GPU implementation of a 3D bayesian CT algorithm and its application on real foam reconstruction

Nicolas Gac, Alexandre Vabre, Ali Mohammad-Djafari, Asier Rabanal, Fanny Buyens

GPU IMPLEMENTATION OF A 3D BAYESIAN CT ALGORITHM AND ITS APPLICATION ON REAL FOAM RECONSTRUCTION

Nicolas Gac[1]*, Alexandre Vabre*[2]*, Ali Mohammad Djafari*[1]*, Asier Rabanal*[1] *and Fanny Buyens*[2]

[1] L2S, Laboratoire des Signaux et Systemes (CNRS-SUPELEC-UPS), F-91191 Gif sur Yvette, France
[2]CEA, LIST, Laboratoire Images et Dynamique, , F-91191 Gif sur Yvette, France

ABSTRACT

A great number of image reconstruction algorithms, based on analytical filtered backprojection, are implemented for X-ray Computed Tomography (CT) [1, 3]. The limits of these methods appear when the number of projections is small, and/or not equidistributed around the object. In this specific context, iterative algebraic methods are implemented. A great number of them are mainly based on least square criterion. Recently, we proposed a regularized version based on Bayesian estimation approach. The main problem that appears when using such methods as well as any iterative algebraic methods is the computation time and especially for projection and backprojection steps. In this paper, first we show how we implemented some main steps of such algorithems which are the forward projection and backward backprojection steps on GPU hardware, and then we show some results on real application of the 3D tomographic reconstruction of metallic foams from a small number of projections. Through this application, we also show the good quality of results as well as a significant speed up of the computation with GPU implementation.

Index Terms— Computed Tomograhy (CT), Iterative 3D reconstruction, Bayesian estimation, GPU implementation

1. INTRODUCTION

The inverse problem we solve is to reconstruct the object $\boldsymbol{f}$ from the projection data $\boldsymbol{g}$ collected by a cone beam 3D CT. The link between $\boldsymbol{f}$ and $\boldsymbol{g}$ can be expressed as :

$$\boldsymbol{g} = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{\epsilon} \tag{1}$$

where $\boldsymbol{H}$ is the forward projection matrix operator modeling the acquisition system and $\boldsymbol{\epsilon}$ represents all the errors (modeling and measurement noise). The element H_{ij} represents the participation of the j pixel in the i data point.

In this discretized presentation of the CT forward problem, the backprojection (BP) solution can be expressed as $\widehat{\boldsymbol{f}}_{BP} = \boldsymbol{H}^t\boldsymbol{g}$ where $\boldsymbol{H}^t$ is the transpose of $\boldsymbol{H}$ and the filtered backprojection (FBP) method which is also equivalent to the Least squares (LS) solution can be expressed as $\widehat{\boldsymbol{f}}_{FBP} = (\boldsymbol{H}^t\boldsymbol{H})^{-1}\boldsymbol{H}^t\boldsymbol{g}$. The LS solution $\widehat{\boldsymbol{f}}_{LS} = \arg\min_{\boldsymbol{f}}\left\{Q(\boldsymbol{f}) = \|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2\right\}$ as well as the quadratic regularization (QR) solution

$$\widehat{\boldsymbol{f}}_{QR} = \arg\min_{\boldsymbol{f}}\left\{J(\boldsymbol{f}) = \|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2 + \lambda\|\boldsymbol{D}\boldsymbol{f}\|^2\right\} \tag{2}$$

can be obtained by a gradient based optimization algorithm which can be described as follows:

$$\left\{\begin{array}{l} \boldsymbol{f}^{(0)} = \boldsymbol{H}^t\boldsymbol{g} \\ \boldsymbol{f}^{(i+1)} = \boldsymbol{f}^{(i)} + \alpha\left[\boldsymbol{H}^t(\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}^{(i)}) + \lambda\boldsymbol{D}^t\boldsymbol{D}\boldsymbol{f}^{(i)}\right] \end{array}\right. \tag{3}$$

where α is a fixed, variable or computed optimally step size and (i) is the iteration number. Looking at this iterative algorithm, we can distinguish, at each iteration the following operations:

1. Forward projection operation: $\widehat{\boldsymbol{g}} = \boldsymbol{H}\widehat{\boldsymbol{f}}$
2. Computation of the residuals: $\delta\boldsymbol{g} = \boldsymbol{g} - \widehat{\boldsymbol{g}}$
3. Backprojection operation of the residual: $\delta\boldsymbol{f}_1 = \boldsymbol{H}^t\delta\boldsymbol{g}$
4. Computation of the regularization or a priori term: $\delta\boldsymbol{f}_2 = \lambda\boldsymbol{D}^t\boldsymbol{D}\widehat{\boldsymbol{f}}$
5. Updating of the solution for the next iteration: $\boldsymbol{f}^{(i+1)} = \boldsymbol{f}^{(i)} + \alpha(\delta\boldsymbol{f}_1 + \delta\boldsymbol{f}_2)$

As we can see the implementation of such iterative algorithm as well as any other more sophisticated algorithm such as the Bayesian estimation approach we propose needs these operations. The two main steps are the steps 1 and 3. As we will see later, we implemented these two steps using GPU. So, one of the main contribution of this paper is the presentation of this implementations and their relative performances. The second contribution of this paper is adaptation of a particular Bayesian estimation approach with appropriate prior modelling which is particularly adapted for our application which is related to Non Destructive Testing (NDT) application.

E-mail : nicolas.gac@lss.supelec.fr, alexandre.vabre@cea.fr, djafari@lss.supelec.fr

In the following, first, we present the basic ideas of our Bayesian estimation approach and in particular the prior model we proposed and used. Then, we present the main steps of the resulting Joint Reconstruction-Segmentation-Characterization Algorithm (JRSCA) we developped. Then, we detail as much as possible the implementation on GPU parts and their performances, and finally, we show the 3D reconstruction results obtained for our application and we conclude on this paper.

1.1. Bayesian method

The proposed Bayesian method lies on a prior model for the object $\boldsymbol{f} = \{f(\boldsymbol{r}), \boldsymbol{r} \in \mathcal{R}\}$ where $\boldsymbol{r} = (x, y, z)$ represents a voxel position. This model considers that the object $f(\boldsymbol{r})$ is composed of a finite number K of materials; all voxels of the same material are grouped in compact regions $\mathcal{R}_k$, labeled by a hidden variable $z(\boldsymbol{r}) = k,\ k = 1, \cdots, K$. We then have $\mathcal{R}_k = \{\boldsymbol{r} : z(\boldsymbol{r}) = k\}$. To translate the homogeneity in each class of matierial, we use:

$$p(f(\boldsymbol{r})|z(\boldsymbol{r}) = k, m_k, v_k) = \mathcal{N}(m_k, v_k) \tag{4}$$

and to translate the desir that all the voxels in a given class be grouped in compact regions, we use a Potts-Markov model for $\boldsymbol{z} = \{z(\boldsymbol{r}), \boldsymbol{r} \in \mathcal{R}$:

$$p(\boldsymbol{z}) \propto \exp\left[\sum_{\boldsymbol{r}\in\mathcal{R}}\sum_{k}\alpha_k\delta(z(\boldsymbol{r}) - k) + \gamma\sum_{\boldsymbol{r}'\in\mathcal{V}(\boldsymbol{r})}\delta(z(\boldsymbol{r}) - z(\boldsymbol{r}'))\right] \tag{5}$$

where $\mathcal{V}(\boldsymbol{r})$ means the neighborhood of $\boldsymbol{r}$ and $\{\alpha_k, k = 1, \cdots, K\}$ and γ are the Potts model parameters. The parameters m_k, v_k and also standard variation of the noise v_ϵ are called the hyperparameters $\theta = \{(m_k, v_k, \alpha_k), k = 1, \cdots, K; v_\epsilon\}$. With this prior model and a centered uncorrelated Gaussian model for the noise, we can obtain the expression of all the probability laws $p(\boldsymbol{g}|\boldsymbol{f}, v_\epsilon)$, $p(\boldsymbol{f}|\boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{v})$, $p(\boldsymbol{z}|\gamma, \boldsymbol{\alpha})$ and the joint a posteriori law $p(\boldsymbol{f}, \boldsymbol{z}, \boldsymbol{\theta}|\boldsymbol{g})$ and all the conditionals $p(\boldsymbol{f}|\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{g})$, $p(\boldsymbol{z}|\boldsymbol{f}, \boldsymbol{\theta}, \boldsymbol{g})$ and $p(\boldsymbol{\theta}|\boldsymbol{f}, \boldsymbol{z}, \boldsymbol{g})$ which are needed to estimate jointly the object $\boldsymbol{f}$, the image of $\boldsymbol{z}$ which will show the segmented and classified volume and the parameters $\boldsymbol{\theta}$ which charcterize all the classes.

The iterative algorithm structure is then constituted of three main steps, as follows:

- Reconstruction step: Updating $\boldsymbol{f}$ by computing $\widehat{\boldsymbol{f}}^{(i+1)} = \arg\max_{\boldsymbol{f}}\{p(\boldsymbol{f}|\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{g})\}$. This is done by using a gradient type optimization algorithm.
- Segmentation step: Updating $\boldsymbol{z}$ by generating a sample from $p(\boldsymbol{z}|\boldsymbol{f}, \boldsymbol{\theta}, \boldsymbol{g})$. This needs a sampling algorithm from a Potts-Markov model.
- Characterization step: Updating the hyperparameters using $p(\boldsymbol{\theta}|\boldsymbol{f}, \boldsymbol{z}, \boldsymbol{g})$. This step can be done either analytically or by sampling from known probability laws such as Gaussians or Inverse Gamma.

More details about this method can be found in [5].

2. ALGORITHM SPEED UP

2.1. Introduction

In this paper, we focus more on a hardware speed up of some of these steps. A preliminary study has been conducted to estimate which hardware architecture is the more appropriate to each calculation step: Cell [6], FPGA, CPU, cluster of PC's, graphic processing units [7, 8]. And so from the literature [9], for gradient descent (95 $\%$ of the calculation time), graphic processors such as GPU seem well adapted. The convergence of the algorithm has to be warranted for the different chosen parameters, such as: N (local number of iteration for gradient descent), M (segmentation number of iterations) and I (global number of iterations). The proposed method includes not only a reconstruction of the CT data but also a segmentation of the volume into classes. Recent works have been carried out on similar approaches for binary cases using discrete tomography [10]. Our approach allows to have any number of materials that needed, and also we associate a probability law to belong to a given class [5]. Priors are also introduced on the voxel class estimation according to their neighborhood.

2.2. Implementation of projector and backprojector

For the iterative step of gradient descent, the two main consuming time operations are projection ($\boldsymbol{Hf}$) and backprojection ($\boldsymbol{H}^t\delta\boldsymbol{g}$) which are used to estimate a convergence criterion and its gradient. These two operations represent 95 $\%$ of the computing time.

The follow up of the work aims at speeding up these two steps. GPU hardware, since 2006 is one of the most used tool inside research community. Both simplicity in implementation and performance improvements have imposed scientic community to migrate to such a tool. Recent improvements from NVidia have allowed to dispose of CUDA, this developing environment allows to design operating software with high computing performances.

In order to compute the two matrix operations ($\boldsymbol{Hf}$ and $\boldsymbol{H}^t\delta g$) without the too expensive memory use of H=(h_{ij}) (1 To is needed to store H for a 2048^3 reconstruction), projection and backprojection geometric operators are widely used. This operators compute in line the coefficient h_{ij}, instead of reading a matrix H stored in memory. Differents kinds of projection and backprojection algorithms can be used [11, 12].

For each operator, we choose the one which enables the best implementation on Nvidia GPUs with CUDA. As a consequence, our projection/backprojection pair is unmatched. Thus each operator defines a different matrix H: H_p for projection and H_{bp} for backprojection. Use of unmatched backprojection/projection pairs is widely used. Indeed, effect on convergence is in pratice not penalising during the first iterations [13]. Main difference on backprojection and projection

algorithm is the main loop of computation : for backprojection, the loop is on voxels (voxel-driven) and for projection it is on X rays (ray-driven).

2.3. Backprojection

Backprojection algorithm used is a voxel-driven (main computation loop on voxels) with a bi-linear interpolation done on detector pixels. Loops are ordered in manner to exploit as much as possible the spatial and temporal locality of memory access as described in [9]. In CUDA parallelization scheme, one thread is responsible to one voxel reconstruction. Memory accesses to the 2D projection of the volume is done via the 2D texture available on GPUs which allows a cache access to global memory and a hardwired bi-linear interpolation. Standard software optimizations techniques have been carefully used : pre-computation stored on constant cache-memory, incremental computation used as much as possible and loop unrolling.

2.4. Projection

Projection algorithm used is a ray-driven (main computation loop on rays) with a tri-linear interpolation done on volume voxels. In CUDA parallelization scheme, one thread is responsible to integrate the 3D volume along one X-ray. The volume integration for a ray is done simply by sampling regularly the volume along the ray. Memory accesses to the 3D volume is done via the 3D texture available on GPUs which allows a cache access to global memory and a hardwired tri-linear interpolation. Standard software optimization techniques have been used for projection as well.

3. REAL DATA RECONSTRUCTION

3.1. Metallic foams

Solid foams are a class of materials with a complex behavior related to the properties of the constitutive material, the geometry and the topology of the material distribution. These materials present a very high porosity, and are thus very light, but nevertheless very resistant due to a good distribution and architecture of matter. The most known examples of such materials are bone and wood, or also coral and sponge.

Metallic foams are very recent materials. The application field of these materials is very large: they can be used as deformation absorbers in mechanical engineering or fluid distributors for many applications such as thermal exchangers, fuel cells and electrolysers. A strong need in modeling tools as reliable as possible is necessary to make clearer the behavior of these materials and to design optimal foams for desired application [14]. It is necessary to estimate the mechanisms that control their deformations, their durability versus time or stresses to employ them. It is also necessary to study their behavior versus mass and thermal transfers to address fluid flow applications. In this context, our work is focused on collecting basic knowledge on fluid two-phase flows in metallic foams [15]. A scientific community works on flows in porous media for geology or oil extraction. Our idea is to implement the modelling methods developed in the context of fractured geologic medias and adapt them to the metallic foam structures [16].

However, in order to obtain reliable results from these modelling methods, it is necessary to obtain of a thin topology and geometry foam structures. For topology characterization, the pore size distribution and the specific surfaces are fundamental parameters, i.e. the normalized surface of the foam. A high spatial resolution of three-dimensional structure of the foam (in the magnitude of 5 m) is required for geometry characterization [17]. In the follow-up, we present our studies on water kinetics in open-cell nickel foams using x-ray microtomography. The experiments are conducted on a small sample size (1 mm^3 foam) to estimate the thin geometry and model the water behavior at a scale of few pores.

Data set is made of 96 projections on the 256^2 plane detector. The volume is reconstructed inside the cylindric field of view of the X ray tomograph. Thus, we reconstruct 256 (z dimension) * $\Pi \cdot 128^2$ (x,y dimensions) voxels.

3.2. Reconstruction time

We have used a Nvidia GTX 295 to reconstruct the metallic foam. Only one GPU is used here, no multi-GPU implementation has been done. Reconstruction time are greatly penalised by memory transfer between CPU and GPU.

The purpose of this work is not to evaluate the acceleration factor obtained on GPU (see [9, 6] for time comparaison on CPU and GPU). But compared to the former reconstruction "C++" software used in the CEA lab , we reach about 100 acceleration factor. Previously a 100 iteration reconstruction took days and now it takes hours.

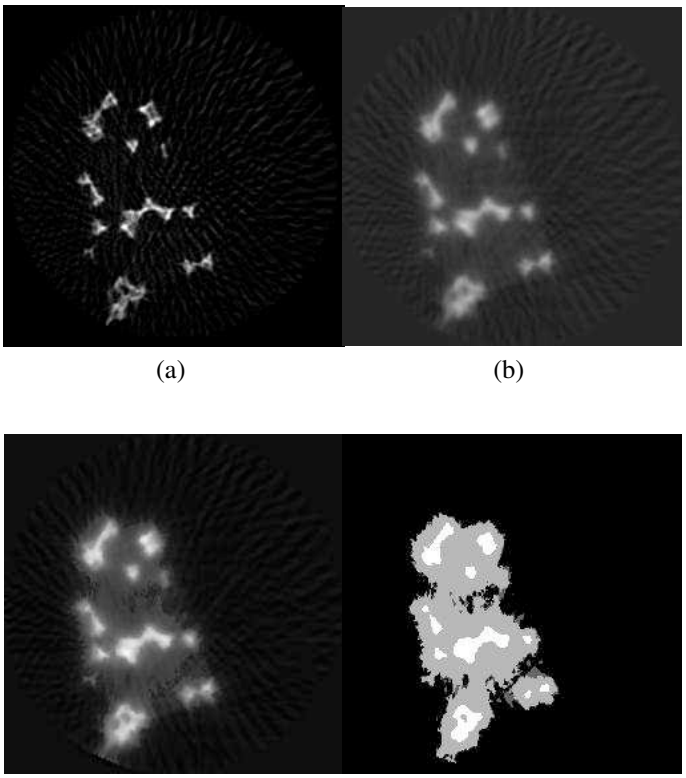
Operator	Time
Projector	755 ms (128 ms for memory transfer)
Backprojector	234 ms (133 ms for memory transfer)

Table 1. Reconstruction time on a GTX 295 ($96 * 256^2$ data)

3.3. Foams reconstructed

As first results, we present here the foam reconstructed with a non Bayesian iterative algorithm and with our Bayesian iterative algorithm. As we can observe on figure 1, while standard algorithm like FDK (a) or an iterative quadratic regularization method (b) does not suceed to reconstruct the water inside the metallic foam, our method suceeds to reconstrut it (c), and provide a segmentation image (d) . The used prior model which suppose that the reconstructed object is constituted of

N compact regions $\mathcal{R}_k$,, is well adapted to this context of data set.



(a) (b)

(c) (d)

Fig. 1. Foam reconstructed : (a) Slice reconstructed with a FDK method (b) Slice reconstructed with a non bayesian method (standard gradient descent after 50 iterations); (c) Slice recontructed with our method (after 50 iterations); (d) Segmentation obtained during iterative reconstruction

4. CONCLUSION AND PERSPECTIVES

We have presented an original method based on a Bayesian statistical method for 3D tomographic reconstructions. The main interest is to apply it to a context of non-consistent data sets, for example with a small number of projections. We have shown a good quality of our first results on an experimental data set with low contrasted regions (air/water as compared to nickel) acquired on the CT set-up of our lab and a signicant speed up of the calculation with GPU implementation.

Both backprojection and projection steps were implemented on GPU. The obtained performance for the global reconstruction time is in the magnitude of 100. Hovewer, performance of our projector can be still improved. In this goal, a Joseph projector implemented also on GPU, would be compared in term of time and qualiy of reconstruction. Our futur work will be focusing on the study of the effect of the unmatched projector/backprojector pair on the reconstruction process. For this purpose, a matched voxel-driven projector has been implemented on CPU.

Our futur goal is to reconstruct 1024^3 real data acquired on the new CT set-up of the lab (1024^2 detector pixels). We are currently working on a multi-GPU implementation in order to handle such large data sets. GPU implementations of computation costly steps as 3D convolution, segmentation step would help to speed-up even more the reconstruction process.

5. REFERENCES

[1] K.J. Batenburg and J. Sijbers, "DART: a fast heuristic algebraic reconstruction algorithm for discrete tomography," 2007, vol. IV, pp. 133–136, IEEE Conference on Image Processing.

[2] Fang Xu and Klaus Mueller, "Real-time 3d computed tomographic reconstruction using commodity graphics hardware," *Physics in Medicine and Biology*, vol. 52, no. 12, pp. 3405–3419, 2007.

[3] Steckmann S. Marone F. Kachelrie M., Knaup M. and Stampanoni M., "Hyperfast o(2048**4) image reconstruction for synchrotron?based x?ray tomographic microscopy," in *MIC proceedings*, 2008.

[4] G. Pratx, G. Chinn, P. D. Olcott, and C. S. Levin, "Fast, accurate and shift-varying line projections for iterative reconstruction using the GPU," *Medical Imaging, IEEE Transactions on*, vol. 28, no. 3, pp. 435–445, March 2009.

[5] S. Fékih-Salem, A. Vabre, and A. Mohammad-Djafari, "Bayesian tomographic reconstruction of microsystems," in *Bayesian Inference and Maximum Entropy Methods, AIP Conf. Proc. 954*, K. et al. Knuth, Ed. MaxEnt Workshops, July 2007, pp. 372–380, American Institute of Physics.

[6] Marc Kachelriess, Michael Knaup, and Olivier Bockenbach, "Hyperfast parallel-beam and cone-beam backprojection using the cell general purpose hardware," *Medical Physics*, vol. 34, no. 4, pp. 1474–1486, April 2007.

[7] W. Xu and K. Mueller, "A performance-driven study of regularization methods for GPU-accelerated iterative CT," in *Workshop on High Performance Image Reconstruction (HPIR)*, 2009.

[8] W. Xu and K. Mueller, "Learning effective parameter settings for iterative CT reconstruction algorithms," in *Workshop on High Performance Image Reconstruction (HPIR)*, 2009.

[9] N. Gac, S. Mancini, M. Desvignes, and D. Houzet, "High speed 3D tomography on CPU, GPU and FPGA," *EURASIP Journal on Embedded Systems, special issue on "Design and Architectures for Signal Image Processing"*, vol. Volume 2008 (2008), pp. Article ID 930250, 2008.

[10] K.J. Batenburg and J. Sijbers, "Automatic multiple threshold scheme for segmentation of tomograms," 2007, vol. 6512.

[11] Peter M. Joseph, "An improved algorithm for reprojecting rays through pixel images," vol. 1, no. 3, pp. 192–196, Nov. 1982.

[12] R.L. Sidon, "Fast calculation of the exact radiological path for a three-dimensional CT array," *Med. Phys.*, vol. 12, no. 2, pp. 252–255, 1985.

[13] G.L. Zeng and G.T. Gullberg, "Unmatched projector/backprojector pairs in an iterative reconstruction algorithm," *Medical Imaging, IEEE Transactions on*, vol. 19, no. 5, pp. 548–555, May 2000.

[14] O. Gerbaux, S. Crosnier, and M. Dubruel, "Experimental and numerical predictions of pressure drops for gas flow through isotropic metallic foams," in *France-Deutschland Fuel Cell Conference, Belfort*, 2004.

[15] M.L. Turner, C.H. Arns L. Knfing, A. Sakellariou, T.J. Senden, A.P. Sheppard, R.M. Sok, A. Limaye, W.V. Pinczewski, and M.A. Knackstedt, "Three-dimensional imaging of multiphase flow in porous media," *Physica A*, vol. 339, pp. 166–172, 2004.

[16] S. Bekri and P.M. Adler, "Dispersion in multiphase flow through porous media," *International Journal of Multiphase Flow*, vol. 28, pp. 665–697, 2002.

[17] O. Brunke, A. Hamann, S.J. Cox, and S. Odenbach, "Experimental and numerical analysis of the drainage of aluminium foams," *Journal of Physics: Condensed Matter*, vol. 17, pp. 6353–6362, 2005.