

Motion Compensation on the GPU for Emission Tomography

Dóra Varnyú, László Szirmay-Kalos and Milán Magdics

Budapest University of Technology and Economics (e-mail: szirmay@iit.bme.hu)

Abstract

The self-motion of the examined patient often cannot be avoided during tomography, which may destroy the examination. Thus, the motion should be measured and this information should be incorporated into the reconstruction process. It would be possible to decompose the measurement time into small intervals, called frames, and consider each frame as a static reconstruction with frame dependent position and orientation of the object. However, higher speed motions, which occur when studying neurological disorders like epilepsy, require very high number of frames, which leads to unacceptable reconstruction times proportional to the number of frames. To attack this problem, we re-formulate the classical Maximum Likelihood Expectation Maximization (ML-EM) scheme to incorporate continuous motion during the frames that are defined according to the tracer dynamics. We conclude that inside frames we should apply 3D motion blur, which needs to be executed before forward projection and during back projection once for the numerator and once for the denominator. For the efficient implementation of the 3D motion blur, the paths of object-space voxels are approximated by polylines and anti-aliasing line drawing methods are adopted to efficiently calculate the blurred result. The 3D motion blur is executed on the GPU exploiting the trilinear interpolators of its texturing units. Having executed the continuous motion blur on the voxel array before the forward projection and during the back projection, all other components of the simulation can be reused from a dynamic reconstruction system developed without motion compensation.

1. Introduction

In dynamic Positron Emission Tomography (PET), we analyze the accumulation and emptying of drugs in organs^{10, 9, 10}. Direct dynamic tomography algorithms usually iterate two steps, first an ML-EM⁶ static forward and back projection are executed for each frame, then the parameters of a time function of pre-defined algebraic form are fit in each of the voxels¹⁴. Thus, the number of projections is proportional to the number of frames, which should be kept under control to avoid prohibiting reconstruction times and memory requirements.

The self-motion of the examined patient often cannot be avoided, e.g. when studying neurological disorders like epilepsy, thus the motion should be measured and taken into account during the reconstruction process⁴. Accurate reconstruction requires projections that simulate all relevant phenomena, including positron range, gamma-photon scattering and absorption both in the measured object and in detector crystals, etc., which are all affected by the motion of the body. Classic forward and back projections should be executed for the transformed data of each frame if we can as-

sume that inside a frame, the subject movement is negligible, i.e. the motion is slow. However, this is usually not the case when neurological disorders are examined, object motion has much quicker dynamics than the tracer diffusion. Gated approaches^{12, 3} keep hits close to discrete time samples to solve the problem of motion inside the frame, but they ignore significant information and thus less reliable for low statistics or low dose experiments.

The other option is to re-bin the list-mode data and find another event list that would be measured if the subject did not move, and execute a conventional reconstruction for the modified sinogram^{5, 1, 2}. This implements a general idea of compensating certain effects by transforming the measured data to show what the measured data would be if this effect had not been happened. Such measured data, also called sinogram filtering techniques are simple to implement and efficient, but cannot be applied for effects happening in the detectors. They either add noise or result in filtered measured data that are not of Poisson distribution, and suffer from the missing or lost data problem. To illustrate this missing data problem, let us follow an event on a pair of detec-

tors when the source is moving. In the simple case of direct hit, the effect of the moving source can be compensated by inverse transforming the line connecting the two detectors and finding those two new detectors that are intersected by the transformed line. It can happen that the transformed line does not intersect the detector modules, so this event must be ignored. However, in theory, the opposite could also occur, when we did not measure an event because its line did not intersect the detector modules, but the inverse transformed line would cause such intersection. As we do not have information about these events, they cannot be taken into account during the reconstruction.

2. Reconstruction algorithm

In this section we propose a generalization of the ML-EM scheme to consider also the object movement. During a PET measurement radiotracer isotopes randomly decay generating events in the detectors of the tomograph. An event is a composition of the identification of the involved detectors, also called *Line Of Response* or *LOR*, and its time of occurrence. To get an efficient numerical method, the space of the measured object is discretized into voxels and the measurement time is discretized by interval boundaries $t_1, t_2, \dots, t_{N_F}$ and events are binned in frames (t_F, t_{F+1}) . The number of voxels and frames are critical because the computational complexity of the reconstruction is proportional to their product.

Using the radiotracer concentration function $K(\mathbf{p}_V, t)$ for voxel V represented by parameter vector $\mathbf{p}_V$, the expected number of decays in time interval $[t_F, t_{F+1}]$ is

$$\tilde{x}_F(\mathbf{p}_V) = \int_{t_F}^{t_{F+1}} K(\mathbf{p}_V, t) e^{-\lambda t} dt \quad (1)$$

where λ is the decay rate of the radiotracer. The algebraic form of $K(\mathbf{p}_V, t)$ is pre-defined and is usually derived from compartment analysis. The goal of reconstruction is thus the identification of the unknown parameters $\mathbf{p}_V$ in voxels.

The correspondence between decays and detected events is established by *system matrix* $\mathbf{A}$ that expresses the probability of generating an event in LOR L given that a decay happened in voxel V . If the measured object is moving, then the system matrix is also time dependent since a point of the object will be at different places. To handle this, we introduce two voxel arrays, an *object-space* array that follows the measured object, and a *tomograph-space* array that is fixed to the tomograph. To make the distinction clear, object-space voxels are denoted by index V or V' , while tomograph-space voxels by W . The system matrix describing the correspondence between tomograph-space voxels and LORs is not time dependent, but tomograph-space voxels have dynamic activity due to both the dynamic radiotracer concentration and the time-dependent geometric transformation between object-space and tomograph-space.

Let us denote the geometric transformation at time t by $\vec{r}_t = T(\vec{r}_o, t)$ where $\vec{r}_t$ is the point in the tomograph's coordinate system, and $\vec{r}_o$ is the point in the object's coordinate system. This transformation is a rigid body transformation, i.e. it contains a translation and rotation, and is measured by an external device. Using this transformation, we can obtain an *overlap factor* $O_{V',W}(t)$ between object-space voxel V' and tomograph-space voxel W that expresses the ratio of the volume in voxel W that actually comes from object-space voxel V' . Formally, the overlap factor is the volume of the intersection of object-space voxel V' and tomograph-space voxel W in time t divided by the volume of a voxel.

The expected number of decays in tomograph-space voxel W during time interval $[t_F, t_{F+1}]$ is

$$\tilde{x}_{W,F} = \sum_{V'} \int_{t_F}^{t_{F+1}} K(\mathbf{p}_{V'}, t) e^{-\lambda t} O_{V',W}(t) dt \quad (2)$$

$$= \sum_{V'} \tilde{x}_F(\mathbf{p}_{V'}) O_{V',W,F}, \quad (3)$$

where $O_{V',W,F}$ is the *average overlapping* of object-space voxel V' and tomograph-space voxel W during frame F :

$$O_{V',W,F} = \frac{\int_{t_F}^{t_{F+1}} K(\mathbf{p}_{V'}, t) e^{-\lambda t} O_{V',W}(t) dt}{\int_{t_F}^{t_{F+1}} K(\mathbf{p}_{V'}, t) e^{-\lambda t} dt}. \quad (4)$$

From now on, we assume that this average overlapping is independent of parameter vector $\mathbf{p}_V$, which is the case when either $K(\mathbf{p}_V, t) e^{-\lambda t}$ or the overlapping factor are constant in time, otherwise, it is a good approximation. In this case, we get

$$O_{V',W,F} \approx \frac{1}{t_{F+1} - t_F} \int_{t_F}^{t_{F+1}} O_{V',W}(t) dt. \quad (5)$$

This simplification makes our numerical algorithm tractable, but it also imposes limitations on the frame lengths. The frames should be short enough to allow the assumption that voxel activities are approximately constant. A better idea is to evaluate the concentration function at the beginning and at the end, and to apply linear interpolation inside the frame.

The *expected number of events* $\tilde{y}_{L,F}$ in LOR L during frame F is the sum of the contributions of all voxels:

$$\tilde{y}_{L,F} = \sum_W \mathbf{A}_{L,W} \tilde{x}_{W,F}, \quad (6)$$

which is the conventional *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, the combined probability considering all LORs and all frames is the product of elementary probabilities. According to the concept of *maximum-likelihood reconstruction*,

unknown parameters are found to maximize the following log-likelihood:

$$\log \mathcal{L} = \sum_L \sum_F (y_{L,F} \log \tilde{y}_{L,F} - \tilde{y}_{L,F}) \quad (7)$$

where $y_{L,F}$ is the measured number of hits in LOR L during frame F .

The likelihood has an extremum where all partial derivatives are zero:

$$\sum_F \sum_W \frac{\partial \tilde{x}_{W,F}}{\partial \mathbf{p}_{V,P}} \sum_L \left(\mathbf{A}_{L,W} \frac{y_{L,F}}{\tilde{y}_{L,F}} - \mathbf{A}_{L,W} \right) = 0 \quad (8)$$

for $V = 1, 2, \dots, N_V$ and $P = 1, \dots, N_P$.

Using Eq. 3 and also exploiting our simplifying assumption that the average overlapping is independent of the parameter vector, we get

$$\frac{\partial \tilde{x}_{W,F}}{\partial \mathbf{p}_{V,P}} = \sum_{V'} \frac{\partial \tilde{x}_F(\mathbf{p}_{V'})}{\partial \mathbf{p}_{V,P}} o_{V',W,F} = \frac{\partial \tilde{x}_{V,F}}{\partial \mathbf{p}_{V,P}} o_{V,W,F}. \quad (9)$$

since derivative $\partial \tilde{x}_F(\mathbf{p}_{V'}) / \partial \mathbf{p}_{V,P}$ is non-zero only if $V = V'$. Thus, Eq. 8 simplifies to

$$\sum_F \frac{\partial \tilde{x}_F(\mathbf{p}_V)}{\partial \mathbf{p}_{V,P}} \sum_L \sum_W o_{V,W,F} \left(\mathbf{A}_{L,W} \frac{y_{L,F}}{\tilde{y}_{L,F}} - \mathbf{A}_{L,W} \right) = 0. \quad (10)$$

Let us denote the *back projection* of the forward projection by $x_{V,F}$, which is defined as:

$$x_{V,F} = \tilde{x}_F(\mathbf{p}_V) \cdot \frac{\sum_W o_{V,W,F} \sum_L \mathbf{A}_{L,W} \frac{y_{L,F}}{\tilde{y}_{L,F}}}{\sum_W o_{V,W,F} \sum_L \mathbf{A}_{L,W}}. \quad (11)$$

From this equation, we can express

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

In this formula, we can recognise the *sensitivity* $S_{V,F}$ of voxel V during frame F :

$$S_{V,F} = \sum_W o_{V,W,F} \sum_L \mathbf{A}_{L,W}$$

The sensitivity is the probability that a decay of the voxel is detected at all by the tomograph.

Substituting these into Eq. 8, we obtain:

$$\sum_F S_{V,F} \frac{\partial \tilde{x}_F(\mathbf{p}_V)}{\partial \mathbf{p}_{V,P}} \left(\frac{x_{V,F}}{\tilde{x}_F(\mathbf{p}_V)} - 1 \right) = 0. \quad (12)$$

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 voxels¹⁴. Taking $x_{V,F}$ from the previous iteration, which is similar to the one-step-late strategy, the non-linear equation is solved independently for each voxel.

2.1. Reconstruction algorithm

The reconstruction algorithm iteratively solves the established equations. An iteration of the motion compensated reconstruction consists of the following steps (Figure 1):

1. For each frame we compute $x_{V,F}$, which in turn, includes the following steps:
 - a. Evaluation of the integral of the radiotracer for each object-space voxel V for this frame to get $\tilde{x}_{V,F}$ (Eq. 1).
 - b. Motion blur computation to compute the activity of tomograph-space voxels W to get $\tilde{x}_{W,F}$ (Eq. 3).
 - c. Forward projection of the tomograph-space voxels to obtain expected LOR hits $\tilde{y}_{L,F}$ (Eq. 6).
 - d. Computation of numerator $n_{W,F} = \sum_L \mathbf{A}_{L,W} y_{L,F} / \tilde{y}_{L,F}$ and denominator $S_{W,F} = \sum_L \mathbf{A}_{L,W}$.
 - e. Motion blur of the tomograph-space numerator and denominator voxel arrays into object-space voxel arrays $n_{V,F}$ and $S_{V,F}$.
 - f. Executing the division to compute $x_{V,F} = n_{V,F} / S_{V,F}$ (Eq. 11).
2. Fitting the parameters on $x_{V,F}$ for each object-space voxel V by the solution of Eq. 12.

3. Efficient motion blur computation

Object motion at any moment can be described with an affine transformation consisting of a translation vector and a quaternion for rotation. This transformation applies for all voxels in the measured volume, i.e. the voxels are moving together. In motion blur computation, the transformation at the beginning and end of the time frame are given. Within the frame the motion is assumed to be linear, i.e. the translation vector can be interpolated linearly and the quaternion with *Spherical Linear Interpolation* (SLERP). An accurate blurring algorithm would take multiple samples from each voxel and calculate their position with the interpolated motion transformation in very small time steps. However, this approach is extremely computationally expensive and requires a long execution time, which cannot be afforded during a PET examination. A reduction in computation expenses can be achieved if each voxel's position is calculated only for the start and end of the frame, and between these two points the voxel is assumed to be moving along a linear path. This way, the voxel's path can be computed with fast and effective line drawing algorithms, at the expense of a reduced blurring accuracy, especially if the measured object has a rapid rotating motion. In our research we examined the ray marching and the Antialiased Bresenham line drawing algorithms.

Thus, we assume that in a frame the movement of a point of the object is piece-wise linear, hence mapping between object-space and tomograph-space voxels can be described with a geometric transformation of the start and end points of a frame. With $\tilde{x}_{V,F}$ already computed for object-space voxel

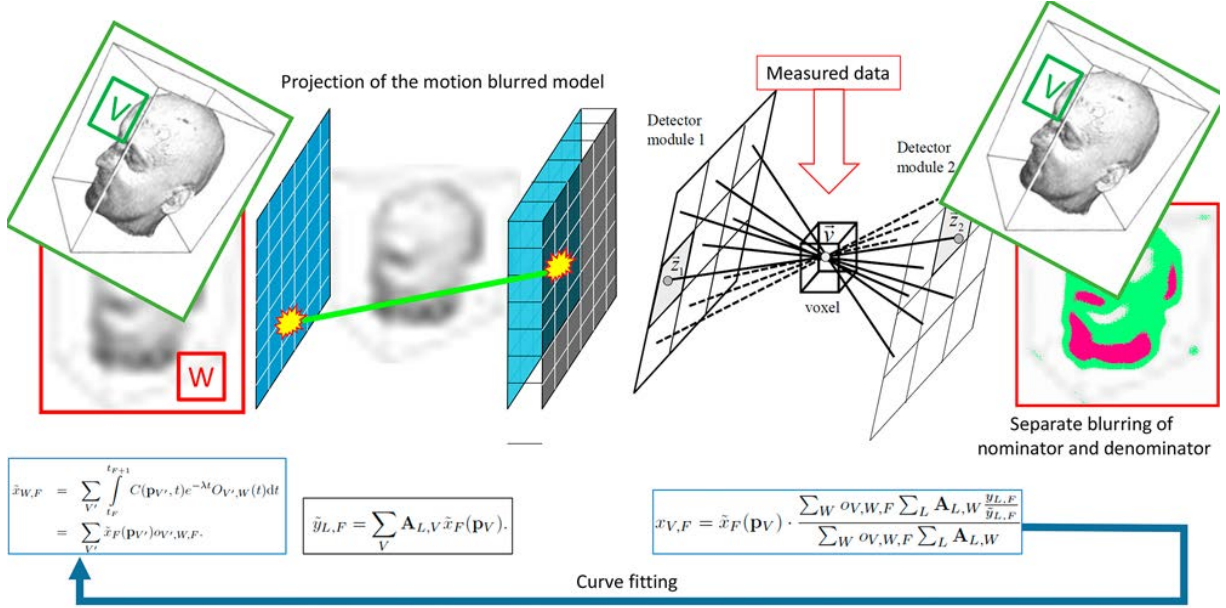


Figure 1: System overview

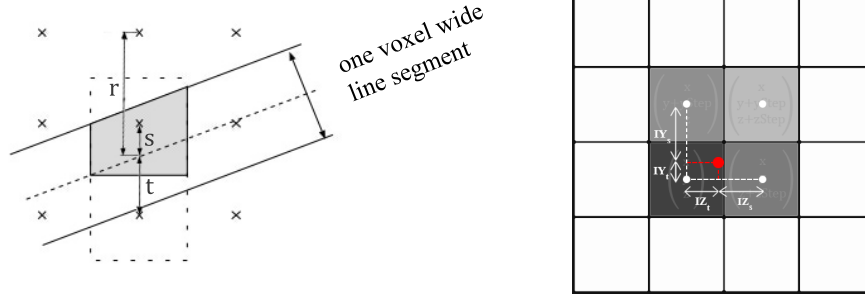


Figure 2: Left: Box filtering of a line segment. Right: Linear interpolation in the Antialiased Bresenham algorithm reduces the number of memory accesses from 4 to 1 per iteration.

V in frame F , the next step is to determine which tomograph-space voxels W overlap with voxel V during this frame. At the beginning and end of the frame, we calculate the position of voxel V in object-space which is mapped to tomograph-space voxel W . Let V_{start} denote the object-space voxel that contains the transformed center of voxel W at the beginning of the frame and V_{end} the voxel that contains the transformed center of W at the end of the frame. As motion is assumed to be linear, each object-space voxel that is overlapped by W will be located on the line segment connecting V_{start} and V_{end} .

In back projection motion blur, the numerator $n_{W,F}$ and denominator $S_{W,F}$ of tomograph-space voxels W are known, and we want to compute $n_{V,F}$ and $S_{V,F}$ for object-space

voxel V . This is achieved in the same manner as during forward projection, by determining tomograph-space starting and end points W_{start} and W_{end} of the motion of voxel V , and visiting tomograph-space voxels along the line segment with the Antialiased Bresenham algorithm. Numerator and denominator $n_{V,F}$ and $S_{V,F}$ for object-space voxel V are increased in each visited tomograph-space voxel W taking into account the overlapping between the voxels.

Note that in forward projection, the tomograph-space is considered to stand still and corresponding voxel traversal is executed in object-space. In back projection, on the other hand, object-space is fixed. Since the motion is relative, both approaches make sense. However, the proposed solution is more efficient for the GPU implementation since the voxel

that accumulates the contribution is fixed and can be assigned to a computational thread. This way, the computation is of gathering type and avoids the need for atomic operations.

3.1. Antialiased Bresenham algorithm

The crucial part of motion blur computation is the determination of those voxels and their weights that are overlapped for some time with the moving voxels of the other array. Each moving voxel follows a linear path, thus this algorithm is in fact a 3D line rasterization method ¹¹.

The Bresenham algorithm iterates in the coordinate axis direction in which the line changes the most rapidly in unit steps and selects a single voxel in each slice orthogonal to the given coordinate axis. The selected voxel in the slice is the one that is closest to the geometric line. When the algorithm steps onto the next slice, it chooses whether the two coordinates identifying voxels in the slice should be incremented or not based on which the closest voxel is to the line. This is decided with the help of an error variable, which keeps track of the distance between the geometric line and the selected voxel center. A drawback of the Bresenham algorithm is that it obtains just a single voxel in each slice. However, even if the moving voxel were a single point, it could cross multiple voxel in each slice. Moreover, when the moving voxel has the same size as the voxel grid standing still, ignoring partial overlapping and selecting just a single voxel in each slice would result in even larger errors.

Based on previous research ¹³, potentially overlapping voxels along the line segment are determined by the *Antialiased Bresenham line drawing algorithm*. This algorithm also iterates through the direction in which the line changes the most rapidly in unit steps. In every iteration the closest and its three neighbor voxels are visited and the overlap factors $o_{V,W,F}$ are calculated as the weights of the Antialiased Bresenham algorithm. Coordinates and weights are calculated incrementally, and the total sum of weights along the line is 1.

The Antialiased Bresenham complements the Bresenham algorithm with on-the-fly box filtering. For this purpose, the intersection of the one-voxel wide line segment and the voxel concerned has to be calculated. Considering first only the 2D case, we can conclude that a maximum of three voxels may intersect the line segment in each column if the slant is between 0 and 45 degrees (Figure 2). Let the vertical distance of the three closest voxels to the center of the line be r , s and t respectively, and suppose $s < t \leq r$.

The areas of intersection, I_s , I_t and I_r depend not only on r , s and t , but also on the slant of the line segment. This dependence, however, can be rendered unimportant by using the following approximation:

$$I_s \approx (1-s), I_t \approx (1-t) = s, I_r \approx 0.$$

These formulae can be evaluated incrementally together with the Bresenham's incremental coordinate calculation. In every column, the contributions of the two sampled voxels are weighted with the area of their intersection with the line segment, i.e. I_s and I_t .

The 3D generalization of the Antialiased Bresenham algorithm samples 4 voxels in every iteration, 2-2 both horizontally and vertically. Each voxel gets two independent weights (one horizontal and one vertical weight), which are computed by two simultaneous 2D Antialiased Bresenham algorithms. These weights are then multiplied to form the final weight of the voxel's contribution.

On GPU, the built-in trilinear interpolation of the texture memory can be utilized in order to reduce the number of memory accesses from 4 to 1 per iteration. Figure 2 shows how the horizontal and vertical weights are used in the interpolation of the four voxels.

4. Results

To examine the proposed methods, first we use a 2D phantom ⁸ where the number of LORs is $N_L = 2115$ and the number of voxels is $N_V = 1024$ (Fig. 3) and reconstruct the activity curves of a 2D brain model.

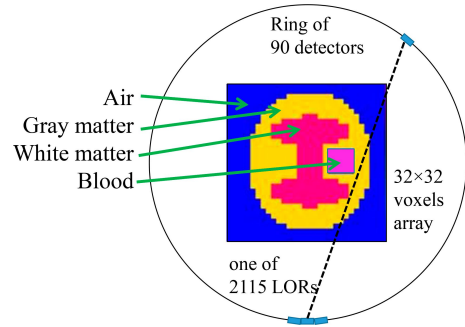


Figure 3: 2D tomograph model: The detector ring contains 90 detector crystals and each of them is of size 2.2 in voxel units and participates in 47 LORs connecting this crystal to crystals being in the opposite half circle, thus the total number of LORs is $90 \times 47/2 = 2115$. The voxel array to be reconstructed is in the middle of the ring and has 32×32 resolution, i.e. 1024 voxels.

4.1. Blood only model

In our first measurement, we examined the reconstruction of a small fraction of blood moving along a circular path with constant angular velocity. Measured events were binned into 5 frames, during which the object completed one full 360 degree turn around the center of the measured volume. Fig. 4 shows the blurred snapshots of its motion in each of the five frames, and the reconstructed time-activity function.

The true solution is a small active square on the inactive background as shown by the right top picture of Figure 4. Below it, we also show the result of the motion compensated reconstruction, which lacks the blurring that is clearly visible in the left images showing the inputs of the forward projection in the five frames.

4.2. Brain phantom

In our second measurement, we used a brain phantom consisting of four regions: gray matter, white matter, blood, and air. Measured events were grouped into 10 frames, during which the phantom completed one full 360 degree rotation around its center point (Figure 6). Reconstruction was first performed without motion compensation to demonstrate the high degree of blurring in the reconstructed image caused by the movement of the object (“No compensation” column in Figure 5). Applying motion compensation eliminated this blurring and significantly reduced reconstruction error, as shown in the “Compensation” column of Figure 5.

Dividing measured events into 25 frames instead of the previous 10 allows finer motion compensation, as in smaller time spans the motion of a voxel can be estimated with a geometric transformation more precisely. Fig. 7 shows that with more frames, motion compensation could reduce reconstruction error further.

4.3. 3D implementation with CUDA

Our implementation used the CUDA platform to accelerate parallel computations with the GPU. For the measured object, we used the NEMA IQ numerical phantom of size $166 \times 166 \times 126$ voxels (width, height, depth) depicted in Figure 8. Algorithms were compared both in terms of execution time and accuracy. A total of six measurements were executed with the NEMA IQ phantom with varying object motions. References were computed with high precision ray marching, taking 100 samples from each voxel and interpolating the motion transformation in 1024 time steps. Table 1 summarizes the results. Rotation was in all cases around the z axis. Blur error was calculated as the sum of the squared differences in each voxel divided by the total reference activity (relative L_2 error).

Results show that when the motion is small, the Antialiased Bresenham algorithm offers greater accuracy at a similar speed as the ray marching. However, as the motion becomes greater (i.e. if the measured object moves fast compared to the number of time frames), the Antialiased Bresenham gets behind both in speed and in accuracy. Another notable result is that if the path of an active voxel leads outside the measured volume, the Antialiased Bresenham introduces significant error.

5. Conclusions

This paper proposed a model based motion compensation algorithm for emission tomography. Based on the generalization of the ML-EM algorithm seeking for the maximum likelihood estimate, we concluded that the motion can be captured by three motion blur operations in every iteration cycle. First, the current activity estimate should be motion blurred before executing forward projection. Then the numerator and the denominator of the back-projection formula should be motion blurred before computing their ratios. The demonstration shows that artifacts can be eliminated by the proposed method.

Acknowledgements

This work has been supported by OTKA K-124124, EFOP 4.2.1-16-2017-00021, and EFOP-3.6.2-16-2017-00013 (EU) New National Excellence Program of the Ministry of Human Capacities.

References

1. P. Buhler, U. Just, E. Will, J. Kotzerke, and J. van den Hoff. An accurate method for correction of head movement in pet. *IEEE Transactions on Medical Imaging*, 23(9):1176–1185, Sept 2004.
2. C. Chan, X. Jin, E. K. Fung, M. Naganawa, T. Mulnix, R. E. Carson, and C. Liu. Event-by-event respiratory motion correction for pet with 3-dimensional internal-external motion correlation. In *2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC)*, pages 2117–2122, Oct 2012.
3. J. Cui, J. Yang, E. Graves, and C. S. Levin. Gpu-enabled pet motion compensation using sparse and low-rank decomposition. In *2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC)*, pages 3367–3370, Oct 2012.
4. J. Jiao, A. Bousse, K. Thielemans, N. Burgos, P. S. J. Weston, J. M. Schott, D. Atkinson, S. R. Arridge, B. F. Hutton, P. Markiewicz, and S. Ourselin. Direct parametric reconstruction with joint motion estimation/correction for dynamic brain pet data. *IEEE Transactions on Medical Imaging*, 36(1):203–213, Jan 2017.
5. M. Menke, M. S. Atkins, and K. R. Buckley. Compensation methods for head motion detected during pet imaging. *IEEE Transactions on Nuclear Science*, 43(1):310–317, Feb 1996.
6. L. Shepp and Y. Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE Trans. Med. Imaging*, 1:113–122, 1982.
7. L. Szirmay-Kalos, M. Magdics, and B. Tóth. Multiple importance sampling for PET. *IEEE Trans Med Imaging*, 33(4):970–978, 2014.
8. L. Szirmay-Kalos, M. Magdics, B. Tóth, and T. Bükki. Averaging and Metropolis iterations for positron emission tomography. *IEEE Trans Med Imaging*, 32(3):589–600, 2013.

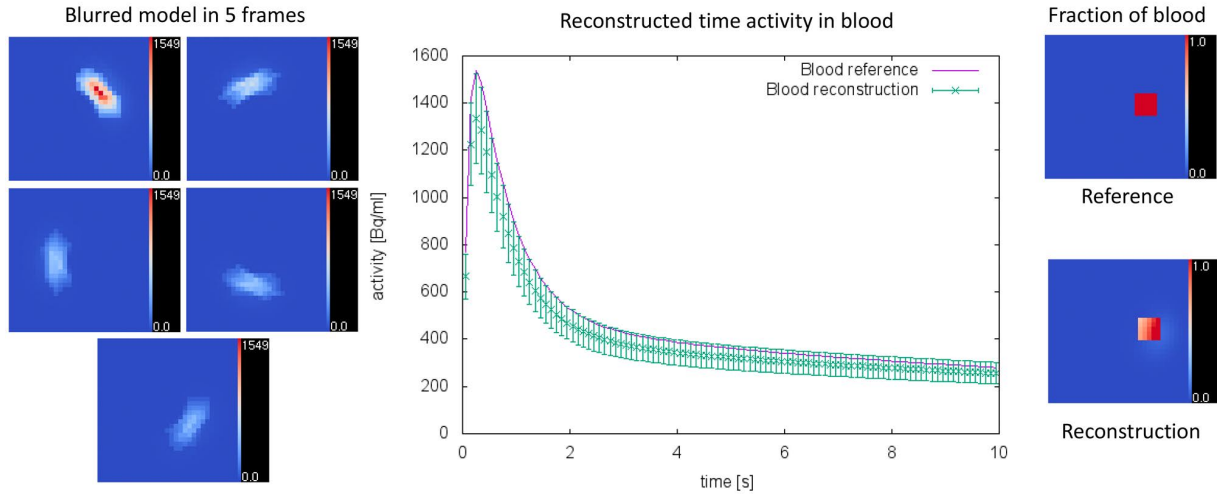


Figure 4: Reconstruction of the blood with 5 frames. Note that the blood object is sharply reconstructed despite to the strong blurring inside frames.

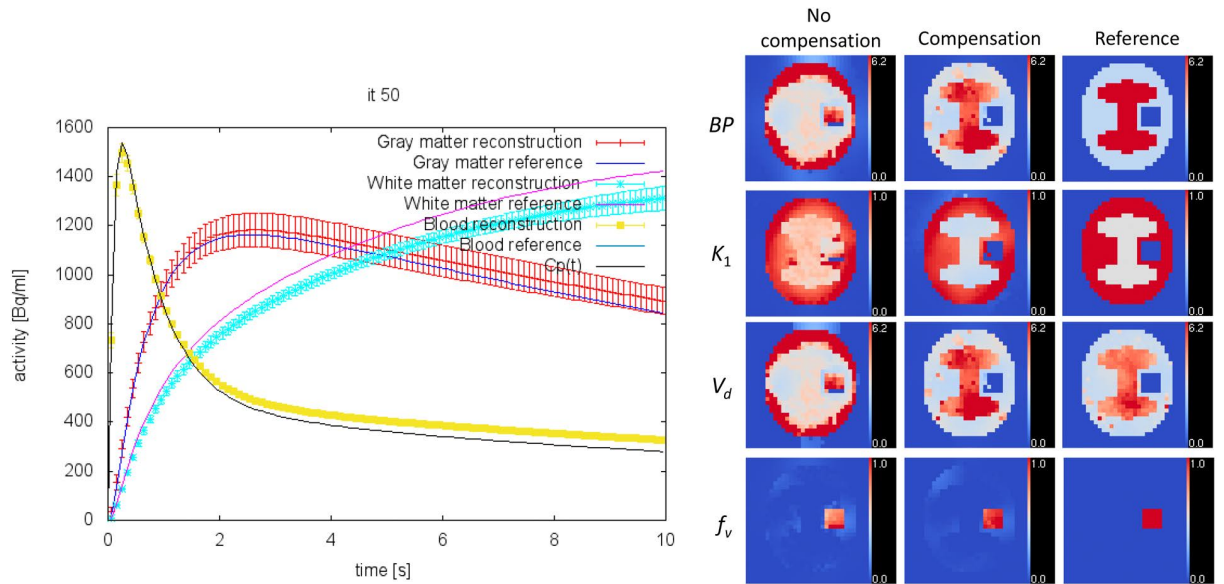


Figure 5: Reconstruction of the brain with 10 frames

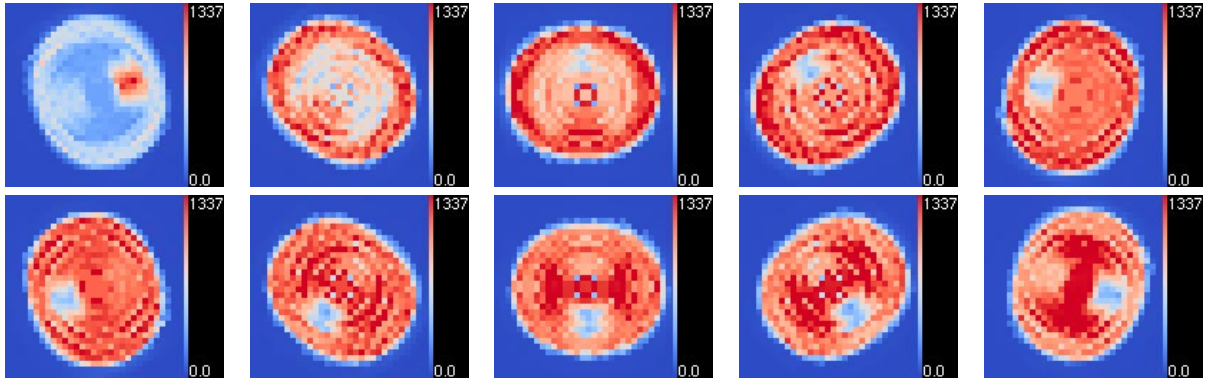


Figure 6: Blurred images before forward projection in 10 frames

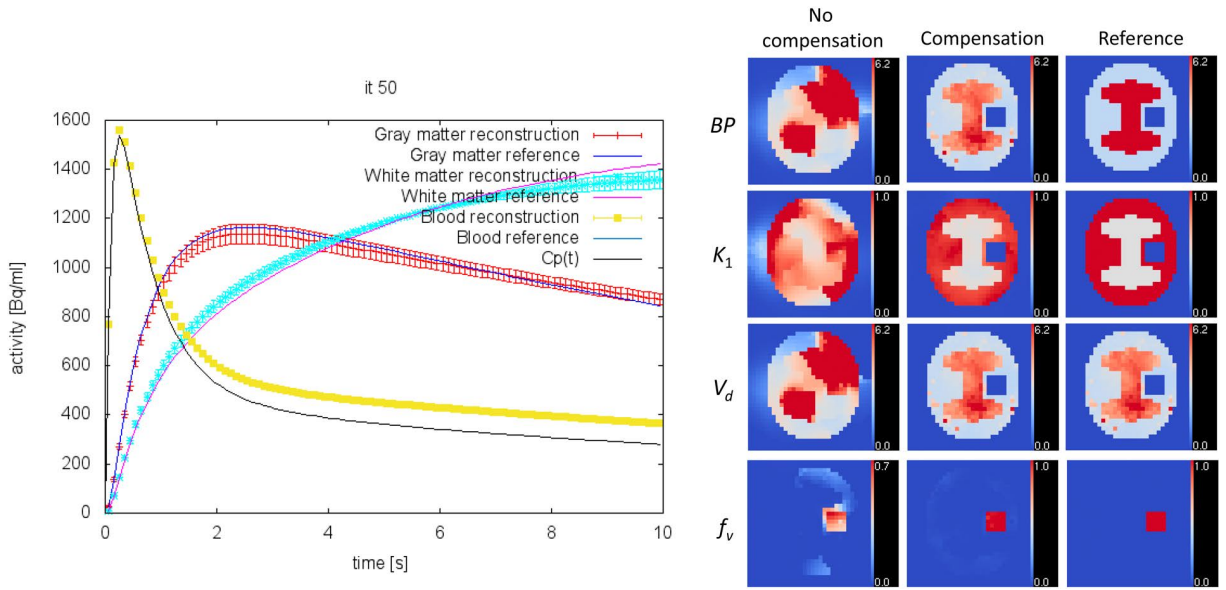


Figure 7: Reconstruction of the brain with 25 frames

9. László Szirmay-Kalos and Ágota Kacsó. Regularizing direct parametric reconstruction for dynamic pet with the method of sieves. In *Molecular Imaging Conference, MIC '16*, pages M16D–1, 2016.
10. László Szirmay-Kalos, Ágota Kacsó, Milán Magdics, and Balázs Tóth. Dynamic pet reconstruction on the gpu. *Periodica Polytechnica Electrical Engineering and Computer Science*, 62(4):134–143, 2018.
11. L. Szirmay-Kalos (editor). *Theory of Three Dimensional Computer Graphics*. Akadémia Kiadó, Budapest, 1995. <http://www.iit.bme.hu/~szirmay>.
12. M. Toussaint, J. P. Dussault, and R. Lecomte. Revisiting motion compensation models in pet image reconstruction. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pages 90–94, April 2016.
13. Dóra Varnyú and László Szirmay-Kalos. Thick line integration with filtered sampling. In *KEPAF 2019, Debrecen*, 2019.
14. Guobao Wang and Jinyi Qi. An optimization transfer algorithm for nonlinear parametric image reconstruction from dynamic pet data. *IEEE Trans Med Imaging*, 31(10):1977–1988, 2012.

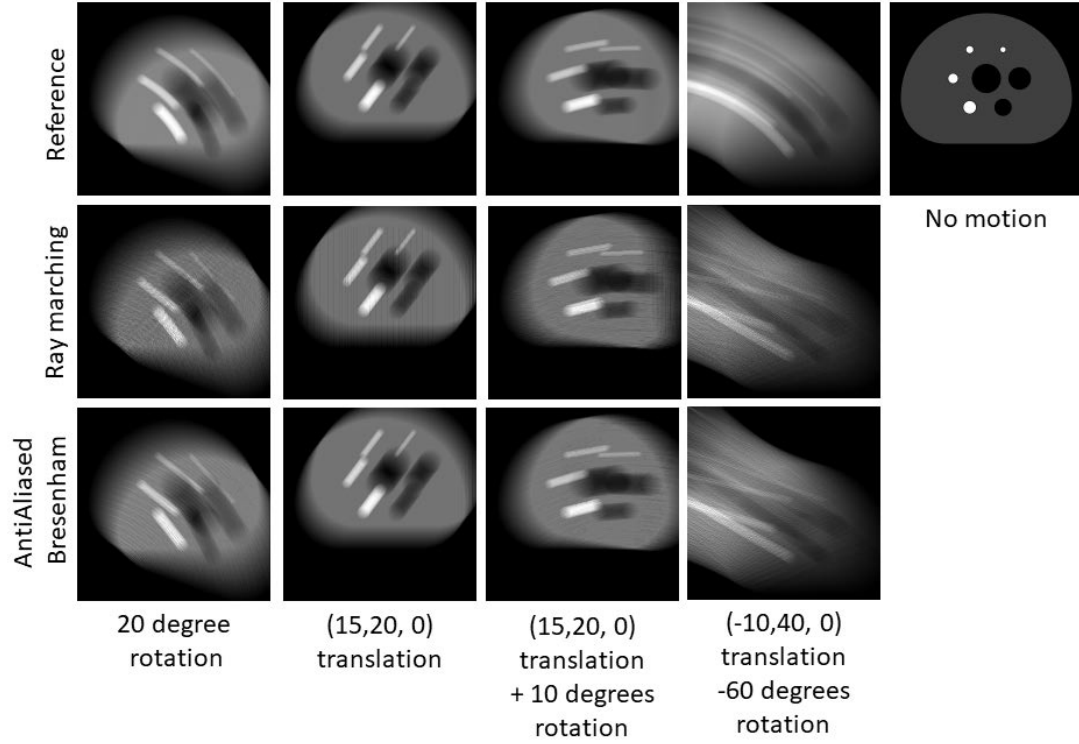


Figure 8: Original and blurred NEMA IQ phantom. The proposed algorithm is compared to the reference obtained with a high number of point samples and also to ray marching.

Object motion		Error [%]		Execution time [ms]	
translation	rotation	Ray marching	Antialiased Bresenham	Ray marching	Antialiased Bresenham
(15, 20, 0)	0	9.5059	13.1769	0.0965	0.0239
(15, 20, 0)	10	12.9566	23.8758	1.6093	0.0921
(15, 20, 30)	0	11.4946	16.3787	0.5052	2.5906
(0, 0, 0)	20	16.9948	59.9531	0.8541	0.7081
(-10, 40, 0)	-60	51.4535	89.5817	13.4989	16.1665
(1.5, 5.0, 2.5)	-5	7.7408	10.1801	0.2789	2.3433

Table 1: Execution times and blur errors of the ray marching and Antialiased Bresenham algorithms. Translation is given in voxel edge units, rotation in degrees.