RT^2: Ray-Tracing Accelerated Radiation Transport Monte Carlo Code

Chang-Min Lee and Sung-Joon Ye*
Department of Applied Bioengineering, Graduate School of Convergence Science and Technology, Seoul National University
**Corresponding author: sye@snu.ac.kr*

***Keywords: Monte Carlo, CUDA, Ray-Tracing, Nvidia OptiX**

1. Introduction

The Monte Carlo method has been recognized as a golden standard of radiation transport calculations due to its superior accuracy. However, the enormous computing time to reach a satisfactory precision of Monte Carlo limits the further applications. Since the advent of General-Purpose computing on Graphics Processing Units (GPGPU) technology, some GPU-based codes have been developed to accelerate time-consuming Monte Carlo calculations. Although those codes have limitations on input geometry and types of transport particles, several tens of computing acceleration compared to CPU Monte Carlo codes has been reported [1].

We have developed the Ray-Tracing accelerated Radiation Transport Monte Carlo code, RT^2, to cross over the limit of geometry and diversity of transport particles while maintaining the acceleration benefits. The comparisons with CPU Monte Carlo code, FLUKA [2], was performed to validate the performance and accuracy of RT^2 in the two cases representing solid and voxel structures.

2. Material and Methods

Since the late decades, Nvidia launched RTX series GPU cards. Nvidia OptiX, an application framework for ray-tracing (RT) based algorithm started a support of the RT hardware acceleration on RTX card [3]. RT^2 has been designed to deal with both of polygon mesh-based and voxel-based geometries by using OptiX framework. RT^2 resolved thread divergence caused by particle types and interaction branches by separating these calculations into individual kernels. By this approach, calculation and memory access pattern are vectorized as much as possible.

RT^2 can simulate neutron, photon, electron and positron simultaneously. Electron and positron transport kernels were developed by using PRESTA condensed-history algorithm of EGSnrc [4]. Neutron transport kernel was developed by in-house group-wised transport algorithm. Neutron kernel used ENDF.VII/0 cross section data that was preprocessed by NJOY21 code system [5].

Two geometries representing solid and voxel structure are tested, 1) ICRP reference phantom-based brain cancer boron neutron capture therapy (BNCT) model [6], 2) X-ray LINAC. Visualization results of these geometries are presented in Fig. 1.

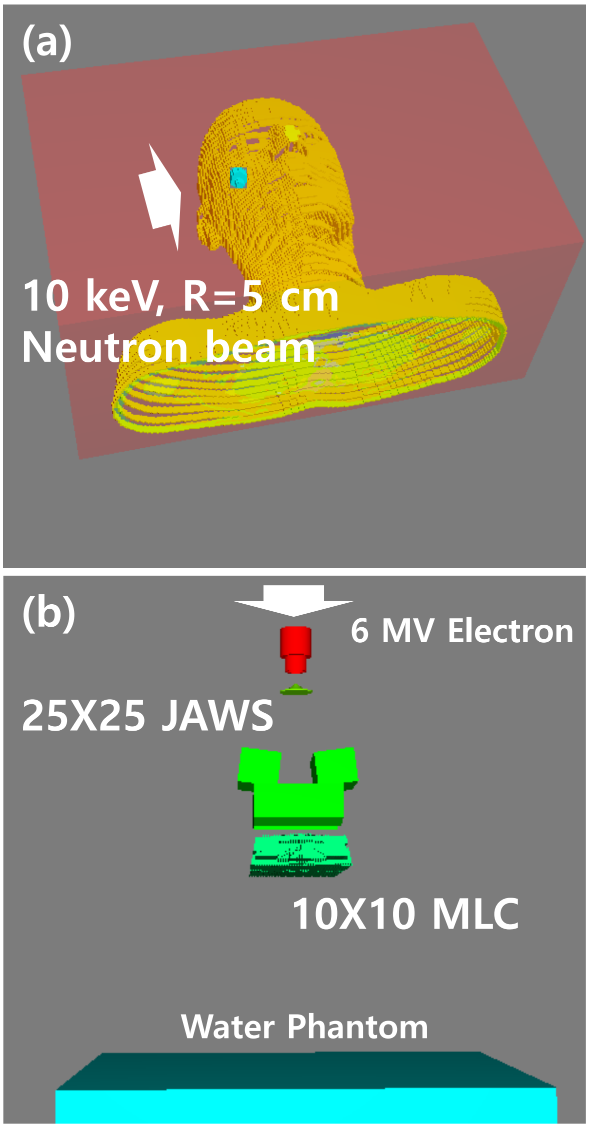

Fig. 1. (a) ICRP Reference phantom based brain cancer BNCT model with arbitrary boron concentrated area (cyan), (b) 6 MV mode X-ray LINAC geometry

LINAC calculation was divided into two parts. First stage set phase-space surface on the isocenter plane and simulate only the accelerator geometry. This phase space data is used as a source of second stage, the water phantom calculation.

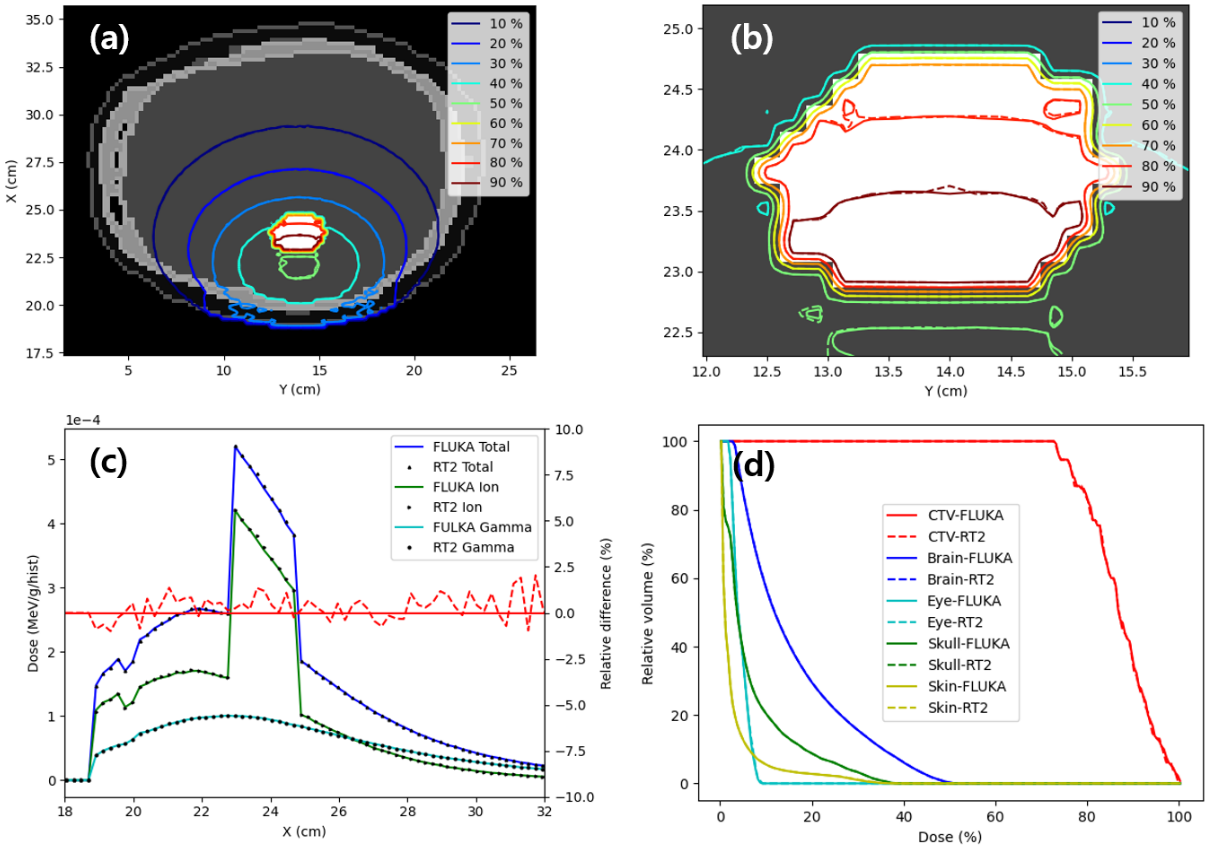

Fig. 2. BNCT dose comparison between FLUKA and RT^2. (a) Isodose curve of FLUKA (solid line) and RT^2 (dashed line), (b) Isodose curve of the area around the tumor, (c) Depth dose profile at the beam-axis. Dashed red line is relative difference of the total dose based on FLUKA result, (d) Dose volume histogram

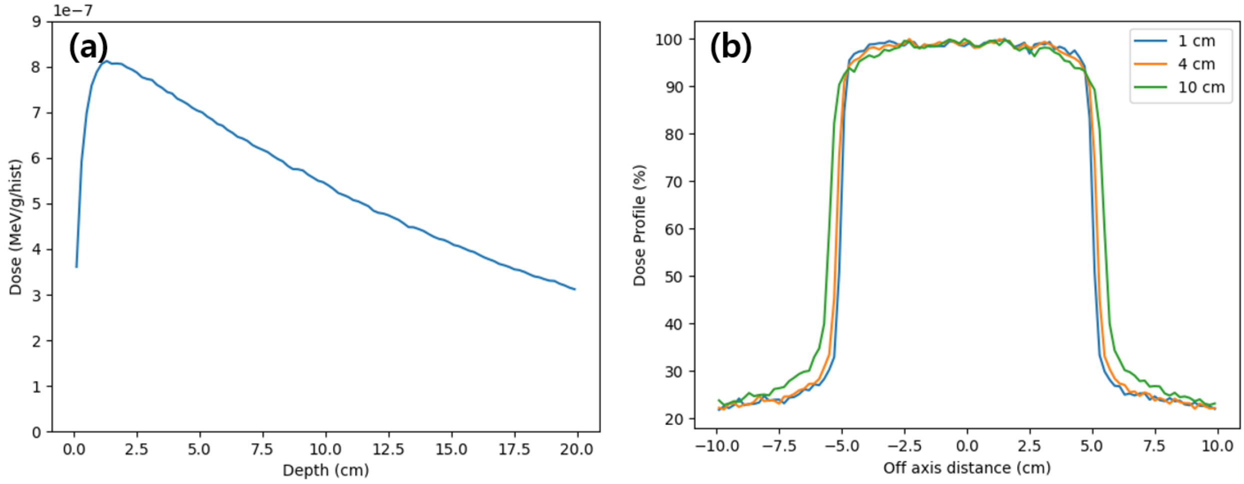

Fig. 3. 6 MV mode X-ray LINAC geometry simulation result of RT^2. (a) Depth dose profile with unit of MeV/g/hist, (b) Lateral dose profile at the depth of 1 cm, 4 cm, 10 cm from isoplane, normalized

To compare the performance and accuracy, inputs whose geometry and cutoff settings are identical was written in both of CPU Monte Carlo (FLUKA) and GPU Monte Carlo (RT^2). The FLUKA code was computed on Intel Xeon E5-2650 v4 and RT^2 code was computed on Nvidia RTX 4090.

3. Results

In CPU case, entire 24 threads are used. 1.2×10^{10} histories were simulated both of CPU and GPU codes in case of Brain cancer model. Fig. 2 illustrates the calculation results and difference between FLUKA and

RT^2. As can be seen, result of RT^2 code calculation results are fit well with FLUKA calculation results. The total computing time of FLUKA code was 113 450 seconds while RT^2 code was 1 020 seconds. Therefore, the RT^2 code was 111.2 times faster than FLUKA code in this case.

The calculation of 6 MV X-ray LINAC structure was finished in RT^2 but is in progress for FLUKA. Fig. 3. shows final RT^2 calculation results. In RT^2, the computing time of the first stage was 1 502 seconds for 5×10^9 histories and 390 seconds for 1×10^{10} histories in case of second stage. Figure 3. shows the depth-dose and lateral dose profile of RT^2 simulation.

4. Conclusions

We have developed RT^2 GPU Monte Carlo code which can simulate various types of radiations in both voxel and solid geometries. To validate the performance and accuracy of this code, we calculated a voxel-based BNCT model and 6 MV X-ray LINAC structure and compare the computing times and dose tallies of FLUKA and RT^2. In the BNCT structure, RT^2 accelerated the computing time more than 100 times than FLUKA while 98% of the voxels had a relative difference within ±2% for all voxels whose recorded doses exceeded 10% of the maximum dose. The LINAC structure is finished for RT^2 but now in progress for FLUKA. Therefore, the accuracy and time comparison with FLUKA result is not available now. We expected to be a similar improving factor and accuracy in LINAC geometry as a voxel geometry.

REFERENCES

[1] Jia, Xun, et al. "GPU-based fast Monte Carlo dose calculation for proton therapy." Physics in Medicine & Biology 57.23 (2012): 7783.

[2] Battistoni, Giuseppe, et al. "Overview of the FLUKA code." Annals of Nuclear Energy 82 (2015): 10-18.

[3] Parker, Steven G., et al. "Optix: a general purpose ray tracing engine." Acm transactions on graphics (tog) 29.4 (2010): 1-13.

[4] Kawrakow I, Rogers DWO, Mainegra-Hing E, Tessier F, Townson RW, Walters BRB. EGSnrc toolkit for Monte Carlo simulation of ionizing radiation transport, doi:10.4224 /40001303 (2000).

[5] Conlin, Jeremy Lloyd, et al. "NJOY21: Next generation nuclear data processing capabilities." EPJ Web of Conferences. Vol. 146. EDP Sciences, 2017.

[6] ICRP, 2009. Adult Reference Computational Phantoms. ICRP Publication 110. Ann. ICRP 39 (2).