 https://doi.org/10.3311/PPee.11739

Periodica Polytechnica Electrical Engineering and Computer Science, 62(4), pp. 134–143, 2018

Dynamic PET Reconstruction on the GPU

László Szirmay-Kalos[1*], Ágota Kacsó[1], Milán Magdics[1], Balázs Tóth[1]

[1] Department of Control Engineering and Information Technology, Budapest University of Technology and Economics, H-1117 Budapest, Magyar Tudósok krt. 2, Hungary
[*] Corresponding author, e-mail: szirmay@iit.bme.hu

Received: 22 November 2017, Accepted: 09 July 2018, Published online: 15 December 2018

Abstract

Dynamic Positron Emission Tomography (PET) reconstructs the space-time concentration function of a radiotracer by observing the detector hits of gamma-photon pairs born during the radiotracer decay. The computation is based on the maximum likelihood principle, i.e. we look for the space-time function that maximizes the probability of the actual measurements. The number of finite elements representing the spatio-temporal concentration and the number of events detected by the tomograph may be higher than a billion, thus the reconstruction requires supercomputer performance. The enormous computational burden can be handled by graphics processors (GPU) if the algorithm is decomposed to parallel, independent threads, and the storage requirements are kept under control. This paper proposes a scalable dynamic reconstruction system where the algorithm is decomposed to phases where each phase is efficiently mapped onto the massively parallel architecture of the GPU.

Keywords

Positron Emission Tomography (PET), GPGPU, Physics simulation

1 Introduction

With dynamic *Positron Emission Tomography* (PET), we can examine biological processes, like accumulation and emptying drugs in certain tissues. Dynamic tomography reconstructs the space-time activity density, i.e. the dynamic concentration of a radiotracer isotope. To represent the spatial dependence with finite data, the domain is decomposed to N_V homogeneous voxels.

We assume that the time dependence of the radiotracer concentration can be expressed by a common *kinetic model* $K(\mathbf{p}_V, t)$, where spatial dependent properties are encoded in an N_P dimensional vector of kinetic parameters $\mathbf{p}_V$. The primary output of the reconstruction is the parameter vector for each voxel, from which the *Time Activity Concentration* (TAC) function can be drawn or additional parameters can be computed for any point of the examined volume. One of the most important additional parameters is the *Binding Potential* (BP), which shows the metabolic rate, i.e. how strongly the tissue can accumulate the drug marked with the radiotracer isotope.

There are various alternatives for the kinetic models. For example, in the *spectral method* [1], the unknown function is assumed to be a linear combination of basis functions that are the convolutions of the *blood input function* $C_p(t)$ describing the radiotracer concentration in the blood from where diffusion can start, and exponential functions of pre-defined coefficients α_P and constrained only for positive time values t by the Heaviside step function $\epsilon(t)$:

$$K(\mathbf{p},t) = \sum_{P=1}^{N_P} a_P b_P(t), \quad b_P(t) = \left(\epsilon(t) e^{-\alpha_P t}\right) * C_p(t),$$

where the parameter vector is $\mathbf{p} = (a_1, \ldots, a_{N_P})$.

The spectral method reflects the concept that the impulse response of the biological system is a sum of exponentials multiplied by the Heaviside step function to enforce causality, and the output is the convolution of the input defined by $C_p(t)$ and the impulse response. Coefficient a_i describes the strength of diffusion at the "frequency" of α_i. The *binding potential* can be directly computed from the coefficients of the spectral method:

$$BP = -1 + \sum_{P=1}^{N_P} \frac{a_P}{\alpha_P}.$$

The *compartmental models* [2] are similar to that of the spectral method with the exception that "frequencies" α_i are also subject to reconstruction.

In PET we use radiotracer isotopes decaying with positron emission. The positron may wander a few millimeters and then may annihilate with an electron, when two

Cite this article as: Szirmay-Kalos, L., Kacsó, Á., Magdics, M., Tóth, B. "Dynamic PET Reconstruction on the GPU", Periodica Polytechnica Electrical Engineering and Computer Science, 62(4), pp. 134–143, 2018. https://doi.org/10.3311/PPee.11739

oppositely directed gamma-photons are born. The system collects the *events* of simultaneous photon incidents in detector pairs. An event is a composition of the identification of the detector pair, also called *Line Of Response* or *LOR*, and its time of occurrence.

The raw input data of the reconstruction is the list of events. If the measurement time is discretized by interval boundaries $t_1, t_2, \ldots, t_{N_F}$ and events are binned in *frames* (t_F, t_{F+1}), the time complexity reduces from the number of events to the number of *frames* N_F.

Using *concentration function* $K(\mathbf{p}_V, t)$, the expected number of radioactive decays, i.e. number of positrons generated in voxel V and in frame $[t_F, t_{F+1}]$ is

$$\tilde{x}_F(\mathbf{p}_V) = \int_{t_F}^{t_{F+1}} K(\mathbf{p}_V, t) \exp(-\lambda t) \mathrm{d}t \tag{1}$$

where λ is the decay rate of the radiotracer.

The correspondence between positron generation and gamma-photon detection is established by *system matrix* **A** that expresses the probability of generating an event in LOR L given that a positron is emitted in voxel V. The expected number of events $\tilde{y}_{L,F}$ in LOR L during frame F is the sum of the contributions of all voxels in the volume during this time:

$$\tilde{y}_{L,F} = \sum_{V=1}^{N_V} \mathbf{A}_{L,V} \tilde{x}_F(\mathbf{p}_V). \tag{2}$$

The computation of the expected values of the detector hits is called the *forward projection*.

The measured number of hits in LOR L in frame F follows a Poisson distribution of expectation $\tilde{y}_{L,F}$. Because of the statistical independence of different LORs and different frames as we required them to be disjoint, the combined probability considering all LORs and all frames is the product of elementary probabilities. According to the concept of *Maximum-Likelihood* (ML) estimation, unknown parameters are found to maximize the following log-likelihood [3]:

$$\sum_{L=1}^{N_L} \sum_{F=1}^{N_F} \left(y_{L,F} \log \tilde{y}_{L,F} - \tilde{y}_{L,F} \right). \tag{3}$$

The optimization problem has very high computational complexity. The number of free variables, i.e. the dimension of the search space is $N_P \times N_V$, the log-likelihood acting as the optimization target is a sum of $N_L \times N_F$ terms. The range of N_V and N_L is typically several hundred millions, N_F is in the order of tens, and N_P is less than 10 since we wish to describe a time function with a few parameters. Thus, both the dimension of the search space and the number of terms in the optimization target can be in the order of billions. This means that high performance computation platforms should be exploited.

This paper proposes a fully 3D reconstruction algorithm that runs on the GPU. In order to achieve this goal, we improve the algorithms, transform them according to the requirements of the GPUs, and exploit their massively parallel architecture. Our approach builds on that of Wang [4, 13] and executes the maximum-likelihood dynamic reconstruction method with a two-level iteration scheme that decomposes the solution to phases where each phase is of gathering type. Data exchange may happen just between phases. The storage complexity of the method is low, we need to represent the compressed list of events, a single LOR array, and just a voxel array in each frame, in addition to the final results, which is an array storing one parameter vector at each voxel.

2 Previous work

The state of the art and previous work on the estimation of kinetic parametric images for dynamic PET are surveyed in review articles [4, 5].

The time-consuming process of fully 3D iterative tomography reconstruction has been targeted by FPGAs [6], multi-CPU systems [7], the CELL processor [8], and GPUs, from which the massively parallel GPU has proven to be the most cost-effective platform for such tasks [9]. GPUs can be programmed with two different programming models. Shader APIs like OpenGL/GLSL or Direct3D/HLSL present the GPU hardware as the direct implementation of the incremental rendering pipeline, including both programmable and fixed processing stages. On the other hand, CUDA, StreamSDK, and OpenCL provide an access to the multiprocessors of the GPU where each multiprocessor contains a set of scalar processors organized into *warps* sharing the instruction unit, and therefore acting as a SIMD hardware.

Previous work targeted the implementation of the geometric projection step on the GPU, and the simplified involvement of other physical effects approximately as a Gaussian blur realization of the system's point spread function. If only geometric effects are considered, the shader API is appropriate for the implementation [10, 11] since the rasterizer and the alpha-blending units accessible through the shader API support these simple calculations. While attenuation correction and the incorporation of the point spread function are relatively straightforward [12],

the physically plausible scatter correction should be replaced by a simple blurring operation to stay within the constraints of the incremental rendering pipeline.

However, when more complex algorithms are implemented, the additional control of the shader processors out-weights the possibility of exploiting the fixed function pipeline elements, like clipping, rasterization, depth-buffering or alpha blending. Therefore, we build our solution on the CUDA platform. High performance implementation requires that CUDA threads run independently without communication.

In order to exploit the computational power of GPUs, we have to cope with their quasi-SIMD architecture and tailor the solution algorithm accordingly. The critical issue of the GPU programming, and parallel programming in general, is the *thread mapping*, i.e. the decomposition of the algorithm to parallel threads that can run efficiently. The following issues must be considered.

Thread divergence: A GPU is a collection of multiprocessors, where each multiprocessor has several SIMD scalar processors that share the instruction unit and thus always execute the same machine instruction. Thus, during algorithm development we should minimize the dependence of flow control on input data.

Coherent memory access and non-colliding writes: Generally, global memory access is slow on the GPU, especially when atomic writes are needed to resolve thread collisions. Particle transport needs the consideration of many sources (inputs) and many detectors (outputs). This kind of "many to many" computation can be organized in two different ways. We can take input values one-by-one, obtain the contribution of a single input value to all of the outputs, and accumulate the contributions as different input values are visited. We call this scheme *input-driven* or *scattering*. The orthogonal approach would take output values one-by-one, and obtain the contribution of all input values to this particular output value. This approach is called *output-driven* or *gathering*. GPUs and in general parallel algorithms favor gathering since it computes a single result from the available data, which can be written out without communicating between and synchronizing of the computational threads.

Reuse: Running independent threads on different processors, we cannot reuse temporary results obtained by different processors, which is obviously possible and worthwhile in a single thread implementation. To reuse partial results of other threads, the algorithm should be broken to phases. When threads implementing a phase are terminated, they write out the results to the global memory. The threads of the next phase can input these data in parallel without explicit communication.

3 Reconstruction algorithm

The reconstruction means the maximization of the log-likelihood of Eq. (3). The likelihood has an extremum where all partial derivatives are zero. Computing the partial derivatives, we obtain

$$\sum_F \frac{\partial \tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_{V,P}} \sum_L \left(\mathbf{A}_{L,V} \frac{y_{L,F}}{\tilde{y}_{L,F}} - \mathbf{A}_{L,V} \right) = 0. \tag{4}$$

for $V = 1, 2, \ldots, N_V$ and $P = 1, \ldots, N_P$. Thus, we have $N_V \times N_P$ equations, each containing N_F terms that depend on the unknown parameters of all voxels, and the computation of each equation requires the consideration of all LORs L for which $\mathbf{A}_{L,V}$ is not zero. Note that accurate reconstruction requires the computation of positron range and scattered particle paths as well, which makes system matrix $\mathbf{A}$ not sparse.

The computation of the derivatives of the log-likelihood requires a forward projection and a back projection in each frame. Indeed, in frame F, the expected number of radioactive decays in voxel V is $\tilde{x}_F(\mathbf{p}_V)$, which is forward projected to obtain $\tilde{y}_{L,F}$ according to Eq. (2).

A gathering type approach is obtained if computational threads are assigned to LORs and each thread computes the expected hits for a single LOR L during frame F. A back projection obtains a new estimate of the activity as

$$x_{V,F} = \tilde{x}_F(\mathbf{p}_V) \cdot \frac{\sum_L \mathbf{A}_{L,V} \frac{y_{L,F}}{\tilde{y}_{L,F}}}{\sum_L \mathbf{A}_{L,V}}.$$

From this equation, we can express

$$\sum_L \mathbf{A}_{L,V} \frac{y_{L,F}}{\tilde{y}_{L,F}} = \frac{x_{V,F}}{\tilde{x}_F(\mathbf{p}_V)} \cdot \sum_L \mathbf{A}_{L,V},$$

which can be substituted into Eq. (4):

$$\sum_F \frac{\partial \tilde{x}_F(\mathbf{p}_V)}{\partial \mathbf{p}_{V,P}} \left(\sum_L \mathbf{A}_{L,V} \right) \left(\frac{x_{V,F}}{\tilde{x}_F(\mathbf{p}_V)} - 1 \right) = 0. \tag{5}$$

Dividing both sides by the sensitivity of the voxel, i.e. by $\sum_L \mathbf{A}_{L,V}$, we get an equivalent requirement for the extremum of the likelihood:

$$\sum_F \frac{\partial \tilde{x}_F(\mathbf{p}_V)}{\partial \mathbf{p}_{V,P}} \left(\frac{x_{V,F}}{\tilde{x}_F(\mathbf{p}_V)} - 1 \right) = 0. \tag{6}$$

In this equation $\tilde{x}_F(\mathbf{p}_V)$ depends on the unknown parameter vector $\mathbf{p}_V$ of voxel V, while $x_{V,F}$ involves forward and back projections and depends on the parameter

vectors of all voxels. Thus, if $x_{V,F}$ were known, then the computation could be decoupled for different voxels, where a system of equations with N_P unknowns needs to be solved. In this way, forward/backward projection can be separated from the calculation of the parameter values, thus the complexity of the algorithm will be the sum of the complexities of the two steps and not their product. Having fixed $x_{V,F}$, the non-linear equation is solved, which can also be imagined as a curve-fitting process.

Concerning the evaluation and the solution of this equation on massively parallel architectures, the process should be decomposed to phases where computational threads are assigned either to voxels or LORs and can be executed without inter-thread communication.

There are various options to regularize the solution. One option is the modification of the optimization target in Eq. (3) by a regularization term that penalizes unacceptable solutions [11], where, for example, the spatial or temporal variation is too high. The *method of sieves* [14], on the other hand, does not modify the optimization target, but filters the iterated approximation in each iteration step. Filtering can also exploit user specified region information or anatomic segmentation based on the data gathered by a CT or an MR [15].

Putting the discussed steps together, the pseudo-code of the reconstruction is as follows:

for $n = 1$ **to** $n_{\max}$ **do** // *main iteration*
for $F = 1$ **to** N_F **do** // *iterate through frames*
// *evaluation: # of decays in voxels*
foreach voxel V **in parallel**

$$\tilde{x}_{V,F} = \int_{t_F}^{t_{F+1}} K(\mathbf{p}_V, t)\exp(-\lambda t)\,\mathrm{d}t$$

// *forward projection: # of hits in LORs*
foreach LOR L **in parallel** $\tilde{y}_L = \sum_{V'} \mathbf{A}_{L,V'} \tilde{x}_{V',F}$
// *back projection: # of decays correction*
foreach voxel V **in parallel** $x_{V,F} = \tilde{x}_{V,F} \cdot \frac{\sum_L \mathbf{A}_{L,V} \frac{y_L}{\tilde{y}_L}}{\sum_L \mathbf{A}_{L,V}}$
// *filtering: method of sieves*
foreach voxel V **in parallel** $x_{V,F} = \mathrm{Filter}\,(x_{V,F})$
endfor
// *curve fitting: kinetic parameters*
for each voxel V **in parallel** Solve Eq. (6)
endfor

The most time consuming steps are the forward projection and back projection. In the next section we discuss the efficient GPU implementation of these steps.

3.1 Forward projection with factoring

Forward projection is the simulation of the physical process of particle transport (Fig. 1). Emitted particles may end up in detectors after traveling in space including possible scattering and type changes. As the source and the detectors have 3D domain, and scattering can happen anywhere in the 3D space, the contribution of sources to detectors is a high-dimensional integral in the domain of source points, detector points and arbitrary number of scattering points. Such high-dimensional integrals are calculated by tracing sample paths. The more paths are computed, the higher precision simulation is obtained.

The idea of *factoring* is that the transport process is decomposed to phases with the introduction of virtual detectors (Fig. 2 shows the decomposition into two phases, but more than two phases are also possible). First the expected values in the first layer of virtual detectors are computed from the source. Then, the detectors of the first layer become sources and a similar algorithm is executed until we arrive in the real detectors.

The advantages of this approach are the followings. The calculation of a single phase can be much simpler than the complete transport process, thus we can eliminate all conditional statements that would reduce GPU efficiency. As a computed sample path ended in a virtual detector is continued by all paths started from here in the next phase, we have much higher number of sample paths to estimate high dimensional integrals, thus the result is more accurate. Note that the number of sample paths increased from 4 (Fig. 1) to 16 (Fig. 2). Considering the forward projection, which is essentially a multiplication with a huge system matrix, factoring can be imagined as decomposing the huge matrix as a product of several sparse matrices [16].

Each phase is computed in parallel on the GPU where threads do not communicate. However, the next phase can reuse the results of all threads of the previous phase, so redundant computations can be eliminated.

The disadvantage of factoring is that virtual detectors discretize the continuous space into finite number of bins, so if their number is small, discretization error occurs.

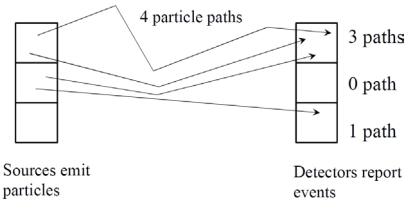

Fig. 1 Conceptual model of emission tomography. We also indicated the number of computed sample paths associated with each detector.

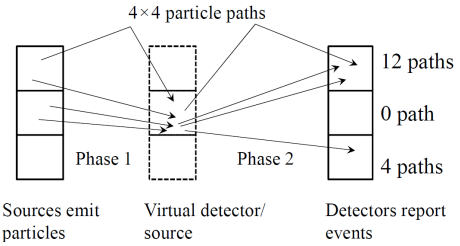

Fig. 2 Conceptual model of factoring. The particle transport process is decomposed to phases by introducing virtual detectors. The simulation of all particles is first executed to the virtual detectors, then virtual detectors become virtual sources and the second phase simulates transport from them to the real detectors.

In our proposed system, the transport process is decomposed to three factored phases:

1. *Positron transport* from the point of generation to the point of annihilation where two oppositely directed gamma-photons are born, computing annihilation density $\tilde{x}_V^a$ from positron emission density $\tilde{x}_V$.
2. *Gamma-photon pair transport* that follows the photon pair from the annihilation point to the surface of detector crystals and computes the expected number of hits on the surfaces of the crystal pairs (LORs), $\tilde{y}_L^{\text{surf}}$, from the annihilation density.
3. *Detector response* that includes all phenomena happening in the detector crystal, including inter-crystal scattering, absorption, and the sensitivity of crystals and electronics.

3.1.1 Output-driven positron transport

In order to simulate positron range, we need to compute the expected density of positron annihilation $\tilde{x}_V^a$ from the expected density of positron generation $\tilde{x}_V$ and the probability of positron migration between generation and annihilation. As the contributions of different voxels are independent and add up, the positron range calculation is similar to a spatially varying filtering

$$\tilde{x}_V^a = \sum_{V'} \tilde{x}_{V'} \mathbf{P}_{V',V},$$

where $\mathbf{P}_{V',V}$ is the probability that a positron born in voxel V' annihilates in voxel V. The actual values of these probabilities depend on the material–isotope pair. In homogeneous medium, this probability depends just on the relative position of the two voxels, thus the computation of the annihilation density is equivalent to a convolution with a larger filter kernel, which can be evaluated in frequency domain applying 3D Fast Fourier Transform, making the complexity independent of the filter kernel size.

In inhomogeneous objects, blurring kernel $\mathbf{P}_{V',V}$ also depends on the absolute locations of the two voxels. The precise treatment of this phenomenon would require the consideration of all possible positron paths, which would lead to a high-dimensional integral for every voxel pair, and would pose prohibitive computational requirements.

The idea of an approximated solution is that instead of considering the material in all points, we take into account the material type only at the beginning of the positron path [17]. This means that we blur each voxel with the filter kernel associated with the material in this voxel and ignore the fact that there might be a material boundary nearby. This simplification replaces a spatially variant filtering by several spatially invariant convolutions and a summation.

3.1.2 Output-driven photon pair transport to the detector surface

Forward projection computes the detector responses from the current emission density. Considering only the geometry, a LOR can be affected only if its detectors are seen at directions $\vec{\omega}$ and $-\vec{\omega}$ from emission point $\vec{v}$. It also means that from detector hit points $\vec{z}_1$ and $\vec{z}_2$, we can determine those emission points $\vec{v}$ and direction $\vec{\omega}$, which can contribute. Thus, the detector response can be expressed as an integral over the detector surfaces. The Jacobian of the change of integration variables is [18, 19]:

$$\mathrm{d}\omega\mathrm{d}v = \frac{\cos\theta_{\vec{z}_1}\cos\theta_{\vec{z}_2}}{\left|\vec{z}_1 - \vec{z}_2\right|^2}\mathrm{d}l\mathrm{d}z_1\mathrm{d}z_2$$

where $\theta_{\vec{z}_1}$ and $\theta_{\vec{z}_2}$ are the angles between the surface normals and the line connecting points $\vec{z}_1$ and $\vec{z}_2$ on the two detectors, respectively.

The LOR integral is expressed as a triple integral over the two detector surfaces D_1 and D_2 of the given LOR and over the line connecting two points $\vec{z}_1$ and $\vec{z}_2$ belonging to the two detectors:

$$\tilde{y}_L^{\text{surf}} = \int_{D_1}\int_{D_2} \frac{\cos\theta_{\vec{z}_1}\cos\theta_{\vec{z}_2}}{2\pi \mid \vec{z}_1 - \vec{z}_2 \mid^2}\left(\int_{\vec{z}_1}^{\vec{z}_2} \tilde{x}^a\left(\vec{l}\right)\mathrm{d}l\right)\mathrm{d}z_2\mathrm{d}z_1. \qquad (7)$$

Eq. (7) can be estimated by taking N_D uniformly distributed point pairs, $(\vec{z}_1^{(i)}, \vec{z}_2^{(i)})$ on the two detectors, and selecting N_{march} equidistant points $\vec{l}_{ij}$ along each line segment $(\vec{z}_1^{(i)}, \vec{z}_2^{(i)})$ (Fig. 3):

$$\tilde{y}_L^{\text{surf}} \approx \frac{D_1 D_2}{2\pi N_D}\sum_{i=1}^{N_D}\left(\frac{\cos\theta_{\vec{z}_1^{(i)}}\cos\theta_{\vec{z}_2^{(i)}}}{\left|\vec{z}_1^{(i)} - \vec{z}_2^{(i)}\right|^2}\sum_{j=1}^{N_{\text{march}}} \tilde{x}^a\left(\vec{l}_{ij}\right)\Delta l_i\right).$$

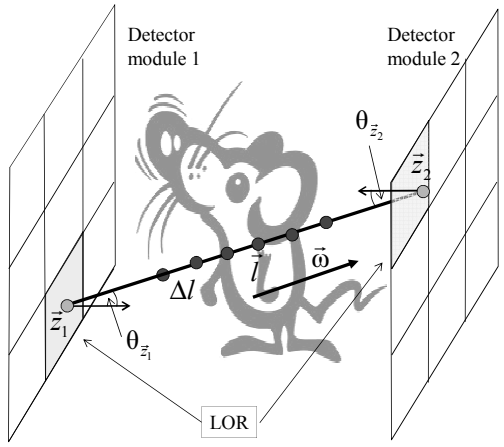

Fig. 3 During the output-driven photon pair transport calculation, a single computational thread takes a detector pair, generates N_D lines and marches on them sampling N_{march} points. This figure depicts the case when $N_D = 1$ and $N_{\text{march}} = 6$.

Evaluating the line integral with ray marching visiting regular samples has some quadrature error, but works with arbitrary finite element representation and is very fast on GPUs since it can exploit its tri-linear filtering and texture caching features.

This geometric calculation is easy to generalize to take into account *extinction cross section* σ_t due to photoelectric absorption and Compton or Rayleigh out-scattering. When the line integral of the activity is approximated with ray marching, the transmittance should also be simultaneously estimated and the integrated activity should be multiplied with the transmittance:

$$\int_{\vec{z}_1}^{\vec{z}_2} \tilde{x}^a\left(\vec{l}_{ij}\right)\mathrm{d}l \cdot \exp\left(-\int_{\vec{z}_1}^{\vec{z}_2} \sigma_t\left(\vec{l}_{ij}\right)\mathrm{d}l\right) \approx$$

$$\sum_{j=1}^{N_{\text{march}}} \tilde{x}^a\left(\vec{l}_{ij}\right)\Delta l_i \cdot \exp\left(-\sum_{j=1}^{N_{\text{march}}} \sigma_t\left(\vec{l}_{ij}\right)\Delta l_i\right)$$

The contribution of single and multiple scattering can be added to the direct contribution.

3.1.3 Output-driven detector response

Photons may get scattered in detector crystals before they get finally absorbed. The number of reported hits due to an absorption may be different in different crystals due to the variation of the *crystal sensitivity.*

In order to reduce the data needed to model detectors, we decompose this phase to two phases and separately consider the photon transport until absorption and the measurement process from the absorption to the output of the electronics. The photon transport is assumed to be translation invariant, so it can be modeled by a single, incident direction dependent blurring kernel. To handle crystal sensitivity and electronics, we assign *sensitivity* $\epsilon_{\mathbf{c}}$ to each crystal **c**, which is the expected number of events reported in this detector by the output of the measuring system, provided that a gamma-photon has been absorbed here. The sensitivity is a parameter for each detector crystal, which can be obtained by calibration measurements. Scattering inside the detectors can be modeled by a *crystal transport probability* $p_{\mathbf{i}\to\mathbf{c}}(\vec{\omega})$ that specifies the conditional probability that a photon is absorbed in crystal **c** provided that it arrived at the surface of crystal **i** from direction $\vec{\omega}$ (Fig. 4). The crystal transport probability is obtained by off-line calculation executed, for example, with GATE [20, 21].

The sum of the crystal transport probabilities is the *detection probability*, i.e. the probability that the photon does not get lost, or from a different point of view, the photon does not leave the module without absorption:

$$\nu\left(\vec{\omega}\right) = \sum_{\mathbf{c}} p_{\mathbf{i}\to\mathbf{c}}\left(\vec{\omega}\right).$$

Let us consider a LOR of direction $\vec{\omega}$ connecting crystals $\mathbf{c}_1$ and $\mathbf{c}_2$. The expected number of hits in this LOR is:

$$\tilde{y}_{L(\mathbf{c}_1,\mathbf{c}_2)} \approx \epsilon_{\mathbf{c}_1}\epsilon_{\mathbf{c}_2}\sum_{\mathbf{i}}\sum_{\mathbf{j}} \tilde{y}^{\text{surf}}_{L(\mathbf{i},\mathbf{j})} p_{\mathbf{i}\to\mathbf{c}_1}\left(\vec{\omega}_{\mathbf{ij}}\right) p_{\mathbf{j}\to\mathbf{c}_2}\left(\vec{\omega}_{\mathbf{ij}}\right) \tag{8}$$

where $\vec{\omega}_{\mathbf{ij}}$ is the direction vector between detectors **i** and **j**. We can observe that detector response is a convolution operation in 4D LOR space.

To obtain a gathering type algorithm, computational threads are assigned to filtered LORs and they fetch and weight neighboring LORs to get the filtered value.

3.2 Back projection

Back projection computes a fraction for each voxel V, where the denominator and the numerator are

$$\text{Denom} = \sum_{L=1}^{N_L} \mathbf{A}_{LV}\,, \quad \text{Num} = \sum_{L=1}^{N_L} \mathbf{A}_{LV}\,\frac{y_L}{\tilde{y}_L}.$$

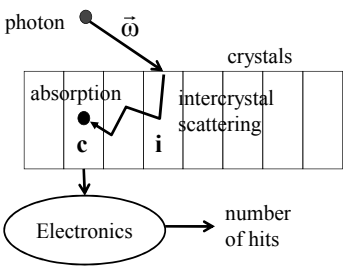

Fig. 4 Inter-crystal scattering. The photon arrives at crystal **i** but is scattered to crystal **c**, were it finally gets absorbed and detected.

In back projection the volumetric integral of the system matrix is estimated from a single position sample $\vec{v}$, and the directional integral is approximated by a single sample per detector $\mathbf{i}$, which subtends solid angle $\Delta\omega_{\mathbf{i}}$ (Fig. 5). Emission point $\vec{v}$ and point $\vec{z}_1$ on detector $\mathbf{i}$ determine direction $\vec{\omega}$. For this LOR, we get:

$$\mathbf{A}_{L(\mathbf{i},\mathbf{j}),V} \approx \frac{\Delta\omega_{\mathbf{i}}}{2\pi}.$$

For other LORs associated with detector $\mathbf{i}$, $\mathbf{A}_{L(\mathbf{i},\mathbf{j}),V} = 0$. Computational threads are assigned to voxels and each thread processes those LORs that go through this voxel.

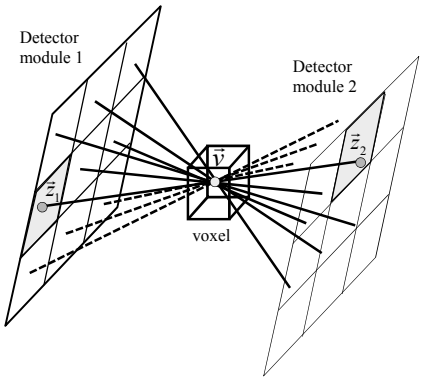

Fig. 5 A single computational thread of the output-driven back projection updates a single voxel considering all LORs crossing it.

3.3 Curve fitting

To get the parameters of a voxel, the fitting error between the parametric curve and the discrete data generated by the back projection needs to be minimized. Eq. (6) is solved iteratively linearizing the $1/\tilde{x}_F$ term in each sub-iteration, which leads to a Gauss-Newton like method. The linearization is done around current estimate $\mathbf{p}$ considering offset $\mathbf{d}$ from the current estimate as the independent parameter:

$$\begin{aligned}\frac{1}{\tilde{x}_F(\mathbf{p}+\mathbf{d})} &\approx \frac{1}{\tilde{x}_F(\mathbf{p})} + \sum_{Q=1}^{N_P} \frac{\partial 1/\tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_Q}\mathbf{d}_Q \\ &= \frac{1}{\tilde{x}_F(\mathbf{p})} - \frac{1}{\tilde{x}_F^2(\mathbf{p})}\sum_{Q=1}^{N_P}\frac{\tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_Q}\mathbf{d}_Q.\end{aligned} \tag{9}$$

After the substitution into Eq. (6) and rearrangement, we obtain a system of linear equations:

$$\mathbf{F}\cdot\mathbf{d} = \mathbf{b} \tag{10}$$

where

$$\mathbf{F}_{P,Q} = \sum_{F=1}^{N_F} \frac{\partial \tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_P}\frac{x_F}{\tilde{x}_F^2(\mathbf{p})}\frac{\partial \tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_Q},$$

$$\mathbf{b}_P = \sum_{F=1}^{N_F} \frac{\partial \tilde{x}_F(\mathbf{p})}{\partial \mathbf{p}_P}\left(\frac{x_F}{\tilde{x}_F(\mathbf{p})} - 1\right).$$

The Levenberg-Marquardt method is a robust or damped version of the Gauss-Newton algorithm. It modifies matrix $\mathbf{F}$ as

$$\mathbf{F}^{LM} = \mathbf{F} + \mu\,\mathrm{diag}(\mathbf{F})$$

where μ is increased if the error gets greater in this step and reduced otherwise. Emphasizing the diagonal of the matrix makes the approach similar to gradient search, which is more robust when the process is far from the solution. Getting closer, the method becomes similar to the Gauss-Newton technique. Matrices $\mathbf{F}$ and $\mathbf{F}^{LM}$ are symmetric and positive-definite if data points x_F are positive. Thus, the efficient Cholesky factorization can be used to solve this equation.

4 Results

The proposed reconstruction algorithm is built into a 3D fully GPU based PET reconstruction framework [22]. The tests are based on the geometry of the Mediso *nanoScan-PET/CT* device, which consists of 12 detector modules, each of them containing 81×39 pieces of crystals, with a size of $1.12^2 \times 13$ mm. The approximate number of LORs is $N_L \approx 1.8 \cdot 10^8$.

We choose the list mode for the storage of the measured data, which represents events as pairs of LOR index and time of occurrence. The projection operators of the reconstruction framework work with the binned mode representation, so switching between the two storage methods involves some extra costs. According to our experience, this conversion is negligible compared to the time of projection operators.

For the 3D tests we use a simplified version of the *Zubal brain phantom* [23]. We defined four regions with different time activity functions: blood, gray matter, white matter, and air. The measurement data were prepared by GATE simulation, where more complex physical effects like scattering were switched off.

4.1 Low activity phantom

The proposed methods are compared with non-parametric reconstruction using a low activity phantom, of resolution $128 \times 128 \times 64$. The linear size of the voxel is 0.4 mm. Unless otherwise indicated, the number of time frames is 10 and the number of iterations is 50. We observed

altogether 8 million LOR hits, which means in average 0.004 hits per LOR per frame.

Fig. 6 demonstrates the differences between parametric and non-parametric reconstructions while regularization is turned off. It can be seen that the parametric reconstruction results in significant image quality improvement compared to the non-parametric method. A similar effect can be observed in Fig. 7 when moderate anatomy based spatial regularization [15] is applied executing Gaussian filtering with standard deviation equal to the half of the linear size of the voxel. The filter built in the reconstruction loop reduced the noise. Fig. 8 compares the errors of the non-parametric versus parametric and non-regularized versus regularized reconstructions.

Figs. 9 and 10 show the TAC functions after 30 iterations for the non-regularized and regularized reconstructions, respectively. Fig. 11 presents the computed Binding Potentials.

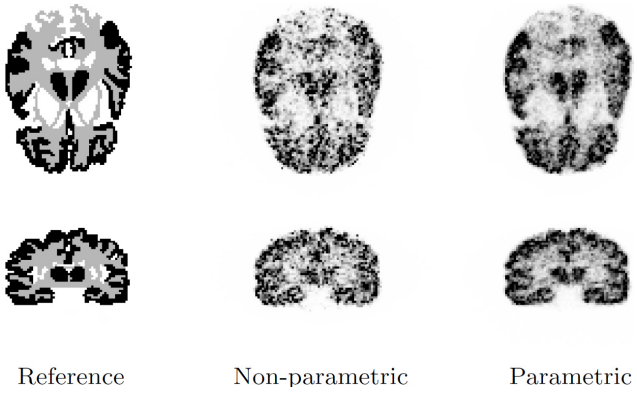

Fig. 6 Comparison of non-parametric and parametric reconstructions with no regularization in frame $F = 8$.

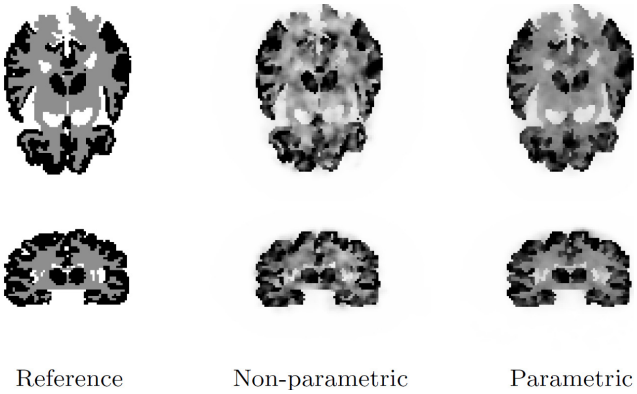

Fig. 7 Comparison of non-parametric and parametric reconstructions with spatial regularization in frame $F = 10$.

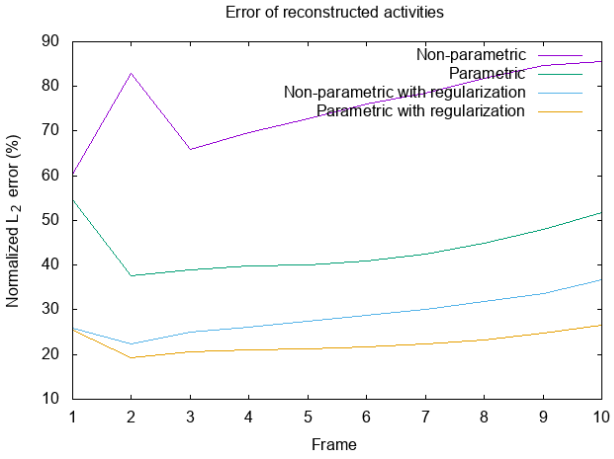

Fig. 8 Comparison of the errors of non-parametric/parametric and non-regularized/regularized reconstructions.

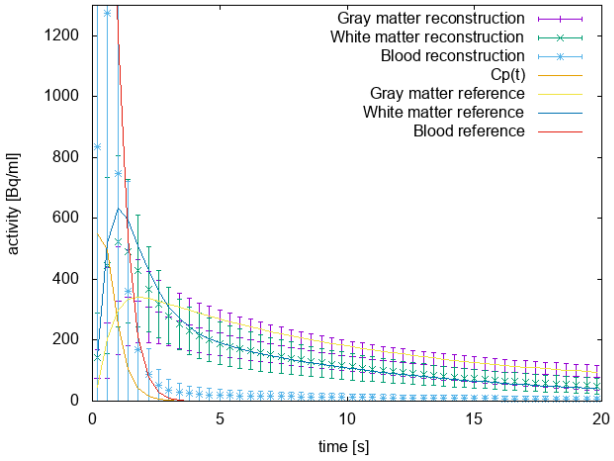

Fig. 9 TAC curves obtained by parametric reconstruction.

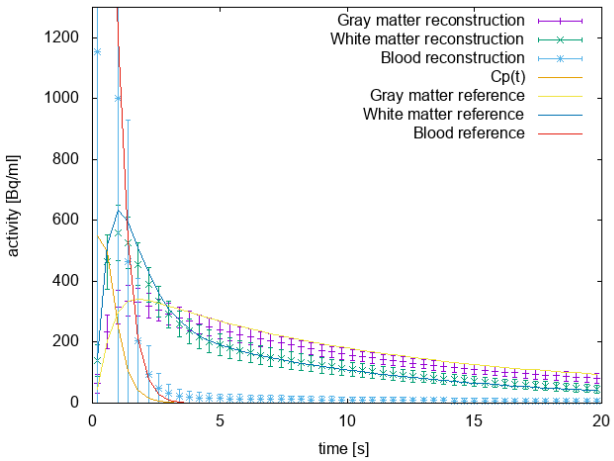

Fig. 10 TAC curves obtained by parametric reconstruction using also spatial regularization.

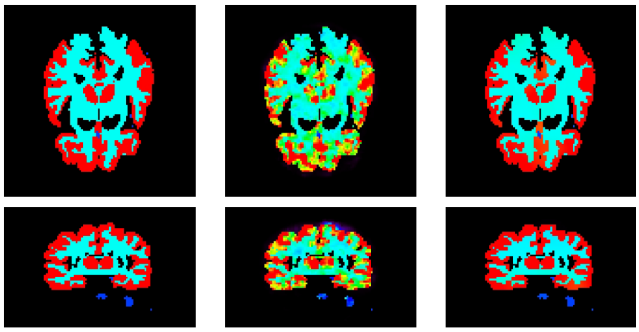

Fig. 11 Binding potentials (BP) using weak and strong spatial regularizations corresponding to 34.57 and 3.61 error levels, respectively.

4.2 High activity phantom

We have also examined a high activity phantom. The number of frames is 5 demonstrating that the algorithm is robust even if the number of frames is very small. As the reconstruction time is proportional to the number of frames, this is an important advantage. The total number of hits is 350 million, the average number of hits per LOR per frame is 0.4. The reconstructed activity in frame 1 and the computed Binding Potentials are shown by Figs. 12 and 13.

4.3 Running times

The running times of the GPU based 3D reconstruction are shown by Table 1. LOR binning, parameter evaluation and forward and back projections are executed in every frame in an iteration. Parameter fitting, on the other hand, needs to be computed only once per iteration. The cost of LOR binning is independent of the voxel resolution, while other steps depend roughly linearly on the number of voxels. The bottleneck of the method is the execution of the forward and back projections.

5 Conclusions

In this paper we investigated the problem of dynamic PET reconstruction when the total activity in a region of interest needs to be reconstructed as a function of time, and presented a set of efficient algorithms suitable for GPU execution.

Acknowledgements

This work has been supported by OTKA K-124124, VKSZ-14 PET/MRI 7T, and EFOP-3.6.2-16-2017-00013.

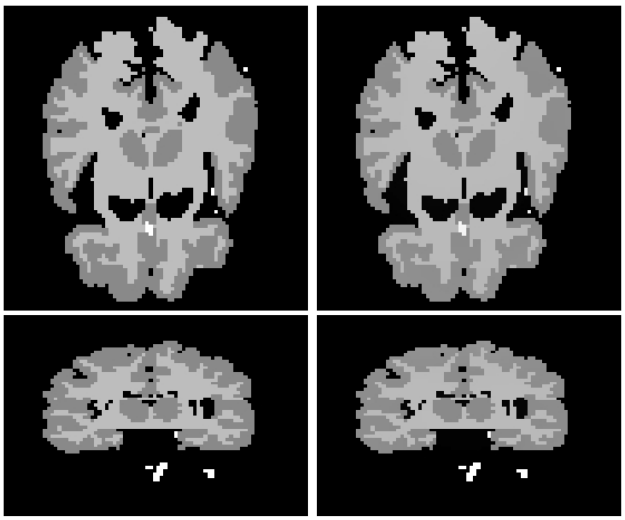

Fig. 12 Reconstructed activity (right) and reference (left) in the first frame.

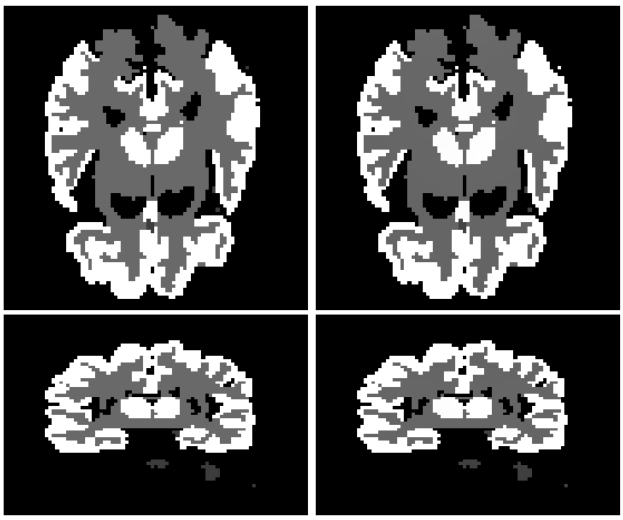

Fig. 13 Reconstructed Binding Potential (right) and its reference (left).

Table 1 Running times in seconds for different voxel grid resolutions

Style name:	64×64×32	128×128×64	256×256×128
LOR binning:	0.8	0.8	0.8
Evaluation	0.001	0.01	0.08
Forward projection	1.2	2.5	6.2
Back projection	7.2	40.3	259.8
Curve fitting	0.12	0.9	1.3

References

[1] Veronese, M., Rizzo, G., Bertoldo, A., Turkheimer, F. E. "Spectral Analysis of Dynamic PET Studies: A Review of 20 Years of Method Developments and Applications", Computational and Mathematical Methods in Medicine, 2016.
https://doi.org/10.1155/2016/7187541

[2] Watabe, H., Ikoma, Y., Kimura, Y., Naganawa, M., Shidahara, M. "PET kinetic analysis - compartmental model", Annals of Nuclear Medicine, 20(9), pp. 583–588, 2006.
https://doi.org/10.1007/BF02984655

[3] Shepp, L. A., Vardi, Y. "Maximum Likelihood Reconstruction for Emission Tomography", IEEE Transactions on Medical Imaging, 1(2), pp. 113–122, 1982.
https://doi.org/10.1109/TMI.1982.4307558

[4] Wang, G., Qi, J. "Direct Estimation of Kinetic Parametric Images for Dynamic PET", Theranostics, 3(10), pp. 802–815, 2013.
https://doi.org/10.7150/thno.5130

[5] Reader, A. J., Verhaeghe, J. "4D image reconstruction for emission tomography", Physics in Medicine and Biology, 59(22), pp. R371–R418, 2014.
https://doi.org/10.1088/0031-9155/59/22/R371

[6] Leeser, M., Coric, S., Miller, E., Yu, H., Trepanier, M. "Parallel-Beam Backprojection: An FPGA Implementation Optimized for Medical Imaging", Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology, 39(3), pp. 295–311, 2005.
https://doi.org/10.1007/s11265-005-4846-5

[7] Shattuck, D. W., Rapela, J., Asma, E., Chatzioannou, A., Qi, J., Leahy, R. M. "Internet2-based 3D PET image reconstruction using a PC cluster", Physics in Medicine and Biology, 47(15), pp. 2785–2795, 2002.
https://doi.org/10.1088/0031-9155/47/15/317

[8] Kachelriess, M., Knaup, M., Bockenbach, O. "Hyperfast parallel-beam and cone-beam backprojection using the cell general purpose hardware", Medical Physics, 34(4), pp. 1474–1486, 2007.
https://doi.org/10.1118/1.2710328

[9] Gac, N., Mancini, S., Desvignes, M., Houzet, D. "High Speed 3D Tomography on CPU, GPU, and FPGA", Eurasip Journal on Embedded Systems, 2008(1), 2008.
https://doi.org/10.1155/2008/930250

[10] Xu, F., Mueller, K. "Real-time 3D computed tomographic reconstruction using commodity graphics hardware", Physics in Medicine and Biology, 52(12), pp. 3405–3419, 2007.
https://doi.org/10.1088/0031-9155/52/12/006

[11] Ha, S., Matej, S., Ispiryan, M., Mueller, K. "GPU-Accelerated Forward and Back-Projections With Spatially Varying Kernels for 3D DIRECT TOF PET Reconstruction", IEEE Transactions on Nuclear Science, 60(1), pp. 166–173, 2013.
https://doi.org/10.1109/TNS.2012.2233754

[12] Wang, Z., Han, G., Li, T., Liang, Z. "Speedup OS-EM image reconstruction by PC graphics card technologies for quantitative SPECT with varying focal-length fan-beam collimation", IEEE Transactions on Nuclear Science, 52(5), pp. 1274–1280, 2005.
https://doi.org/10.1109/TNS.2005.858231

[13] Wang, G., Qi, J. "An optimization transfer algorithm for nonlinear parametric image reconstruction from dynamic PET data", IEEE Transactions on Medical Imaging, 31(10), pp. 1977–1988, 2012.
https://doi.org/10.1109/TMI.2012.2212203

[14] Szirmay-Kalos, L., Kacsó, A. "Regularizing direct parametric reconstruction for dynamic PET with the method of sieves", In: 2016 IEEE Nuclear Science Symposium, Medical Imaging Conference and Room-Temperature Semiconductor Detector Workshop (NSS/MIC/RTSD), Strasbourg, France, 2016, pp. M16D–1.
https://doi.org/10.1109/NSSMIC.2016.8069601

[15] Szirmay-Kalos, L., Magdics, M., Tóth, B. "Volume enhancement with externally controlled anisotropic diffusion", Visual Computer, 33(3), pp. 331–342, 2017.
https://doi.org/10.1007/s00371-015-1203-y

[16] Zhou, J., Qi, J. "Fast and efficient fully 3D PET image reconstruction using sparse system matrix factorization with GPU acceleration", Physics in Medicine and Biology, 56(20), pp. 6739–6757, 2011.
https://doi.org/10.1088/0031-9155/56/20/015

[17] Szirmay-Kalos, L., Magdics, M., Tóth, B., Csébfalvi, B., Umenhoffer, T., Lantos, J., Patay, G. "Fast positron range calculation in heterogeneous media for 3D PET reconstruction", In: 2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC), Anaheim, CA, USA, 2012, pp. 2150–2155.
https://doi.org/10.1109/NSSMIC.2012.6551492

[18] Magdics, M., Szirmay-Kalos, L., Tóth, B., Umenhoffer, T. "Filtered sampling for PET", In: 2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC), Anaheim, CA, USA, 2012, pp. 2509–2514.
https://doi.org/10.1109/NSSMIC.2012.6551572

[19] Szirmay-Kalos, L., Magdics, M., Tóth, B. "Multiple importance sampling for PET", IEEE Transactions on Medical Imaging, 33(4), pp. 970–978, 2014.
https://doi.org/10.1109/TMI.2014.2300932

[20] Lantos, J., Czifrus, S., Legrady, D., Cserkaszky, A. "Detector response function of the NanoPET™/CT system", IEEE Nuclear Science Symposuim & Medical Imaging Conference, Knoxville, TN, USA, 2010, pp. 3641–3643.
https://doi.org/10.1109/NSSMIC.2010.5874491

[21] Assié, K., Breton, V., Buvat, I., Comtat, C., Jan, S., Krieguer, M., Lazaro, D., Morel, C., Rey, M., Santin, G., Simon, L., Staelens, S., Strul, D., Vieira, J.-M., Van De Walle, R. "Monte Carlo simulation in PET and SPECT instrumentation using GATE", Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 527(1-2), pp. 180–189, 2004.
https://doi.org/10.1016/j.nima.2004.03.117

[22] Magdics, M., Szirmay-Kalos, L., Tóth, B., Csendesi, Á., Penzov, A. "Scatter Estimation for PET Reconstruction", In: Dimov, I., Dimova, S., Kolkovska, N. (eds) Numerical Methods and Applications. NMA 2010. Lecture Notes in Computer Science, Vol. 6046, Springer, Berlin, Heidelberg, 2011, pp. 77–86.
https://doi.org/10.1007/978-3-642-18466-6_8

[23] Zubal, I. G., Harrell, C. R., Smith, E. O., Rattner, Z., Gindi, G., Hoffer, P. B. "Computerized three-dimensional segmented human anatomy", Medical physics, 21(2), pp. 299–302, 1994.
https://doi.org/10.1118/1.597290