Universidad Carlos III de Madrid
e-Archivo

Institutional Repository

This document is published in:

*Proceedings of the 2012 10th IEEE International Symposium on Parallel and Distributed Processing with Applications: 10-13 July 2012. Madrid, Spain* (2012). IEEE, 867-868.
DOI: http://dx.doi.org/10.1109/ISPA.2012.138

© 2012 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

# Exploiting parallelism in a X-ray tomography reconstruction algorithm on hybrid multi-GPU and multi-core platforms

Ernesto Liria, Daniel Higuero - *University Carlos III, Madrid, Spain*
Monica Abella, Claudia de Molina, Manuel Desco - *Hospital General Universitario Gregorio Marañón, Madrid, Spain.*

***Abstract*—Most small-animal X-ray computed tomography (CT) scanners are based on cone-beam geometry with a flat-panel detector orbiting in a circular trajectory. Image reconstruction in these systems is usually performed by approximate methods based on the algorithm proposed by Feldkamp et al. Currently there are a strong need to speed-up the reconstruction of X-Ray CT data in order to extend its clinical applications. We present an efficient modular implementation of an FDK-based reconstruction algorithm that takes advantage of the parallel computing capabilities and the efficient bilinear interpolation provided by general purpose graphic processing units (GPGPU). The proposed implementation of the algorithm is evaluated for a high-resolution micro-CT and achieves a speed-up of 46, while preserving the reconstructed image quality.**

## I. Introduction

Many small animal X-ray computed tomography (CT) scanners are based on cone-beam geometry with a flat-panel detector orbiting in a circular trajectory [2]. This configuration presents advantages over other alternatives used in clinical and preclinical applications: reduction of acquisition time, large axial field of view (FOV) without geometrical distortions, and optimization of radiated dose. Despite the existence of a remarkable progress in statistical reconstruction algorithms, approximate methods based on the algorithm proposed by Feldkamp et al. [3] (namely FDK) are still widely used for solving the 3D reconstruction task because of its straightforward implementation and computational efficiency [2]. This algorithm is the extension of the filtered backprojection for cone beam geometry correction factors, which incorporates the length of the rays.

With the evolution of the technology, the acquisition time has been reduced. On the other hand, the evolution of the detector panels has resulted in an increase of detector elements density, which produces a higher amount of data to process [6]. Together with this increase of data, there is a need of faster reconstructions to address the newest uses of CT: planning and monitoring in radiotherapy, image assisted surgery, and other clinical modalities required the real time imaging [5]. On the other hand, the recent advances in algorithms have not been exploited yet at the full potential in high performance implementations, which represents a barrier for extending the use of this technology [5]. All this motivates the need to look for optimizations that can handle the increasing complexity and demand of the reconstruction task.

One possibility to speed up the reconstruction is to use alternative algorithms. These algorithms could be classified in three groups. The first ones are based on regrinding the projection data in the Fourier domain into a Cartesian grid in order to be able to use FFT directly. The second group is based on accelerating the back projection step by a recursive process of partial sums and treating all projections simultaneously. Finally, the third group is based on dividing the image in smaller parts (in space or Fourier domain). The algorithms from the last group have been reported to reach a speed-up of 40, although the quality of the image is degraded and they lack generality due to their dependency on the image properties.

Another strategy is the speed up of FDK using parallel computing techniques and architectures. In this direction there are many approaches, some of which are rigid and costly, as the use of application-specific integrated circuit (ASIC) or FPGA devices. Currently, one promising alternative is exploiting the parallelism inherent to the General Purpose Processor Unit (GPGPU). This paper presents an hybrid multi-core/multi-GPU modular implementation of the FDK algorithm based on a C implementation of Mangoose [1]. The implementation takes advantage of parallel computing capabilities and the efficient bilinear interpolation provided by GPU, in order to speed-up the two main stages of the algorithm implementation.

## II. Hybrid multi-GPU and multi-core implementation

Mangoose implementation consists of two main stages: filtering and backprojection. Both stages are highly parallelizable due to the lack of data dependencies and high number of data parallel operations. However, the main challenge is to highly utilize the available parallelism at both CPU and GPU levels. Our approach is based on hierarchical decomposition and dynamic scheduling. The hierarchical decomposition consists of a coarse-grain OpenMP implementation and fine-grain CUDA-based modules.

The OpenMP implementation leverages multi-core parallelism in order to support a variable number of heterogenous GPUs. Multi-core processors are exploited through coarse-grained data parallelism by assigning different projections to different threads dynamically. We employ dynamic scheduling in order to be able to map the parallel application on multi-GPU heterogeneous systems. This approach is motivated by

the fact that, as shown in Figure 2, the application shows a high variability of performance depending on the GPU architecture.

The dynamic scheduler of OpenMP assigns a set of projections in order to be processed consecutively in the filtering and backprojection stages. The filtering stage is implemented using CUFFT, the library of Fast Fourier Transforms (FFT) included in the development tools of version 4.0 of CUDA.

The strategy for backprojection relies on maximazing the utilization of the GPU global memory by storing there the highest fitting part of volume, while loading a set number of projections at a time into texture memory (the number of simultaneous projection depends on the current texture memory size available in the GPU). This approach significantly reduces processing time and increases the data locality in the texture memory.

## III. Experimental results

Data were acquired with the CT subsystem of an ARGUS PET/CT, based on cone-beam geometry and circular path. In order to provide standard timings, we used two standard resolution studies (pixel size of 0.2 mm), a one-bed study with 360 projections of 512x512 pixels and a two-bed study with 360 projections of 526x526 pixels (beds are units of reconstruction that are to be merged into the final volume). We also used a high resolution study (pixel size of 0.05 mm), with 360 projections of 2048x2048 pixels to evaluate the effect of handling big volumes on the performance. Reconstructed volumes had a resolution of 512x512x512 pixels and 2048x2048x2048, respectively. The computer system used in the evaluation is equipped with two Intel Xeon E5640 (2.67 GHz quad-core processors), 64 GBytes of RAM, and two NVIDIA Tesla C2050 and two ASUS GTX 470.

Figure 1 plots the times obtained for filtering and backprojection stages and the total processing time. Results suggest that for small volumes the best result is achieved with 2 GPUs. This is mainly due to the additional cost of transferring data between the devices and the CPU. However, for large volumes, increasing the degree of parallelism significantly reduce the execution time.

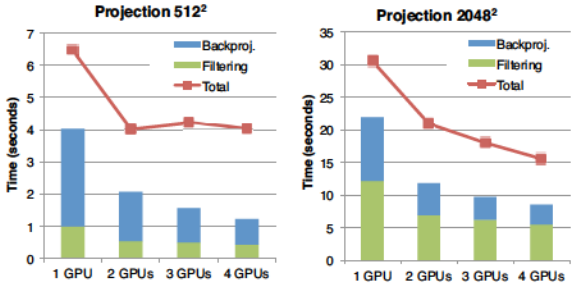


Fig. 1. Mangoose time breakdown for 1 to 4 GPUs and projection sizes of 512 and 2048 pixels.

Finally, Figure 2 shows the results of four projection sizes over three different GPU devices. As shown in the figure, the choice of the device brand is a key factor in reducing computation times.

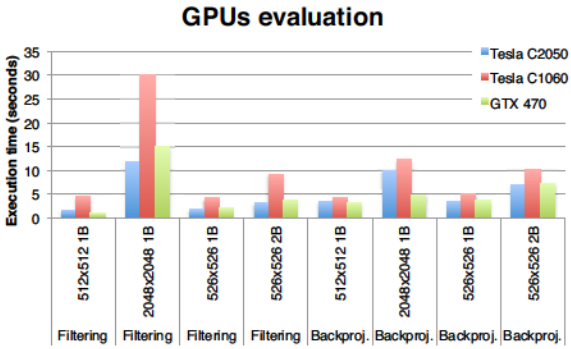


Fig. 2. Execution times for different GPU NVidia models and four sizes of projection with one bed (1B) and two beds (2B) .

## IV. Discussion and conclusion

In this work, we have presented an efficient modular implementation of a FDK-based algorithm for cone-beam CT. The main disadvantage of having different modules for filtering and backprojection is the inability to exploit synergies between the different stages, leading to a significant increase in data transfer between CPU and GPU. However, the modularity approach allows an efficient replacement algorithms implementation, facilitating the adaptability of the proposed solution to new architectures and incoming devices. Results show a speed-up of 25 and 46 for filtering and backprojection stages respectively. Although a direct comparison between different implementations from the literature is difficult due mainly to differences in the employed hardware, we found that the proposed implementation achieves an improvement over recently published works by a factor of 4 [1], [4].

## Acknowledgements

This work has been partially funded by AMIT Project CDTI CENIT, TEC2007-64731, TEC2008-06715-C02-01, RD07/0014/2009, TRA2009 0175, RECAVA-RETIC, RD09/0077/00087 (Ministerio de Ciencia e Innovacion), ARTEMIS S2009/DPI-1802 (Comunidad de Madrid), and TIN2010-16497 (Ministerio de Ciencia e Innovacion).

## References

[1] M. Abella, J. Vaquero, A. Sisniega, J. Pascau, A. Udias, V. Garcia, I. Vidal, and M. Desco. Software Architecture for Multi-Bed FDK-based Reconstruction in X-ray CT Scanners. *Computer methods and programs in biomedicine, in press*, 2011.

[2] C. T. Badea, M. Drangova, D. W. Holdsworth, and G. A. Johnson. In vivo small-animal imaging using micro-CT and digital subtraction angiography. *Physics in Medicine and Biology*, 53(19):R319, 2008.

[3] L. A. Feldkamp, L. C. Davis, and J. W. Kress. Practical cone-beam algorithm. *J. Opt. Soc. Am. A*, 1(6):612–619, Jun 1984.

[4] B. Wang, L. Zhu, K. Jia, and J. Zheng. Accelerated cone beam CT reconstruction based on OpenCL. In *2010 International Conference on Image Analysis and Signal Processing (IASP)*, pages 291 –295, april 2010.

[5] F. Xu and K. Mueller. Real-time 3D computed tomographic reconstruction using commodity graphics hardware. *Physics in Medicine and Biology*, 52(12):3405, 2007.

[6] X. Zhao, J.-J. Hu, and P. Zhang. GPU-based 3D cone-beam CT image reconstruction for large data volume. *Journal of Biomedical Imaging*, 2009:8:1–8:8, January 2009.