GPU Accelerated Image Reconstruction in a Two-Strip J-PET Tomograph

P. Białas[a,*], J. Kowal[a], A. Strzelecki[a], T. Bednarski[a], E. Czerwiński[a], A. Gajos[a],
D. Kamińska[a], Ł. Kapłon[a,b], A. Kochanowski[c], G. Korcyl[a], P. Kowalski[d], T. Kozik[a],
W. Krzemień[a], E. Kubicz[a], P. Moskal[a], Sz. Niedźwiecki[a], M. Pałka[a], L. Raczyński[d],
Z. Rudy[a], O. Rundel[a], P. Salabura[a], N.G. Sharma[a], M. Silarski[a], A. Słomski[a],
J. Smyrski[a], A. Wieczorek[a,b], W. Wiślicki[d], M. Zieliński[a] AND N. Zoń[a]

[a]Faculty of Physics, Astronomy and Computer Science, Jagiellonian University,
St. Łojasiewicza 11, 30-348 Kraków, Poland

[b]Institute of Metallurgy and Material Science, Polish Academy of Sciences, W.S. Reymonta 25, 30-059 Kraków, Poland

[c]Faculty of Chemistry, Jagiellonian University, R. Ingardena 3, 30-060 Kraków, Poland

[d]Świerk Computing Centre, National Centre for Nuclear Research, 05-400 Otwock-Świerk, Poland

We present a fast GPU implementation of the image reconstruction routine, for a novel two strip PET detector that relies solely on the time of flight measurements.

1. Introduction

In this paper we present a graphics processing unit (GPU) implementation of list-mode reconstruction algorithm of a 2D strip PET. This detector consists of two parallel bars (strips) of scintillator with a photomultiplier attached to each end [1, 2].
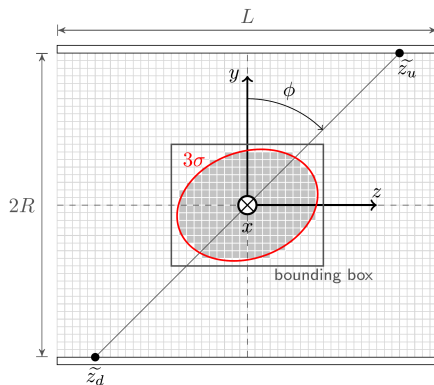

Fig. 1.　2D-strip detector geometry.

By measuring the time of the arrivals of photons to each of the photomultipliers we can reconstruct the position at which γ quanta have interacted with the scintillators as well as the position along the line-of-response (LOR) (see Fig. 1). Application of the state of the art electronics developed at the Jagiellonian University allowed to achieve the required resolution [3, 4].

A double-strip prototype can be regarded as an elementary part of the full 3D "J-PET" [5] detector under construction at our faculty [1, 2, 6–8]. The detector will consist of cylindrically arranged scintillator strips (as shown schematically in Fig. 2) enabling a full 3D reconstruction. However, the two strip prototype is also of interest as a cheap scanning device.
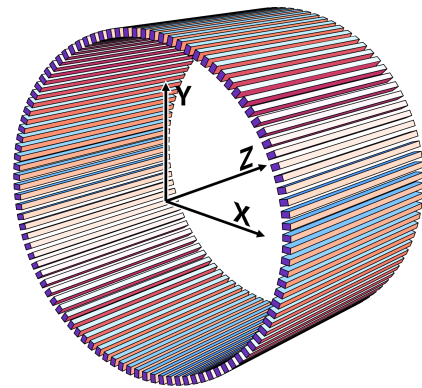

Fig. 2.　An example of the possible 3D detector geometry of the J-PET detector.

2. Setup

The description of the readout system electronics is beyond the scope of this paper, we will just assume that for each event we are given three numbers $(\widetilde{z_u}, \widetilde{z_d}, \Delta \widetilde{l})$ (see Fig. 1). By convention we use the tilde to denote measured quantities as opposite to "real" or exact values. The $\widetilde{z_u}$ and $\widetilde{z_d}$ denote respectively the reconstructed position along the upper and lower strip and $\Delta \widetilde{l}$ is the difference of the distances along the LOR from the emission point to the upper and lower strips

$$\Delta \widetilde{l} = \sqrt{(R - \widetilde{e}_y)^2 + (\widetilde{z}_u - \widetilde{e}_z)^2}$$
$$- \sqrt{(R + \widetilde{e}_y)^2 + (\widetilde{z}_d - \widetilde{e}_z)^2}. \quad (1)$$

From those measurements the emission position and angle can be reconstructed directly

$$\tan \widetilde{\theta} = \frac{\widetilde{z}_u - \widetilde{z}_d}{2\bar{R}},$$

$$\widetilde{y} = -\frac{1}{2} \frac{\Delta \widetilde{l}}{\sqrt{1 + \tan^2 \widetilde{\theta}}} = \frac{2\bar{R}\Delta \widetilde{l}}{\sqrt{\widetilde{z}_u - \widetilde{z}_d + 4\bar{R}^2}},$$

$$\widetilde{z} = \frac{1}{2}\left(\widetilde{z}_u + \widetilde{z}_d + 2y \tan \widetilde{\theta}\right) =$$

$$\frac{1}{2}\left(\widetilde{z}_u + \widetilde{z}_d + \frac{(\widetilde{z}_u - \widetilde{z}_d)\Delta \widetilde{l}}{\sqrt{\widetilde{z}_u - \widetilde{z}_d + 4\bar{R}^2}}\right). \quad (2)$$

It is however subject to measurement errors (see the correlation matrix description at the end of Sect. 3). In Fig. 3b we present results of such direct reconstruction of the phantom depicted in Fig. 3a. It is clear that the resolution of the detector is not sufficient for direct reconstruction and statistical reconstruction methods need to be applied.

The statistical reconstruction is done iteratively using the List-Mode version of the maximal likelihood expectation maximization (MLEM) algorithm. Each iteration of this algorithm is defined by the following formula [9]:

$$\rho(l)^{(t+1)} = \sum_{j=1}^{N} \frac{P(\widetilde{e}_j|l)\rho(l)^{(t)}}{\sum\limits_{i=1}^{M} P(\widetilde{e}_j|i)s(i)\rho(i)^{(t)}}. \quad (3)$$

The $\rho(l)$ is the sought tracer emission density given as the average number of emissions from pixel l during the examination. The $P(\widetilde{e}|i)$ is a *reconstruction kernel* that represents the probability that an event originating in pixel i will be detected as $\widetilde{e}$. The $s(i)$ is the *sensitivity* of the pixel i, i.e. the probability that an event emitted from pixel i will be detected at all. This sensitivity can be easily calculated from the geometry

$$s(y, z) = \pi^{-1}\left(\arctan \min\left(\frac{\frac{1}{2}L - z}{R - y}, \frac{\frac{1}{2}L + z}{R + y}\right)\right.$$

$$\left. - \arctan \max\left(-\frac{\frac{1}{2}L + z}{R - y}, \frac{-\frac{1}{2}L + z}{R + y}\right)\right). \quad (4)$$

In derivation we have assumed that the detection probability along the strip is constant and that it does not depend on the angle of incidence. These conditions should be approximately fulfilled for incidence angles not exceeding 30°.

The formula (3) can be rewritten as

$$\rho'(l)^{(t+1)} = \sum_{j=1}^{N} \frac{P(\widetilde{e}_j|l)\rho'(l)^{(t)}}{\sum\limits_{i=1}^{M} P(\widetilde{e}_j|i)\rho'(i)^{(t)}}. \quad (5)$$

with

(a) Ideal reconstruction   **(b)** Direct reconstruction

(c) after 1 iteration   **(d)** after 5 iterations

(e) after 10 iterations   **(f)** after 25 iterations
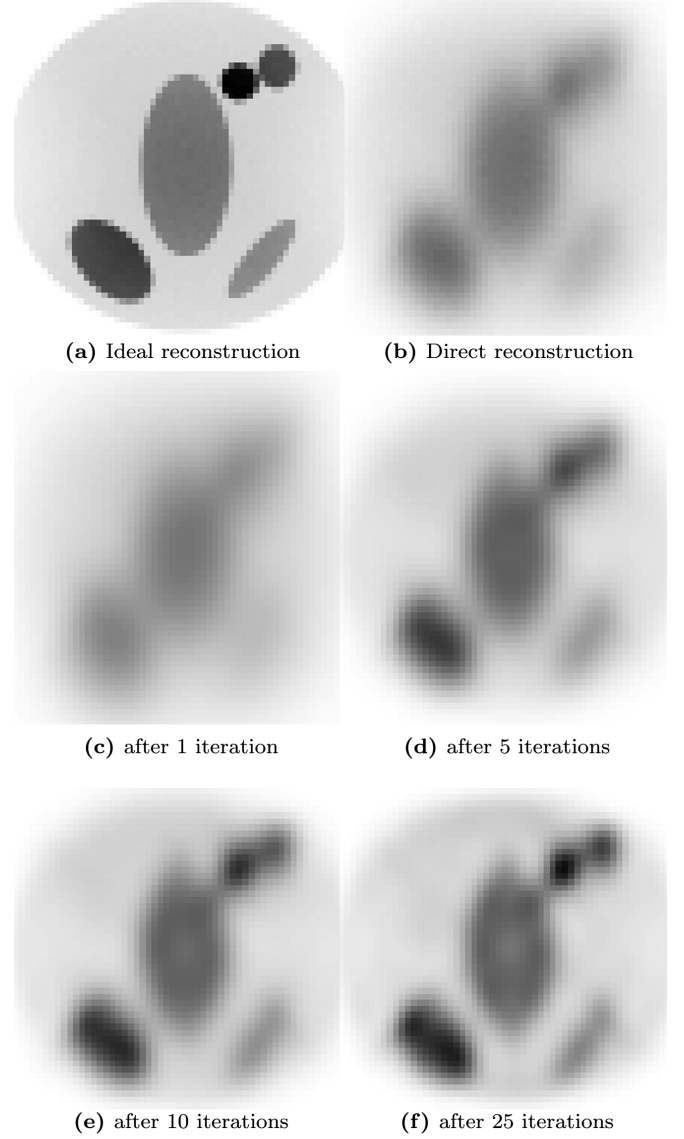
Fig. 3. Phantom used in reconstruction ($R = 130$mm, $L = 300$mm and 4×4mm^2 pixel size).

$$\rho'(i) \equiv s(i)\rho(i). \quad (6)$$

In the following we will give the results of the reconstruction of $\rho'(i)$.

The sum over j in Eq. (5) runs over all collected events $\{\widetilde{e}_j\}$. Considering that up to hundred millions of events can be collected during a single scan this is a very time consuming calculation so the efficient calculation of the kernel P is essential.

3. Kernel and correlation matrix

In Refs. [10, 11] we have found analytical approximation of $P(\widetilde{e}|i)$ given by

$$P(\widetilde{e}|i) \approx \frac{\det^{\frac{1}{2}} C}{2\pi\sqrt{aC^{-1}a + 2oC^{-1}b}}$$

$$\times \exp\left(-\frac{1}{2}\left(\boldsymbol{b}C^{-1}\boldsymbol{b} - \frac{\left(\boldsymbol{b}C^{-1}\boldsymbol{a}\right)^2}{\boldsymbol{a}C^{-1}\boldsymbol{a} + 2\boldsymbol{o}C^{-1}\boldsymbol{b}}\right)\right). \quad (7)$$

The $\boldsymbol{o}$, $\boldsymbol{a}$, $\boldsymbol{b}$ are defined as follows:

$$\boldsymbol{o} = \begin{pmatrix} -(\Delta y + \widetilde{y} - R)\tan\widetilde{\theta}\cos^{-2}\widetilde{\theta} \\ -(\Delta y + \widetilde{y} + R)\tan\widetilde{\theta}\cos^{-2}\widetilde{\theta} \\ -(\Delta y + \widetilde{y})\cos^{-1}\widetilde{\theta}(1 + 2\tan^2\widetilde{\theta}) \end{pmatrix}, \quad (8)$$

$$\boldsymbol{a} = \begin{pmatrix} -(\Delta y + \widetilde{y} - R)\cos^{-2}\widetilde{\theta} \\ -(\Delta y + \widetilde{y} + R)\cos^{-2}\widetilde{\theta} \\ -(\Delta y + \widetilde{y})\cos^{-1}\widetilde{\theta}\tan\widetilde{\theta} \end{pmatrix}, \quad (9)$$

$$\boldsymbol{b} = \begin{pmatrix} \Delta z - \Delta y\tan\widetilde{\theta} \\ \Delta z - \Delta y\tan\widetilde{\theta} \\ -2\Delta y\cos^{-1}\widetilde{\theta} \end{pmatrix}, \quad (10)$$

and

$$\Delta y = y - \widetilde{y} \quad and \quad \Delta z = z - \widetilde{z}. \quad (11)$$

The $\widetilde{y}$ and $\widetilde{z}$ are the coordinates of the reconstructed emission point and $\widetilde{\theta}$ is the reconstructed emission angle of the event $\widetilde{\boldsymbol{e}}$. The y and z are the coordinates of the center of pixel i. C is the correlation matrix which in general can be of the form

$$C^{-1} = \begin{pmatrix} \frac{1}{\sigma_z^2} & 0 & \gamma \\ 0 & \frac{1}{\sigma_z^2} & -\gamma \\ \gamma & -\gamma & \frac{1}{\sigma_{\Delta l}^2} \end{pmatrix}. \quad (12)$$

This matrix depends on the $\widetilde{z}_u$ and $\widetilde{z}_d$, for σ_z. Experimentally we have found this dependence to be quite weak on the order of 10% from the center (lowest) to the edge (highest). We have found out that coefficient γ can be neglected as long as we do not take into account events with $z_{u(d)}$ near the edge of the scintillators. This may change when we consider the full detector with longer (500 mm) scintillator strips, but in this contribution we assume correlation matrix to be diagonal. Currently we achieve $\sigma_z \approx 10$ mm and $\sigma_{\Delta l} \approx 40$ mm. Please note that the last number corresponds to 20 mm error for the position along the emission line as the distance from the line midpoint is equal to $\frac{1}{2}\Delta l$.

Formula (7) is, at least for the range of parameters we have studied, strongly dominated by the Gaussian term $\boldsymbol{b}C^{-1}\boldsymbol{b}$. This term defines an 3σ ellipse (see Fig. 1). For practical purposes we can assume that the kernel is zero outside this ellipse. As it is easier to work with rectangular shapes we also define a bounding box consisting of an rectangle that is circumscribed on the ellipse (see Appendix B).

4. Implementation

The iteration step described by formula (3) can be implemented as described by the pseudocode in listing of implementation of the reconstruction iteration routine:

```
1  for (auto p_l :  pixels) {
2      rho_new[p_l] = 0.0;
3  }
```

```
4  for (auto e_j :  events) {
5      auto denominator = 0.0;
6      for (auto i :  ellipse(e_j)) {
7          kernel[i] = p(e_j, i);
8          denominator += kernel[i] * rho[i];
9      }
10     for (auto i :  ellipse(e_j)) {
11         rho_new[i] += rho[l] * kernel[i] / denominator;
12     }
13 }
```

Loops `for (auto e_j :  events)` on lines 6 and 10 iterate over all pixels in the 3σ ellipse of the event $\widetilde{\boldsymbol{e}}$. To calculate pixels contributing to this ellipse we first need to determine its bounding box in pixel space. Once bounding box is calculated we loop only through pixels inside this bounding box. Each pixel is then tested if its center point resides inside or outside of the ellipse. Only then the whole kernel is calculated. The results are cached and used subsequently in the second loop.

4.1. CPU

The central processing unit (CPU) implementation follows essentially the algorithm from listing. We use OpenMP to parallelize the outer loop (line 5) over the events. Each thread writes to its own copy of `rho_new` array which are added at the end of the iteration. Currently we do not take direct advantage of the AVX/SSE instruction set aside of automatic vectorization provided by Intel C++ Compiler.

4.2. GPU implementation

Next step was a naive GPU implementation based on our reference CPU implementation where each thread processes all pixels of single event, so few thousands of events are processed simultaneously by hardware threads.

Such approach has however serious drawback on GPU hardware, which is essentially a vector computer. On the NVIDIA CUDA architecture that we use, the threads are collected in batches of 32 threads called *warps*. All threads in a warp must execute the same instruction in parallel (SIMD). In the naive implementation each thread is processing different events with different number of pixels. That amounts to a double loop with loops bounds different across the threads of a warp. This leads to severe *thread divergence* and as we have discovered carries a much higher penalty than naively expected. One would expect that the execution time of a warp would be approximately the time needed to execute the longest loops, but as it turned out it is much higher. Additionally, we cannot cache visited pixels and their kernel results since it is not enough registers or shared memory to store such information given each thread processes separate event.

We can circumvent that using different pixel calculation scheduling where whole warp of 32 threads calculates a single event. This is called by us *warp granularity* (see Fig. 4). As each thread in a warp processes a single pixel from the same event there is no divergence. Different

events are processed by different warps which can run independently. This algorithm also lets us better leverage available shared memory and registers. However, processor cycles are still wasted by the threads that fail the bounding ellipse test.

First it has to be noted that single event is calculated in two passes. First we need to calculate denominator of (3). This pass needs bounding box to be calculated first, then each pixel in this pass is tested with *3-sigma* ellipse equation.
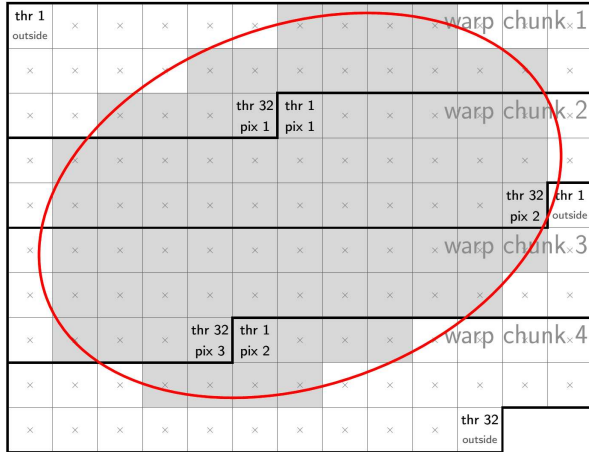

Fig. 4. Warp granularity (whole event processed by single warp).

During first pass *warp granularity* gives us opportunity to cache visited pixels and kernel (7) in shared memory and registers, so the second pass can loop only through visited already pixels without a need to test them for ellipse inclusion. Also we can cache kernel results in registers, since each thread in warp is likely to visit just few pixels of single event.

Calculation of the denominator requires adding the contributions from the 32 threads of the warp. We have done this using the new shuffle instructions introduced in the Kepler architecture. This gave a notable performance boost over standard reduction algorithm using shared memory [12].

Final optimization is to access ρ (previous iteration image buffer) as texture. This produces noticeable performance boost by using hardware GPU texture unit cache and special 2D access optimized memory layout. However it can be observed that memory access still takes around 35% of overall iteration time after optimizations.

5. Benchmarks and results

We have benchmarked our GPU implementation on NVIDIA GeForce GTX 770 commodity card with 4 GB memory and compute capability 3.0 using CUDA SDK 6.5, CPU implementation on Intel Xeon CPU E5-1650 v2 @ 3.50 GHz with 6 cores using ICC 15.0.0 (Intel Composer XE 2015). The benchmark results for 10^7 events are as follows: CPU OpenMP — 11.87 s, GPU Thread

— 0.69 s, GPU Warp — 0.47 s, speedup CPU/Warp — 25. The results of reconstruction of the phantom after different number of iterations are presented in Fig. 3c–f.

6. Summary and outlook

We have implemented and tested our reconstruction kernel on simulated data using realistic parameters obtained from experimental measurements. As seen from Fig. 3c–f the results are very encouraging, considering the simplicity and the resolution of our setup. Implementing the reconstruction algorithm on the commodity GPU provided a 25-fold speedup that allows real-time processing. One should note however that this speed is partly due to not taking advantage of the CPU vector AVX instruction set. The reason for this is that as we have already pointed out in [13] the CUDA and OpenCL programming model is inherently vectorized while CPU is still viewed as superscalar processor with vector instructions mixed in. This is only now slowly changing with introduction of new compiler pragmas to deal explicitly with vectorization in a similar way as OpenMP deals with parallelization.

In derivation of Eq. (7) we have assumed a very simple detector geometry with scintillators approximated by thin lines. In reality they have a rectangular cross-section of 5×20 mm^2. To some extent this was taken into account by using the errors estimated from real scintillators. The model however must be validated on real data (which is already collected) and this is a subject of an ongoing investigation.

Appendix A, Phantom

Phantom definition is given in Table. Each row corresponds to an ellipse with center (x, y) and the half-axes a and b rotated by angle ϕ counterclockwise. The ρ denotes the relative density of the tracer. When two ellipses overlap the ρ is taken from the topmost one in Table.

TABLE

Phantom description, all dimensions are in milimeters, and angles in degrees.

x	y	a	b	ϕ	ρ
0	0	30	60	0	0.3
50	-62	10	33	-40	0.3
-50	-63	20	33	45	0.5
60	65	13	14	0	0.5
35	55	12	12	0	0.7
0	0	120	110	0	0.1

Appendix B, Bounding box

Given an ellipse defined by the equation

$$Ay^2 + Cyz + Bz^2 = R^2 \tag{13}$$

its bounding box is a rectangle with lower left corner at

$$y = -\frac{R}{\sqrt{A - \frac{C^2}{B}}}, \qquad z = -\frac{R}{\sqrt{B - \frac{C^2}{A}}} \tag{14}$$

and symmetric upper right corner. This combined with

$$\boldsymbol{b}C^{-1}\boldsymbol{b} = 2\frac{(\Delta z - \Delta y \tan\widetilde{\theta})^2}{\sigma_z^2} + 4\frac{\Delta y^2}{\sigma_{\Delta l}^2 cos^2\widetilde{\theta}} =$$
$$\Delta z^2 \frac{2}{\sigma_z^2} - 2\Delta z \Delta y \frac{2\tan\widetilde{\theta}}{\sigma_z^2}$$
$$+\Delta y^2 \left(\frac{2\tan^2\widetilde{\theta}}{\sigma_z^2} + \frac{4}{\sigma_{\Delta l}^2 \cos^2\widetilde{\theta}}\right) \tag{15}$$

allows us to calculate the bounding box of the 3σ ellipse for each event.

Acknowledgments

References

[1] P. Moskal, N. Zoń, T. Bednarski, P. Białas, E. Czerwiński, A. Gajos, D. Kamińska, Ł. Kapłon, A. Kochanowski, G. Korcyl, J. Kowal, P. Kowalski, T. Kozik, W. Krzemień, E. Kubicz, Sz. Niedźwiecki, M. Pałka, L. Raczyński, Z. Rudy, O. Rundel, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, A. Strzelecki, A. Wieczorek, W. Wiślicki, M. Zieliński, *Nucl. Instrum. Methods Phys. Res. A* **764**, 317 (2014).

[2] P. Moskal, N. Zoń, T. Bednarski, P. Białas, E. Czerwiński, A. Gajos, D. Kamińska, Ł. Kapłon, A. Kochanowski, G. Korcyl, J. Kowal, P. Kowalski, T. Kozik, W. Krzemień, E. Kubicz, Sz. Niedźwiecki, M. Pałka, L. Raczyński, Z. Rudy, O. Rundel, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, A. Strzelecki, A. Wieczorek, W. Wiślicki, M. Zieliński, *Nucl. Instrum. Methods Phys. Res. A* **775**, 54 (2015).

[3] M. Pałka, P. Moskal, T. Bednarski, P. Białas, E. Czerwiński, Ł. Kapłon, A. Kochanowski, G. Korcyl, J. Kowal, P. Kowalski, T. Kozik, W. Krzemień, M. Molenda, Sz. Niedźwiecki, M. Pawlik, L. Raczyński, Z. Rudy, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, A. Strzelecki, W. Wiślicki, M. Zieliński, N. Zoń, *Bio-Algorithms Med-Syst.* **10**, 41 (2014).

[4] G. Korcyl, P. Moskal, T. Bednarski, P. Białas, E. Czerwiński, Ł. Kapłon, A. Kochanowski, J. Kowal, P. Kowalski, T. Kozik, W. Krzemień, M. Molenda, Sz. Niedźwiecki, M. Pałka, M. Pawlik, L. Raczyński, Z. Rudy, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, A. Strzelecki, W. Wiślicki, M. Zieliński, N. Zoń, *Bio-Algorithms Med-Syst.* **10**, 37 (2014).

[5] http://koza.if.uj.edu.pl/pet/.

[6] L. Raczyński, P. Moskal, P. Kowalski, W. Wiślicki, T. Bednarski, P. Białas, E. Czerwiński, Ł. Kapłon, A. Kochanowski, G. Korcyl, J. Kowal, T. Kozik, W. Krzemień, E. Kubicz, M. Molenda, I. Moskal, Sz. Niedźwiecki, M. Palka, M. Pawlik-Niedźwiecka, Z. Rudy, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, A. Strzelecki, A. Wieczorek, M. Zieliński, N. Zoń, *Nucl. Instrum. Methods Phys. Res. A* **764**, 186 (2014).

[7] P. Moskal, T. Bednarski, P. Białas, M. Ciszewska, E. Czerwiński, A. Heczko, M. Kajetanowicz, Ł. Kapłon, A. Kochanowski, G. Konopka-Cupiał, G. Korcyl, W. Krzemień, K. Łojek, J. Majewski, W. Migdał, M. Molenda, Sz. Niedźwiecki, M. Pałka, Z. Rudy, P. Salabura, M. Silarski, A. Słomski, J. Smyrski, J. Zdebik, M. Zieliński, *Nucl. Med. Rev. C* **15**, 81 (2012).

[8] P. Moskal, T. Bednarski, P. Białas, M. Ciszewska, E. Czerwiński, A. Heczko, M. Kajetanowicz, Ł. Kapłon, A. Kochanowski, G. Konopka-Cupiał, G. Korcyl, W. Krzemień, K. Łojek, J. Majewski, W. Migdał, M. Molenda, Sz. Niedźwiecki, M. Pałka, Z. Rudy, P. Salabura, M. Silarski, A. Słomski, J. Smyrski, J. Zdebik, M. Zieliński, *Nucl. Med. Rev. C* **15**, 68 (2012).

[9] L. Parra, H.H. Berret, *IEEE Trans. Med. Imaging* **17**, 228 (1998).

[10] P. Białas, J. Kowal, A. Strzelecki, T. Bednarski, E. Czerwiński, Ł. Kapłon, A. Kochanowski, G. Korcyl, P. Kowalski, T. Kozik, W. Krzemień, M. Molenda, P. Moskal, Sz. Niedźwiecki, M. Pałka, M. Pawlik, L. Raczyński, Z. Rudy, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, W. Wiślicki, M. Zieliński, *Acta Phys. Pol. B Proc. Suppl.* **6**, 1027 (2013).

[11] P. Białas, J. Kowal, A. Strzelecki, T. Bednarski, E. Czerwiński, Ł. Kapłon, A. Kochanowski, G. Korcyl, P. Kowalski, T. Kozik, W. Krzemień, M. Molenda, , I. Moskal, P. Moskal, Sz. Niedźwiecki, M. Pałka, M. Pawlik, L. Raczyński, Z. Rudy, P. Salabura, N.G. Sharma, M. Silarski, A. Słomski, J. Smyrski, W. Wiślicki, M. Zieliński, N. Zoń, *Bio-Algorithms Med-Syst.* **10**, 9 (2014).

[12] M. Harris, http://docs.nvidia.com/cuda/samples/6_Advanced/reduction/doc/reduction.pdf.

[13] P. Bialas, J. Kowal, A. Strzelecki, *Comput. Inform.* **33**, 1001 (2014).