Physics in Medicine & Biology

PAPER

A GPU-accelerated Monte Carlo code, RT^2 for coupled transport of photon, electron/positron, and neutron

To cite this article: Chang-Min Lee and Sung-Joon Ye 2024 *Phys. Med. Biol.* **69** 175005

IOP Publishing Phys. Med. Biol. 69 (2024) 175005 https://doi.org/10.1088/1361-6560/ad694f

Physics in Medicine & Biology

IPEM Institute of Physics and Engineering in Medicine

RECEIVED
18 April 2024

REVISED
9 July 2024

ACCEPTED FOR PUBLICATION
30 July 2024

PUBLISHED
14 August 2024

PAPER

A GPU-accelerated Monte Carlo code, RT^2 for coupled transport of photon, electron/positron, and neutron

Chang-Min Lee[1] and **Sung-Joon Ye**[1,2,3,*]

[1] Department of Applied Bioengineering and Research Institute for Convergence Science, Graduate School of Convergence Science and Technology, Seoul National University, Seoul 08826, Republic of Korea
[2] Advanced Institute of Convergence Technology, Seoul National University, Suwon 16229, Republic of Korea
[3] T-ROH Inc., Seoul 08812, Republic of Korea
[*] Author to whom any correspondence should be addressed.

E-mail: sye@snu.ac.kr

Keywords: GPU-acceleration, Monte Carlo, radiation, branch divergence, ray tracing

Abstract

Objective. This work aims to develop a graphics processing unit (GPU)-accelerated Monte Carlo code for the coupled transport of photon, electron/positron and neutron over a broad range of energies for medical applications. *Approach.* By separating the MC evolution of radiation into source, transport, and interaction kernels, the branch divergence was alleviated. The memory coalescence was achieved by vectorizing the access pattern in which the secondary particles were archived. To accelerate further particle tracking, ray-tracing hardware acceleration in the Nvidia OptiX^{TM} framework was applied. For photon and electron/positron, the EGSnrc interaction modules were ported as a GPU-optimized configuration. For neutron, a group-wised transport based on NJOY21 preprocessed data was implemented. The developed code was validated against CPU-based FLUKA. Neutron, x-ray and electron beams incident on water and ICRP phantoms were simulated. The neutron energy group and the transport parameters of photon and electron were set to be the same in both codes. A single Nvidia RTX 4090 card was used in this code while all 20 threads of a single Intel Core i9-10900K node were used in FLUKA. *Main results.* The number of histories was set to ensure that statistical uncertainties lower than 2% for all voxels whose doses were larger than 20% of the maximum. In all cases, the dose differences in the voxels between the codes were within 2.5%. For photons and electrons, the developed code was 150–300 times faster than FLUKA in both geometries. For neutrons, the code was respectively 80 and 135 times faster in the water and ICRP phantoms than FLUKA. *Significance.* This study offers an appropriate solution for uncoalesced memory access and branch divergence commonly encountered in coupled MC transport on the GPU architecture. The formidable acceleration in computing times and accuracy shown in this study can promise a routine clinical use of MC simulations.

1. Introduction

The calculation of energy deposition by radiation is a major concern of radiation physics and dosimetry. Algorithms that can simulate this problem are divided into two categories: deterministic and Monte Carlo methods. As the Monte Carlo method describes the whole problem as a combination of random events based on well-defined probability distributions and measured data, it can solve the problem in any complicated geometry with high accuracy. General-purpose Monte Carlo codes, such as EGSnrc, MCNP, GEANT4, FLUKA, and PHITS have been developed and successfully used in many fields (Agostinelli *et al* 2003, Battistoni *et al* 2015, Sato *et al* 2015, Werner *et al* 2018, National Research Council of Canada 2021). However, its application for radiation treatment planning system (TPS) have not been fully developed yet. As the Monte Carlo method requires a sufficient number of histories to gain statistically meaningful results, its computing time becomes much longer than that of the deterministic method. Therefore, despite its low

accuracy, the dose-calculation engine of TPS typically relies on a deterministic method to satisfy the computing performance required in routine clinical applications (Bedford 2002).

Recently, general-purpose computing on graphics processing units (GPGPU) technology has been advanced to accelerate computer simulations, numerical analysis, and machine learning. In problems suitable for parallelization, GPUs show theoretically better performance per cost than CPUs (Navarro *et al* 2014). To takes advantage of GPUs superior performance and cost-effectiveness, many attempts have been expended to develop GPU-based Monte Carlo codes. These include CUDA EGS, MCGPU, Shift, and gDPM (Badal and Badano 2009, Jia *et al* 2011, Lippuner and Elbakri 2011, Hamilton and Evans 2019). However, owing to the characteristics of the GPU architecture that will be described subsequently, these codes have restrictions associated with the simulations of types of transport particles and the secondaries. To date GPU codes capable of simultaneously simulating neutron, photon, and electron have not been reported yet.

In one of our previous work, we reported a GPU-accelerated Monte Carlo code for neutron transport only for boron neutron capture therapy (BNCT) (Lee and Lee 2022). In subsequent research, we attempted to extend this using a naive approach, but it was found that there was a significant performance degradation. When various types of particles are involved in the Monte Carlo transport, the probability of each thread performing different operations is also increased, thus resulting in the degradation of the entire performance. To mitigate this, a GPU algorithm was developed to separate the Monte Carlo evolution of radiation into source, transport and interaction kernels. The memory coalescence was achieved by vectorizing the access pattern in which the secondary particles were archived. In this study, we report a recently developed algorithm capable of being efficiently operated on the GPU architecture to transport simultaneously neutrons, photons, electrons, and positrons. The developed code was benchmarked by the CPU-based FLUKA code in terms of accuracy and computing efficiency for two different modes of coupled photons/electrons, and neutrons with the secondary gamma-rays.

2. Material and method

The physics and sampling algorithms of photon, electron and positron transport were ported from EGSnrc to develop GPU-accelerated applications using CUDA (Vingelmann and Fitzek 2020). These algorithms were modified and optimized to enhance the performance on the GPU architecture. The physics and sampling algorithms of neutron transport were similar to those used in FLUKA and MCNP. The ENDF/B-VII.0 neutron data library was pre-processed into group-wised cross-sections to achieve additional acceleration (Chadwick *et al* 2006).

Since 2018, GPU cards in Nvidia's RTX series have featured ray-tracing (RT) hardware acceleration. In these cards, the RT core, which specializes in tree traversal and ray-triangle intersection algorithms, replaces software emulation. This advancement resolved the major bottleneck of the RT algorithm, enabling us to render RT in real-time (Burgess 2020). Beyond rendering, several efforts have been expended to utilize this RT acceleration for scientific simulations. The feasibility of Monte Carlo acceleration for OpenMC code was reported by Salmon and McIntosh-Smith (2019). In this publication, RTX series cards (Turing architecture) demonstrated the RT acceleration speedup by 5–15, depending on the geometry. A higher speedup factor was observed in more complicated geometries. The Geant4-based GPU-accelerated optical photon simulation code, Opticks, was developed by Blyth *et al* (2021). They reported that GPU-accelerated Opticks for optical photon simulations was over 1000 times faster than Geant4 with a single CPU core. Motivated by these developments, the RT acceleration using the Nvidia OptiX^{TM} framework has been integrated into our GPU Monte Carlo code to enhance further the performance of particle transport (Parker *et al* 2010). Hereafter the code developed in this study is called 'Monte Carlo code for Ray-Tracing accelerated Radiation Transport (RT^2)'.

2.1. Monte Carlo algorithm optimization for the GPU architecture

The memory access is one of the key issues in optimizing GPGPU programs (Sanders and Kandrot 2010). In the Nvidia GPU, a group of 32 threads (so-called warp) launches, schedules and accesses memories simultaneously at the same clock cycle. When a warp accesses the global memory, the entire set of 32 threads requests a 32-byte sector, equivalent to eight 4-byte data. If the target memory addresses of each thread are arranged in a row and the length of the data structure is smaller than 4 bytes, this memory access ends successfully with four sectors, which is equivalent to 128 bytes. Otherwise, additional memory request for every unlined thread are required to fill out the target memory completely. This is called memory coalescing. As the Monte Carlo algorithm is based on repeated stochastic samplings, many factors can worsen memory coalescing. When threads of a warp request separate memory addresses, e.g. cross-sections of neutron–carbon and the neutron–hydrogen interactions, then uncoalesced memory access happens.

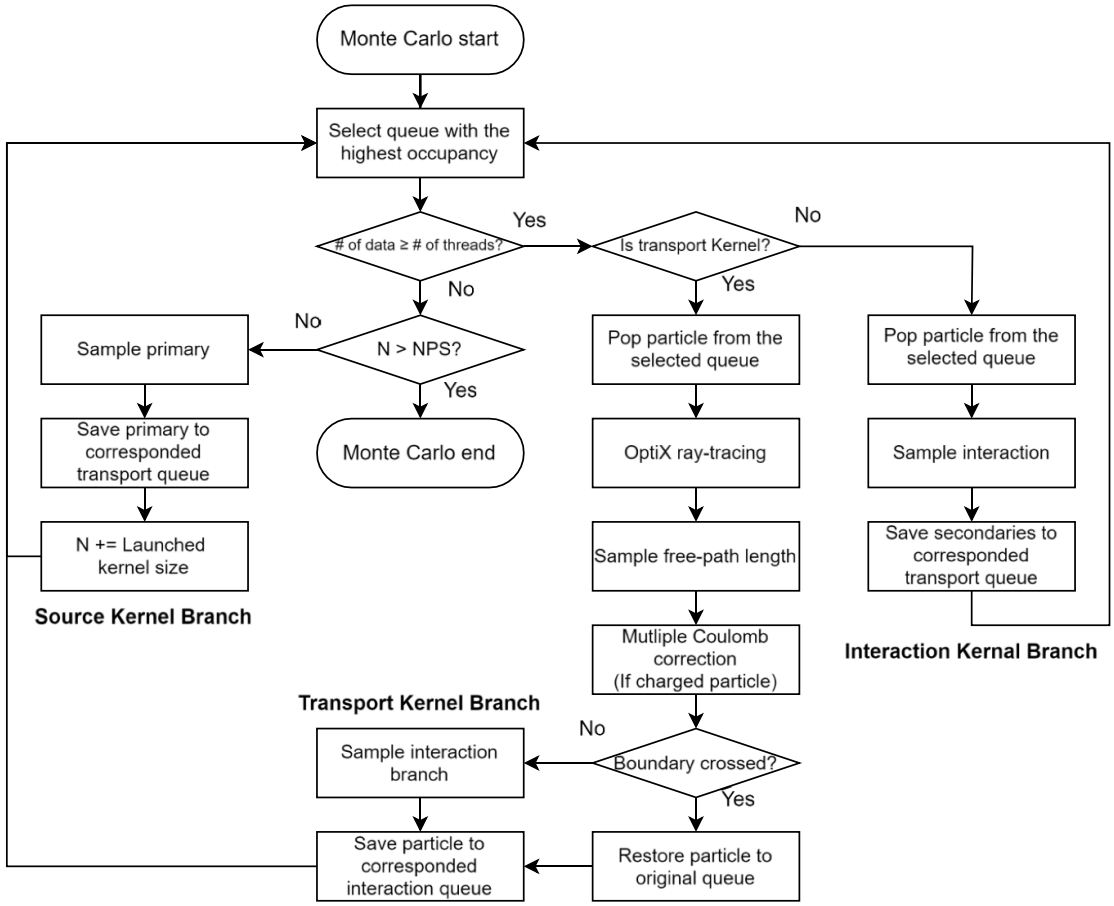

Figure 1. Algorithm workflow of this work (so called 'RT^2' code).
Note: N = current number of histories, NPS = total number of original source histories.

The branch divergence is another key factor that needs to be carefully considered in GPGPU programming. To maximize efficiency within limited resources, all 32 threads in a warp must execute a single instruction in the same clock cycle. If a warp encounters a conditional branch such as if-else statements of a code, then instructions are serialized. Threads that are not involved in the current condition wait until other threads complete conditional instructions. These stalled threads are called predicated-off threads. In the worst-case scenario, where all 32 threads follow their unique branches, the GPU efficiency can be degraded to 1/32. If there are branches where no threads are executing, those branches that are not executed by any thread are skipped, resulting in no effect aside from the overhead introduced by the condition check.

Similarly, as for the memory coalescence, Monte Carlo algorithm on the GPU architecture can cause a severe branch divergence. For instance, if a warp simulates the Compton scattering and pair production simultaneously, these calculations are then serialized. Rejection sampling also can cause branch divergence (Ridley and Forget 2021). For example, if some threads are rejected five times and other threads are rejected only once, the thread that passes first waits until all threads pass the rejection sampling process. The branch divergence and uncoalesced memory access often encountered in radiation transport are summarized as follows: diverged particle types, interaction branches, uncoalesced stack memory of secondary particles, rejection sampling, diverged cross-sections caused by different materials, and geometry divergence.

Figure 1 illustrates the overall algorithm and the data flow of RT^2. To customize the Monte Carlo algorithm optimally on the GPU architecture, we divided the entire Monte Carlo evolutions of radiation into three groups, i.e. source, transport and interaction. Every different type of particle and interaction has its own kernel and phase-space buffer. The phase space includes position, direction, kinetic energy, weight and the current cell index of individual particles. This algorithm is justified by assuming timely-independent events between the kernels as well as no interactions or interferences between the radiation particles during transport.

For memory efficiency, most of the data in RT^2 adopt the single-precision format. However, variables that required multiple accumulations or divisions involving very small divisors use double precision to

Table 1. The list of defined buffers and their types: the source buffer is activated when the phase-space source is defined. All transportable particles have their own phase-space buffers to archive secondary phase-space of interaction kernel.

Buffer ID	Name	Type
0	Source	Source sampling
1	Neutron	Transport
2	Proton	
3	Photon	
4	Electron	
5	Positron	
6	Rayleigh Scattering	Interaction
7	Compton Scattering	
8	Pair Production (Bethe–Heitler, over 85 MeV)	
9	Pair Production (NRC alias sampling, under 85 MeV)	
10	Electron Bremsstrahlung	
11	Moller Scattering	
12	Positron Bremsstrahlung	
13	Bhabha Scattering	
14	Positron-Electron Annihilation	

Note: Buffer ID = 2 (Proton) is under development.

minimize rounding errors. For instance, tally and variance data used double precision since they need millions of summations resulting each particle history.

The list of kernels developed and under development is outlined in table 1. At the beginning of a Monte Carlo cycle, the occupancies of each queue are searched and the queue with the highest occupancy is selected. If the number of recorded phase-space of the selected queue is lower than the total number of threads, this means that every buffer has not been occupied yet to be processed. In this case, the source kernel is selected. Otherwise, the type of selected queue is checked to determine whether it is either 'transport' or 'interaction'.

When dealing with a source defined by multiple types of particles, each warp samples a type of particle according to the given weights; it then selects only one type of particle at a time. For instance, if photon neutron beams have weights of 0.6 and 0.4, respectively, the warp generates 32 primary photons with a probability of 60% and 32 primary neutrons with a probability of 40%. This approach is adopted to vectorize the memory access pattern, as more than two transport kernel buffers are separated.

2.2. RT hardware acceleration for particle transport

In computer graphics, three data sets are required to construct a scene. The mesh data which is a set of facets presents the shape of individual objects. The texture data usually contains red, green, and blue colors or patterns of individual triangles. The last dataset is a shader, which contains a set of programs such as rays' reflection, refraction and smoothing. In radiation transport, an individual facet represents a boundary of two materials. Instead of color or pattern (required in rendering), material or cell information before and after the boundary crossing of radiation are required. These data are used to calculate a particle's free path and to search associated boundary-crossing tallies. Therefore, texture data were replaced with two cell indices.

As the RT algorithm is optimized for triangle meshes, any geometry (from a cubic phantom to computer-tomography (CT) based voxels) should be transformed into a set of triangle meshes. A cubic phantom can be translated as a set of 12 triangle facets and any CT-based geometry can be represented as a set of these cube meshes. As all solid geometries can be expressed or approximated with a triangle mesh, RT^2 is extensible to any arbitrary solid geometry.

The logical and memory flow between OptiX and Monte Carlo code is depicted in figure 2. During the preprocessing stage, a Monte Carlo engine converts CT data or cubic phantoms into triangle meshes. Each triangle contains two cell indices, one before and the other after the boundary crossing. Two shader programs, closest-hit and miss are initialized to handle boundary-crossing events. Additionally, the acceleration structure which includes mesh and a hierarchy tree for searching ray-hit positions is constructed in this preprocessing stage.

The OptiX kernel is invoked at the beginning of the particle transport. Phase-space data vectorized for each type of particles are loaded from the buffer memory of the transport kernel. Position and direction information are then given to initialize a ray. This ray traverses the scene recursively until it intersects a triangle. Upon intersection, the closest-hit program is executed, and the two cell indices are retrieved from the binding table. If the ray hits the back face of the triangle, these cell indices are swapped. When the miss

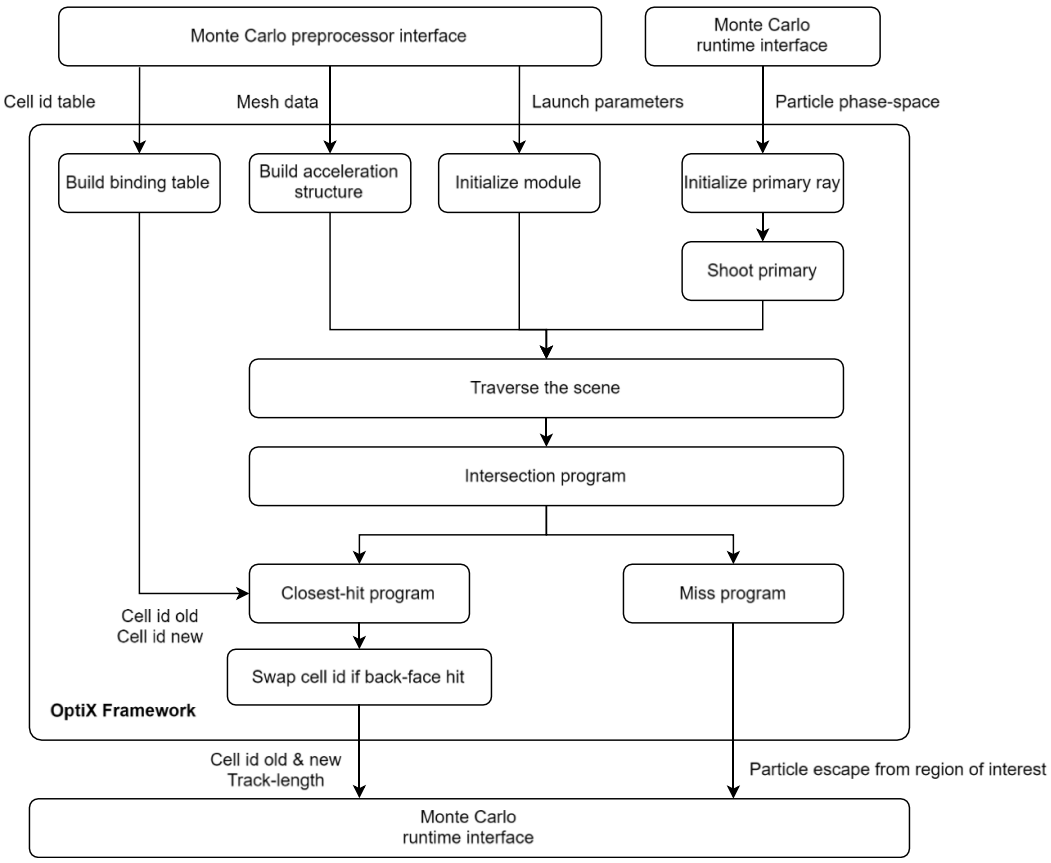

Figure 2. Logical flow of OptiX kernel for geometry tracking in RT[2].

program is invoked, the ray is outside the geometry and away from the region of interest. In this case, the particle discard flag turns on to send information back to the transport kernel.

Figure 3 shows the logical and memory flow of the transport kernel. In this branch, phase-space data are popped from one of the pre-defined transport buffers (see table 1) and loaded into the register during the initial stage. As all threads transport the same type of particles simultaneously, this memory access can be vectorized. Subsequently, the OptiX kernel is called as previously described and followed by the energy cutoff. Threads that escape from the region of interest or do not have enough energy are predicated off and remain disabled. The remaining threads sample the free-path length from the current cell index and the particle energy. In particular, for charged particles, multiple Coulomb scattering theory is applied. If particles cross the cell boundary, they are advanced to the ray-cell intersected positions, and the corresponding threads become predicated-off in the interaction branch sampling stage. Otherwise, each thread advances its particle by the free-path length and then samples its type of interaction. In the final stage, threads that crossed the region boundary put their particles back into the original transport buffer for repeated transport. Threads that undergo interactions place particles into the corresponding interaction buffers. As both boundary-crossing and interaction samplings are stochastic, phase-space data within a warp must be sorted before the final memory access operation, as shown in the last step of figure 3.

2.3. Interaction kernel

Figure 4 illustrates the evolution of the interaction kernel. In the interaction kernel branch, the final states of particles advanced by the transport kernel are popped from the buffer that is associated with the unique interaction branch. After the memory load, all threads generate secondaries from the corresponding interaction model. The final memory write operation can be vectorized as the number and types of secondaries are predetermined. For instance, the bremsstrahlung kernel pops 32 primary electrons from the bremsstrahlung interaction buffer and generates the 32 secondary electrons and photons from primary electrons, which then advance to their respective transport buffers in the global memory. In the subsequent cycle, the pair production kernel retrieves 32 photons from the pair production interaction buffer and generates 32 electrons and positrons from the primary photons, which then advance to their respective

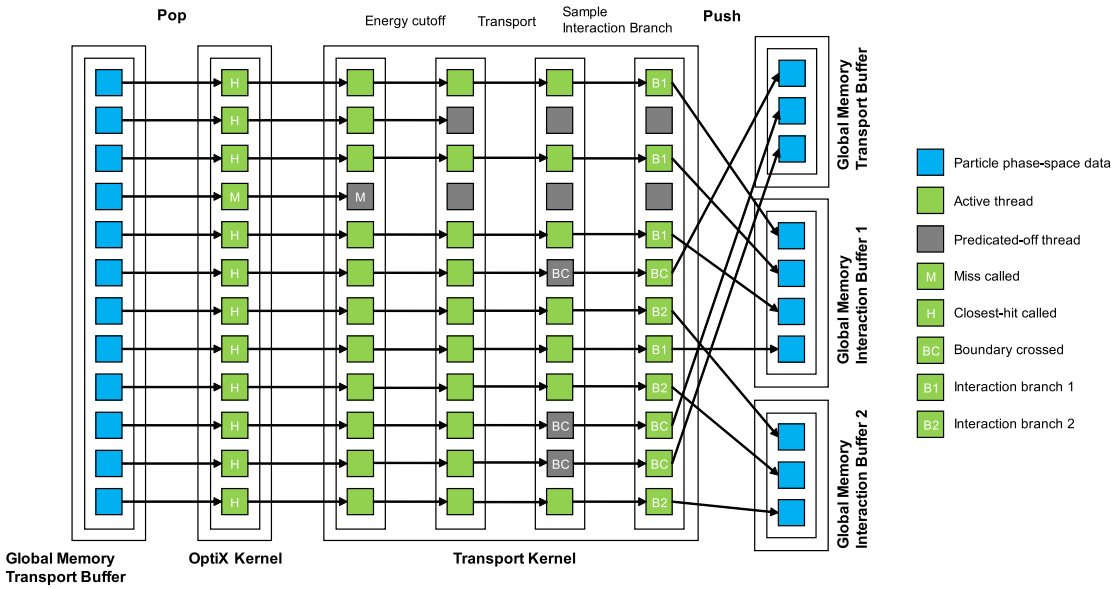

Figure 3. Logical and memory flow of transport kernel.

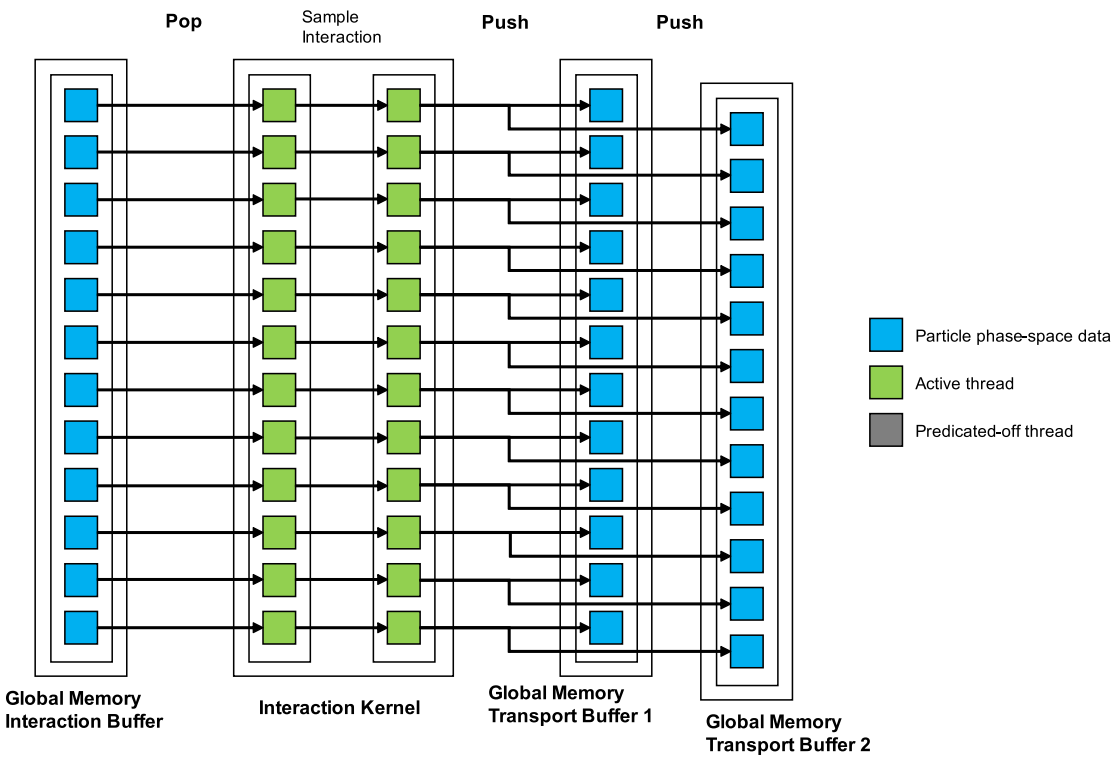

Figure 4. Logical and memory flow of interaction kernel.

transport buffers in the global memory. This approach ensures that all threads simulate only one type of particles or an interaction at a time.

2.4. Cross-section and physics

The detailed physics underlain in the Monte Carlo transport can be found in the aforementioned references. Herein, we only describe the necessary physics that was implemented to accelerate the sampling process and to alleviate the branch divergence and approximated in this code.

In photon and electron transport, electron binding energy was not considered in inelastic scatterings. Atomic relaxation was not implemented yet in this study. Triplet production and photonuclear reaction were not considered because the cross sections of these reactions are insignificant. As this study aims to simulate

dose distributions in human tissue and water, which are low-Z materials, these treatments can be justified. The detailed models will be implemented in the near future for precise simulations in high-Z materials, such as x-ray targets and shielding materials.

In neutron transport, a group-wised transport scheme was employed. Since this study aims to simulate neutron treatment doses mostly by thermal and epithermal neutrons in human tissue and water, potential artifacts in resonance peaks can be avoided. The point-wise transport option will be implemented for fast neutron simulations or neutron transport in metals.

2.4.1. Photon

In RT^2, photon interaction modules ported from EGSnrc were converted into CUDA programming. The EPDL97 cross-section data were pre-processed and transferred to GPU memory (Cullen *et al* 1997). The range of photon transport energy was 1 keV to 1 GeV.

2.4.1.1. Coherent (Rayleigh) scattering

In Rayleigh scattering, an alias sampling scheme of EGSnrc was fully employed in this work. Tabulated atomic form factors of Hubbell and Overbo's report was imported from the EGSnrc source code (Hubbell and Overbo 1979). The branch divergence may occur in Rayleigh scattering since the sampling algorithm contains rejection sampling.

2.4.1.2. Photoelectric effect

As the module for calculating the atomic relaxation has not been developed yet, secondary particles generated during atomic relaxation were not transported. Hence, their average energy was directly deposited where the reaction occurred. When the photoelectric effect occurs for photons with incident energy E_0, the probability of colliding with electrons in an atomic shell, s can be expressed as follows,

$$p_s\left(E_0\right) = \frac{\sigma_{pe,s}\left(E_0\right)}{\sum \sigma_{pe,i}\left(E_0\right)}. \tag{1}$$

The expected value of the locally deposited energy can be expressed as follows,

$$\bar{E}_{\text{depo}}\left(E_0\right) = \sum p_s\left(E_0\right) U_s, \tag{2}$$

where U_s is the potential energy of each atomic shell. Equation (2) is computed for all materials during the preprocessing stage. The kinetic energy of electrons generated in the photoelectric effect can be expressed as follows,

$$E_{\text{e}}\left(E_0\right) = E_0 - \bar{E}_{\text{depo}}\left(E_0\right). \tag{3}$$

The direction of electrons generated in the photoelectric effect inherits the direction of the incident photon without considering Sauter's distribution. Algorithms for electron direction sampling and atomic relaxation will be implemented in the future.

2.4.1.3. Incoherent (Compton) scattering

Assuming a free electron, the Klein–Nishina differential cross-section was employed. As mentioned earlier, the incoherent scattering function accounting for electron bound energy and Doppler broadening will be implemented later.

2.4.1.4. Pair production

The first-Born approximation formula presented by Motz, Olsen, and Koch was employed (Motz *et al* 1969). In the latest version of EGSnrc, the alias table based on the exact partial-wave-analysis by Øverbø, Mork, and Olsen can be available (Øverbø *et al* 1968). By using this method, the pair production below 86.4 MeV can be simulated without using equation-based rejection sampling. As the rejection method provokes a branch divergence, RT^2 employed this method for interactions under 85 MeV. As shown in table 1, the pair production buffer was divided into two parts. If the energy of primary photon is below 85 MeV, the phase-space data are stored in the NRC buffer. Otherwise, they are stored in the Bethe–Heitler buffer and processed separately. By using this method, we have eliminated the branch divergence in a low energy pair production caused by rejection sampling.

In secondary direction sampling, all three options of EGSnrc were implemented and were thus selectable. They include the original EGS4 algorithm (iprdst = 0), 2BS formula from Koch and Motz (iprdst = 2), and the leading term of the 2BS formula (iprdst = 1).

2.4.2. Electron/positron

In electron and positron transport, the PRESTA-II algorithm used in EGSnrc has been employed (Kawrakow 2000). Cross-section data were generated by theoretical formulas that are identical to the EGSnrc's PEGSLESS mode. They include bremsstrahlung, Moller/Bhabha scattering, and annihilation for incident energies from 1 keV to 1 GeV. The PRESTA-II, bremsstrahlung, inelastic scattering, and annihilation sampling algorithms ported from the EGSnrc mortran subroutines were converted into CUDA C++ and optimized for the GPU architecture.

2.4.2.1. Bremsstrahlung

EGSnrc provides three options for the bremsstrahlung sampling algorithm. In the EGS4 mode (ibr_nist = 0), the Bethe–Heitler cross-section with an empirical correction factor is used for energies below 50 MeV, and the extreme relativistic, Coulomb-corrected cross-section calculated from Koch–Motz's formula is used for higher energies. In the NIST mode (ibr_nist = 1), Tseng and Pratt's partial wave analysis calculations are used for energies below 2 MeV and the Coulomb-corrected extreme relativistic cross-section are used for above 50 MeV. For the intermediate range between them, the spline interpolation from the upper and lower energy regions is used. In the NIST mode (ibr_nist = 1) or NRC mode (ibr_nist = 2), a pre-calculated tabulated alias table is used for energy-angle sampling, while the first Born approximation is used for generating cross-sections. The Bethe–Heitler options exhibit energy-dependent branching and the rejection method is used in the sampling process. Therefore, the NIST and NRC modes were employed in RT^2 and alias tables were imported from the EGSnrc source code.

In photon direction sampling, all three options of EGSnrc were implemented. They included turning off the direction sampling, 2BS formula from Koch and Motz (ibrdst = 1) or from the leading term of the 2BS formula (ibrdst = 0). The bremsstrahlung splitting option for the variance reduction was also implemented.

2.4.2.2. Inelastic scattering

In inelastic scattering, a free electron in an atom is assumed by ignoring atomic binding energies. Both Moller and Bhabha scatterings rely on the rejection method so that a branch divergence may occur. The electron impact ionization feature will be implemented in the future.

2.4.2.3. Positron-electron annihilation

If the kinetic energy of the incident positron is small, two photons move in opposite directions and their distributions are isotropic. Otherwise, the directions of two photons are expressed in terms related to the direction of the incident positron. In the EGSnrc, in flight and at-rest annihilations are separately sampled for performance optimization. However, this branching reduces the performance on the GPU architecture. Therefore, an in flight sampling algorithm was applied even in at-rest situation.

2.4.2.4. Multiple elastic scattering

The differential cross-section of screened Rutherford elastic scattering is defined as follows,

$$\frac{\mathrm{d}\sigma_{\mathrm{SR}}}{\mathrm{d}\mu} = \frac{2\pi r_0^2 Z^2}{\beta^2 \tau (\tau+2)} \frac{R_{\mathrm{mott}}(Z,T,\mu)}{(1-\mu+2\eta)^2}, \tag{4}$$

where μ is the direction cosine, and η is the screening parameter from Moliere elastic scattering theory Moliere (1947). $R_{\mathrm{mott}}(Z,T,\mu)$ represents a spin correction which is derived by Mott (1929). Owing to the extremely large cross-section of charged particle elastic scattering, the condensed history method is commonly used to reduce computing times. According to the multiple elastic scattering theory, the energy fluence of the scattered electron, Φ_0 after it travels a track length s can be expressed as follows,

$$\Phi_0(\mu,\phi,E) = \frac{1}{2\pi}\sum_{l=0}^{\infty}\left(l+\frac{1}{2}\right)\exp(-G_l)P_l(\mu), \tag{5}$$

where E is the final energy after slowing down, μ is the cosine of the angle between the initial direction and scattered direction, ϕ is the azimuthal angle and $P_l(\mu)$ is the Legendre polynomial. G_l represents the Goudsmit–Saunderson moment, which is defined as follows (Goudsmit and Saunderson 1940),

$$G_l = \int_E^{E_0} \frac{\mathrm{d}E'}{L(E',E_{\gamma,c},T_c)}\kappa_l(E'), \tag{6}$$

where E_0 is the initial energy before slowing down, $L(E', E_{\gamma,c}, T_c)$ represents the restricted stopping power, and κ_l is the moment of the elastic scattering cross-section, which is expressed as follows,

$$\kappa_l(E) = 2\pi \int_{-1}^{1} d\mu \Sigma_{\mathrm{SR}}(\mu, E)\left[1 - P_l(\mu)\right]. \tag{7}$$

If a step size of multiple scattering is sufficiently short, E becomes significantly smaller than E_0; in this case, it can be postulated that there is no energy loss. In this situation, equation (5) can be expressed as follows (Kawrakow and Bielajew 1998),

$$2\pi\Phi_0(\mu, \phi, E) = \mathrm{e}^{-\lambda}\delta(1-\mu) + \lambda\mathrm{e}^{-\lambda}\frac{1}{\sigma_{\mathrm{SR}}}\frac{\mathrm{d}\sigma_{\mathrm{SR}}(\mu)}{\mathrm{d}\mu} + \left(1 - \mathrm{e}^{-\lambda} - \lambda\mathrm{e}^{-\lambda}\right)F_{\mathrm{SR}}^{(2+)}(\mu), \tag{8}$$

where δ is Dirac's delta function, λ is the expected number of elastic scattering, thus $\lambda = \Sigma_{\mathrm{SR}}s$. The first term on the right side represents the case where no scattering occurs, the second represents the case of a single scattering event, and the last term represents the case of two or more scattering events. $F_{\mathrm{SR}}^{(2+)}(\mu)$ represents the probability distribution function of the scattering angle of the multiple scattering events. This function is precomputed in three-dimensional space and represented as an alias table in EGSnrc code. We have imported this table to our code as this logical structure has a major advantage on the GPU architecture. The final directional cosine of equation (8) is followed by rejection sampling for $R_{\mathrm{mott}}(Z, T, \mu)$ in equation (4) to introduce the spin effect. The spin rejection table of $R_{\mathrm{mott}}(Z, T, \mu)$ was also imported from the EGSnrc source code.

2.4.3. Neutron

In neutron transport, a group-wised transport algorithm was developed. To accomplish this, the ENDF/B-VII.0 neutron cross-section data library was used (Chadwick *et al* 2006). The Doppler broadening, thermal neutron scattering and grouping were processed by the NJOY21 code system (Macfarlane *et al* 2017). The output of the NJOY21 code represented by the Legendre polynomial was translated to an alias table by an in-house preprocessing code. Any arbitrary energy structure can be defined by users. However, for comparison purpose, the same energy group structure from 10^{-5} eV to 20 MeV as in FLUKA was used in RT^2. Further, the equiprobable cosine bin method which is used in MCNP5, was applied to sample the angular distribution of secondaries (X-5 Monte Carlo Team 2003).

However, the multigroup scheme can cause artifacts, especially in thin-target and resonance problems. As the angular distribution is uniform in a single equiprobable bin, the band-shape angular distribution of secondaries can be observed in a single scattering situation (thin target). Averaged resonance peaks constitute the other possible artefact type. The sharp cross-section fluctuations, called resonance peaks, are averaged in a single energy group in the multigroup scheme. As a result, shelf-shielding effects can be underestimated. In FLUKA, the fine energy bins are set in the resonance regions for some important metals, such as Al, Fe, Cu, Au, Pb, and Bi to avoid this artifact (Battistoni *et al* 2015). Figure 5 illustrates pointwise and group-wised total cross-sections of ^{14}N from ENDF/B-VII.0. As shown, the resonance peaks are averaged and coarse structures are smoothed.

Types of secondaries in neutron reactions are neutron, proton, photon and heavy ions. In our code, secondary heavy ions and protons are treated as high-LET particles. Thus, they are assumed to deposit their kinetic energy locally (KERMA approximation). The secondary photon has an energy group structure that is distinct from the neutron's. In default, the LANL 48-group structure option of NJOY was adopted. The energy and angular sampling tables were also generated by an in-house code. Examples of secondary energy and angular distributions of neutron and gamma-ray are illustrated in figure 6. As shown in figure 6(a), tabulated data of the ENDF library were converted to a Legendre polynomial by the NJOY code system for the interpolation. With this NJOY conversion, the Legendre polynomial was converted to an equiprobable angular distribution for alias sampling. The group transition probability of each reaction was also calculated by NJOY and converted to an alias table. The example energy spectrum of secondary photon from the alias table is shown in figure 6(b).

In photon and electron interactions, the secondary sampling algorithms are different for each interaction. In contrast, the same table lookup algorithm was utilized in all types of neutron reactions. Therefore, neutron interaction kernels were not separately developed as shown in table 1 for photon and electron interactions. All neutron reactions defined in the ENDF library are individually computed. For instance, (n, n), (n, n′), and (n, 2n) reactions have a unique kinetic energy transition alias table and a equiprobable angle table for directional sampling. Reactions do not generate any secondary neutron or proton, such as (n, α) and (n, t) are merged and treated as a single table that contains an averaged value of

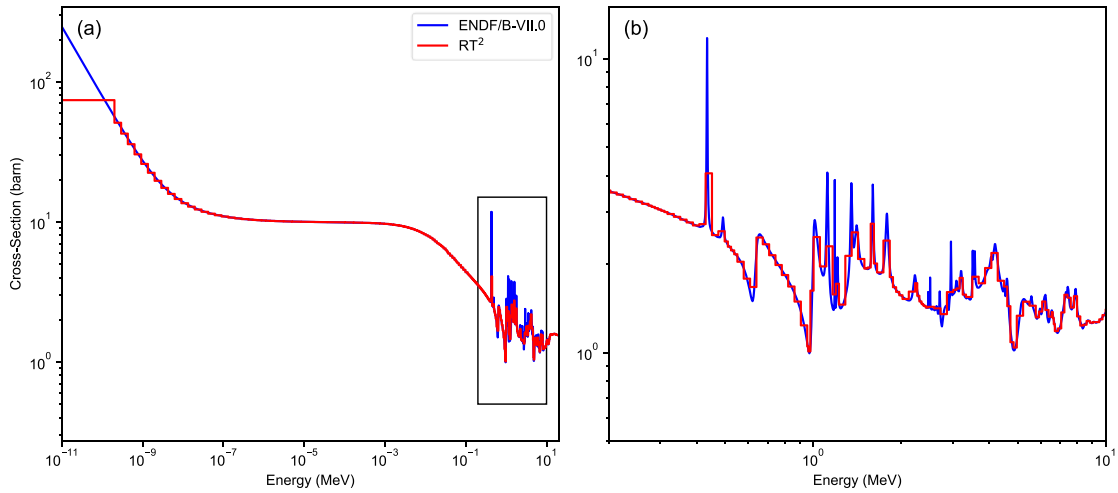

Figure 5. Total cross-section of neutron reaction with ^{14}N. The point-wised cross section of ENDF/B-VII.0 database (blue) and the group-wised cross section processed by NJOY21 (red) (a). Cross-section in the resonance region (b).

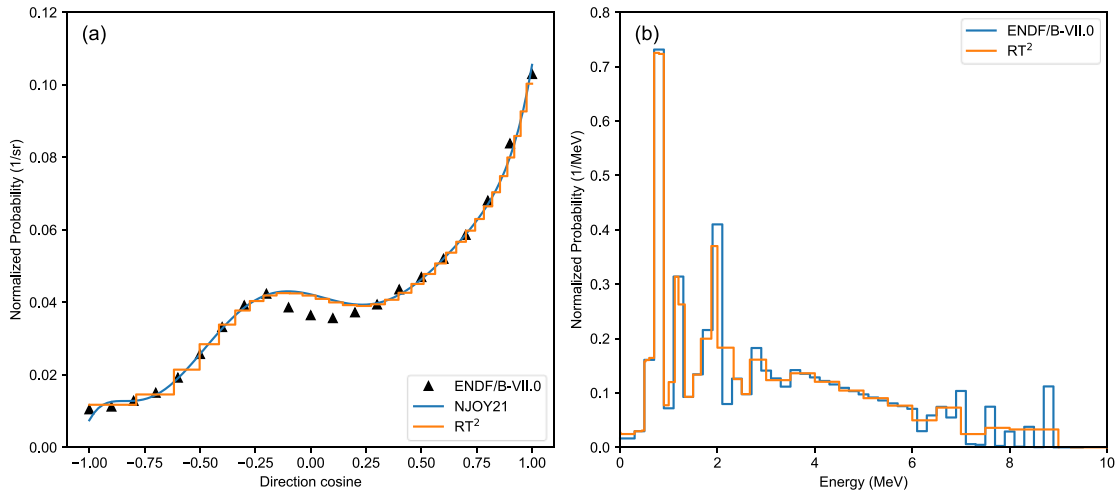

Figure 6. Example of pre-processed data of ^{9}Be(n,2n)^{8}Be reaction with 17 MeV primary neutron. The angular distribution of secondary neutron with 3 MeV energy in tabulated data from ENDF/B-VII.0 (symbol), group-wised data represented by Legendre polynomial (blue) and equiprobable table generated by RT^2 preprocessing module (orange) (a). Pre-processed photon spectrum data of ^{35}Cl(n,γ)^{36}Cl reaction with 1 MeV primary neutron (b).

energy deposition for memory optimization. The photon spectra and multiplicities of these reactions are used as averaged values. This treatment is justified by the KERMA approximation of heavy ion transport. Reactions that generate protons have been distinguished for the proton transport, which will be implemented later. The neutron reaction sampling procedure is as follows: (1) sampling target nucleus from compound alias table, (2) sampling reaction type from reaction alias table, (3) sampling energy of secondary from energy transition alias table, (4) sampling direction of secondary from equiprobable angular table. Herein, steps 3 and 4 were repeated for each secondary.

3. Benchmarking environment

As FLUKA and RT^2 operate on different architectures, a direct performance comparison on the same platform is not feasible. Therefore, we selected a CPU and a GPU released in proximate temporal intervals for a fair comparison. For the FLUKA calculation, the Intel i9-10900k (which was released in the second quarter of 2020) was used. For the calculation of RT^2, the Nvidia RTX 4090 (which entered the market in the third quarter of 2022) was employed. Both devices are commonly used for workstations or high-end personal computing purposes. All 20 available threads of the CPU were utilized for the FLUKA calculation, and a sufficient number of threads was allocated on the GPU to attain performance saturation for RT^2.

Benchmarking was performed in a water phantom with a mono-energetic electron, photon and neutron beams, as well as in voxel geometry, which was converted from the ICRP adult reference phantom with a 6 MV x-ray beam and a neutron beam from the accelerator (ICRP 2009). The neutron beam spectrum was obtained from the MCNP simulation outcomes of a proton accelerator with a beryllium target (Lee *et al* 2020). The same number of histories were simulated in both codes to obtain sufficient statistical uncertainty. The accuracy and performance were evaluated based on the following metrics,

$$\text{Dose Difference} = \frac{D_{RT^2} - D_{FLUKA}}{D^{MAX}_{FLUKA}} \times 100\,(\%) \tag{9}$$

$$\text{Acceleration Factor} = \frac{T_{FLUKA}}{T_{RT^2}}, \tag{10}$$

where D_{RT^2} is the computed dose of RT^2, D_{FLUKA} is the computed dose of FLUKA code and $DMAX_{FLUKA}$ is the maximum dose of FLUKA. T_{FLUKA} and T_{RT^2} represent the computing time of FLUKA and RT^2 respectively.

In both codes, electron and photon cutoffs were set to 100 keV and 10 keV, respectively. The electron spin correction was activated in both codes for enhanced simulation accuracy. The values of default and maximum electron step size were taken from the default EGSnrc settings. The atomic relaxation was inactivated in FLUKA given that RT^2 does not support this feature yet. A neutron energy structure comprising 260 groups, identical to that employed in FLUKA, was adopted to RT^2. The scoring algorithm of RT^2 followed the FLUKA USRBIN XYZ type detector, which is independent of the geometry. The numbers of histories for all benchmarking cases were sufficient, thus ensuring that the uncertainty was maintained below 2% for all voxels receiving at least 20% of the maximum dose.

4. Results and discussions

4.1. Water phantom

A rectangular parallel, 50 MeV mono-energetic electron beam with a size of 10×10 cm^2 was incident perpendicularly at the center of a water phantom with a volume of $1 \times 1 \times 1$ m^3. The scoring mesh grid was set to $1 \times 1 \times 1$ cm^3; this resulted in a grid composed of $100 \times 100 \times 100$ mesh cells. Both the FLUKA and RT^2 simulated 1×10^9 histories. Comparison results are illustrated in figure 7. The depth and lateral isodose curves and dose profiles are illustrated in parts (a), (b) and (c), (d) respectively. Colored solid lines denote FLUKA isodose, whereas dotted symbols denote the isodose curve of RT^2. The lateral dose distribution was obtained at a depth of 8 cm from the surface where the dose was maximized. The differences in delivered dose between FLUKA and RT^2 were under 1% for every point along the beam axis, except in intervals where the dose exhibited abrupt decreases. The lateral dose differences were also less than 1% for every point. The total energy deposition was 48.119 MeV/history for FLUKA and 48.171 MeV/history for RT^2.

Similarly, a 10 MeV mono-energetic electron beam, identical in shape to the previous case, was simulated. A water phantom with a volume of $20 \times 20 \times 20$ cm^3 was divided by a $2 \times 2 \times 2$ mm^3 scoring mesh grid, yielding $100 \times 100 \times 100$ mesh cells. Both codes simulated 5×10^9 histories. Figure 8 presents the calculation results of FLUKA and RT^2. The lateral dose distribution was analyzed at a depth of 3 cm from the surface. As in the previous case, the dose differences between FLUKA and RT^2 were under 1% for all points on the beam axis, except in positions where the dose decreased abruptly and on the surface. The total energy deposition was 9.729 MeV/history for FLUKA and 9.743 MeV/history for RT^2.

The 10 MeV mono-energetic photon beam with a size of 10×10 cm^2 was also simulated in a water phantom with dimensions of $1 \times 1 \times 1$ m^3. The dimensions and size of the scoring mesh grid were identical to the first case. In total, 5×10^9 histories were simulated in both codes. The comparison results are presented in figure 9. In contrast to the previous two cases, the photon calculation results yielded dose differences of less than 1% for every point along the beam axis and laterally. The total energy deposition of the FLUKA calculation was 8.228 MeV/history and RT^2 was 8.218 MeV/history.

For a scenario involving BNCT, 15 parts per million (ppm) of boron-10 was applied to a water phantom in neutron beam simulation. A 10 keV mono-energetic neutron beam with a circular shape cross-section of 10 cm in diameter was directed perpendicularly into a boronated water phantom of $20 \times 20 \times 20$ cm^3. The scoring mesh grid was set to $2 \times 2 \times 2$ mm^3, and 7.5×10^{10} histories were simulated in both codes. The dose was segregated into two components, namely gamma-ray dose and ion dose. The gamma-ray dose included contributions from the 477 keV gamma-ray from the $^{10}B(n,\alpha)^7Li$ reaction, the 2.22 MeV gamma-ray from the $^1H(n,\gamma)^2H$ reaction, and other radiative capture reactions. The ion dose included all the components except the gamma-ray dose. Owing to the absence of heavy ion transport in this study, secondary ions were considered as local energy deposits, a reasonable approximation given their minimal range in BNCT scenarios (Swanepoel 2010). The calculation results are shown in figure 10. Depth and lateral isodose curves

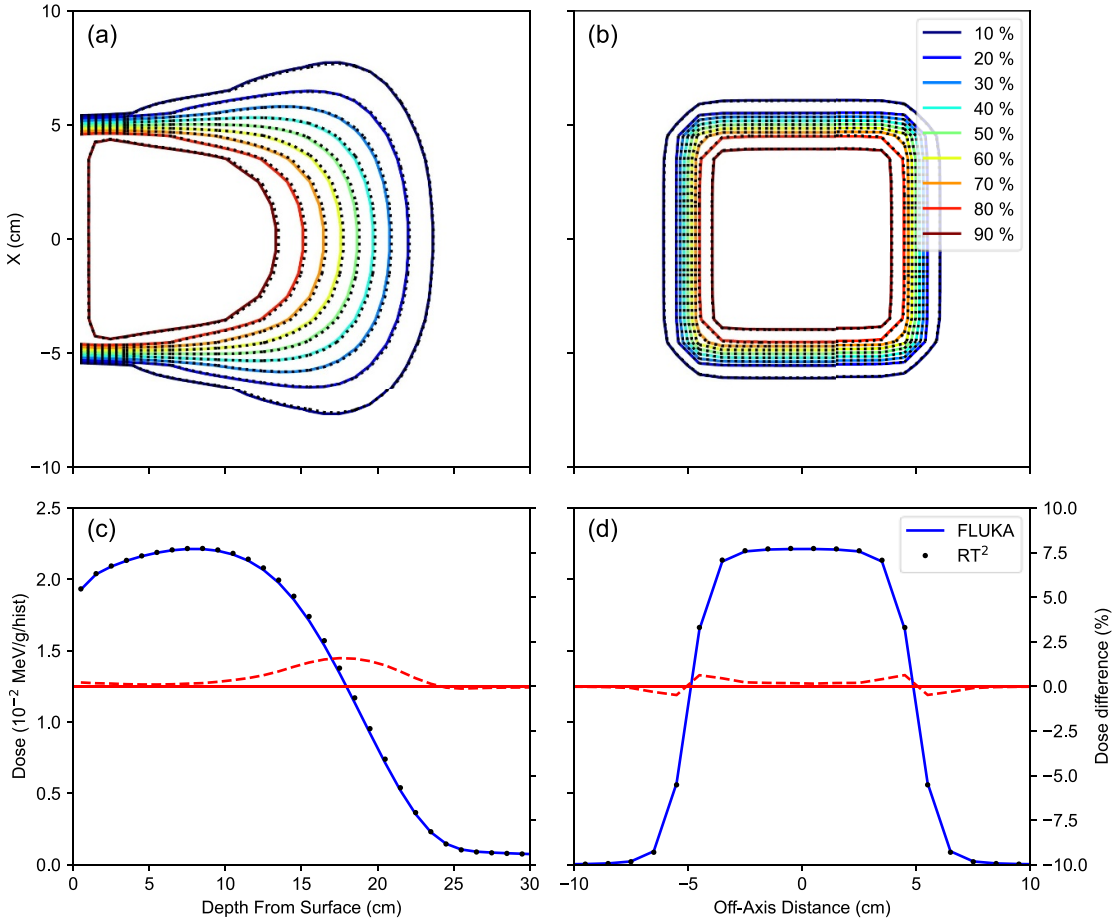

Figure 7. Dose calculation results for a 50 MeV electron beam in a water phantom. The isodose curve of FLUKA calculation results (colored solid lines) and RT^2 (black symbols) are shown in 2D image. Isodose curve in *XZ* plane that contain the beam axis (a), in *XY* plane at 3 cm from the surface (b). The depth-dose and lateral-dose profile are shown in (c) and (d) respectively. Dotted red lines present dose differences between the FLUKA and RT^2.

of the ion component are presented in figures 10(a), (b) and curves of the gamma-ray component are presented in figures 10(c), (d), respectively. These two components are normalized to their respective maximum values. The dose difference was calculated by a total dose of FLUKA and RT^2. The energy deposition of the ion component was 131.292 keV/history in the FLUKA and 132.107 keV/history in RT^2. The gamma-ray energy deposition was 165.480 keV/history and 164.618 keV/history, respectively.

4.2. ICRP reference phantom

The adult male model from the ICRP reference phantom was supplied as a voxel geometry to FLUKA and RT^2 respectively. The reference phantom was a cuboid of 254 × 127 × 222 voxels, and each voxel has a size of 2.137 × 2.137 × 8 mm^3. The material composition of each segment was specified in both codes based on the reference (ICRP 2009). Simulations were conducted for two distinct scenarios: x-ray beam incident on the stomach and a BNCT beam incident on the head.

For the photon beam case, a mesh-grid tally was set up from a height of 94.8 cm to 127.6 cm to cover the stomach region. The scoring mesh size was the same as the geometry voxel size. A rectangular, parallel photon beam with a size of 10 × 10 cm^2 was horizontally incident at the center of the stomach. The spectral photon source was defined consistently in both codes, which was obtained from the FLUKA calculations with 6 MV electron collision on a tungsten target with a thickness of 0.889 mm. The calculated results of both codes are shown in figure 11. The lateral dose distribution was obtained at 14 cm, which is the center of the human body phantom. Excluding the surface voxels, the dose difference was less than 1% for all voxels along the beam axis and laterally. Therefore, the photon simulation result of RT^2 showed a strong agreement with FLUKA even in heterogeneous geometry.

In the neutron case, an arbitrary tumor region was placed within the head of the reference phantom to model a BNCT case. Boron-10 concentrations of 45 ppm and 15 ppm were applied to the designated tumor and other tissue, respectively. Similar to the previous case, the scoring mesh size was set to be the same as the

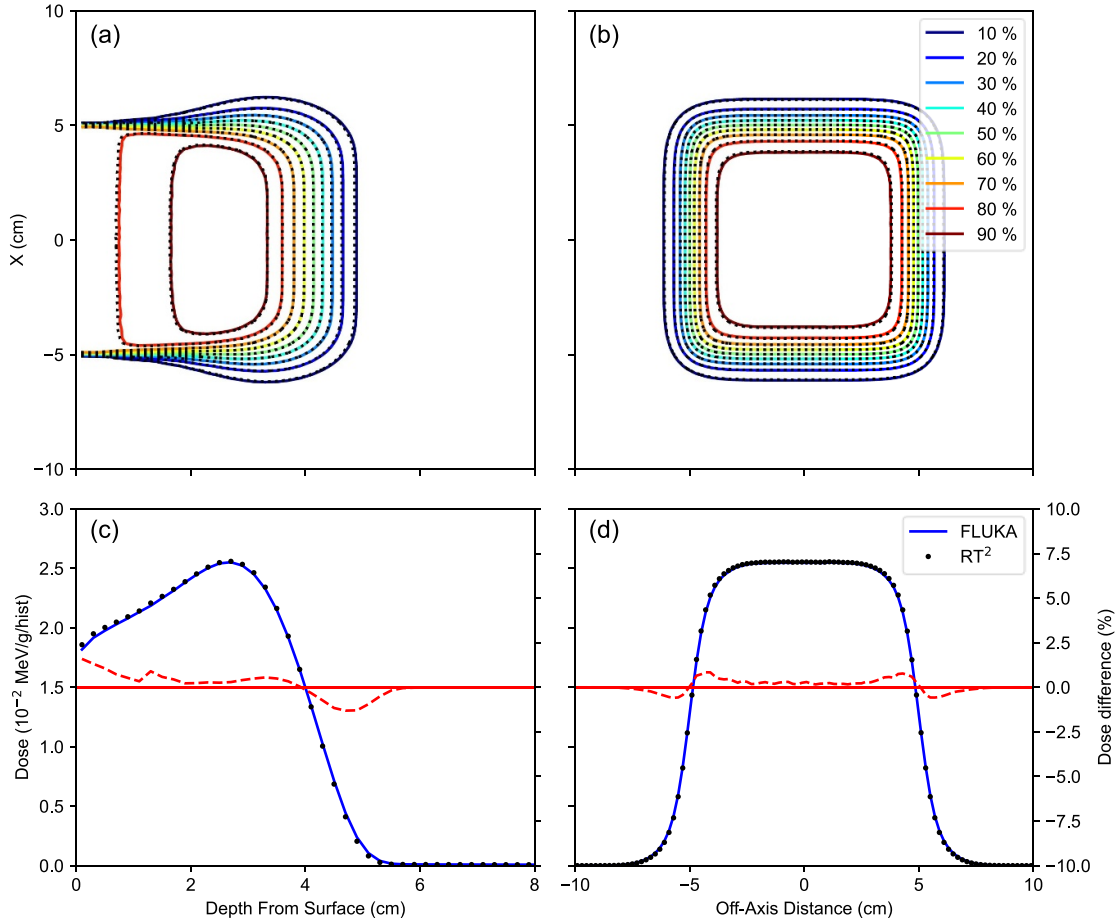

Figure 8. Dose calculation results for a 10 MeV electron beam in a water phantom. The isodose curve of FLUKA calculation results (colored solid lines) and RT^2 (black symbols) are shown in 2D image. Isodose curve in *XZ* plane that contain the beam axis (a), in *XY* plane at 3 cm from the surface (b). The depth-dose and lateral-dose profile are shown in (c) and (d) respectively. Dotted red lines present dose differences between the FLUKA and RT^2.

voxel size. The neutron beam which has a circular cross-section with a radius of 5 cm was applied. The center of mass of the designated tumor was penetrated by the beam axis. Figure 12 illustrates the comparison results of FLUKA and RT^2. The white region inside of the brain indicates a designated tumor. The depth and lateral dose distribution were obtained on the axes that penetrate the center of mass of the tumor. The dose difference was below 1% for all positions along the beam and lateral axes in this case.

4.3. Performance analysis against FLUKA

The summarized results of performance and uncertainty comparisons are listed in table 2. It should be noted that all available 20 threads were used in the FLUKA calculation in all cases. The maximum and averaged statistical uncertainty of all voxels whose doses were larger than 20% of the maximum are listed in $\sigma_{\max}$ and σ_{mean}, respectively. In the case of a heterogeneous phantom, some mesh bins exhibited significantly higher uncertainties than the average. This is primarily due to the occurrence of fewer actual energy deposition events despite a notable dose presence in low-density voxels.

In the photon and electron cases, acceleration factors of more than 150 were observed for all cases, while values in the range of 77–140 were observed in the neutron transport cases. These outcomes are most likely attributed to the memory access pattern. In the case of photon and electron transport, only a small amount of cross-section data was required, and most of the time was spent on executing the PRESTA-II algorithm. Conversely, neutron reactions did not require special operations as there was no specific physics model. Instead, the detailed angular distribution tables for each energy were required for every single scattering in neutron transport. As previously described, these operations lead to uncoalesced memory access and degrade GPU efficiency. This is the reason for the decreased acceleration factor in neutron transport. Nevertheless, RT^2 yielded significantly higher speeds than CPUs, offering the potential for medical applications of neutron and photon/electron dose calculations.

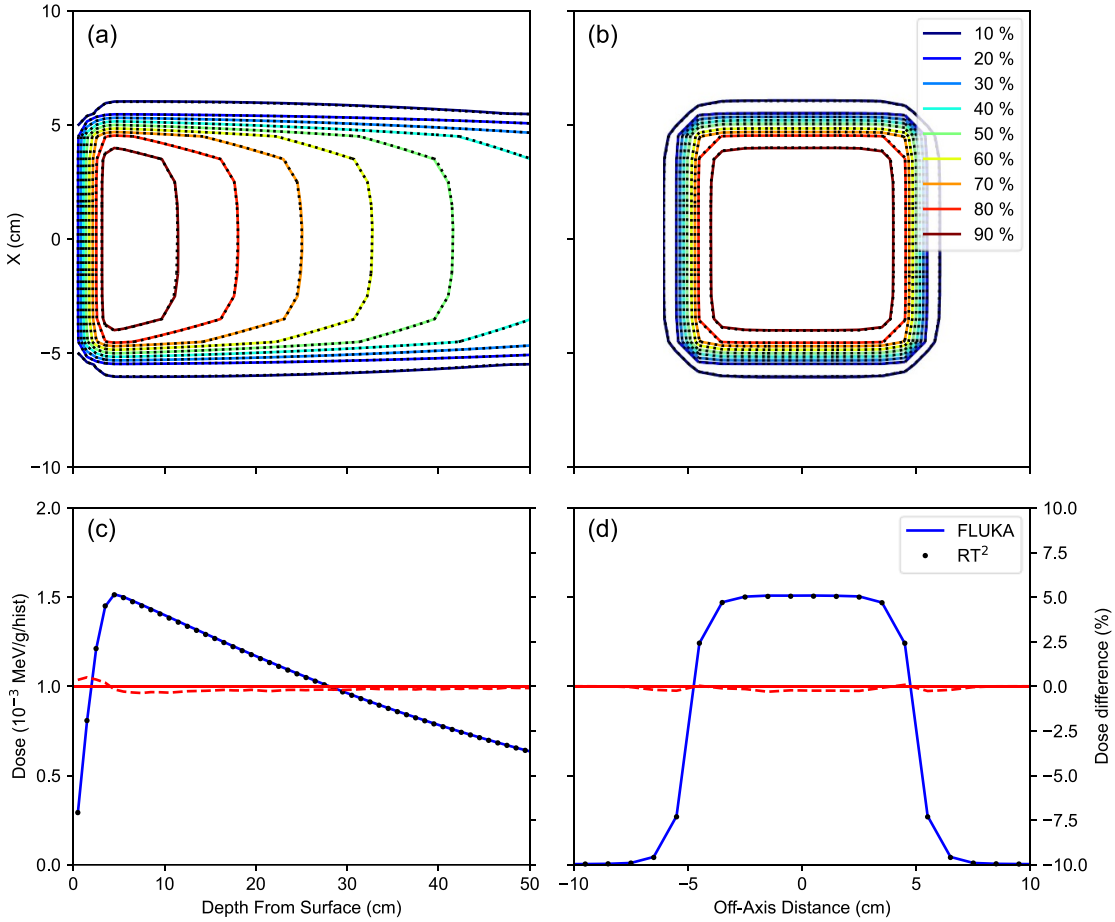

Figure 9. Dose calculation results for a 10 MeV x-ray beam in a water phantom. The isodose curve of FLUKA calculation results (colored solid lines) and RT^2 (black symbols) are shown in 2D image. Isodose curve in *XZ* plane that contain the beam axis (a), in *XY* plane at 5 cm from the surface (b). The depth-dose and lateral-dose profile are shown in (c) and (d) respectively. Dotted red lines present dose differences between the FLUKA and RT^2.

In a heterogeneous geometry, the acceleration factor for neutron calculation increased. Owing to the hardware RT acceleration applied in RT^2, the geometry tracking efficiency of the GPU architecture was much higher than CPU in the complicated geometry of CT. Therefore, the relative gain has become larger than the simple cube phantom geometry.

4.4. Kernel statistical analysis

The Nvidia Nsight Compute was utilized to analyze the number of calls, execution time, compute throughput, memory throughput and branch divergence for each kernel (NVIDIA 2023).

4.4.1. Metrics overview

As shown in figure 1, each bunch of events is processed as the kernel is repeatedly called. In this study, the dimensions of launched kernels were set to 16 384 × 128, indicating that 2 097 152 events were processed in parallel in a single kernel invocation. The share (%) metric represents a ratio of the number of individual kernel calls to the total number of calls. The summation of the share of all kernels is 100%. Time (μs) is the average execution time per a single kernel invocation.

Memory and compute throughput (%) represent the achieved percentage of averaged utilization of the compute unit and memory pipeline with respect to the theoretical maximum respectively. According to Nsight guidance, the program can be considered optimally optimized if both throughputs exceed 80%. If metrics are unbalanced, this is attributed to either a memory-bound or compute-bound algorithm.

As in CPU memory hierarchy, GPU also uses register, L1, L2 and global memory to minimize memory access latency. The global memory corresponds to the CPU's DRAM and has a large capacity of several GB, but has high-access latency. As Monte Carlo algorithms rely on large-scale sampling tables and cross-section data, their performance is significantly influenced by the global memory access pattern. Therefore, two

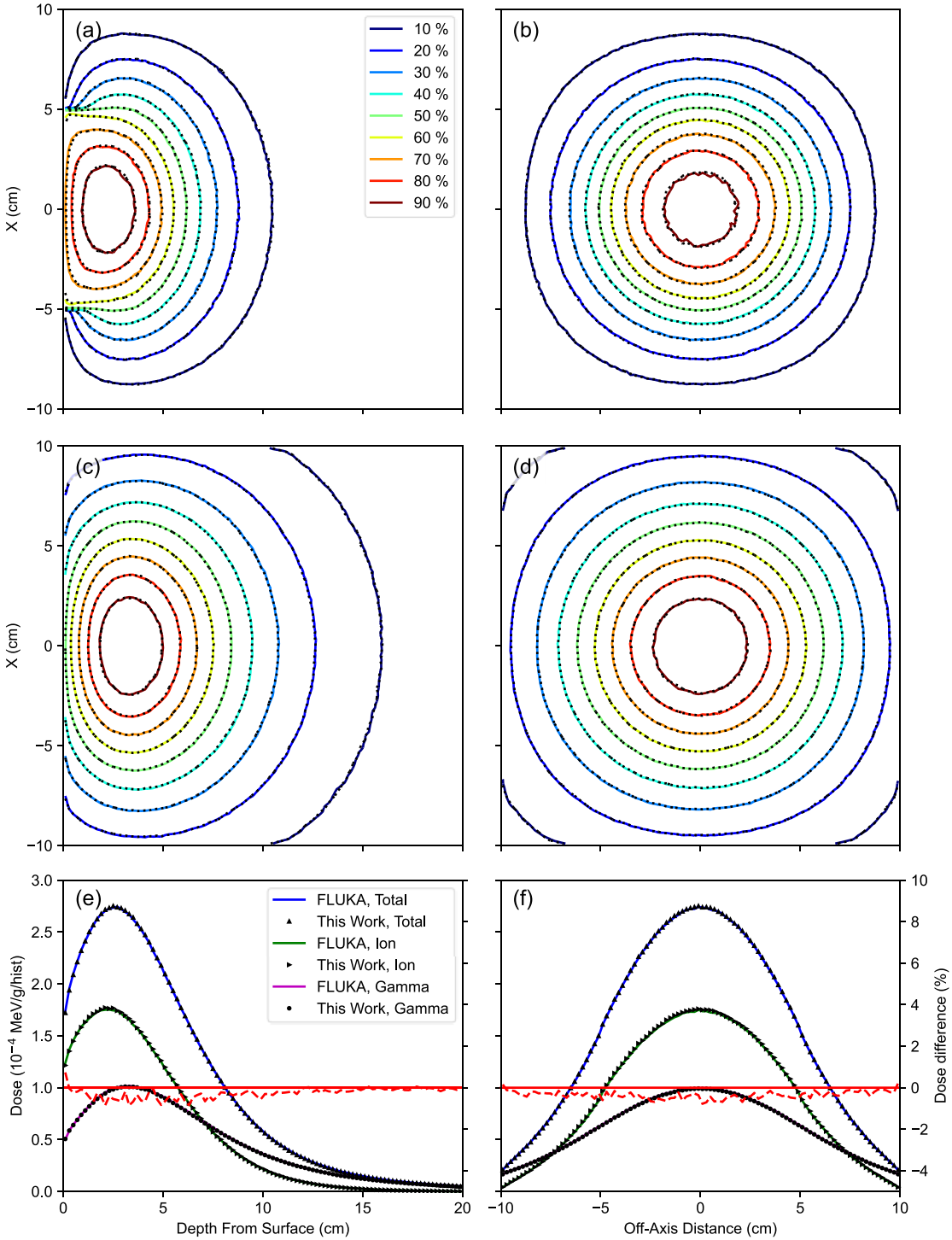

Figure 10. Dose calculation results for a 10 keV neutron beam in a boronated water phantom. The isodose curve of FLUKA calculation results (colored solid lines) and RT^2 (black symbols) are shown in 2D image. Isodose curve of ion dose in *XZ* plane that contain the neutron beam axis (a), ion dose in *XY* plane at 3 cm depth from surface (b), gamma-ray dose in *XZ* plane (c) and gamma-ray dose in *XY* plane in same position with (b) (d). The depth-dose and lateral-dose profile are shown in (e) and (f) respectively. Dose differences of RT^2 against to FLUKA are presented as red-dotted lines.

metrics, global load and store hit rate (%) metrics are considered in the performance analysis. These metrics indicate the number of sector hits for memory access per sector of instruction. A lower hit rate can suggest inefficient memory coalescing in a global memory access.

Branch efficiency indicates the proportion of branch targets where all active threads selected the same instruction. Branch divergence can lower branch efficiency. A branch efficiency of 100% indicates that no branches diverged in a kernel.

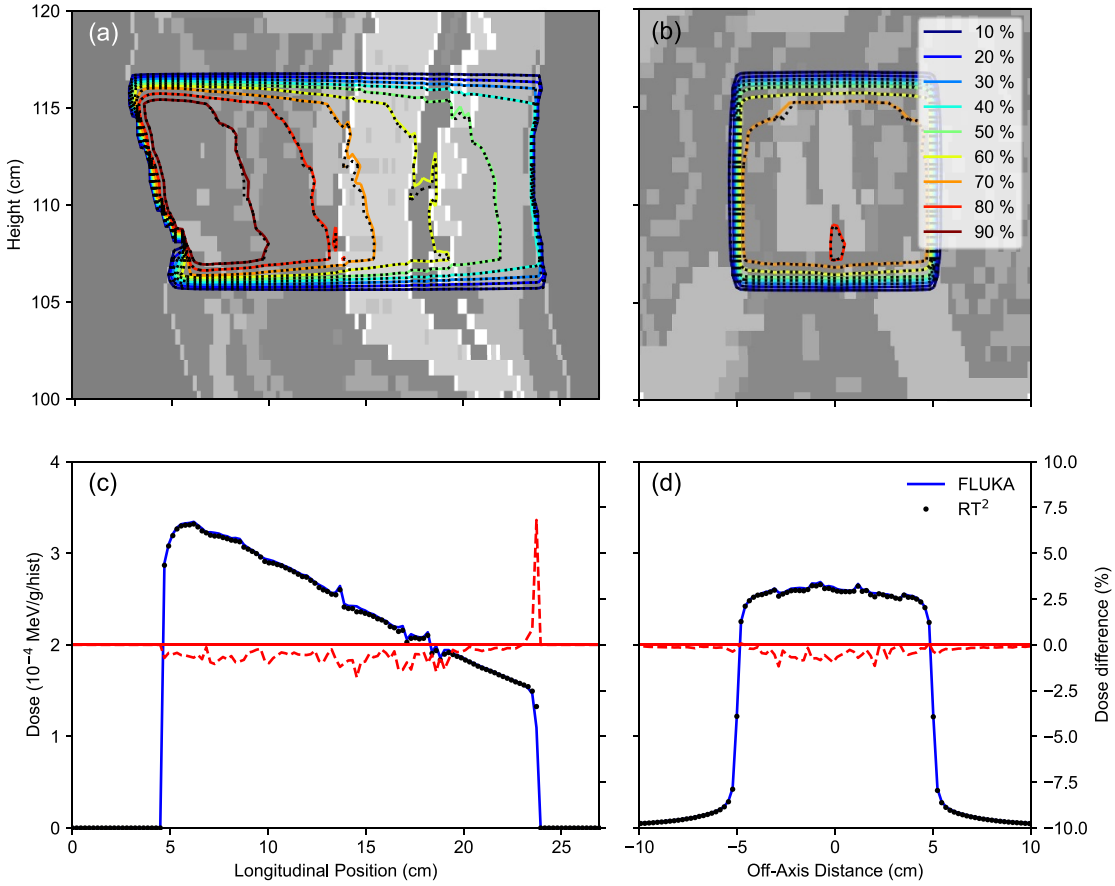

Figure 11. Dose calculation results for a 6 MV x-ray beam in the stomach. The isodose curve of FLUKA calculation results (colored solid lines) and RT^2 (black symbols) are shown in 2D image. Isodose curve in sagittal plane that contain the x-ray beam axis (a), in frontal plane at 14 cm longitudinal position (b). The depth-dose and lateral-dose profile are shown in (e) and (f) respectively. Dotted red lines present dose differences between the FLUKA and RT^2.

4.4.2. Performance analysis

Table 3 summarize kernel launch statistics measured by Nvidia Nsight Compute. Each column presents the major four kernels and their shares proportional to the number of their launches for six cases, rather than the execution time. In the water phantom with electrons, the dominant kernel was the electron transport. Even in the case of 10 MeV photon beam, the electron transport kernel was dominant. Since electrons underwent a series of multiple scatterings in the interval between hard interactions, the share of electron transport became larger than that of photon transport. In contrast, the photon transport kernel was dominant in the 6 MV x-ray primary, CT-based phantom case. In this geometry, photons through a series of boundary-crossing events in numerous voxels. Conversely, electron steps were already sufficiently small compared with the CT voxel owing to the multiple Coulomb scattering. These factors made the photon transport kernel dominant in the CT voxel geometry.

The details of kernel launch statistics for the 50 MeV electron case are presented in table 4. Any transport or interaction kernels that have a proportion higher than 1% are listed. Transport kernels are composed of a set of several sub-kernels. The OptiX RT kernel handles the geometry tracking and calculates the track length of the next closest surface boundary. As these operations are identical regardless of the type of particles, both the photon and electron transport kernels share the same OptiX RT kernel. Consequently, all metrics are identical in the electron and photon transport. The PRESTA-II kernel calculates the lateral displacement and path length correction in the condensed-history algorithm. This kernel was used only in charged particle processes. The transport kernel calculated the free path length and sample interactions. The total execution time is the sum of the execution time of these sub-kernels.

In all the kernels, memory throughput was much higher than compute throughput. Even in the PRESTA-II kernel, which is associated with many mathematical operations, memory throughput was dominant as the alias tables $R_{\mathrm{mott}}(Z, T, \mu)$ in equation (4) and $F_{\mathrm{SR}}^{(2+)}(\mu)$ in equation (8) were accessed

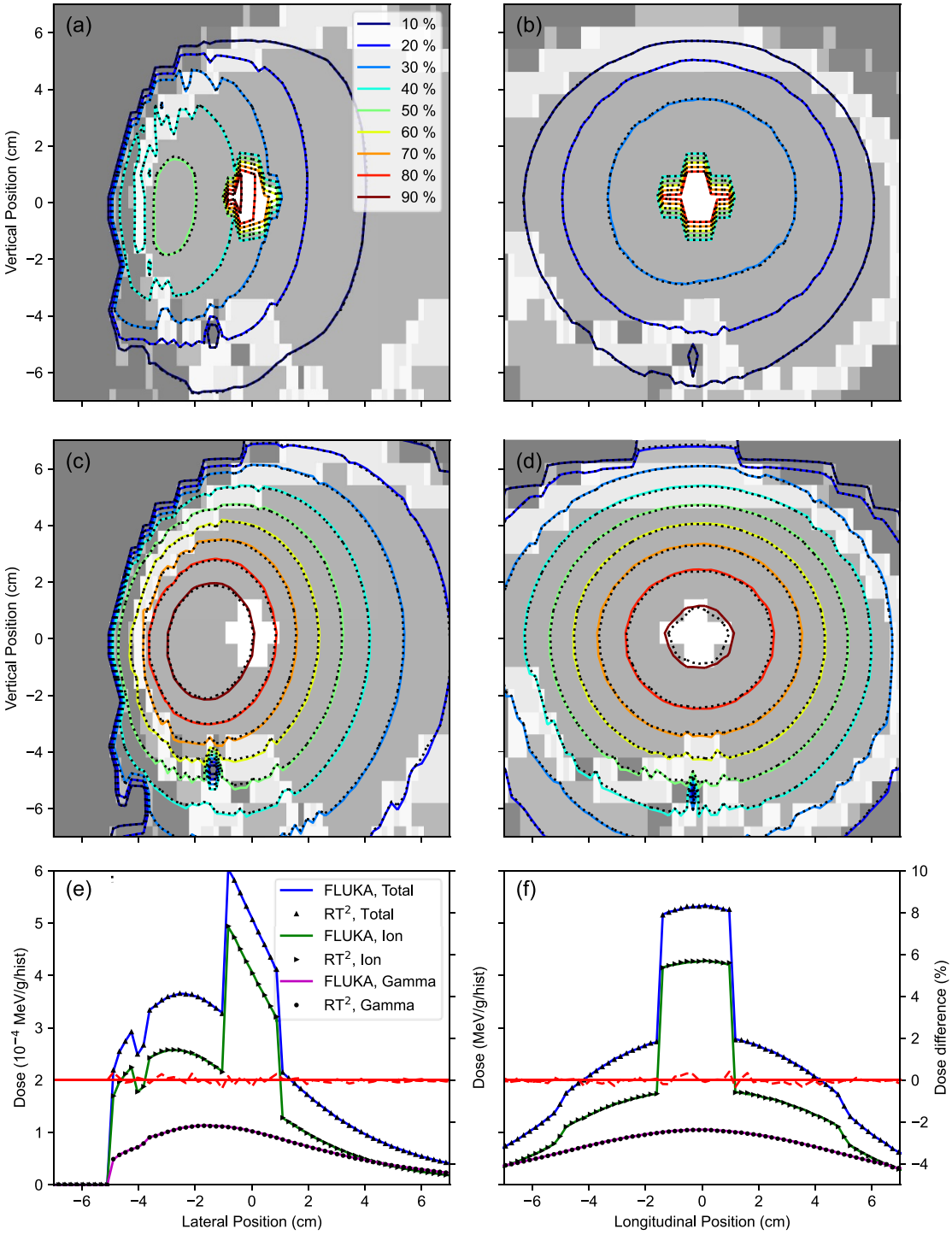

Figure 12. Dose calculation results in a scenario of head and neck BNCT. The isodose curve of FLUKA calculation results (colored solid lines) and this work (black symbols) are shown in 2D image. Isodose curve of ion dose in frontal plane that contains the neutron beam axis (a), ion dose in sagittal plane (b), gamma-ray dose in frontal plane (c) and gamma-ray dose in sagittal plane (d). The depth- and lateral-dose profiles are shown in (e) and (f), respectively. The red-dotted lines present dose differences of RT^2 against to the FLUKA.

frequently in multiple Coulomb scattering. The branch efficiency of the PRESTA-II kernel was also decreased by rejection sampling in the spin effect. The memory throughputs of electron and photon transport were smaller than those for other kernels such as PRESTA-II and interaction, as the memory access pattern in the last step was uncoalesced as illustrated in figure 3. In the interaction kernel, high compute throughput was observed in all types. This was because of the coalesced memory access patterns.

Table 2. The performance comparison for four monoenergetic beams in a water phantom and x-ray and neutron beam in the reference phantom. FLUKA calculations utilized all 20 threads of a single Intel Core i9-10900k processor, while RT^2 computations were performed on a single Nvidia RTX 4090 card. The acceleration factors, defined by equation (10), are presented in the last column.

		FLUKA			RT2			
Cases	Histories	Time (s)	σ_{max} (%)	σ_{mean} (%)	Time (s)	σ_{max} (%)	σ_{mean} (%)	Acceleration factor
50 MeV electron	1×10^9	47 675	0.08	0.04	215	0.04	0.03	221.7
10 MeV electron	5×10^9	62 150	0.17	0.08	212	0.11	0.06	293.2
10 MeV photon	5×10^9	61 850	0.13	0.07	238	0.06	0.04	259.9
10 keV neutron	7.5×10^{10}	199 650	0.66	0.46	2 565	0.59	0.44	77.8
6 MV x-ray	1×10^{11}	380 250	1.36	0.07	2 269	1.35	0.04	167.6
BNCT	4×10^{10}	220 200	0.30	0.20	1 559	1.12	0.20	141.2

Table 3. Kernel launch statistics measured by Nvidia Nsight Compute: each column presents the major four kernels and their shares proportional to the number of their launches for six cases, rather than the execution time.

Cases	1st kernel & share (%)		2nd kernel & share (%)		3rd kernel & share (%)		4th kernel & share (%)	
50 MeV electron	e− transport	65.68	γ transport	14.04	Compton	11.27	Moller	5.65
10 MeV electron	e− transport	82.96	Moller	7.06	γ transport	5.44	Compton	2.74
10 MeV photon	e− transport	58.50	γ transport	18.61	Compton	14.21	e+ transport	3.80
10 keV neutron	n transport	95.00	e− transport	2.89	γ transport	1.56	Compton	0.46
6 MV x-ray	γ transport	70.66	e− transport	21.38	Compton	7.33	Moller	0.26
BNCT	n transport	91.15	γ transport	5.99	e− transport	2.38	Compton	0.38

Table 4. Details of kernel statistics for a 50 MeV electron simulation in a water phantom. The launched kernel dimension was 16 384 × 128. Transport and interaction kernels above 1% share are presented.

Kernel	Time (μs)	Compute throughput (%)	Memory throughput (%)	Global load hit rate (%)	Global store hit rate (%)	Branch efficiency (%)	Share (%)
Electron transport							
OptiX RT	120.42	27.05	56.14	59.15	44.95	99.99	
PRESTA-II	1285.80	9.66	93.26	83.43	84.35	84.75	65.68
Transport	335.64	22.77	66.66	60.61	36.98	83.53	
Photon transport							
OptiX RT	120.42	27.05	56.14	59.15	44.95	99.99	14.04
Transport	542.10	13.56	65.15	73.36	79.36	82.38	
Interactions (>1%)							
Compton	436.37	9.37	89.77	53.62	65.90	75.18	11.27
Moller	783.41	9.11	91.27	56.31	81.53	64.92	5.65
e− Bremss	454.37	11.79	86.28	71.92	77.19	95.23	1.61

The Compton and Moller scattering kernels exhibited lower branch efficiencies than electron bremsstrahlung. In these kernels, rejection sampling was utilized to simulate the directional cosine of the secondary. In contrast, bremsstrahlung under 85 MeV relies on an alias table so that rejection sampling can be avoided as described in section 2.4.2.1. As rejection sampling causes branch divergence, the branch efficiencies of Compton and Moller scatterings were lower than those of bremsstrahlung.

The details of Kernel launch statistics for 10 keV neutron in the water phantom and BNCT in the heterogeneous CT-based phantom are presented in tables 5 and 6, respectively. The neutron transport kernel was dominant in both cases, while the proportion of the photon transport kernel was much larger in the CT-based geometry than in the bulky water phantom. The execution time of the OptiX RT kernel was increased from 120 to 400, as the geometry was more complicated. However, this increment of computation time was tolerable even in case in which the geometry complexity was tremendously increased. The number of triangle segments in the water phantom was 12, whereas in the CT-based geometry it was 467 360. The RT hardware acceleration allows RT^2 to calculate the complex geometry very efficiently, and the gain compared to FLUKA increased from 77 to 140 as shown in table 2.

Table 5. Details of kernel statistics for a 10 keV neutron simulation in a boronated water phantom. The launched kernel dimension was 16 384 × 128. Transport and interaction kernels above 1% share are presented. As the neutron transport kernel was dominant, none of the interaction kernels launched with >1% share.

Kernel	Time (μs)	Compute throughput (%)	Memory throughput (%)	Global load hit rate (%)	Global store hit rate (%)	Branch efficiency (%)	Share (%)
Neutron transport							
OptiX RT	121.61	26.98	55.14	59.34	47.21	98.42	95.00
Transport	791.00	11.40	91.11	77.70	86.17	90.83	
Electron transport							
OptiX RT	121.61	26.98	55.14	59.34	47.21	98.42	
PRESTA-II	809.25	13.56	92.16	90.34	81.18	90.99	2.89
Transport	469.53	15.52	49.36	58.68	21.59	84.44	
Photon transport							
OptiX RT	121.61	26.98	55.14	59.34	47.21	98.42	1.56
Transport	381.84	13.32	83.30	62.86	75.42	87.33	
Interactions (>1%)							

Table 6. Details of kernel statistics for a BNCT scenario in a reference phantom. The launched kernel dimension was 16 384 × 128. Transport and interaction kernels above 1% share are presented. As the neutron transport kernel was dominant, none of the interaction kernels launched with >1% share.

Kernel	Time (μs)	Compute throughput (%)	Memory throughput (%)	Global load hit rate (%)	Global store hit rate (%)	Branch efficiency (%)	Share (%)
Neutron transport							
OptiX RT	396.67	8.32	76.77	27.76	22.72	97.93	91.15
Transport	686.77	13.80	90.31	71.94	84.63	95.68	
Photon transport							
OptiX RT	396.67	8.32	76.77	27.76	22.72	97.93	5.99
Transport	375.47	15.42	89.33	60.65	65.65	83.28	
Electron transport							
OptiX RT	396.67	8.32	76.77	27.76	22.72	97.93	
PRESTA-II	826.61	13.41	89.44	81.24	80.63	88.53	2.38
Transport	573.36	20.99	41.85	66.29	41.05	89.20	
Interactions (>1%)							

5. Conclusions

In this study the uncoalesced memory access and branch divergence commonly encountered in coupled Monte Carlo transport on the GPU architecture were appropriately resolved. The RT core of Nvidia OptiX™ framework was successfully utilized to accelerate further particle tracking. Thus, the formidable acceleration of computation times compared with those of the CPU was achieved while maintaining an accuracy comparable to that obtained using the CPU Monte Carlo code, FLUKA. The performance achieved in this study can promise a routine use of Monte Carlo simulations in medical applications.

As the neutron transport kernel is memory bound, the on-the-fly calculation of Legendre polynomial in point-wised neutron transport is more efficient than lookup table in group-wised transport. The performance impact profile of these two methods will be investigated in the optimization process. The atomic relaxation, electron-impact-ionization and Compton scattering of orbital electron are under development.

Data availability statement

The data cannot be made publicly available upon publication because they contain commercially sensitive information. The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Ministry of Science and ICT (MSIT) (No. RS-2023-00237149), Ministry of Education (No. 2022R1A6A1A03063039) to Seoul National University and by the Commercialization Promotion Agency for R&D Outcome (COMPA) to establish T-ROH Inc. (No. 1711198939-00254046).

ORCID iDs

Chang-Min Lee https://orcid.org/0009-0008-8412-8015
Sung-Joon Ye https://orcid.org/0000-0001-8714-6317

References

Agostinelli S *et al* 2003 GEANT4—a simulation toolkit *Nucl. Instrum. Methods Phys. Res.* A **506** 250–303
Badal A and Badano A 2009 Accelerating Monte Carlo simulations of photon transport in a voxelized geometry using a massively parallel graphics processing unit *Med. Phys.* **36** 4878–80
Battistoni G *et al* 2015 Overview of the FLUKA code *Ann. Nucl. Energy* **82** 10–18
Bedford J L 2002 Speed versus accuracy in a fast convolution photon dose calculation for conformal radiotherapy *Phys. Med. Biol.* **47** 3475–84
Blyth S, Biscarat C, Campana S, Hegner B, Roiser S, Rovelli C I and Stewart G A 2021 Integration of JUNO simulation framework with Opticks: GPU accelerated optical propagation via NVIDIA® OptiX™ *EPJ Web Conf.* **251** 03009
Burgess J 2020 RTX on—the NVIDIA turing GPU *IEEE Micro* **40** 36–44
Chadwick M B *et al* 2006 ENDF/B-VII.0: next generation evaluated nuclear data library for nuclear science and technology *Nucl. Data Sheets* **107** 2931–3060
Cullen D E, Hubbell J H and Kissel L 1997 EPDL97: the evaluated photo data library'97 version *Technical Report* https://doi.org/10.2172/295438
Goudsmit S and Saunderson J L 1940 Multiple scattering of electrons *Phys. Rev.* **57** 24–29
Hamilton S P and Evans T M 2019 Continuous-energy Monte Carlo neutron transport on GPUs in the Shift code *Ann. Nucl. Energy* **128** 236–47
Hubbell J H and Overbo I 1979 Relativistic atomic form factors and photon coherent scattering cross sections *J. Phys. Chem. Ref. Data* **8** 69–106
ICRP 2009 Adult reference computational phantoms: joint ICRP/ICRU report. ICRP publication 110 *Ann. ICRP* **39** 1–164
Jia X, Gu X, Graves Y J, Folkerts M and Jiang S B 2011 GPU-based fast Monte Carlo simulation for radiotherapy dose calculation *Phys. Med. Biol.* **56** 7017–31
Kawrakow I 2000 Accurate condensed history Monte Carlo simulation of electron transport. II. Application to ion chamber response simulations *Med. Phys.* **27** 499–513
Kawrakow I and Bielajew A F 1998 On the representation of electron multiple elastic-scattering distributions for Monte Carlo calculations *Nucl. Instrum. Methods Phys. Res.* B **134** 325–36
Lee C-M and Lee H-S 2022 Development of a dose estimation code for BNCT with GPU accelerated Monte Carlo and collapsed cone Convolution method *Nucl. Eng. Technol.* **54** 1769–80
Lee S, Chang H, Lee J, Kye Y U and Ye S-J 2020 Neutron yields of Be-9(p,xn) reactions and beam characterization for accelerator-based boron neutron capture therapy facility using MCNP6, PHITS, and GEANT4 simulation results *Nucl. Instrum. Methods Phys. Res.* B **478** 233–8
Lippuner J and Elbakri I A 2011 A GPU implementation of EGSnrc's Monte Carlo photon transport for imaging applications *Phys. Med. Biol.* **56** 7145–62
Macfarlane R, Muir D W, Boicourt R M, Kahler A C III and Conlin J L 2017 *The NJOY Nuclear Data Processing System, Version 2016* https://doi.org/10.2172/1338791
Moliere G 1947 Theorie der Streuung schneller geladener Teilchen I. Einzelstreuung am abgeschirmten Coulomb-Feld *Z. Naturforsch.* A **2** 133–45
Mott N F 1929 The scattering of fast electrons by atomic nuclei *Proc. R. Soc.* A **124** 425–42
Motz J W, Olsen H A and Koch H W 1969 Pair production by photons *Rev. Mod. Phys.* **41** 581–639
National Research Council of Canada, Metrology Research Centre, Ionizing Radiation Standards 2021 EGSnrc: software for Monte Carlo simulation of ionizing radiation (National Research Council of Canada) https://doi.org/10.4224/40001303
Navarro C A, Hitschfeld-Kahler N and Mateu L 2014 A survey on parallel computing and its applications in data-parallel problems using GPU architectures *Communications in Computational Physics* vol 15 (Global Science Press) pp 285–329
NVIDIA 2023 *NVIDIA Nsight Compute* (March) (available at: https://docs.nvidia.com/nsight-compute)
NVIDIA, Vingelmann P and Fitzek F H P 2020 CUDA, release: 10.2.89 (available at: https://developer.nvidia.com/cuda-toolkit)
Øverbø I, Mork K J and Olsen H A 1968 Exact calculation of pair production *Phys. Rev.* **175** 1978–81
Parker S G *et al* 2010 OptiX: a general purpose ray tracing engine *ACM Trans. Graph.* **29** 66
Ridley G and Forget B 2021 A simple method for rejection sampling efficiency improvement on SIMT architectures *Stat. Comput.* **31** 30

Salmon J and McIntosh-Smith S 2019 Exploiting hardware-accelerated ray tracing for Monte Carlo particle transport with OpenMC *Proc. PMBS 2019: Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems—Held in Conjunction with SC 2019: The Int. Conf. for High Performance Computing, Networking, Storage and Analysis* (*November 2019*) (Institute of Electrical and Electronics Engineers Inc.) pp 19–29

Sanders J and Kandrot E 2010 *CUDA by Example an Introduction to General-purpose GPU Programming* (Addison-Wesley Professional)

Sato T *et al* 2015 Overview of particle and heavy ion transport code system PHITS *Ann. Nucl. Energy* **82** 110–5

Swanepoel M W 2010 The role of the 14N(n,p)14C reaction in neutron irradiation of soft tissues *Radiat. Meas.* **45** 1458–61

Werner C *et al* 2018 *MCNP Version 6.2 Release Notes* https://doi.org/10.2172/1419730

X-5 Monte Carlo Team 2003 *MCNP-A General Monte Carlo N-Particle Transport Code, Version 5* (*Overview and Theory* vol I) (Monte Carlo Team)