Fast Parallel GPU Implementation for Clinical Helical CT using Branchless DD

Ayan Mitra[1], Soysal Degirmenci[1], David G. Politte[2], Joseph A. O'Sullivan[1]
[1]Department of Electrical and Systems Engineering, Washington University, Saint Louis, MO, USA
[2]Mallinckrodt Institute of Radiology, Washington University School of Medicine, Saint Louis, MO, USA

Abstract

Rapid improvement of detector technology has led to an exponential increase in the number of measurements in many applications, including X-ray CT. As a result, clinical use of iterative reconstruction algorithms is rejected due to excessive computational time. However, these algorithms have the potential to reduce patient dosage. In an attempt to meet the increasing demand of computational performance, there is an overwhelming trend to shift towards multi-threaded CPU and GPU implementations. Due to their inherently parallel architecture, GPUs can provide significant performance improvement for algorithms with a highly pipelined architecture. In this paper we present an implementation of the Branchless Distance Driven (DD) Projection and Back projection methods using multiple CPU threads to launch multiple concurrent kernels on GPUs. Preliminary results showed that this implementation decreased reconstruction times approximately 5x for Back projection and 2x for Forward projection by using 3 NVIDIA GE Force Titan X GPUs in parallel compared to its OpenMP CPU implementation using 16 threads in parallel.

Introduction

Graphical Processing Units (GPUs) over the years have provided quite an impressive improvement on the computational cost and speed of iterative image reconstruction. Current GPUs also provide much global memory storage, which is ideal for fitting all of the data and the image volume in the GPU itself during kernel execution, which eliminates the high latency penalty for accessing external memory. The main aim of this paper is to provide a parallel implementation of the Branchless Distance Driven algorithm proposed by Basu et. al[1]. The branchless approach employs a simple factorization that can be computed in multiple passes without any branch prediction, which makes it ideal for implementation on a single thread of the GPU. However, the local memory constraint of a single GPU thread acts as a bottleneck to further improvement on time performance. The geometry of data acquisition in helical, multi-detector-row CT is shown below.

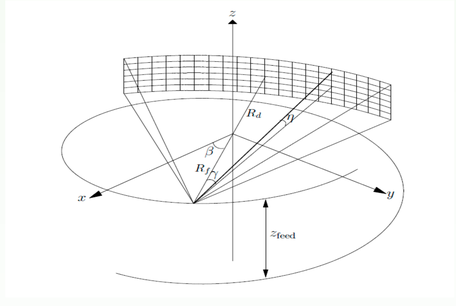


Figure 1. The Multi-Slice Helical CT geometry used in this work

For our specific reconstruction we used helical CT data for which we have quarter rotation symmetry mentioned in [3] which significantly reduces the computational burden.

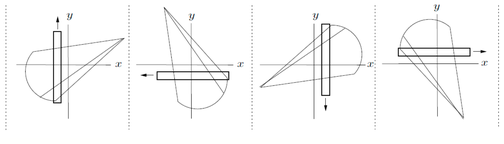


Figure 2. Transverse view of quarter rotation symmetry

Algorithms

The motivation for using branchless distance driven kernels and its full geometric description for 3D can be found in [2]. The system matrix is not explicitly calculated. Rather, it's implicitly taken into account during each computation of a forward or back projection. The core calculation of the algorithm is done at the slab level for the image being projected for a particular view.

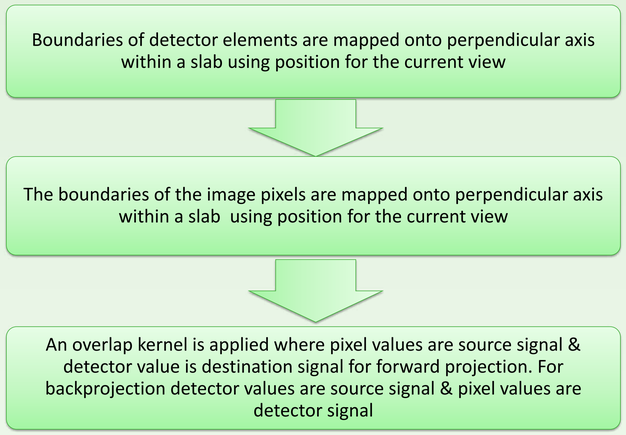


Fig.3. Basic outline of the Branchless DD kernels

A basic sequence of steps used for overlap kernel computation in Branchless DD projection & backprojection is shown below:

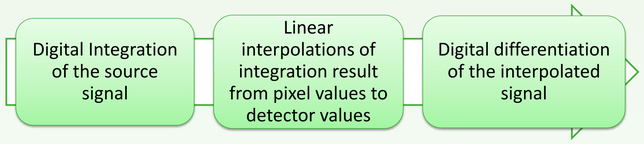


Fig. 4. Block diagram of the overlap kernel for forward projection

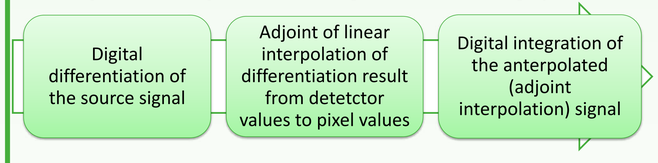


Fig. 5. Block diagram of the overlap kernel for back-projection

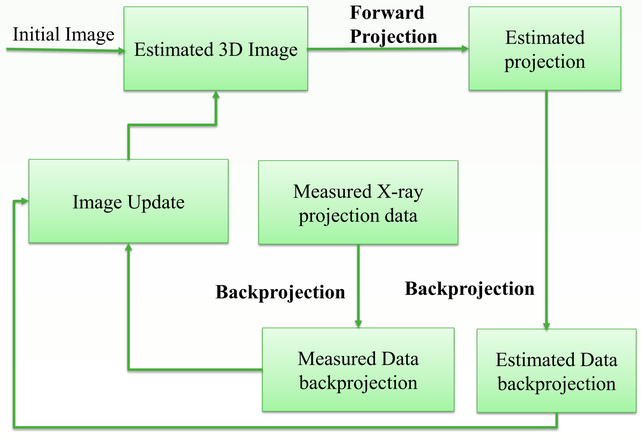


Fig. 6. Basic alternating minimization algorithm using penalized log-likelihood

Algorithm implementation on GPU

The main motivation for GPU implementation of the branchless DD is that the overlap kernel calculation for each slab within a particular view is completely independent of one another due to branchless property of the algorithm which can be run in parallel on a single thread on GPU.

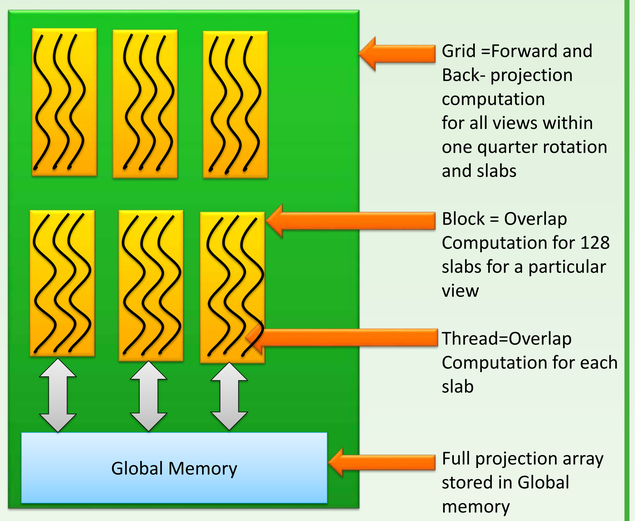


Fig. 7. A basic outline of the 3D implementation on a single GPU

Implementation on Multiple GPUs

The fact that the system matrix is symmetric for each quarter rotation makes it quite natural to implement parallelism at the granularity of a quarter rotation of data. Each GPU is assigned a contiguous group of projections whose cardinality is a multiple of the number of views in a quarter rotation. This design allows for theoretically perfect load balancing (in the absence of memory-related latencies) during forward and back projection.

Forward projection is straightforward in terms of global memory access, since each device stores values in separate portions of the projection data array, and access to the images is read-only. However, if we were to perform back projection directly into the full-size images, we would have serious memory contention issues since multiple devices would be writing to the same array elements simultaneously. Instead, each device performs back projection to its own private image arrays (of reduced size compared to the full size arrays). This eliminates any need for synchronization during the back projection of a device's set of views. Once each device is done backprojecting its set of views, the partial accumulation image arrays are summed into the full-size accumulation image arrays.

Results

To compare time performance, we start with an Intel Core i7 5960x with 8 cores, 8 threads, clocked at 3 GHz, with 20 MB cache and 64 GB of memory. We used hyper-threading to utilize 16 threads. We also used raw data from Siemens Sensation 16 DICOM data obtained from Saint Louis Children's Hospital. The parameters of the measured data and reconstructed images are tabulated as follows:

No. of views	13920
No. of detector channels	672
No. of detector rows	16
No. of pixels per slice	512x512
No. of slices	164
Image voxel dimensions (in mm)	1x1x1

Table 1. Parameters of measured data and reconstructed image

Results

The time performance for one forward projection and backprojection with different multi-threaded CPU and GPU configuration is shown below:

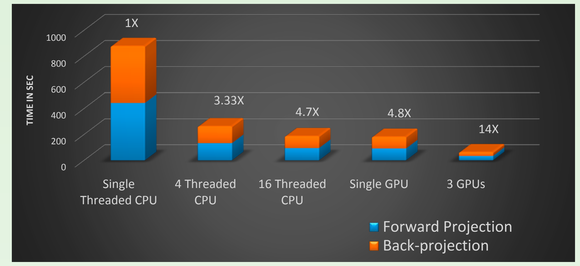


Fig. 9. Time performance improvement using different CPU & GPU configurations for single Branchless DD forward & back projection

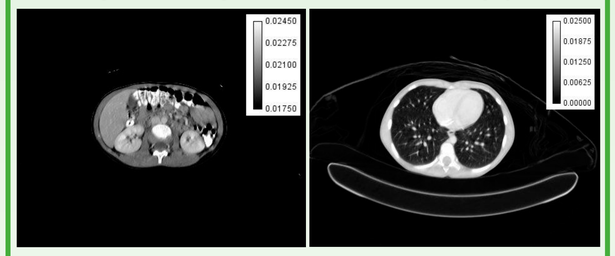
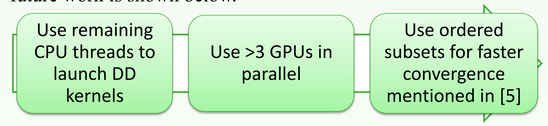


Fig. 10. Axial-Slices of 3D AM Reconstruction of abdomen at left and lung at right after 400 iterations

Conclusions and Future Work

We observed that our approach to use multiple GPUs to reconstruct images gives us better performance in computational cost compared to our best available CPU configuration. We can also observe that computational time show linear decrease in time performance with addition of more GPUs. We can expect to get better computational efficiency with having higher number of GPUs which open the door to exciting new possibilities in clinical settings. The direction of our future work is shown below.



References

[1] S. Basu and B. DeMan, "Branchless distance driven projection and backprojection," in Proc. SPIE: Electronic Imaging, vol. 6065, 2006.

[2] B. De Man and S. Basu, "Distance-driven projection and backprojection in three dimensions," Phys. Med. Biol., vol. 49, pp. 2463–2475, 2004.

[3] Keesing, Daniel, "Development and Implementation of Fully 3D Statistical Image Reconstruction Algorithms for Helical CT and Half-Ring PET Insert System" (2009). *All Theses and Dissertations (ETDs)*. http://openscholarship.wustl.edu/etd/427

[4] Van-Giang Nguyen; Soo-Jin Lee, "Parallelizing a Matched Pair of Ray-Tracing Projector and Backprojector for Iterative Cone-Beam CT Reconstruction," in Nuclear Science, IEEE Transactions on , vol.62, no.1, pp.171-181, Feb. 2015

Acknowledgment

Authors would like to thank Daniel B. Keesing for letting us use the HECTARE software package for out multi-threaded CPU implementation.