Accelerated X-Ray Imaging: Real-time Multi-plane Image Reconstruction with CUDA

Prashanth B. Bhat, Vinay Kumar, Akshaya M., Devendra Shastry, Venkata Subbarao, Niranjan U.C.

Manipal Dot Net Pvt. Ltd., India
http://www.manipal.net
Manipal Dot Net

• Technology Outsourcing Firm in South India
• Founded in 2004 by Dr. Narasimha Bhat, PhD: UC Berkeley
• Software and Hardware Engineering Teams
• Areas of Expertise:
  • Embedded Systems Software
  • Digital Design
  • High Performance Computing
  • Video and Image Processing
Agenda

• Introduction to X-Ray Fluoroscopy
• SBDX: Scanning Beam Digital X-Ray
• Image Reconstruction: Problem Definition
• Computational Aspects
• Parallel Implementation with CUDA
• Performance Results
• Summary and Future Work
Introduction: X-Ray Fluoroscopy
X-Ray Fluoroscopy
X-Ray Fluoroscopy

• A Medical Imaging Technique
  • In use for over 40 years

• Useful during Cardiac Interventional Procedures
  • For catheter guidance

• Uses a Continuous X-Ray Beam

• Display Screen shows *Live Video*
X-Ray Fluoroscopy
X-Ray Fluoroscopy: Typical Architecture

- Single Point Source
- Large Area Detector

Concerns:
- Detector very Close to Patient
- Obstructs Access
- Scatter Degrades Images
- Radiation Hazard for Medical Personnel
X-Ray Fluoroscopy
SBDX: Scanning Beam Digital X-Ray
Scanning Beam Fluoroscopy

- 2D Array of X-Ray Sources
  - Collimators
  - Activated One at a Time
  - Raster Scan Fashion
- 2D Detector Array
  - Orthogonal Grid Topology
  - Readings Taken for Every Collimator
- Each Collimator Covers a Small Portion of the Imaging Area
Collimator Array
SBDX: Improved Image Quality

Point Source

Large Detector

Scatter Degrades Image

Scanning Beam

Small Detector

Scatter Free
SBDX: Scanning Beam Digital X-Ray

- Prototype System Designed at University of Wisconsin, School of Medicine

- References:
NovaRay Medical: ScanCath™
Problem Definition: Image Reconstruction
Input: Stream of Detector Data Samples
Input: Stream of Detector Data Samples

A Detector Data Set for each Collimator Hole
Each Data Set Covers a Small Portion of the Entire Imaging Area
Input Data Set Size

- Detector Array: $W_d \times H_d$ Elements (160 x 80)
- Scanning Source: $W_c \times H_c$ Collimators (100 x 100)
- 1 byte per sample
- Each data set: Approximately 128 Mbytes (1 Gbits)
  - $(160 \times 80) \times (100 \times 100)$
Output: Reconstructed Image Planes

- **32 \( (N) \) Distinct Image Planes**
  - Different Positions within Patient
  - Resolution: 1 Mega Pixel: 1000 x 1000
  - Output Data Size:
    - Approximately 128 MBytes
    - 4 Bytes per pixel

- Each input data set must generate 32 \( (N) \) different output images
Multi-Slice Reconstruction

Each Slice Shows Some Anatomical Objects in Focus
Continuously Arriving Data Sets
Computational Requirements

- **Desired Frame Rate**: 30 fps
- **Input Data rate**: 30 Gbits/sec
  - 1 Gbits/data set * 30 frames
- **Output Data rate**: 30 Gbits/sec
  - 128 MBytes * 30
- **Computations**: 1.5 Tera Flops
  - 8 multiplications + 4 adds per pair of (detector element, collimator hole)
- **Memory Bandwidth**: 245 Gbps
Image Reconstruction: Computational Steps
Fundamental Steps

• **Step 1: Back-projection**
  • With One Collimator Hole Lit, Compute the Contribution to an Image Plane

• **Step 2: Accumulation**
  • Aggregate the Back Projection from all Collimators
Back Projection

• A Straight Line Connects a Detector Element to a Collimator Hole
• Intersects each Image Plane at a Point
• Defines a Rectangular ROI (Region of Influence) for the 2D Detector Array
  • Reconstruction Window
  • Reconstruction Ratio $n$
Reconstruction Windows

• For a Given Image Plane
  • Reconstruction Windows Proceed in Raster Scan Fashion
  • All Reconstruction Windows have the Same Size and Pattern
  • There is Overlap Between Reconstruction Windows

• Across Different Image Planes
  • Different Sizes of Reconstruction Windows
  • Size Increases with Distance from Collimator Array
Reconstruction Windows

- \( n \): Reconstruction Ratio
- Constant for an Image Plane
- Increases as Image Plane moves away from Collimator Array

- \( m \): Collimator Offset Constant for a Given Image Plane Resolution
Local View of Reconstruction

Weights (Bilinear Coefficients):

\[
\begin{align*}
(l_x, l_y) &: (1-f_x) \times (1-f_y) \\
(l_{x+1}, l_y) &: f_x \times (1-f_y) \\
(l_x, l_{y+1}) &: (1-f_x) \times f_y \\
(l_{x+1}, l_{y+1}) &: (1-f_x) \times (1-f_y)
\end{align*}
\]
Detector Readings for all Collimators

Input Data Set:

Image Reconstruction: Implementation

Output Data:

<table>
<thead>
<tr>
<th>4 6 1 0 0 5 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 3 4 9 3 8 1</td>
</tr>
<tr>
<td>15 0 3 9 8 7 6</td>
</tr>
<tr>
<td>4 6 1 0 0 5 12</td>
</tr>
<tr>
<td>11 3 4 9 3 8 1</td>
</tr>
<tr>
<td>15 0 3 9 8 7 6</td>
</tr>
<tr>
<td>4 6 1 0 0 5 12</td>
</tr>
<tr>
<td>11 3 4 9 3 8 1</td>
</tr>
<tr>
<td>15 0 3 9 8 7 6</td>
</tr>
</tbody>
</table>

15 0 31 8 72 60 52 12 33 12 33 84 90 103 20 134 90 78 190 202 88 99 1 0 0 0 0 3 33
25 0 31 8 12 60 52 10 33 103 44 33 0 0 123 33 84 90 133 10 144 90 55 190 202 88 99 1 0 0 0 0 3 33
15 0 31 8 72 60 52 12 33 123 54 33 3 0 223 43 84 30 203 30 134 90 48 190 202 88 99 1 0 0 0 0 3 33
15 0 31 8 72 60 52 12 33 123 83 33 0 0 123 33 84 90 103 20 134 90 78 190 202 88 99 1 0 0 0 0 3 33
25 0 31 8 12 60 52 10 33 103 44 33 0 0 123 33 84 90 133 10 144 90 55 190 202 88 99 1 0 0 0 0 3 33
15 0 31 8 72 60 52 12 33 123 54 33 3 0 223 43 84 30 203 30 134 90 48 190 202 88 99 1 0 0 0 0 3 33
Image Reconstruction: Data Flow

• **Objective**
  - For each pixel, compute the grayscale value
  - Aggregate of detector contributions in neighborhood

• **All the Detector Data is Ready in Memory Before Reconstruction Begins**
Parallel Implementation with CUDA
Scope For Parallelism

• **Parallelization Schemes**
  • Different image planes can be reconstructed concurrently
  • Within a plane, every pixel can be reconstructed concurrently
  • Every detector reading can be processed concurrently

• **Data Mapping**
  • Input Detector Data: Texture Memory
  • Output Pixel Data: Global Memory
Detector Centric View

• Each Processing Thread Handles a Single Detector Reading
  • For all Collimator Holes

• Main Computation: Apportion the Detector’s Output to the 4 Nearest Pixels

Data Push Model
CUDA Block and Grid Configuration

- CUDA Blocks for a Subset of Detector Data Elements
  - 2 D Block: $W_d \times H_b$
  - Maximum of 1024 Threads
- Blocks Organized in a 2D Grid
  - $H_d / H_b \times N$
Computational Mapping
Computational Flow

Texture Memory

Global Memory
Detector Centric Reconstruction

- Non-Local and Asymmetric Writes
- Memory Write Contentions
- Poor Arithmetic Intensity
- Multi-GPU Scaling Cumbersome
Pixel Centric View

• Each Thread Handles the Reconstruction of a Single Pixel
  • From all Detector Elements

• Main Computation: Update the Pixel Value Using the Readings of Detectors Closest to it
CUDA Block and Grid Configuration

• A CUDA Block for each Row of Pixels
  • 1 D Block: 1000 Threads

• Blocks Organized in a 2D Grid
  • 1000 x N
Computational Mapping
Computational Flow

Texture Memory

Global Memory
Pixel Centric Reconstruction: Observations

- Each Pixel is Affected by a Range of Collimators
- Across Collimators, the Affected Pixels Have the Same Pattern
- Use Look Up Tables (LUTs) to Remember these, Avoiding Repeated Computations
- Save LUTs in Shared Memory
Computational Flow

Texture Memory

Look Up Tables

Shared Memory

Global Memory
Pixel Centric Reconstruction

- Redundant Memory Writes Avoided
- Four Memory Write Contentions/pixel
- Better Arithmetic Intensity
- Easier Multi-GPU Scaling
Performance Results
## Kepler GPU Specifications

<table>
<thead>
<tr>
<th></th>
<th>Tesla K10 1 card 2 GPU</th>
<th>Tesla K20 1 card 1 GPU</th>
<th>GeForce GTX 690 1 card 2 GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of cores</strong></td>
<td>2 x 1536 = 3072</td>
<td>2496</td>
<td>2 x 1536 = 3072</td>
</tr>
<tr>
<td><strong>GPU Clock rate</strong></td>
<td>745MHz (0.75GHz)</td>
<td>704MHz (0.71 GHz)</td>
<td>1020MHz (1.02GHz)</td>
</tr>
<tr>
<td><strong>Global Memory Size</strong></td>
<td>2 x 4 = 8 GB</td>
<td>5 GB</td>
<td>2 x 2 = 4 GB</td>
</tr>
<tr>
<td><strong>Single Precision</strong></td>
<td>4.58 T Flops</td>
<td>3.52 T Flops</td>
<td>4.58 T Flops</td>
</tr>
<tr>
<td><strong>Double Precision</strong></td>
<td>190 G Flops</td>
<td>1.17 T Flops</td>
<td>190 G Flops</td>
</tr>
<tr>
<td><strong>Memory Clock rate</strong></td>
<td>2500 MHz</td>
<td>2600 MHz</td>
<td>3004 MHz</td>
</tr>
<tr>
<td><strong>Max. No. of threads / MP</strong></td>
<td>2048</td>
<td>2048</td>
<td>2048</td>
</tr>
<tr>
<td><strong>Max. No. of threads / block</strong></td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td><strong>Shared Memory / block</strong></td>
<td>49152 bytes</td>
<td>49152 bytes</td>
<td>49152 bytes</td>
</tr>
</tbody>
</table>
Performance Experiments

• Measure Execution Time for Image Reconstruction
• 3 Kepler GPU Models:
  • Tesla K10, Tesla K20, GeForce GTX 690
• N = 32 planes, 1000 x 1000 pixels each
• Detector Array Size: 160 x 80
• Collimator Array: 100 x 100
• Detector Inputs: Random numbers in the range 0 - 15
Performance: Detector Centric

<table>
<thead>
<tr>
<th>Range of n</th>
<th>Run time on 1 GPU of GTX690 (msec)</th>
<th>Run Time On Q8400 CPU (sec)</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.60 – 2.25</td>
<td>1029.64</td>
<td>394.85</td>
<td>383.483</td>
</tr>
<tr>
<td>1.00 – 1.50</td>
<td>1039.86</td>
<td>366.90</td>
<td>352.83</td>
</tr>
<tr>
<td>1.00 – 2.01</td>
<td>1003.45</td>
<td>387.36</td>
<td>386.02</td>
</tr>
<tr>
<td>1.00 – 3.82</td>
<td>966.57</td>
<td>443.21</td>
<td>458.53</td>
</tr>
</tbody>
</table>
## Performance: Pixel Centric

<table>
<thead>
<tr>
<th>Range of n</th>
<th>Run Time on GPUs (msec)</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Tesla K10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Single GPU</td>
<td>Dual GPU</td>
<td>Single GPU</td>
<td>Dual GPU</td>
<td>Single GPU</td>
<td>Dual GPU</td>
</tr>
<tr>
<td>0.60 – 2.25</td>
<td></td>
<td>155.16</td>
<td>77.11</td>
<td>152.26</td>
<td>74.89</td>
<td>126.55</td>
<td>62.98</td>
</tr>
<tr>
<td>1.00 – 1.50</td>
<td></td>
<td>147.10</td>
<td>73.18</td>
<td>145.33</td>
<td>71.60</td>
<td>111.02</td>
<td>54.32</td>
</tr>
<tr>
<td>1.00 – 2.01</td>
<td></td>
<td>165.70</td>
<td>82.32</td>
<td>162.46</td>
<td>79.90</td>
<td>125.50</td>
<td>59.76</td>
</tr>
<tr>
<td>1.00 – 3.82</td>
<td></td>
<td>227.84</td>
<td>95.66</td>
<td>223.60</td>
<td>109.23</td>
<td>174.94</td>
<td>80.94</td>
</tr>
</tbody>
</table>
Performance: GeForce GTX 690

Linear Speedup: Dual GPU Runtime = 50% of Single GPU Runtime
Observations

• GTX 690 performed slightly better than Tesla K10
• Tesla K10 performed better than Tesla K20
  • Better single precision Flops rating
• Linear scaling with 2 GPUs
• 15 fps with one GPU card (2 GPUs) of GTX 690
• When 2 GPUs were used, the image was divided by rows – top half and bottom half
Summary and Future Work
Summary

• **Application**
  - Scanning Beam X-Ray Fluoroscopy
  - Image Reconstruction

• **CUDA Implementation**
  - Kepler GPUs
  - Single Precision Floating Point

• **Performance**
  - Real time Performance at 15 fps
  - Scalable Parallel Implementation
Future Work: Plane Reduction

• Combine Several Planes into One
Future Work

• Plane Reduction

• Generalize to More Configurations
  • Non-uniform Detector Arrays

• Use GPU-Direct for Data Movement
Thanks to …

- NVIDIA GPU Test Drive
  - Tesla K10, Tesla K20
- NovaRay Medical, Inc.
  - Introductory Slides with Animations
- Triple Ring Technologies
  - Problem Definition
Questions?

• Email: prashanth.bhat@manipal.net

• Web page: http://www.manipal.net