Accelerating 3D CT Reconstruction Using GPUs

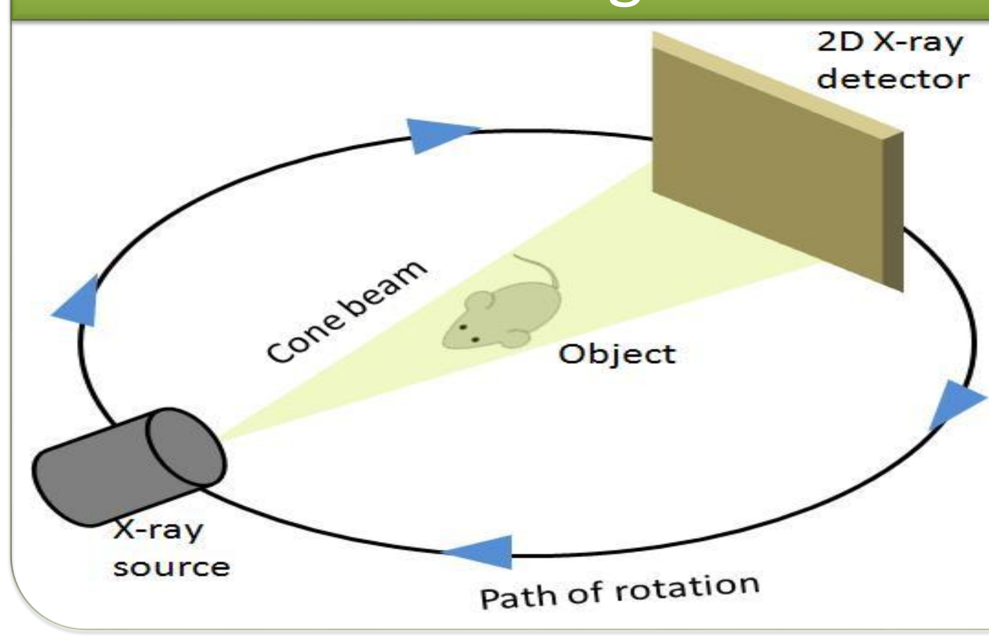
Saoni Mukherjee, Nicholas Moore, James Brock, Miriam Leeser

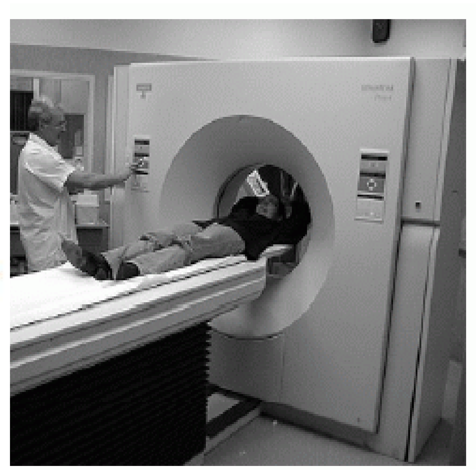
Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA

Abstract

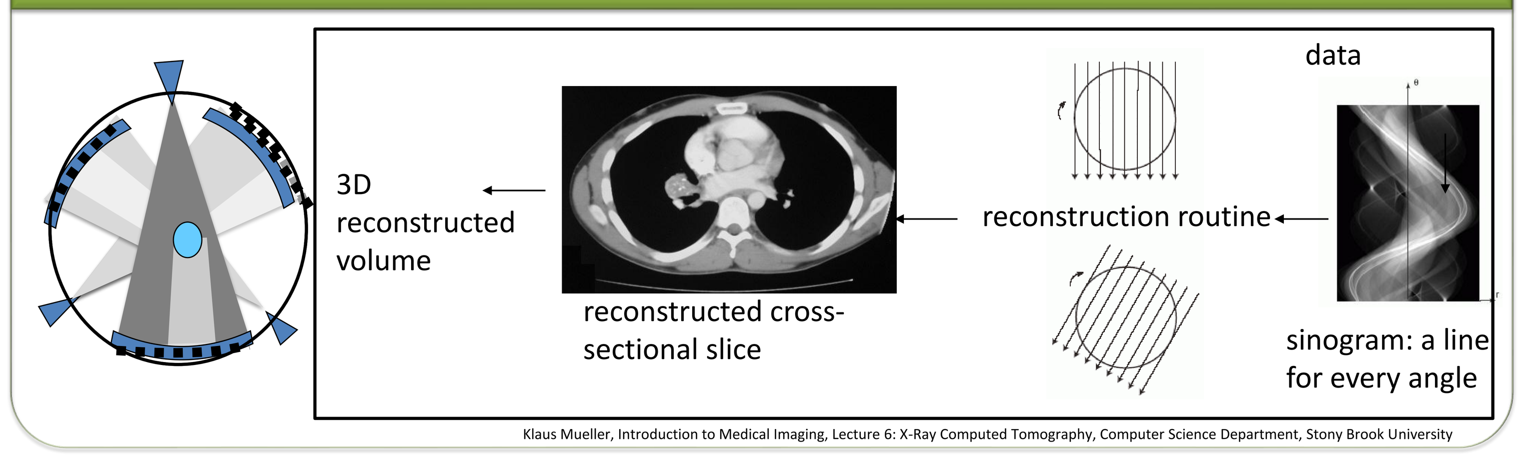
Biomedical image reconstruction applications with large datasets can benefit from acceleration. Graphic Processing Units(GPUs) are particularly useful in this context as they can produce high fidelity images rapidly. An image algorithm to reconstruct conebeam computed tomography(CT) using two dimensional projections is implemented using GPUs. The implementation takes slices of the target, weighs the projection data and then filters the weighted data to backproject the data and create the final three dimensional construction. This is implemented on two types of hardware: CPU and a heterogeneous system combining CPU and GPU. The CPU codes in C and MATLAB are compared with the heterogeneous versions written in CUDA-C and OpenCL. The relative performance is tested and evaluated on a mathematical phantom as well as on mouse data. Speedups of over thirty times using the GPU are seen for phantom data and close to fifty times for the larger mouse datasets.

What is CT Scanning?

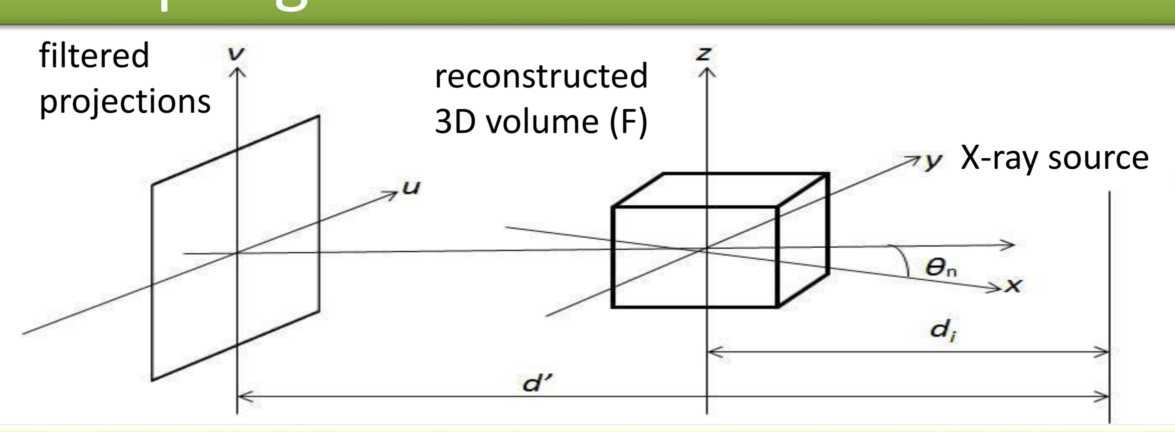




3D CT Reconstruction



Feldkamp Algorithm



Weighted Projection: Weighted and ramp filtered raw data produce filtered projections $Q_1, Q_2, ..., Q_K$, collected at an angle ϑn where $1 \le n \le K$. d_i = distance between the volume origin and the source. $F(x, y, z)$ = value of voxel (x, y, z) in volume F. Volume F in xyz space and Projections are in uv space.

Backprojection: The volume F is reconstructed using the following equations:

$$F(x,y,z) = \frac{1}{2\pi t}\sum_{i=1}^{t} W_2(x,y,i)\,Q_i\big(u(x,y,i), v(x,y,z,i)\big),$$

Co-ordinates

$$u(x,y,i) = \frac{d'(-x\sin\theta_i + y\cos\theta_i)}{d_i - x\cos\theta_i - y\sin\theta_i}$$

$$v(x,y,z,i) = \frac{d'z}{d_i - x\cos\theta_i - y\sin\theta_i}$$

Weight value,

$$W_2(x,y,i) = \frac{d_i}{d_i - x\cos\theta_i - y\sin\theta_i}$$

Advantage

i) Reduced X-ray exposure,

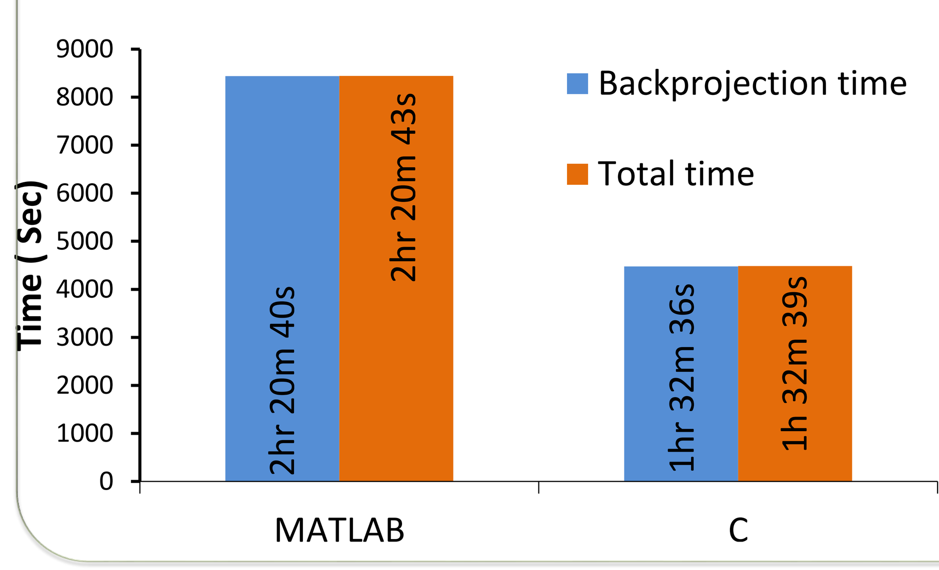
ii) Image accuracy - more accurate than MRI!

Disadvantage

The longer time it takes to reconstruct the volume!

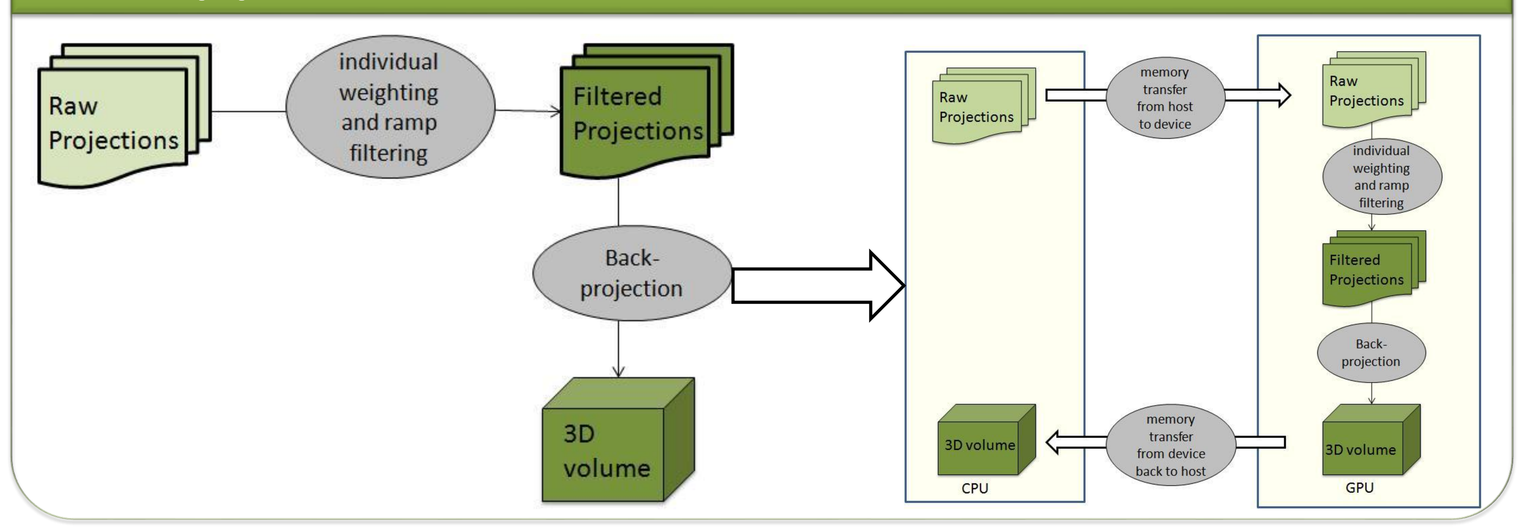
- interruption in treatment/ diagnosis.

Motivation

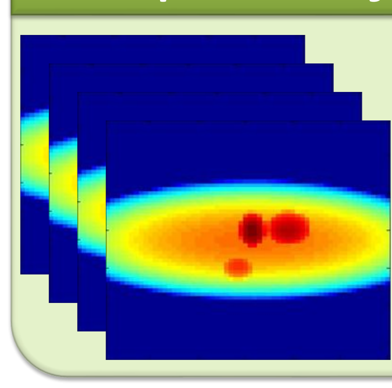


- Backprojection takes most of the time, but highly parallelizable.
- Different voxels are independent.
- Fessler's image reconstruction toolbox[6] implements Feldkamp CBCT in MATLAB. Widely used in academia.

Our approach



Sample Projections



Mathematical phantom

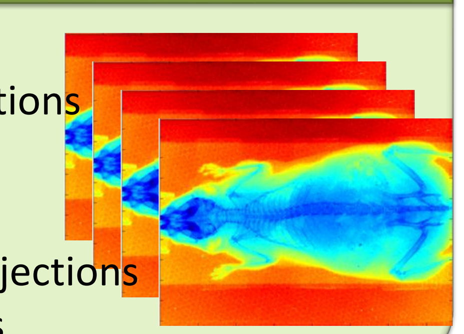
Input: 64 × 60 pixels with 72 projections

final volume: 64 × 60 × 50 voxels

Mouse scan

Input: 512 × 768 pixels with 361 projections

final volume: 512 × 512 × 768 voxels



Architectures and Languages used

Host	Device	Language
Intel Core i7 quad-core processor with @ 3.4 GHz		MATLAB MATLAB PCT
Intel Xeon W3580 quad-core processor @ 3.33 GHz	NVIDIA Tesla C2070	C C with OpenMP CUDA
Intel Xeon CPUs E5520 @ 2.27GHz	AMD Radeon HD5870	OpenCL

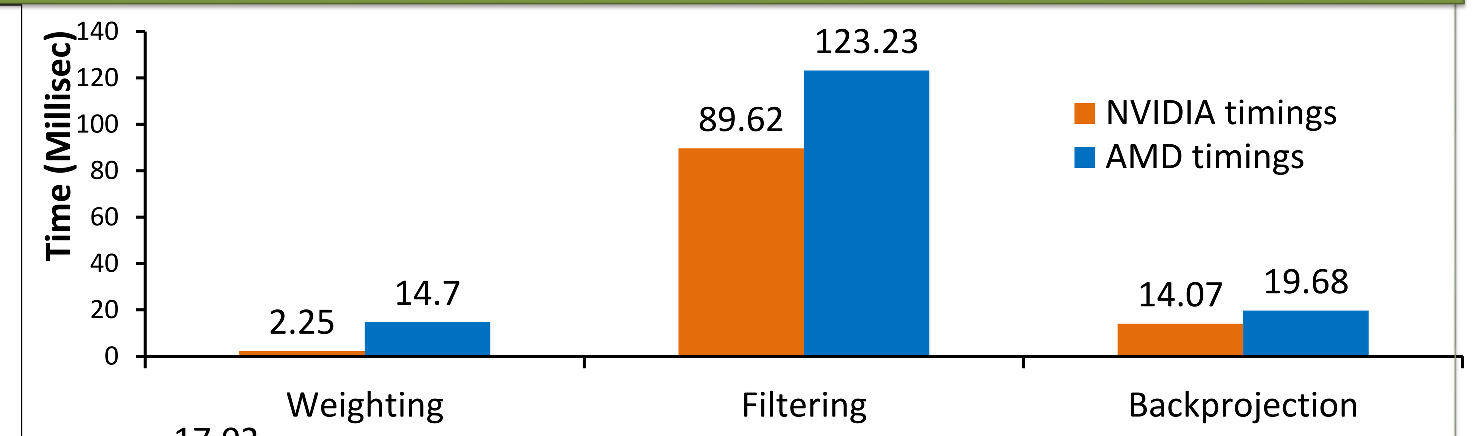
Acknowledgments

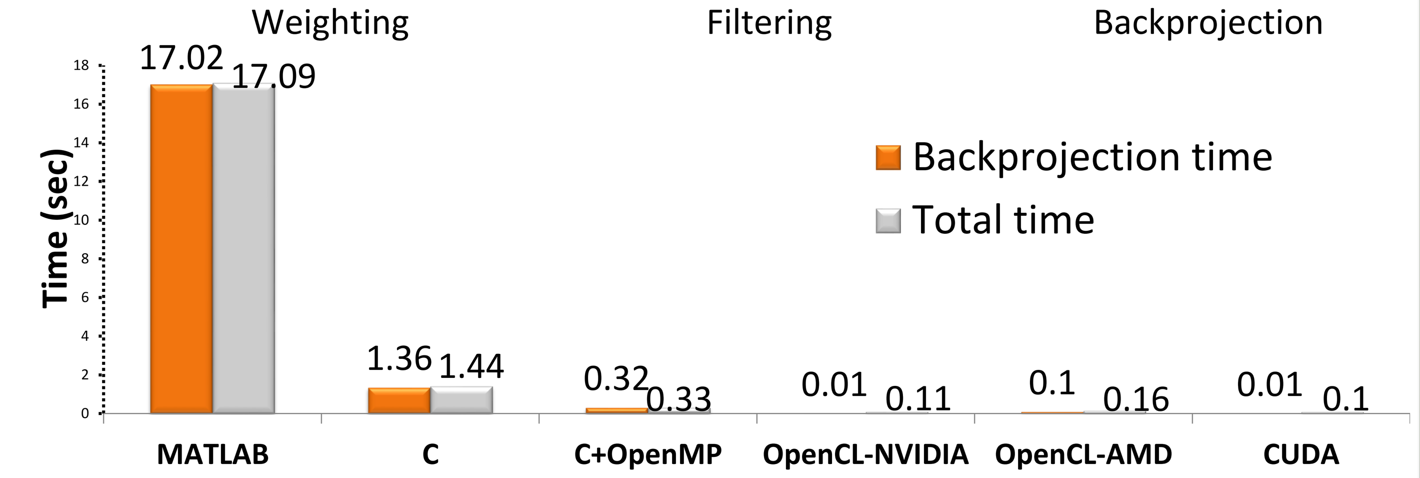
More information and software available:

http://www.coe.neu.edu/Research/rcl//projects/CBCT.php

Results

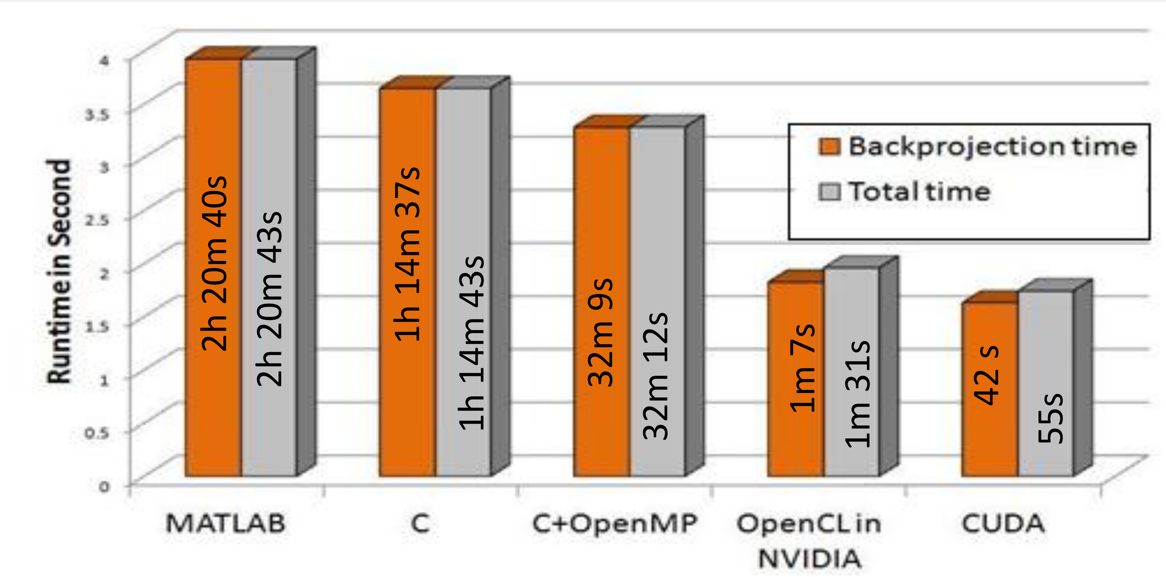
PHANTOM





Programming Paradigm	Speedup over single threaded MATLAB	Speedup over single threaded C	Speedup over multi-threaded C
C with OpenMP	50x	4x	-
OpenCL (NVIDIA)	**1700x**	**136x**	**32x**
OpenCL (AMD)	170x	13x	3x
CUDA	**1700x**	**136x**	**32x**

MOUSE SCAN



Programming Paradigm	Speedup over single threaded MATLAB	Speedup over multi-threaded MATLAB	Speedup over single threaded C	Speedup over multi-threaded C
MATLAB PCT	1.5x	-	-	-
C with OpenMP	4x	-	2x	-
OpenCL (NVIDIA)	125x	80x	70x	30x
CUDA	200x	130x	100x	45x

Future work

1) The next bottleneck- Weighted Filtering. Was *not* earlier! 2) More configurations to be tested with auto-tuning- number of kernels to be launched, number of threads. 3) Streaming for bigger datasets. 4) Overlapping computation and communication.

References

[1] S. Mukherjee, N. Moore, J. Brock, M. Leeser, CUDA and OpenCL Implementations of 3D CT Reconstruction for Biomedical Imaging, Proc. of IEEE High Performance Extreme Computing, (2012).

[2] L. A. Feldkamp, L. C. Davis, J. W. Kress, Practical cone-beam algorithm, J. Opt. Soc. Am., Volume 1(A), (1984).

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

[4] F. Ino, S. Yoshida, K. Hagihara, RGBA Packing for Fast Cone Beam Reconstruction on the GPU, Proc. of SPIE, Vol. 7258, (2009).

[5] NVIDIA corporation, NVIDIA CUDA C Programming Guide, CUDA Toolkit 4.1.

[6] Fessler's image reconstruction toolbox, http://www.eecs.umich.edu/~fessler/irt/fessler.tgz.