16e-04	6.60e-04	3.00e+01	3.00e+01	33.1
1E10	1.33e+3	3.98e-04	3.98e-04	6.63e-04	4.22e+01	4.22e+01	31.5

*FOM=CPU time / GPU time

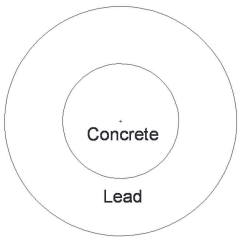

Figure 3. Two Region Test Problem for the Photon Monte Carlo Transport Code

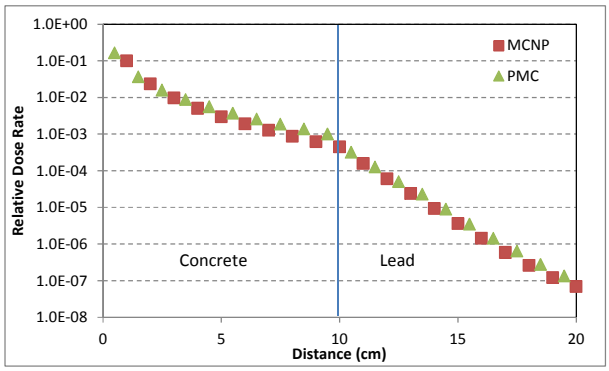

Figure 4. Relative Dose Rate for Two Region Test Problem for the Photon Monte Carlo Transport Code

REFERENCES

[1] J.K. Shultis and R.E. Faw, An MCNP Primer, Kansas State University, 2005.
[2] B. Toth and M. Magdics, "Monte Carlo Radiative Transport on the GPU", Fifth Hugarian Conference on Computer Graphics and Geometry, Gudapest, 2010.
[3] X. Gu, D. Choi, H. Pan, and A. Majumdar, "GPU-based ultra fast dose calculation using a finite size pencil beam model", Phys. Med. Biol., V.54, p.6287-6297, 2009.
[4] H. Lensch and R. Strzodka, Massive Parallel Computing with CUDA, 2008 (http://www.mpi-inf.mpg.de/~strzodka/lectures/ParCo08).
[5] L.L. Carter and E.D. Cashwell, Particle-Transport Simulation with the Monte Carlo Method, ERDA Critical Review Series, 1975.