NUCL SCI TECH (2021) 32:32
https://doi.org/10.1007/s41365-021-00866-2

THUBrachy: fast Monte Carlo dose calculation tool accelerated by heterogeneous hardware for high-dose-rate brachytherapy

An-Kang Hu[1,2] · Rui Qiu[1,2] · Huan Liu[1,2] · Zhen Wu[1,3] · Chun-Yan Li[1,3] · Hui Zhang[1,2] · Jun-Li Li[1,2] · Rui-Jie Yang[4]

Received: 21 August 2020 / Revised: 21 December 2020 / Accepted: 5 January 2021 / Published online: 20 March 2021

Abstract The Monte Carlo (MC) simulation is regarded as the gold standard for dose calculation in brachytherapy, but it consumes a large amount of computing resources. The development of heterogeneous computing makes it possible to substantially accelerate calculations with hardware accelerators. Accordingly, this study develops a fast MC tool, called THUBrachy, which can be accelerated by several types of hardware accelerators. THUBrachy can simulate photons with energy less than 3 MeV and considers all photon interactions in the energy range. It was benchmarked against the American Association of Physicists in Medicine Task Group No. 43 Report using a water phantom and validated with Geant4 using a clinical case. A performance test was conducted using the clinical case, showing that a multicore central processing unit, Intel Xeon Phi, and graphics processing unit (GPU) can efficiently accelerate the simulation. GPU-accelerated THUBrachy is the fastest version, which is 200 times faster than the serial version and approximately 500 times faster than Geant4. The proposed tool shows great potential for fast and accurate dose calculations in clinical applications.

This work was supported by the National Natural Science Foundation of China (No. 11875036).

✉ 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

[4] Department of Radiation Oncology, Peking University Third Hospital, Beijing 100191, China

Keywords High-dose-rate brachytherapy · Monte Carlo · Heterogeneous computing · Hardware accelerators

1 Introduction

High-dose-rate (HDR) brachytherapy is widely used in the treatments of cancers, such as cervical cancer, breast cancer, and prostate cancer [1]. It aims to deliver planned doses to target areas and control doses on other organs and tissues as low as possible via treatment planning. Dose calculation is therefore a critical part of a treatment planning system (TPS) to guarantee accurate dose delivery.

The standard clinical TPS currently uses a dose calculation method based on the American Association of Physicists in Medicine Task Group No. 43 Report (TG-43) [2–4]. The TG-43 parameters are calculated via a Monte Carlo (MC) simulation or measurements under the condition where the source is placed in an infinite water medium. The dose distribution obtained via the TG-43 method is different from the results of the MC simulation by as much as 15% for the skin and approximately 15% for the bone [5]. For these reasons, the dose calculation method is becoming patient based and necessitates more accurate techniques [6,[7].

The MC simulation is often regarded as the gold standard for dose calculation [8], considering that it takes almost all physics, geometry, and material factors into account. However, a precise MC simulation consumes a large amount of computing resources. For personal computers or workstations, calculating the dose distribution using a general-purpose MC code would take tens of minutes, which is unacceptable for clinical applications.

With the rapid development of high-performance computing, heterogeneous computing is becoming a trend for supercomputers [9]. Heterogeneous computing is the use of hardware accelerators with architectures that are different from those of the central processing unit (CPU). Examples of these accelerators include the graphics processing unit (GPU), Intel Xeon Phi, Matrix-2000 [10], and SW26010 [9]. Many parallel programming models have been developed to provide programming tools (e.g., OpenMP, CUDA, OpenCL, and OpenACC) so that developers can make full use of these accelerators. By cutting down the complex control function of processors and strengthening their computational function, these hardware accelerators provide a high computing speed and low power consumption to solve some computational problems. This modification efficiently speeds up the code that deals with a heavy computational load and a light control load. In the MC simulation for brachytherapy and other conditions, the physics processes of photons are relatively simple as the maximum photon energy is below 3 MeV. The geometry of a patient is a simple three-dimensional (3D) voxel mesh [11]. Therefore, heterogeneous computing using hardware accelerators can potentially be used to accelerate MC simulations in brachytherapy. The GPU is not the only choice for code acceleration using a hardware accelerator.

Several groups have developed GPU-accelerated MC codes for the dose calculation for radiotherapy. bGPUMCD, a GPU-based fast MC code, has been developed for brachytherapy with a dose calculation method using the track-length estimator [12]. Jia et al. developed another GPU-based MC code for brachytherapy using the Woodcock photon transport method [13]. ARCHER$_{RT}$, a fast MC code for radiotherapy, which is based on GPU, was developed by Xu et al. [14]. All these codes are substantially accelerated by the GPU. However, hardware accelerators other than GPUs are rarely used in the acceleration of dose calculation for brachytherapy.

Accordingly, this study develops a fast MC dose distribution tool, namely HUBrachy, based on heterogeneous computing devices for HDR brachytherapy. This tool is one of the first tools that can use several types of existing hardware accelerators to speed up the simulation. It also has the potential to be used in merging hardware accelerators in the future.

2 Materials and methods

This section describes in detail the processes for the development of THUBrachy. The structure of THUBrachy is shown in Fig. 1. THUBrachy contains pretreatment modules in the preparation stage and memory management modules for parallel computing. All parts of the MC simulation are listed in a time order. Because of the complex memory structure of hardware accelerators, THUBrachy contains a memory management module to deal with data exchange between the main memory and the accelerator's memory. All modules in the diagram and code process are explained in detail below.

2.1 Physics model

2.1.1 Photon transport and interactions

THUBrachy is a special-purpose MC tool developed for brachytherapy. Because the energy values of all photons emitted by clinically used sources are below 3 MeV, the energy range of interest is set to 1 keV–3 MeV.

The photon transport for a heterogeneous medium is more complex than that for a uniform medium. The Woodcock tracking method is an efficient way of transporting photons in a heterogeneous medium. The efficiency of the Woodcock method is determined by the ratio of the average cross section to the maximum cross section. Thus, if there were some areas of the cross sections that were extremely higher than those of the average cross section, the efficiency of the transport would be very low. From another aspect, the Woodcock method cannot directly calculate the fluence in each voxel. Therefore, THUBrachy mainly adopts the direct transport method, which calculates the intersection point for each passing voxel to directly sample the range. The track length and fluence in each voxel were directly recorded.

For the energy range of interest, four types of interactions of photons (i.e., photoelectric absorption, Compton scattering, electron pair production, and Rayleigh scattering) are taken into account. Some complex and unimportant effects are ignored for this energy range. Photoelectric absorption is modeled by ignoring the effect of atomic relaxation and deducting the photoelectric part weight of the photon according to the cross section to avoid deviation instead of killing the photon. The weight of the photon after interaction is calculated by Eq. (1):

$$w = w_0 \cdot \frac{\Sigma_{\text{ph}}}{\Sigma_{\text{total}}}, \tag{1}$$

where w is the weight of the photon after interaction, w_0 is the weight of the photon before interaction, Σ_{ph} is the cross section of photoelectric absorption, and Σ_{total} is the total cross section. This method reduces the branches of code and permits photon transport to further regions, which results in variance reduction.

For pair production, electrons and positrons are not simulated; however, the original photon is replaced with a pair of 511 keV annihilation photons generated from the positron.

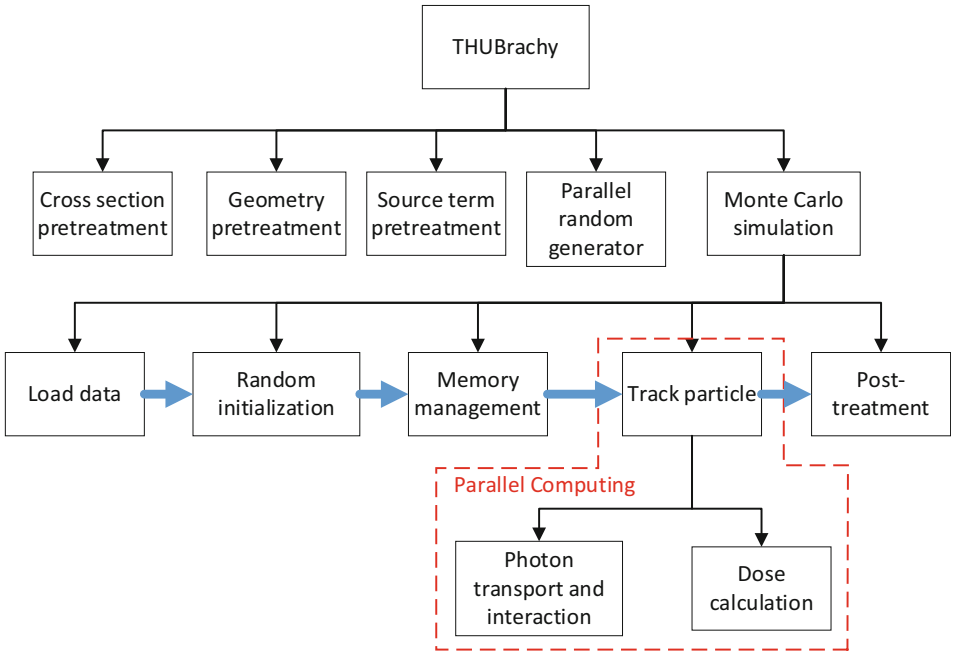

Fig. 1 Code structure and key modules of THUBrachy

Compton scattering is simulated according to the Penelope model, which takes into account the atomic binding effects and Doppler broadening. The scattered photon is sampled via the differential cross section of Compton scattering described in Eq. (2) [15].

$$\frac{\mathrm{d}\sigma(E)}{\mathrm{d}\Omega} = \frac{r_\mathrm{e}^2}{2}\frac{E_\mathrm{C}^2}{E^2}\left(\frac{E_\mathrm{C}}{E}+\frac{E}{E_\mathrm{C}}-\sin^2\theta\right) \times \sum_{\text{shells}} f_i \Theta(E-U_i) n_i(p_z^{\max}), \tag{2}$$

where r_e is the classical radius of the electron, θ is the scattering angle, E_C is the Compton energy, E is the energy of the photon, f_i is the number of electrons in the ith atomic shell, U_i is the ionization energy of the i-th atomic shell, p_z is the projection of the initial momentum of the electron in the direction of the scattering angle, $p_z^{\max}$ is the highest possible of p_z, and n_i is a function based on the Hartree–Fock atomic orbitals.

Rayleigh scattering is processed to sample the final state of the scattered photon with the differential cross section described in Eq. (3) [16].

$$\frac{\mathrm{d}\sigma(E)}{\mathrm{d}\Omega} = \frac{1+\cos^2\theta}{2}\left[F\left(2\frac{E}{c}\sin\left(\frac{\theta}{2}\right), Z\right)\right]^2, \tag{3}$$

where E is the photon energy, θ is the scattering angle, F is the atomic form factor related to energy, and Z is the atomic number.

The secondary electron generated in each type of interaction was not simulated. To ensure that each scattering has been correctly modeled, the sampled scattering-angle distribution and energy distribution were compared with the analytical results.

2.1.2 Dose calculation

The dose calculation is related to the transport method. For the Woodcock method and direct transport method, the absorbed dose is approximated by kerma because the secondary electron is ignored. However, for the direct transport, this dose calculation method is inefficient. A linear track length estimator is used to reduce the number of simulated photons while maintaining a low statistical uncertainty. Because the track length in each voxel can be directly recorded when the direct transport method is used, the dose in each voxel was calculated using Eq. (4).

$$D = \frac{\mu_\mathrm{en}}{\rho}\cdot E\cdot\varphi = \frac{\mu_\mathrm{en}}{\rho}\cdot E\cdot\frac{l_\mathrm{track}}{V}, \tag{4}$$

where D is the dose in the voxel, μ_en/ρ is the mass- energy absorption coefficient, φ is the fluence in the voxel, E is the photon energy, l_track is the track length in the voxel, and V is the volume of the voxel.

With this method, every track in the voxel crossed by the photon contributes to the dose calculation. The variance of the result is significantly lower than that of the kerma approximation for the same number of tracked particles. The direct transport along with a track-length dose estimator is the default method used in THUBrachy.

2.2 Key modules of THUBrachy

2.2.1 Pretreatment modules

Fast MC code requires an accurate and fast cross-sectional data calculation method. THUBrachy adopts the

NIST photon cross-section database XCOM [17]. THUBrachy is especially used for photon energies lower than 3 MeV and requires a high calculation speed. A lookup table at 0.1 keV energy resolution is generated via a log–log cubic spline interpolation of basic data. The cross-sectional value for a determined energy point is simply calculated via linear interpolation using this pre-calculated lookup table during the simulation.

For a clinical brachytherapy treatment planning, information on the patient is obtained via computerized tomography (CT) scanning. The geometry of the patient is constructed according to the CT images, generating a voxel phantom. The density of each voxel is determined by converting the Hounsfield unit (HU) value to the density using a conversion curve [18]. The HU value is mapped to a kind of material based on the table measured by W. Schneider et al., and the element composition is decided according to ICRU Report 46: Photon, Electron, Proton and Neutron Interaction Data for Body Tissues [19].

Sources used in HDR brachytherapy are hermetically sealed in metal shells. Radioactive sources are often cylindrical ^{192}Ir sources several millimeters in length and sub-millimeter in diameter. The shape of radioactive sources can hardly be described in voxel geometry, and the energy spectrum of ^{192}Ir is complex. THUBrachy uses a phase-space file generated by simulating the transport of photons emitted from sources using the general-purpose MC code Geant4. Photons emitted from the source and transported through the source shell are recorded when they reach the outside surface of the source case. In this study, two types of widely used sources, i.e., VS2000 and GammaMedPlus, are utilized, for which one million source photons are stored in the phase-space file.

2.2.2 *Random number generators*

The random number generator is an important part of MC simulations. Traditional simple generators, such as 32-bit linear congruential generators and simple linear-feedback shift registers, are not suitable for THUBrachy. Thus, this study adopts two types of random number generators, XORSHIFT-ADD (XSadd) and Philox [20, 21].

XORSHIFT-ADD is a modified version of the XORSHIFT generator introduced by Mutsuo Saito and Makoto Matsumoto (February 2014). XORSHIFT-ADD has a 128-bit internal state and a period of 2^{128}–1. It passes the strict random-number generator test TestU01 BigCrush [22]. Compared with a widely used generator in GPU computing, XORWOW, XORSHIFT-ADD is faster and gains better scores in statistical tests. For parallel generation, THUBrachy uses a jumping function to skip the *N*-th random number of the series to generate a new internal state, which ensures that each parallel thread uses an independent sub-series of random numbers. Conversely, Philox is a counter-based random-number generator. This generator uses the counter as the internal state, relying on a complex mapping function to generate random numbers. Each parallel thread uses a different interval of the counter to obtain an independent subseries of random numbers, which means that Philox is more suitable for parallel computing. Philox also passes all the rigorous BigCrush tests in the extensive TestU01.

THUBrachy chooses XORSHIFT-ADD as the default random-number generator because it runs faster than Philox. However, for large-scale parallel computing or situations with high demand for statistical performance, Philox is recommended.

2.3 Heterogeneous computing implement

THUBrachy is accelerated via heterogeneous computing with the application of hardware accelerators. Hardware accelerators often have a memory separated from the main memory of the computer. The code therefore needs to be modified or rewritten based on a suitable programming model corresponding to the architecture and memory structure of the accelerator.

THUBrachy is aimed at using these different kinds of hardware accelerators to speed up the simulation. It was originally designed to be suitable for parallel execution by programming principles. The requirements for parallel computing were fully considered when the basic code of THUBrachy, that is, the serial version, was developed. All functions are designed as reentrant functions to ensure that the code can be safely executed in parallel. The data structures are designed to be as simple as possible to fit different types of accelerators. The particles to simulate are divided into thousands of groups along with a subseries of random numbers to ensure that the result is the same despite the diversity in executing orders. In the same group, executing orders of different versions with different hardware accelerators are the same regardless of the accelerator used. In theory, differences among these THUBrachy versions are the execution orders of groups, which will not affect the results.

These efforts help to develop the basic code of THUBrachy into an extendible parallel code. Then, parallel codes for different kinds of hardware accelerators are generated by combining the code with the parallel-programming model and some modifications.

The serial version of THUBrachy is modified using OpenMP, OpenACC, and CUDA to fit different hardware accelerators, granting the tool the ability to use most types of hardware accelerators. All modified versions of THUBrachy are combined with MPI to run on larger-scale multi-node computing systems. These versions of

THUBrachy and the fitted hardware accelerators are listed in Table 1.

Parallel computing is suitable for the acceleration of MC simulation because the particles are independent. For multicore CPUs, computing loads are divided into threads on average. The results are gathered and summarized when all threads finish the simulation. The hyper-threading technology is supported by most × 86 CPUs. The number of threads is set to twice that of the cores used. The characteristic of the MC method is random simulation, which often leads to cache miss. Hyper-threading technology improves the utilization of computing resources. Tests were also performed to prove that the code set with twice the number of threads of the cores is faster than the code using the same number of cores.

Owing to the different architectures of hardware accelerators, specific methods were utilized when developing parallel codes for hardware accelerators. The acceleration of code achieved by the GPU is mainly due to the single-instruction-multiple-threads mode. This means that a group of threads in the GPU can only execute the same instruction at the same time. Many efforts have been made to reduce the divergence between threads as much as possible. According to the code analysis, the main computing time is consumed by calculating the photon intersecting point with a voxel boundary. This process was optimized to avoid branches. The photoelectric effect is simulated by modifying the weight instead of killing the particle. The processes of simulating Compton and Rayleigh scattering were optimized to reduce the branches as much as possible. Other methods aimed to overlap the computing and the delay and make full use of the memory. The GPU utilizes a static mode to manage threads in a streaming multiprocessor (SM) so that it can rapidly switch the threads without any extra expenses. To decrease the influence of delay brought by thread divergences and access to global memory, the number of threads in an SM in the GPU was set to several times the number of CUDA cores in the SM. By using this method, the SM can switch threads to continue computing rather than wait for the delay to overlap computing and the delay. The number of threads in an SM was also determined by the number of registers in the SM to limit the number of required registers less than the registers provided by the SM, which makes full use of the high-speed memory of the GPU and reduces the access to the global memory as much as possible.

Table 1 Devices and corresponding programming models of all versions of THUBrachy

Version	Device	Programming model
1	Multicore CPU	OpenMP + MPI
2	Multicore CPU	OpenACC + MPI
3	Intel Xeon Phi	OpenMP + MPI
4	NVIDIA GPU	CUDA + MPI
5	NVIDIA GPU	OpenACC + MPI

The architecture of Intel Xeon Phi is close to that of a CPU, but Intel Xeon Phi supports four threads in a core. Considering that the MC simulation would lead to many delays, the number of threads in Intel Xeon Phi was set to four times the number of cores in it to make full use of the computing resource. The code executing on Intel Xeon Phi was translated from the code of a multi-core CPU without additional optimization, except for the number of threads.

2.4 Validation of THUBrachy

To ensure that THUBrachy is accurate enough for clinical applications, several tests were performed. The results of the three versions of THUBrachy using hardware accelerators were first compared to guarantee the consistency of results obtained by the three versions. THUBrachy was validated by benchmarking it against TG-43 and by comparing its results with those of Geant4 using a clinical case.

2.4.1 Comparison among different versions of THUBrachy.

The process of heterogeneous computing implementation ensures the consistency of results from different versions theoretically. The results from the different versions with various hardware accelerators were compared to guarantee consistency. Dose distributions of the same situation were calculated by the serial version, the multi-core-CPU-accelerated version, the GPU-accelerated version, and the Intel-Xeon-Phi-accelerated version.

2.4.2 Benchmark against TG-43

TG-43 formalism, which provides dose-distribution parameters in an "infinitely" large water medium, is widely used in clinical brachytherapy dose calculation. Parameters are provided by radical dose functions and anisotropic functions to describe the effect of source geometry, attenuation, and anisotropy. This study performed a simulation using a prepared phase-space file to validate the code. Because of photon backscattering, a phantom larger than the region of interest is needed for benchmarking against TG-43 to avoid the discrepancy due to the difference in the phantom setup between TG-43 and THUBrachy. A large water phantom was set with a size of $80 \times 80 \times 80\ \mathrm{cm}^3$ to test THUBrachy against TG-43 in an area of the phantom with the distance to source less than

20 cm. The parameters of TG-43 are provided as a "point" dose. Thus, a small-sized voxel ($0.1 \times 0.1 \times 0.1$ mm^3) is required when the distance to the source is small. In areas where the distance to the source is larger than 10 cm, a large-sized voxel ($2 \times 2 \times 2$ mm^3) is used as a compromise between the accuracy and memory requirement. Doses are recorded to generate TG-43 parameters and compare THUBrachy with the TG-43 method. The benchmark against TG-43 is aimed at validating the phase-space file and dose calculation code in a simple uniform geometry.

2.4.3 Comparison with Geant4 using a clinical case

Because THUBrachy aims to calculate doses in cases with a heterogeneous medium, benchmarking against patient cases is important for the accuracy test. A cervical carcinoma case from the Peking University Third Hospital was used in this test. As previously mentioned, the MC simulation using a general-purpose MC code can provide accurate dose results and hence is regarded as the gold standard. An accuracy test using the clinical case was performed by comparing the results from THUBrachy and the widely used general-purpose MC code Geant4. Due to the short range of the secondary electrons produced by interactions from photons emitted by the radionuclides in brachytherapy, TG-43 recommends that electron transport is not required and collision kerma closely approximates the absorbed dose. The physics process in Geant4 was set to only transport photons, and the dose was calculated using the track-length estimator. The physics processes and the method of dose calculation for Geant4 and THUBrachy are similar. Moreover, the results of Geant4 and THUBrachy are comparable.

The case used a Fletcher-Suit Delclos-style T&O applicator with a VS2000 source. The case included a set of CT images with $512 \times 512 \times 204$ voxels with a resolution of $1.17 \times 1.17 \times 2.00$ mm^3 for dose planning. The CT images used in the dose planning of the case were down-sampled and processed via the module described in Sect. 2.2.1 to generate a phantom with $256 \times 256 \times 204$ voxels for the MC dose calculation. The density and material of the voxels in the phantom were determined by CT images so that the phantom represents the real geometry of the patient in dose calculation and planning. There were 22 source dwell positions included in the treatment plan for the case.

The dose distribution, dose of the target area, dose of organs at risk, and dose-volume histogram (DVH) of the case were generated using THUBrachy and general-purpose MC code Geant4. The results were compared to validate THUBrachy.

3 Results and discussion

3.1 Validation results

3.1.1 Comparison among the different versions of THUBrachy

Dose distributions in the conditions including TG-43 and the clinical case were calculated by all the versions of THUBrachy. The results were compared voxel by voxel. The values of the dose in each voxel are consistent for at least seven significant digits. However, even minor differences are still possible due to rounding errors.

3.1.2 Benchmark against TG-43

The TG-43 formalism provides some parameters to describe the difference in dose distribution between the real situation and the isotropic and inverse-square distributions. This study calculates a series of parameters of radial dose functions and anisotropic functions to validate the code in the TG-43 condition. The results are shown and compared with Talyor's [23, 24] (recommendation of AAPM TG-43, labeled as TG-43 in the figure) in Fig. 2.

The results obtained by THUBrachy are almost consistent with those obtained by TG-43. For 98% of the positions, the differences between THUBrachy and TG-43 are less than 2%. When the distance between the source and the chosen point is too close, a larger discrepancy can be found. As previously mentioned, because of the small voxel resolution in the close area, the kerma approximation is no longer correct. However, this effect cannot affect the code accuracy in clinical conditions where the voxel size is $\sim$ 1 mm.

Benchmarking against TG-43 is just a preliminary validation of THUBrachy using a widely reliable result. Considering that the real application is more complex than this, tests in conditions close to those of clinical application should be performed.

3.1.3 Clinical case

The test conditions are described in the validation section. Dose distributions were compared voxel by voxel. To compare the difference in doses between THUBrachy and Geant4, 4×10^4 primary photons were simulated for each case, limiting the uncertainty of the average organ dose to within 2%. A typical cross section of the 3D dose distributions of the clinical case is chosen to show the dose distributions obtained from the two codes and their difference in Fig. 3. The relative differences in the voxel

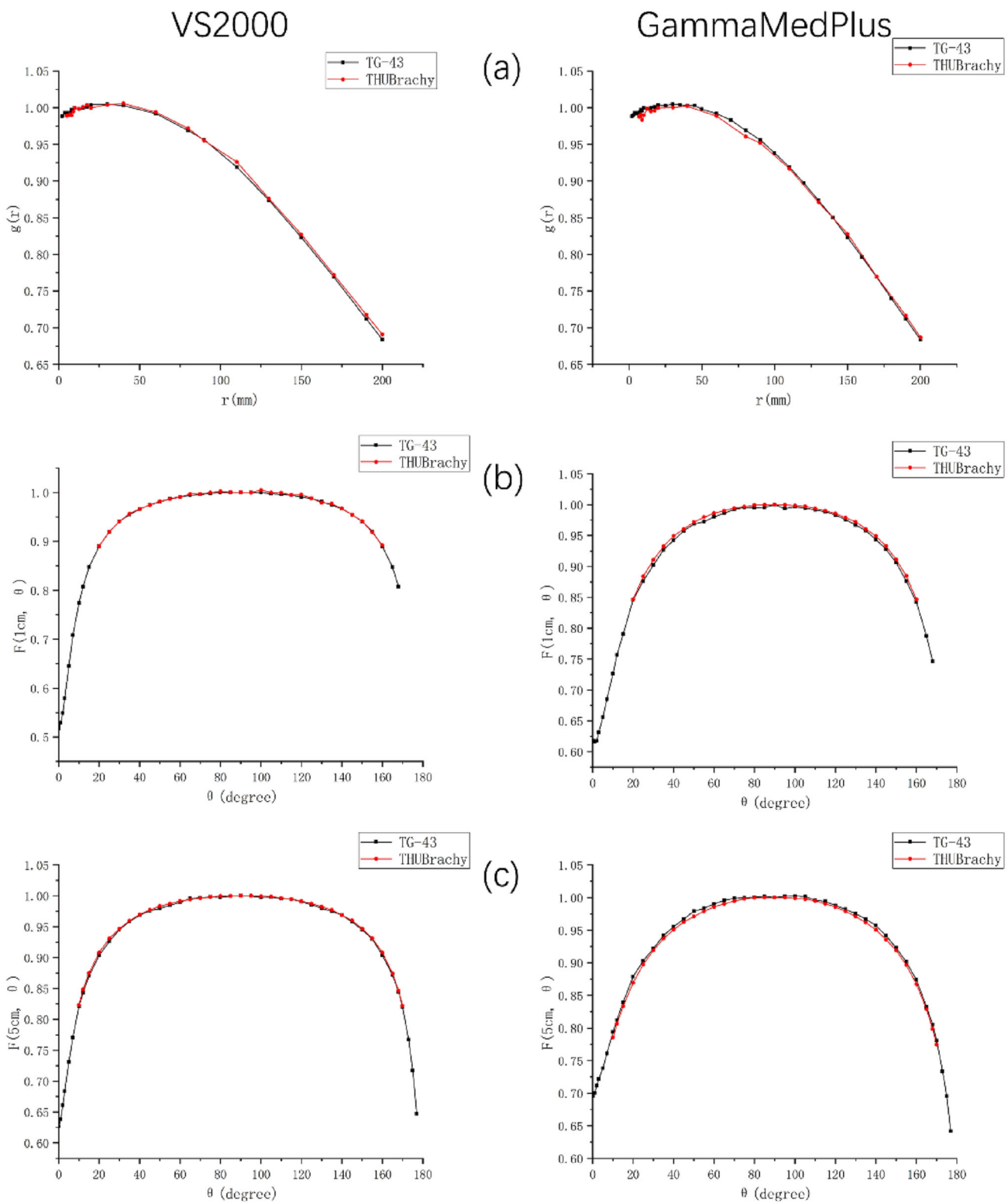

Fig. 2 (Color online) Comparison of single-source dosimetry between THUBrachy and TG-43 in water for VS2000 (left) and GammaMedPlus (right). **a** Radial dose function, **b**, **c** anisotropic functions at r = 1, 5 cm, respectively, where r denotes the radial distance from the source.

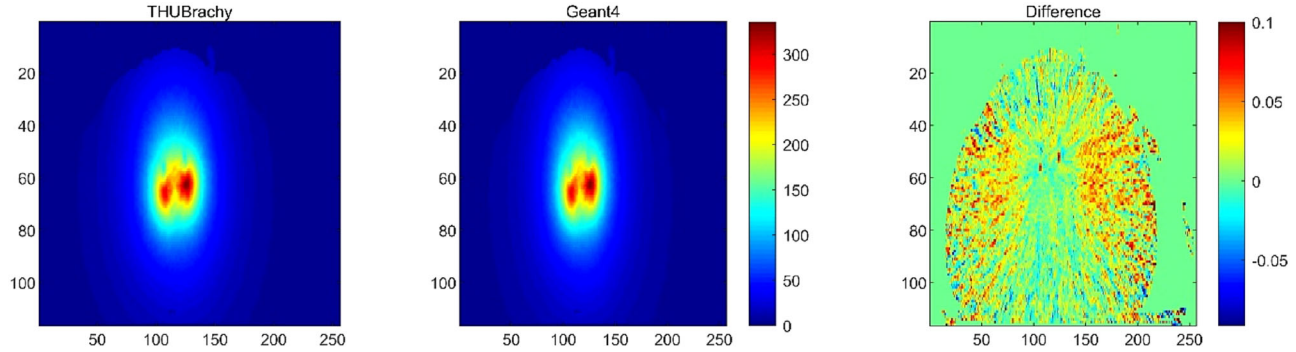

Fig. 3 (Color online) Comparison of dosimetry distribution between THUBrachy and Geant4 for the case (unit of dose: cGy)

doses between THUBrachy and Geant4 were calculated using Eq. (5).

$$\text{difference} = \frac{D_{\text{THUBrachy}} - D_{\text{Geant4}}}{D_{\text{Geant4}}}. \quad (5)$$

The dose calculated by Geant4 is regarded as the reference. Considering that the uncertainty on voxel dose would be great when the dose is too low, a 5% prescription dose was set as the cut-off (the prescription dose of the clinical target volume (CTV) is 6 Gy in this case, and the cut-off is 0.3 Gy). A voxel dose lower than 0.3 Gy is not compared.

Then, the differences in the voxel doses were counted to generate a histogram by a 0.01 interval step. The histogram of differences for all voxels with doses higher than the cut-off is shown in Fig. 4.

The results show that the relative dose discrepancies in 99% of voxels are less than 5% for the case, including voxels far away from sources. The average difference of the voxel dose between THUBrachy and Geant4 is approximately equal to zero. Note that the differences are calculated voxel by voxel, and the statistical error contributes most to the discrepancy. Because of the non-uniform dose distribution, a large number of photons should be simulated to limit the dose uncertainty in all voxels, especially voxels far away from sources, to a very low level. For Geant4, this method would require a large number of computing resources that are too heavy to be practical. Therefore, a test with lower uncertainty was not performed because it hardly affects the clinical application. To validate the results of THUBrachy with less uncertainty, the differences in the organ doses were also compared. The doses for all the related organs in the case calculated with THUBrachy and Geant4 are shown in Table 2.

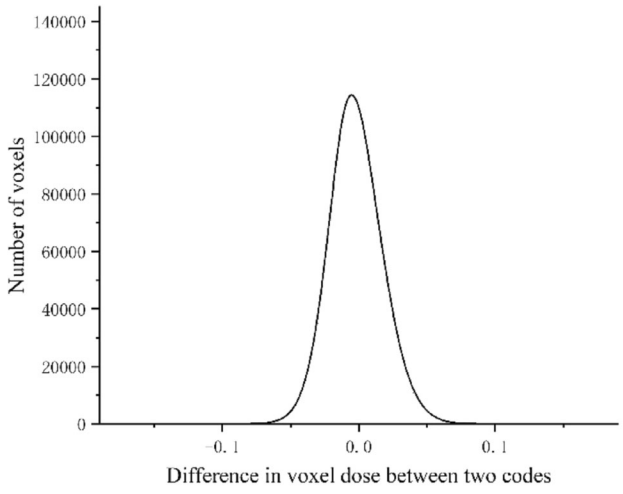

Fig. 4 Histogram of the difference in voxel doses between THUBrachy and Geant4

Differences in the organ doses were much lower than those on a single voxel dose, which indicates that the dose difference was mainly attributed to the statistical error. The residual part of the difference was due to the difference in the physics model between Geant4 and THUBrachy [25, 26].

For clinical applications, some characteristic values, such as the dose of the target area, dose of organs at risk, and DVH, are important. The values of the tested case were generated using THUBrachy and general-purpose MC code Geant4. The DVH is shown in Fig. 5, and other values are listed in Table 3, including D_{2cc} in the organs at risk and D_{90} in the CTV.

The results show good agreement between the two codes for all parameters, including dosimetric parameters, DVH,

Table 2 Comparison of the organ dose of the case (VS2000) between Geant4 and THUBrachy

Organ	THUBrachy (Gy)	Geant4 (Gy)	Difference (%)
Bladder	1.506	1.493	0.87
Colon	1.673	1.676	− 0.18
Rectum	0.881	0.884	− 0.34
Small Intestine	0.507	0.507	0.04

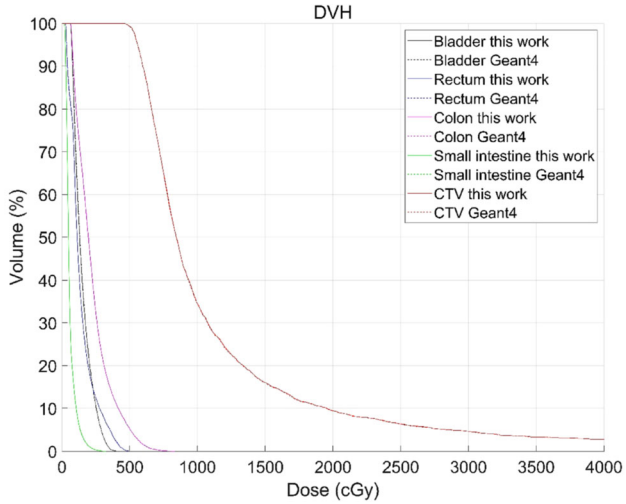

Fig. 5 DVH of the tested clinical case

and organ doses. The differences in the results between the two codes are almost entirely attributable to statistical uncertainty. This result indicates that the accuracy of THUBrachy is comparable with that of the general-purpose MC code for dose calculation in brachytherapy, but it consumes much less computing time.

The dose distribution calculated by THUBrachy was also compared with the results from Varian's Eclipse TPS. The dose distribution was calculated via TG-43 formalism in TPS. The same cross section is shown in Fig. 6. The DVH is shown in Fig. 7, and the dosimetric parameters are shown in Table 4.

The results of the TPS and THUBrachy showed similar dose distributions and DVHs in general, but evident differences were also existent. The TG-43 formalism used in the TPS regards a patient as a uniform water phantom when calculating the dose. The difference in the density and composition between the tissue and organ of the patient and the water phantom is ignored by TG-43, which contributes to the difference TG-43 and the result of the MC simulation. As shown in Sect. 3.1.1, the results of THUBrachy and TG-43 are the same when the dose distribution was calculated using the same water phantom (the water phantom used in the TG-43 formalism is described in Sect. 2.4.2).

3.2 Performance evaluation

As previously mentioned, this study aims to develop a fast MC code as a dose calculation tool for clinical use. Execution speed is a key goal of this code while producing accurate results. A server equipped with an NVIDIA GPU and an Intel Xeon Phi was chosen as the test platform. The codes were directly executed on the server operating system without virtualization. Information on the equipment is listed in Table 5. The execution speed is affected by many factors, including the voxel number, source position, and

Table 3 Comparison of the dosimetric parameters of the case (VS2000) between Geant4 and THUBrachy

Organ	Dosimetric parameter	THUBrachy (Gy)	Geant4 (Gy)	Difference (%)
Bladder	D_{2cc}	3.170	3.156	0.44
Colon	D_{2cc}	5.862	5.872	− 0.17
Rectum	D_{2cc}	3.879	3.868	0.28
Small intestine	D_{2cc}	1.736	1.745	− 0.52
CTV	D_{90}	5.975	5.972	0.05

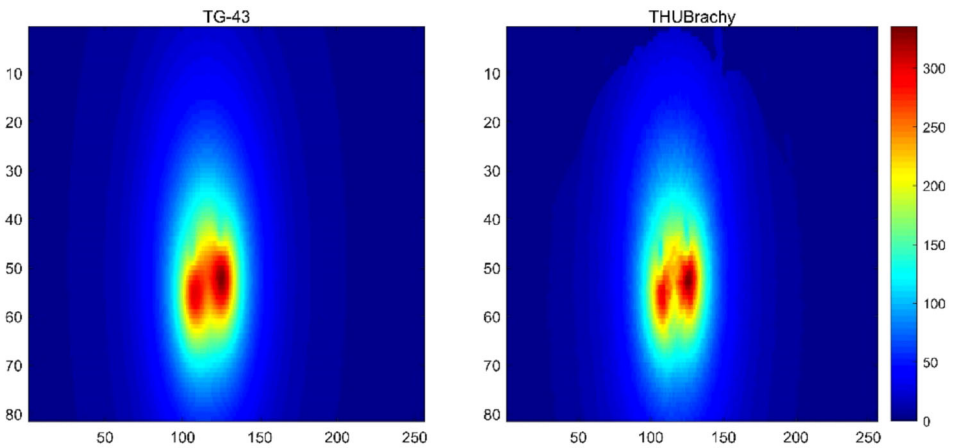

Fig. 6 (Color online) Comparison of the dosimetry distribution between TG-43 and THUBrachy for the case (unit of dose: cGy)

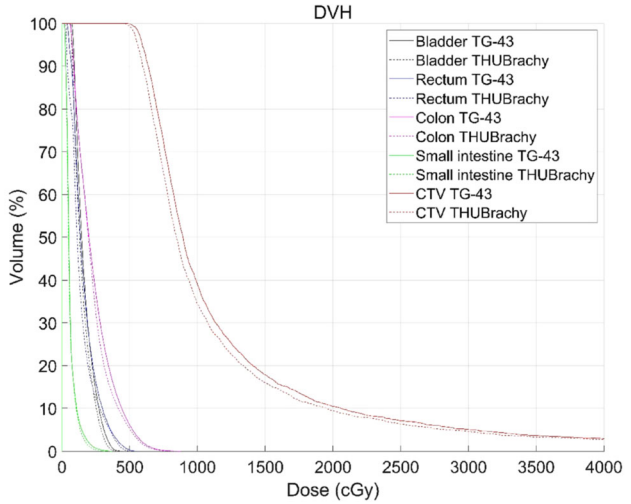

Fig. 7 (Color online) DVHs of the tested clinical case calculated by TPS (TG-43) and THUBrachy

load of the platform. The speed of the code in the chosen case provides a reference for performance.

To compare the performance of the different types of accelerators, this study compares the execution speed of the aforementioned clinical case using THUBrachy with different accelerators. For multicore CPUs, 32 parallel threads were used in the performance test. As the speedup is nearly proportional to the number of threads for CPU acceleration, 32 parallel threads were set to test the code via many threads while leaving some threads for other users. For Intel Xeon Phi, 240 threads were used to maximize the computing ability provided by the card. The computing resources of all GPUs are used by setting the number of threads to four times that of the CUDA cores. Because the GPU can switch threads without additional expense, four threads of the CUDA cores were set to fully use the resources of the GPU and overlap computing and delay. Programs compiled by Intel Compiler were compiled with interprocedural optimization. The codes compiled by PGI compilers and NVCC were compiled with -O3 optimization level. The performance test results of the different hardware accelerators are shown in Table 6.

Table 6 Execution speeds of different hardware accelerators for the tested case (unit: s/10^7 primary photons)

Device	Execution time	Speedup
CPU (1 thread/serial)	820	Baseline
CPU (32 threads)	30	~ 27 ×
GPU	4	~ 200 ×
Intel Xeon Phi	13	~ 63 ×

The comparison of performances between different types of accelerators shows the effect of acceleration brought by the type of accelerator. Because of the different architectures of the accelerators, the calculation efficiencies greatly differ. The efficiency is greatly influenced by the structure, algorithm, and optimization of the code, so the results of this study represent the performance of THUBrachy and provide a reference for the acceleration of the MC code via accelerators.

The results indicate that multicore CPU, GPU, and Intel Xeon Phi can accelerate simulation efficiently. Accelerating the MC code using GPUs is of great research interest. More tests and analyses have been performed to study the acceleration performance of the GPU. Instructions for

Table 4 Comparison of the dosimetric parameters of the case (VS2000) between TG-43 and THUBrachy

Organ	Dosimetric parameter	TG-43 (Gy)	THUBrachy (Gy)	Difference (%)
Bladder	D_{2cc}	3.381	3.170	6.66
Colon	D_{2cc}	6.233	5.862	6.32
Rectum	D_{2cc}	4.257	3.879	9.74
Small intestine	D_{2cc}	2.039	1.736	17.45
CTV	D_{90}	6.383	5.972	6.88

Table 5 Equipment information of the performance evaluation platform server

Configuration	Information
CPU	2 × Intel Xeon E5-2630 v4 (10cores/20threads per CPU)
GPU	1 × NVIDIA GTX 1080Ti
Intel Xeon Phi	1 × Intel Xeon Phi 7120P (61cores/244threads)
RAM	64 GB 2400 MHz DDR4
Operating System	Ubuntu 16.04
Compiler	Intel Parallel Studio XE Cluster 18.1 & PGI 18.4 (for OpenACC)
CUDA version	8.0

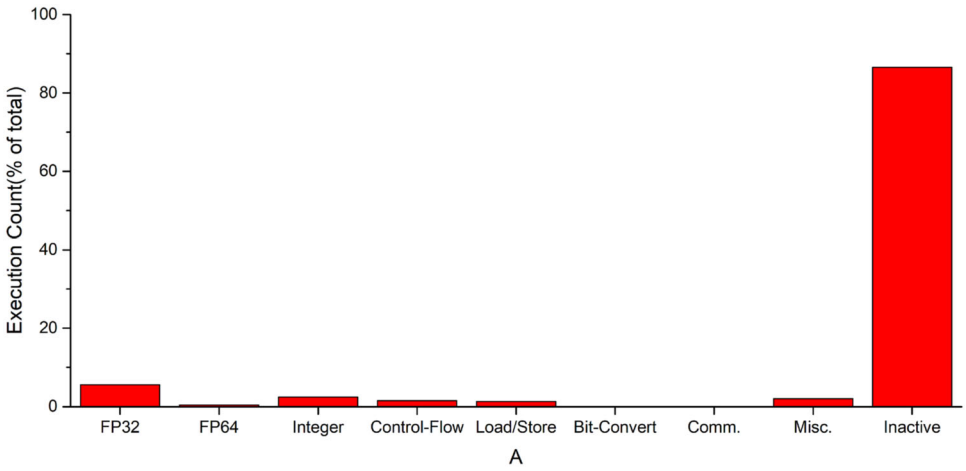

Fig. 8 Instruction execution counts of GPU-accelerated THUBrachy

executing this code on the GPU are counted and listed in Fig. 8. This figure is generated by nvprof, a GPU code-performance analysis tool.

The analysis of instructions shows that more than 80% of the instructions are inactive. GPU uses single instruction multiple data (SIMD) to achieve high computing efficiency. A warp of threads can execute only one instruction to deal with different data at the same time. Threads should wait when the code encounters a divergence. The MC code can hardly avoid divergence, especially when the geometry or physics are complex. Furthermore, the summary of the dose should be atomic to guarantee correctness. Different threads should wait in a queue when reading and writing the same data, which makes the code execute in series instead of parallel. Therefore, accelerating the MC code using a GPU cannot make full use of the computing capacity of the GPU.

For multicore CPUs, the speedup is almost proportional to the number of cores. The Intel Xeon Phi also showed good parallel efficiency. The MC simulation process is independent for every particle. The particles are divided into cores equally for multicore CPUs and Intel Xeon Phi. Every core contains a complete instruction system so that it can work independently. The high parallel-efficiency results are consistent with the architectures of multicore CPUs and Intel Phi.

However, the results also show that the GPU is more efficient than the multicore CPU and Intel Xeon Phi even though the high parallel efficiency of multicore CPUs and Intel Xeon Phi. The reason is that the efficiency of a single core is low. The main computing time of the simulation in brachytherapy dose calculation with a voxel geometry is consumed by photons intersecting with the voxel boundary. This method requires a wide memory bandwidth and high computing performance. The geometry intersecting calculation can be accelerated by the massive computing resources provided by the GPU, which contains thousands of floating-point units. By contrast, codes executing on the CPU and Intel Xeon Phi are not optimized using SIMD instructions and pre-fetch instructions and do not make full use of caches. SIMD instructions, pre-fetch instructions, and caches are designed especially for CPU and Intel Xeon Phi. Codes executing on the CPU and Intel Xeon Phi without SIMD and pre-fetch instructions optimizations simply make use of small parts of computing resources provided by the core of the CPU and Intel Xeon Phi. The low utilization rate of the computing resource provided by the core results in a relatively low-performance single-thread code. The performances of multi-thread codes are not high compared to that the GPU, although the codes show a high parallel-efficiency when executing on the multicore CPU and Intel Xeon Phi. These characteristics explain the low efficiency in multicore CPUs and Intel Xeon Phi.

To compare the performance between THUBrachy and a general-purpose MC code, Geant4 and THUBrachy are used to simulate the aforementioned case with the same number of particles. The results are listed in Table 6. As the physics processes of Geant4 and THUBrachy are similar, THUBrachy gives accurate results compared to Geant4. The comparison of performance between these codes shows the improvement in performance brought by THUBrachy while maintaining accuracy (Table 7).

The results indicate that THUBrachy is much faster than the general-purpose MC code for brachytherapy dose

Table 7 Execution speeds of Geant4 and THUBrachy for the clinical case (unit: s/10^7 primary photons)

Code	Execution time	Speedup
Geant4 (1 thread)	2200	Baseline
THUBrachy-GPU	4	$\sim$ 550 $\times$

calculation. THUBrachy is faster mainly because it concentrates on low-energy photons and voxel geometry, which are suitable for accelerating via heterogeneous computing. The execution time of the THUBrachy-CPU with a single thread and the same case as that of Geant4 is 820 s/10^7primary photons. As the design of Geant4 fully considers the flexibility of the code, the performance of Geant4 is approximately 40% of the THUBrachy-CPU. The comparison between THUBrachy and Geant4 provides a reference for the acceleration of MC code via specific-purpose simplification and heterogeneity.

The architecture of an accelerator determines whether it can accelerate the calculation for a certain problem. A simple physics model, simple geometry, and large requirements for memory bandwidth and computing capacity in brachytherapy dose calculation make it suitable for GPU and similar accelerators to accelerate efficiently. However, it is likely to fail when the physics model is much more complex. For a situation with complex geometry and complex physics processes, the warp divergence and frequent atomic instructions would greatly reduce the efficiency of the GPU. Other choices of accelerators should be considered. New hardware accelerators will appear in the recent future with the boom in high-performance computing. The THUBrachy developed in this work tries to fit with the trend of development of computer science and prepares for future accelerators.

Of note, THUBrachy requires more validation, especially for clinical applications. There may be inaccuracies caused by a series of reasons, such as inaccuracies in the physics model and phase-space file. Because of the inaccuracies of the patient position, the applicator geometry, and mapping CT image to material, the discrepancy in clinical use cannot be avoided.

4 Conclusion

This study developed a fast MC dose calculation tool, THUBrachy, for HDR dose calculation. The code can be accelerated by multicore CPU, GPU, Intel Xeon Phi, and other hardware accelerators. THUBrachy considers the major physics interactions of photons in the energy range from 10 keV to 3 MeV. An accuracy test validated the code by comparing the results generated by THUBrachy and Geant4. The performance tests indicated that accelerators can accelerate the dose calculation effectively, and the speed of THUBrachy is much faster than that of the general-purpose code. The GPU-accelerated THUBrachy is the fastest version, which is approximately 200 times faster than that of the serial version. There is great potential for clinical application in brachytherapy dose planning using THUBrachy. For future research, more attempts will be made to provide a more accurate and more efficient tool for brachytherapy dose calculation.

Author contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by An-Kang Hu, Huan Liu, Hui Zhang, Zhen Wu and Rui-Jie Yang. The first draft of the manuscript was written by An-Kang Hu and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

1. B.R. Thomadsen, J.F. Williamson, M.J. Rivard et al., Anniversary paper: past and current issues, and trends in brachytherapy physics. Med. Phys. **35**(10), 4708–4723 (2008). https://doi.org/10.1118/1.2981826
2. R. Nath, L.L. Anderson, G. Luxton et al., Dosimetry of interstitial brachytherapy sources: recommendations of the AAPM Radiation Therapy Committee Task Group No. 43. Med. Phys. **22**(2), 209–234 (1995). https://doi.org/10.1118/1.597458
3. M.J. Rivard, B.M. Coursey, L.A. DeWerd et al., Update of AAPM Task Group No 43 Report: a revised AAPM protocol for brachytherapy dose calculations. Med. Phys. **31**, 633–674 (2004). https://doi.org/10.1118/1.1646040
4. M.J. Rivard, W.M. Butler, L.A. DeWerd et al., Supplement to the 2004 update of the AAPM Task Group No. 43 Report. Med. Phys. **34**, 2187–2205 (2007). https://doi.org/10.1118/1.2736790
5. G. Anagnostopoulos, D. Baltas, E. Panteli et al., The effect of patient inhomogeneities in oesophageal 192Ir HDR brachytherapy: a Monte Carlo and analytical dosimetry study. Phys. Med. Biol. **49**(12), 2675–2685 (2004). https://doi.org/10.1088/0031-9155/49/12/014
6. A. Ahnesjö, Collapsed cone convolution of radiant energy for photon dose calculation in heterogenous media. Med. Phys. **16**, 577–592 (1989). https://doi.org/10.1118/1.596360
7. K.A. Gifford, J.L. Horton, T.A. Wareing et al., Comparison of a finite-element multigroup discrete-ordinates code with Monte Carlo for radiotherapy calculations. Phys. Med. Biol. **51**(9), 2253–2265 (2006). https://doi.org/10.1088/0031-9155/51/9/010
8. L.L. Beaulieu, A. Carlsson-Tedgren, J.F. Carrier et al., Report of the Task Group 186 on model-based dose calculation methods in brachytherapy beyond the TG-43 formalism: current status and recommendations for clinical implementation. Med. Phys. **39**(10), 6208–6236 (2012). https://doi.org/10.1118/1.4747264
9. H. Fu, J. Liao, J. Yang et al., The Sunway TaihuLight supercomputer: system and applications. Sci. China Inform. Sci. **59**(7), 1–16 (2016). https://doi.org/10.1007/s11432-016-558-7
10. P. Zhang, J. Fang, C. Yang et al., MOCL: An efficient OpenCL implementation for the matrix-2000 architecture, in *CF '18 Proceedings of the 15th ACM International Conference on Computing Frontiers*. May 8–10, 2018, Ischia, Italy.
11. O. Chibani, C.C. Ma, HDRMC, an accelerated Monte Carlo dose calculator for high dose rate brachytherapy with CT-compatible applicators. Med. Phys. **41**(5), 051712 (2014). https://doi.org/10.1118/1.4873318
12. S. Hissoiny, M. D'Amours, B. Ozell et al., Sub-second high dose rate brachytherapy Monte Carlo dose calculations with bGPUMCD. Med. Phys. **39**(7), 4559–4567 (2012). https://doi.org/10.1118/1.4730500
13. Z. Tian, M. Zhang, B. Hrycushko et al., Monte Carlo dose calculations for high-dose–rate brachytherapy using GPU-

accelerated processing. Brachytherapy **15**(3), 387–398 (2016). https://doi.org/10.1016/j.brachy.2016.01.006
14. L. Su, Y. 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**(7), 071709 (2014). https://doi.org/10.1118/1.4884229
15. D.E. Cullen, A simple model of photon transport. Nucl. Instrum. Meth. B. **101**(4), 499–510 (1995). https://doi.org/10.1016/0168-583X(95)00480-7
16. D. Brusa, G. Brusa, J.A. Riveros et al., Fast sampling algorithm for the simulation of photon compton scattering. Nucl. Instrum. Meth. Phys. Res. A **379**(1), 167–175 (1996). https://doi.org/10.1016/0168-9002(96)00652-3
17. J.H. Hubbell, I. Overbo, Relativistic atomic form factors and photon coherent scattering cross sections. J. Phys. Chem. Ref. Data. **8**(1), 69–106 (1979). https://doi.org/10.1063/1.555593
18. J.R. Rumble, D.M. Bickham, C.J. Powell, The NIST x-ray photoelectron spectroscopy database. Surf. Interface Anal. **19**(1), 241–246 (1992). https://doi.org/10.1002/sia.740190147
19. W. Schneider, T. Bortfeld, W. Schlegel, Correlation between CT numbers and tissue parameters needed for Monte Carlo simulations of clinical dose distributions. Phys. Med. Biol. **45**(2), 459–478 (2000). https://doi.org/10.1088/0031-9155/45/2/314
20. ICRU, 2007. Photon, Electron, Proton and Neutron Interaction Data for Body Tissues (Report 46). International Commission on Radiation Units and Measurements (1992).
21. S. Vigna, Further scramblings of Marsaglia's xorshift generators. J. Comput. Appl. Math. **315**, 175–181 (2017). https://doi.org/10.1016/j.cam.2016.11.006
22. J.K. Salmon, M.A. Moraes, R.O. Dror et al., Parallel random numbers: As easy as 1, 2, 3. Conference on High Performance Computing Networking, Storage and Analysis, SC 2011, Seattle, WA, USA, November 12–18, 2011 DBLP (2011).
23. P. L'Ecuyer, R.J. Simard, TestU01: a C library for empirical testing of random number generators. ACM Trans. Math. Softw. **33**(4), 22 (2007). https://doi.org/10.1145/1268776.1268777
24. R.E. Taylor, D.W. Rogers, EGSnrc Monte Carlo calculated dosimetry parameters for 192Ir and 169Yb brachytherapy sources. Med. Phys. **35**(11), 4933–4944 (2008). https://doi.org/10.1118/1.2987676
25. Y. Seppenwoolde, I.K. Kolkman-Deurloo, D. Sipkema et al., HDR prostate monotherapy: dosimetric effects of implant deformation due to posture change between TRUS- and CT-imaging. Radiother. Oncol. **86**(1), 114–119 (2008). https://doi.org/10.1016/j/radonc.2007.11.004
26. G. Landry, B. Reniers, L. Murrer, Sensitivity of low energy brachytherapy monte carlo dose calculations to uncertainties in human tissue composition. Med. Phys. **37**(10), 5188–5198 (2010). https://doi.org/10.1118/1.3477161