Studying GPU based RTC for TMT NFIRAOS

Lianqi Wang

Thirty Meter Telescope Project

RTC Workshop
Dec 04, 2012
Tomography with iterative algorithms on GPUs
Matrix vector multiply approach
  – Assembling AO control matrix
  – Applying matrix vector multiply
GPU based RTC
Benchmarking results
Conclusion
Minimum Variance Reconstructor

- Minimizing $\sigma^2$ over target FoV (9 directions in $\Phi 30"$)
  $\sigma^2 = \langle \| H_x x - H_a a \|^2 \rangle$
  with $g = G_p H_x x + n$

- Gives tomography
  $$\tilde{x} = \left( H_x^T G_p^T C_{nn}^{-1} G_p H_x + C_{xx}^{-1} \right)^{-1} H_x^T G_p^T C_{nn}^{-1} g$$

- And DM fitting over target FoV
  $$a = \left( H_a^T W H_a \right)^{-1} H_a^T W \tilde{H}_x \tilde{x}$$
Tomography

\[ x = \left( H_x^T G_p^T C_{nn}^{-1} G_p H_x + C_{xx}^{-1} \right)^{-1} H_x^T G_p^T C_{nn}^{-1} g \]

- \( H_x \): ray tracing from \( x \) to \( p \)
- \( G_p \): compute gradient from \( p \)
- \( C_{nn} \): Noise covariance matrix
- \( C_{xx}^{-1} \): Using bi-harmonic approximation

The inverse is solved using iterative algorithms like Conjugate Gradients

Turbulence grid
\( \frac{1}{2} \) or \( \frac{1}{4} \) m

Actuator grid
\( \frac{1}{2} \) m

Pipul grid
\( \frac{1}{2} \) m
DM Fitting

\[ a = \left( H_a^T W H_a \right)^{-1} H_a^T W \tilde{H} x x \]

Use sparse matrix based operation for the moment.
Benchmarking

Hardware
- Single Core i7 3820 @ 3.60 GHz
- 2 NVIDIA GTX 580 GPU board
  - 3 GB graphics memory with 192GB/s theoretical throughput
  - 512 stream processors with 1.6TFlops theoretical throughput

Software
- 64 bit Linux
- CUDA 4.0 C runtime library with nvcc
- cublas, cuFFT, cuSparse, cuRand, etc from CUDA package
- Use single precision floating number
## Benchmarking Results of Iterative Algorithms for Tomography

<table>
<thead>
<tr>
<th>Method</th>
<th>Timing (ms)</th>
<th>Incr WFE (nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG30OS0</td>
<td>5.17</td>
<td>44.3</td>
</tr>
<tr>
<td>CG30OS4</td>
<td>18.20</td>
<td>0</td>
</tr>
<tr>
<td>CG30OS6</td>
<td>12.3</td>
<td>11.2</td>
</tr>
<tr>
<td>FD1OS0</td>
<td>0.49</td>
<td>52.8</td>
</tr>
<tr>
<td>FD1OS6</td>
<td>1.37</td>
<td>33.8</td>
</tr>
<tr>
<td>FD2OS0</td>
<td>0.78</td>
<td>42.9</td>
</tr>
<tr>
<td>FD2OS6</td>
<td>2.60</td>
<td>-16.9</td>
</tr>
<tr>
<td>FD3OS0</td>
<td>1.04</td>
<td>42.6</td>
</tr>
<tr>
<td>FD3OS6</td>
<td>3.04</td>
<td>-19.7</td>
</tr>
</tbody>
</table>

**CG**: Conjugate Gradients  
**FD**: Fourier Domain Preconditioned CG  
**OSn**: Over sampling n tomography layers (¼ m spacing)
### Tomography Detailed Timing

\[ x = \left( H_x^T G_p^T C_{nn}^{-1} G_p H_x + C_{xx}^{-1} \right)^{-1} H_x^T G_p^T C_{nn}^{-1} g \]

<table>
<thead>
<tr>
<th></th>
<th>micro-sec</th>
<th>Flop</th>
<th>Mem</th>
<th>GB/s</th>
<th>GFlops</th>
</tr>
</thead>
<tbody>
<tr>
<td>( H_x )</td>
<td>74</td>
<td>10616832</td>
<td>15925248</td>
<td>215</td>
<td>143</td>
</tr>
<tr>
<td>( G_p )</td>
<td>45</td>
<td>278856</td>
<td>1921008</td>
<td>43</td>
<td>6</td>
</tr>
<tr>
<td>( G_p^T )</td>
<td>50</td>
<td>402792</td>
<td>2106912</td>
<td>42</td>
<td>8</td>
</tr>
<tr>
<td>( H_x' )</td>
<td>122</td>
<td>10616832</td>
<td>15925248</td>
<td>131</td>
<td>87</td>
</tr>
<tr>
<td>( C_{xx}^{-1} )</td>
<td>48</td>
<td>626688</td>
<td>2064384</td>
<td>43</td>
<td>13</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>339</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Preconditioner: \( M_x = F^{-1} [A F [x]] \) where \( A \) is block diagonal matrix

<table>
<thead>
<tr>
<th></th>
<th>micro-sec</th>
<th>Flop</th>
<th>Mem</th>
<th>GB/s</th>
<th>GFlops</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F )</td>
<td>115</td>
<td>79,531,761</td>
<td>1769472</td>
<td>15</td>
<td>692</td>
</tr>
<tr>
<td>( A )</td>
<td>188</td>
<td>5,308,416</td>
<td>10616832</td>
<td>56</td>
<td>28</td>
</tr>
<tr>
<td>( F^{-1} )</td>
<td>114</td>
<td>79,531,761</td>
<td>1769472</td>
<td>16</td>
<td>698</td>
</tr>
</tbody>
</table>
DM Fitting uses sparse matrix approach. Haven’t yet optimized. Potential to speed up by a few times.

<table>
<thead>
<tr>
<th></th>
<th>micro-sec</th>
<th>LHS</th>
<th>RHS</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tomography (2 Iterations)</td>
<td>2016</td>
<td>584</td>
<td></td>
<td>2600</td>
</tr>
<tr>
<td>DM Fitting (4 iterations)</td>
<td>1641</td>
<td>2862</td>
<td></td>
<td>4503</td>
</tr>
</tbody>
</table>
What limits our performance?

- We are not limited by the steady rate throughput
  - 1581 GFlops of single precision floating point number operation
  - 192 GB/s device memory

- We are limited by latency
  - Kernel launch overhead:
    - ~2.3 micro-second for asynchronous launch,
    - ~6.5 micro-second for synchronization
  - Device memory latency: 600 cycles, ~0.3 micro-second, for intermediate quantities.
  - Sparse matrix vector multiply need to be carefully optimized
  - PCI-E interface (2.0): 8GB/s, 11 micro-second latency, for gradients and actuator commands input/output
Still a long way to go with iterative algorithms for <1.25 ms latency
  – Hard to parallel across GPUs due to low PCIe bandwidth and high latency

MVM is the easiest to implement in parallel
  – Regular memory access pattern avoids memory latency issue
  – GPU is good at it with ~200 GB/s device memory bandwidth

Need to obtain the control matrix
  – Update the control matrix every 10 seconds

Solution: Using iterative algorithms to solve columns of \( I \)
  – Update the control matrix with warm restart
Tomography + fitting can be summarized as \( E = F_L^{-1}F_RR_L^{-1}R_R \)

With

\[
R_L = H_x^T G_p C_{nn}^{-1} G_p H_x + C_{xx}^{-1} ; \quad R_R = H_x^T G_p C_{nn}^{-1} \quad \text{Tomography}
\]

\[
F_L = H_a^T W H_a ; \quad F_R = H_a^T \tilde{W} \tilde{H}_x \quad \text{DM Fitting}
\]

Matrix dimensions are

\[
(7083 \times 30984) = (7083 \times 7083)\times (7083 \times 62311) \times (62311 \times 62311)\times (62311 \times 30984)
\]

- 7083: number of active actuators
- 30984: number of WFS gradients
- 62311: number of points in tomography grid

We assemble \( E \) by solving each column one at a time

\[
E(:,j) = F_L^{-1}F_R R_L^{-1}R_R e_j
\]

There are 30984 tomography operations total

- 1500 seconds to create (FDPCG with 50 iterations)
- 150 seconds to update (when condition changes. 5 iterations, using warm restart)
Assembling the transpose of control matrix in GPUs

- Solve for the transpose $E^T = R_R^T R_L^{-1} F_R^T F_L^{-1}$
- The dimensions are
  $$(30984 \times 7083) = (30984 \times 62311) (62311 \times 62311)^{-1} \times (62311 \times 7083)(7083 \times 7083)^{-1}$$
- A factor of 4 reduction in number of tomography operations compared to solve E directly
  - $F_L^{-1}$ can be reused
  - 400 seconds to create (50 FD iterations, 2.2ms each step)
  - 40 seconds to update (5 FD iterations)
- With a 8 GPU machine
  - 50 seconds to create (can be avoided by warm warm restart)
  - 5 seconds to update (5 FD iterations, using warm restart)
  - 10 seconds for 10 FD iterations when condition varies significantly
  - NFIRAOS requirement is 10 seconds.
RMS wavefront error in science FoV is comparable to baseline algorithm (CG30) with 50 FDPCG iterations (OS6)
GPUs required to apply MVM at 800 Hz for NFIRAOS

Assuming 1.00 ms total time

<table>
<thead>
<tr>
<th>NGPU</th>
<th>Compute MFLOP</th>
<th>Memory MB</th>
<th>PCI-E kB</th>
<th>Compute GFLOPS</th>
<th>Device Mem GB/s</th>
<th>PCI-E MB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>209</td>
<td>837</td>
<td>149</td>
<td>409</td>
<td>818</td>
<td>145</td>
</tr>
<tr>
<td>6</td>
<td>35</td>
<td>140</td>
<td>48</td>
<td>82</td>
<td>136</td>
<td>47</td>
</tr>
<tr>
<td>8</td>
<td>26</td>
<td>105</td>
<td>43</td>
<td>51</td>
<td>102</td>
<td>42</td>
</tr>
<tr>
<td>Rating</td>
<td>3G</td>
<td></td>
<td>1581</td>
<td>192</td>
<td>8192</td>
<td></td>
</tr>
</tbody>
</table>

Red: No achievable
Yellow: Nearly achievable
Green: Achievable

A minimum of 6-8 GTX 580 GPU is needed to apply MVM
Proposed GPU RTC Architecture

2 GPUs per WFS

- NGS WFS
- LGS WFS 1
- LGS WFS 2
- LGS WFS 3
- LGS WFS 4
- LGS WFS 5
- LGS WFS 6

1U Rack server with 2 GPU

2U Rack server with 2 CPU
PCI-E: 2 sFPDP x4
PCI-E: Camera Link

Infiniband Switch

TMT Lan Switch

2U Rack server with 2 CPU
PCI-E: Camera Link

1U Rack Server with 3 GPU

1U Rack Server with 3 GPU

PSF Reconstructor

1U Rack Server with 3 GPU

Telemetry Storage with Cluster File System

Computer Room

RPG Updates MVM matrix

OIWWSs

DME

TTS

LGSF

NGS FSM

Private Lan Switch

Cameralink

sFPDP x7
1U Server for Each LGS WFS

- Infiniband for control matrix and telemetry
- Cameralink (or else?) for WFS pixel data
Benchmarked for an LGS WFS with 2 GTX 580

- MVM takes most of the time.
- Memory copying is indeed concurrent with computing
1GPU: 1.11-1.16 ms
2GPU: 0.66-0.71 ms
Jitter: 48 µs p/v, 2.2 µs RMS
CPU shield is used
For 1000 seconds

Occasional large jitter

Ways to avoid it?
Other real time tasks

- Copying updated MVM matrix to RTC
  - Do so after DM actuator commands are ready
  - Measured 0.1 ms for 10 columns
  - 519 time steps to copy 5182 columns

- Collect statistics to update matched filter coefficients
  - Do so after DM actuator commands are ready
  - Benchmark next

- Etc

- 0.5 ms to spare
Background process

- Updating MVM matrix when condition varies
  - Role of reconstruction parameter generator (RPG).
  - Copy to RTC over Infiniband or ethernet
- etc
Current gen GPU can handle iterative wavefront reconstruction algorithms in a few ms.

Control matrix for MVM can be updated every 10 seconds using FDPCG tomography algorithm to cope with varying conditions.

With MVM, a 2 GPU server per LGS WFS can turn pixels into DM actuator commands in 0.7 ms, meeting the requirement with good margin.

Any other concerns?
Acknowledgements

The author gratefully acknowledges the support of the TMT collaborating institutions. They are
– the Association of Canadian Universities for Research in Astronomy (ACURA),
– the California Institute of Technology,
– the University of California,
– the National Astronomical Observatory of Japan,
– the National Astronomical Observatories of China and their consortium partners,
– and the Department of Science and Technology of India and their supported institutes.

This work was supported as well by
– the Gordon and Betty Moore Foundation,
– the Canada Foundation for Innovation,
– the Ontario Ministry of Research and Innovation,
– the National Research Council of Canada,
– the Natural Sciences and Engineering Research Council of Canada,
– the British Columbia Knowledge Development Fund,
– the Association of Universities for Research in Astronomy (AURA)
– and the U.S. National Science Foundation.