Faster 3D CT Reconstruction using CUDA and OpenCL

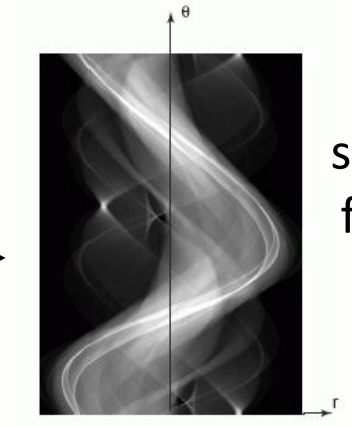
Saoni Mukherjee, Nicholas Moore, James Brock
and Miriam Leeser

April 24, 2012

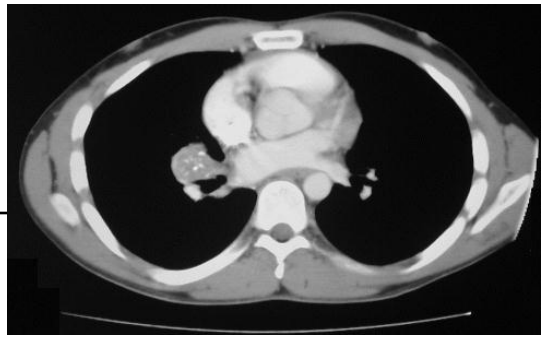
Northeastern

Introduction to 3D Computer Tomography Scan

data

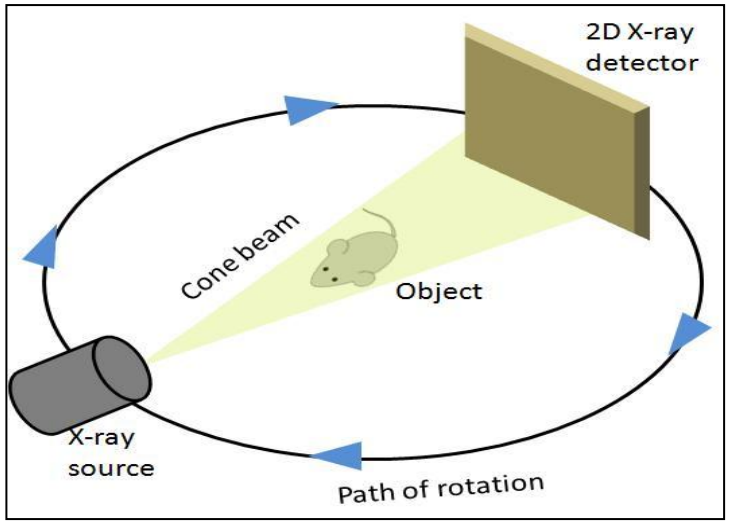


sinogram: a line for every angle



reconstructed cross-sectional slice

3D reconstructed volume



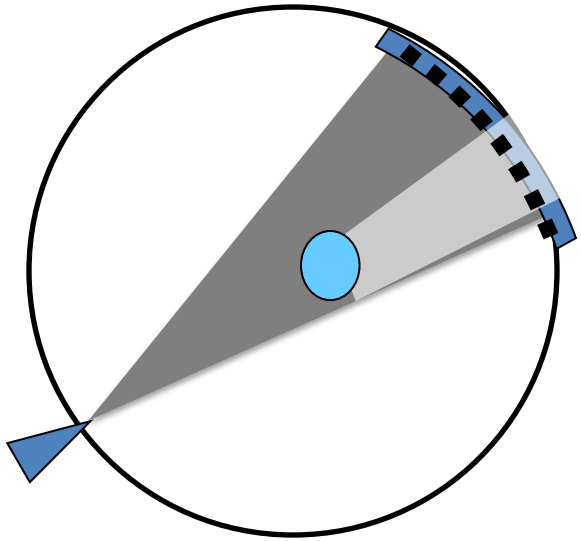
FDK Cone beam CT reconstruction

- Feldkamp, Davis and Kress (FDK)[1] developed in 1984.

- Most commercial CT scanners use FDK.

- The raw projections individually weighted and ramp filtered. Weighting includes cosine weighting and short-scan weighting.

- Reconstructions of filtered projections for the final volume.



[1] http://www.eecs.umich.edu/~fessler/irt/irt

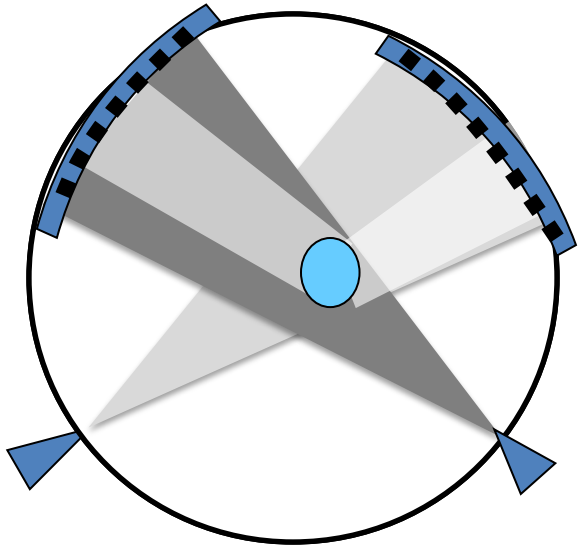
FDK Cone beam CT reconstruction

- Feldkamp, Davis and Kress (FDK)[1] developed in 1984.

- Most commercial CT scanners use FDK.

- The raw projections individually weighted and ramp filtered. Weighting includes cosine weighting and short-scan weighting.

- Reconstructions of filtered projections for the final volume.



[1] http://www.eecs.umich.edu/~fessler/irt/irt

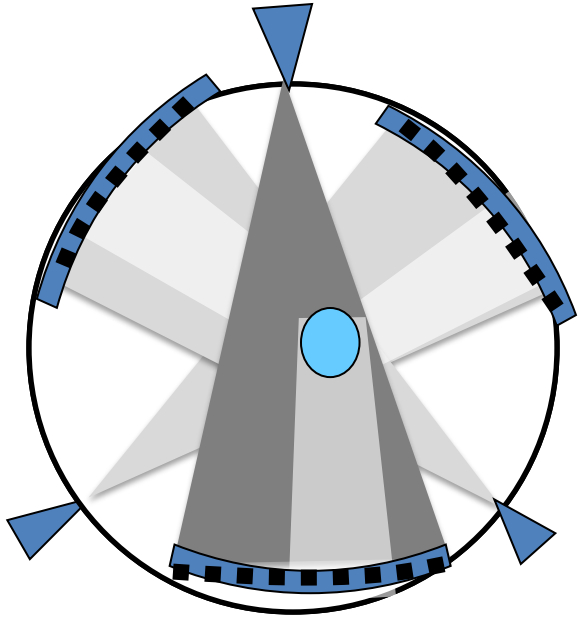
FDK Cone beam CT reconstruction

- Feldkamp, Davis and Kress (FDK)[1] developed in 1984.

- Most commercial CT scanners use FDK.

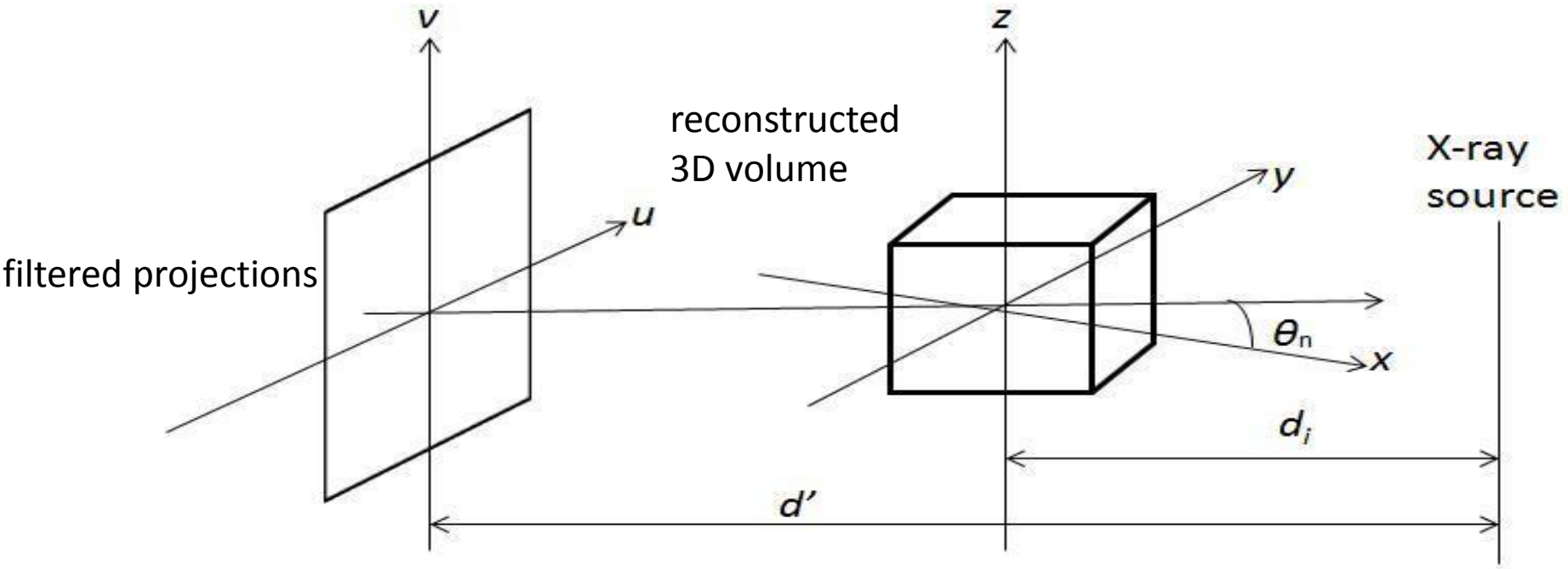
- The raw projections individually weighted and ramp filtered. Weighting includes cosine weighting and short-scan weighting.

- Reconstructions of filtered projections for the final volume.



[1] http://www.eecs.umich.edu/~fessler/irt/irt

Feldkamp CT reconstruction geometry- 1



Feldkamp CT reconstruction geometry- 2

Backprojection:

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

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,

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

What's the problem?

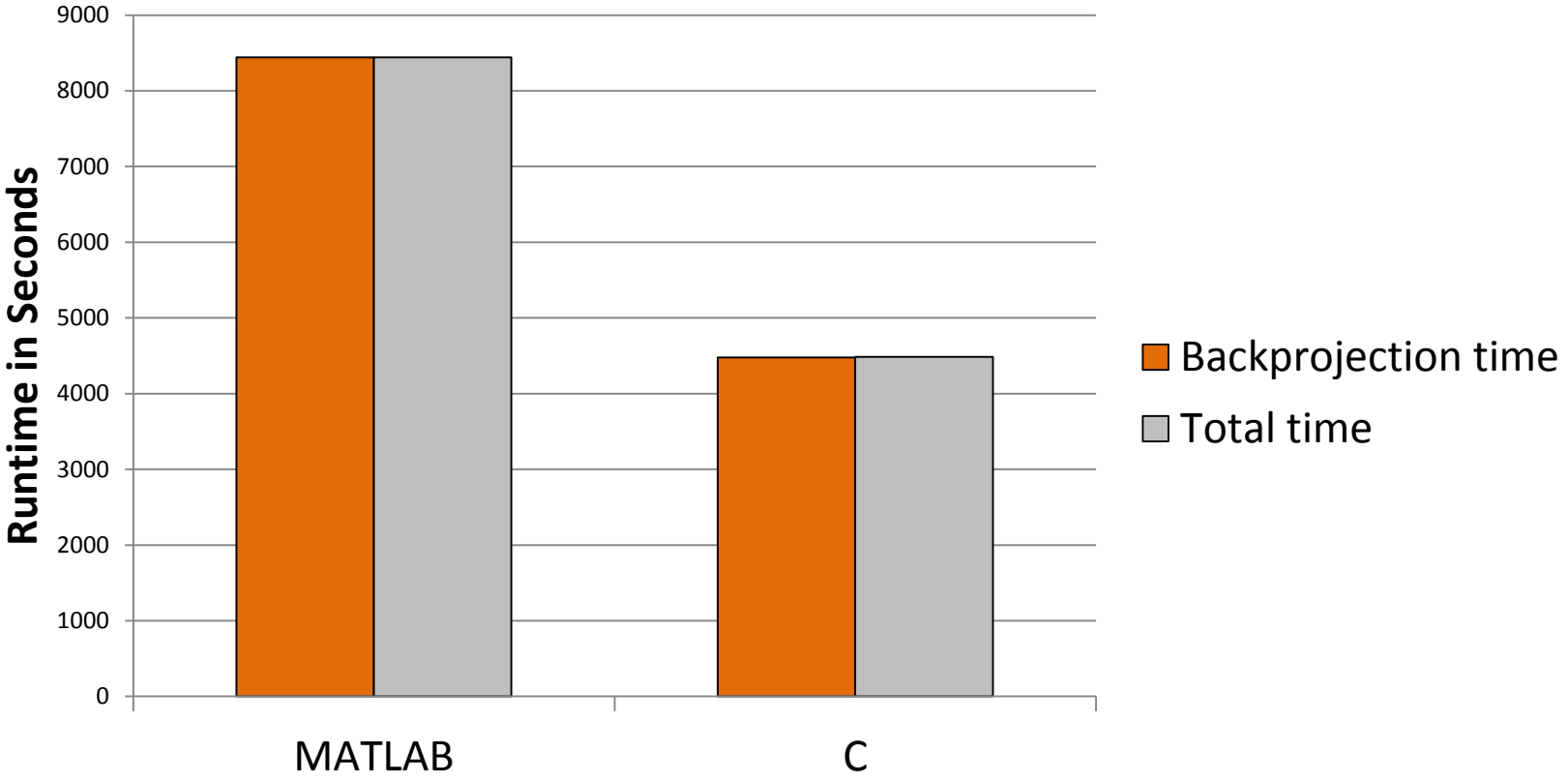
The long time it takes to reconstruct the volume!

- Interruption in treatment/diagnosis

- Capturing data takes ~9 seconds.

Philips Brilliance CT Scanner

Time spent in single-threaded code

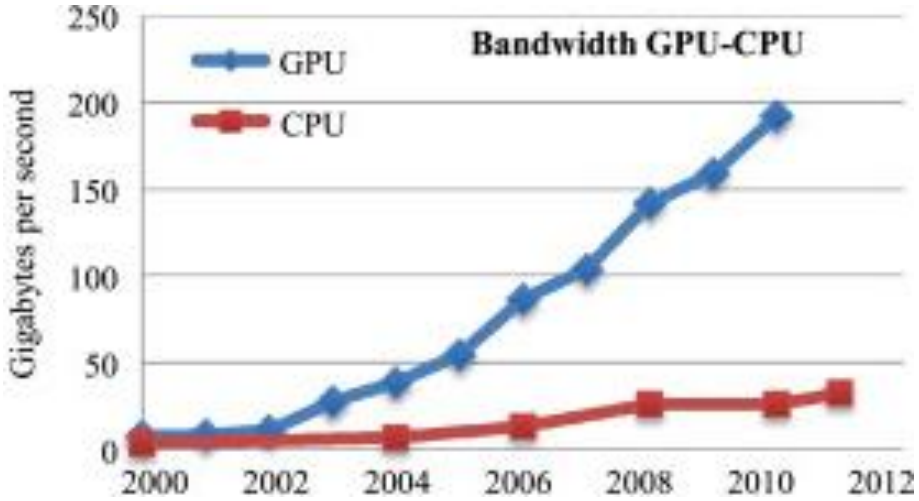


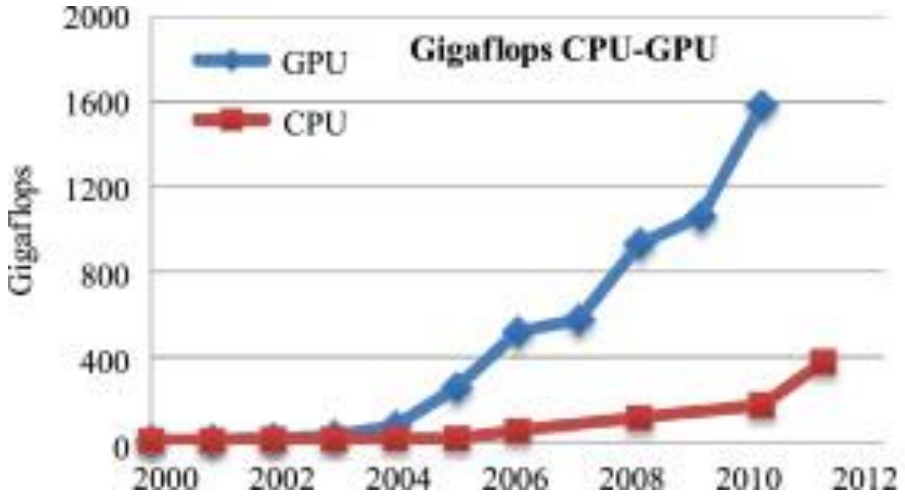
Programming paradigm	Time to run Backprojection	Total time
MATLAB	2h 20m 40s	2h 20m 43s
C	1h 32m 36s	1h 32m 39s

GPUs provides faster way to compute

GPU computing key ideas:

- Massively parallel
- Hundreds of cores
- Thousands of threads
- Cheap
- Highly available





Goal - GPU as an accelerator in CBCT

- Backprojection *most computationally intensive* part taking most of the time, but *highly parallelizable*.

- Independent different voxels to be *processed simultaneously*.

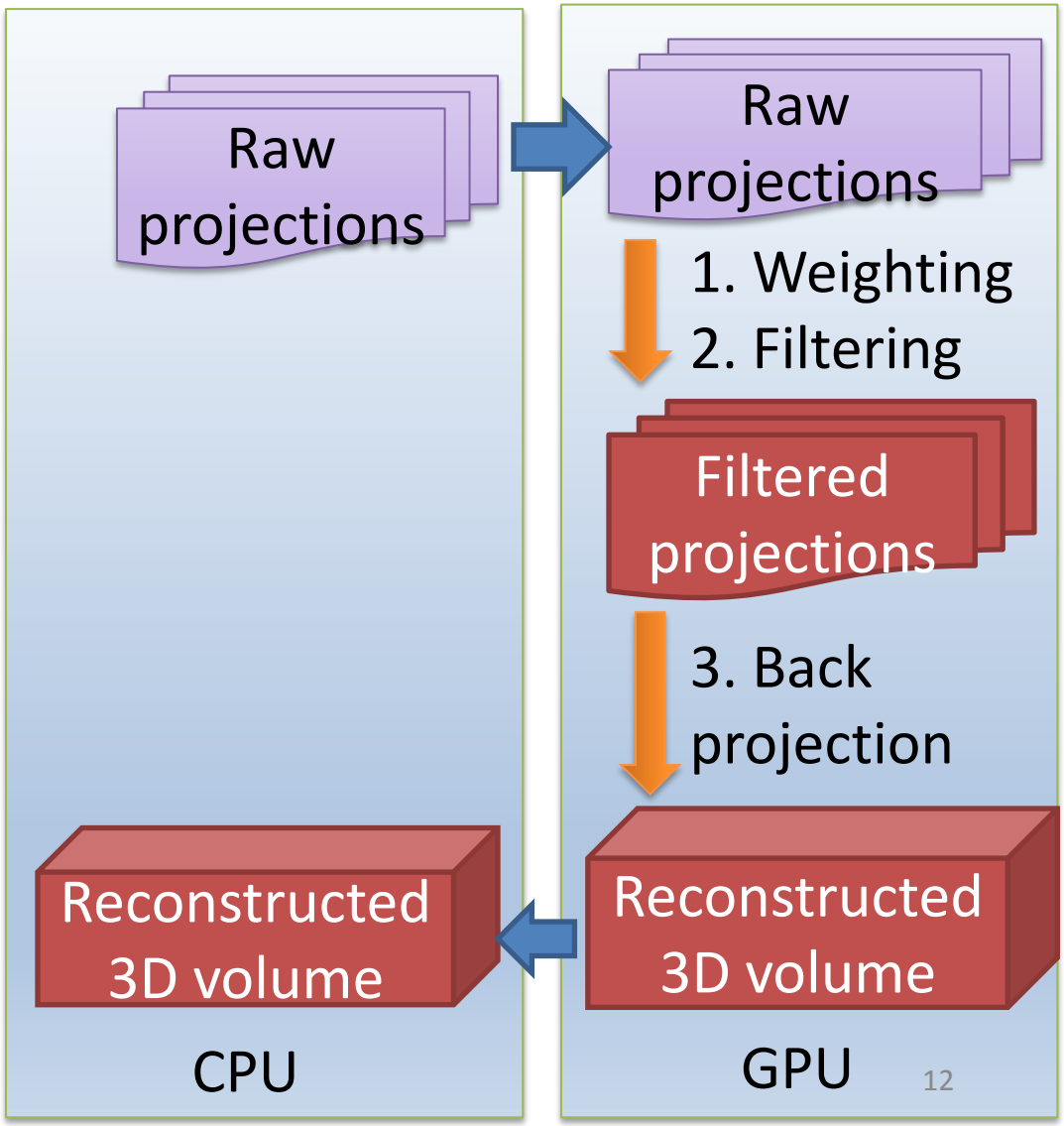
- *Fessler's image reconstruction toolbox*[1] an implementation of Feldkamp CBCT in MATLAB. Widely used in Academia.

- Our goal is to implement *faster Feldkamp CT.*

[1]: http://www.eecs.umich.edu/~fessler/irt/irt

GPU implementation of Feldkamp CBCT

- Processing divided into three steps: *weighting, filtering and backprojection*.
- Each step executed in *each kernel*.
- *Non-blocking kernel calls*, but executed in series.
- *Minimization of expensive memory transfers*



GPUs used to test the implementations

NVIDIA TESLA C2070

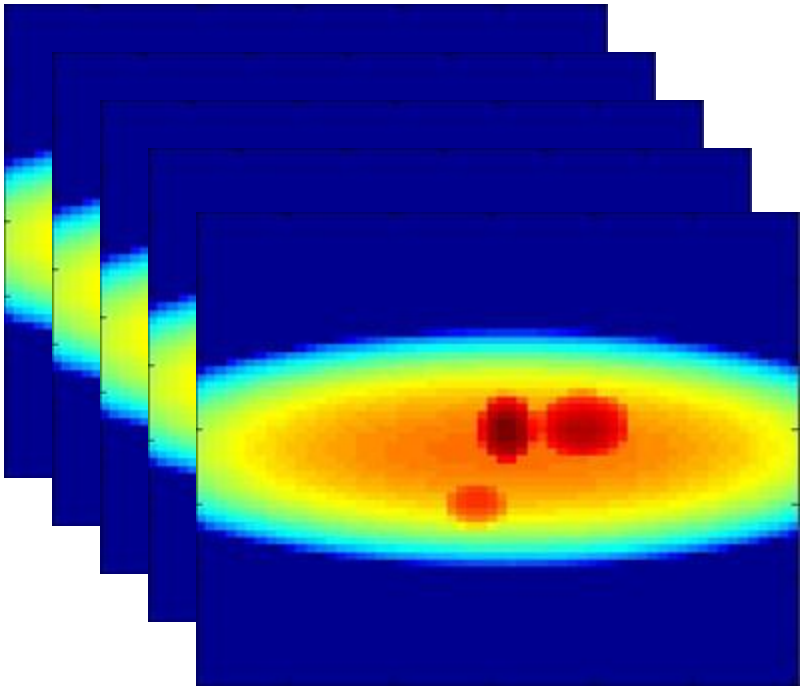
- Maximum 1536 resident threads in each multiprocessor
- 14 streaming multiprocessors
- Theoretical limit on the number of threads in flight at once is 21,504
- Memory size 6GB

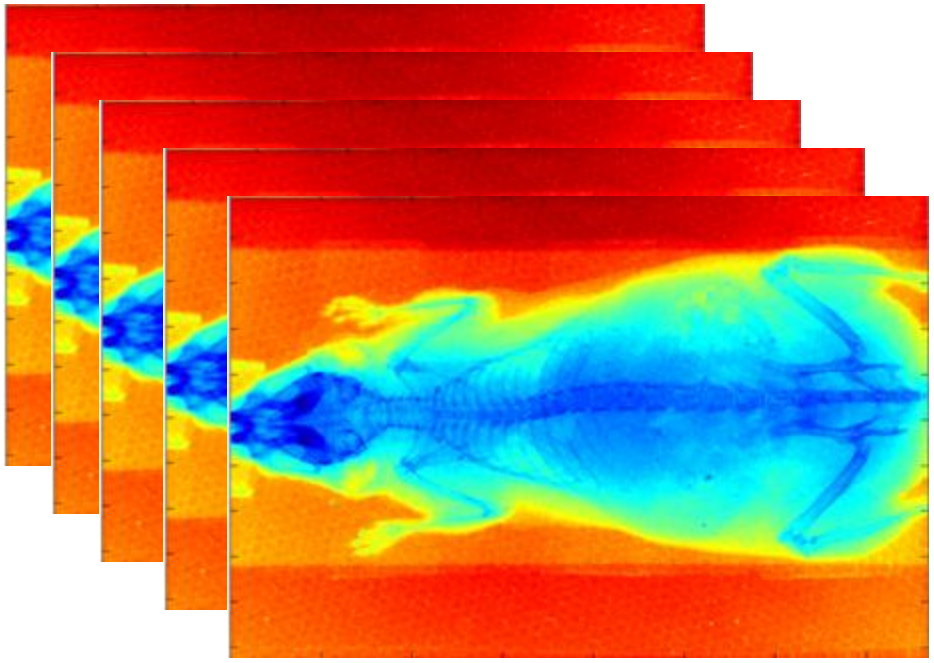
Similar generation

AMD Radeon HD 5870

- Can run up to 31,744 threads concurrently
- Memory size 1GB

Sample Projections





Mathematical phantom

Input: 64 × 60 pixels with 72 projections

final volume: 64 × 60 × 50 voxels

Size : 1MB + 1MB

Mouse scan

Input: 512 × 768 pixels with 361 projections

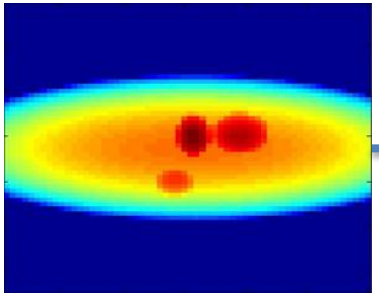
final volume: 512 × 512 × 768 voxels

Size : 542 MB + 768 MB **> 1GB**

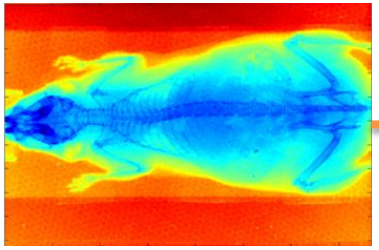
Architectures and Languages used

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

Roadmap for Results

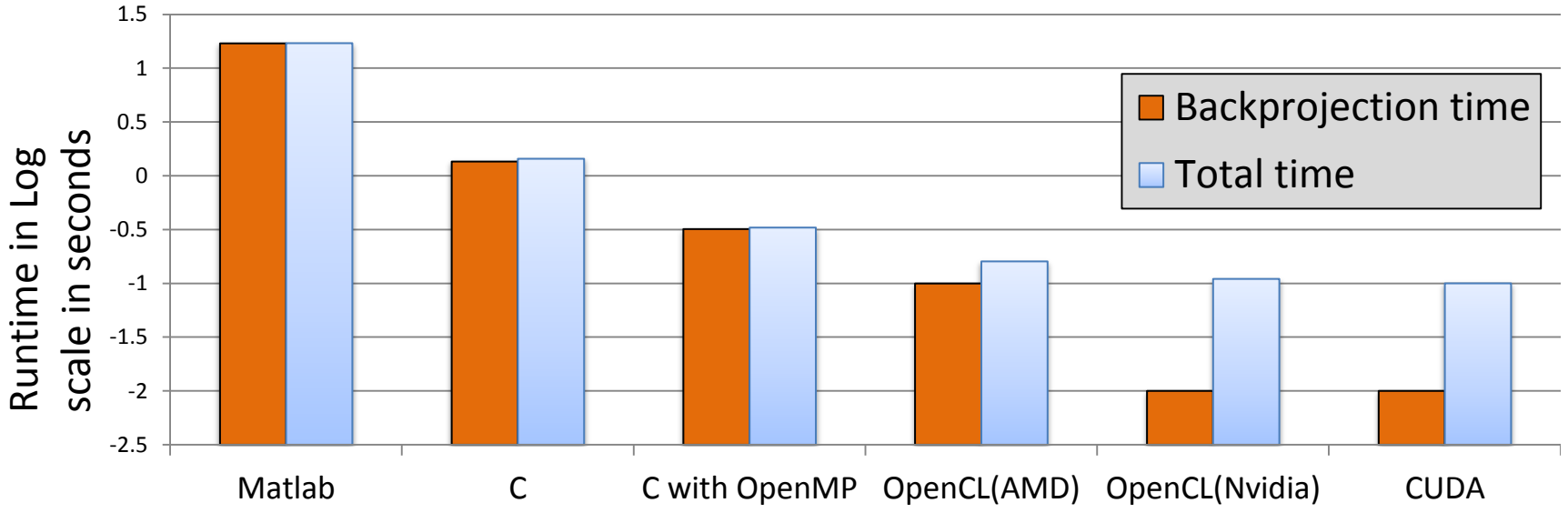


- Total runtime
- Speed up
- Runtime for each kernel: Nvidia vs. AMD



- Total runtime
- Speed up

Results on phantom data



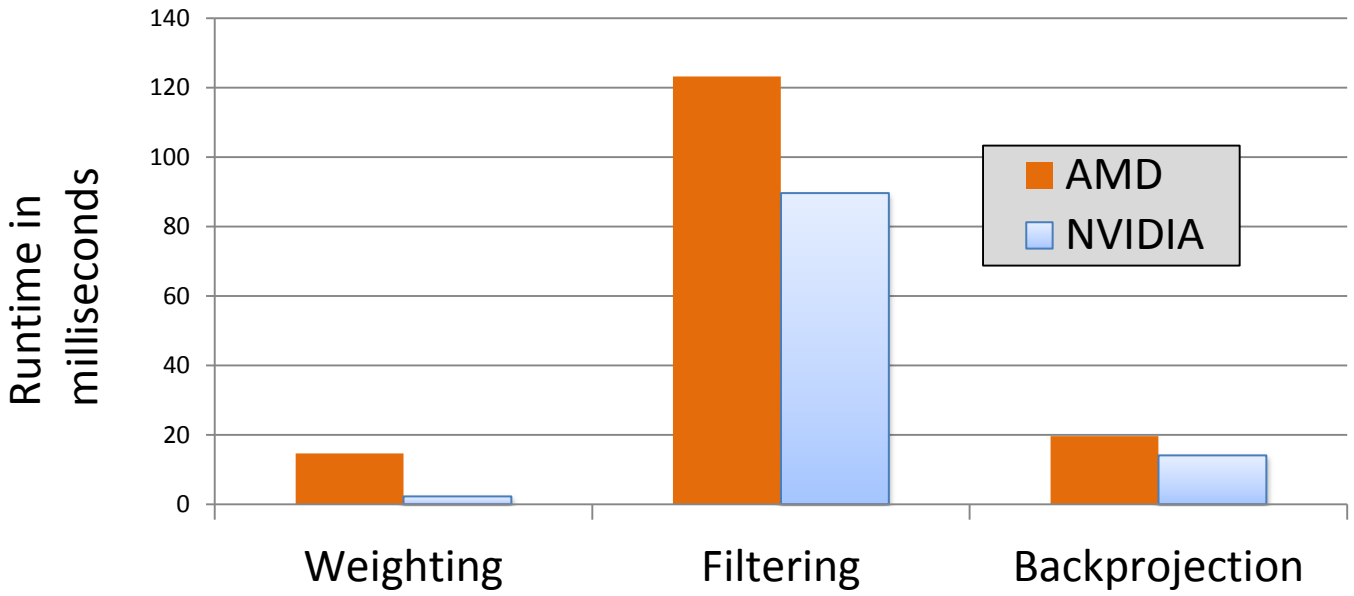
Programming paradigm	Time to run Backprojection (sec)	Total time (sec)
MATLAB	17.02	17.09
C	1.36	1.44
C with OpenMP (4 thrds)	0.32	0.33
OpenCL (AMD)	0.10	0.16
OpenCL (NVIDIA)	0.01	0.11
CUDA	0.01	0.10

Speedups for phantom data

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

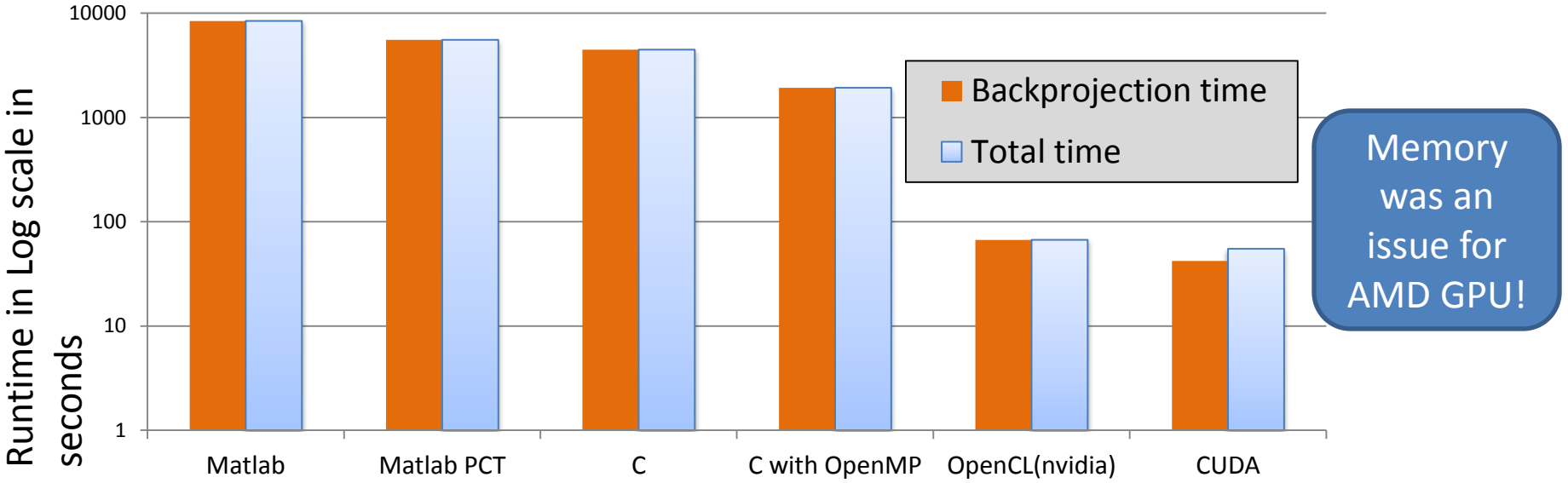
Comparisons are based on the time taken by Backprojection

Results – comparing NVIDIA vs. AMD



GPU	Kernel	Time (millisecond)	Total time (millisecond)
AMD	Weighting	14.70	
	Filtering	123.23	157.61
	Backprojection	19.68	
NVIDIA	Weighting	2.25	
	Filtering	89.62	105.94
	Backprojection	14.07	

Results on mouse scan data



Programming paradigm	Hardware	Time to run Backprojection (sec)	Total time (sec)
MATLAB	Intel Core i7	2h 20m 40s	2h 20m 43s
MATLAB PCT (8thrds)	Intel Core i7	1h 32m 36s	1h 32m 39s
C	Intel Xeon W3580	1h 14m 37s	1h 14m 43s
C with OpenMP (4thrds)	Intel Xeon W3580	32m 09s	32m 12s
OpenCL	NVIDIA Tesla 2070	01m 07s	01m 31s
CUDA	NVIDIA Tesla 2070	42s	55s

Speedups for mouse scan data

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

Comparisons are based on the time taken by Backprojection

Future Work

- Optimize other GPU kernels
- More configurations to be tested with auto-tuning
- Streaming for bigger datasets
- Overlapping computation and communication
- Improve performance on AMD device

Conclusions

- **Faster 3D cone beam reconstruction** using GPU.
- **Compatible with Fessler's** image reconstruction tool box.
- **Compared CUDA and OpenCL**, to **serial and multithreaded** C and MATLAB implementations.
  - Tested on two types of hardware
    - CUDA code takes 43 sec to backproject mouse scan.
      - ➢ **200x** faster than single-threaded MATLAB,
      - ➢ **100x** faster than single-threaded C,
      - ➢ **45x** faster than multi-threaded C with OpenMP.

Saoni Mukherjee, saoni@coe.neu.edu

Software can be downloaded from :
http://www.coe.neu.edu/Research/rcl/projects/CBCT.php

Acknowledgments

Award Number
EEC-0946463