GPU Accelerated Rotation-Based Emission Tomography Reconstruction

S. Pedemonte[1], A. Bousse[2], K. Erlandsson[2], M. Modat[1], S. Arridge[1], B. F. Hutton[2], S. Ourselin[1]

***Abstract*—Stochastic methods based on Maximum Likelihood Estimation (MLE) provide accurate tomographic reconstruction for emission imaging. Moreover methods based on MLE allow to include an accurate physical model of the imaging setup in the reconstruction process, thus enabling quantitative reconstruction of radio-tracer activity distribution. It has been shown that inclusion of a spatially dependent PSF that models dependence of the CDR with distance from the detector, improves the quality of reconstruction in terms of noise and bias.**
The computational complexity associated with stochastic methods has limited adoption of such algorithms for clinical use and inclusion of the PSF further increases the computational cost. This work proposes an accelerated implementation of a reconstruction algorithm specifically designed to take advantage of the architecture of a General Purpose Graphics Processing Unit (GPGPU).

I. Introduction

Iterative reconstruction methods based on a stochastic model of the emission process [1], [2], [3] have been widely shown to provide better image quality than analytic reconstruction [4], [5]. The reason for the improvement in image quality is that photon count statistics are taken in account in the model of the imaging system; furthermore stochastic methods facilitate the inclusion of complex system models that take into account detailed collimator and detector response (CDR).
The CDR, including collimator geometry, septal penetration and detector response, may be taken into account in a stochastic reconstruction algorithm in the form of a Point Spread Function (PSF) that modulates the response of an ideal Gamma Camera [6][7].
The computational complexity associated with stochastic reconstruction methods however still limits their application for clinical use and inclusion of complex system models further increases the computational demand. Projection and backprojection constitute the most burdensome part of a reconstruction algorithm in terms of computational resources and memory and are performed recursively at each iteration step.

Efficient computation of line integrals for projection and backprojection by ray-tracing was proposed by Siddon [8]. However with a ray-based approach it becomes inefficient to include a depth-dependent PSF, as this requires the casting of a number of rays within a *tube of response* for PET [9] and a *cone of response* for SPECT [6]. Furthermore, though there exist GPGPU accelerated implementations [10], ray-based projectors cannot fully exploit Single Instruction Multiple Data (SIMD) architectures such as GPGPUs because of sparsity of data representation and low arithmetic intensity due to independence of the rays.

This work proposes a GPU accelerated implementation of the rotation-based projection and backprojection algorithm proposed by Zeng and Gullberg [6]. This algorithm drastically reduces memory requirements and allows to perform convolution with the PSF in the frequency domain in $O(N\ logN)$ by means of Fast Fourier Transform.
Rotation-based projection and backprojection are particularly well suited to GPGPU acceleration because reorganization of the data into a regular grid yields efficient use of the *shared* memory and of the global memory bandwidth provided by the GPU architecture. Moreover the algorithm takes advantage of the hardware based trilinear re-sampling, offered by the GPU's 3-D texture memory fetch units.

The aim of the proposed algorithm and software design is to provide a software library for fast GPU accelerated iterative emission tomographic reconstruction. Design criteria include performance, modularity, accessibility of the code, re-usability and portability.

II. Method

A. Projector and backprojector

Let the radio-pharmaceutical activity within the region of interest of the patient's body be a continuous function denoted by $\tilde{y}$. In order to readily discretize the reconstruction algorithm, it is convenient to imagine that the activity is in first place discrete in space [1]. Let us approximate $\tilde{y}$ by a set

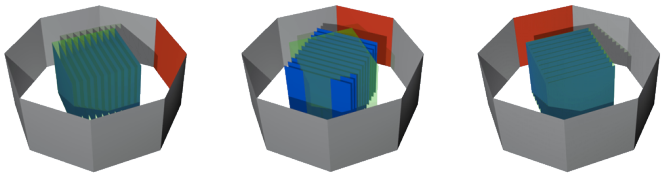

Fig. 1. Rotation-based projection: the activity is re-sampled on a regular grid aligned with a camera and then projected. This enables FFT based convolution with the (depth-dependent) collimator-detector response.

This grant has been supported by the Engineering and Physical Sciences Research Council under Grant EP/G026483/1.
UCL/UCLH is partially supported by a UK Depart of Health NIHR Biomedical Research Centres funding scheme.

[1]Centre for Medical Image Computing, University College London, London, UK. (Email: *s.pedemonte@cs.ucl.ac.uk*)
[2]Institute of Nuclear Medicine, University College London Hospitals NHS Trust, 235 Euston Road (T 5), London NW1 2BU, UK

of point sources $y = y_b, b = 1, .., N_b$ displaced on a regular grid.

As each point source emits radiation at an average rate y_b proportional to the local density of radio-tracer and emission events in a same voxel are not time correlated, the number of emissions in the unit time is a Poisson distribution of expected value y_b. The geometry of the system and attenuation in the patient determine the probability p_{bd} that a photon emitted in b is detected at detector pixel d. From the sum property of the Poisson distribution, the photon count in d has Poisson *pdf* with expected value $\sum_b p_{bd} y_b$. Given activity y, the probability to observe counts z is

$$p(z|y) = \prod_{d=1}^{N_d} \mathcal{P}(\sum_b p_{bd} y_b, z_d) \tag{1}$$

Amongst all the activity configurations that might have generated the observed photons, the activity that maximizes the *likelihood* function is optimal in the sense of the L2 norm of the difference from the true value, for the log linear distribution $p(z|y)$ [11].

$$\hat{y} = \arg\max_y \ p(z|y) = \arg\max_y \ \log p(z|y) \tag{2}$$

Expanding $log\ p(z|y)$:

$$\hat{y} = \arg\max_y \sum_{d=1}^{N_d} \left(\sum_b p_{bd} y_b + z_d \log \sum_b p_{bd} y_b \right) \tag{3}$$

A gradient-based optimization algorithm, such as gradient ascent, requires the gradient of the likelihood function with respect of the activity in each point source, differentiating 3:

$$\frac{\partial p(z|y)}{\partial y_r}|_{y=\bar{y}} = \sum_{d=1} p_{bd} + \sum_{d=1} p_{bd} \frac{z_d}{\sum_{b'} p_{b'd}\ y_{b'}}|_{y=\bar{y}} \tag{4}$$

$\sum_{b'} p_{b'd}\ y_{b'}$ is referred to as projector, and $\sum_d p_{bd} f_d$ as backprojector of f_d.

Similarly the Expectation Maximization algorithm for maximization of the *likelihood* (MLEM) implies projection and backprojection [1]:

$$\hat{\lambda}_b^{(k+1)} = \hat{\lambda}_b^{(k+1)} \frac{1}{\sum_d p_{bd}} \sum_d p_{bd} \frac{z_d}{\sum_{b'} p_{b'd}\ y_{b'}} \tag{5}$$

In case of ideal CDR, with a parallel hole collimator, the system matrix p_{bd} is non-zero only along lines perpendicular to the collimator entry surface and the projection is a line integral operator. A more detailed system model accounts for the sensitivity of each detector pixel to radiation emissions from each voxel. With a planar detector, coupled to a parallel hole, cone beam or fan beam collimator, the sensitivity is invariant to shift along the detector plane. With shift invariant CDR, projection is factorisable into a line integral operator and a convolution operator [6]. The CDR is generally dependent upon the distance from the detector plane.

B. Rotation-based projector and backprojector

The rotation-based projection and backprojection algorithm proposed by Zeng and Gullberg [6] was adopted as it suits the GPU architecture and is convenient to be incorporated with depth-dependent response functions. For each position of the gamma camera, the image matrix volume is rotated so that the front face of the volume faces the detection plane - Figure 1. As the image is re-interpolated on a grid that is aligned with the detection plane, all the point sources that lay on a same plane parallel to the detector are now at the same distance from the detector. A depth-dependent PSF that models the CDR can be incorporated efficiently by convolving each parallel plane with the PSF that models the relative to the distance of the plane from the collimator. The convolution can be performed by multiplication in the frequency domain, reducing the complexity from $O(N^2)$ to $O(N\ logN)$.

Backprojection similarly takes advantage of rotation to incorporate the depth dependent PSF. Details of the implementation of the projector and backprojector are given in the next section.

C. GPU accelerated rotation-based projector and backprojector

For problems that present enough task parallelism, state of the art GPUs can provide an acceleration of up to 10x over a high end CPU, however the speedup can increase by another order of magnitude if the structure of the algorithm allows for efficient use of the *shared memory* of the GPU [12], [13], [14]. While on CPUs the cache hierarchy compensates costly accesses to external RAM and cache heuristics account for a large class of computational problems, the simplified memory hierarchy of GPUs requires careful design of the algorithms for efficient memory access. The external memory is directly exposed to the programmer, who has to consider explicitly coalesced access due to the mismatch between data rate and cycle time of the DDR memory. On the other hand the simpler structure of the memory and vicinity of the RAM to the processor yield data throughput 10 times higher than the throughput between CPU and RAM - Figure 2.

The fast RAM memory of the GPUs explains the $10\times$ speedup, however the Single Instruction Multiple Data (SIMD) architecture and the *shared memory* of the GPU provide additional speedup for a class of computational problems. In the SIMD architecture, many processor cores fit on the same chip due to simplified design of the processor cores, which are grouped, in the case of NVidia GPUs, in a *multiprocessor* with a single common fetch unit. Multiple cores can operate

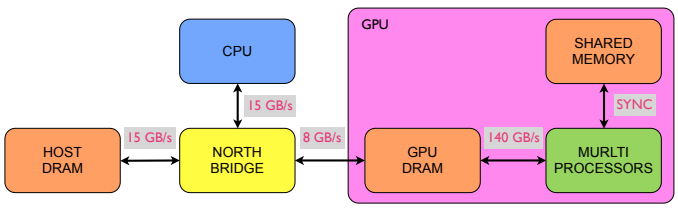

Fig. 2. Memory structure of the host machine and GPU and typical memory bandwidth.

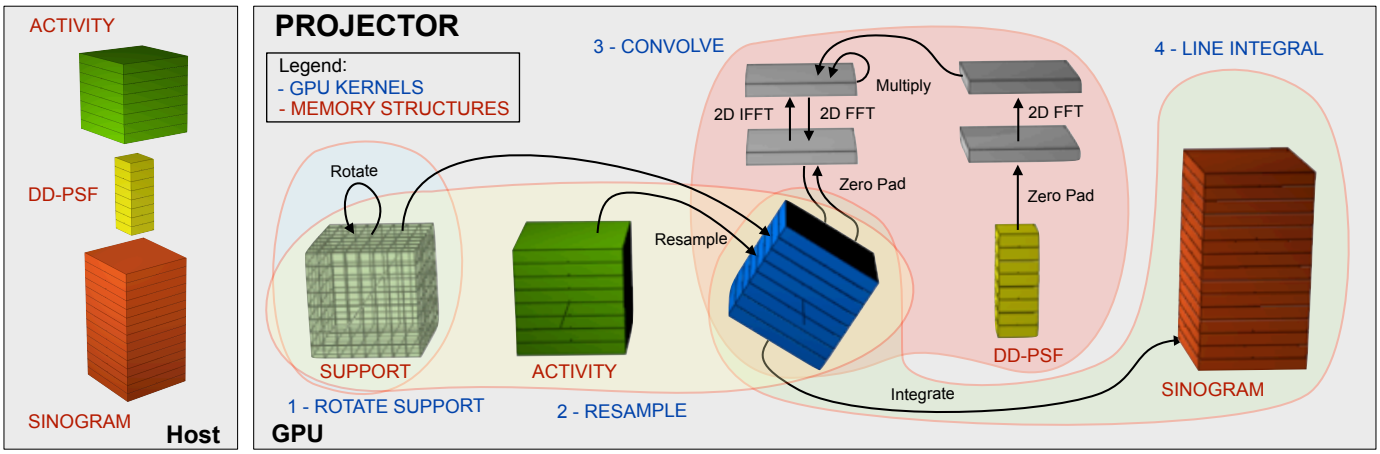

Fig. 3. Rotation-based projection on GPU

concurrently in a *multiprocessor* if they execute the same instruction, so if the computational problem is such that the same operation is performed on multiple segments of data, the GPU can use a great number of processors at the same time. As the RAM data throughput would still be too low to continuously feed data to all the cores, the processor cores that are grouped in a *multiprocessor* have access to an on-chip memory that can be read and written concurrently by all the cores in a single clock cycle, through multiple data paths. The size of the *shared memory* is limited to a few Kbytes on currently available GPUs. This design offers the possibility to exploit the full power of the cores as long as the cores in a multiprocessor can reuse the data that resides in the *shared memory*, hiding accesses to the global memory, that is hundreds of times slower. In order to take full advantage of the GPU architecture, a computational problem needs to expose parallel tasks that run on each multiprocessor, each task being partition-able into serial tasks that use limited memory (up to a few Kbytes) and present a high ratio between the number of operations and the accesses to memory.

Another feature offered by Nvidia GPUs is hardware trilinear interpolation. A portion of memory may be specified as a 1D, 2D or 3D array and floating point memory addresses are accepted by the memory access unit, which decodes the non-integer address, reads the values stored in the nearest memory locations and interpolates linearly.

Ray-tracing on GPU can take advantage of the fast RAM memory of the GPU and of hardware interpolation, however independence of the rays impedes efficient use of the *shared memory*. It might be possible to take advantage of the shared memory as the rays share some information, however that would imply processing concurrently multiple partial rays in blocks in a way that exposes the data in common. Rotation-based projection and backprojection reorganize data in a way that exposes the data locality.

1) Projector: Activity and the depth dependent PSF are copied to the GPU global memory and additional memory is allocated for the sinogram and for each of the structures depicted in Figure 3. The support of the image (ordered list of the x, y, z indexes of the image voxels) is extracted and stored in global memory.

For each position of the gamma camera, the activity matrix volume is rotated so that the front face of the volume faces the detection plane. In order to optimize the usage of the GPU, rotation is performed by multiplying the support of the image by the rotation matrix, then the image is re-sampled at the locations specified by the rotated support. Rotation of the support maximizes the *device occupancy* (concurrent usage of the multiprocessors) and takes advantage of the *shared memory* by partitioning the matrix multiplication [15]. The trilinear interpolation is performed in hardware by the *texture fetch* unit of the GPU at the cost of a memory access and coalesced memory accesses are obtained by partitioning the memory transfers in blocks.

For each camera's position, the activity is rotated from its initial position, rather than from the previous camera's position, in order to minimize interpolation errors. After reinterpolation, each image plane parallel to the camera is convolved with the PSF. Convolution is performed by zero-padding the plane to double its linear size, computing its 2D FFT, multiplying by the FFT of the zero-padded PSF, back-transforming and truncating. As depicted by the arrows in Figure 3, the convolution is performed in place, in order to minimize memory occupancy. 2D FFT is performed by means of the CUDA CUFFT library that takes into account all the architectural factors and constraints of CUDA, memory coalesced access, bank conflicts and efficient shared memory usage.

Finally a kernel sums all planes and stores the result in the sinogram data structure. Shared memory cannot be used in the summation step as the number of operations (sums) is exactly equal to the number of memory accesses, however

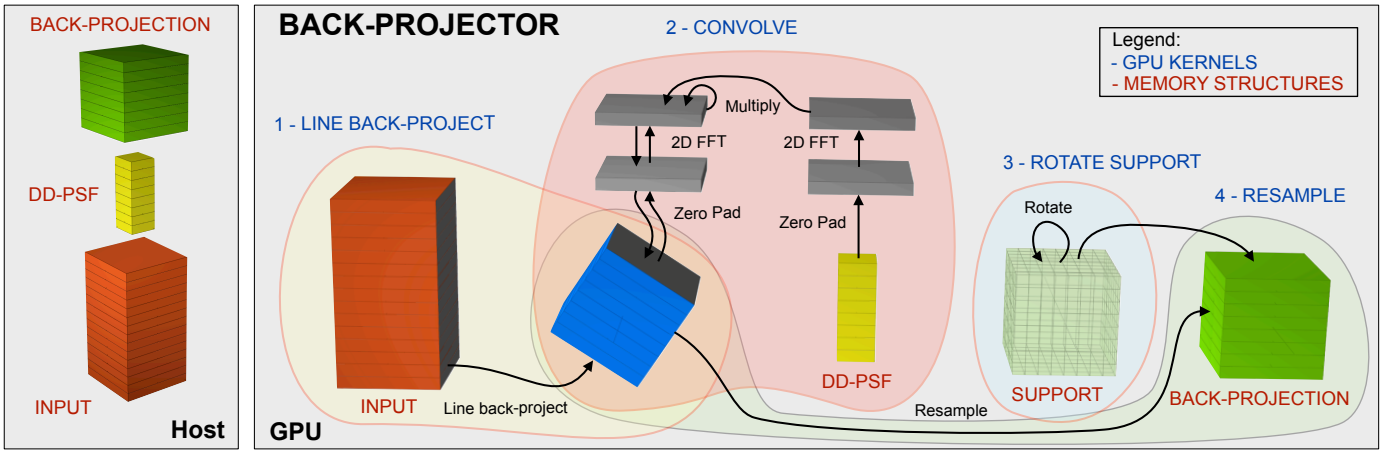

Fig. 4. Rotation-based backprojection on GPU

device occupancy and memory coalescing are optimized by partitioning the sums in blocks.

2) Backrojector: The sinogram and the PSF are transfered from the host machine RAM to the GPU global memory and all the structures depicted in Figure 4 are allocated on the GPU memory. Two volumes are allocated for the backprojection, a rotating volume and a fixed volume that is initialized to 0 and contains in the end the result of the backprojection.

One projection at a time is extracted from the sinogram data structure and the value on each pixel is backprojected to the rotating volume along lines perpendicular to the detection plane. This step is performed by a GPU *kernel* that copies a pixel to a local *register* and then copies it back into all the voxels in the same column of the rotating volume. Each plane of the rotating volume is then convolved in place with the PSF by multiplication in the frequency domain. The rotating volume is rotated to the zero position and then accumulated into the fixed volume.

The same series of operations is repeated for each camera's position.

D. Software design

A software library for projection and backprojection was designed with the aim of accessibility and re-usability. The code was developed in C and CUDA, adopting NiftiLib data structures to conveniently handle image volumes. The library provides a development Application Programming Interface (API) based on NiftiLib data structures and a C array API to simplify integration and usability - Figure 5.

On top of the C array interface, ctypes-based Python and mex-based Matlab extensions provide a high level interface to the projector and backprojector, simplifying the development and prototyping of reconstruction algorithms. The complexity of the GPU accelerated algorithms for projection and backprojection is hidden from the Python and Matlab programmer. Though MLEM reconstruction could be performed entirely on the GPU, the software has been designed to expose functions for projection and backprojection through the C array API and to the Matlab/Python bindings. This way the gradient of the likelihood is made available to the Matlab/Python workspace, enabling the development of optimization schemes, Bayesian image reconstruction algorithms, algorithms for joint estimation of image parameters and multi-modal reconstruction.
This comes at the cost of a number of data transfers from the RAM of the host machine to the global memory of the GPU, however the number of transfers is only equal to the number of iterations of the given iteration algorithm and practically reduces performance by less than 5%.

The sample code below implements MLEM reconstruction in Matlab. The two functions *et_project* and *et_backproject* hide the complexity of the GPU implementation of the projection and backprojection algorithms.

```
N           = 128;
iterations  = 100;
phantom     = et_spherical_phantom(N,N,N,N/2,N/3,N/2,0,100);
cameras     = [0:1:180]*pi/180;
activity    = ones(N,N,N);
psf         = ones(N,5,5);

sinogram    = poissrnd(et_project(phantom,cameras,psf));
sens        = et_backproject(ones(N,N,180),cameras,psf);

for i=1:iterations
  residual = sinogram ./ et_project(activity, cameras, psf);
  activity = activity .* et_backproject(residual) ./ sens;
end
```

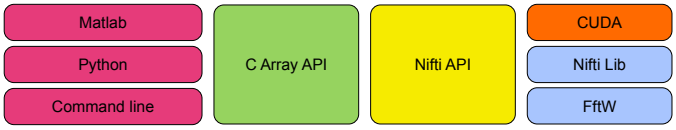

Fig. 5. Application Program Interface (API) for GPU accelerated reconstruction software.

III. Performance

While the same algorithms and software apply to different emission tomographic imaging modalities, the performance of the proposed projection and backprojection algorithms was evaluated in the case of OSEM reconstruction for SPECT. The iterative reconstruction algorithm was implemented with the object oriented language Python, in order to ease programming and re-usability. However this choice is independent of the proposed implementation of the projector and backprojector.

Table I reports reconstruction times with GPGPU accelerated projector and backprojector and with an equivalent CPU implementation that was developed and optimized along with the GPU version. For a tomographic image of size 128^3, reconstruction with the GPGPU accelerated algorithms provides a 75-fold speedup, when compared with efficient C implementation of the same algorithm, executed on a Intel Xeon E5430@2.66GHz CPU. Reconstruction time, with GPGPU acceleration, for 128^3 tomographic image size, 180 camera positions and 60 iterations of OSEM (subset order 8) is 12 seconds instead of 15 minutes.

TABLE I
SPECT RECONSTRUCTION WITH OSEM ITERATIVE ALGORITHM, SUBSET ORDER 8, 180 CAMERA POSITIONS, 60 ITERATIONS. RECONSTRUCTION TIME IS REPORTED FOR SEVERAL SIZES OF THE TOMOGRAPHIC DATA, WITH AND WITHOUT GPGPU ACCELERATION.

Tomography size	$\mathbf{32^3}$	$\mathbf{64^3}$	$\mathbf{128^3}$	$\mathbf{256^3}$
Intel Xeon E5430,2.66GHz	$30sec$	$3min$	$15min$	$65min$
NVidia GeForce GTX-285	$6sec$	$8sec$	$12sec$	$45sec$
Acceleration factor	5	22.5	75	86.6

While rotation-based projection and backprojection are alone significantly faster than ray-driven approaches [6], the GPU implementation provides a further dramatic 80 fold speedup. This is about the ideal increase of performance that one would expect with GPU acceleration for algorithms that present enough parallelism, arithmetic intensity and memory coalescing, which validates our approach and implementation. The combination of a rotation-based approach and GPU acceleration provides very fast projection and backprojection with accurate collimator detector response based on depth-dependent point spread function. As reconstruction time is significantly lower than acquisition time, which is typically of the order of $10-20$ minutes for SPECT, the GPU accelerated algorithm achieves real-time processing.

IV. Future work

The software can be extended in order to take into account attenuation of the photon flux. Inclusion of attenuation correction in the rotation-based projector and backprojector, as described in [6], does not involve much additional computational complexity. At present, only the parallel hole collimator geometry for SPECT has been considered. The software may however be modified to take into account other collimator geometries such as cone beam and fan beam, by adding a shearing component to the transformation matrix of the image support. Rebinning of PET sinograms is currently under investigation.

V. Download

The GPU and CPU versions of the presented rotation-based projector and backprojector are open source and can be freely downloaded [16]. The package includes the source of the C/CUDA projector and backprojector, the compiled shared libraries for Linux and Windows, and a mex-based Matlab toolbox with documentation and examples. As the software makes use of the CMake cross platform build system, it can easily be ported to other platforms.

References

[1] L.A. Shepp and Y. Vardi. Maximum Likelihood Reconstruction for Emission Tomography. *IEEE Transactions on Medical Imaging*, 1(2):113–22, 1982.

[2] J. Qi and R.M. Leahy. Iterative reconstruction techniques in emission computed tomography. *Physics in Medicine and Biology*, 51(15):R541–R578, 2006.

[3] S. Borman. The expectation maximization algorithm: A short tutorial. Unpublished paper available at http://www.seanborman.com /publications. Technical report, 2004.

[4] C.A. Johnson, J. Seidel, Carson R.E., Gandler W.R., Green M.V., E. Daube-Witherspoon, and A. Sofer. Evaluation of 3D reconstruction algorithms for a small animal PER camera. *IEEE Transactions on Nuclear Science*, 44(3):1303–1308, Jun 1997.

[5] T. Frese, N.C. Rouze, C.A. Bouman, K. Sauer, and G.D. Hutchins. Quantitative comparison of FBP, EM, and Bayesian reconstruction algorithms for the IndyPET scanner. *IEEE Transactions on Medical Imaging*, 22(2):258–276, Feb 2003.

[6] G.L Zeng and G.T. Gullberg. Frequency Domain Implementation of the Three-Dimensional Geometric Point Response Correction in SPECT Imaging. *IEEE Transactions on Nuclear Science*, 39(5):1444–1454, 1992.

[7] A. Rahmim and H. Zaidi. PET versus SPECT: strengths, limitations and challenges. *Nuclear Medicine Communications*, 29(3):193–207, 2008.

[8] R.L. Siddon. Fast calculation of the exact radiological path for a three-dimensional CT array. *Medical Physics*, 12(2):252–255, 1985.

[9] S. Cho, Q. Li, S. Ahn, B. Bai, and R.M. Leahy. Iterative Image Reconstruction Using Inverse Fourier Rebinning for Fully 3-D PET. *IEEE Transactions on Medical Imaging*, 26:745–756, 2007.

[10] F. Xu. Fast Implementation of Iterative Reconstruction with Exact Ray-Driven Projector on GPUs. *Tsinghua Science & Technology*, 05(20):30–35, 2009.

[11] H. Cramer. *Mathematical Methods of Statistics*. Princeton University Press, 1957.

[12] J.E. Stone, J.C. Phillips, P.L. Freddolino, D.J. Hardy, L.G. Trabuco, and K. Schulten. Accelerating molecular modeling applications with graphics processors. *Journal of Computational Chemistry*, 28:2618–2640, 2007.

[13] I. Ufimtsev and T. Martinez. Quantum chemistry on grapical processing units. 1. Strategies for two-electron integral evaluation. *Journal of Chemical Theory and Computation*, 4(2):222–231, 2008.

[14] S. Pedemonte, A. Gola, A. Abba, and C. Fiorini. Optimum real-time reconstruction of gamma events for high resolution Anger camera. In *IEEE Nuclear Science Symposium Conference Record*, pages 3388–3394, November 2009.

[15] NVidia CUDA Programming Guide, version 2.3, Jan 2008.

[16] http://niftyrec.sourceforge.net/ - NiftiRec tomographic reconstruction toolbox.