Nuclear Science and Techniques (2026) 37:18
https://doi.org/10.1007/s41365-025-01838-6

GPU-accelerated Monte Carlo method for dose calculation of mesh-type computational phantoms

Shu-Chang Yan[1,2] · Rui Qiu[1,2] · Xi-Yu Luo[1,2] · An-Kang Hu[1,2] · Zhen Wu[1,2,3] · Jun-Li Li[1,2]

Received: 7 June 2024 / Revised: 17 August 2024 / Accepted: 27 August 2024 / Published online: 9 December 2025

Abstract

Computational phantoms play an essential role in radiation dosimetry and health physics. Although mesh-type phantoms offer a high resolution and adjustability, their use in dose calculations is limited by their slow computational speed. Progress in heterogeneous computing has allowed for substantial acceleration in the computation of mesh-type phantoms by utilizing hardware accelerators. In this study, a GPU-accelerated Monte Carlo method was developed to expedite the dose calculation for mesh-type computational phantoms. This involved designing and implementing the entire procedural flow of a GPU-accelerated Monte Carlo program. We employed acceleration structures to process the mesh-type phantom, optimized the traversal methodology, and achieved a flattened structure to overcome the limitations of GPU stack depths. Particle transport methods were realized within the mesh-type phantom, encompassing particle location and intersection techniques. In response to typical external irradiation scenarios, we utilized Geant4 along with the GPU program and its CPU serial code for dose calculations, assessing both computational accuracy and efficiency. In comparison with the benchmark simulated using Geant4 on the CPU using one thread, the relative differences in the organ dose calculated by the GPU program predominantly lay within a margin of 5%, whereas the computational time was reduced by a factor ranging from 120 to 2700. To the best of our knowledge, this study achieved a GPU-accelerated dose calculation method for mesh-type phantoms for the first time, reducing the computational time from hours to seconds per simulation of ten million particles and offering a swift and precise Monte Carlo method for dose calculation in mesh-type computational phantoms.

Keywords GPU Monte Carloference · Mesh-type phantom · External exposure · Heterogeneous

1 Introduction

Computational phantoms play a crucial role in radiation dosimetry and health physics and are remarkably useful in radiation treatments, unforeseen radiation incidents, and occupational exposures [1–7]. Voxel- and mesh-type phantoms are the primary models currently used in domestic applications [7–10]. Compared with voxel-type models constructed from uniformly sized cubic voxels, mesh-type models, representing the latest generation of anatomical models, are depicted by boundary surfaces or meshes, offering the dual benefits of pose adaptability and high resolution [8]. This allows for the definition of human tissue and organ profiles with arbitrary precision and the maintenance of smooth contours [11, 12], overcoming the limitations often observed in voxel-type models, such as stair-stepped surfaces, resolution restrictions, and adaptability issues [3]. Therefore, mesh-type phantoms exhibit substantial potential in the domains of radiation therapy, radiation protection, and individual dosimetry assessment, where an accurate anthropomorphic model is crucial for obtaining precise individual doses [13, 14].

The use of computational phantoms in Monte Carlo simulations is essential for assessing human doses [7, 9,

This work was supported by the National Natural Science Foundation of China (Nos. U2167209 and 12375312), Open-end Fund Projects of China Institute for Radiation Protection Scientific Research Platform (CIRP-HYYFZH-2023ZD001).

✉ Rui Qiu
qiurui@tsinghua.edu.cn

[1] Department of Engineering Physics, Tsinghua University, Beijing 100084, China

[2] Key Laboratory of Particle and Radiation Imaging, Tsinghua University, Ministry of Education, Beijing 100084, China

[3] Nuctech Company Limited, Beijing 100084, China

15]. In Monte Carlo simulations, it is critical to determine the relationship between the particles and the surrounding geometry [16, 17]. When mesh models are directly utilized in computational simulations, aligning their positions with all facets significantly decreases computational speed. Research conducted at Hanyang University in South Korea demonstrated that mesh models require 70–150 times more computation time than voxel models [18]. In 2014, Professor Chan Hyeong Kim et al. developed a method to accelerate the computation of mesh phantoms by employing tetrahedral decomposition [19]. This approach significantly reduces the computation time because the particle transport is determined by comparing the position of the particle with only the four faces of a tetrahedron, rather than the entire mesh, thereby achieving speeds comparable to those of the voxel-model geometries. However, in fields where stringent time constraints are vital, such as clinical radiation therapy, nuclear medicine, and dose reconstruction for accidents, the computation time of tetrahedral models still significantly exceeds the acceptable limits, thus restricting their applicability [20, 21].

GPU acceleration is a method employed to expedite Monte Carlo simulations, thus enhancing the computational efficiency [17, 22, 23]. Recently, GPUs have undergone a meteoric development. Compared to CPUs, GPUs exhibit superior single-precision floating point computational capabilities and memory bandwidths. Moreover, GPU hardware systems offer enhanced ease of maintenance at lower costs. Simultaneously, in the post-2010 era, a variety of GPU-accelerated photonic transport and photon–electron coupled transport programs, such as gDPM [24], gSMC [25], and FRED [26], have been developed, either by building upon the original fast Monte Carlo framework or by creating entirely new implementations, with extensive verification and efficiency assessments. Nonetheless, to the best of our knowledge, none of the prevailing GPU programs have yet attained GPU acceleration predicated on mesh-type models. The primary challenge stems from the complexity of achieving rapid particle transport within tetrahedral mesh-type models on a GPU, whereas the limited stack depth of GPUs complicates the execution of the recursive algorithms necessary for navigating these intricate structures, further hindering performance [27].

To compensate for the deficiencies in Monte Carlo simulations for mesh-type phantoms, this study crafted a GPU-based photon–electron coupled accelerated dose calculation program, realizing the transport of particles within tetrahedral models on GPU devices for the first time. Rigorous validation of the program was conducted, demonstrating precise computational outcomes and significant acceleration, thus successfully addressing the challenges associated with dose-acceleration calculations for mesh-type phantoms.

2 Materials and methods

2.1 Manipulation of a mesh-type phantom

2.1.1 Constructing the acceleration structure

During the Monte Carlo simulation, to determine the physical and geometric step lengths, the particle transport process requires traversal of all geometries for particle positioning and intersection operations [16, 17, 28]. The voxel-type phantom is depicted as a three-dimensional array consisting of cuboidal voxels. Organized into columns, rows, and slices, each voxel acts as a volumetric unit. An entry within this array defines a specific organ or tissue corresponding to each voxel [3, 8]. Because the voxel phantom features a uniform grid of identically sized voxels, the voxel in which a particle is located based on its position can be ascertained swiftly. By contrast, the mesh-type phantom employs an unstructured grid composed of non-uniform rational B-spline(NURBS) surfaces or polygon surfaces, where the NURBS-surface phantoms cannot be directly calculated, and computations involving polygon-surface phantoms are time-intensive [8, 11, 12]. To address this issue, mesh-type phantoms are typically tetrahedralized, resulting in a tetrahedral mesh-type phantom that retains the external shape of the original model, but internally comprises numerous stacked tetrahedra [19]. This configuration significantly reduces the time required for intersection operations because it only requires a comparison of the distances to the four faces of the tetrahedron surrounding the particle. Although tetrahedralization simplifies the intersection operations, the resulting tetrahedral mesh-type phantom remains unstructured, with each tetrahedron differing in shape. Consequently, particle positioning still requires traversing all tetrahedra. Simultaneously, calculating the distance from an external particle to the surface of the tetrahedral model involves an exhaustive search across all the tetrahedra to find the one intersected by the particle's direction of motion at the shortest distance. These traversal processes in the tetrahedral model are time-consuming.

To optimize data traversal efficiency, it becomes essential to implement an acceleration structure [29, 30]. By systematically organizing data into layers, this structure significantly reduces search time and improves query efficiency, thereby facilitating the execution of complex computational tasks. With this consideration in mind, the construction of a tree-like acceleration structure for all tetrahedra within the tetrahedral model was proposed. This strategy is expected to notably decrease the temporal complexity of data traversal. Considering the temporal intricacy of construction and the utilization of memory resources, we adopted the bounding volume hierarchy (BVH) tree [31].

Central to the BVH is the concept of encapsulating scene entities within bounding volumes at various levels of the hierarchy, which effectively reduces unnecessary object testing and enhances algorithmic efficiency. In tetrahedral mesh-type phantoms, each tetrahedron is initially calculated for its bounding box to form a parent node. The tetrahedral mesh-type phantom, composed of numerous stacked tetrahedra, is simplified for a clearer visualization of the construction process of our acceleration structure. As illustrated in Fig. 1, the model is represented by eight discrete tetrahedra that are functionally equivalent. Initially, the bounding box is calculated for all the tetrahedra to form the parent node. These tetrahedra are then divided into two groups based on a specific pattern, such as the average division by length, and each group's bounding box is subsequently calculated to form child nodes. This division process continues for the left and right child nodes until the number of tetrahedra in a subdivided child node falls below a specified threshold, at which point the node becomes a leaf node. Consequently, only the leaf nodes contain detailed information specific to the tetrahedra, whereas the other nodes only store information regarding the bounding boxes and their child nodes. Figure 1 shows that the maximum number of tetrahedra in a leaf node is two. This means that the division stops when the number of tetrahedra in a child node is less than or equal to two, thereby establishing it as a leaf node. Importantly, a smaller threshold number of tetrahedra in the leaf nodes is not always advantageous because it increases the tree's depth, which in turn increases the memory space requirements and reduces the access speed, whereas a larger number might diminish the acceleration effect. In practice, during the division of the tetrahedral mesh-type phantom, the threshold number of tetrahedra in the leaf nodes was eight.

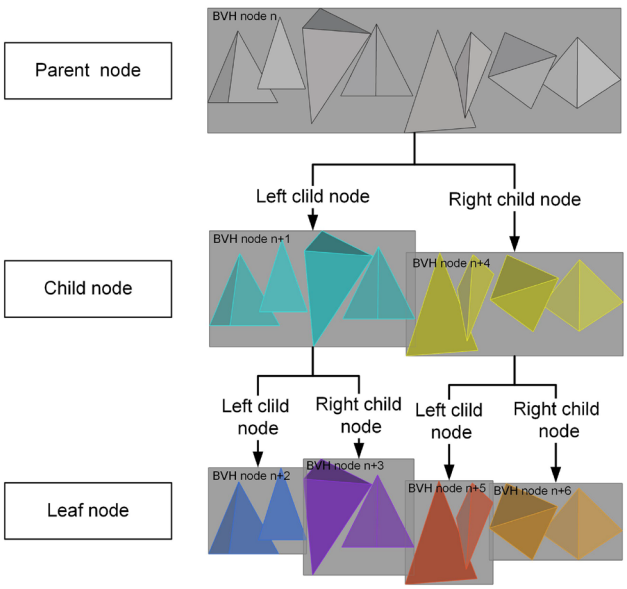


Fig. 1 (Color online) Schematic representation of a bounding volume hierarchy (BVH) tree. The gray boxes represent the minimum bounding boxes encompassing all tetrahedra within each node. Only the leaf nodes contain detailed information specific to the tetrahedra, whereas the other nodes solely store information about the bounding boxes and their child nodes

2.1.2 Optimization and treatment

When constructing a BVH tree, selecting an appropriate division strategy for each node is crucial. Simple division methods such as equal splits by axial length or tetrahedron count may not always yield optimal results. For example, dividing a complex phantom into upper and lower body sections of equal axial length can result in a disproportionate number of tetrahedra in the upper body owing to its denser organ composition. This disparity can significantly increase the depth difference between the two branches, thereby reducing the efficiency of the tree's access structure. Similarly, an equal division by the tetrahedron count may balance the number of tetrahedra; however, the varying sizes and uneven distribution can lead to substantial volume differences in the bounding box between branches, thereby disrupting the structural balance of the entire tree.

To address these challenges, a more complex partitioning strategy such as the surface area heuristic (SAH) algorithm [31, 32] can be employed. The SAH algorithm optimizes the construction of acceleration structures, such as BVH trees, by selecting the most cost-effective node-partitioning strategy. The core principle of the SAH algorithm involves calculating the expected cost of different partitioning schemes during node division, with the aim of selecting the scheme with the lowest cost to achieve the fastest traversal speed of the structure. Considering the necessity of similar traversal operations in our particle transport process in the phantom, employing the SAH algorithm to optimize the construction of our acceleration structures is a valuable strategy. Specifically, the SAH algorithm estimates the costs of traversing child nodes by assessing their surface area and the likely number of geometries they contain after a split. This space is divided into two parts, each of which forms a new child node. For each possible split, the SAH algorithm evaluates two main costs: the probability of traversing a child node (usually proportional to the node's surface area) and the cost of further traversal within that node (usually related to the number of tetrahedra in the node). Using this method, the SAH algorithm seeks the partitioning scheme with the lowest total cost, thereby minimizing the time required to traverse the acceleration structure. This approach is particularly effective in scenarios featuring complex geometries and uneven distributions, as it adapts to various shapes and sizes of geometries, ensuring that the construction of the acceleration structure is both balanced and efficient. Although the

SAH method may demand more upfront effort during the construction phase, the payoff in terms of enhanced traversal efficiency and speed is worth it, particularly considering the frequent traversal operations necessary during particle transport processes.

The substantial number of tetrahedra in a tetrahedral mesh-type phantom leads to a complex acceleration structure with a large nesting depth. This complexity presents significant challenges for GPU calculations, which are limited by finite stack depths [27]. These limitations constrain the number of recursive calls and the depth of the data structures that can be efficiently processed in parallel. Consequently, GPUs may struggle to effectively manage deeply nested tree-like structures because of their limited stack depth, which adversely affects the performance of recursive algorithms that are critical for traversing these structures. To overcome this limitation, flattening the BVH tree is essential, which means transforming the tree structure into a linear format for efficient parallel processing on GPUs [33]. Our implementation of the "tree flattening" method is illustrated in Fig. 2. Instead of directly creating new child nodes within a node, we record only the child node index in the node sequence. Consequently, this strategy results in two sequences: a BVH nodes list that sequentially stores information for each BVH node and a tetrahedra list that sequentially stores the geometric and material information of the tetrahedra corresponding to the leaf nodes. The flattening of the BVH tree structure circumvents the finite stack depth limitations of GPUs. This architectural adjustment ensures that the acceleration structures of more intricate phantoms can be handled efficiently by the GPU.

2.2 Procedural implementation of the transport algorithm on a GPU

2.2.1 Transport in the acceleration structure composed of numerous tetrahedra

Following the study outlined above, we constructed a tetrahedron mesh-type phantom into a flattened BVH acceleration structure to facilitate efficient particle transport, which is crucial for Monte Carlo simulations. The particle transport process within the acceleration structure is illustrated in Fig. 3. The effective execution of particle transport critically depends on two pivotal assessments, which are marked in red in Fig. 3: first, we determine whether the particles are positioned inside the confines of the tetrahedral model; second, we evaluate whether the external particles have trajectories that might intersect with the model. Both scenarios require iterations over all the tetrahedra. The traversal operation of a flattened tree generally requires the establishment of a stack structure, where nodes are placed in order in a stack, allowing depth-first traversal by first accessing the most recently added nodes. In our GPU program, this functionality was achieved using a predefined stack array with a capacity of 256.

When locating particles, the traversal begins from the root node, which is then pushed into the stack array. The top-level node is extracted from the stack array to determine whether it is a leaf node. If it is, then all the tetrahedra within the leaf node are traversed for positioning. Otherwise, it is assessed as to whether it lies within the bounding boxes of the left and right child nodes. If this is the case, the corresponding child nodes are pushed into the stack array. This process continues in loops until no nodes remain in the stack array. Intersection computation is similar, but the distinction lies in the fact that positioning only requires finding the tetrahedron in which the particle is located, whereas intersection computation requires traversing all intersecting nodes to identify the closest distance.

Particle transport within an acceleration structure composed of numerous tetrahedra begins with source sampling to generate the primary particles. The particle sampling process can accommodate various types of external irradiation sources, including parallel beams (e.g., anterior-posterior (AP) and posterior-anterior (PA)) and divergent beams (e.g., isotropic point sources), and can perform energy spectrum sampling for polyenergetic sources. If the primary particle is located within a specific tetrahedron, the tetrahedron's identifier is retrieved.

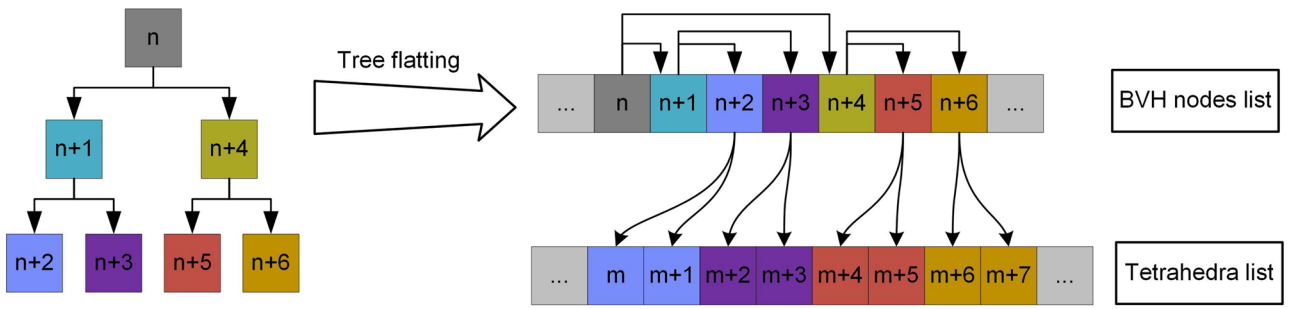


Fig. 2 (Color online) Flattening of the BVH tree, with a simplified BVH tree structure on the left and the two resulting linear lists on the right

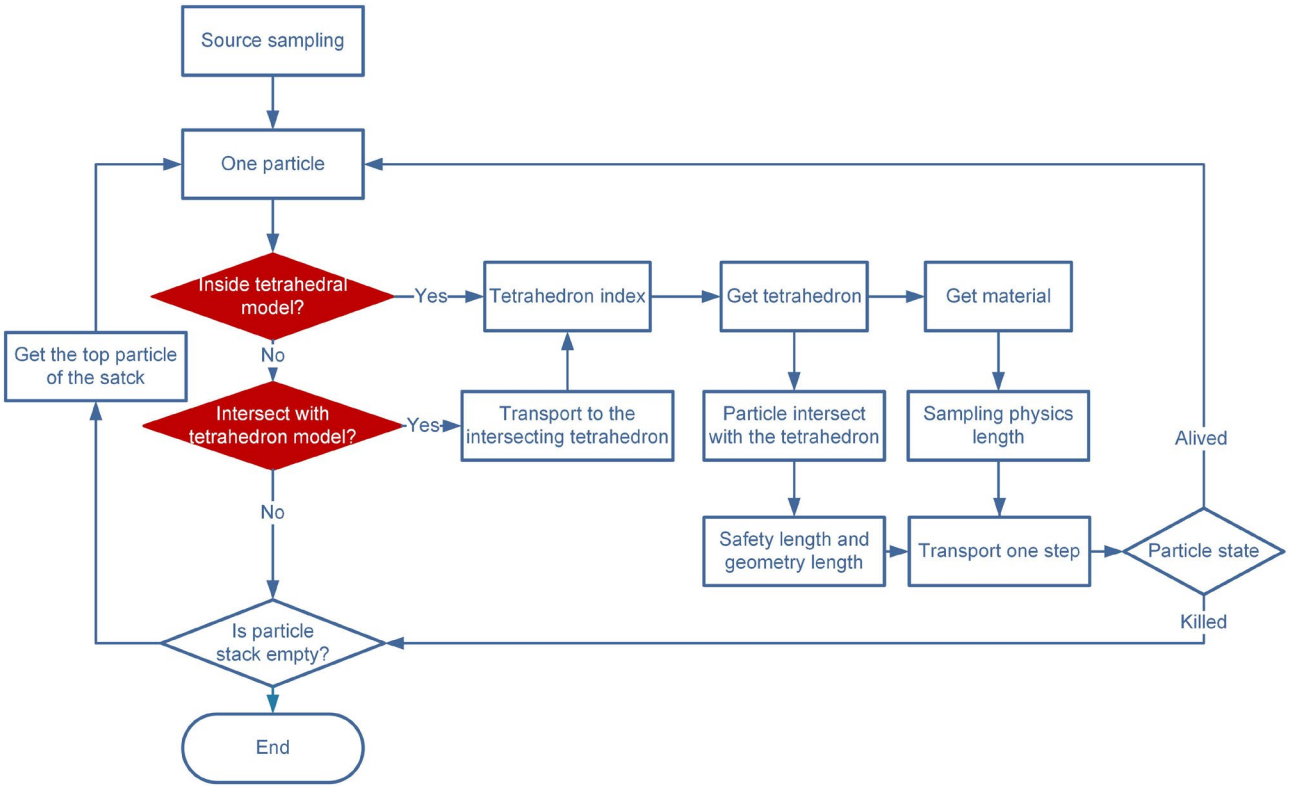


Fig. 3 (Color online) Particle transport within the BVH acceleration structure composed of numerous tetrahedra

Otherwise, the algorithm assesses whether the particle intersects the tree along its trajectory. Absent an intersection, the particle is immediately terminated. Conversely, if an intersection occurs, the particle is directed to the intersecting tetrahedron and the index of this tetrahedron is acquired. The geometry and material properties of the current tetrahedron are ascertained based on the sequence of tetrahedra and the tetrahedral index. By determining the particle's relative position to the tetrahedron, the minimum distance to the tetrahedron, which is defined as the safety distance, and the geometric distance from the particle to the tetrahedron surface along its path of motion are calculated. The mean free path is determined based on the elemental composition and concentration of the tetrahedron's material from which the physical step length is sampled. Subsequently, the transport method is determined to be either geometric transport or particle interaction, and any generated secondary particles are added to the particle stack array. The particle state is evaluated after a transport step. The primary particle is iteratively processed until termination, at which point the top secondary particle from the stack is selected. The above steps are repeated until the particle stack is empty, signifying the end of transport.

2.2.2 Physical models of the GPU program

Given the scope of applicability of the photon–electron coupled transport Monte Carlo program realized in this study, particularly considering the calculation of human radiation doses, judicious selection of appropriate physical models is essential. Our methodology was designed to prioritize the most impactful physical interactions that significantly influenced the outcomes of our simulations within the scope and specific applications of our research. In this context, the decision to exclude the coherent scattering of photons from our simulations was made after careful consideration. Coherent scattering, although relevant in certain scenarios, was deemed to have a negligible impact on the outcomes that we intended to investigate, thereby allowing us to streamline our computational efforts and focus on more critical interactions. We also addressed the scattering processes of electrons using a multiple scattering approach [16, 21, 34]. This approach leverages the condensed-history technique, which is instrumental in enhancing the efficiency and accuracy of our simulations by reducing the computational burden associated with tracking individual scattering events.

The program primarily investigates the physical process of photon–electron coupled transport. For clarity and

Table 1 Physical models of photons and electrons

Physical process	Physical model	Energy range
Photoelectric effect	Livermore	100 eV~100 MeV
Compton effect	Livermore	1.02 MeV~100 MeV
Electron pair production	Livermore	100 eV~100 MeV
Bremsstrahlung	Seltzer-Berger	1 keV~100 MeV
Ionization	Moller-Bhabha	100 eV~100 MeV
Multiplescattering	Urban	100 eV~100 MeV
Positrons annihilation	Eplus2gg	100 eV~100 MeV

detailed reference, Table 1 outlines the specific physical models we employed for each of these processes, along with the corresponding energy ranges in which they were considered effective. In particular, the photon physical processes predominantly draw upon the Livermore low-energy model provided by Geant4. In contrast to the parameterized calculations prevalent in the low-energy domain of Geant4's other standard physics models, the Livermore low-energy model directly leverages the nuclear shell reaction cross-sectional information, resulting in heightened precision and a lower energy threshold for photon simulation, with an energy floor

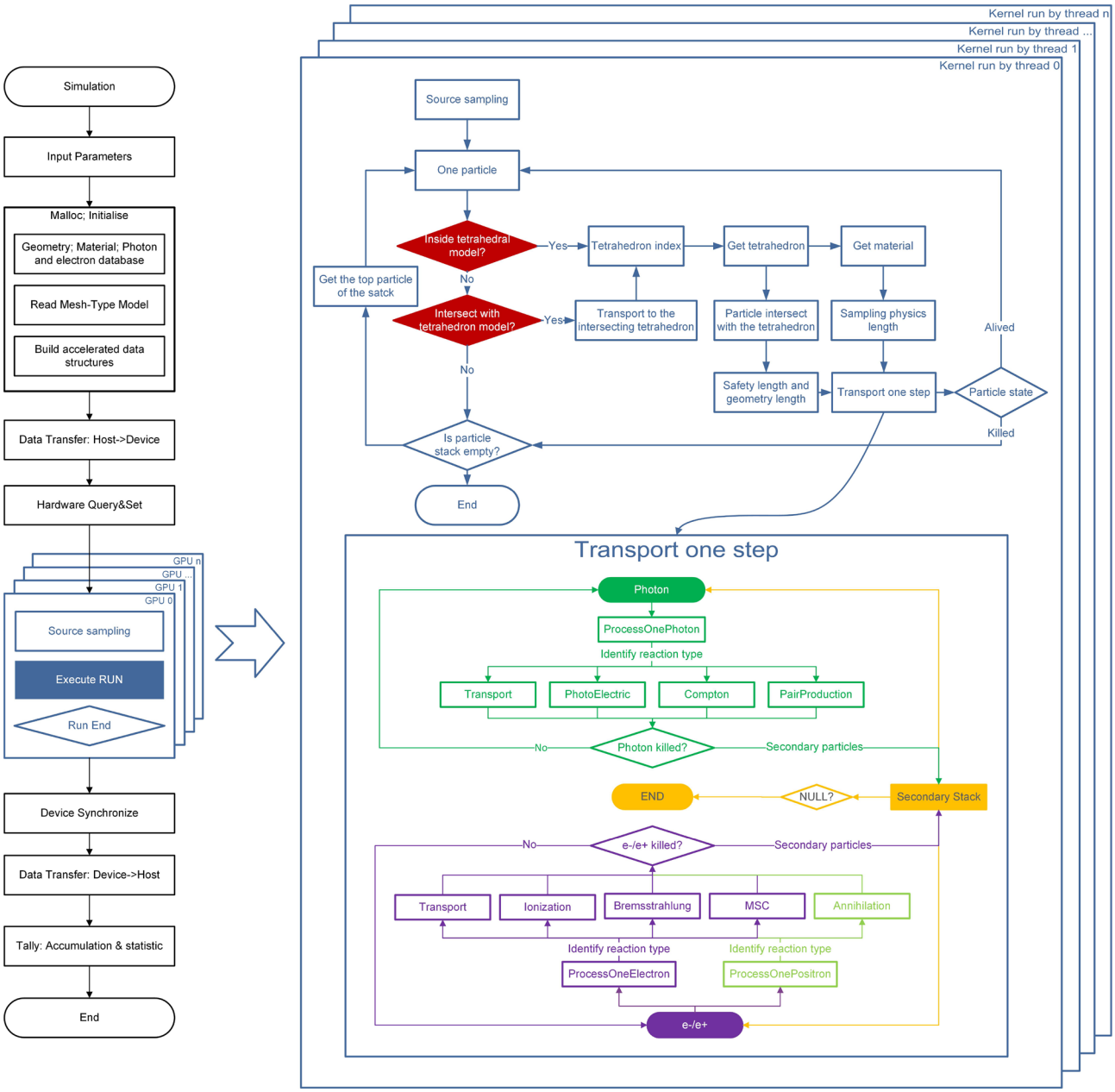


Fig. 4 (Color online) Framework of the GPU-accelerated Monte Carlo program implementation process. Black indicates operations executed on the CPU, whereas the other colors represent processes running on the GPU. The right side illustrates the particle transport flow and the transport-one-step operations executed on the GPU

as low as 100 eV within the scope of isotopes spanning Z from 1 to 99 [16]. Meanwhile, the multiple scattering model primarily incorporates an urban physical model based on the Lewis theory [35].

2.2.3 Program implementation flow

Drawing upon the particle transport method and physical model previously discussed, and utilizing the widely embraced Monte Carlo technique as a reference, the transport of photons and electrons was actualized within the GPU domain. The program implementation flow is shown in Fig. 4. The detailed process is further explained by pseudocode in the supplementary materials, which are openly available in the Science Data Bank at https://doi.org/10.57760/sciencedb.28433.

The implementation process begins by allocating memory space to the GPU, which is a critical step that ensures that sufficient resources are available for the subsequent complex computations. Initially, the data crucial for the simulation, such as particle properties and initial conditions, are prepared on the CPU. These data are then transferred to the GPU using compute unified device architecture (CUDA) data transfer functions, which are specifically designed to handle the efficient movement of large datasets between the CPU and GPU. Subsequently, an appropriate configuration for the device kernel functions is selected, including the number of thread blocks and thread grids to be executed. Then, the kernel function for the transport of three particles (photons, positrons, and negatrons) is executed. These device functions can be invoked based on the type of particles within the incident or secondary particle stack.

During particle transport, it is essential to determine the location of each particle within its corresponding tetrahedron accurately. This process involves calculating both physical and geometric step lengths. The physical step length is determined by the particles' energies of the tetrahedral material, whereas the geometric step length is calculated in relation to the boundaries of the tetrahedron. These calculations are vital because they directly influence the nature of particle interactions and the subsequent reactions that occur within the simulation.

In addition to the primary transport processes, a secondary particle stack is established to manage the particles generated by the initial interactions. This stack operates on a last-in, first-out principle and is designed to handle up to 500 sary particle arrays per thread, which helps maintain an orderly and efficient simulation process. Because each particle interaction can produce additional secondary particles, they are systematically processed and transported in sequence, ensuring that all resultant particles are accounted for in the simulation. Finally, once all the particle interactions have been adequately simulated and the particle stack is depleted, the results are transferred back to the CPU for analytical processing.

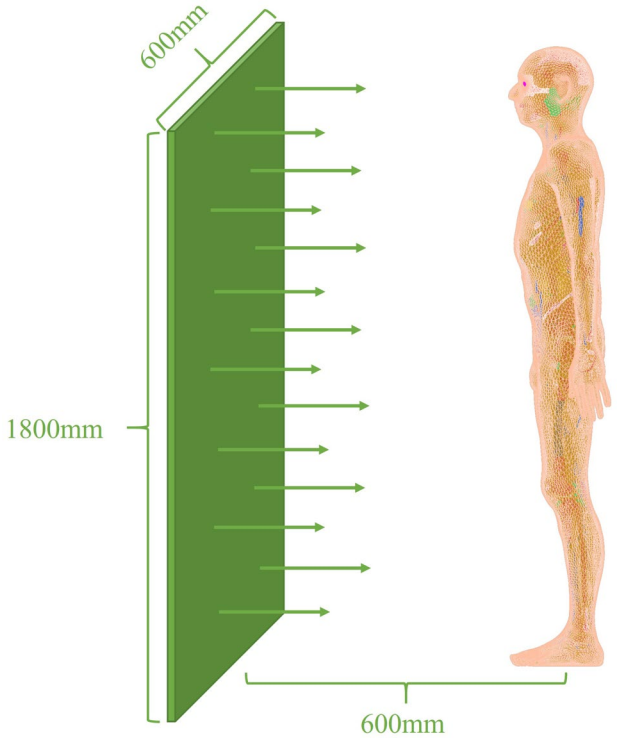
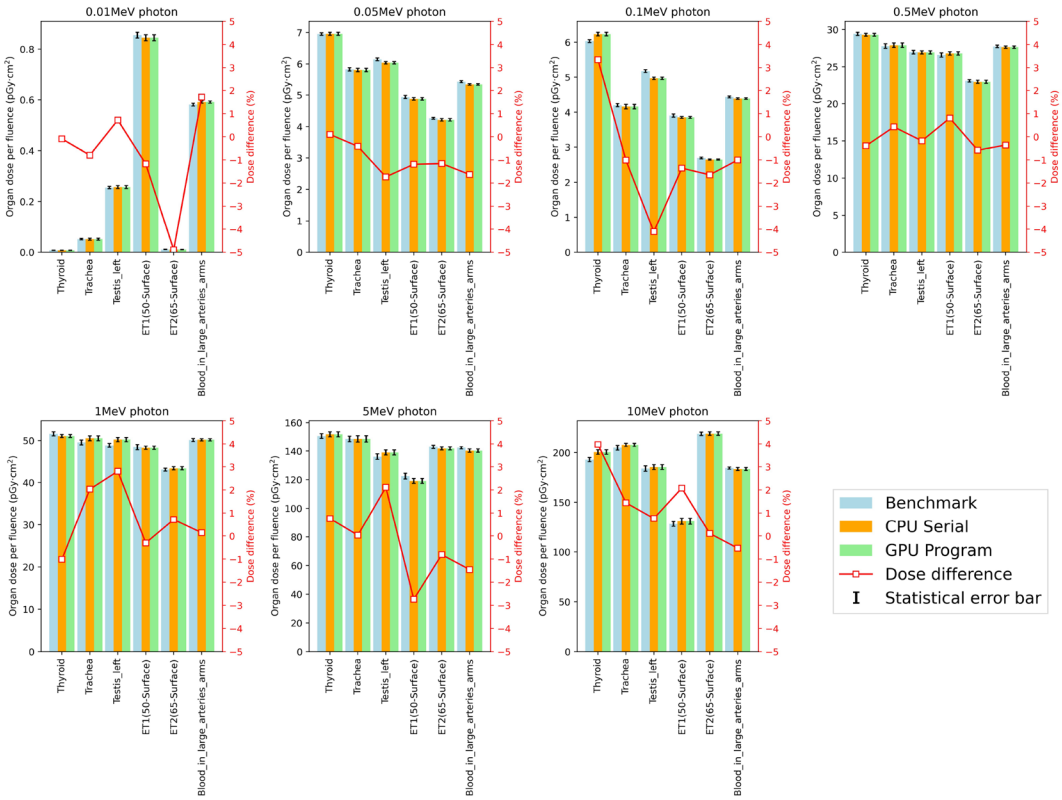


Fig. 5 (Color online) Schematic of anterior-–posterior irradiation (AP) geometries

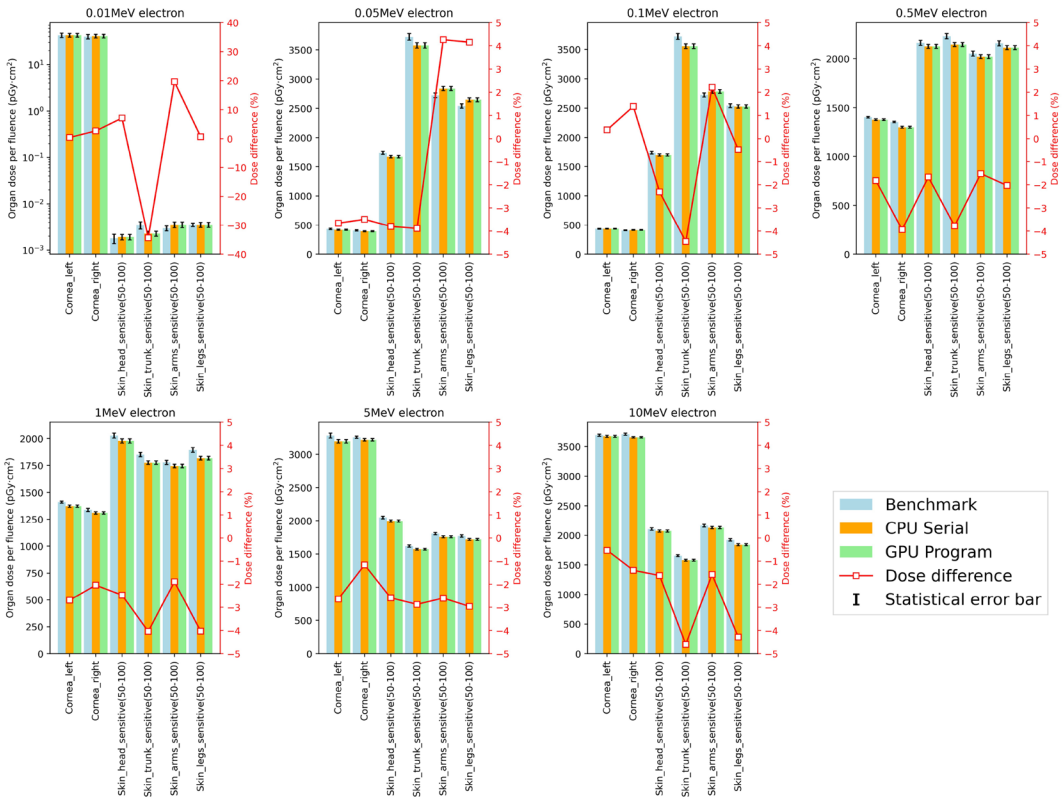
2.3 Dose calculations and efficiency assessments

To validate the accuracy of our GPU program and ascertain its acceleration efficiency, we simulated a standard external irradiation scenario. The MRCP phantom was exposed to unidirectional and parallel beams of photons and electrons originating from a planar source with dimensions of 600 mm × 1800 mm, located 600 mm from the center of the phantom. These beams converged from the anterior direction, as shown in Fig. 5 with energies of 0.01, 0.05, 0.1, 0.5, 1, 5, and 10 MeV.

Using Geant4 for simulations on the CPU platform served as a benchmark for the calculations. The source code for the Geant4 project employed in our research was derived from the supplemental material provided by ICRP report 145 [8]. Drawing on the Geant4 code for external exposure provided in the supplemental material, we adapted it for our specific source, physical models, and energy thresholds to perform our simulations. The computational framework operated on Geant4 version 10.04, incorporating the Livermore physics model, with secondary electron and photon energy thresholds set to 0.2 and 0.002 MeV, respectively.


0.01MeV photon
0.05MeV photon
0.1MeV photon
0.5MeV photon
1MeV photon
5MeV photon
10MeV photon
Organ dose per fluence (pGy·cm²)
Dose difference (%)
Thyroid
Trachea
Testis_left
ET1(50-Surface)
ET2(65-Surface)
Blood_in_large_arteries_arms
Benchmark
CPU Serial
GPU Program
Dose difference
Statistical error bar

(a)


0.01MeV electron
0.05MeV electron
0.1MeV electron
0.5MeV electron
1MeV electron
5MeV electron
10MeV electron
Organ dose per fluence (pGy·cm²)
Dose difference (%)
Cornea_left
Cornea_right
Skin_head_sensitive(50-100)
Skin_trunk_sensitive(50-100)
Skin_arms_sensitive(50-100)
Skin_legs_sensitive(50-100)
Benchmark
CPU Serial
GPU Program
Dose difference
Statistical error bar

(b)

◂**Fig. 6** (Color online) Comparison of organ doses per photon and electron fluence among the benchmark, CPU serial program, and GPU program using tetrahedral mesh-type phantom. The dose difference signifies the computational outcomes of the GPU program in relation to benchmark results, expressed as (GPU program—Benchmark)/Benchmark. Subfigure **a** displays results for photons, whereas subfigure **b** shows results for electrons

This careful selection struck a balance between ensuring dose accuracy and maintaining a reasonable computational speed. The simulation scenario was the standard AP external exposure, as described above. The CPU hardware used was an Intel(R) Xeon(R) CPU E5–2660 v4 @ 2.00GHz and the random access memory (RAM) was 64G.

The GPU program developed in this study utilized the physical models outlined in Sect. 2.2.2, maintaining consistency with the benchmark in terms of secondary electron and photon energy thresholds, as well as the irradiation scenarios. It was executed on an NVIDIA GeForce RTX 4090 GPU with 24GB of video random access memory (VRAM) and running on CUDA version 12.2. The objective was to compute organ doses under the same irradiation conditions, thereby assessing the acceleration performance of our GPU program.

To further explore the acceleration effects achieved through our methodology independent of hardware influences [22, 36, 37], we developed a serial CPU program derived from our GPU parallel program. This serial program replicated the physical models and acceleration structures used in its GPU counterpart by employing identical secondary particle cutoff parameters. It was executed on the same CPU hardware platform as the benchmark, simulating the same standard AP irradiation scenario. Our CPU serial program mirrored the GPU program in nearly every aspect, with the primary difference being the hardware platform on which it operated. This strategy enabled us to distinctly assess the acceleration effects resulting exclusively from the methodology itself, as well as the benefits derived specifically from the hardware capabilities.

The results of our simulation and subsequent data processing efforts were the dose conversion coefficients, which serve as pivotal factors for converting particle fluence into organ dose (see $\mathrm{Gy} \cdot \mathrm{cm}^2$) [38]. These coefficients were calculated precisely based on the ratio of the organ-absorbed dose to the particle fluence, thus facilitating a direct correlation between the measured particle fluence and the resulting dose absorbed by specific organs. Our calculations focused on the key organs that substantially affect the effective dose, particularly those with higher tissue-weighting factors, including the extrathoracic regions, trachea, thyroid, blood, and skin. The internal organ doses of electrons are generally low owing to their limited penetration ability. Therefore, we compared the dose results for surface organs such as the cornea and sensitive skin areas.

We conducted dose calculations for specific irradiation scenarios using three different software tools: the benchmark Geant4 program, CPU serial program, and GPU program. The objective of employing these various approaches was to evaluate both the accuracy of the dose calculations and the acceleration efficiency of the GPU program. Although the programs can execute Monte Carlo simulations using a single- or double-precision floating point, considering computational efficiency, we employed a single-precision floating point for the calculations. The simulations employed 10^8 particles to attain lower statistical accuracy. Within our GPU program, the statistical uncertainty was determined using a batch-based method [39], which involves conducting simulations over multiple batches of particles and evaluating the variance among these batches to calculate the uncertainty. Although the batch-based approach has some drawbacks [40], we opted for this method because of its high efficiency, which is particularly advantageous for GPU-based calculations.

3 Results and discussion

3.1 Comparison of calculated dose values

Figure 6 compares the organ-averaged absorbed doses per photon and electron fluence using the benchmark, CPU serial program, and GPU program, employing tetrahedral mesh phantoms for six specific organs across seven different photon and electron energies.

As illustrated in Fig. 6, the results from the CPU serial execution closely align with those from the GPU execution. This similarity is expected because the CPU serial program is a direct adaptation of the GPU's parallel program framework, utilizing identical physical models and transport methodologies. This direct adaptation ensures that the fundamental physics and computational principles remain consistent across both platforms, providing a reliable basis for comparing the simulation times on different platforms.

Moreover, the maximal dose relative differences between the GPU program and the benchmark remained within 5% for most scenarios, demonstrating the accuracy of our GPU program. However, an exception is noted in the case of the 0.01 MeV electron AP irradiation, where the GPU program's results displayed a notable deviation from the established benchmark. The substantial dose difference observed specifically at 0.01 MeV for electron exposure in the skin's most sensitive regions can be attributed to the limited penetration capability of such low-energy electrons. These electrons tend to deposit minimal energy within the critical 50–100μm sensitive zones, leading to lower absorbed doses. This results in

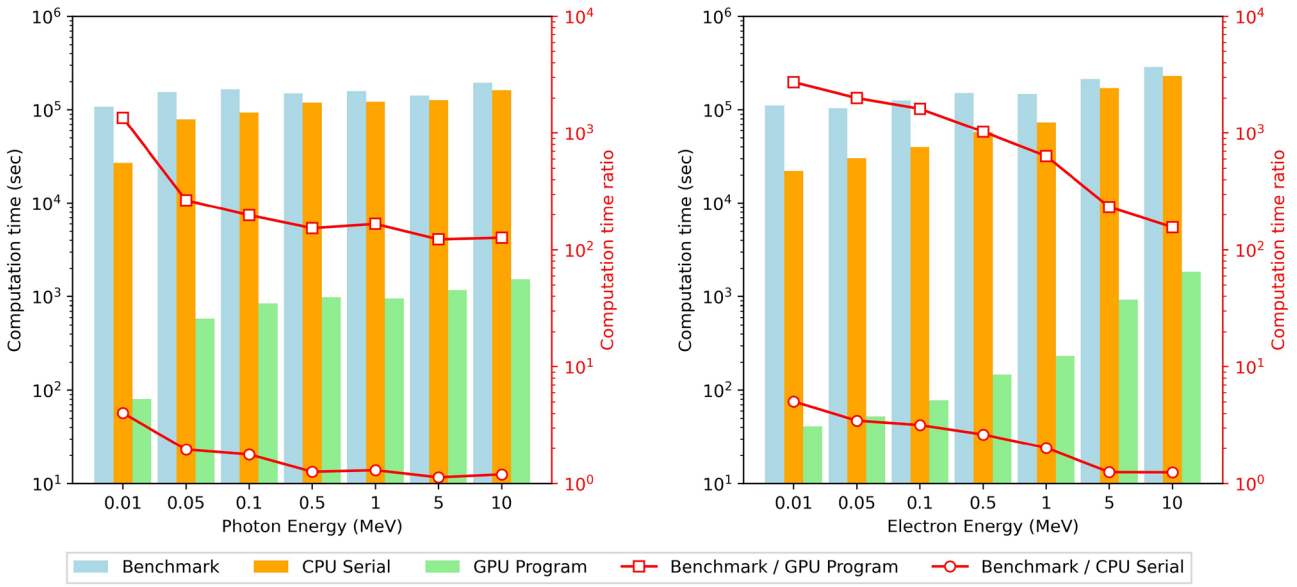


Fig. 7 (Color online) Computation time of mesh-type phantoms in the standard anterior-posterior (AP) irradiation conditions among the benchmark, CPU serial program, and GPU parallel program per simulation of 10^8 particles. The benchmark/GPU program is obtained by dividing the calculation time of the benchmark by the calculation time of the GPU program, whereas the benchmark/CPU serial is expressed as the division of the calculation time of the benchmark by the calculation time of the CPU serial program

larger statistical errors and more pronounced dose deviations from the expected values.

It is important to note that, as shown in Fig. 6, the relative dose differences for most scenarios were confined to within twice the range of the statistical error (indicated by the black horizontal bars in the histogram). This observation suggests that, apart from the noted exception, there is no substantial discrepancy in the dose values between the benchmark results and those derived from the GPU program. Overall, the findings underline the accuracy of the GPU-based Monte Carlo simulation program, which demonstrates its ability to produce dose calculations that are within acceptable statistical variances when compared with conventional CPU-based programs such as Geant4.

3.2 Comparison of computation speed

Figure 7 summarizes the computation times. This outcome indicates that the GPU program achieved considerable acceleration by employing tetrahedral models. For photons and electrons with energies ranging from 0.01 to 10 MeV, the GPU program exhibited an acceleration ratio, which refers to the ratio of the calculated time, for tetrahedral models between 120 and 2700 times compared with the benchmark simulated using Geant4 on a CPU with one thread, minimizing the computational duration from hours to seconds for simulating 10^7 particles.

By comparing the CPU serial and benchmark results, it was evident that the acceleration achieved through the utilization of acceleration structures and physical models was of the order of single digits. This implies that the enhancement in the computational speed attributed to the implementation of acceleration structures and physical models on the CPU side was marginal. The predominant contribution to the overall acceleration effect largely emanated from the utilization of the GPU.

Moreover, the improvement in acceleration was more significant for electrons than for photons. This is associated with thread divergence [41, 42]. In the GPU-based Monte Carlo simulations, when one particle finishes its path but others within the same thread group are still moving, the completed thread waits idly because of thread divergence, leading to inefficiencies in resource use. Considering the intrinsic characteristics of charged particles, electrons exhibit a higher propensity than photons for interactions within the material medium. This propensity for interaction results in shorter ranges and transport distances for electrons, and the variation in transport times between different electrons is relatively smaller. Consequently, thread divergence was reduced, thereby amplifying the acceleration effect.

Similarly, the illustration shows that as the energy of the particles increases, the acceleration efficiency decreases. This is because the transport times for

low-energy particles are similar. With less thread divergence among these particles, the acceleration effect is enhanced.

4 Conclusion

This study successfully accomplished GPU-accelerated dose calculation of mesh-type models for the first time, providing an expeditious and accurate tool for dose calculation in mesh-type computational phantoms. These swift calculations of mesh-type phantoms are particularly beneficial in scenarios such as radiation treatments, unforeseen radiation incidents, occupational exposures, and individual dosimetry assessments, where finely detailed and personalized phantoms, along with faster computational methods, are critical. This advancement facilitates more precise radiation dose assessments, which are crucial for effective and safe medical practice.

The GPU-accelerated Monte Carlo program employed an acceleration structure coupled with the optimization of traversal processes, realizing particle transport within tetrahedral models. Addressing typical scenarios of external photon and electron AP irradiation, most organ dose relative differences remained within 5%. Compared with the benchmark simulated by Geant4 on a CPU with a single thread, the computational speed increased by a factor ranging from 120 to 2700, thereby reducing the time required to simulate 10^7 particles from hours to seconds. A minor portion of the acceleration effect arises from the acceleration structures and varied physical models, whereas the majority stems from the utilization of GPUs. Future research will focus on enhancing the efficiency of particle transport simulations to achieve real-time dose calculations for mesh-type phantoms. Key improvements will involve individualizing phantoms based on specific exposure scenarios and optimizing GPU Monte Carlo programs to further reduce GPU thread divergence.

Author contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Shu-Chang Yan, Rui Qiu, Xi-Yu Luo, An-Kang Hu, Zhen Wu and Jun-Li Li. The first draft of the manuscript was written by Shu-Chang Yan and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Data availability The data that support the findings of this study are openly available in Science Data Bank at https://cstr.cn/31253.11.sciencedb.28433 and https://www.doi.org/10.57760/sciencedb.28433.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

References

1. H. Liu, J. Li, R. Qiu et al., Dose conversion coefficients for Chinese reference adult male and female voxel phantoms from idealized neutron exposures. J. Nucl. Sci. Tech. **54**, 921–932 (2017). https://doi.org/10.1080/00223131.2017.1323685
2. Y. Wu, M. Cheng, W. Wang et al., Development of Chinese female computational phantom rad-human and its application in radiation dosimetry assessment. Nucl. Technol. **201**, 155–164 (2018). https://doi.org/10.1080/00295450.2017.1411717
3. H.G. Menzel, C. Clement, P. DeLuca, ICRP Publication 110. Realistic reference phantoms: an ICRP/ICRU joint effort. A report of adult reference computational phantoms. Ann. ICRP **39**, 1–164 (2009). https://doi.org/10.1016/j.icrp.2009.09.001
4. A. Alfuraih, O. Kadri, K. Alzimami, Investigation of SPECT/CT cardiac imaging using Geant4. Nucl. Sci. Tech. **29**, 105 (2018). https://doi.org/10.1007/s41365-018-0435-8
5. R. Qiu, J.L. Li, S.M. Huang et al., Organ dose conversion coefficients for external neutron irradiation based on the Chinese mathematical phantom (CMP). J. Nucl. Sci. Technol. **49**, 263–271 (2012). https://doi.org/10.1080/00223131.2011.649081
6. M.E. Miyombo, Y.K. Liu, A. Ayodeji, An analytical approach to organ equivalent dose estimation based on material composition photon attenuation coefficient. Prog. Nucl. Energy **141**, 103955 (2021). https://doi.org/10.1016/j.pnucene.2021.103955
7. X.Y. Luo, R. Qiu, Z. Wu et al., THUDose^{PD}: a three-dimensional Monte Carlo platform for phantom dose assessment. Nucl. Sci. Tech. **34**, 164 (2023). https://doi.org/10.1007/s41365-023-01315-y
8. C.H. Kim, Y.S. Yeom, N. Petoussi-Henss et al., ICRP Publication 145: Adult Mesh-Type Reference Computational Phantoms. Ann. ICRP **49**, 13–201 (2020). https://doi.org/10.1177/0146645319893605
9. Y. Liu, T.W. Xie, Q. Liu, Monte Carlo simulation for internal radiation dosimetry based on the high resolution Visible Chinese Human. Nucl. Sci. Tech. **22**, 165–173 (2011)
10. K. Zhao, M.Y. Cheng, P.C. Long et al., A hybrid voxel sampling method for constructing Rad-HUMAN phantom. Nucl. Sci. Tech. **25**, 020503 (2014). https://doi.org/10.13538/j.1001-8042/nst.25.020503
11. W.P. Segars, B.M. Tsui, D.S. Lalush et al., Development and application of the new dynamic Nurbs-based Cardiac-Torso (NCAT) phantom. J. Nucl. Med. **42**, 7P-7P (2001)
12. Y.S. Yeom, M.C. Han, C.H. Kim et al., Conversion of ICRP male reference phantom to polygon-surface phantom. Phys. Med. Biol. **58**, 6985–7007 (2013). https://doi.org/10.1088/0031-9155/58/19/6985
13. C. Choi, T.T. Nguyen, Y.S. Yeom et al., Mesh-type reference Korean phantoms (MRKPs) for adult male and female for use in radiation protection dosimetry. Phys. Med. Biol. **64**, 085020 (2019). https://doi.org/10.1088/1361-6560/ab0b59
14. C. Cumur, T. Fujibuchi, K. Hamada, Dose estimation for cone-beam computed tomography in image-guided radiation therapy using mesh-type reference computational phantoms and assuming head and neck cancer. J. Radiol. Prot. **42**, 021533 (2022). https://doi.org/10.1088/1361-6498/ac7914
15. M. Bhar, O. Kadri, K. Manai, Assessment of self- and cross-absorbed SAF values for HDRK-man using Geant4 code: internal photon and electron emitters. Nucl. Sci. Tech. **30**, 149 (2019). https://doi.org/10.1007/s41365-019-0675-2
16. S. Agostinelli, J. Allison, K. Amako et al., GEANT4-a simulation toolkit. Nucl. Instrum. Methods Phys. Res. Sect. A-Accel. Spectrom. Dect. Assoc. Equip. **506**, 250–303 (2003). https://doi.org/10.1016/s0168-9002(03)01368-8

17. Z.F. Luo, R. Qiu, M. Li et al., Study of a GPU-based parallel computing method for the Monte Carlo program. Nucl. Sci. Tech. **25**, S010501 (2014). https://doi.org/10.13538/j.1001-8042/nst.25.S010501
18. C.H. Kim, J.H. Jeong, W.E. Bolch et al., A polygon-surface reference Korean male phantom (PSRK-Man) and its direct implementation in Geant4 Monte Carlo simulation. Phys. Med. Biol. **56**, 3137–3161 (2011). https://doi.org/10.1088/0031-9155/56/10/016
19. Y.S. Yeom, J.H. Jeong, M.C. Han et al., Tetrahedral-mesh-based computational human phantom for fast Monte Carlo dose calculations. Phys. Med. Biol. **59**, 3173–3185 (2014). https://doi.org/10.1088/0031-9155/59/12/3173
20. Z. Peng, Y. Lu, Y. Xu et al., Development of a GPU-accelerated Monte Carlo dose calculation module for nuclear medicine, ARCHER-NM: demonstration for a PET/CT imaging procedure. Phys. Med. Biol. **67**, 06NT02 (2022). https://doi.org/10.1088/1361-6560/ac58dd
21. L. Su, Y.M. Yang, B. Bednarz et al., ARCHERRT - A GPU-based and photon-electron coupled Monte Carlo dose computing engine for radiation therapy: Software development and application to helical tomotherapy. Med. Phys. **41**, 071709 (2014). https://doi.org/10.1118/1.4884229
22. A.K. Hu, R. Qiu, H. Liu et al., THUBrachy: fast Monte Carlo dose calculation tool accelerated by heterogeneous hardware for high-dose-rate brachytherapy. Nucl. Sci. Tech. **32**, 32 (2021). https://doi.org/10.1007/s41365-021-00866-2
23. W.G. Li, C. Chang, Y. Qin et al., GPU-based cross-platform Monte Carlo proton dose calculation engine in the framework of Taichi. Nucl. Sci. Tech. **34**, 77 (2023). https://doi.org/10.1007/s41365-023-01218-y
24. X. Jia, X.J. Gu, J. Sempau et al., Development of a GPU-based Monte Carlo dose calculation code for coupled electron-photon transport. Phys. Med. Biol. **55**, 3077–3086 (2010). https://doi.org/10.1088/0031-9155/55/11/006
25. Y. Li, W. Sun, H. Liu et al., Development of a GPU-superposition Monte Carlo code for fast dose calculation in magnetic fields. Phys. Med. Biol. **67**, 125002 (2022). https://doi.org/10.1088/1361-6560/ac7194
26. G. Franciosini, G. Battistoni, A. Cerqua et al., GPU-accelerated Monte Carlo simulation of electron and photon interactions for radiotherapy applications. Phys. Med. Biol. **68**, 044001 (2023). https://doi.org/10.1088/1361-6560/aca1f2
27. K. Yang, B.S. He, Q.O. Luo et al., Stack-based parallel recursion on graphics processors. ACM SIGPLAN Not. **44**, 299–300 (2009). https://doi.org/10.1145/1594835.1504224
28. L. Deng, G. Li, T. Ye et al., MCDB Monte Carlo code with fast track technique and mesh tally matrix for BNCT. J. Nucl. Sci. Technol. **44**, 1518–1525 (2007). https://doi.org/10.1080/18811248.2007.9711401
29. A. Aman, S. Demirci, U. Güdükbay et al., Multi-level tetrahedralization-based accelerator for ray-tracing animated scenes. Comput. Anim. Virtual Worlds **32**, e2024 (2021). https://doi.org/10.1002/cav.2024
30. X. Wang, Y. Yu, X. Li et al., Development of a hybrid parallelism Monte Carlo transport middleware on mesh geometry. Ann. Nucl. Energy **190**, 109872 (2023). https://doi.org/10.1016/j.anucene.2023.109872
31. M. Vinkler, V. Havran, J. Sochor, Visibility driven BVH build up algorithm for ray tracing. Comput. Graph. **36**, 283–296 (2012). https://doi.org/10.1016/j.cag.2012.02.013
32. J.D. MacDonald, K.S. Booth, Heuristics for ray tracing using space subdivision. Vis. Comput. **6**, 153–166 (1990). https://doi.org/10.1007/bf01911006
33. W. Boukaram, G. Turkiyyah, D. Keyes, Hierarchical matrix operations on GPUs: matrix-vector multiplication and compression. ACM Trans. Math. Softw. **45**, 3 (2019). https://doi.org/10.1145/3232850
34. Z. Tian, F. Shi, M. Folkerts et al., A GPU OpenCL based cross-platform Monte Carlo dose calculation engine (goMC). Phys. Med. Biol. **60**, 7419–7435 (2015). https://doi.org/10.1088/0031-9155/60/19/7419
35. V.N. Ivanchenko, O. Kadri, M. Maire et al., Geant4 models for simulation of multiple scattering. J. Phys. Conf. Ser. **219**, 032045 (2010). https://doi.org/10.1088/1742-6596/219/3/032045
36. T. Liu, X.G. Xu, C.D. Carothers, Comparison of two accelerators for Monte Carlo radiation transport calculations, Nvidia Tesla M2090 GPU and Intel Xeon Phi 5110p coprocessor: A case study for X-ray CT imaging dose calculation. Ann. Nucl. Energy **82**, 230–239 (2015). https://doi.org/10.1016/j.anucene.2014.08.061
37. Y. Wang, J. Liang, Q. Zhang et al., Development and verification of Geant4-based parallel computing Monte Carlo simulations for nuclear logging applications. Ann. Nucl. Energy **172**, 109079 (2022). https://doi.org/10.1016/j.anucene.2022.109079
38. N. Petoussi-Henss, W.E. Bolch, K.F. Eckerman et al., ICRP Publication 116. Conversion coefficients for radiological protection quantities for external radiation exposures. Ann. ICRP **40**, 1–257 (2010). https://doi.org/10.1016/j.icrp.2011.10.001
39. H. Lee, J. Shin, J.M. Verburg et al., MOQUI: an open-source GPU-based Monte Carlo code for proton dose calculation with efficient data structure. Phys. Med. Biol. **67**, 174001 (2022). https://doi.org/10.1088/1361-6560/ac8716
40. B.R.B. Walters, I. Kawrakow, D.W.O. Rogers, History by history statistical estimators in the BEAM code system. Med. Phys. **29**, 2745–2752 (2002). https://doi.org/10.1118/1.1517611
41. S.P. Hamilton, T.M. Evans, Continuous-energy Monte Carlo neutron transport on GPUs in the Shift code. Ann. Nucl. Energy **128**, 236–247 (2019). https://doi.org/10.1016/j.anucene.2019.01.012
42. S.P. Hamilton, S.R. Slattery T.M. Evans, Comparison of history- and event-based algorithms: Multigroup Monte Carlo on GPUs. Ann. Nucl. Energy **113**, 506–518 (2018). https://doi.org/10.1016/j.anucene.2017.11.032