Metropolis Monte Carlo simulation scheme for fast scattered X-ray photon calculation in CT

YUAN XU,1,3 YUSI CHEN,1,3 ZHEN TIAN,2 XUN JIA,2,5 AND LINGHONG ZHOU1,4

1School of Biomedical Engineering, Southern Medical University, Guangzhou 510515, China
2Department of Radiation Oncology, University of Texas Southwestern Medical Center, Dallas, TX 75235, USA
3contributed equally
4smart@smu.edu.cn
5xun.jia@utsouthwestern.edu

Abstract: Monte Carlo (MC) method is commonly considered as the most accurate approach for particle transport simulation because of its capability to precisely model physics interactions and simulation geometry. Conventionally, MC simulation is performed in a particle-by-particle fashion. In certain problems such as computing scattered X-ray photon signal at a detector of CT, the conventional simulation scheme suffers from low efficiency mainly due to the fact that abundant photons are simulated but do not reach the detector. The computational resources spent on those photons are therefore wasted. To solve this problem, this study develops a novel GPU-based Metropolis MC (gMMC) with a novel path-by-path simulation scheme and demonstrates its effectiveness in an example problem of scattered X-ray photon calculation in CT. In contrast to the conventional MC approach, gMMC samples an entire photon path extending from the X-ray source to the detector using Metropolis-Hasting algorithm. The path-by-path simulation scheme ensures contribution of every sampled event to the signal of interest, improving overall efficiency. We benchmark gMMC against an in-house developed GPU-based MC tool, gMCDRR, which performs simulations in the conventional particle-by-particle fashion. gMMC reaches speed up factors of 37~48 times in simple phantom cases and 20-34 times in real patient cases. The results calculated by gMCDRR and gMMC agree well with average differences < 3%.

© 2019 Optical Society of America under the terms of the OSA Open Access Publishing Agreement

1. Introduction

Monte Carlo (MC) simulation [1,2] was first proposed in the 1940s to investigate radiation shielding and distances that neutrons travel through various materials. Over the years, it has become one of the most important numerical algorithms to handle problems in biomedical field [3–6] that requires particle transport simulations in a high degree of accuracy. It is commonly considered as the golden standard particle transport simulation approach [7] because of faithful computations based on fundamental physics principles and flexibility to handle simulation geometry.

Due to its inherent probabilistic characteristics, MC method unavoidably faces the problem [8] that efficiency and accuracy act against each other. During the last two decades, tremendous efforts have been dedicated to improving efficiency while maintaining accuracy [9,10]. Numerous MC tools have been developed, such as EGSnrc [11], MCNP [12], PENELOPE [13] and gMCDRR [14,15]. To improve efficiency, these packages often employ variance reduction techniques [16] such as fixed splitting technique [17] and deterministic importance functions [18]. Efficiency enhancement methods have also been introduced to improve the computational speed at a cost of acceptable accuracy loss. Novel computational platforms have also been employed. In addition to conventional CPU clusters, graphics processing unit (GPU) [14,15,19] has recently demonstrated its enormous capability to substantially boost the efficiency.
Compared with CPU, GPU structure can reach speed up factors of above 100 times, while the accuracy remains intact.

Existing MC packages for particle transport simulation commonly perform computations via a particle-by-particle scheme [11–15]. In this scheme, particle travels from one interaction site to another followed by scattering toward a certain direction governed by differential cross section (DCS) functions. This conventional approach has little control about where the particle travels. Yet a simulation task usually aims at computing quantities at a specific geometric region, e.g. computing X-ray scatter signal at the detector of a CT system. Simulations under the conventional scheme unavoidably spend extensive efforts on transporting those particles that do not contribute to the signal of interest, deteriorating overall computational efficiency. Driven by the desire to improve efficiency by solving this problem, we propose an innovative MC scheme with a path-by-path sampling method realized through a Metropolis algorithm [20–22]. It is fundamentally different from the conventional approach sampling a particle and following it sequentially through a series of physical interactions. In this paper, we report our development of a GPU-based Metropolis MC (gMMC) package and its application for fast X-ray scatter signal calculation in CT. Specifically, gMMC samples the entire photon transport path extending from the X-ray source to the detector and utilizes the Metropolis algorithm to ensure correct probability among photon paths. This algorithm allows us to focus computations only on those paths that deposit signals to the detector, which hence enables a much higher computational efficiency than the conventional MC approaches.

The rest of this paper is organized as follows. In Section 2, we will first present the basic idea and overall structure of the proposed method. A number of gMMC key techniques will be described in detail. Section 3 will contain results of scatter calculation in both phantom and patient cases. Finally, we will conclude our paper and present some discussions in Section 4.

2. Method and materials

2.1 gMMC overall structure and GPU implementation

The main structure of gMMC workflow is illustrated in Fig. 1(A). Once the process begins, the simulation is initialized with necessary data such as geometry parameters, X-ray spectrum and physics data. All attenuation coefficient data and DCS data are stored in three-dimensional texture memory of GPU to achieve quick access because of cache-function of GPU. Other variables are stored in GPU constant memory or global memory depending on their sizes. All MC tally counters are reset. After that, simulation is performed in a batched fashion to allow estimation of uncertainties using results among batches.

The intensity of scatter signal at a pixel is sum of the signal contributed by all paths. To compute this value, specifically designed mutation strategies are employed to generate photon paths with different initial photon energies, directions, numbers of scattering events, scatter interaction locations, interaction types. All the paths connect the X-ray source to the detector, as shown in Fig. 1(B). The lines n, m, k are examples of photon paths with 1, 2, 3 scatter events before hitting the detector D, respectively. The physical probability of each photon path is evaluated, which determines the chance that a path is accepted or rejected according to the Metropolis-Hasting sampling algorithm [20–22]. This strategy maintains that the accepted paths follow the correct distribution. Once a path is accepted, we deposit photon energy on the detector. To employ the rapid parallel processing power of GPU, all GPU threads are simultaneously launched, each corresponding to a photon path. The simulation ends, when all batches of photon transport are finished.
2.2 Metropolis-Hasting algorithm for gMMC

Generally speaking, Metropolis-Hasting algorithm [20–22] is a method to draw samples from a given probability distribution \( p(x) \). Specifically, a sequence of random samples \( x_i \) are generated through the algorithm described in Table 1. In line 3, the term mutation refers to generating a new sample \( y \) based on the current sample \( x \) with a mutation probability \( T(x \rightarrow y) \). The design of the mutation strategies can be very creative and flexible. The probability to accept this mutation in line 4 is:

\[
p_{\text{accept}}(x \rightarrow y) = \min \left[ 1, \frac{p(y)T(y \rightarrow x)}{p(x)T(x \rightarrow y)} \right]
\]

where \( p(x) \) is the targeted probability to be sampled from. Specific to the problem of interest here, namely scatter calculation [23], the variable \( x_i \) in Table 1 refers to an entire photon path that starts from the X-ray source and ends on the detector.

Table 1. Metropolis-Hasting path sampling Method.

| Line | Description |
|------|-------------|
| 1    | Initialize \( x_0 \); |
| 2    | For \( i = 1, 2, 3... \) |
| 3    | \( y = \text{mutate}(x_{i-1}); \) |
| 4    | Compute \( p_{\text{accept}}(x_{i-1} \rightarrow y); \) |
| 5    | \( x_i = y \) with probability \( p_{\text{accept}} \), and \( x_i = x_{i-1} \) with probability \( 1 - p_{\text{accept}} \); |
| 6    | End         |

The mutation probability \( T(x \rightarrow y) \) is determined by the mutation strategy employed in line 3. In the Metropolis-Hasting algorithm, one has a large degree of freedom to design mutation strategies. The acceptance/rejection step in Table 1 maintains that the accepted events follow the targeted probability. For simplicity, we employ in this study a mutational strategy that generates a state \( y \) independent of the initial state \( x \). This leads to \( T(x \rightarrow y) = p_{\text{accept}} \).
probably of sampling a state. Hence the probability to accept this mutation in line 4 is simplified to
\[
p_{\text{accept}}(x \rightarrow y) = \min \left[ 1, \frac{p(y) p_{\text{sampling}}(x)}{p(x) p_{\text{sampling}}(y)} \right].
\]
Note that this choice makes the mutation step in fact a sampling step. But we still use the term mutation to follow the convention.

2.3 Derivation of photon path probability

To realize path sampling, it is essential to build photon path probability in the gMMC framework. An illustration of a photon path is shown in Fig. 2. Let us consider a photon path that starts from the X-ray source $S$, goes through a number of interaction points $A_i = (i = 1, 2, ..., N)$, and ends at a point $B$ on the detector $D$. Here $N$ is the total number of interactions for this path. The entire photon trajectory consists of $N+1$ segments. The probability of this trajectory $p(x)$ is the product of those of all segments, as well as those at all interaction points.

Let us start with the first segment that connects the source $S$ with the first scatter point $A_1$. Suppose the X-ray source emits photons with a directional probability function $F(A)$ and a spectrum $\phi(E)$, i.e. the probability of emitting a photon with initial energy $E_0$ to a unit solid angle towards a point $A_i$. The probability having a certain interaction at point $A_i$ is written as
\[
\mu(A_i) \exp \left[ -\int_{A} \mu(s) ds \right],
\]
where $\mu(\cdot)$ denotes the X-ray linear attenuation coefficient as a function of spatial coordinate. Note that physics quantities such as linear attenuation coefficients are energy dependent. This fact is not explicitly expressed here and below to avoid complex notations. The probability for the interaction type $T_1$ to occur at this point is expressed as \[
\mu_{T_1}(A_i) / \mu(A_i),
\]
with $\mu_{T_1}(\cdot)$ being the X-ray linear attenuation coefficient of interaction type $T_1$. Multiplying these factors together yields the probability of having a photon with energy $E_0$, moving from the source $S$ to a unit solid angle towards $A_i$, and experiencing an interaction $T_1$ at this point, $\phi(E_0) F(A) \mu_{T_1}(A_i) \exp[\int_{A} \mu(s) ds]$. Similarly, we can derive the probability for the second segment. The only difference is that the probability for the photon moving from the scatter point $A_1$ to the point $A_2$ is governed by the angular distribution of the scattered photon $p_{T_1}(A_1)$ after an event of type $T_1$. This probability is the normalized DCS function. Note that the argument of this probability function is the scatter angle defined by the scattering event $S \rightarrow A_i \rightarrow A_2$. Here, we use $A_i$ instead to simplify notation. With this term, the probability for a photon to move from the point $A_i$ to a unit solid angle towards the point $A_2$, having an interaction $T_2$ at this point is
Generally speaking, the probability associated with the path \( l_i \) for \( i = 2, 3, ..., N \) is:

\[
\rho_{\tau_i}(A_i) \mu_{\tau_i}(A_i) \exp \left[ -\int_{l_i} \mu(s) ds \right].
\]

For the last segment that extends from the last scatter point \( A_N \) to the detector, let us consider a small area \( dA \) on the detector plane. Denote the angle between the photon travel direction and the normal direction of the detector with \( \alpha \). The probability of having the photon at the point \( A_N \) scattered towards the area \( dA \) on the detector is:

\[
\rho_{\tau_N}(A_N) \cos \alpha dA / l_{N+1}^2.
\]

Note that the factor \( \cos \alpha dA / l_{N+1}^2 \) is the solid angle of \( dA \) with respect to \( A_N \). Furthermore, the probability for the photon reaching the detector is:

\[
\exp \left[ -\int_{l_{N+1}} \mu(s) ds \right].
\]

With the assumption of an ideal detector that captures all the photons, the probability of the last segment is:

\[
\rho_{\tau_N}(A_N) \exp \left[ -\int_{l_{N+1}} \mu(s) ds \right] \cos \alpha dA / l_{N+1}^2.
\]

With the probability defined for all the segments, we can multiply them together to form the probability of the entire path, yielding the expression used in the proposed Metropolis algorithm:

\[
p(x) = \phi(E_0) F(A) \prod_{i=1}^{N} \rho_{\tau_i}(A_i) \prod_{k=1}^{N+1} \exp \left[ -\int_{l_k} \mu(s) ds \right] dA \cos \alpha / l_{N+1}^2.
\]  

### 2.4 Photon path mutation

As mentioned in Sec 2.2, we employ a simple strategy that samples a new path \( y \) independent of the previous path \( x \). A photon path is completely characterized by the initial photon energy, the number of scattering events, interaction positions, types of scatter interactions at each interaction point, and the final location at the detector. Note that once all the scatter interaction points and the interaction types are determined, photon energy variation along the entire path is determined. In our implementation, the following five steps are performed to generate a new photon path each time.

1. **Source energy mutation.** An X-ray source has an energy spectrum. We sample the initial energy of the photon \( E_0 \) uniformly in the allowed energy range.

2. **Scatter order mutation.** The number of interactions can in principle be any positive integer. However, contributions of high order photon paths are expected small. Hence, we set a maximum number of interaction number \( N_{\text{max}} \) in our simulation and sample uniformly an integer \( N \) in the range of \([1, N_{\text{max}}]\).

3. **Interaction position mutation.** We first determine the initial direction of the photon by uniformly sampling among the directions extending from the X-ray source to the phantom. This is realized by predetermining a large direction region encompassing the phantom, uniformly sampling the direction within this region, and keeping the one intersecting with the phantom. After determining the direction of the initial photon, we sample the first interaction point \( A_1 \) uniformly in the range between the points entering and exiting the phantom. For interactions of higher order, a Gaussian distribution \([21] N(\mu, \sigma^2)\) is utilized to sample the scatter angle and a uniform distribution for the azimuthal angle. The parameters \( \mu \) and \( \sigma \) controlling the shape of the Gaussian distribution are parameters of the algorithm. \( \mu = 0.25 \pi \) and \( \sigma = 0.2\)
are empirically chosen. The distance to the next interaction is sampled uniformly in 
the range between zero and the intersection point of the scattered photon direction 
with the phantom boundary. This process iterates to generate all interaction positions. 
We do not consider interaction positions outside the phantom, because air has 
negligible probability to scatter a photon.

(4) Scatter type mutation. There are two possible types of photon interactions at each 
interaction point, namely Compton scatter and Rayleigh scatter. Photoelectric 
interaction is not possible, as it physically absorbs the photon and hence terminates 
the path, which cannot occur for a path that extends to the detector. In the simulation, 
we assign Compton scatter to an interaction point with a probability $\varsigma$ and hence 
Rayleigh scatter $1 - \varsigma$. An empirical value $\varsigma = 0.7$ is used. We purposely set $\varsigma > 0.5$.
because Compton scatter is more probable in the imaging X-ray energy range. We 
would like to clarify that we use the total X-ray linear attenuation coefficient $\mu$ when 
evaluating the path probability in Eq. (2). Therefore the effect of x-ray attenuation due 
to photoelectric interaction has been accounted in simulation, although the sampled 
paths will never contain photoelectric events. This is indeed a major advantages of our 
method, as it focuses simulations to those paths contributing to detector signal.

(5) Location on detector mutation. The point $B$ for the photon reaching the detector is 
sampled uniformly inside the detector.

### 2.5 Acceptance probability calculation

Once a path is generated, we first sequentially go through the interaction points to determine 
the path probability. First, scatter angles $\varphi_i$ are calculated at each scattering point $A_i$ based on 
geometry. The energy after each interaction $E_i, i = 1, 2, ..., N$ is needed to evaluate the 
probability of the path. For a Compton scattering event, the photon energy is 
$$E_i = \frac{E_{i-1}}{1 + \frac{E_{i-1}}{m_e c^2} (1 - \cos \varphi_i)}$$ 
where $m_e c^2$ is electron mass energy. Rayleigh scattering is 
elastic and hence $E_i = E_{i-1}$. Given the scattering angle and photon energy, we query the DCS 
data to obtain probability of scattering to this direction. In this work, PENELOE physical 
database used in gMCDRR [14] is employed. Figure 3 shows examples of Rayleigh DCS and 
Compton DCS of aluminum within the energy range of 0-150 keV. With all these quantities 
determined, we evaluate the probability of the path using Eq. (2).

![Fig. 3. Rayleigh DCS (A) and Compton DCS (B) of Aluminum within the energy range of 0-150 keV.](image)

As for the transition probability $T(x \rightarrow y) = p_{\text{sampling}}(y)$, it is the product of the 
probabilities in steps (1)-(5) described in Sec 2.4. Moreover, since only the ratio
\( P_{\text{Sampling}}(y) / P_{\text{Sampling}}(x) \) is needed to evaluate the acceptance probability, the probability terms corresponding to steps (1), (2) and (5) cancel between paths \( x \) and \( y \) due to uniform sampling. Hence, for the path \( x \), what matters is

\[
P_{\text{Sampling}}(x) \propto \left[ \prod_i \frac{1}{L_i} G(\theta_i; \mu, \sigma^2) \right]^{N_C} (1 - \xi)^{N_R},
\]

where \( i \) indexes each scatter event, \( L_i \) is length of sampling range each time, \( G(\theta_i; \mu, \sigma^2) \) is the Gaussian function evaluated at scatter angle \( \theta_i \), \( N_C \) and \( N_R \) are numbers of Compton and Rayleigh events on the path. Finally, we evaluate the acceptance probability using Eq. (1).

2.6 Validation studies

We first perform scattering calculations on simple homogeneous and inhomogeneous phantom cases to demonstrate effectiveness of the proposed gMMC package. We then use a head-and-neck cancer patient (HN) case to further test gMMC code in a realistic setting. For the geometry configuration in all the cases, the X-ray source-to-axis distance is 1000 mm and axis-to-imager distance is 500 mm. We consider a flat X-ray source fluence, i.e. photons are emitted from source to all directions evenly. The detector resolution is \( 512 \times 384 \) pixels with a pixel size of \( 0.781 \times 0.781 \) mm\(^2\). With regard to computational hardware, we use an NVIDIA GeForce GTX TITAN Z card that is equipped with 5.7k processors and 6.0 GB global memory.

![Fig. 4. (A) and (A1), homogeneous Al phantom and illumination geometry; (B) and (B1), inhomogeneous two-material phantom and illumination geometry; (C) and (D), HN cancer patient phantom under the IFOV and OFOV setting; (C1)-(C2), side view and axial view of the X-ray illumination geometry for the IFOV case; (D1)-(D2), side view and axial view of the X-ray illumination geometry for the OFOV case.](image)

We first demonstrate the performance of the proposed method in a homogeneous Aluminum phantom of ~1000 HU. The geometry is shown in Fig. 4 (A) and (A1). A monoenergetic X-ray
beam of 60 keV normally impinges on the phantom. The phantom dimension is $250 \times 280 \times 250$ voxels with a voxel size of $0.4 \times 0.1 \times 0.4$ mm$^3$. To further test gMMC code in an inhomogeneous case, we perform scatter calculation for a two-material phantom composed by soft tissue (~200 HU) and bone (~600 HU) as shown in Fig. 4(B) and (B1). The phantom dimension and voxel size are the same as those of the homogeneous phantom case.

As for the HN case, one real patient CT image consisting of air, tissue and bone is used. To test gMMC in different geometry configurations, we perform computations at two different situations: in-field of view (IFOV) case and out-of-field of view (OFOV) case. For the IFOV case, the whole phantom is under the X-ray illumination. The voxelized 3D phantom has a dimension of $100 \times 512 \times 512$ voxels and a voxel size of $0.625 \times 0.370 \times 0.370$ mm$^3$, as shown in Fig. 4(C). The simulation geometry of this case is depicted in Fig. 4 (C1) and (C2). For the OFOV case, the voxel size of this phantom is enlarged to $0.740 \times 0.740 \times 1.25$ mm$^3$ as shown in Fig. 4(D), so that a certain portion of the phantom is out of the X-ray illumination field, see Fig. 4(D1) and (D2). Both cases use a 100 kVp poly-energetic source with photon energy ranging from 8 to 100 keV.

gMMC is first benchmarked against gMCDRR [14] running on the same GPU card, which performs X-ray photon transport simulation in the conventional particle-by-particle way and is the state-of-the-art GPU-based MC simulation package for medical physics applications. gMCDRR has been extensively validated against established MC packages at its development stage [14]. To evaluate accuracy, the first and second order Compton scattering signal at the detector, first and second order Rayleigh scattering signal, and total scattering signal are recorded. The results are compared with those computed by gMCDRR using $5 \times 10^{11}$ source photons. The relative uncertainty of total scattering signal computed by gMCDRR under this number of photons is ~1%. We consider gMCDRR results as ground truth. For gMMC, the number of paths is $5 \times 10^9$ in all cases, under which the relative uncertainty of total scattering signal is also ~1%. We quantitatively compute relative difference between gMCDRR and gMMC results, e.g. $\|r-t\|/\|t\|$, to measure accuracy, where $r$ is the scattering signal computed by gMCDRR, and $t$ is that computed by gMMC. We have also validated our simulations against EGSnrc, a commonly used MC package for photon transport simulation. As such, the scatter images in the homogeneous Al phantom case and the OFOV HN case are calculated with EGSnrc. The results are compared with those computed by gMCDRR and gMMC.

3. Result

3.1 Homogeneous Al phantom and inhomogeneous two-material phantom cases

The scatter signals computed by gMMC and gMCDRR are presented in Fig. 5. The resulting images are visually in good agreement. The computation time to transport $5 \times 10^{11}$ photons in gMCDRR is 28.33 hrs. In contrast, it takes gMMC 35 mins to finish computations of $5 \times 10^9$ photons paths, yielding an overall speed up factors of ~48.57 times. The relative difference of the total scatter signals is 1.76%.
Fig. 5. Scatter signals of the homogeneous Al phantom case. (A1)-(A5) are first order Compton scatter, first order Rayleigh scatter, second order Compton scatter, second order Rayleigh scatter, and total scatter signals (sum of previous four figures) computed by gMMC; (B1)-(B5) are corresponding results computed by gMCDRR; (C1)-(C5) are profiles on the yellow lines of corresponding images.

For the inhomogeneous two-material phantom case as shown in Fig. 6, the scattering images computed by gMMC are visually very similar with those of gMCDRR results. The relative difference of total scatter signals is 1.76%. The computation time of gMMC is 46 mins. Comparing with the computation time of 28.33 hrs in gMCDRR, gMMC reaches an acceleration factor of ~36.96 times.

3.2 HN case

Computational results of the IFOV and OFOV HN cases are presented in Figs. 7 and 8, respectively. For the two cases, the relative differences between total scatter signals computed by gMCDRR and gMMC are 1.99%, and 2.89%. Regarding computational efficiency, the computation time of gMMC in the IFOV case is 53 mins, whereas that of gMCDRR is 30 hrs. Hence, the acceleration factor is ~33.96 times. As for the OFOV HN case, it takes gMCDRR 26.67 hrs and the time for gMMC is 77 mins. As a result, an overall acceleration factor of ~20.78 times is achieved.
3.3 Validation with EGSnrc

Figure 9(a)-9(c) shows the total scatter image of the homogeneous Al phantom case computed using EGSnrc, gMCDRR and gMMC. Visually, these three images are very close. Profiles at a horizontal line are plotted in Fig. 9(d). The agreements among them demonstrate the accuracy of the proposed scatter simulation method. The results of the OFOV HN case are shown in Fig. 10.
4. Discussion

In all the cases studied, the number of particles in gMCDRR is 100 times of the number of paths in gMMC. Yet the acceleration factors range in 20 ~50 times. This indicates that the efficiency per particle/path in gMMC is lower than that of gMCDRR. This is ascribed by the complex computations to evaluate path probability via Eq. (1). In particular, gMCDRR transports particles using the Woodcock scheme that effectively eliminates the need for complex voxel crossing computations [24]. In contrast, gMMC has to perform ray-tracing operations to go through each voxel on a photon path to evaluate the path integrals in the exponential term in Eq. (2), which is the major burden of computation.

This study serves as an initial study proposing the path-by-path simulation scheme and has several limitations. Future studies could be conducted to further improve the algorithm. A major issue in Metropolis-Hasting path sampling is the mutation scheme. The choice of mutation scheme can be very creative and flexible. Different strategies have different efficiency, but are expected to all converge to the same result, as the rejection step would maintain that those accepted events follow the targeted probability. As a general principle, the closer the distribution of sampled events is to the targeted distribution, the more efficient the Metropolis-Hasting algorithm is, as less sampled events are rejected. However, the computational load to sample events from a complex distribution may prolong the overall computation time. It is based on the trade-off between sampling efficiency and rejection efficiency that one determines the sampling/mutation strategy.

For instance, an alternative way to address the X-ray source spectrum is to sample the initial energy of each photon directly by numerically inverting the distribution defined by the spectrum. Nonetheless, the numerical inversion is relatively computationally intensive. We implemented this approach, and found that it only led to an extra speed up factors of 3%-5%. Hence, we kept the simple uniform sampling of initial energy in this study. A similar situation occurs when sampling the scatter event locations. It is possible to sample them following the actual position probability which follows an exponentially decayed form. However, the challenge is that, in the highly heterogeneous case such as a patient, the ray-tracing operation is needed to evaluate accumulated probability of interaction and to sample the event locations. Although this method makes sampled paths follow the targeted probability more closely, hence
enhancing the efficiency of the Metropolis-Hasting algorithm, the individual step of sampling becomes complicated, presumably reducing overall efficiency.

There are many parameters in the mutation strategy used in this study, such as the variance $\sigma$ of the Gaussian width. The parameters can be tuned, so that the generated paths are more desired, hence potentially improving efficiency. Furthermore, adaptive Metropolis algorithm [25] and random walk sampling [26] are also possible approaches to enhance efficiency. On the hardware side, multi-GPU can be an effective platform to further boost efficiency. Under this platform, photon paths can be distributed among all the GPUs. As computations among GPUs are independent, a linear scalability of the efficiency with respect to the number of GPUs is expected [27,28]. All of these directions will be explored in our future studies.

![Image](a) (b) (c) (d)

Fig. 9. (a)-(c) the total scatter image of Homogeneous Al phantom case from EGSnrc, gMCDRR and gMMC; (d) the profile of each result a blue line in (a).

![Image](a) (b) (c) (d)

Fig. 10. (a)-(c) the total scatter image of and OFOV HN case from EGSnrc, gMCDRR and gMMC; (d) the profile of each result a blue line in (a).

The purpose of our method is for fast scatter correction of cone-beam CT. For this problem in a clinical setting with mostly low-Z materials, it is expected that scatter signal is dominated by the first a few order events. Hence, we limited the mutation to scatter events up to second order for efficiency considerations. Quantifying the convergence with increasing order will be our future study. Meanwhile, it is also expected that our method is still valid when including
higher order events. In this case, biased sampling more low order paths when generating the paths would be helpful to improve efficiency.

When comparing different testing cases, the HN cases experienced lower computational efficiency compared to the other two relatively simple cases. This can be ascribed to the larger number of voxels and more complex phantom geometry and phantom material compositions. It is also noted that the efficiency of the OFOV HN case is lower than that of the IFOV HN case for two reasons. First, computations of path probability are more intensive due to more numerical integral operations associated with the longer photon paths. Second, in this OFOV case, a lot of scattered signals come from those paths that initially hit the phantom inside the FOV, are then scattered to outside the FOV, and are scattered back to reach the detector. However, the mutation strategy allows little chances to generate these paths, which requires large scattering angles. As a consequence, more paths are required to reach the correct path probability, hence reducing overall efficiency. We expect adjusting the mutation scheme can further improve efficiency in the OFOV case.

Generally, the proposed method can be used in all applications that the conventional MC photon simulation packages have been utilized. For instance, cone-beam CT is the standard image guidance tool for patient setup in radiation therapy. However, due to its large illumination field, scattered photons severely degrade its image quality. While kernel-based scatter correction methods have been used routinely in the clinic, it is still desirable to develop MC simulation based methods because of accuracy. Our previous studies have demonstrated feasibility of using conventional MC tools for this purpose. However, some approximations were employed to overcome the computational burden, which inevitably leads to accuracy concern [23]. With the help of the proposed method, it is potentially possible to estimate the scatter in a more accurate fashion without making approximations.

5. Conclusions

In this study, we proposed a Metropolis-Hasting sampling algorithm to perform photon transport simulations via a path-by-path scheme. We demonstrate effectiveness of this method in the context of computing scattered photon signals in CT. Different from conventional MC schemes that lacks the control of particle paths due to the particle-by-particle simulation scheme, the proposed method is able to control the sampled paths, which allows for a more effective use of sampled particles. This leads to a substantial reduction of the number of particles needed and therefore improved efficiency. It is found that gMMC reaches speed up factors of 37~48 times in simple phantom cases and 20~34 times in patient cases, while the differences from the results computed by the conventional MC package gMCDRR are within 3%. The combined accuracy and efficiency make the proposed scheme attractive in some certain applications. We are open to collaborations with interested users for initial tests and different applications of our package. After it is tested, we will build a user-friendly interface in future and make the package open source to the research community.

Funding

National Institutes of Health (NIH) (1R21EB021545, 1R01CA214639, 1R01CA227289); Cancer Prevention and Research Institution of Texas (RP160661); Chinese National key R&D development program (2016YFA0202003); National Natural Science Foundation of China (NSFC) (81601493, 81571771); Guangdong Natural Science Foundation of China (2016A030310388 and 2017A030313692)

Acknowledgments

We thank the members of the MCSimulation group from Department of Radiation Oncology at University of Texas Southwestern Medical Center for their assistance with comments and suggestions on the manuscript. We are grateful to Zhenqiang Ding for EGSnrc simulation.
References

1. N. Metropolis and S. Ulam, “The Monte Carlo method,” J. Am. Stat. Assoc. 44(247), 335–341 (1949).
2. K. Binder, D. Heermann, L. Roelofs, A. J. Mallinckrodt, and S. McKay, “Monte Carlo simulation in statistical physics,” Comput. Phys. 7(2), 156–157 (1993).
3. D. Boas, J. Culver, J. Stott, and A. Dunn, “Three dimensional Monte Carlo code for photon migration through complex heterogeneous media including the adult human head,” Opt. Express 10(3), 159–170 (2002).
4. E. Alerstam, W. C. Lo, T. D. Han, J. Rose, S. Andersson-Engels, and L. Lilge, “Next-generation acceleration and code optimization for light transport in turbid media using GPUs,” Biomed. Opt. Express 1(2), 658–675 (2010).
5. D. P. Kroese, T. Breuereon, T. Taimre, and Z. I. Botev, “Why the Monte Carlo method is so important today,” Wiley Interdiscip. Rev. Comput. Stat. 6(6), 386–392 (2014).
6. P. Li, C. Liu, X. Li, H. He, and H. Ma, “GPU acceleration of Monte Carlo simulations for polarized photon scattering in anisotropic turbid media,” Appl. Opt. 55(27), 7468–7476 (2016).
7. D. W. Rogers, “Fifty years of Monte Carlo simulations for medical physics,” Phys. Med. Biol. 51(13), R287–R301 (2006).
8. P. H. Waarts, “Efficiency and accuracy of Monte Carlo (importance) sampling,” in Bedford, T. Gelder, PHAJM van, Safety and Reliability. Proceedings of ESREL. European Safety and Reliability Conference 2003, Maastricht, the Netherlands, 15–18 June 2003, 1663–1678, (AA Balkema Publishers, 2003)
9. G. Pratx and L. Xing, “GPU computing in medical physics: a review,” Med. Phys. 38(5), 2685–2697 (2011).
10. X. Jia, P. Zienkine, and S. B. Jiang, “GPU-based high-performance computing for radiation therapy,” Phys. Med. Biol. 59(4), R151–R182 (2014).
11. I. Kawrakow, “The EGSnrc code system, Monte Carlo simulation of electron and photon transport,” NRCC Report Pirs-701 (2001).
12. X.-M. C. Team, “MCNP - A General Monte Carlo N-Particle Transport Code, Version 5,” (Los Alamos National Laboratory Los Alamos, NM, 2003).
13. J. Baró, J. Sempau, J. M. Fernández-Varea, and F. Salvat, “PENELOPE: An algorithm for Monte Carlo simulation of the penetration and energy loss of electrons and positrons in matter,” Nucl. Instrum. Methods Phys. Res. 100(1), 31–46 (1995).
14. X. Jia, H. Yan, L. Cerviño, M. Folkerts, and S. B. Jiang, “A GPU tool for efficient, accurate, and realistic simulation of cone beam CT projections,” Med. Phys. 39(12), 7368–7378 (2012).
15. X. Jia, H. Yan, X. Gu, and S. B. Jiang, “Fast Monte Carlo simulation for patient-specific CT/CBCT imaging dose calculation,” Phys. Med. Biol. 57(3), 577–590 (2012).
16. E. Mainegra-Hing and I. Kawrakow, “Variance reduction techniques for fast Monte Carlo CBCT scatter correction calculations,” Phys. Med. Biol. 55(16), 4495–4507 (2010).
17. Z. I. Botev and D. P. Kroese, “Efficient Monte Carlo simulation via the generalized splitting method,” Stat. Comput. 22(1), 1–16 (2012).
18. A. Hughighat and J. C. Wagner, “Monte Carlo variance reduction with deterministic importance functions,” Prog. Nucl. Energy 42(1), 25–53 (2003).
19. A. Badal and A. Badano, “Accelerating Monte Carlo simulations of photon transport in a voxelized geometry using a massively parallel graphics processing unit,” Med. Phys. 36(11), 4878–4880 (2009).
20. W. K. Hastings, “Monte Carlo sampling methods using Markov chains and their applications,” Biometrika 57(1), 97–109 (1970).
21. SiddharthaChib and EdwardGreenberg, “Understanding the Metropolis-Hastings Algorithm,” Am. Stat. 49, 327–335 (1995).
22. C. P. Robert and G. Casella, “The Metropolis-Hastings algorithm,” Springer Texts in Statistics 49, 327–335 (2016).
23. Y. Xu, T. Bai, H. Yan, L. Ouyang, A. Pompos, J. Wang, L. Zhou, S. B. Jiang, and X. Jia, “A practical cone-beam CT scatter correction method with optimized Monte Carlo simulations for image-guided radiation therapy,” Phys. Med. Biol. 60(9), 3567–3587 (2015).
24. E. Woodcock, T. Murphy, P. Hennings, and S. Longworth, “Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry,” in Proc. Conf. Applications of Computing Methods to Reactor Problems, 1965)
25. H. Haario, E. Saksman, and J. Tamminen, “An adaptive Metropolis algorithm,” Bernoulli 7(2), 223–242 (2001).
26. H. Haario, E. Saksman, and J. Tamminen, “Adaptive proposal distribution for random walk Metropolis algorithm,” Comput. Stat. 14(3), 375–396 (1999).
27. X. Jia, X. Gu, Y. J. Graves, M. Folkerts, and S. B. Jiang, “GPU-based fast Monte Carlo simulation for radiotherapy dose calculation,” Phys. Med. Biol. 56(22), 7017–7031 (2011).
28. S. Hissoiny, B. Ozel, H. Bouchard, and P. Després, “GPU/MCD: A new GPU-oriented Monte Carlo dose calculation platform,” Med. Phys. 38(2), 754–764 (2011).