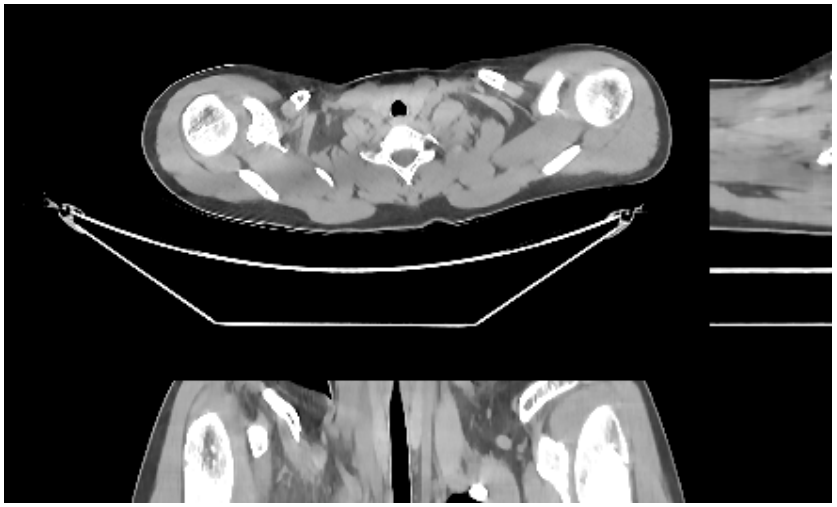
Fast GPU-Driven Model-Based X-ray CT Image Reconstruction via Alternating Dual Updates

Madison G. McGaffin Jeffrey A. Fessler

13th International Fully 3D Conference
June 3rd, 2015

Model-based X-ray CT Image Reconstruction

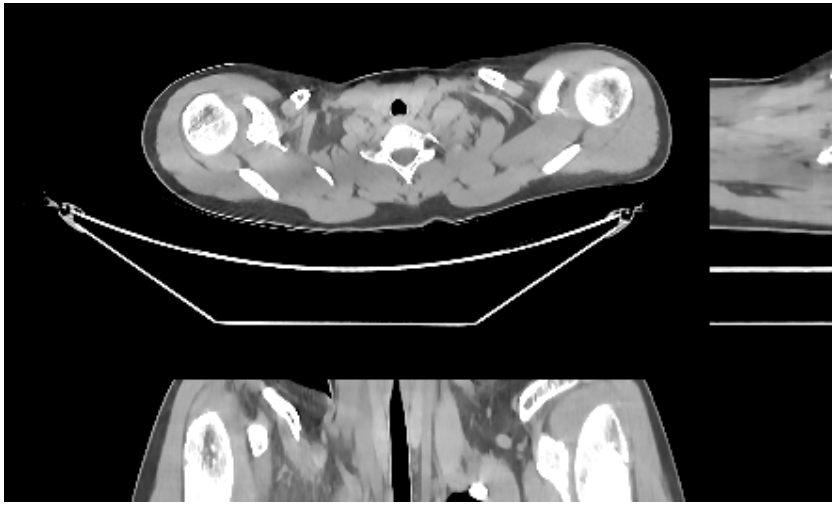
Filtered backprojection
(Very fast)



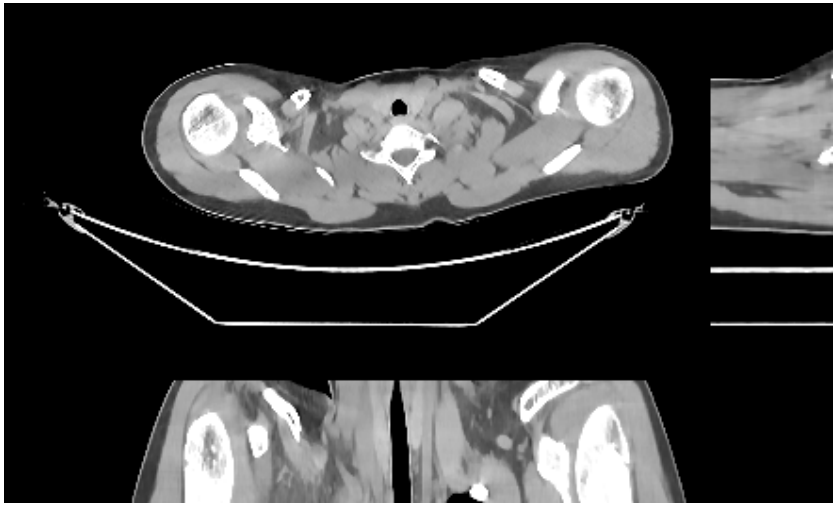
Model-based
(Slow?)

Model-based X-ray CT Image Reconstruction

Filtered backprojection



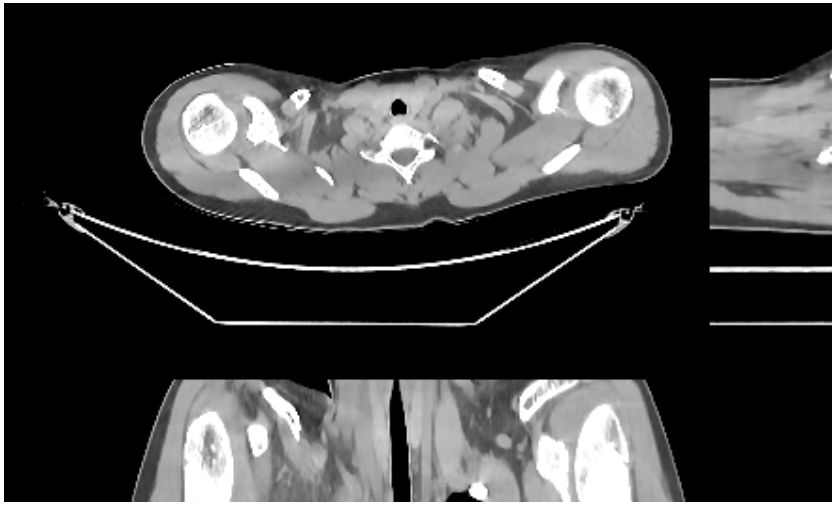
Model-based



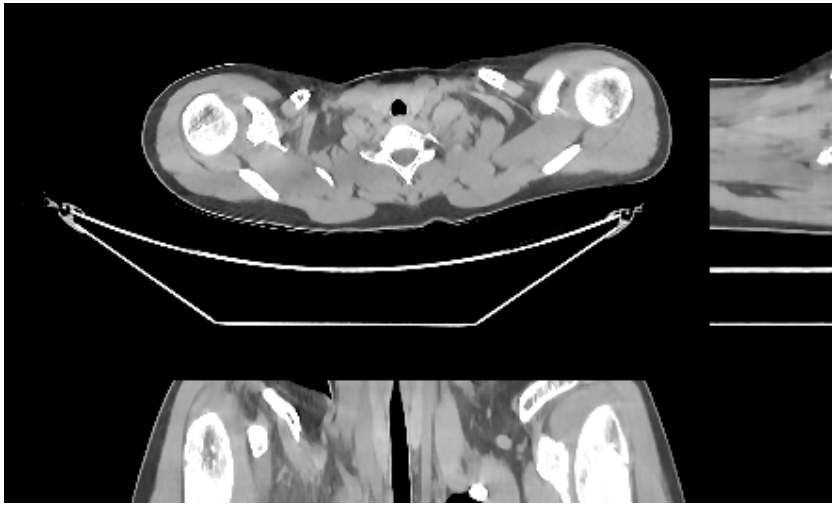
Proposed
4.8 Minutes (3 HU RMSD)

Model-based X-ray CT Image Reconstruction

Filtered backprojection



Model-based



Proposed
4.8 Minutes (3 HU RMSD)

Use duality & group coordinate ascent to exploit structure, embrace parallelism, and interleave memory- and computationally-intensive tasks to maximize throughput.

Model-based X-ray CT Image Reconstruction

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Data-fit

Fits data

$$\mathsf{L}(\mathbf{p}) = \sum_{i=1}^{M} l_i(p_i)$$

$$l_i(p_i) = \frac{w_i}{2}(p_i - y_i)^2$$

A: System model

Regularizer

Penalizes roughness

$$\mathsf{R}(\mathbf{d}) = \sum_{k=1}^{K} r_k(d_k)$$

$$r_k(d_k) = \beta_k \psi(d_k)$$

C: Finite differences

Nonnegativity

Makes nonnegative

$$\mathsf{I}(\mathbf{x}) = \sum_{j=1}^{N} \iota_j(x_j)$$

$$\iota_j(x_j) = \begin{cases} 0, & x_j \geq 0 \\ \infty, & \text{else} \end{cases}$$

Model-based X-ray CT Reconstruction

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Data-fit

☹

Lots of data, slow gradients, poorly conditioned

☺

Highly "correlated" views (ordered subsets)

Regularizer

☹

Nonquadratic, sometimes nondifferentiable

☺

Localized, Markov-like

Nonnegativity

☹

Line searches are hard

☺

Clamping is easy

We want to exploit these structures!

Duality Trick

Iterative procedure:

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Just as hard to solve as the original problem?

Duality Trick

Iterative procedure:

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Rewrite terms using convex conjugates:

$$= \arg\min_{\mathbf{x}} \sup_{\mathbf{u}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathbf{x}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{u} - \mathsf{L}^*(\mathbf{u}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Duality Trick

Iterative procedure:

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Rewrite terms using convex conjugates:

$$= \arg\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathbf{x}^{\mathsf{T}}(\mathbf{A}^{\mathsf{T}}\mathbf{u} + \mathbf{C}^{\mathsf{T}}\mathbf{v}) - \mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) + \mathsf{I}(\mathbf{x})$$

Duality Trick

Iterative procedure:

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) + \mathsf{I}(\mathbf{x})$$

Rewrite terms using convex conjugates:

$$\begin{aligned} &= \arg\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathbf{x}^\mathsf{T}(\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}) \\ &\qquad\qquad - \mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) - \mathsf{I}^*(\mathbf{z}) \\ &= \arg\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathsf{S}^{(n)}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \mathbf{z}) \end{aligned}$$

$\mathsf{S}^{(n)}$ is a very "simple" function of $\mathbf{x}$!

Duality Trick

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \frac{\mu}{2}\left|\left|\mathbf{x} - \mathbf{x}^{(n)}\right|\right|_2^2 + \mathbf{x}^\intercal(\mathbf{A}^\intercal\mathbf{u} + \mathbf{C}^\intercal\mathbf{v} + \mathbf{z})$$
$$- \mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) - \mathsf{I}^*(\mathbf{z})$$
$$= \arg\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathsf{S}^{(n)}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \mathbf{z})$$

By Fenchel duality we can reverse minimization, and maximization:

$$\min_{\mathbf{x}} \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathsf{S}^{(n)}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \mathbf{z}) = \sup_{\mathbf{u},\mathbf{v},\mathbf{z}} \min_{\mathbf{x}} \mathsf{S}^{(n)}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \mathbf{z}),$$

and by strong convexity of $\mathsf{S}^{(n)}$ in $\mathbf{x}$, $\mathbf{x}^{(n+1)}$ is a function of the dual variables:

$$\mathbf{x}^{(n+1)} = \widetilde{\mathbf{x}}^{(n+1)}\left(\mathbf{u}^{(n+1)}, \mathbf{v}^{(n+1)}, \mathbf{z}^{(n+1)}\right)$$
$$= \mathbf{x}^{(n)} - \frac{1}{\mu}\left(\mathbf{A}^\intercal\mathbf{u}^{(n+1)} + \mathbf{C}^\intercal\mathbf{v}^{(n+1)} + \mathbf{z}^{(n+1)}\right)$$

Duality Approach

Duality-based approach:

$$\mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$
$$- \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

$$\mathbf{u}^{(n+1)}, \mathbf{v}^{(n+1)}, \mathbf{z}^{(n+1)} \approx \arg\max_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z})$$
$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}\left(\mathbf{A}^\mathsf{T}\mathbf{u}^{(n+1)} + \mathbf{C}^\mathsf{T}\mathbf{v}^{(n+1)} + \mathbf{z}^{(n+1)}\right)$$

Technical point: under ADMM-like conditions, approximate maximization leads to convergence.

Duality Approach

$$\mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$
$$- \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

$$\mathbf{u}^{(n+1)}, \mathbf{v}^{(n+1)}, \mathbf{z}^{(n+1)} \approx \arg\max_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z})$$
$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}\left(\mathbf{A}^\mathsf{T}\mathbf{u}^{(n+1)} + \mathbf{C}^\mathsf{T}\mathbf{v}^{(n+1)} + \mathbf{z}^{(n+1)}\right)$$

Each dual variable corresponds to a one-dimensional data-fit, regularizer, or nonnegativity term.

- $\mathrm{D}^{(n)}$ is separable except for the first quadratic term.
- By updating the variables in groups, we can exploit the structure of the cost function.

Duality Approach

$$\mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$
$$- \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

$$\mathbf{u}^{(n+1)}, \mathbf{v}^{(n+1)}, \mathbf{z}^{(n+1)} \approx \arg\max_{\mathbf{u},\mathbf{v},\mathbf{z}} \mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z})$$

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}\left(\mathbf{A}^\mathsf{T}\mathbf{u}^{(n+1)} + \mathbf{C}^\mathsf{T}\mathbf{v}^{(n+1)} + \mathbf{z}^{(n+1)}\right)$$

Group coordinate ascent:

In each iteration n, repeat N_{op} times:

- Choose a group of elements of $\mathbf{u}$, $\mathbf{v}$, or $\mathbf{z}$:
  - a view-sized group $\mathbf{u}_g$,
  - a "half-difference" $\mathbf{v}_g$,
  - the nonnegativity variable $\mathbf{z}$.
- Increase $\mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z})$ by optimizing over the selected group of elements and holding all others constant.

Duality Approach

Duality-based approach:

$$\mathrm{D}^{(n)}(\mathbf{u}, \mathbf{v}, \mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2 - \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

Group coordinate ascent:
In each iteration n, repeat N_{op} times:

- Choose a group of elements of $\mathbf{u}$, $\mathbf{v}$, or $\mathbf{z}$:
  - a view-sized group $\mathbf{u}_g$,
  - a "half-difference" $\mathbf{v}_g$,
  - the nonnegativity variable $\mathbf{z}$.
- Increase $\mathrm{D}^{(n)}(\mathbf{u}, \mathbf{v}, \mathbf{z})$ by optimizing over the selected group of elements and holding all others constant.

Tomography ($\mathbf{u}$) Update

Update a single $\mathbf{u}_g$ view at a time. Entries are coupled by $\mathbf{A}_g\mathbf{A}_g^\intercal$. Majorize with with precomputed diagonal majorizer

$$\mathbf{M}_g = \operatorname*{diag}_i\left\{\left[\mathbf{A}_g\mathbf{A}_g^\intercal\mathbf{1}\right]_i\right\} \succeq \mathbf{A}_g\mathbf{A}_g^\intercal,$$

and update:

$$\mathbf{u}_g^+ = (\mathbf{W}_g\mathbf{M}_g + \mu\mathbf{I})^{-1}\mathbf{W}_g\left(\mu(\mathbf{A}_g\widetilde{\mathbf{x}} - \mathbf{y}_g) + \mathbf{M}_g\mathbf{u}_g^-\right).$$

Updates the induced image $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}}^{(n+1)}(\mathbf{u}, \mathbf{v}, \mathbf{z})$

$$\widetilde{\mathbf{x}}^+ = \widetilde{\mathbf{x}}^- - \frac{1}{\mu}\mathbf{A}_g{}^\intercal\left(\mathbf{u}_g^+ - \mathbf{u}_g^-\right)$$

Requires one-view projection, one-view backprojection, and vector operations. Like a **single-view** ordered subsets, but convergent!

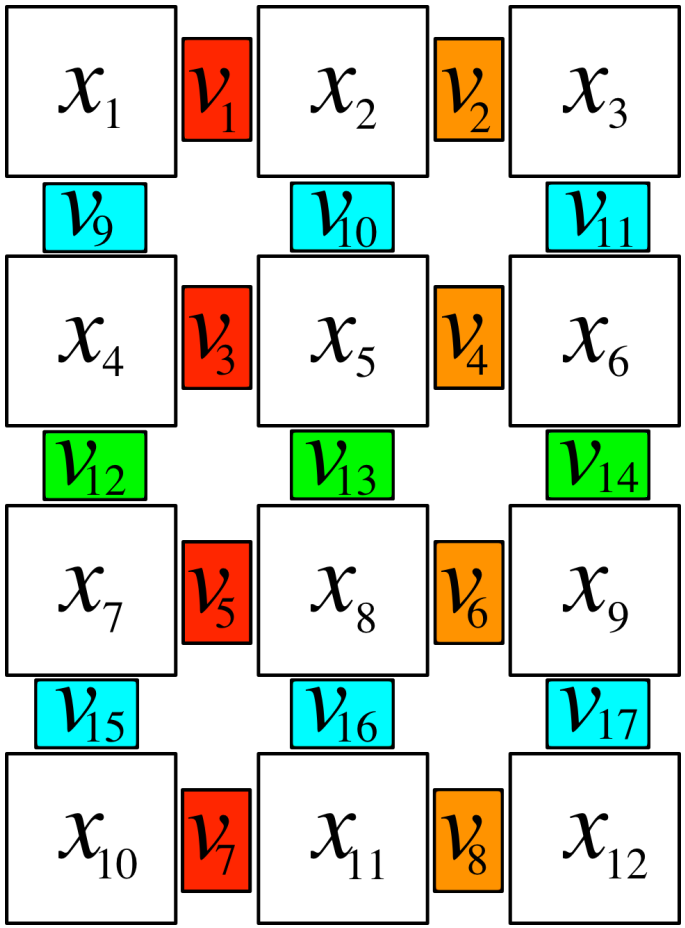
Duality Approach

$$\mathrm{D}^{(n)}(\mathbf{u}, \mathbf{v}, \mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$
$$- \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

Group coordinate ascent:
In each iteration n, repeat N_{op} times:

- Choose a group of elements of $\mathbf{u}$, $\mathbf{v}$, or $\mathbf{z}$:
  - a view-sized group $\mathbf{u}_g$,
  - a "half-difference" $\mathbf{v}_g$,
  - the nonnegativity variable $\mathbf{z}$.
- Increase $\mathrm{D}^{(n)}(\mathbf{u}, \mathbf{v}, \mathbf{z})$ by optimizing over the selected group of elements and holding all others constant.

Denoising ($\mathbf{v}$) Update



- Entries of $\mathbf{v}_g$ are differences, coupled by $\mathbf{C}_g\mathbf{C}_g^\mathsf{T}$.
- **Every other** difference in each direction affects disjoint pixels.
- Optimizing over groups of differences touching disjoint pixels becomes a set of independent, **one-dimensional subproblems**.
- Updating each element of $\mathbf{v}_g$ updates two pixels.

For example, $\{v_9, v_{10}, v_{11}, v_{15}, v_{16}, v_{17}\}$ one half-vertical difference group and $\{v_{12}, v_{13}, v_{14}\}$ is another.

Group Coordinate Ascent

$$\mathrm{D}^{(n)}(\mathbf{u},\mathbf{v},\mathbf{z}) = -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$
$$- \mathrm{L}^*(\mathbf{u}) - \mathrm{R}^*(\mathbf{v}) - \mathrm{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

Loop iterations n:

- Update N_{tomo} randomly-chosen view-sized groups of $\mathbf{u}$
- Update $\mathbf{z}$ (clamping)
- Update N_{tomo} randomly-chosen view-sized groups of $\mathbf{u}$
- Loop N_{denoise} times:
  - Update N_{tomo} randomly-chosen view-sized groups of $\mathbf{u}$
  - Update randomly-chosen "half-difference" of $\mathbf{v}$
  - Update N_{tomo} randomly-chosen view-sized groups of $\mathbf{u}$
- Compute $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}(\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})$

Relative Computational Costs

Operation	Computation
Tomography ($\mathbf{u}$)	$\mathbf{A}_g$, $\mathbf{A}_g^\mathsf{T}$
Denoising ($\mathbf{v}$)	1D Shrinkage
Nonnegativity ($\mathbf{z}$)	1D Clamping

Structure-exploiting updates via group coordinate ascent.

Relative Computational Costs

Operation	Memory required	Computation
Tomography ($\mathbf{u}$)	View-sized, $\approx$ 4MB	$\mathbf{A}_g$, $\mathbf{A}_g^\mathsf{T}$
Denoising ($\mathbf{v}$)	Half-image $\mathbf{v}_g$, 100s MB	1D Shrinkage
Nonnegativity ($\mathbf{z}$)	Image-sized $\mathbf{z}$, 100s MB	1D Clamping

Storing all the dual variables on the GPU is impractical.

Relative Computational Costs

Operation	Memory required	Computation
Tomography ($\mathbf{u}$)	Small	Large
Denoising ($\mathbf{v}$)	Large	Small
Nonnegativity ($\mathbf{z}$)	Large	Small

Trick: interleave computationally intensive operations and large memory transfers!

Group Coordinate Ascent

Loop iterations n:

- Transfer $\mathbf{z}$ from host to GPU
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Update $\mathbf{z}$
- Transfer $\mathbf{z}$ from GPU to host
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$

Group Coordinate Ascent

Loop iterations n:

- Transfer $\mathbf{z}$ from host to GPU
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Update $\mathbf{z}$
- Transfer $\mathbf{z}$ from GPU to host
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Loop N_{denoise} times:
  - Select random "half-difference" $\mathbf{v}_g$
  - Transfer $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
  - Update $\mathbf{v}$
  - Transfer updated $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$

Group Coordinate Ascent

Loop iterations n:

- Transfer $\mathbf{z}$ from host to GPU
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Update $\mathbf{z}$
- Transfer $\mathbf{z}$ from GPU to host
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Loop N_{denoise} times:
  - Select random "half-difference" $\mathbf{v}_g$
  - Transfer $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
  - Update $\mathbf{v}$
  - Transfer updated $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Compute $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}(\mathbf{A}^\intercal \mathbf{u} + \mathbf{C}^\intercal \mathbf{v} + \mathbf{z})$

Group Coordinate Ascent

"Hide" the cost of memory transfers behind computation!

Loop iterations n:

- Transfer $\mathbf{z}$ from host to GPU
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Update $\mathbf{z}$
- Transfer $\mathbf{z}$ from GPU to host
  - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Loop N_{denoise} times:
  - Select random "half-difference" $\mathbf{v}_g$
  - Transfer $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
  - Update $\mathbf{v}$
  - Transfer updated $\mathbf{v}_g$ from host to GPU
    - Meanwhile, update N_{tomo} view-sized groups of $\mathbf{u}$
- Compute $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}(\mathbf{A}^\top \mathbf{u} + \mathbf{C}^\top \mathbf{v} + \mathbf{z})$

Experiment Setup

Hardware:

- Host: 48 GB of RAM
- GPUs: Four NVIDIA Tesla C2050s with 3GB of memory each

Other algorithms (12 subsets for each):

- Ordered subsets with separable quadratic surrogates (OS)[1]
- OS-SQS with Nesterov's fast gradient method (FGM)[2]
- OS-SQS with Kim's optimized gradient method (OGM)[3]

All algorithms initialized with FBP, used the Fair potential function ψ, Regularizer penalized all 3D neighbor differences.

[1] Erdogan and Fessler, "Ordered subsets algorithms for transmission tomography".

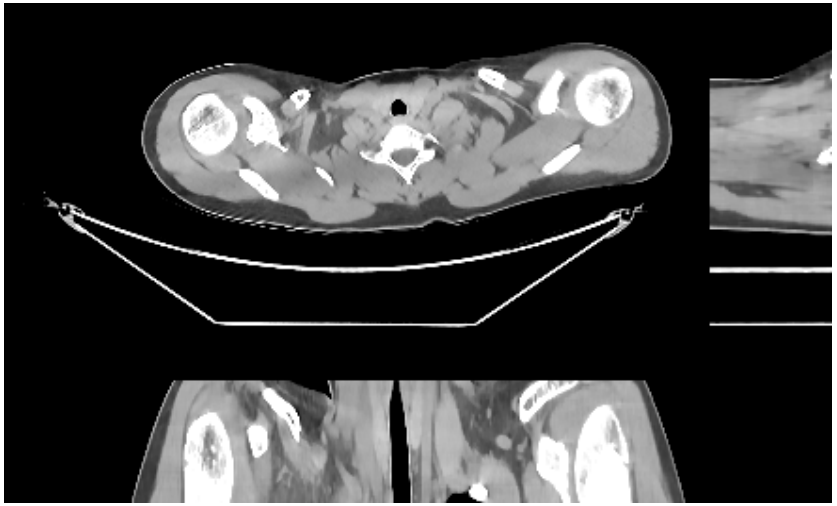
[2] Kim, Ramani, and Fessler, "Combining ordered subsets and momentum for accelerated X-ray CT image reconstruction".

[3] Kim and Fessler, "Optimized gradient methods for smooth convex minimization".

Experiment: Helical Shoulder Scan

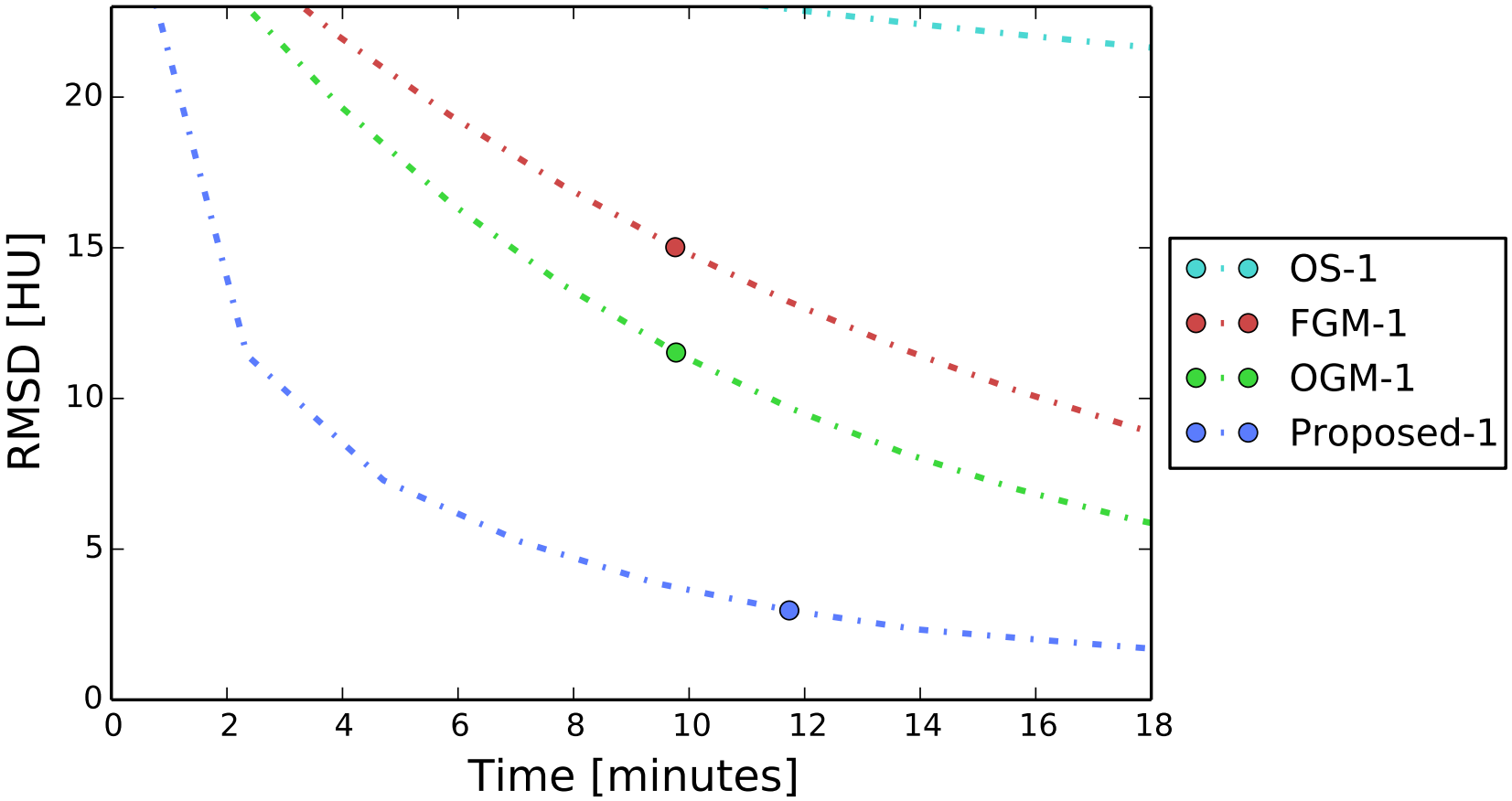
- 888 channels, 32 rows, 6852 views
- Reconstructed on $512 \times 512 \times 109$-pixel grid

(a) Filtered backprojection

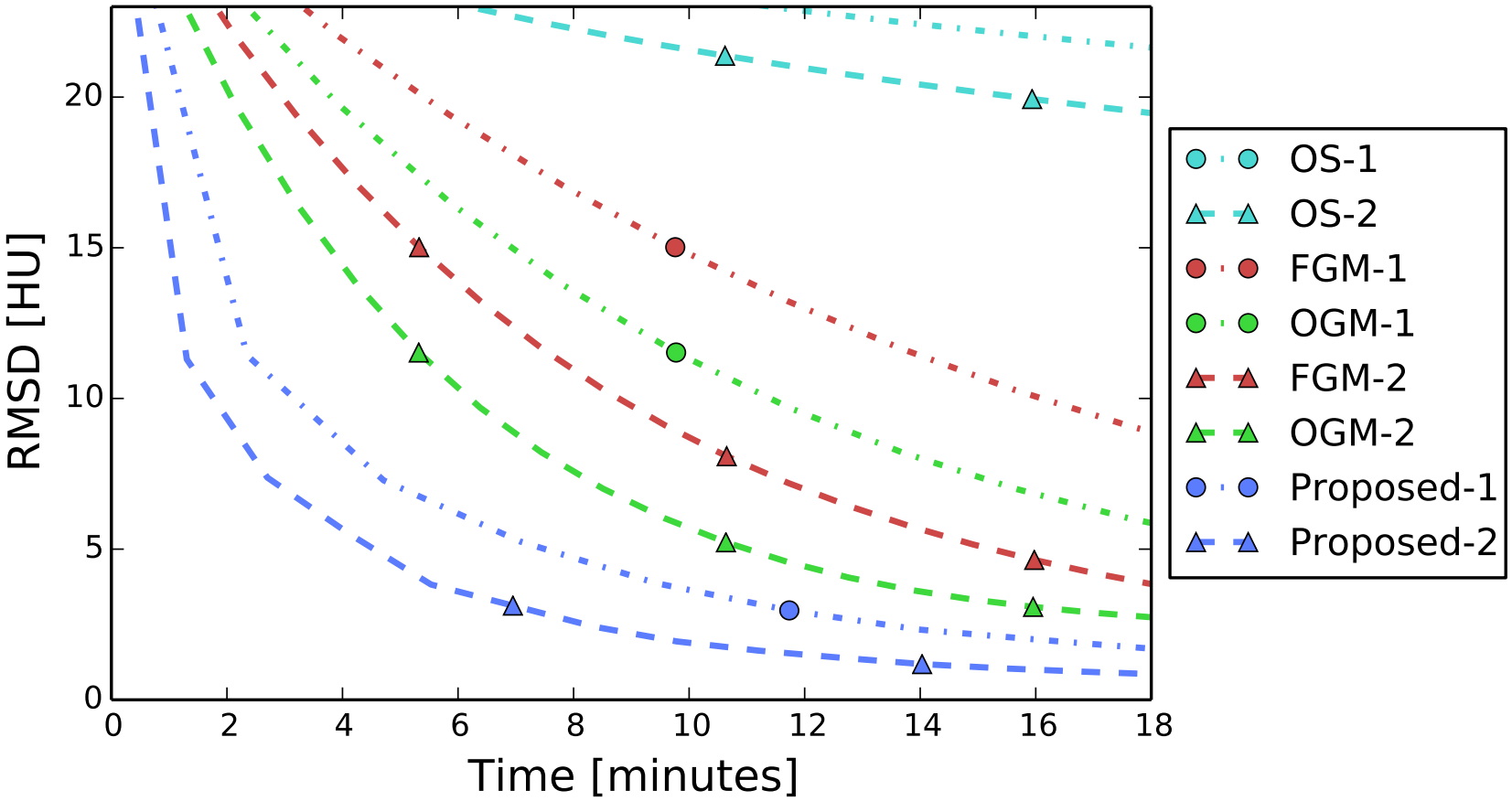


(b) Reference

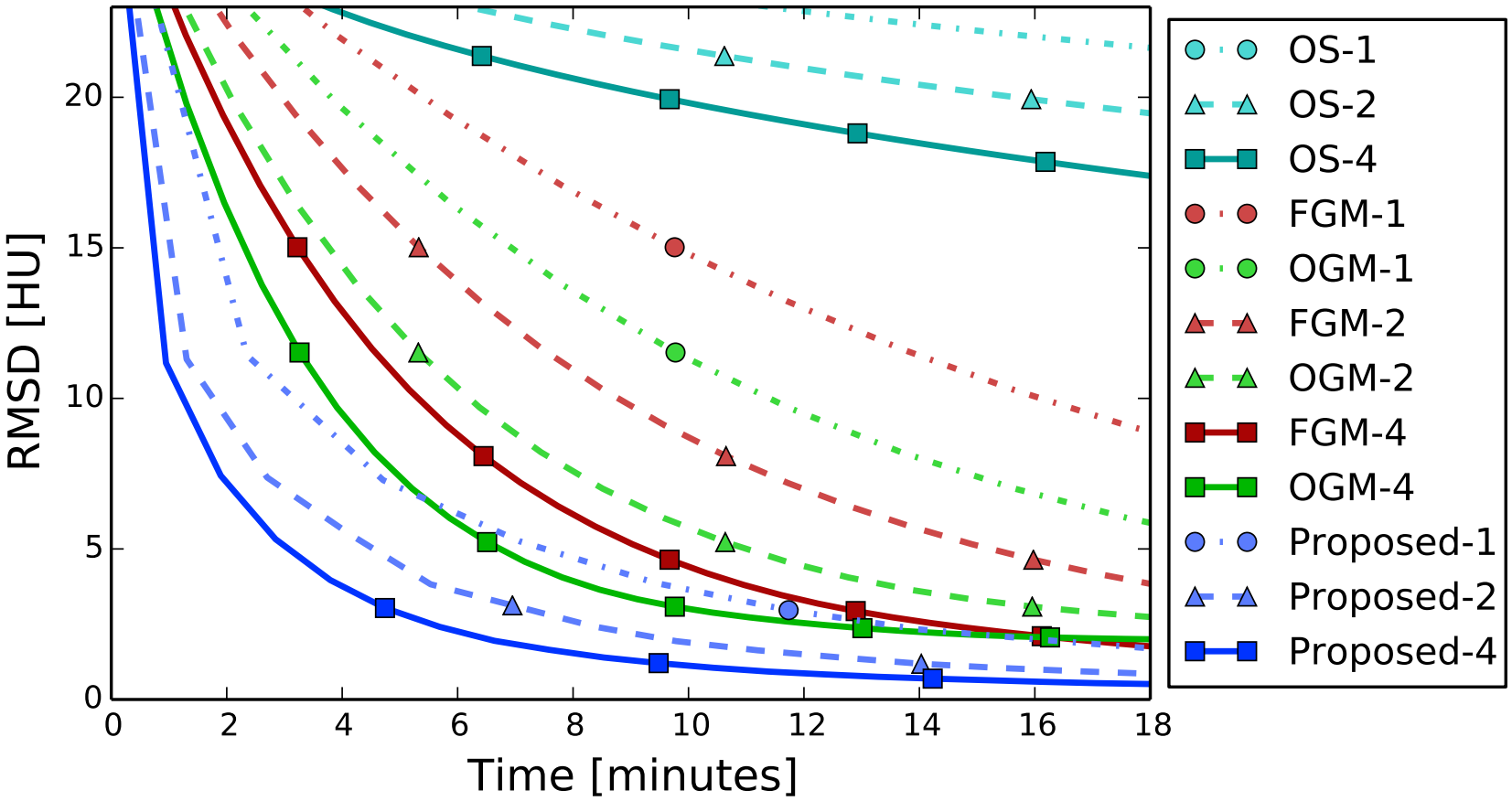
Experiment: Helical Shoulder Scan



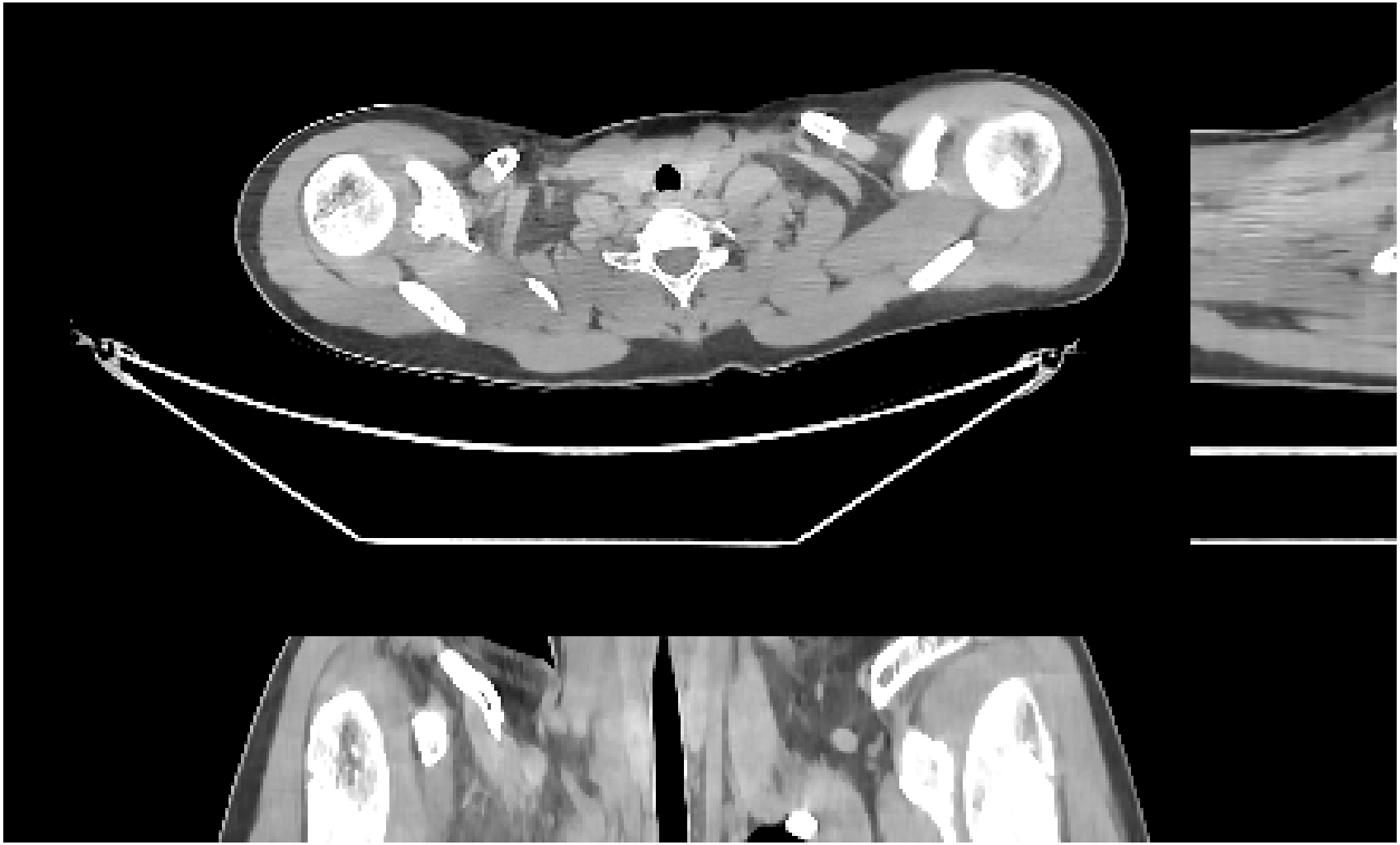
Experiment: Helical Shoulder Scan



Experiment: Helical Shoulder Scan

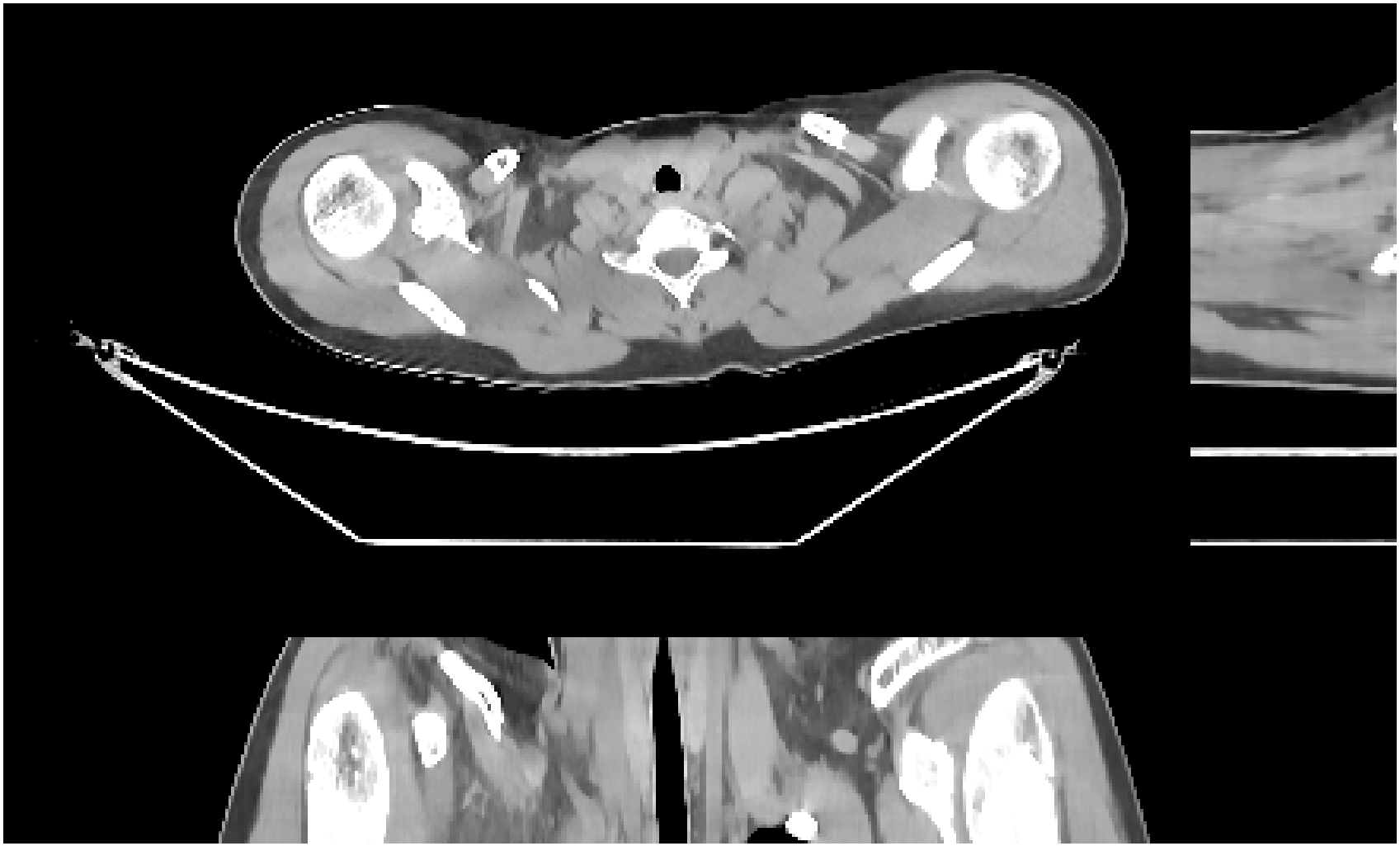


Experiment: Helical Shoulder Scan



OS-OGM after 5.2 min.

Experiment: Helical Shoulder Scan



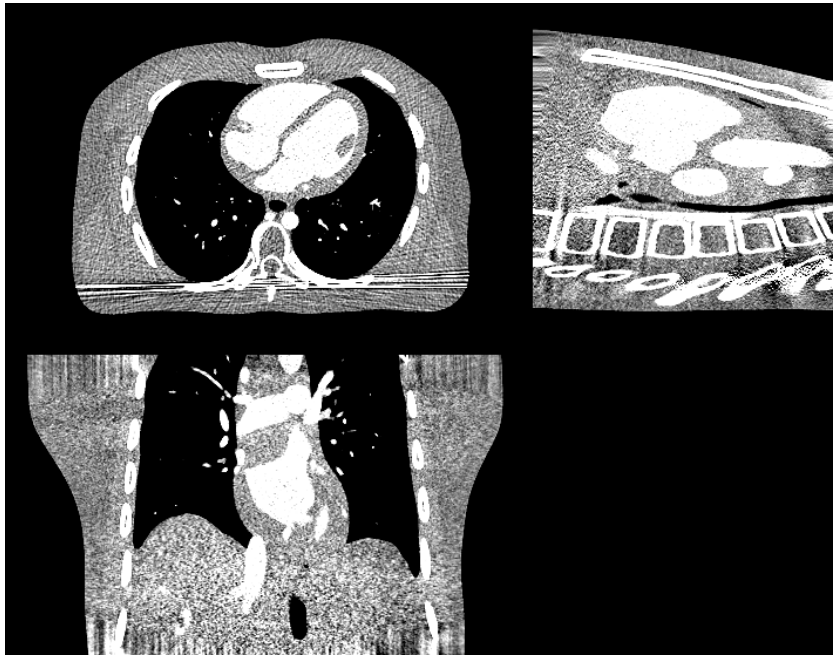
Proposed algorithm after 4.8 min.

Experiment: Helical Shoulder Scan

Algorithm	Time to converge within 5 HU RMSD		Time to converge within 2 HU RMSD	
OGM-1	1290 sec.	21.5 min.	3519 sec.	58.6 min.
OGM-2	702 sec.	11.7 min.	1786 sec.	29.8 min.
OGM-4	430 sec.	7.2 min.	1092 sec.	18.2 min.
Proposed-1	565 sec.	9.4 min.	973 sec.	16.2 min.
Proposed-2	332 sec.	5.5 min.	587 sec.	9.8 min.
Proposed-4	229 sec.	3.8 min.	408 sec.	6.8 min.

Experiment: Wide-cone Axial Simulation

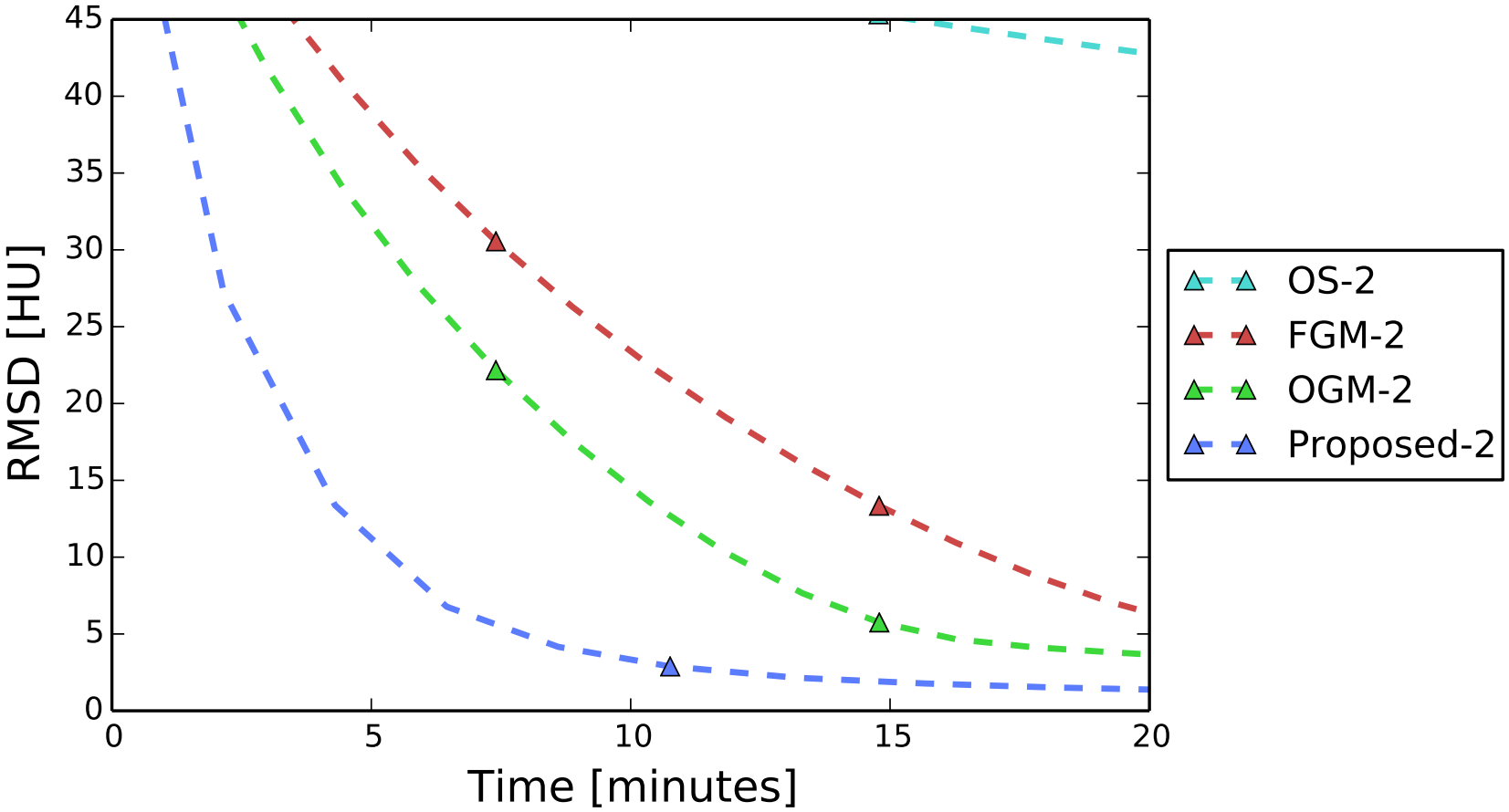
- 888 channels, 256 rows, 984 views
- Reconstructed on $720 \times 720 \times 440$-pixel grid
- Too large for one GPU!



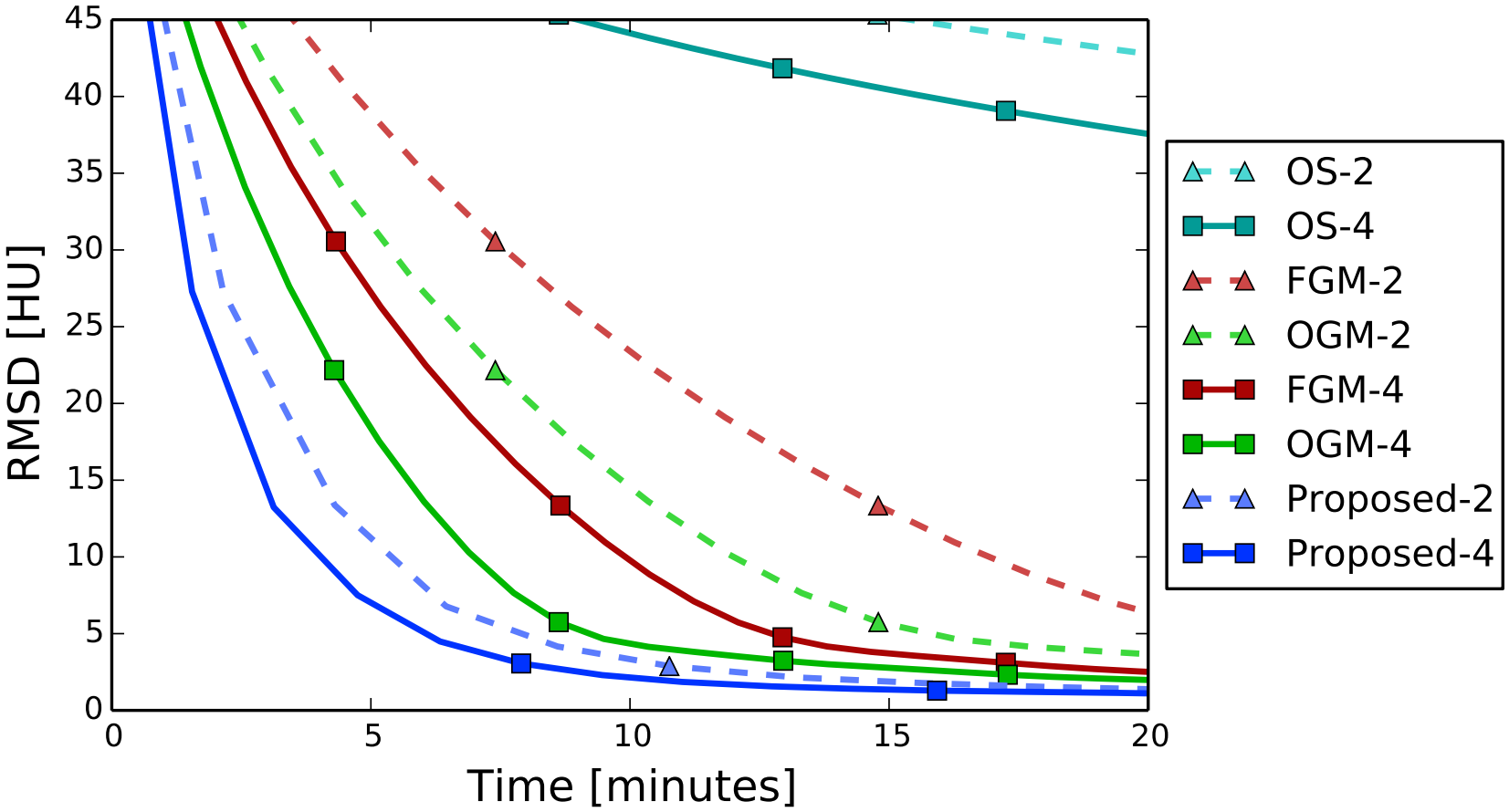
(a) Filtered backprojection

(b) Reference

Experiment: Wide-cone Axial Simulation



Experiment: Wide-cone Axial Simulation



Experiment: Wide-cone Axial Simulation

OS-OGM after 5.2 min.

Experiment: Wide-cone Axial Simulation

Proposed algorithm after 4.7 min.

Experiment: Wide-cone Axial Simulation

Algorithm	Time to converge within 5 HU RMSD		Time to converge within 2 HU RMSD	
OGM-2	976 sec.	16.3 min.	–	–
OGM-4	569 sec.	9.5 min.	1193 sec.	19.9 min.
Proposed-2	516 sec.	8.6 min.	939 sec.	15.6 min.
Proposed-4	380 sec.	6.3 min.	661 sec.	11.0 min.

Conclusions

In this talk:

- Duality to turn coupled cost function terms into variables.
- Group coordinate ascent to exploit structure of different parts of the cost function.
- Parallel operations to get the most out of the GPU.
- Interleave computation- and memory-intensive operations to "hide" the cost of memory transfers.

Conclusions

In this talk:

- Duality to turn coupled cost function terms into variables.
- Group coordinate ascent to exploit structure of different parts of the cost function.
- Parallel operations to get the most out of the GPU.
- Interleave computation- and memory-intensive operations to "hide" the cost of memory transfers.

Thank you!

Duality Approach

$$\mathbf{u}^{(n+1)}, \mathbf{v}^{(n+1)}, \mathbf{z}^{(n+1)} \approx \arg\max_{\mathbf{u},\mathbf{v},\mathbf{z}} -\frac{1}{2\mu}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z}||_2^2$$

$$- \mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) - \mathsf{I}^*(\mathbf{z}) + (\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{z})^\mathsf{T}\mathbf{x}^{(n)}$$

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{\mu}\left(\mathbf{A}^\mathsf{T}\mathbf{u}^{(n+1)} + \mathbf{C}^\mathsf{T}\mathbf{v}^{(n+1)} + \mathbf{z}^{(n+1)}\right)$$

$$\mathsf{L}^*(\mathbf{p}) = \sum_{i=1}^{M} l_i^*(p_i); \qquad l_i^*(u_i) = \frac{1}{2w_i}u_i^2 + u_i y_i$$

$$\mathsf{R}^*(\mathbf{v}) = \sum_{k=1}^{K} r_k^*(d_k); \qquad r_k^* \text{ depends on } \psi$$

$$\mathsf{I}^*(\mathbf{z}) = \sum_{j=1}^{N} \iota_j^*(z_j); \qquad \iota_j^*\text{: characteristic function on } (-\infty, 0]$$