



Computational scatter correction in near real-time with a fast Monte Carlo photon transport model for high-resolution flat-panel CT

Ammar Alsaffar¹ · Steffen Kieß¹ · Kaicong Sun¹ · Sven Simon¹

Received: 31 March 2022 / Accepted: 6 August 2022 / Published online: 24 August 2022
© The Author(s) 2022

Abstract

In computed tomography (CT), scattering causes severe quality degradation of the reconstructed CT images by introducing streaks and cupping artifacts which reduce the detectability of low contrast objects. Monte Carlo (MC) simulation is considered the most accurate approach for scatter estimation. However, the existing MC estimators are computationally expensive, especially for high-resolution flat-panel CT. In this paper, we propose a fast and accurate MC photon transport model which describes the physics within the 1 keV to 1 MeV range using multiple controllable key parameters. Based on this model, scatter computation for a single projection can be completed within a range of a few seconds under well-defined model parameters. Smoothing and interpolation are performed on the estimated scatter to accelerate the scatter calculation without compromising accuracy too much compared to measured near scatter-free projection images. Combining the fast scatter estimation with the filtered backprojection (FBP), scatter correction is performed effectively in an iterative manner. To evaluate the proposed MC model, we have conducted extensive experiments on the simulated data and real-world high-resolution flat-panel CT. Compared to the state-of-the-art MC simulators, the proposed MC model achieved a 15× acceleration on a single-GPU compared to the GPU implementation of the Penelope simulator (MCGPU) utilizing several acceleration techniques, and a 202 × speed-up on a multi-GPU system compared to the multi-threaded state-of-the-art EGSnrc MC simulator. Furthermore, it is shown that for high-resolution images, scatter correction with sufficient accuracy is accomplished within one to three iterations using a FBP and the proposed fast MC photon transport model.

Keywords X-ray computed tomography · Scatter correction · Monte Carlo (MC) simulation · Photon transport model

1 Introduction

Computed tomography (CT) imaging is a powerful non-destructive testing technique. It provides inception about the inner of the scanned object and is widely used for industrial and medical applications. However, this technique suffers from severe quality degradation artifacts. Among these artifacts the scatter, which results from the change in the direction or the direction and the energy of the photon penetrating the object, degrades the quality of the CT reconstructed image by inserting cupping and streak artifacts. These artifacts reduce the contrast of this image [1, 2], and the contrast-to-noise [3–5]. MC method can estimate the scatter accurately due to the accurate modeling of the

physics involved in the photon matter interactions, the shape and the composition of the object require no simplification and could be easily simulated, the ability of this method to model any order of scatter, and requires no simplification or alteration of the real-world scanner configuration, i.e., any scan protocol and setting can be adopted by the MC method [1]. Although it is accurate, the MC method requires a huge computation time.

An accurate and fast MC model accelerated over multiple-GPU is implemented in this paper, which simulates the fundamentals physics involved within the keV range including Compton scattering, Rayleigh scattering, and photoelectric absorption. This MC model has been extensively evaluated against other MC simulators and real-world CT scanner. On the other hand, an iterative scatter correction algorithm which requires a few iterations for scatter compensation is also implemented, the latter has been embedded with the implemented MC model for fast scatter estimation. The estimated scatter from this MC model is used to correct

✉ Ammar Alsaffar
alsaffar@cis.itl.uni-stuttgart.de

¹ Institute TI, University of Stuttgart, Stuttgart, Germany

the scatter-corrupted raw projections from the real-world scanner. Moreover, smoothing and interpolation are used to accelerate the iterative scatter correction algorithm. Smoothing reduces the time needed for the MC simulation by denoising the estimated scatter acquired using a reduced number of photons, while interpolation is used to estimate a full set of projections from half the number, and to up-sampling the simulated projection from low-resolution to high-resolution. The correction result from the iterative scatter correction algorithm is in fine agreement with the near scatter-free collimator-based reconstructed data set.

1.1 Contributions

1.1.1 Implementation of an accurate and parameterized multi-GPU accelerated MC forward projection model with postprocessing

The proposed MC model incorporates basic physical properties such as Compton scattering, Rayleigh scattering, and photoelectric absorption, as well as parameterized post-processing such as filtering and resolution interpolation to reduce the computation time through slight approximations that lead to minor but tolerable deviations in the computation result. In addition, the basic parameters of the MC model, such as the step size of the ray tracing, which control the number of voxels crossed during the ray tracing, and the splitting number of photons, can be controlled to balance the trade-off between the speed-up and the accuracy of the scattering estimate. It is shown that the proposed MC model has achieved more than $15 \times$ acceleration in comparison to the GPU implementation of the Penelope simulator (MCGPU) [35] utilizing these parameters. Moreover, this MC model has been accelerated using the multi-GPU platform such that thousands of photons are simulated simultaneously. In comparison with the state-of-the-art multi-threaded CPU MC simulator EGSnrc, our model achieves $202 \times$ speed-up on a four GPUs system.

1.1.2 Fast iterative scatter correction near real-time

MC simulation is considered the gold standard for the scatter estimation [5–11, 14], due to the accurate modeling of the physics involved in the photon transport. However, it is computationally expensive. This has led to a choice of methods other than MC simulation for scatter correction for the reconstruction of volume representations in computed tomography in the past. The huge computation time is particularly true for the high resolutions in non-medical computed tomography, i.e., computed tomography for materials, engineering, and natural sciences in general. In this work, we show that by optimizing the parameters of the GPU-accelerated MC simulation and by the use of smoothing and

interpolation techniques, accurate and fast scatter correction are achieved simultaneously. This makes the MC simulation and correction of scattering suitable for integration into a real-world high-resolution flat-panel CT reconstruction. Based on this approach, we achieved a significant speed-up of MC simulation in a demonstrated use case from 45.4 to 0.28 h for 3000 projections of a computed tomography scan with a flat-panel detector of $2k \times 3k$ resolution. This computation time for MC simulation is even lower than the acquisition time required by the real-world CT scanner for the same number of projections which takes almost 0.8 h. In Ref. [15], and for the same scanner model, the reported scan time is 0.435 h for 3142 projections with an exposure time of 500 ms, a voltage of 120 kV, and a current of $90 \mu\text{A}$. In the proposed work, the scan was performed using 3000 projections with an exposure time of 1 s, a voltage of 200 kV, and a current of $50 \mu\text{A}$. Moreover, this simulation time using this model is less than the time of the non-iterative filtered backprojection (FBP). Furthermore, the achieved simulation time is very close to the time required by the fast Boltzmann equation solver methods [17–20]. However, the simulation time achieved by the proposed MC model is without the need to down-sample the detector and the voxelized volume to the same extent suggested by these methods. Considering all the above mentioned, scatter correction using the proposed MC model can be achieved in near real-time or even in real-time, i.e., faster processing time than the required acquisition time for the projections images from the real-world CT scanners [16].

Additionally, it is shown that the proposed MC model and the scatter correction algorithm provide a solution to the problem of scatter correction with sufficiently high accuracy, which is close to a nearly scatter-free measurement generated using a collimator.

1.2 Related work

Several methods have been introduced to compensate for scatter. According to [21], these methods are divided into two approaches. The first approach is the direct elimination of the scatter during the scan process by the use of an anti-scatter grid [22], bowtie filter [23], or optimizing the geometry by increasing the distance between the scattering object and detector [24–26]. These methods can only reduce the scatter in the reconstructed image due to a better CT-scanner setup and do not lead to a complete scatter correction. The second approach is by computing the scatter and its subtraction in each projection using different computational models like an analytical or empirical approach, by neural networks [11] or by MC simulation.

Recently, a new approach to estimate the scatter has been introduced [17–20]. Unlike the MC approach, which solves the Boltzmann problem of the photon transport

stochastically, these methods try to solve the problem deterministically. Using this approach, the estimation time of a single scatter projection by these methods is within seconds on the GPU platform. To reach this execution time, a severe reduction of the detector and the volume resolutions is required in these works [17, 18], this hardens the possibility to track high-frequency features of the scatter. In addition, the recent use of the neural network to estimate the scatter results in very fast estimation. Nevertheless, a significant reduction in the accuracy can be encountered when the network is used to predict the scatter in a region that is not a part of the training data set of this network or under a different scan voltage far from the training data [12]. This imposes limitations on the flexibility of the scan geometry and the type of object which can be used.

MC simulation is an accurate approach to model the scatter due to the accurate modeling of the physics involved in the photon-matter interaction. Many MC simulators have been introduced for this purpose. Among these simulators are MCNP [27], aRTist [28, 29], EGSnrc [30], Penelope [31], and others. The main drawback of the existing MC simulators is their required computational effort. To reduce the computation time, many acceleration methods are adopted. In Refs. [5, 8, 32–34], it is mentioned that the use of what is known as the variance reduction techniques, such as photon splitting, Russian Roulette, and forced detection could enhance the efficiency of the photon transport and reduce the execution time by several orders of magnitude. However, even with the extensive use of the variance reduction techniques, the MC simulation still required long computation times [35]. Single instruction multiple data (SIMD) based computation is an approach that is used to accelerate the computation through the vectorization of the data which results in an enhanced matrix multiplication [36]. The SIMD technique shows good performance when it is used to accelerate the ray-tetrahedron intersection for the MC photon transport simulator in Ref. [37]. Moreover, random number generation using the SIMD acceleration is also tested in Ref. [37]. The combination of these techniques results in a 22% speed improvement in comparison to the non-SIMD case.

Apart from the aforementioned algorithmic acceleration techniques, GPU is employed to further accelerate the MC simulation from the hardware aspect. In Ref. [35], the authors have accelerated the MC simulation in a voxelized geometry using the Penelope MC simulation physics by the use of the CUDA programming model. A speed-up factor of 27 compared to the CPU version of this simulator is achieved. The authors in Ref. [38] have implemented the simulation of the photon transport of the EGSnrc simulator on the GPU by the use of the CUDA programming model. Between 20 to 40 × speed-up is achieved depending on the number of voxels used in the simulation. The Geant4 MC simulator is the base of the work in Ref. [39] in which the

authors took advantage of the varieties of the physics available in this simulator and performed the acceleration on the GPU with a speed-up factor of 86. Variance reduction techniques have not been implemented in the aforementioned works, which imposes a limitation on the speed and the efficiency of these works [41, 53]. In contrast to the all methods mentioned above, which perform the MC simulation using a voxelized object, the method in Ref. [40] uses quadratic functions to represent the bounding surfaces of a region. As a result, a $\sim 3 \times$ higher computation time is required in comparison to the use of a voxelized geometry. A hybrid approach of using a GPU accelerated MC simulation combined with the use of the variance reduction technique is introduced in Ref. [41]. A GPU accelerated MC simulation for mega-voltage cone-beam computed tomography (MV-CBCT) is introduced in Ref. [42]. The main physics simulated using this model is the photoelectric and the Compton scattering only, while ignoring the Rayleigh scattering.

In Ref. [43] the authors have developed a GPU-based MC code for dose calculation in the radiotherapy treatment known as GPU dose planning method (gDPM) based on the original CPU-based dose calculation package dose planning method (DPM) [45]. An improved version of this work has been published in Ref. [44]. In this version, the simulations of the electron and the photon have been separated by placing the particles to be simulated into two different arrays in which the GPU simulates particles in only one array at a time. Following this approach reduces the GPU thread divergence that occurs due to the different particle transport physics for different types of particles [44, 46], which results in a faster execution time in comparison to the work in Ref. [43]. The GPU-based coupled photon–electron MC dose calculation package, GPU Monte Carlo dose (GPUMCD), has been implemented for a voxelized geometry in Ref. [47]. This work follows the same aforementioned approach to separate the transport of the electrons and the photons. The work in Ref. [48] introduces the MC photon–electron transport package GPU Monte Carlo (GMC). In contrast to the gDPM and the GPUMCD simulators, which simulate the full trajectory of an electron through their GPU kernels, the GPU kernel in this simulator transports the electron by only one step [46, 48]. The GPU-based MC photon dose code (MCPDC), introduced in Ref. [49], supports the dose calculation by simulating only the photon transport. However, the above-mentioned works are for dose calculation and radiotherapy problems only. Their main application is for treatment planning [50–52]. In contrast to the proposed MC model, the dose calculation methods can not be used for non-destructive testing in industrial applications. In addition, their implementation followed simple photon physics treatment. In this treatment, the Rayleigh scattering has been ignored, and the Compton scattering is considered to occur with free electrons only. Such an assumption does not

include the electron-atom binding effect on both the angular and energy distribution (Doppler broadening) of the scattered photon. As a consequence, they are only suitable for high-energy photons (above 1 MeV simulation). Full physics simulation has been reported in Ref. [53]. In this work, a GPU accelerated MC dose calculation code (gCTD) has been introduced. The result from the gCTD simulator is in good agreement with the EGSnrc simulator. However, in the same work, it is stated that there is no implementation of the variance reduction techniques in the gCTD simulator. As mentioned before, this affects both the convergence rate and efficiency of the MC simulator [53].

The acceleration of the MC simulation over the GPU platform has been conducted for positron emission tomography (PET) in Refs. [54, 55]. The accuracy of the GPU-based MC gPET simulation tool has been compared with the GATE 8.0 (Geant4 application for tomography emission). It is mentioned that in terms of accuracy, both simulators are very close. The work in Ref. [56] uses the GPU platform to accelerate the ray-tracing calculations that are involved in the MC simulation. Another GPU-based MC code for PET named pet aimed novel nuclear images (PANNI) is implemented in Ref. [57]. This work aims to implement a PET image reconstruction with the iterative maximum likelihood expectation maximization (MLEM) relying on the accuracy of MC to calculate the elements of a PET system matrix. The authors in Ref. [58] have proposed a fast GPU-based PET simulator named Ultra-fast MC PET simulator (UMC-PET). All the relevant physics related to the transport, emission and detection of the radiation in the PET simulation have been considered in this simulator. This includes the scatter, positron range, the attenuation inside the patient, photon interaction from the environment, and the detector response. The simulation results from this simulator are in fine agreement with the results from the PeneloPET. PET imaging is a powerful tool that is mainly used in the medical field to evaluate organs and tissues for possible diseases. Therefore the main application of the above-mentioned MC-based PET simulator is in the medical field [59], in which they do not target the industrial non-destructive testing applications as well.

Scatter correction based on MC computation is the focus of several works. In Ref. [8], the authors have used the EGSnrc simulator supported by the use of the variance reduction techniques to correct the scatter-corrupted projections iteratively for low-resolution projections. The same approach is extended for a real phantom study in Ref. [5] also for low-resolution projections. Other works based on CPU MC simulators are found in Refs. [6, 7, 60, 64]. Although the number of projections and photons used in these works is low, their simulations required long computation time beyond the applicability to high-resolution flat-panel CT and with compromised correction quality [32].

Acceleration techniques are employed to accelerate the MC simulators and the scatter correction algorithms. The iterative scatter correction in Ref. [1] is based on the fast estimation of the scatter using a low number of photons. The resultant noisy scatter estimation was then efficiently denoised by a three dimensional fitting of the Gaussian basis function. However, this approach works well for small objects only [34]. The MC-based scatter correction algorithm proposed in Ref. [65] estimates the scatter on a small number of detector nodes combined with a reduced number of projections. Linear interpolation is then used between the nodes and projections to derive the complete scatter estimation of the scan. Such an approach could not track possible high spatial frequencies in the scatter distribution especially if the interpolation grid is too coarse [34]. The fast scatter correction algorithm proposed in [32], which is based on a single-GPU MC scatter simulation and extended in Ref. [9] for multi-GPU, relies again on the usage of a very low number of photons and projections and requires the availability of a priori information such as the planning CT scan. By utilizing a GPU-accelerated MC model in Ref. [41], the scatter is corrected using a two-stage of scattering estimation which is rather expensive. The fully iterative GPU-based scatter correction algorithm in Ref. [63] is used to correct the scatter from breast examination only. Every voxel in the volume is assumed to be composed of two materials glandular and adipose. The range of the voltages supported by this work is from 10 to 50 keV with a simplified version of MC simulation, this includes the assumption of 100% detector efficiency.

Unlike most of the previously mentioned works, the scatter correction algorithm introduced here does not rely on heavily smoothing or interpolating the simulated projections. By applying certain acceleration techniques, the proposed MC model could achieve a $162 \times$ speed-up in comparison to the standard case without acceleration using the same model.

2 Material and methods

2.1 The proposed MC photon transport model

A detailed MC photon transport model through a voxelized geometry in a GPU has been implemented using the OpenCL 2.1 programming model. The implemented MC model simulates the photon physics within the 1 keV to 1 MeV range. In this model, the X-ray source is treated as a point source that emits a polychromatic spectrum. This spectrum is simulated using the Geant4 MC simulator and it is imported into this model as a discrete spectrum with several energy bins of 2 keV resolution. The total number of photons to be simulated is distributed among the energy bins according to the spectrum. Simulation is then conducted for each energy

bin sequentially. The attenuation coefficients of the materials used in the simulation are from the national institute of standard and technology (NIST) database [66]. The major physics of the photon-matter interaction such as Compton scattering, Rayleigh scattering, and photoelectric absorption have been considered in this model. The Compton scattering angular distribution was simulated using the Compton scattering PDF which is an extension of the Klein–Nishina PDF by including the form factor tabulated by Hubbel [68], this PDF is given in Eq. 1 [67].

$$\frac{d\sigma_{\text{Incoherent}}}{d\Omega} = \frac{r_0^2}{2} \left(\frac{\alpha'}{\alpha}\right)^2 \left(\frac{\alpha'}{\alpha} + \frac{\alpha}{\alpha'} - \sin(\theta)^2\right) S(q, z), \quad (1)$$

where $\Omega = 2\pi(1 - \cos(\theta))$, θ is the scatter angle of the photon, r_0 is the electron radius 2.8179×10^{-15} m, α and α' are the incident and final photon energies in units of 0.511 MeV, $\alpha = E/(mc^2)$, where m is the mass of the electron and c is the speed of light, and $\alpha' = \alpha/[1 + \alpha(1 - \cos(\theta))]$, $S(q, Z)$ is an appropriate scattering factor modifying the Klein–Nishina cross-section taken from [68] with q being the inverse length and Z being the atomic number of the material. Taking into account the scattering form factor is necessary to include the binding effect of the electron-atom on the angular distribution of the photon. Moreover, to include the binding effect on the energy distribution, the Doppler broadening of the energy of the photon is also simulated according to [69, 70]. For the Rayleigh scattering, the angular distribution is simulated using the Rayleigh scattering PDF which is the extension of the Thomson scattering by the inclusion of the scattering form factor, this PDF is given in Eq. 2 [71].

$$\frac{d\sigma_{\text{coherent}}}{d\Omega} = \frac{r_0^2}{2} \left(1 + \cos(\theta)^2\right) F(q, z)^2, \quad (2)$$

where $F(q, z)$ is a form factor modifying the energy-independent Thomson cross-section taken from [68].

To simulate the scatter event, a photon that reaches the volume from the source is transported to the first interaction point with a distance $l = -(1/\mu(E, \rho)) \log(\eta)$ [27], where μ is the linear attenuation value, E is the energy of the photon, ρ is the density of the material, and η is a random number generated. The scattering interaction type is then determined by sampling from the cumulative probability formed using the probability of each interaction type. If the photon encounters a photoelectric absorption it is then immediately terminated. Otherwise, the scattering angle is sampled, using the rejection and inversion methods [71–74], from Eq. 1 or Eq. 2 depending on the selected interaction type. Figure 1a, b show the sampling results from the Compton scattering, and the Rayleigh scattering PDFs, respectively, taken from the proposed MC model. To reduce the variance of the scattering image, the photon is split into what is known as pseudo-particles to randomly

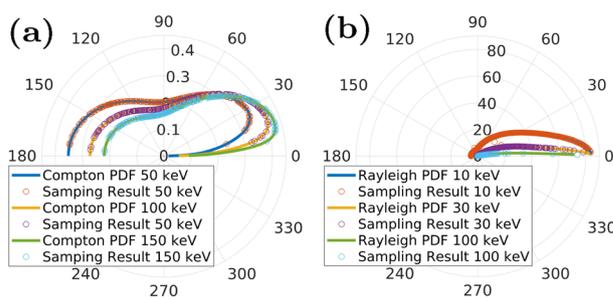


Fig. 1 Sampling from the Compton and the Rayleigh scattering PDFs. **a** Compton scattering PDFs and the sampling results. **b** Rayleigh scattering PDFs and the sampling result. Both results are for aluminum at different photon energies

selected pixels on the detector [27, 34]. The weight of the photon is then divided by the selected split number to keep the result unbiased [27]. The scoring of the scatter on the detector in this model is done by the use of the point detector method, i.e., the contribution to the detector from each interaction point is calculated using the following equation [27].

$$\chi_s = \frac{D(E)p(\theta_d)W}{2\pi L^2} \exp\left(-\int_0^L \mu(E, \mathbf{x}, \rho) d\mathbf{x}\right), \quad (3)$$

where $D(E)$ is the energy response of the detector, $p(\theta_d)$ represents the probability of scattering toward the detector given by Eqs. 4 and 5 for the Compton and the Rayleigh scattering, respectively, [27], θ_d is the cosine of the angle between the photon path and the direction to the detector [27], W is the photon’s weight, L is the distance from the interaction point to the detector. From each interaction point, the term $\exp(-\int_0^L \mu(E, \mathbf{x}, \rho) d\mathbf{x})$, where \mathbf{x} is the spatial position, is calculated by checking the material type in a specific step length which represents the number of voxels that the ray-tracer crosses. At each step, the code checks the material type and interpolates the μ value from the table of the selected material. The tables of the μ values of the relevant materials are imported to the code from external files already prepared and stored within the code directory. The voxelized volume is segmented and assigned to different materials using multi-threshold derived using the Otsu method [76] before the start of the simulation.

$$p(\theta_d) = \frac{\pi r_0^2}{\sigma_{\text{Incoh}}(Z, \alpha)} \left(\frac{\alpha'}{\alpha}\right)^2 \left(\frac{\alpha'}{\alpha} + \frac{\alpha}{\alpha'} - \sin(\theta)^2\right) S(q, z). \quad (4)$$

$$p(\theta_d) = \frac{\pi r_0^2}{\sigma_{\text{Coh}}(Z, \alpha)} \left(1 + \cos(\theta)^2\right) F(q, z)^2. \quad (5)$$

πr_0^2 is a constant and σ_{Incoh} , σ_{Coh} are the integrated incoherent and coherent cross-section, respectively, taken from [68].

The scatter intensity projection on the detector is calculated by summing up the contributions from all the interaction points of all the simulated photons according to Eq. 6. The selection of the pixel, where the scoring occurs, is done randomly.

$$I_s = \sum_0^N \sum_0^M \sum_0^K \chi_s, \quad (6)$$

where N is the number of energy bins of the polychromatic source, M is the number of photons in the energy bin, and K is the number of interactions.

Regarding the GPU implementation, two kernels were implemented in this model. The first kernel is to calculate the primary projection as given in Eq. 7 [75].

$$I_p = M \int_0^{E_m} D(E)S(E)dE \exp\left(-\int_0^L \mu(E, \mathbf{x}, \rho)d\mathbf{x}\right)g. \quad (7)$$

While the second kernel is used to calculate the scatter projection. The number of threads that are used in the simulation is 16,384, as this number gives the fastest execution. On the other hand, the number of work items per work group was determined automatically by the OpenCL implementation. In addition, the GPU MC model supports both single and double-precision with a significant enhancement of the performance regarding the speed using the single-precision format. The implemented GPU-based MC model is designed to estimate the scatter and the primary projections using a voxelized volume to enable the use of this model in the correction of the scatter artifacts iteratively [5].

This GPU-based MC model is a parameterized model in which it is not only accelerated using the multi-GPU platform but is embedded with several controllable key parameters to speed up the simulation. An example of these powerful key parameters is the ray-tracing step size, in which the number of voxels that are crossed during the ray-tracing can be controlled to reduce the simulation time. Other key parameters are the number of photons and the splitting number of the photons. To demonstrate the effect of these parameters on the simulation time, it is shown in Sect. 3.5 that the implemented model can achieve $15 \times$ speed-up in comparison to the MCGPU simulator utilizing these key parameters. Moreover, Sect. 3.6 shows how the proper adjustment of these parameters can lead to a real-time MC simulation with a simulation time that is even lower than the time required by the real scanner considering the same number of projections. From the hardware aspect, the number of projections, that are needed to be simulated, are distributed among multi-GPU. This scale the simulation time almost linearly with the number of GPUs. As a result, the simulation time for the full set of projections is divided by the number of GPUs. Moreover, the achievable speed-up is neither with the extensive

reduction in the projection and the volume resolution nor by the use of a very low number of photons and projections, which is the approach almost used in all the scatter correction methods that adopt the MC simulation to estimate the scatter [1, 5–7, 32, 60, 64, 65].

This GPU-based MC model is made a part of an in-house tool that contains several CT reconstruction algorithms and solutions. The GPU implementation of this CT tool is achieved using the OpenCL 2.1 framework. Consequently, the GPU implementation of the proposed MC model has been performed using the same OpenCL version.

The CPU version of the implemented MC model is designed to perform the simulation on a computer-aided design (CAD) surface model using triangular elements. This version of the implemented MC model is embedded by a fast highly vectorized ray-tracer algorithm named Embree [61]. This MC version is also parallelized using multi CPU threads. This parallelization on the CPU platform is implemented using the OpenMP framework [62]. However, being independent of the CAD model provides numerous advantages if the estimated scatter is to be used in the iterative scatter correction process. First, the CAD model is not always available in all the cases or it only describes parts of the object, second, it is required a registration that should be robust against the CT artifacts, and finally, it is possible that doing the correction using the CAD model could bias the measurement toward the CAD model itself [13]. The implementation and execution of the GPU and the CPU versions of the implemented MC model are performed using a Linux system.

2.2 Applied acceleration schemes

From the hardware aspect, we have accelerated the proposed MC model using the GPU platforms. The GPU implementation is achieved using the OpenCL framework in which multiple thousands of photons are simulated simultaneously by exploiting the threads of the GPU. Regarding multi-GPU implementation, the number of projections is equally assigned to the available GPUs so that these projections can be simulated simultaneously.

In addition to the GPU acceleration, other techniques are utilized to accelerate the scatter correction process. To reduce the variance in the scatter image and enhance the speed, the variance reduction techniques are used [34]. Examples of these techniques are splitting and Russian Roulette. Splitting reduces the variance in the image by distributing any high contribution on the detector among many pixels. Russian Roulette is used to discard some of the split particles that have low contributions to the result overall to save the time of the expensive tracking process [27].

Moreover, the ray-tracing through the voxelized volume is an expensive method, as millions of voxels need to

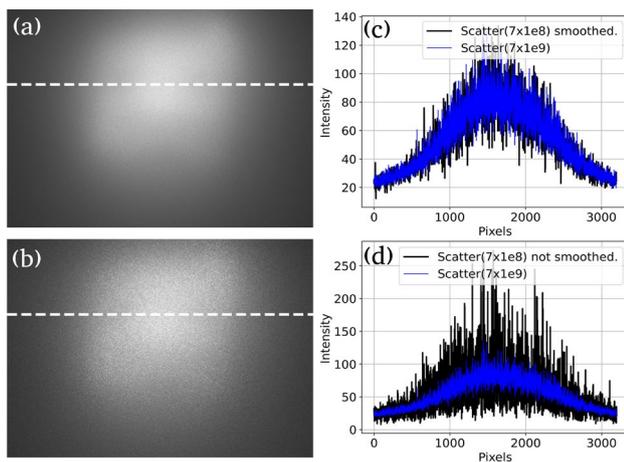


Fig. 2 The effectiveness of the smoothing operation using the Savitzky–Golay filter. **a** Scatter projection from the proposed MC model using 7×10^9 photons. **b** Scatter projection from the proposed MC model using 7×10^8 photons smoothed using this filter. **c** The white dashed profile lines of **a** and **b**. **d** Profile lines of **a** and the scatter result from the proposed MC model using 7×10^8 photons without smoothing (not shown here)

be crossed by these rays to score the values on the detector. Therefore, the number of voxels of the original volume is down-sampled from $3164 \times 2304 \times 3164$ voxels to $791 \times 576 \times 791$ voxels to fit the GPU first, and to reduce the number of voxels that the ray-tracer crosses. In addition, we have shown, in Sect. 3.6, that instead of looping on every voxel and checking the material type during the ray-tracing, one can adjust the loop to make the ray crosses several voxels before checking the material type through the use of a higher step size. This has led to a significant improvement in the simulation time while maintaining the accuracy of the result. A study of the effect of different step sizes on the accuracy of the scatter correction can be found in Sect. 3.6.

Interpolation techniques have been also used. First, cubic interpolation is used to up-sample the simulated primary and scatter projections from 576×800 pixels resolution to 2304×3200 pixels resolution; second, only half the number of scatter projections is simulated using the proposed MC model, and the other half is generated using the linear interpolation.

On the other hand, the smoothing operation enhances the estimation speed of the scatter by more than 50% by reducing the number of photons required in the MC simulation [8]. Many previous works adopt smoothing filters [1, 78]. The EGSnrc simulator uses the Savitzky–Golay filter [8, 79]. The Savitzky–Golay filter is used to denoise the noisy scatter estimates in this work, as this type of filter can preserve the high-frequency components of the smoothed image [80]. Figure 2 shows that the profile line from the denoised scatter projection fits well with the profile line of the scatter projection acquired with a $10 \times$

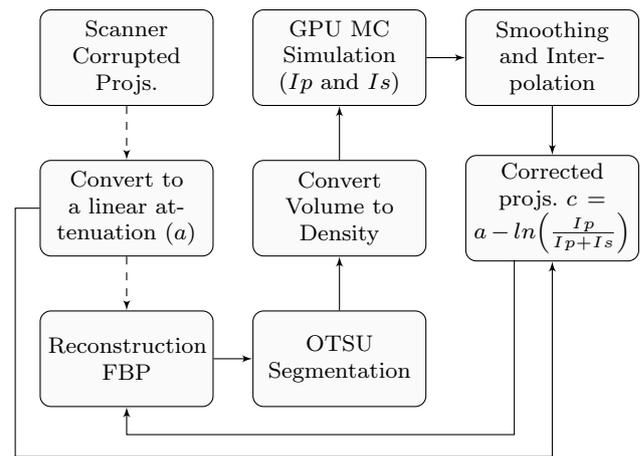


Fig. 3 Flowchart of the iterative scatter correction algorithm. The dotted line is a one time execution

higher number of photons in comparison to the first case [8].

2.3 Simulation of the polychromatic behavior

To achieve an accurate CT scan simulation, the polychromatic behavior of the source and the detector should be accurately simulated. The simulations of the two have been performed using the Geant4 MC simulator taking into account the internal construction and the components setup of both.

2.4 Iterative scatter correction

The iterative scatter correction algorithm based on a fast FBP and the proposed fast photon transport model is shown in Fig. 3. Usually, one to three iterations are required for the scatter correction in considered cases shown later.

The algorithm starts by importing the scatter corrupted raw intensity projections from the scanner. These projections are of 2304×3200 pixels resolution. The intensity projections are then converted into linear attenuation projections and reconstruction using the FBP method is then performed to get a volume using these projections. The reconstructed volume is down-sampled to $791 \times 576 \times 791$ voxels to fit the GPU. The different materials in the down-sampled volume are segmented using multi-threshold derived from the Otsu method and then converted into densities. A MC simulation is then applied, using the proposed MC model, to acquire the scatter and the primary intensity projections using the voxelized volume. These projections are then used in Eq. 8 to perform the scatter correction in the iterative algorithm.

$$c = a - \ln \left(\frac{I_p}{I_p + I_s} \right), \quad (8)$$

where c is the corrected projection, a is the scatter-corrupted linear attenuation projection from the scanner, and I_s , I_p are the scatter and the primary intensity projections taken from the proposed MC model, respectively. As mentioned before, this work aims to correct high-resolution projection from the scanner. Two approaches can be followed in the iterative scatter correction algorithm. The first approach is to calculate the scatter and the primary projections using the proposed MC model with the same resolution of 2304×3200 from the scanner. These projections should be denoised by the smoothing filter before using them in the correction step. This approach is time-consuming, as performing the MC simulation on a high-resolution detector requires a high number of photons to get a low noise projection. The second approach is to calculate the scatter and the primary projections using a lower resolution, i.e., 576×800 with less number of photons. These projections are then up-sampled to the high-resolution case using cubic interpolation to prepare them for the correction step.

In CT, a high number of projections should be used to get a proper reconstruction quality using the FBP method. Simulating the scatter for all these projections is timely expensive. Since the scatter tends to change slowly between projections [77], only half of the projections are simulated and the other half is acquired using linear interpolation between every two projections.

The primary and the scatter intensities projections, prepared in the previous step, are used in the initial correction of the projections from the scanner using Eq. 8. This initial correction lacks accuracy since the estimation of the scatter and the primary intensities in this step has been done using a scatter-corrupted volume.

The correction process is repeated again using the partially corrected projections until an adequate correction is derived. Different parameters of the MC setting were tested for the sake of optimizing the time. It is shown that certain settings of these parameters minimize the MC simulation time to the lowest extent while maintaining the accuracy of the scatter correction. This is one of the main differences between the iterative scatter correction algorithm given in Fig. 3 compared to previous works [5, 8]. Further differences are the low-resolution, low number of photons, low number of projections, and the CPU instead of GPU-based scattering corrections.

The iterative scatter correction algorithm has been accelerated using multi CPU threads. Only the MC simulation, as it is the most time-consuming part of the algorithm, has been accelerated using multi-GPU.

3 Experiments and results

In this section, the proposed MC model is evaluated on real-world data sets and compared with other simulators. The real-world datasets were acquired using the Nikon scanner available at our institute for different test objects. In particular, several examples of cement-based materials were acquired using objects prepared also at our institute. In addition, results of the scatter correction algorithm and comparison of these results with and without the use of different acceleration and optimization techniques are also shown for different experimental objects.

3.1 Collimator scatter-suppressed results

Two copper blocks of $2 \text{ cm} \times 2 \text{ cm} \times 4 \text{ cm}$ dimension have been used as a simple collimator in this work. These blocks were placed on top of each other in front of the X-ray source in which the long side was positioned perpendicularly with the exit window of the X-ray source with an opening slit of 1 mm thickness. As a result, the original cone beam is converted into a fan beam and the scanned slit with this fan beam is used as ground truth in this work.

3.2 Verification of the proposed MC model

Several methods were used to verify the proposed GPU-based MC model. First, the model is used to correct a scatter-corrupted projection from the real scanner. This correction result is compared with the experimental near scatter-free projection acquired experimentally using the collimator.

Second, the model is used to correct a scatter-corrupted projection from the EGSnrc simulator in which the result is compared with the scatter-free projection available from this simulator.

Figure 4 shows the result of the scatter correction of the scatter-corrupted linear attenuation projection from the scanner. The correction was done using Eq. 8 utilizing the scatter and the primary projections simulated by the proposed MC model. The result of the correction of this projection matches well with the near scatter-free result of the same scan acquired using the collimator.

On the other hand, the result of the scatter correction of the scatter-corrupted projection from the EGSnrc simulator is shown in Fig. 5. It is shown that the scatter correction result matches the scatter-free projection from this simulator.

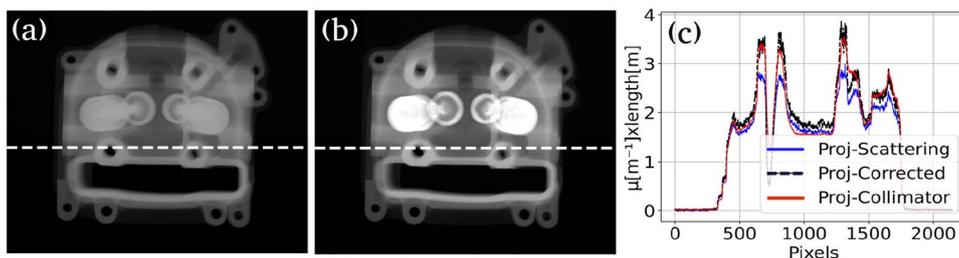


Fig. 4 Correction of a scatter-corrupted attenuation projection acquired from the scanner for an aluminum motorcycle cylinder head. **a** Scatter-corrupted projection. **b** Scatter-corrected projection. **c** The

white dashed profile lines of **a**, **b**, and from the near scatter-free projection acquired using a collimator

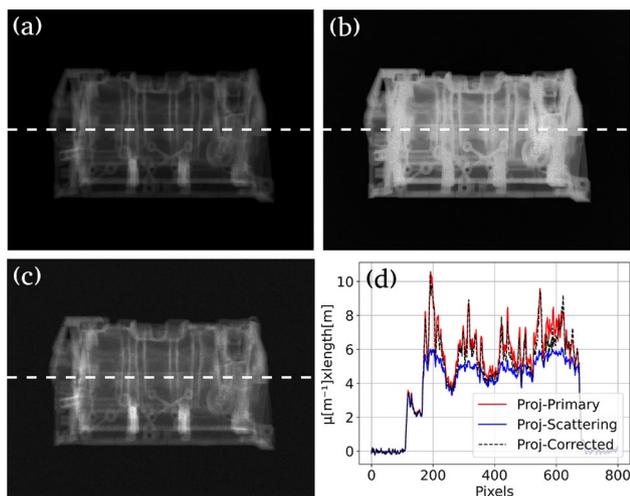


Fig. 5 Correction of a scatter-corrupted attenuation projection from EGSnc simulator for an iron engine. **a** Scatter-free projection. **b** Scatter-corrupted projection. **c** Result of the scatter correction using the proposed MC model. **d** The white dashed profile lines of the images in **a–c**

3.3 Scatter correction for motorcycle cylinder head

The scatter is corrected for an aluminum motorcycle cylinder head using the proposed iterative scatter correction method with three iterations only. 1500 scatter and 3000 primary projections were simulated using the proposed MC model with 2.5×10^8 photons and a 576×800 resolution. The scatter projections were linearly interpolated to 3000 projections and then up-sampled along with the primary projections to the same resolution of the scanner before using them in the iterative scatter correction method. Figure 6 shows the results of the third iteration of the scatter correction. As shown in the figure, the results of the scatter correction using the iterative algorithm coincide with the collimator results.

The scatter correction results in each one of the three iterations are shown in Fig. 7 (the collimator result is not shown in this figure). This figure shows that some parts of the aluminum material were wrongly corrected as steel in the first and the second iterations, as the thresholds derived by the Otsu method fail to distinguish between the steel and the aluminum

Fig. 6 Results of the third iteration of the iterative scatter correction algorithm. **a** Slice from the scatter-corrupted volume (front view). **b** Same slice from the corrected volume. **c** The white dashed profile lines of **a**, **b**, and the near scatter-free volume’s slice from the collimator. **d** Slice from the scatter-corrupted volume (side view). **e** Same slice from the corrected volume. **f** The white dashed profile lines of **d**, **e**, and the near scatter-free volume’s slice from the collimator. The profile lines in this example are averaged over multiple rows to suppress the noise

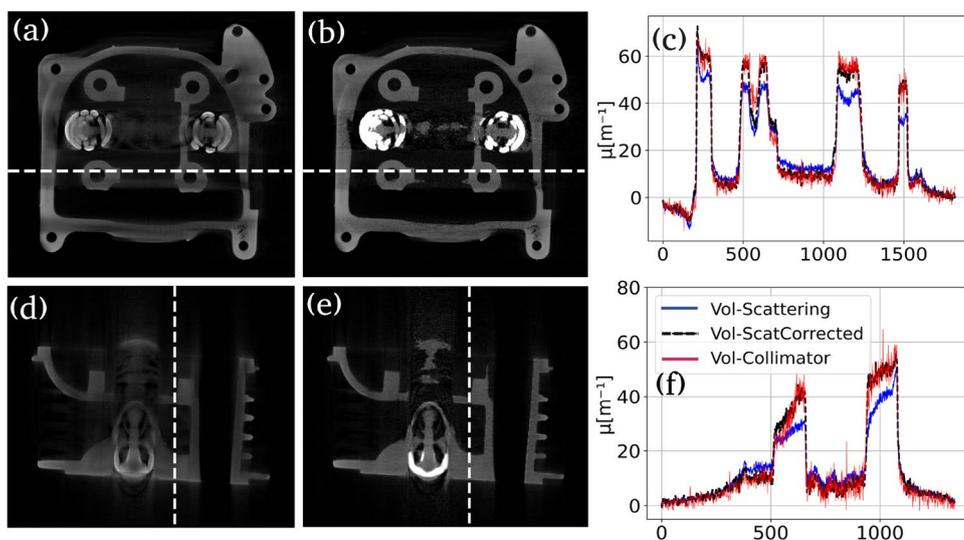
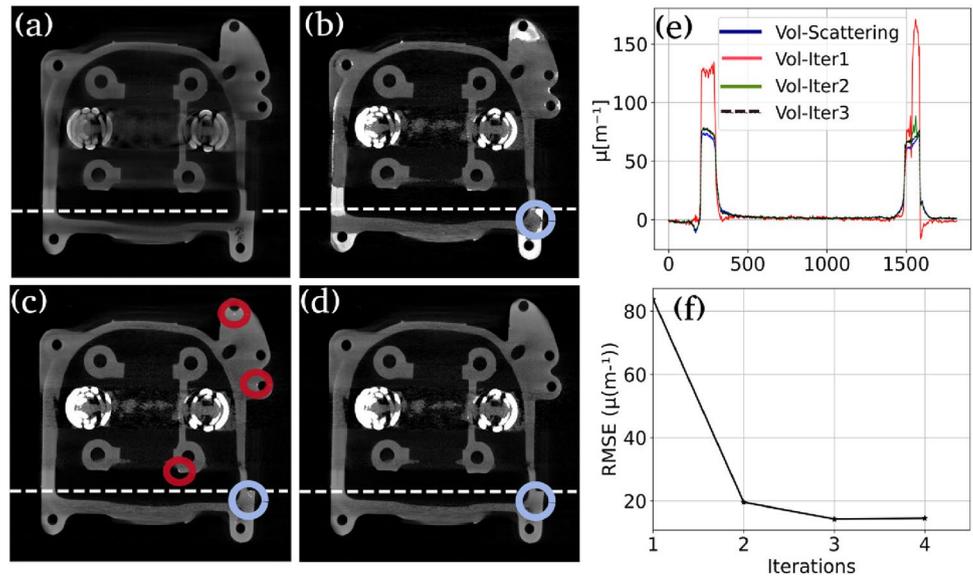


Fig. 7 Results of the three iterations of the iterative scatter correction algorithm. **a** Slice from the scatter-corrupted volume. **b** Result of the first iteration. **c** Result of the second iteration (the red and blue circles mark the regions which are still considered wrongly as steel while they are aluminum). **d** Result of the third iteration. **e** The white dashed profile lines of **a–d**. **f** RMSE values as a function of scatter correction iteration number calculated in the regions marked by the blue circles in **b–d**, and from the result of the fourth iteration. The profile lines in this example are averaged over multiple rows to suppress the noise



materials due to the severe scatter artifacts. However, the third iteration shows a good scatter correction result has been achieved.

In Ref. [5], it is suggested that the result of each iteration can be quantified with the reference scatter-free result by the use of the square root of the mean square error (RMSE), given in Eq. 9, of the reconstructed voxels. Figure 7f shows the plot of the RMSE values for all the iterations. These RMSE values are calculated for the regions marked with blue circles in Fig. 7b–d, and from the result of the fourth iteration (not shown here). The scatter-free reference image, used in Eq. 9, is taken from the collimator. According to Fig. 7f, the fourth iteration gives almost the same RMSE value as the third iteration. Therefore, only three iterations were used to correct the scatter artifact.

$$\text{RMSE} = \sqrt{(\mu_i - \mu_{i,\text{ref}})^2}. \quad (9)$$

Here μ_i and $\mu_{i,\text{ref}}$ are the mean of the linear attenuation values in the scatter-corrected volume region of interest (ROI) and the reference near scatter-free volume ROI, respectively, for material i .

The setting of the scan parameters used in this example and the time required for the scatter correction of this object using three iterations are shown in Tables 1 and 2, respectively. In Sect. 3.6, it is shown that a near real-time scatter correction is achieved for this example following certain optimization of the MC key parameters.

3.4 Scatter correction of cement objects with steel rods

Figure 8 shows the scatter correction results of different cement-based objects. The first row shows the scatter

Table 1 Measurements parameters for the motorcycle cylinder head and the cement objects scans

Parameter	Cylinder head	Cement objects
SDD ^(a)	1.282 m	1.282 m
SOD ^(b)	0.862 m	0.907 m
Resolution (pixel)	2304 × 3200	2304 × 3200
Pixel size	0.127 μm	0.127 μm
X-ray voltage	200 kV	200 kV
X-ray current	36 μA	50 μA
Exposure time	1 s	1 s

^(a)SDD denotes the source-detector distance

^(b)SOD denotes the source-object distance

Table 2 Time required for three iterations of scatter correction using four GPUs with and without the use of the interpolation technique for the cylinder head example, the MC simulation time is for one iteration only

Method	Photons no.	Iterative corr. (s)	MC sim. ^(a) (s)
W/interp.	2.5×10^8	39,240	8110
W/o interp.	7×10^8 ^(b)	175,405	50,000

^(a)W/interpolation: 1500 scatter and 3000 primary projections with size of 576×800 ; w/o interpolation: 3000 scatter and primary projections with size of 2304×3200

^(b)A higher number of photons is used in the case of high-resolution

correction result of a cement cylinder with 11 cm outer diameter and 5 cm inner diameter with eight steel rods inserted in this cylinder. The second row shows the correction result of a 7 cm diameter cement cylinder with eight steel rods. The last row shows the result for a 7 cm diameter cement cylinder without any insertion. 3000

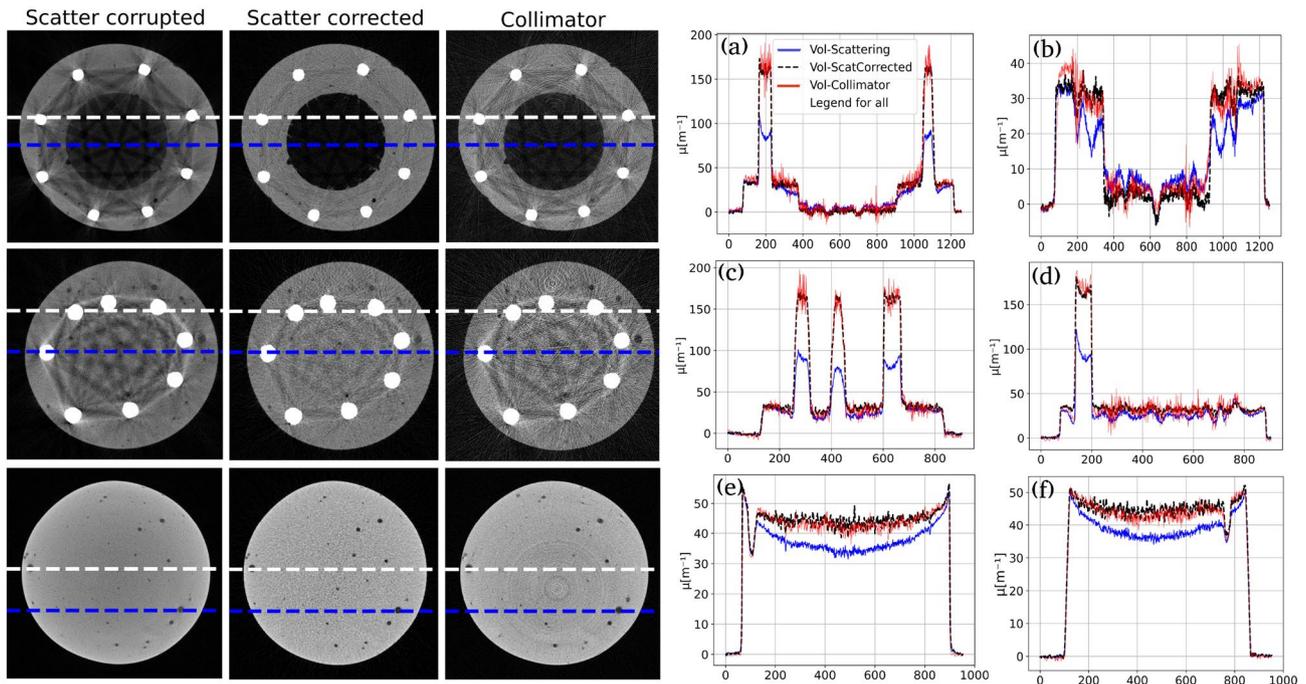


Fig. 8 Scatter correction results in different cement-based objects. **a, b** Show the white and the blue central profile lines of the images in the first row, respectively. **c, d** Show the white and the blue central

profile lines of the images in the second row, respectively. **e, f** Show the white and the blue central profile lines of the images in the third row, respectively

projections from the scanner for each one of these objects were corrected using the implemented iterative scatter correction algorithm following the same approach which is used to correct the motorcycle cylinder head object. The number of photons used in the MC simulation, in this case, was 4.9×10^8 photons.

Since the reconstructed volumes from the scanners' projections are well segmented, only a single iteration was used for the correction. The good segmentation result is a consequence of the high diversity between the linear attenuation values of the cement and the steel. Although the steel material is highly affected by the scatter in these examples, the value of the linear attenuation of the steel is still easily distinguishable from the one of the cement. The primary and the scatter projections, simulated using the proposed MC model in this single iteration, were accurate enough for the scatter correction process.

The results of the scatter correction of these objects resemble the near scatter-free results from the collimator. Besides, it is also shown that the scatter correction successfully reduces the cupping and the streak artifacts that are severely affecting the original volumes from the scanner. Table 1 shows the parameters of the scan which are used to acquire the projections from the scanner. Table 3 shows the time required for a single iteration of the scatter correction and for the MC simulation of the three objects.

Table 3 Time required for a single iteration of scatter correction for the cement objects using four GPUs with and without the use of the interpolation technique

Method	Objects	Photons no.	Iterative corr. (s)	MC sim. (s)
W/interp.	Object1	4.9×10^8	19,615	16,606
	Object2	4.9×10^8	22,861	19,982
	Object3	4.9×10^8	18,727	15,950
W/o interp.	Object1	1.9×10^9	72,473	69,464
	Object2	1.9×10^9	84,323	81,444
	Object3	1.9×10^9	65,409	62,632

3.5 Evaluation of execution time of the proposed MC model

Three different objects were used to evaluate the required execution time of the CPU and the GPU versions of the MC model against other CPU-based simulators. Object (3) is shown in Fig. 5, objects (1) and (2) are not shown here however, the occupation of these objects on the screen is the same as the object in Fig. 5. The dimensions of the objects are $18.4 \text{ cm} \times 20 \text{ cm} \times 18.4 \text{ cm}$, $18.1 \text{ cm} \times 22.4 \text{ cm} \times 13.5 \text{ cm}$, and $16.7 \text{ cm} \times 13.7 \text{ cm} \times 9.8 \text{ cm}$ for object (1), (2) and (3),

Table 4 Time for the execution on different platforms using a single-GPU for the MC model and a single thread for the CPU models

	GPU model (s)	CAD model (s)	EGSnrc (s)	aRTist
Object(1)	50	840	8500	1008
Object(2)	44	790	15,300	832
Object(3)	44.2	1750	4400	1218

Table 5 Time for simulating 3000 projections on different platforms using four GPUs for the proposed MC model and 12 threads for the CPU models (no interpolation applied)

	GPU model (h)	CAD model (h)	EGSnrc (h)	aRTist (h)
Object(1)	10.4	111	1011	108
Object(2)	9	105	1820	88.3
Object(3)	9.2	209	523	120.8

respectively. The platform of the execution of the CPU simulators is Intel(R) Core(TM) i7-5820k CPU 3.30 GHz, while a GeForce GTX 1080 Nvidia GPU has been used for the proposed MC model. The number of photons used in the simulation was 4.2×10^8 photons from 190 energy bins. The detector resolution in these simulations was 576×800 pixels. The evaluation of the execution time was done with two different cases.

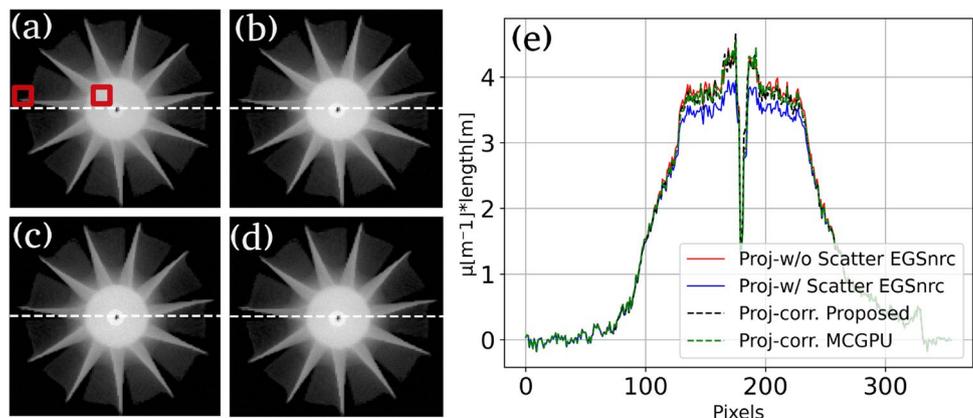
In the first case, a single thread was used in the simulations of the CPU simulators, whereas a single GPU was used in the simulation of the proposed MC GPU model. Table 4 shows the time required for a single projection. Considering the comparison case of the object (2), $347 \times$ acceleration is achieved compared to the EGSnrc simulator, while more than $18 \times$ acceleration is achieved compared to the aRTist simulator using the GPU version of the MC model. Compared to the CPU version of the implemented MC model, a $19 \times$ acceleration is achieved compared to the EGSnrc simulator, and with a slightly better performance compared to the aRTist simulator. It is worth

mentioning that the aRTist simulator is one of the fastest simulators available on the CPU platform [28].

In the second case, the evaluation is done by comparing the total simulation time required for the full set of projections. Thus, 3000 projections were simulated on four GPUs using the proposed MC model. Whereas 12 threads were used in the simulation of the CPU simulators. Table 5 shows the time required for the simulation. For the comparison case of the object (2), the multi-GPU implementation achieves a $202 \times$ and $9 \times$ acceleration compared to the EGSnrc simulator and the aRTist simulator, respectively. While using the CPU version, a $17 \times$ acceleration has been achieved compared to the EGSnrc simulator. In comparison to the aRTist simulator, the CPU version is slightly slower. This slow performance is due to the use of some atomic operations in the OpenMP implementation of our CPU version. On the other hand, a lower speed-up factor is achieved using the GPU MC model in this case, as the use of the multi-GPU imposes an extra time to copy the data from all the devices to the CPU memory and merge them, which is a time-consuming process. It is worth mentioning that the achieved speed-up factor is not the same for all the objects. It is shown that the performance using object (2) is better than the others. This is related to the occupation of the object on the detector and how complex the object is, as ray tracing through complex objects requires more time.

Moreover, the execution time of the proposed GPU-based MC model has been evaluated against the MCGPU simulator. The two GPU-based models were used to correct a scatter-corrupted projection from the EGSnrc simulator. Figure 9 shows that the scatter correction results from the two models match the scatter-free projection from the EGSnrc simulator. Regarding the speed and the accuracy, the proposed MC model achieved a scatter correction result with a better contrast-to-noise ratio (CNR) and speed compared to the MCGPU simulator considering the use of other acceleration methods in addition to the GPU platform. In contrast to the MCGPU model, the proposed MC model supports the usage of variance reduction and other acceleration techniques to accelerate the acquisition of the projections. By

Fig. 9 Scatter correction using the MCGPU and the proposed GPU model. **a** Scatter-corrupted projection from the EGSnrc simulator. **b** Scatter-free projection from the EGSnrc simulator. **c** Corrected projection using the proposed MC model. **d** Corrected projection using the MCGPU. **e** The white dashed profile lines of **a–d**. The red squares represent the ROI and the background regions used to calculate the CNR



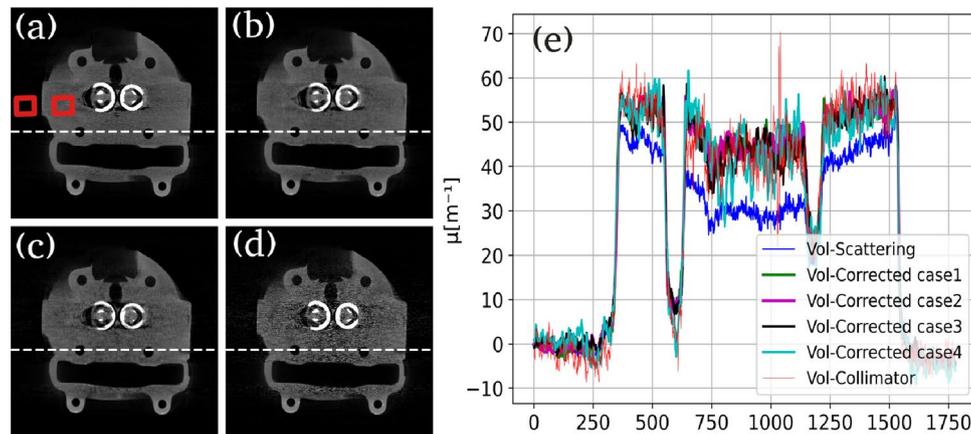


Fig. 10 Correction results with different parameters' optimization. **a** Scatter-corrected case 1. **b** Scatter-corrected case 2. **c** Scatter-corrected case 3. **d** Scatter-corrected case 4. **e** The white dashed profile lines of the images in **a–d**, from the near scatter-free volume's slice

utilizing these techniques, i.e., a photon's splitting number of 30, and a step size of four, and with 5×10^8 photons, the proposed MC model simulates the primary and the scatter projections in 18 seconds using a single-GPU. These projections produce a scatter correction result, shown in Fig. 9c, with a CNR of 68.19. This CNR is very close to the one of the scatter-free projection from the EGSnc simulator which is 69.43. On the other hand, by utilizing 10^{10} photons and with a simulation time of 280 seconds, the MCGPU model produces a scatter correction result, shown in Fig. 9d, with a CNR of 67.96. Thus, using the proposed acceleration techniques, better image correction quality and more than $15\times$ acceleration have been achieved compared to the MCGPU simulator.

3.6 Optimization of the runtime of MC simulation

In this work, key parameters are made controllable to optimize the runtime for the MC simulation as it is the most expensive part of the correction process. The MC simulation alone consumes almost 85% of the complete simulation time required for the iterative scatter correction process shown in Fig. 3. Table 6 shows five cases of simulations. The first column shows the setting of the standard MC simulation. In this standard case, there are no optimization of the key parameters, no smoothing, and no interpolation techniques used. This approach results in a very long computation time for the MC simulation and the scatter correction algorithm. In the rest of the simulation cases, both the interpolation and the smoothing techniques are used to accelerate the simulation in addition to the optimization of the MC key parameters. Columns 2–5 show different settings of these key parameters. These settings gradually decrease the simulation time of both the MC simulation and the scatter

from the collimator, and from the scatter-corrupted volume. The profile lines in this example are averaged over multiple rows to suppress the noise. The red squares represent the ROI and the background regions used to calculate the CNR

Table 6 Summary of the simulation parameters and the time required for the MC simulation and the scatter correction for the test cases (1–4)

	Standard	Case 1	Case 2	Case 3	Case 4
FBP Projs.	3000	3000	3000	3000	3000
FBP time	1670 s	1670 s	1670 s	1670 s	1670 s
Photons	2.4e9	2.4e8	1.2e8	8.5e7	5e7
Splitting	20	20	10	5	10
Step size	1	1	2	3	2
Projs. sim.	3000	1500	1500	1500	1500
MC/proj. ^(a)	218 s	21.6 s	4.6 s	2.7 s	2 s
MC/iter. ^(b)	45.4 h	2.25 h	0.48 h	0.28 h	0.2 h
Corr./iter. ^(b)	46.3 h	3.6 h	1.9 h	1.7 h	1.2 h

^(a)This represents the simulation time of a single projection over single-GPU

^(b)The simulations are performed using four GPUs

correction algorithm. The number of projections simulated in the simulation cases shown in columns 2–5 is only 1500 projections. In simulation case 1, and as a comparison to the standard case given in column 1, the number of photons is divided by 10, while the step size of the ray-tracer and the splitting number of photons are not changed. The time for the MC simulation in this case for the 1500 projections is 2.25 h. For simulation case 2, the number of photons is reduced further in comparison to the standard case and case 1. Moreover, the step size of the ray-tracing is increased, i.e., we use 2 instead of 1. In addition, the splitting number of the photons is changed from 20 to 10 only. The MC simulation time for case 2 is 0.48 h. We further decreased the number of photons in case 3, the splitting number of photons is reduced from 10 to 5 only, and the step size is increased to 3. The MC

Table 7 Quantitative evaluation of the scatter correction results for the test cases (1–4)

	Corrupted	Case 1	Case 2	Case 3	Case 4
CNR	13.1	16	18.5	13.5	7
MSE	–	–	74.8	84.7	117
NCC	–	–	0.956	0.951	0.937

simulation time is reduced to only 0.28 h using these parameter settings. Considering the last simulation case, case 4, the number of photons and the splitting number of photons are both reduced in comparison to cases 1–3, while the step size is set to 2. The simulation time is recorded as 0.2 h only.

The quality of the result of the scatter correction has been assessed using quantitative evaluation methods such as the CNR, the mean square error (MSE), and the normalized cross-correlation (NCC). The results of this evaluation are shown in Table 7. According to this evaluation, the results of both cases 2 and 3 which are shown in Fig. 10b, c, respectively, produce good scatter correction quality with a great enhancement of the computation time in comparison to case 1 (shown in Fig. 10a) and the standard case (not shown). Case 3, for example, can simulate a single scatter projection within 2.7 seconds only, whereas the full set of the projections is calculated within 0.28 h using four GPUs in near real-time, which is comparable to real-world CT acquisition. This gives a $162 \times$ speed-up in comparison to the standard case (column 1) which requires 45.4 h to acquire the full set of projections. Although further optimization of the key parameters in case 4 results in less required computation time in comparison to the other cases, the result of the scatter correction, in this case, is noisy as shown in Fig. 10d.

4 Conclusion

In this work, a multi-GPU accelerated MC photon transport model has been implemented and embedded into the iterative scatter correction algorithm for high resolution flat-panel CT. Especially, fundamental physics including Compton scattering, Rayleigh scattering, and photoelectric absorption are implemented in the proposed MC model. The MC model is accelerated by splitting the projections equally over multi-GPU to enable a full parallelization. In addition, several key parameters are made adjustable in this model. It is shown that certain adjustments of these key parameters enhanced the speed of the scatter simulation while maintaining accuracy. The most effective factor in the speed-up achieved using the proposed MC model is the use of the multi-GPU in conjunction with the above-mentioned key

parameters speeding up the simulation significantly with a proper adjustment. Moreover, the MC code itself is efficiently written, by excluding the multi-GPU acceleration and the key parameters from the count, the CPU version itself achieves a comparable speed to the aRTist simulator, which is one of the fastest simulators on the CPU platform. To validate the speed and accuracy of the proposed MC model, comparisons have been conducted with the state-of-the-art MC simulators, i.e., EGSnrc, aRTist, MCGPU, and with the real-world scanner. In comparison with the multi-threaded MC simulator EGSnrc, a $202 \times$ speed-up is achieved for a 2304×3200 pixels detector using four GPUs. Furthermore, using the implemented iterative scatter correction algorithm, we show the possibility to perform the correction of the scatter artifact iteratively in near real-time. Compared to the reference images acquired using the collimator, the scattering artifacts are effectively suppressed within one to three iterations of the proposed scatter correction algorithm. It is shown that the proposed MC photon transport model is both sufficiently fast (real-time) compared to the acquisition time by the real-world CT scanner and to FBP and sufficiently accurate for scattering artifact correction in high-resolution flat-panel CT reconstruction. With the achieved speed using the proposed GPU MC model, the integration of this model into an iterative reconstruction method, e.g., the MLEM method, can increase its accuracy by considering the full physics in the forward projection part of the reconstruction iteration.

Acknowledgements This work was supported by the German Academic Exchange Service (DAAD, no. 57381412) and the German Research Foundation (DFG, Germany) under the DFG-project SI 587/18-1 in the priority program SPP 2187.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

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

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Zbijewski, W., et al.: Efficient Monte Carlo based scatter artifact reduction in cone-beam micro-CT. *IEEE Trans. Med. Imaging* **25**(7), 1 (2006)
- Endo, M., et al.: Effect of scattered radiation on image noise in cone beam CT. *Med. Phys.* **28**(4), 469–474 (2001)
- Siewerdsen, J., et al.: Cone-beam computed tomography with a flat-panel imager: magnitude and effects of X-ray scatter. *Med. Phys.* **28**(2), 20–31 (2001)
- Ding, G., et al.: Characteristics of kilovoltage X-ray beams used for cone-beam computed tomography in radiation therapy. *Phys. Med. Biol.* **52**(6), 1595–1615 (2007)
- Watson, P., et al.: Implementation of an efficient Monte Carlo calculation for CBCT scatter correction: phantom study, Ottawa, Canada. *J. Appl. Clin. Med. Phys.* **16**(4), 216–227 (2015)
- Jarry, G., et al.: Characterization of scattered radiation in kv cbct images using Monte Carlo simulations. *Med. Phys.* **33**(11), 4320–4329 (2006)
- Jarry, G., et al.: Scatter correction for kilovoltage cone-beam computed tomography (CBCT) images using Monte Carlo simulations. In: *Medical Imaging: Physics of Medical Imaging* (2006)
- Hing, E., et al.: Fast Monte Carlo calculation of scatter corrections for CBCT images. Ottawa, Canada. In: *Third McGill International Workshop. J. Phys. Conf. Ser.* **102** **4**(2) (2008)
- Zhang, Y., et al.: Scatter correction based on adaptive photon path-based Monte Carlo simulation method in multi-GPU platform. *Comput. Methods Progr. Biomed.* **194**, 105487 (2020)
- A general framework and review of scatter correction methods in cone beam CT. Part 2: scatter estimation approaches. *Med. Phys.* **38**(9), 5186–5199 (2011)
- Maier, J., et al.: Deep scatter estimation (DSE): Accurate real-time scatter estimation for x-ray ct using a deep convolutional neural network. *J. Nondestruct. Eval.* **37**(57), 1–9 (2018)
- Maier, J., et al.: Real-time scatter estimation for medical CT using the deep scatter estimation (DSE): method and robustness analysis with respect to different anatomies, dose levels, tube voltages and data truncation. *Med. Phys.* **46**(1), 238–249 (2019)
- Maier, J.: Artifact correction and real-time scatter estimation for X-ray computed tomography in industrial metrology. Ph.D. thesis, Heidelberg University (2019)
- Chan, H., et al.: Physical characteristics of scattered radiation in diagnostic radiology: Monte Carlo simulation studies. *Med. Phys.* **12**, 152–165 (1984)
- Warnett, J., et al.: Towards in-process X-ray CT for dimensional metrology. *Meas. Sci. Technol.* **27**, 035401 (2016)
- Wedekind, M., et al.: Real-time high-resolution cone-beam ct using GPU-based multi-resolution sampling. In: *2018 25th IEEE international conference on image processing (ICIP)*, pp. 1163–1167 (2018)
- Maslowski, A., et al.: Acuros CTS: a fast, linear boltzmann transport equation solver for computed tomography scatter-part I: core algorithms and validation. *Med. Phys.* **45**(5), 1899–1913 (2018)
- Wang, A., et al.: Acuros CTS: a fast, linear Boltzmann transport equation solver for computed tomography scatter-part II: system modeling, scatter correction, and optimization. *Med. Phys.* **45**(5), 1914–1925 (2018)
- Mingshan, S., et al.: Rapid scatter estimation for CBCT using the Boltzmann transport equation. In: *Proceedings of the SPIE*, vol. 9033 (2014)
- Shiroma, A., et al.: Scatter correction for industrial cone-beam computed tomography (CBCT) Using 3D VSHARP, a fast GPU-based linear Boltzmann transport equation solver. In: *9th conference on industrial computed tomography*, Padova, Italy (2019)
- Jin, J., et al.: Combining scatter reduction and correction to improve image quality in cone-beam computed tomography (CBCT). *Med. Phys.* **37**, 5634–5644 (2010)
- Sorenson, J., et al.: Performance characteristics of improved antiscatter grids. *Med. Phys.* **7**(5), 525–528 (1980)
- Graham, S., et al.: Compensators for dose and scatter management in cone-beam computed tomography. *Med. Phys.* **34**(7), 2691–2703 (2007)
- Neitzel, U.: Grids or air gaps for scatter reduction in digital radiography: a model calculation. *Med. Phys.* **19**(2), 475–481 (1992)
- Sorenson, J., et al.: Scatter rejection by air gaps: an empirical model. *Med. Phys.* **12**(3), 308–316 (1985)
- Persliden, J., et al.: Scatter rejection by air gaps in diagnostic radiology. Calculations using Monte Carlo collision density method and consideration of molecular interference in coherent scattering. *Phys. Med. Biol.* **42**, 155–175 (1997)
- MCNP-A General Monte Carlo N-Particle Transport Code. Version 5 Volume I: Overview and Theory, April (2003)
- Jaenisch, G., et al.: Monte Carlo radiographic model with CAD-based geometry description. *Insight* **48**(10), 618–623 (2006)
- Bellon, C., et al.: Artist analytical RT inspection simulation tool. In: *International Symposium on Digital Industrial Radiology and Computed Tomography*, Lyon, France, June (2007)
- Kawrakow, I., et al.: The EGSnrc code system Monte Carlo simulation of electron and photon transport, NRCC report PIRS (2011)
- Baro, J., et al.: PENELOPE: an algorithm for Monte Carlo simulation of the penetration and energy loss of electrons and positrons in matter. *Nucl. Instrum. Methods Phys. Res. Sect. B* **100**(1), 31–46 (1995)
- Xu, Y., et al.: A practical cone-beam CT scatter correction method with optimized Monte Carlo simulations for image-guided radiation therapy. *Phys. Med. Biol.* **60**, 3567–3587 (2015)
- Thing, S., et al.: Optimizing cone beam CT scatter estimation in egs-cbct for a clinical and virtual chest phantom. *Med. Phys.* **41**(7), 071902 (2014)
- Hing, E., et al.: Variance reduction techniques for fast Monte Carlo CBCT scatter correction calculations. *Phys. Med. Biol.* **55**, 4495 (2010)
- Badal, A., et al.: Accelerating Monte Carlo simulations of photon transport in a voxelized geometry using a massively parallel graphics processing unit. *Med. Phys.* **36**, 4878–4880 (2009)
- Altameemi, A., et al.: Enhancing FPGA softcore processors for digital signal processing applications. In: *Sixth International Symposium on Embedded Computing and System Design* (2016)
- Fang, Q., et al.: Accelerating mesh-based Monte Carlo method on modern CPU architectures. *Biomed. Opt. Express* **3**(12), 3223–3230 (2012)
- Lippuner, J., et al.: A GPU implementation of EGSnrCs Monte Carlo photon transport for imaging applications. *Phys. Med. Biol.* **56**, 7145–7162 (2011)
- Bert, J., et al.: Geant4-based Monte Carlo simulations on GPU for medical applications. *Phys. Med. Biol.* **58**, 5593–5611 (2013)
- Chi, Y., et al.: Modeling parameterized geometry in GPU-based Monte Carlo particle transport simulation for radiotherapy. *Phys. Med. Biol.* **61**, 5851–5867 (2016)
- Sisniega, A., et al.: High-fidelity artifact correction for cone-beam CT imaging of the brain. *Phys. Med. Biol.* **60**, 1415–1439 (2015)
- Myronakis, S., et al.: GPU-accelerated Monte Carlo simulation of MV-CBCT. *Phys. Med. Biol.* **65**(23), 235042 (2020)
- Jia, X., et al.: Development of a GPU-based Monte Carlo dose calculation code for coupled electron-photon transport. *Phys. Med. Biol.* **55**(7), 3077–3086 (2010)

44. Jia, X., et al.: GPU-based fast Monte Carlo simulation for radiotherapy dose calculation. *Phys. Med. Biol.* **56**(22), 7017–7031 (2011)
45. Sempau, J., et al.: A fast, accurate Monte Carlo code optimized for photon and electron radiotherapy treatment planning dose calculations. *Phys. Med. Biol.* **45**(8), 2263–2291 (2000)
46. Jia, X., et al.: GPU-based high-performance computing for radiation therapy. *Phys. Med. Biol.* **59**(4), R151–R182 (2014)
47. Hissoiny, S., et al.: GPUMCD: a new GPU-oriented Monte Carlo dose calculation platform. *Med. Phys.* **38**(2), 754–764 (2011)
48. Jahnke, L., et al.: GMC: a GPU implementation of a Monte Carlo dose calculation based on Geant4. *Phys. Med. Biol.* **57**, 1217–1229 (2012)
49. Karbalaee, M., et al.: A novel GPU-based fast Monte Carlo photon dose calculating method for accurate radiotherapy treatment planning. *Biomed. Phys.* **10**(3), 329–340 (2020)
50. Brualla, L., et al.: Monte Carlo systems used for treatment planning and dose verification. *Strahlentherapie und Onkologie* **193**(11), 243–259 (2016)
51. Onizuka, R., et al.: Monte Carlo dose verification of vmat treatment plans using Elekta Agility 160-leaf mlc. *Phys. Med.* **51**, 22–31 (2018)
52. Yamamoto, T., et al.: An integrated Monte Carlo dosimetric verification system for radiotherapy treatment planning. *Phys. Med. Biol.* **52**(7), 1991–2008 (2007)
53. Jia, X., et al.: Fast Monte Carlo simulation for patient-specific CT/CBCT imaging dose calculation. *Phys. Med. Biol.* **57**, 577–590 (2012)
54. Lai, Y., et al.: gPET: a GPU-based, accurate and efficient Monte Carlo simulation tool for PET. *Phys. Med. Biol.* **64**(3), 245002 (2019)
55. Lai, Y., et al.: Development of a GPU-based Monte Carlo simulation tool for PET. In: *IEEE Nuclear Science Symposium and Medical Imaging Conference* (2019)
56. Wang, Z., et al.: Acceleration of PET Monte Carlo simulation using the graphics hardware ray-tracing engine. In: *IEEE Nuclear Science Symposium and Medical Imaging Conference* (2010)
57. Wang, Z., et al.: GPU based Monte Carlo for PET image reconstruction parameter optimization. In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering* (2011)
58. Lahoz, G., et al.: Multi purpose ultra fast Monte Carlo PET simulator. *III jornadas RSEF/IFIMED de Física Médica* (2020)
59. Jiang, W., et al.: Sensors for positron emission tomography applications. *Sensors* **19**(22), 5019 (2019)
60. Thing, R., et al.: Patient-specific scatter correction in clinical cone beam computed tomography imaging made possible by the combination of Monte Carlo simulations and a ray tracing algorithm. *Acta Oncol. Informa Healthc.* **52**, 1477–1483 (2013)
61. Wald, I., et al.: Embree: a kernel framework for efficient CPU ray tracing. *ACM Trans. Gr.* **33**(4), 1–8 (2014)
62. Chandra, R., et al.: *Parallel Programming in OpenMP*. Elsevier, Amsterdam (2000)
63. Kim, K., et al.: Fully iterative scatter corrected digital breast tomosynthesis using GPU-based fast Monte Carlo simulation and composition ratio update. *Med. Phys.* **42**(9), 5342–5355 (2015)
64. Bertram, M., et al.: Monte-Carlo scatter correction for cone-beam computed tomography with limited scan field-of-view. *Med. Imaging Phys. Med. Imaging*, **6913**, 69131Y (2008)
65. Poludniowski, G., et al.: An efficient Monte Carlo-based algorithm for scatter correction in keV cone-beam CT. *Phys. Med. Biol.* **54**, 3847–3864 (2009)
66. Chantler, C., et al.: X-ray Form Factor, Attenuation, and Scattering Tables. *Physical Measurement Laboratory* (2005)
67. Klein, O., et al.: Über die streuung von strahlung durch freie elektronen nach der neuen relativistischen quantendynamik von Dirac. *Z. Phys.* **52**(11–12), 853–868 (1929)
68. Hubbell, J., et al.: Atomic form factors, incoherent scattering functions, and photon scattering cross sections. *J. Phys. Chem. Ref. Data* **4**(3), 471–538 (1975)
69. Namito, Y., et al.: Implementation of the Doppler broadening of a Compton-scattered photon into the EGS4 code. *Nucl. Inst. Methods Phys. Res. A* **349**, 489–494 (1994)
70. Sood, A., et al.: Doppler energy broadening for incoherent scattering in MCNP5, part II. Los Alamos national laboratory, LA-UR-04-0488 (2004)
71. Plante, I., et al.: Développement de codes de simulation 1269 Monte-Carlo de la radiolyse de l'eau par des électrons, 1270 ions lourds, photonset neutrons. Applications à divers su- 1271 jets d'intérêt expérimental. PhD dissertation, Université 1272 de Sherbrooke (2008)
72. Plante, I., et al.: Monte Carlo Simulation of Ionizing Radiation Tracks. NASA Johnson Space Center, Houston (2011)
73. Kahn, H.: Applications of Monte Carlo. RAND, Santa Monica (1956)
74. Raeside, D.: Monte-Carlo principles and applications. *Phys. Med. Biol.* **21**, 181–197 (1976)
75. Zhao, W., et al.: Robust beam hardening artifacts reduction for computed tomography using spectrum modeling. *IEEE Trans. Comput. Imaging* **5**(2), 333–342 (2019)
76. Otsu, N.: A threshold selection method from gray-level histograms. *Automatica* **11**, 23–27 (1975)
77. Ning, R., et al.: X-ray scatter correction algorithm for cone-beam CT imaging. *Med. Phys.* **31**, 1195–1202 (2004)
78. Colijn, A., et al.: Accelerated simulation of cone beam X-ray scatter projections. *IEEE Trans. Med. Imaging* **23**(5), 584–590 (2004)
79. Savitzky, A., et al.: Smoothing and differentiation of data by simplified least squares procedures. *Anal. Chem.* **36**(8), 1627–1639 (1964)
80. Chakraborty, M., et al.: Determination of signal to noise ratio of electrocardiograms filtered by band pass and Savitzky–Golay filters. *Procedia Technol.* **4**, 830–833 (2012)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ammar Alsaffar received the B.Sc. degree in electronics and communications engineering from the University of Baghdad in 2007, and the M.Sc. degree in information technology from the University of Stuttgart in 2012. He is currently a Ph.D. student at the department of Computational Imaging Systems of the University of Stuttgart's Institute of Computer Architecture and Computer Engineering. His current research interest include computed tomography (CT), image and signal processing.

Steffen Kieß received the Diploma degree in software engineering from the University of Stuttgart, Stuttgart, Germany in 2010. In 2011, he joined the Parallel Systems Department, Institute of Parallel and Distributed Systems, University of Stuttgart. His current research interests include hardware and software concepts for advanced X-ray and CT imaging.

Kaicong Sun received the B.Sc. degree in mechatronics from Tongji University in 2011, M.Sc. degree and Ph.D. degree in electrical engineering from the Technical University of Berlin in 2014 and the University of Stuttgart in 2021, respectively. His current research interests include biomedical imaging, 3D imaging and image processing.

Sven Simon received the Diploma degree in electrical engineering from RWTH Aachen University, Aachen, Germany, in 1992 and the Ph.D. degree in electrical engineering from Technical University of Munich, Munich, Germany, in 1996. He joined Siemens AG, Munich, in 1996 and Infineon Technologies AG, Munich, Germany, in 1998, focusing on hardware architectures and digital signal processing. In 1998, he became a manager for data path methodologies within the

company. In 2001, he became a Professor with Hochschule Bremen, Bremen, Germany, heading a research group for hardware architectures and computational imaging systems. In 2007, he became full Professor and Head of the Department of Parallel Systems at the Institute for Parallel and Distributed Systems, University of Stuttgart, Stuttgart, Germany. His research interest include hardware architectures and software algorithms for computational imaging systems.