REAL-TIME VISUALIZATION OF 3D MEDICAL SCAN DATA

Master’s Thesis Presentation

Advisors: Don Greenberg, Alex Vladimirsky

May 5, 2010

Konstantin Shkurko

Overview

- motivation
- previous work
- introduction
  - CT data
  - illumination model
  - basic idea
  - transfer function

Overview

- motivation
- previous work
- introduction
- accelerations
  - bricking hierarchy
  - empty space skipping

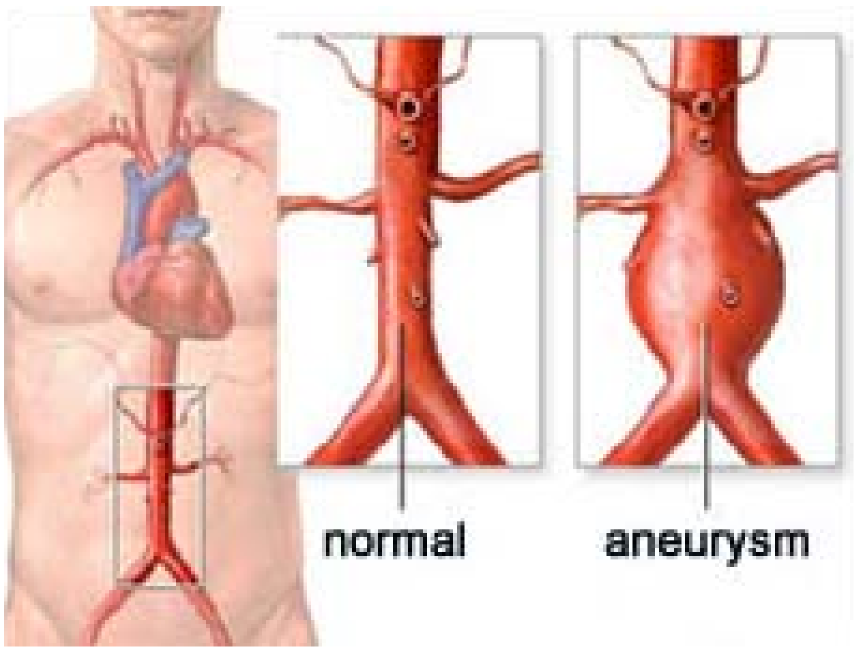
Overview

- motivation
- previous work
- introduction
- accelerations
- parallelizing bricked raycaster
- results
- conclusion

Motivation

- intravascular surgery to treat aortic aneurysms
- patient's CT scan needed to design stent
  - takes hours to scan and reconstruct 3D data
- fluoroscope – real-time X-ray machine used to precisely position stent
  - radioactive dye used to mark blood vessels

Motivation



http://www.westondiagnostics.com/images/Preventative/preventaorta.jpg

http://upload.wikimedia.org/wikipedia/en/4/44/Covered_Stent_Graft.jpg

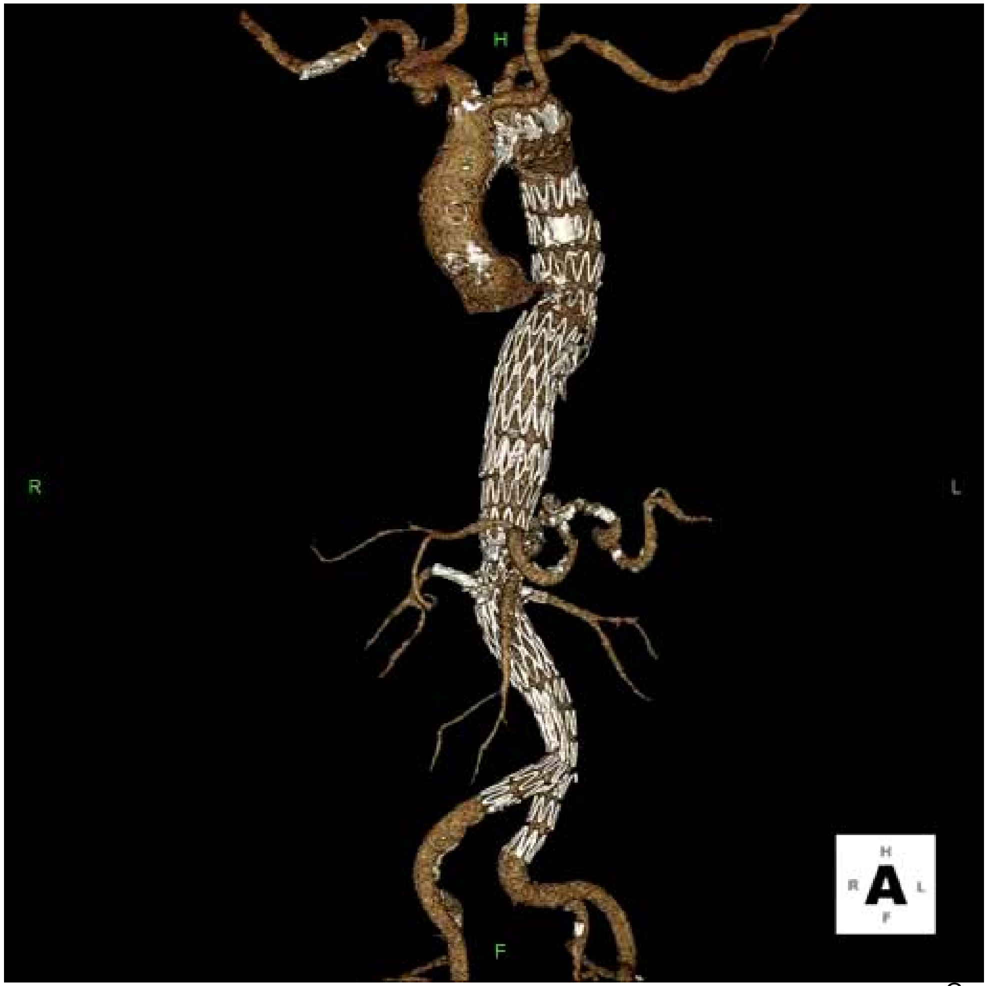
Motivation

- pre-operation video

Courtesy: Ron Greenberg, Cleveland Clinic

Motivation

- post-operation video



Courtesy: Ron Greenberg, Cleveland Clinic

Motivation

- raw data rendering desired by doctors
  - data approximation inhibits diagnosis
  - state of the art is surface-based (typically triangles)
- cost of hardware is extremely high
  - $60,000-$240,000 for VolumePro 1000 workstation[1]
  - use limited to one workstation
- operating room use will improve intravascular surgery
  - use visualization to approximate output of fluoroscope
  - decreases exposure of patient to radiation

1. http://imaging-radiation-oncology.advanceweb.com/sharedresources/advanceforioa/resources/DownloadableResources/AR110103_p120chartsmart.pdf

Previous Work

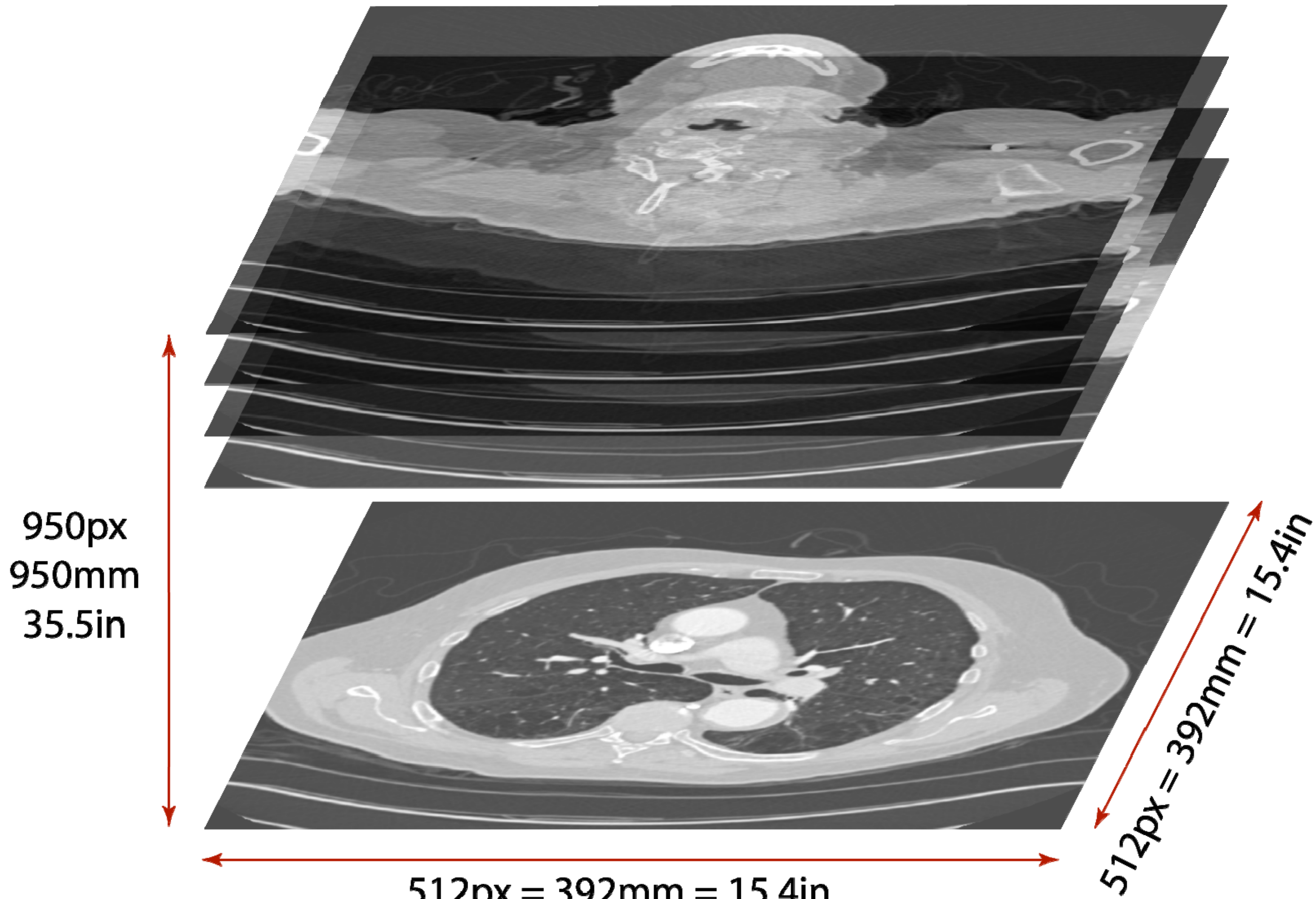
- Marching Cubes [Lorensen, Cline ‘87]
- Raycasting [Levoy ’88] [Levoy ‘90]
- Shear-Warp [Lacroute, Levoy ‘94] [Sweeney, Mueller ‘02]
- Splatting [Westover ’90] [Mueller, Crawfis ‘98] [Zwicker et. al. ‘04]
- Special Hardware [Kruger, Westermann ’03]
  - Cube [Kreeger, Kaufman ‘99]
  - Vizard [Knittel, Strasser ‘97] [Meissner et. al. ‘02]
  - VolumePro [Pfister et. al. ‘99]

Special Hardware: VolumePro

- benefits (VolumePro 1000)
  - very powerful
  - up to 10-15fps for 3D data sets (1st hit raycasting)
  - FDA approved
- problems
  - very expensive
  - algorithms are not easily extended
  - hardware itself not easily extended

http://www.terarecon.com/products/vp_prod_med.htm

Introduction: CT Dataset



Introduction: CT Dataset

- Hounsfield Units, HU
  - CT unit of measure
  - range: [-1024, 3071] or [0, 4095]
  - air (-1000), fat (-50), water (0), bone (1000+)
- currently
  - 512 x 512 pixels per slice
  - 12 bits per voxel grayscale data
- future
  - 1024 x 1024 pixels per slice

Introduction: CT Dataset

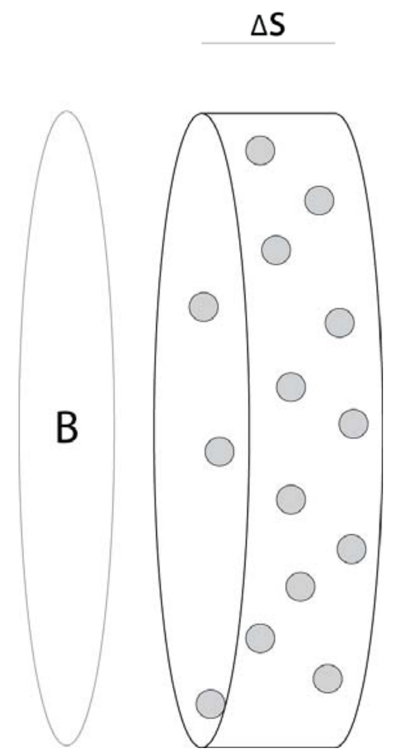
- dimensions
  - 15.4 x 15.4 x 35.5 in
  - 512 x 512 x 950 voxels
- memory requirements

	data (2B per voxel)	gradients (12B per voxel)
512^2 x 950	475 MB	2.78 GB
1024^2 x 950	1.86 GB	11.13 GB

Introduction: Illumination Model

- low-albedo formulation
  - homogeneous particle distribution
  - only single scattering
  - emission and absorption
- differential eq for light intensity

$$\frac{dI}{ds} = g(s) - \tau(s) I(s)$$

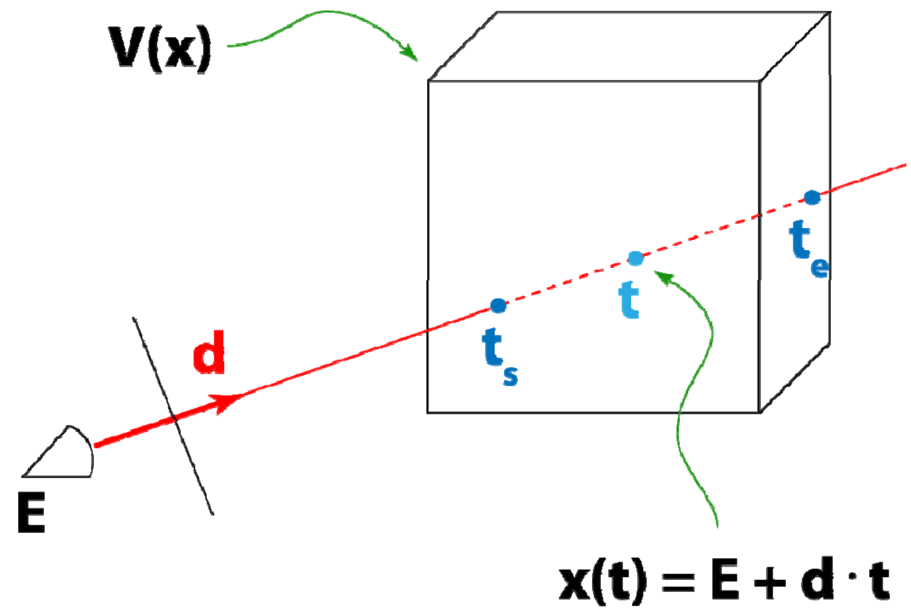


- volume rendering integral

$$I(L) = I_0 T(0, L) + \int_0^L g(s) T(s, L)\, ds \qquad T(s_1, s_2) = \exp\left(- \int_{s_1}^{s_2} \tau(t)\, dt \right)$$

Introduction: Basic Idea

- consider dataset as a volumetric material
- simple front-to-back raycaster



  - V(**x**) – 3D data
  - **E** – eye location
  - **d** – ray's direction
  - t_s , t_e – start, end intersections of ray with volume
  - **x**(t) – sample location inside volume t units along ray

Introduction: Basic Idea

- output color:

$$C_{out} = \int_{t_e}^{t_s} C\big(V(s)\big)\tau\big(V(s)\big)T(s,t_s)\,ds \qquad T(s_1,s_2) = \exp\left(-\int_{s_1}^{s_2} \tau\big(V(t)\big)\,dt\right)$$

$$\approx \sum_{i=1}^{n} C_i\alpha_i \prod_{j=i+1}^{n} (1-\alpha_j) \qquad \alpha_i = 1 - t_i \approx 1 - \exp\big(-\tau(i\triangle s)\triangle s\big)$$

  - where

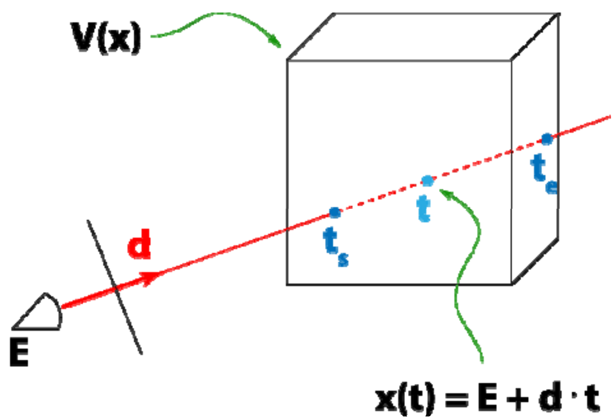
    $V(s) = V\big(\mathbf{x}(s)\big)$

  - transfer functions

    $C : [0,\ 4095] \to [0,\ 255]^3$ - color

    $\alpha : [0,\ 4095] \to [0,\ 1]$ - opacity

  - n – ray steps between t_s and t_e



Introduction: Transfer Function

- responsible for assigning color and opacity to a data sample
- provided by the user

Introduction: Transfer Function

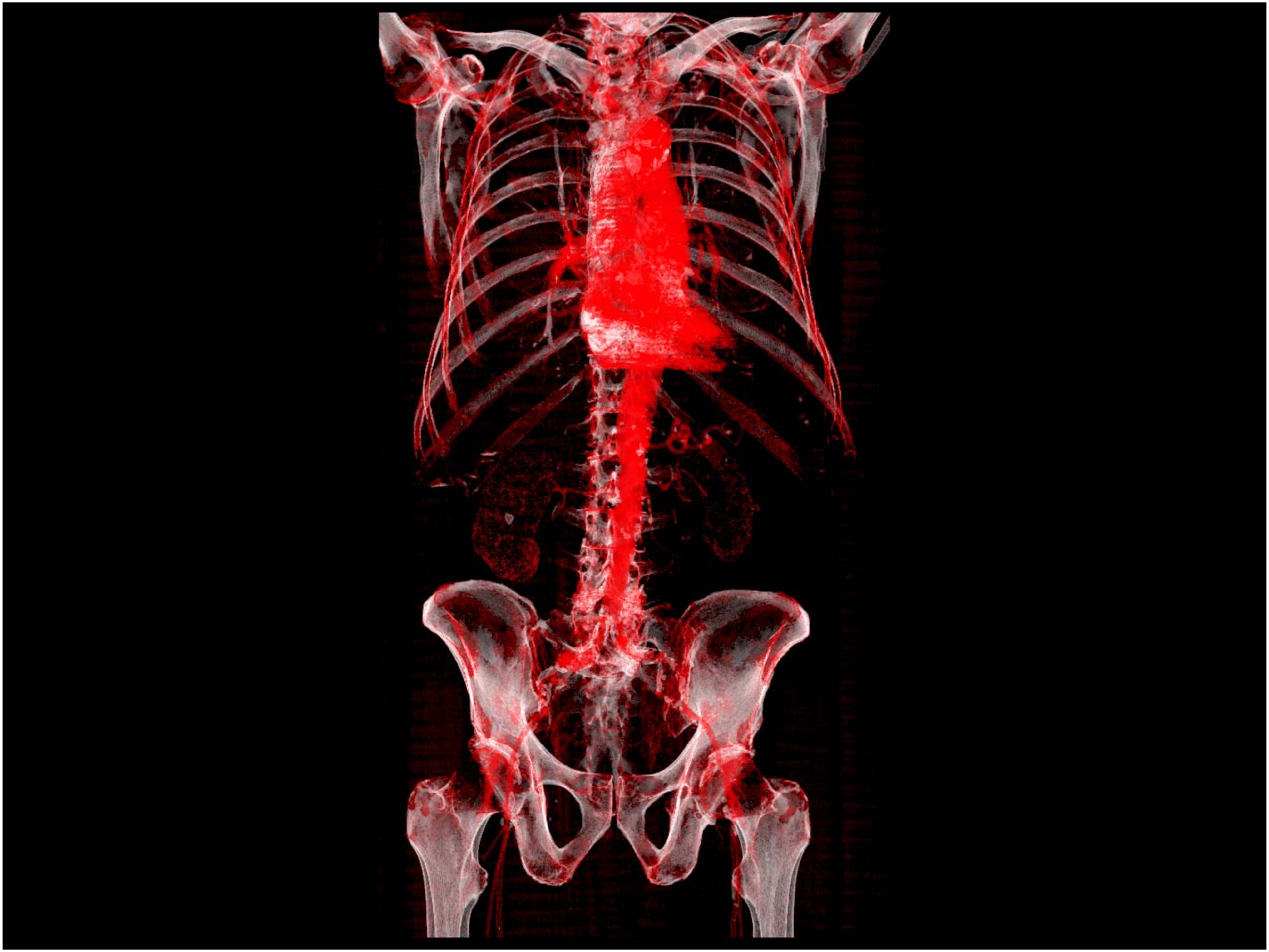
- responsible for assigning color and opacity to a data sample
- provided by the user
- selects emission, diffuse and specular colors

Sample Color, C = **Emission** + **Diffuse Reflection** + **Specular Reflection**

From Color TF

Blinn – Phong BRDF

http://upload.wikimedia.org/wikipedia/commons/6/6b/Phong_components_version_4.png

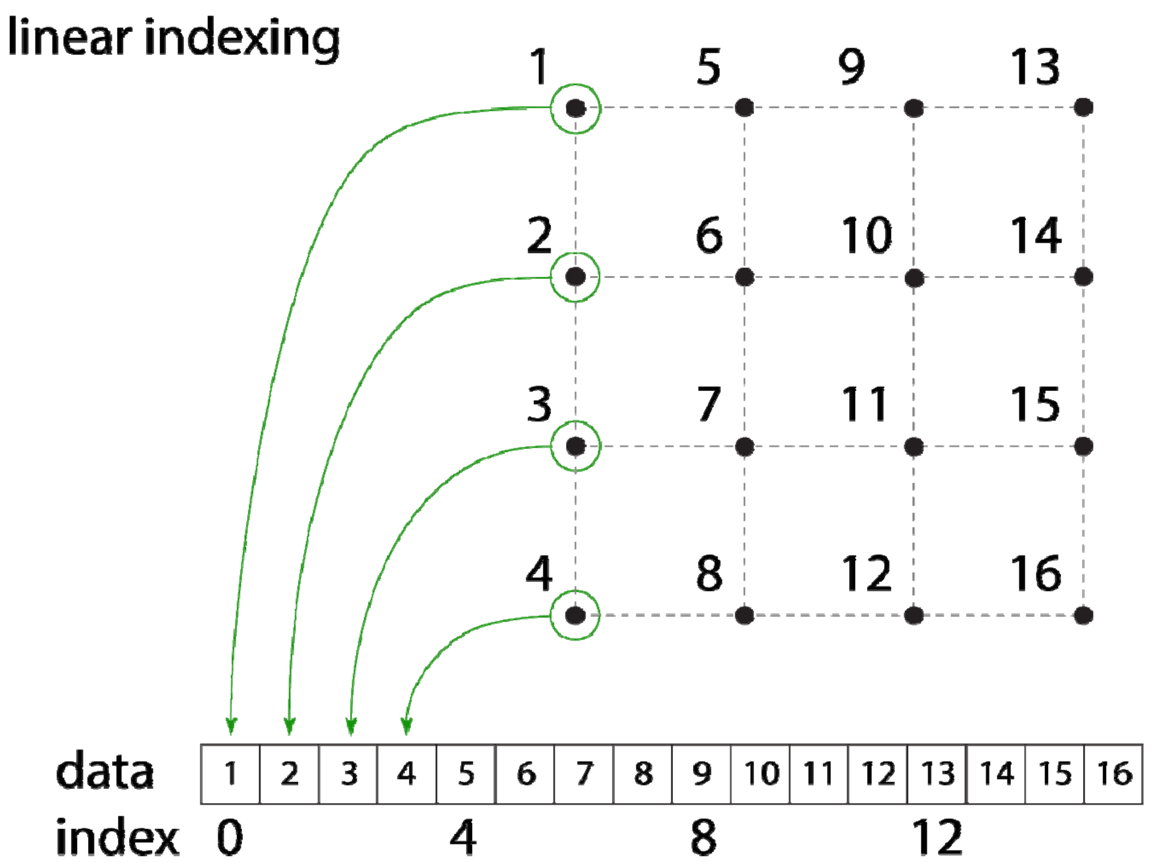


Accelerations

- multiple resolutions
- gradient pre-computation
- early ray termination
- transparent sample skipping
- bricking hierarchy
  - set-up
  - non-linear data storage order
  - local gradient storage
  - empty space skipping

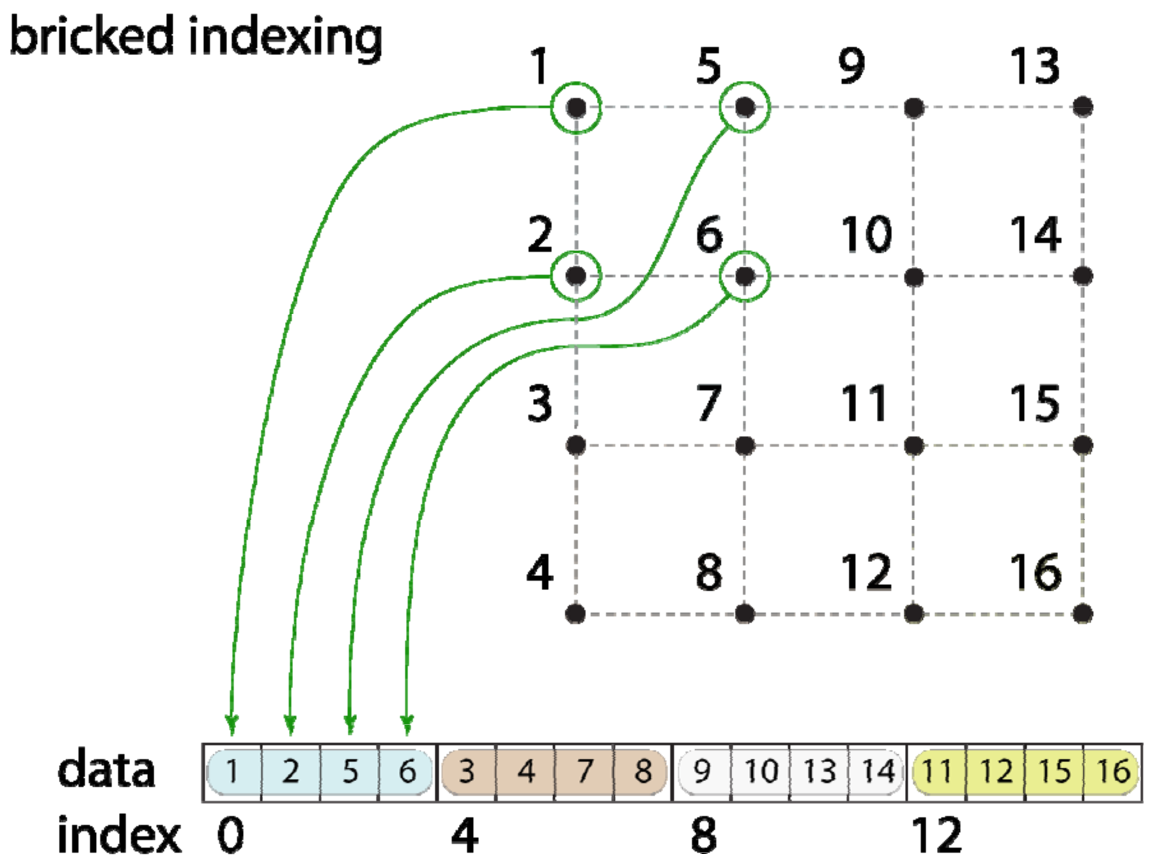
Bricks: Set-up

- 3D voxel data stored in 1D array in XYZ order

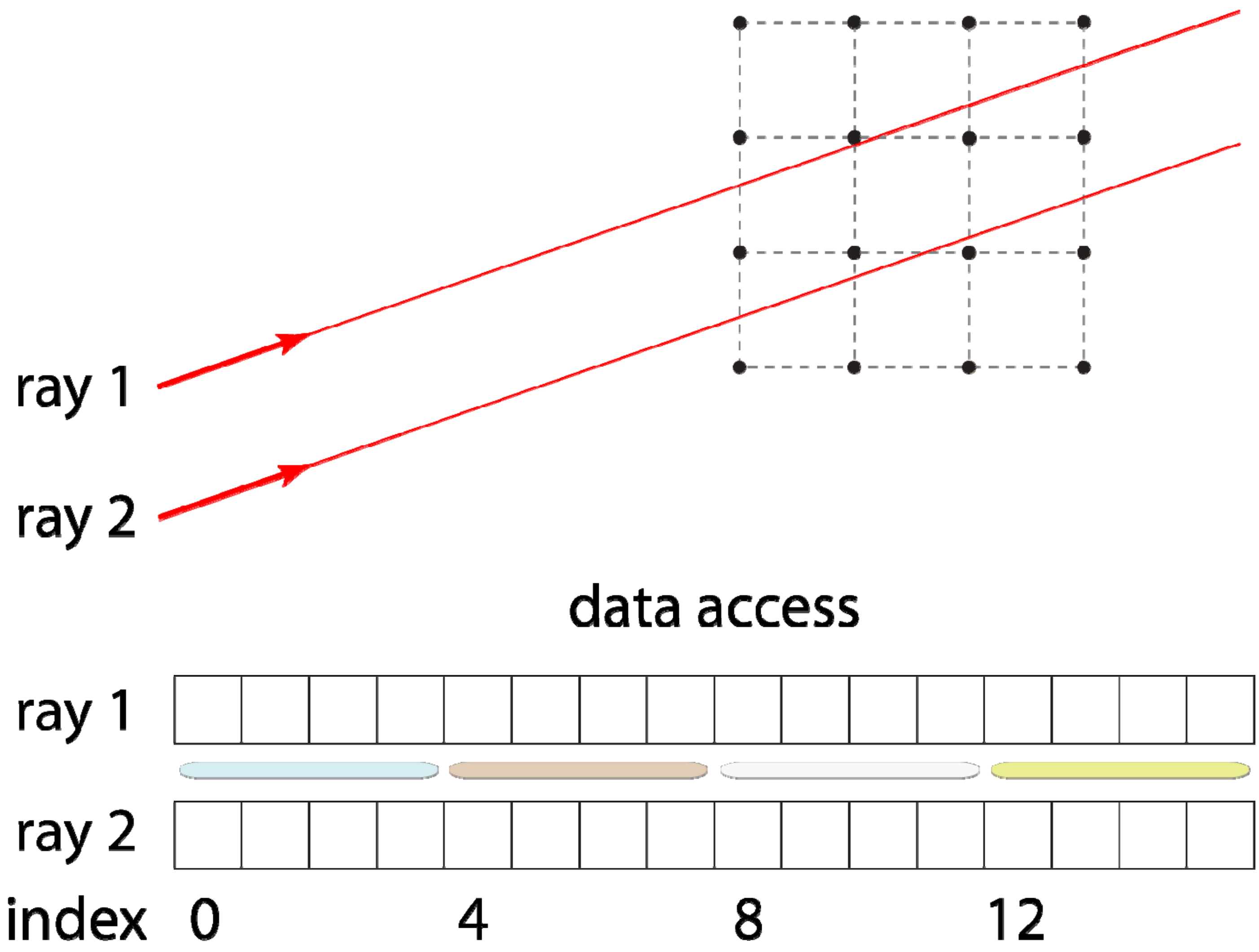


Bricks: Set-up

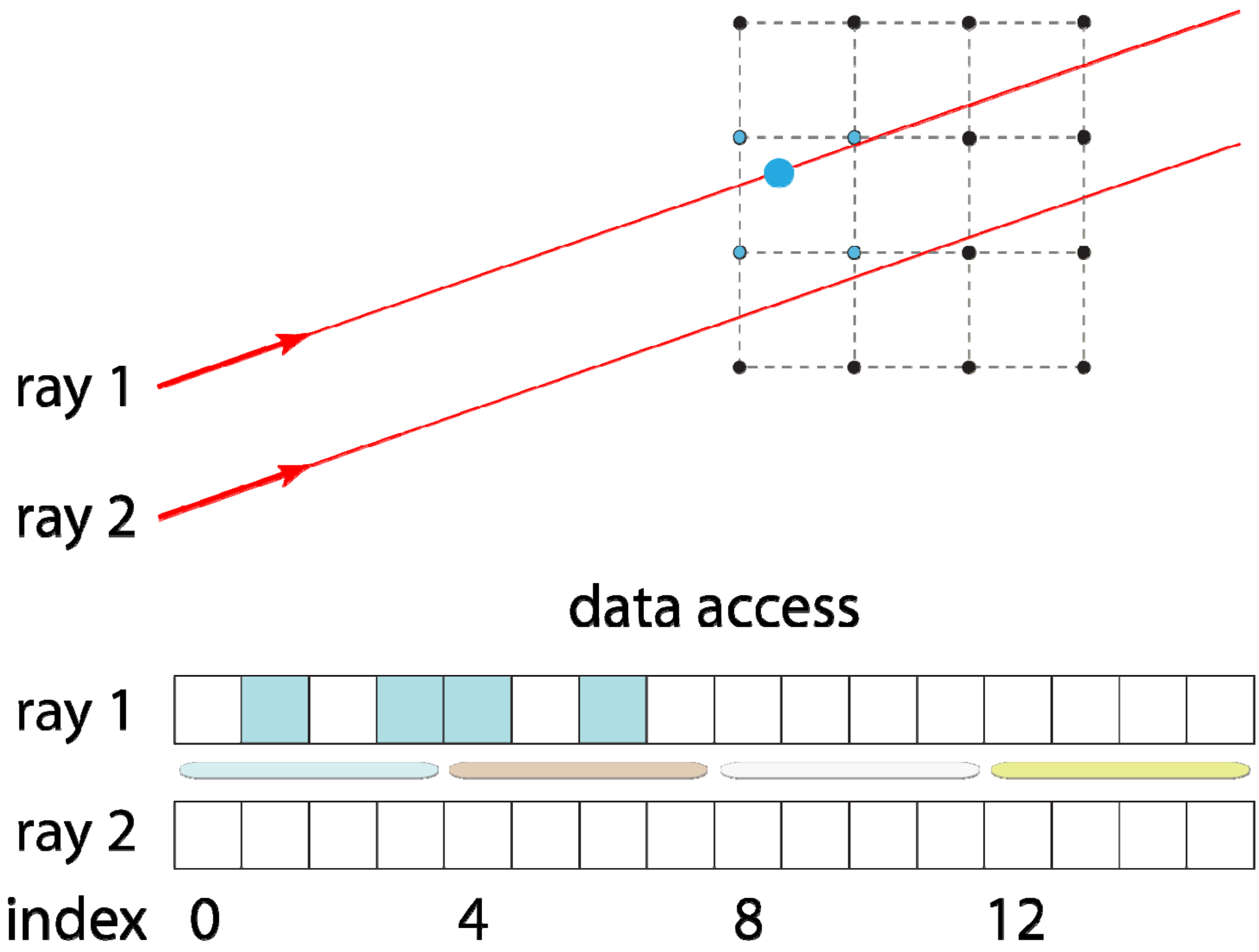
- reorder to XYZ-ordered bricks arranged in XYZ order



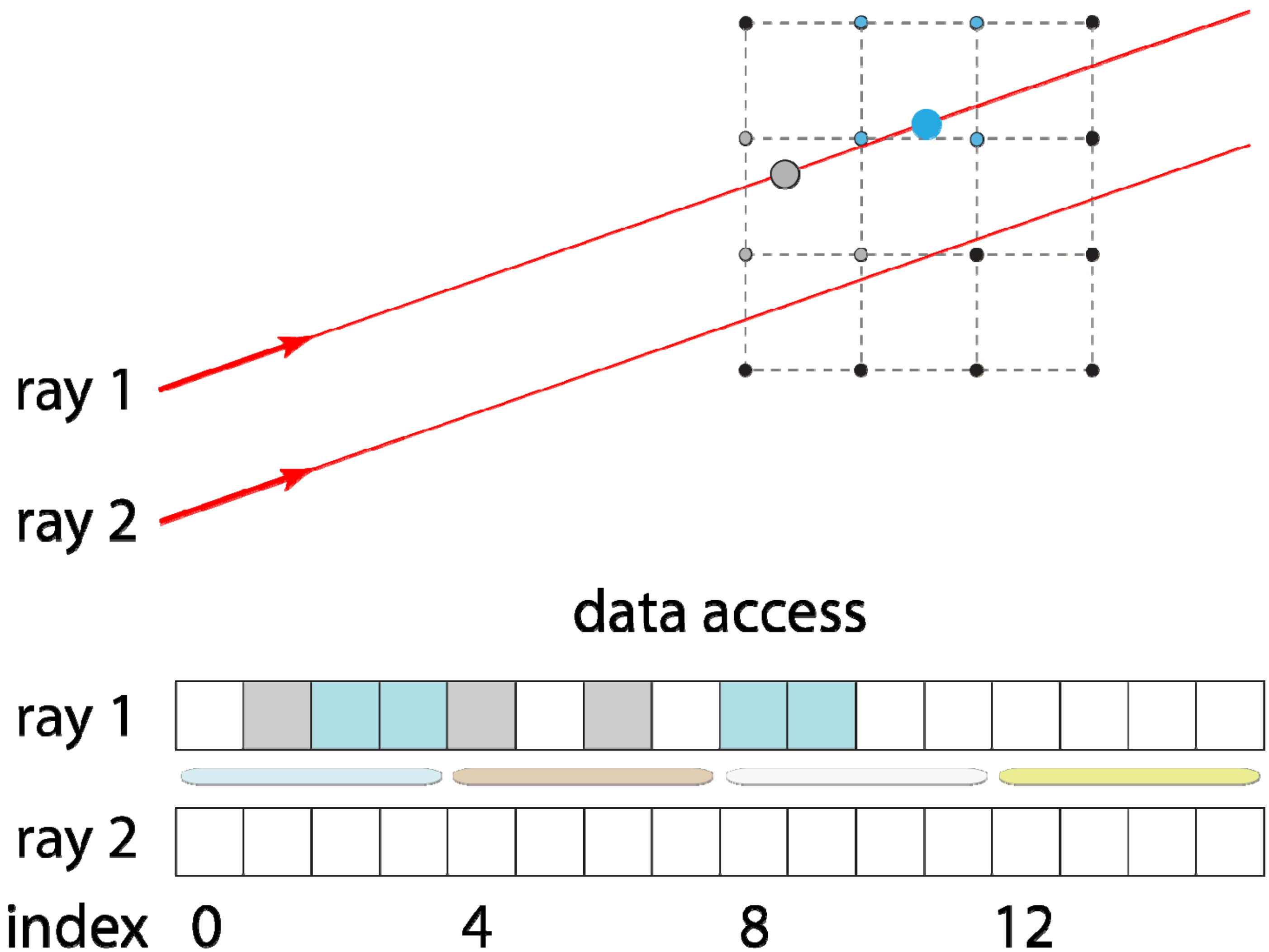
Bricks: Data Access Order



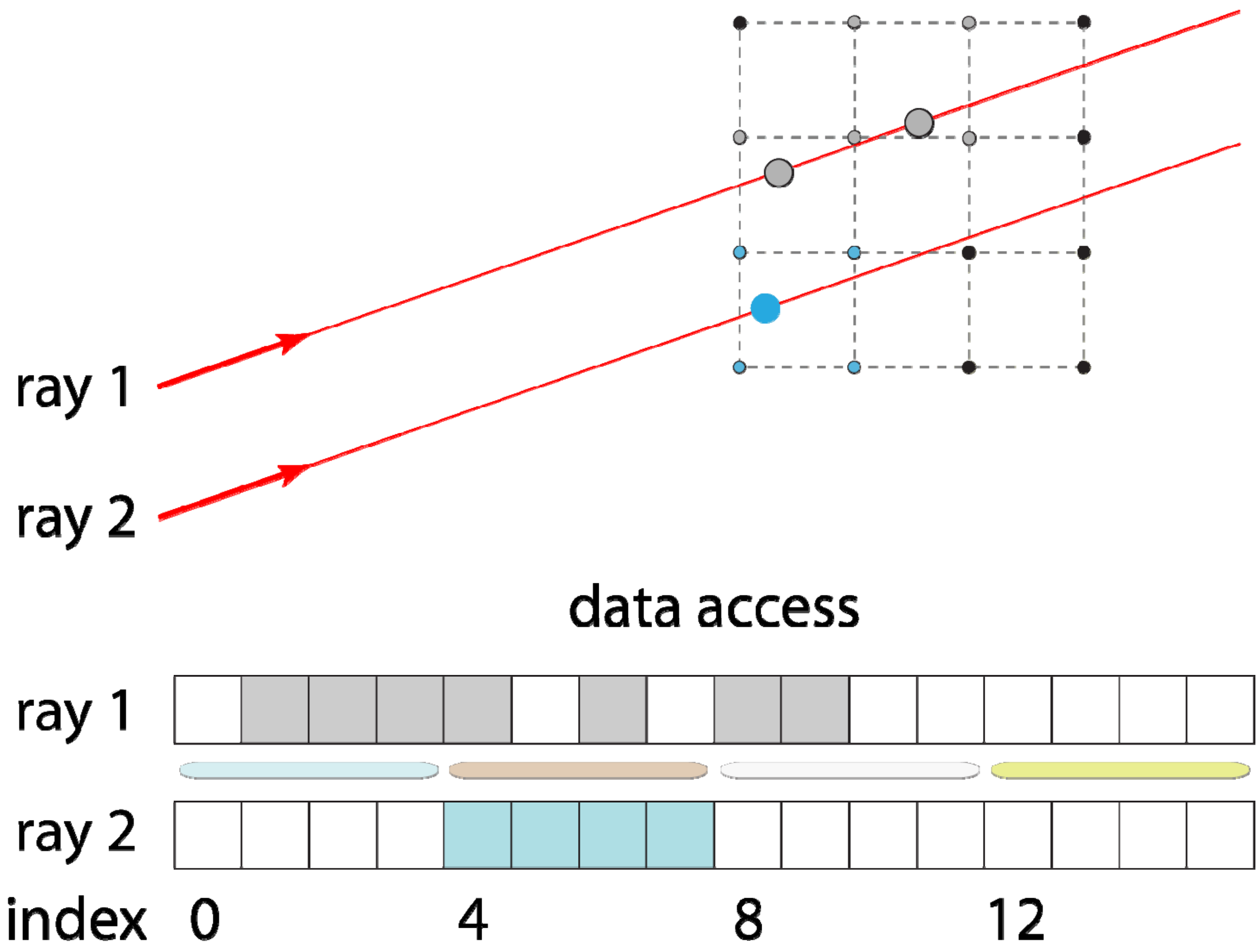
Bricks: Data Access Order



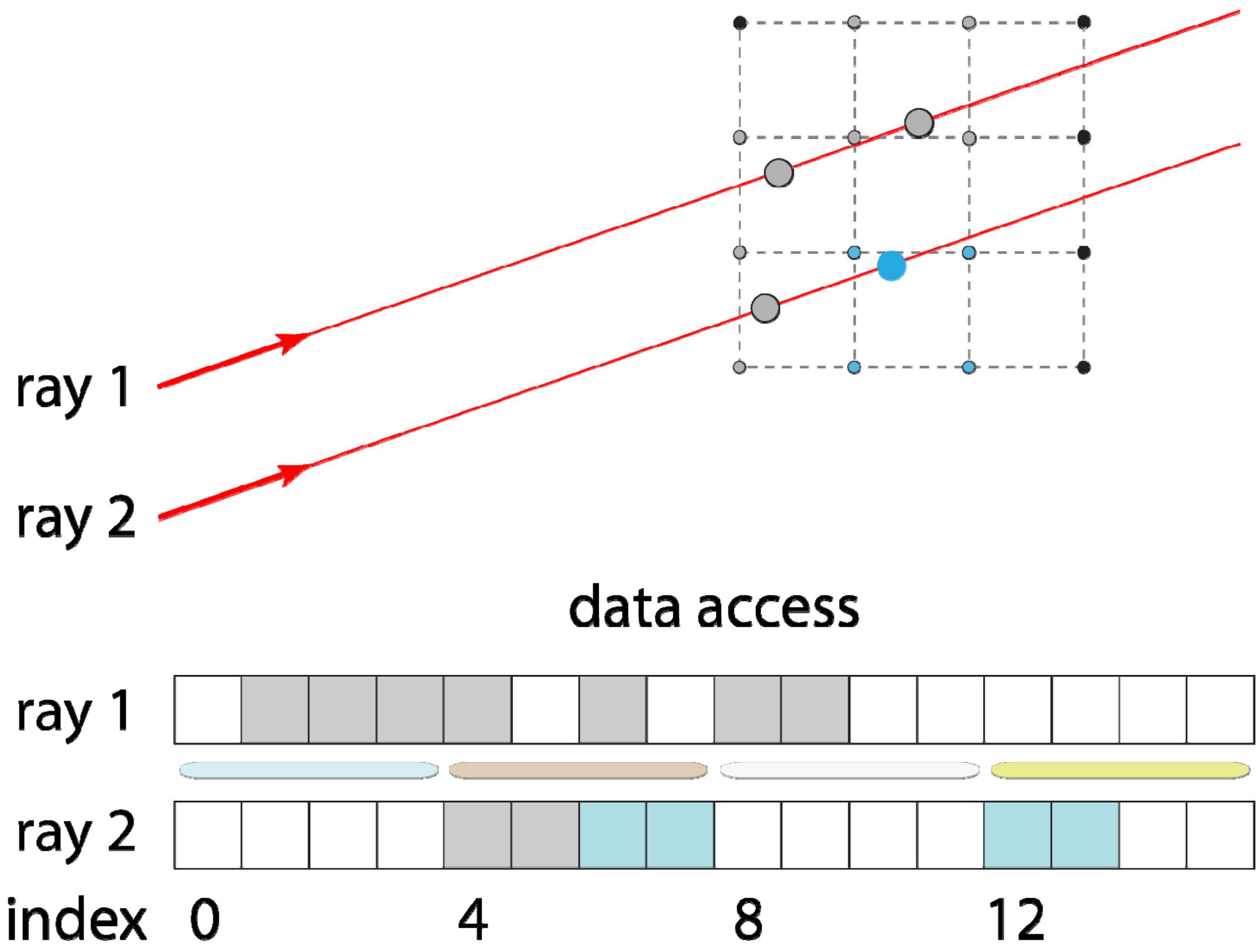
Bricks: Data Access Order



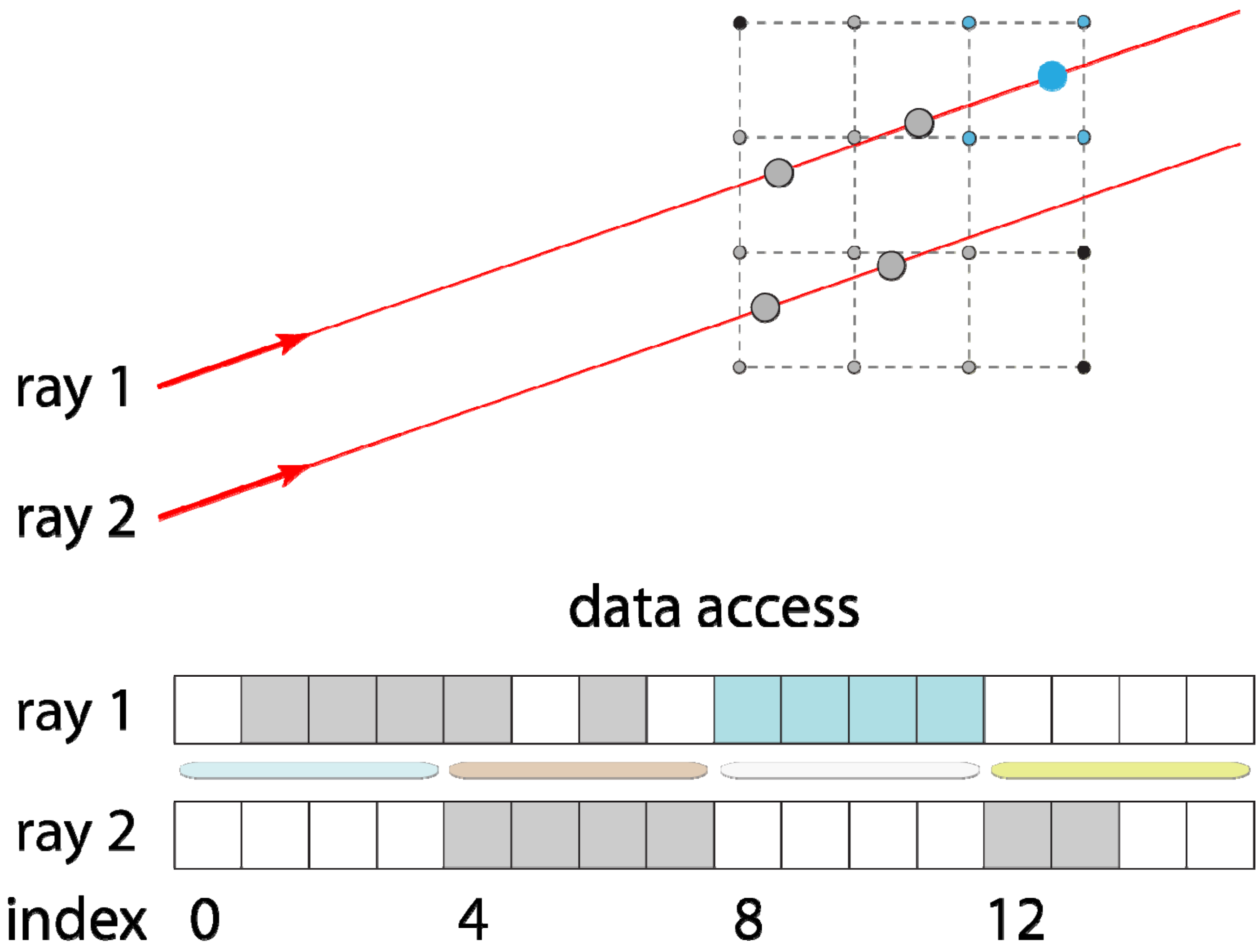
Bricks: Data Access Order



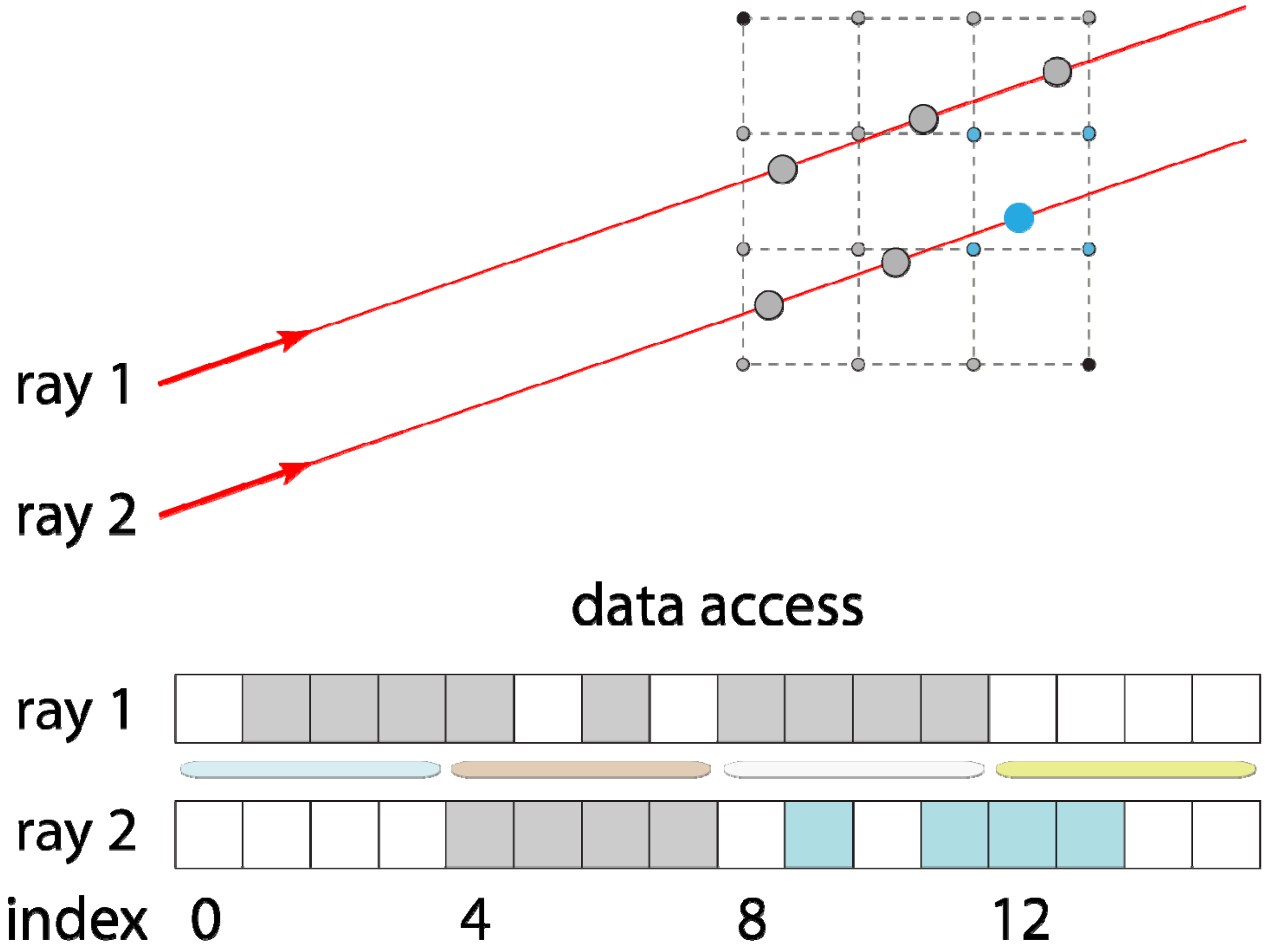
Bricks: Data Access Order



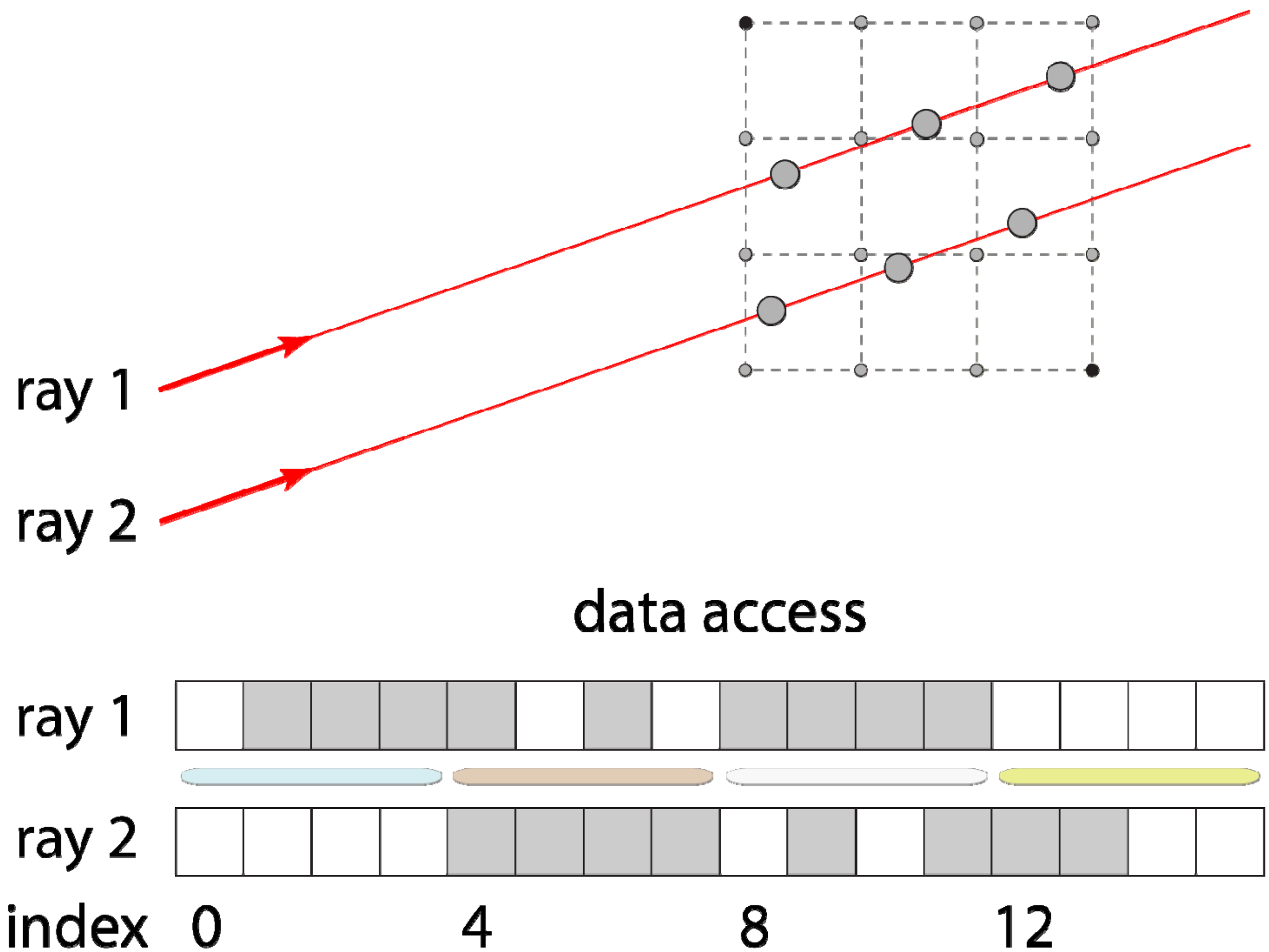
Bricks: Data Access Order



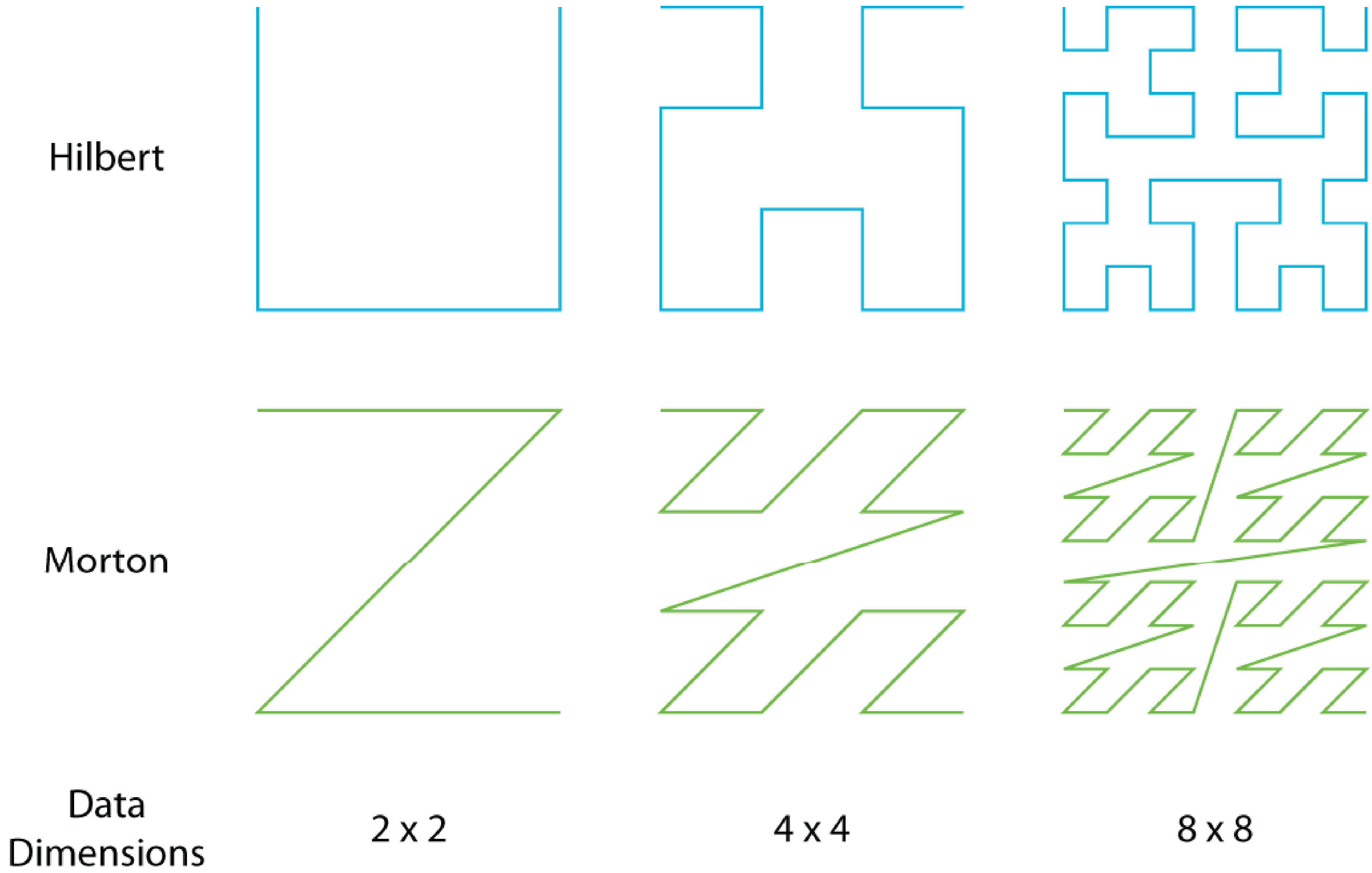
Bricks: Data Access Order



Bricks: Data Access Order



Bricks: Non-linear Data Storage Order



Bricks: Empty Space Skipping

- skip empty space at any level in brick hierarchy via 64-bin binary histograms
  - data inside a brick

$$\sigma_A(i) = \begin{cases} 1, & \exists\, x, \text{s.t. } x \in A,\ x \in \left[64 \cdot i, 64 \cdot (i+1)\right) \\ 0, & otherwise \end{cases}$$

  - opacity transfer function

$$\lambda(i) = \begin{cases} 1, & \exists\, x, \text{s.t. } \alpha_x \neq 0,\ x \in \left[64 \cdot i, 64 \cdot (i+1)\right) \\ 0, & otherwise \end{cases}$$

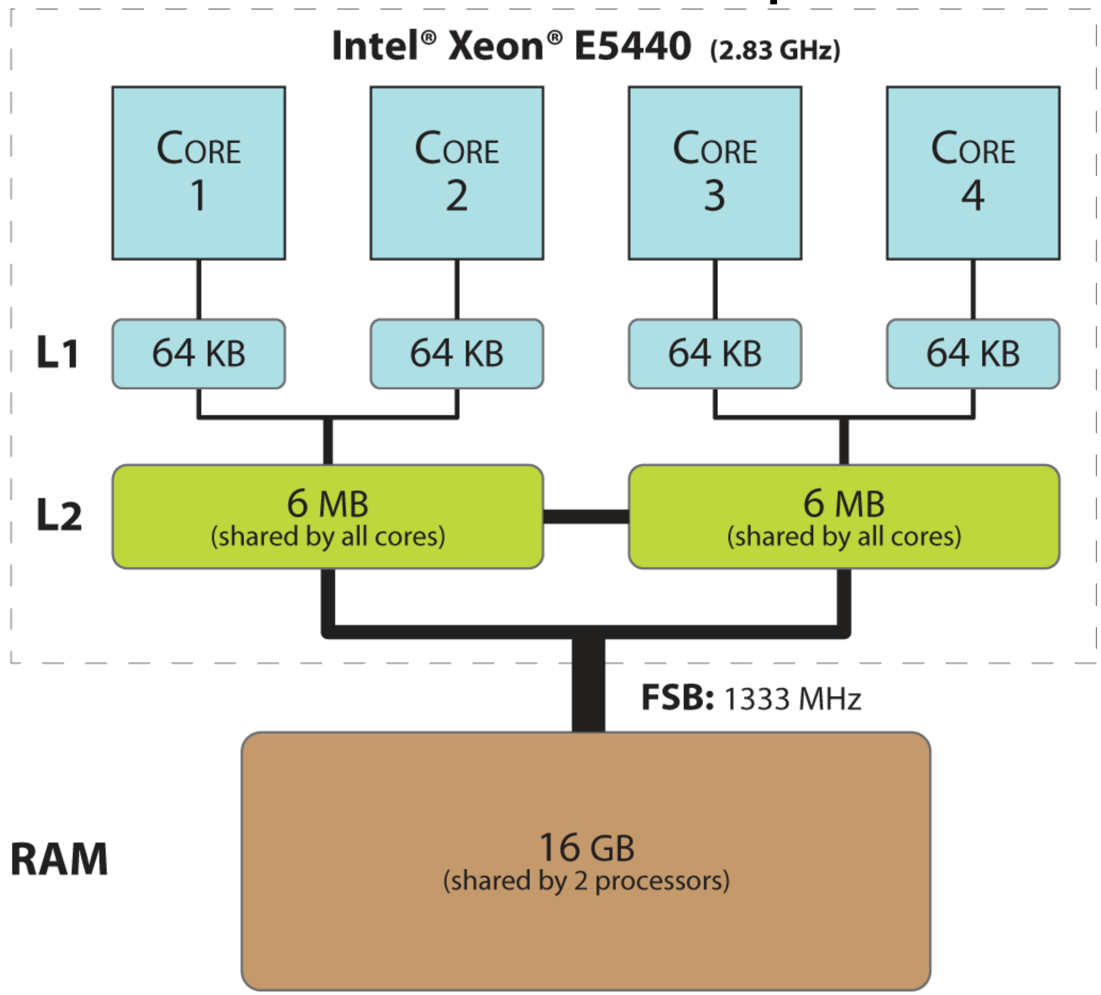
  - combining bins for higher level brick

$$\forall\, i \in [0, 63],\ \sigma_{\bar{A}}(i) = \bigcup_{\forall A_j \subset \bar{A}} \sigma_{A_j}(i)$$

- a brick is transparent if $\forall\, i \in [0, 63],\ \sigma(i) \wedge \lambda(i) = 0$
- adds < 2% extra memory for 1 level hierarchy

Parallelizing Bricked Raycaster

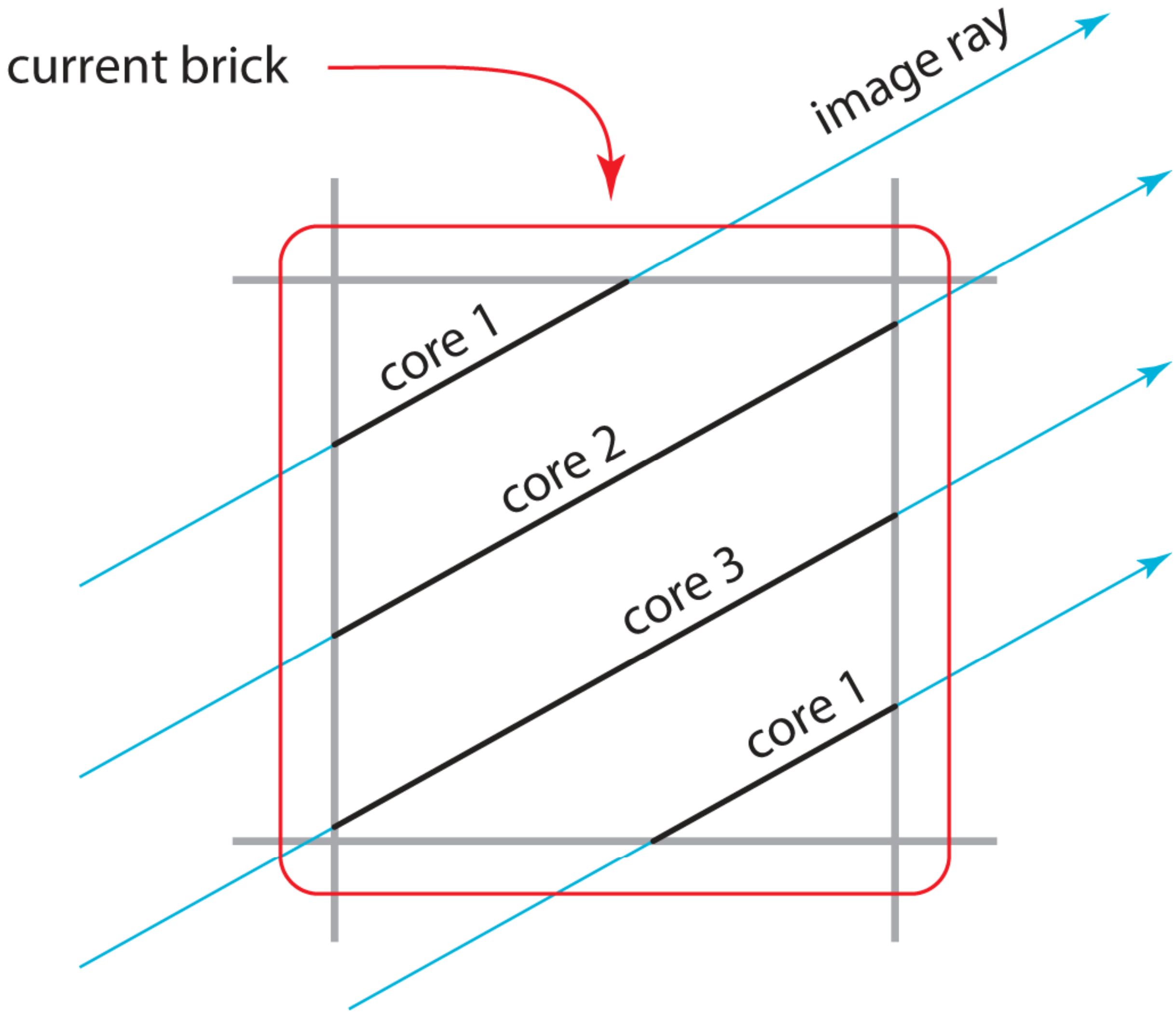
- map to a cluster of 32 dual quad-core nodes



Parallelizing Bricked Raycaster

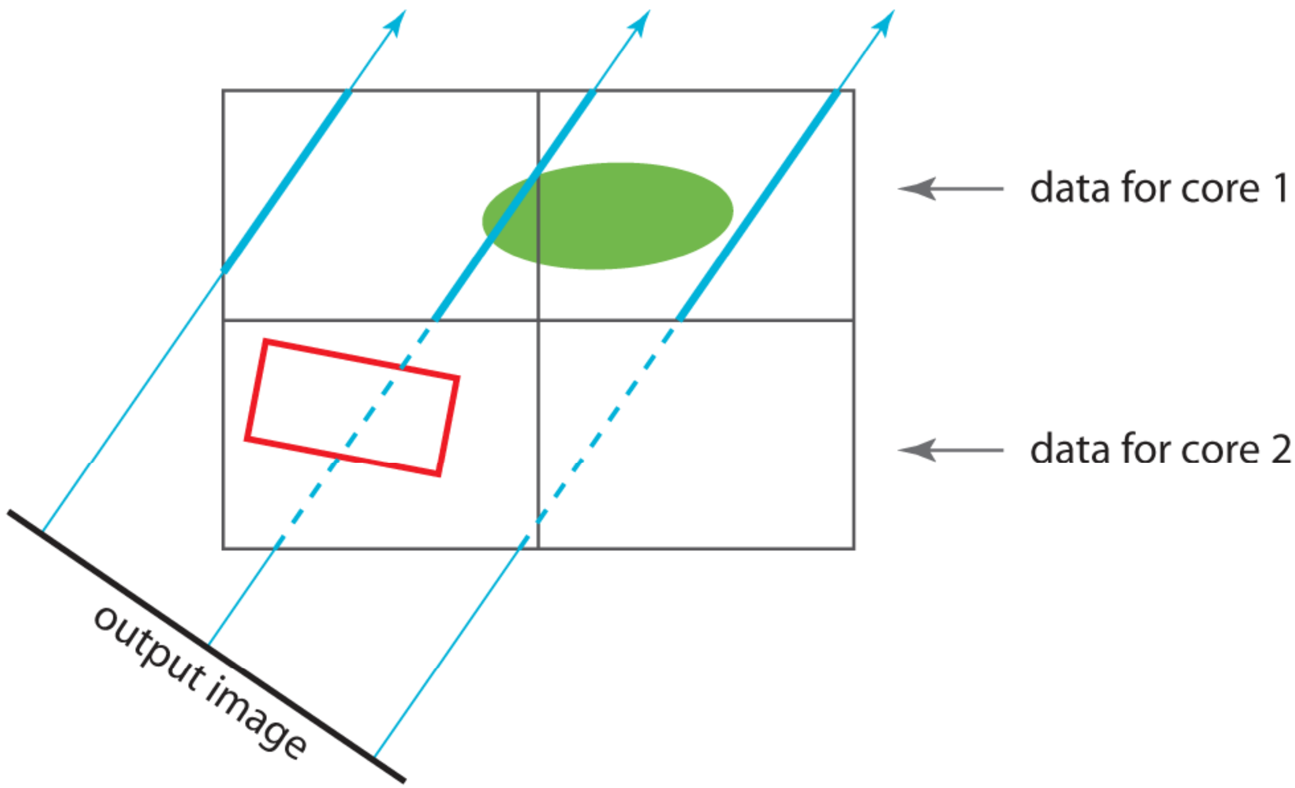
- map to a cluster of 32 dual quad-core nodes
- exploit independence in data and output pixels
- multi-core for one node
  - rays in a brick
  - volume data subdivision
  - output image subdivision
- many nodes
  - output image subdivision

Multi-Core: Rays in a Brick



Multi-Core: Volume Data Subdivision

- split the volume between cores, each rendering the entire output image

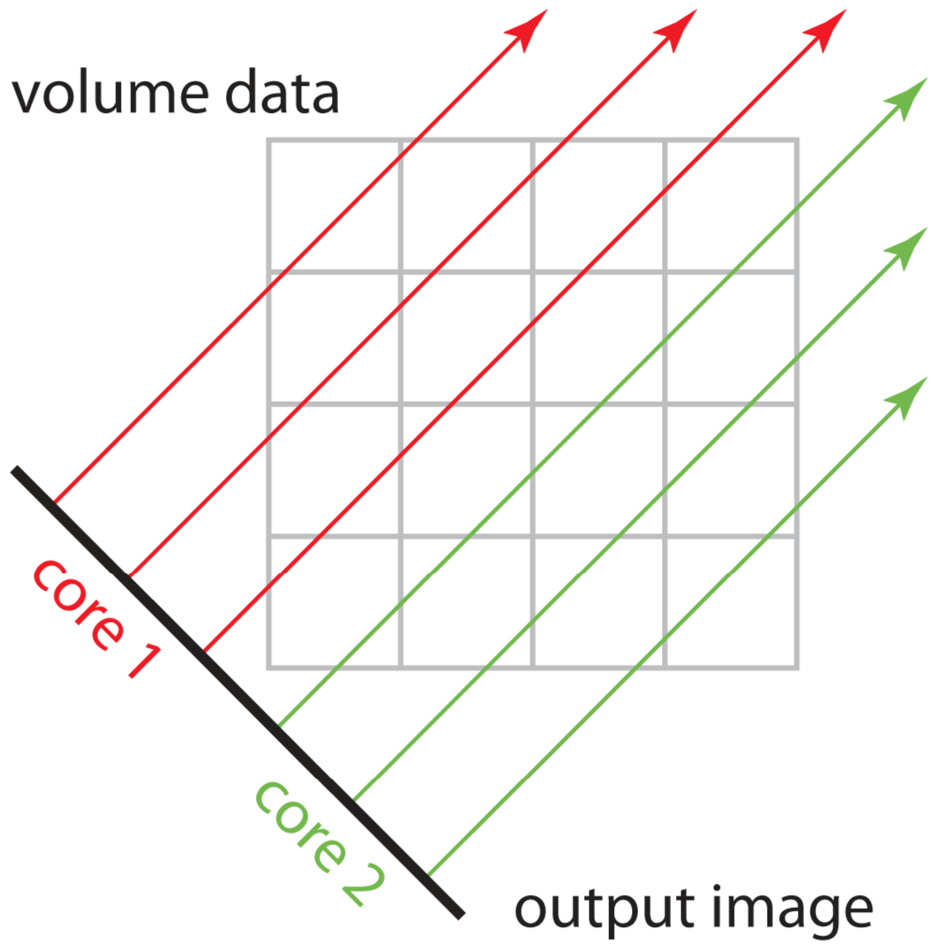


Multi-Core: Volume Data Subdivision

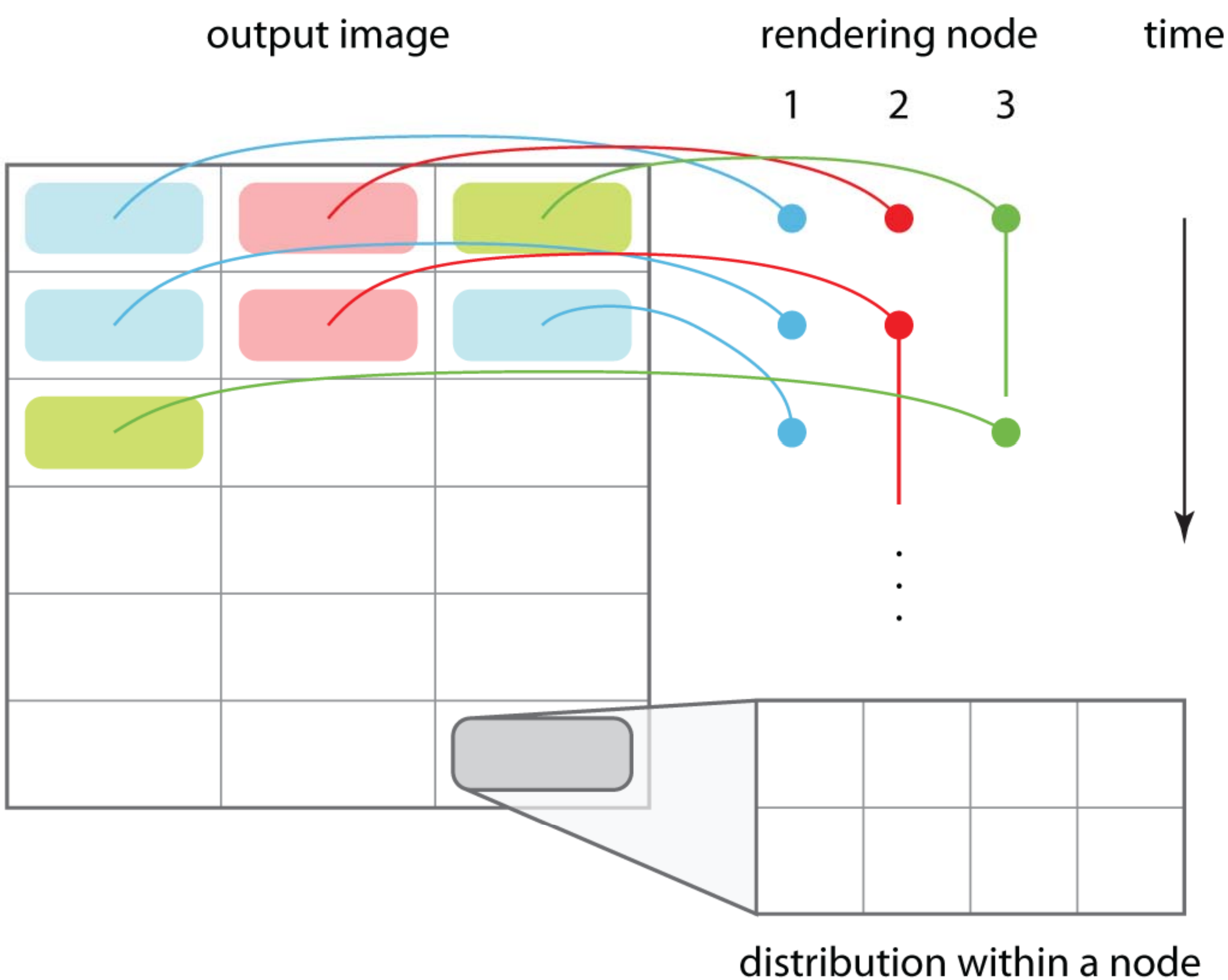
- split the volume between cores, each rendering the entire output image
  - rays pass through volume contiguously
  - distance between ray samples constant across subvolume boundaries
- compositing

Multi-Core: Output Image Subdivision

- each core renders own subimage



Many Nodes: Output Image Subdivision

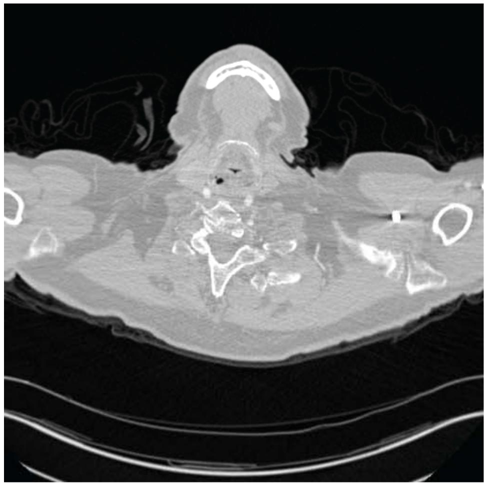


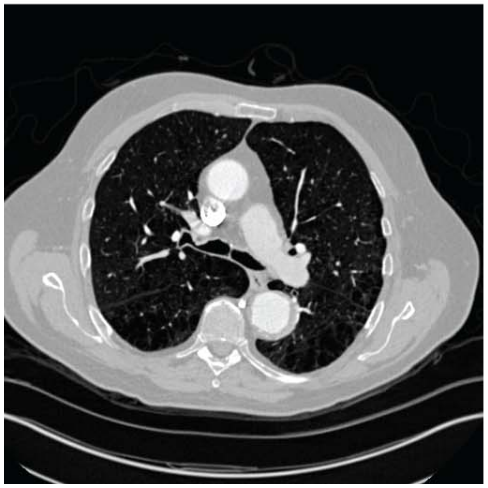
Results

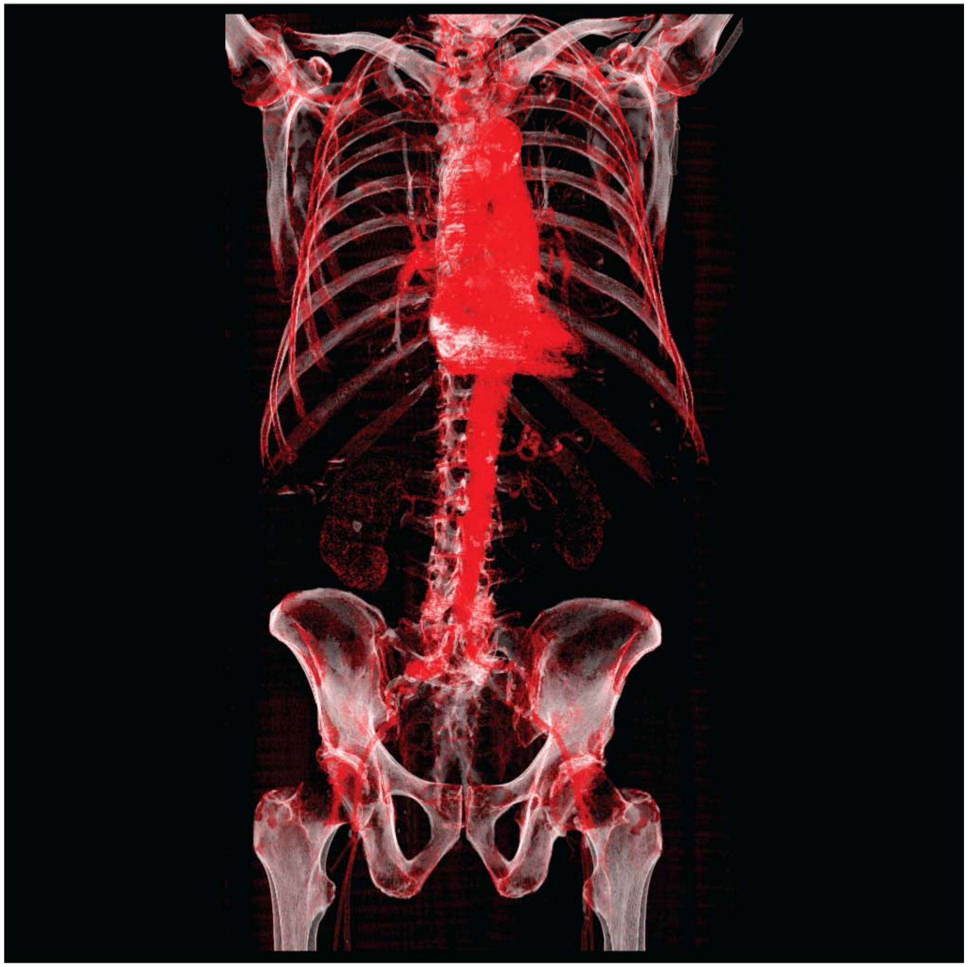
- datasets
- accelerations
  - brick sizes
  - data storage order
  - gradient computation
- scalability
  - number of cores
  - number of nodes

Results: Datasets: Pre-Operation

- 512 x 512 x 928 voxels

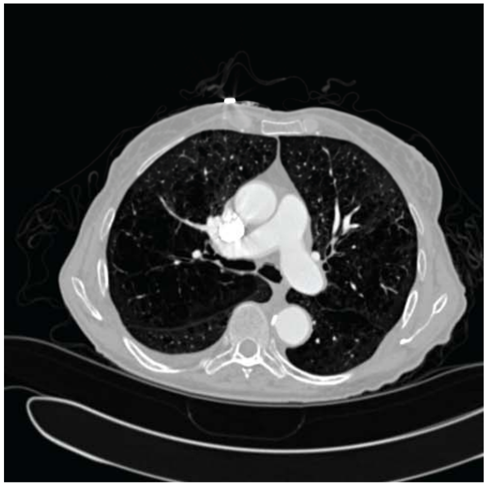


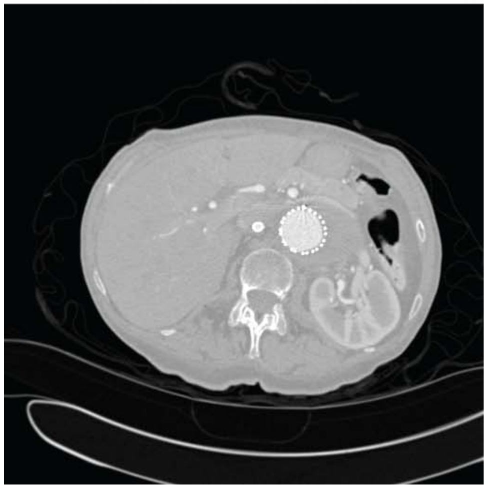


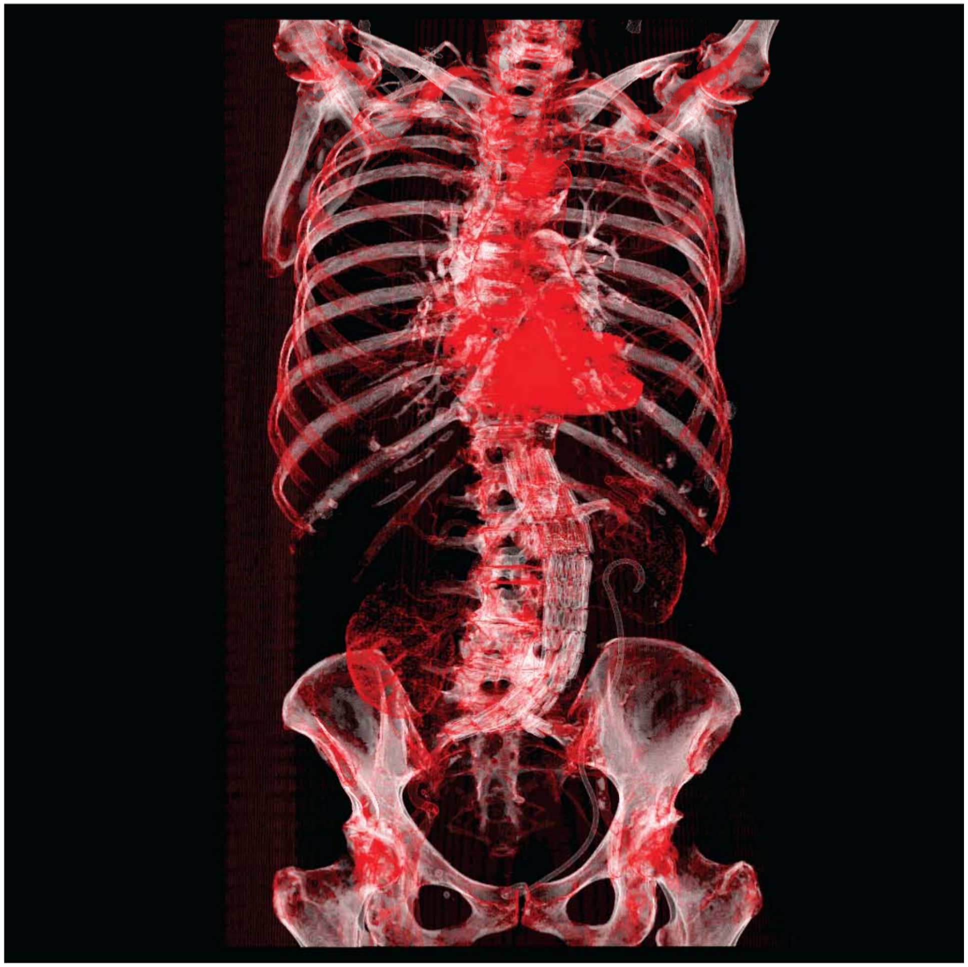


Results: Datasets: Post-Operation

- 512 x 512 x 768 voxels

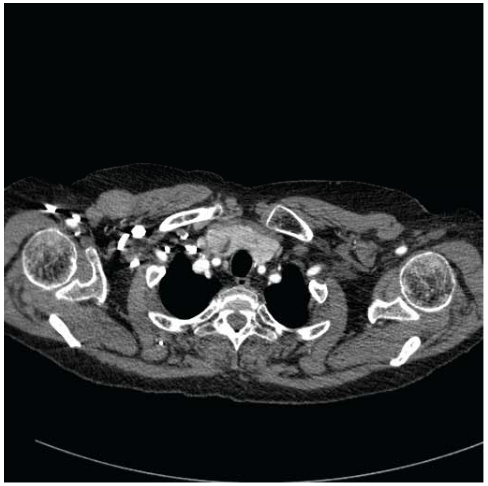


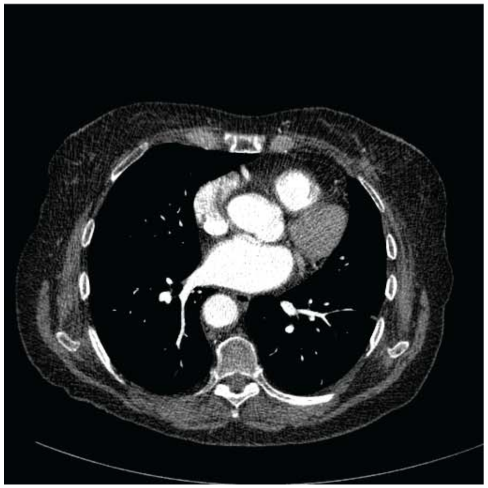


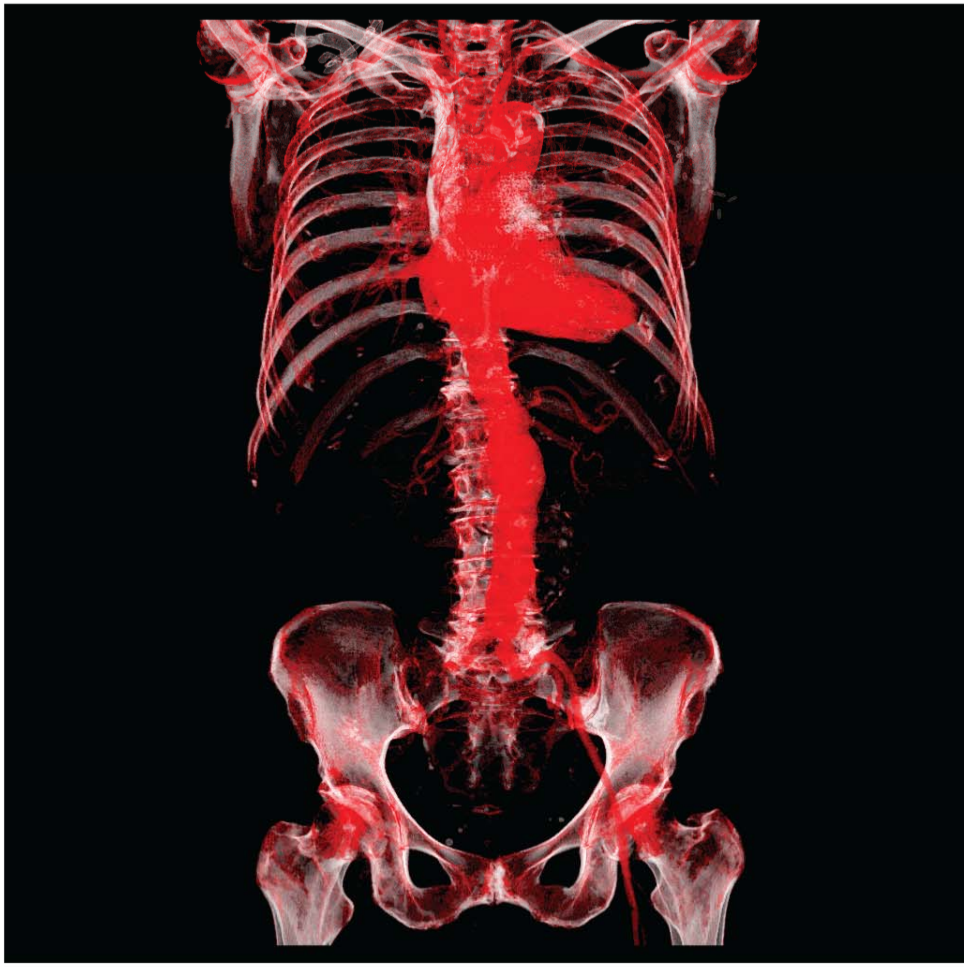


Results: Datasets: CT14

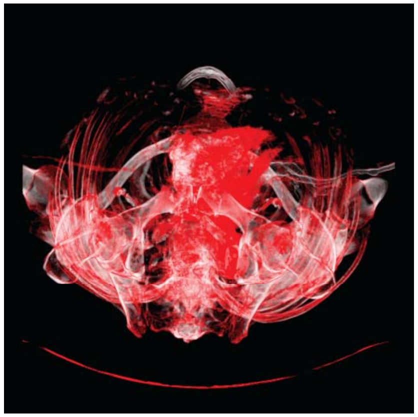
- 512 x 512 x 768 voxels



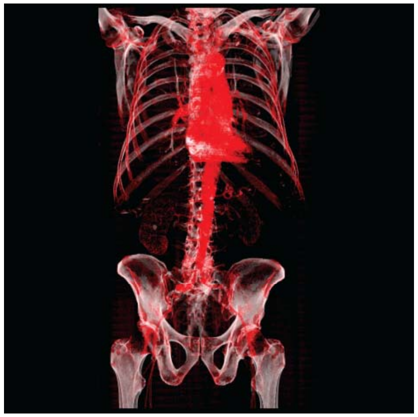




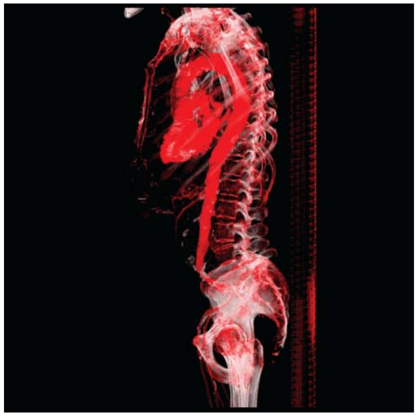
Results: Dataset Views



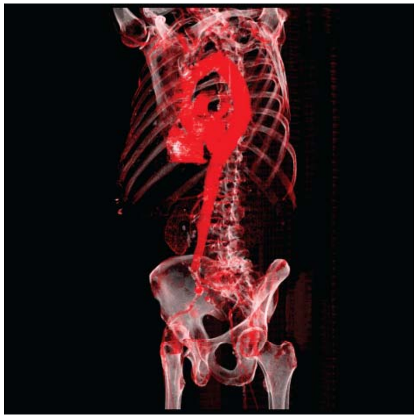
Bottom View



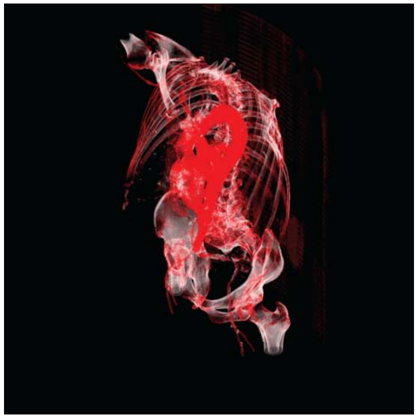
Front View



Side View



45^0 Side View



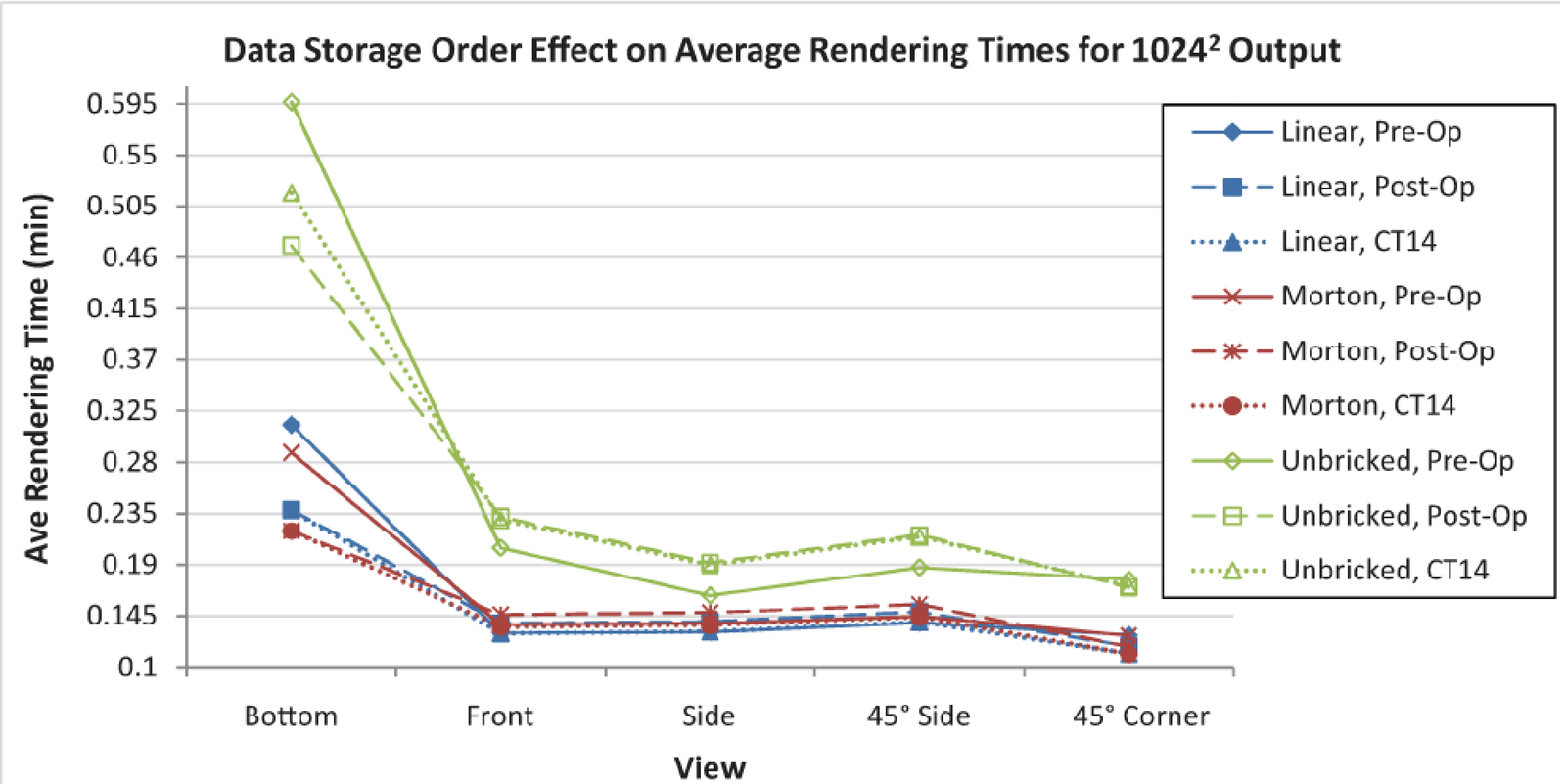
45^0 Corner View

Results: Single-Core: Brick Sizes

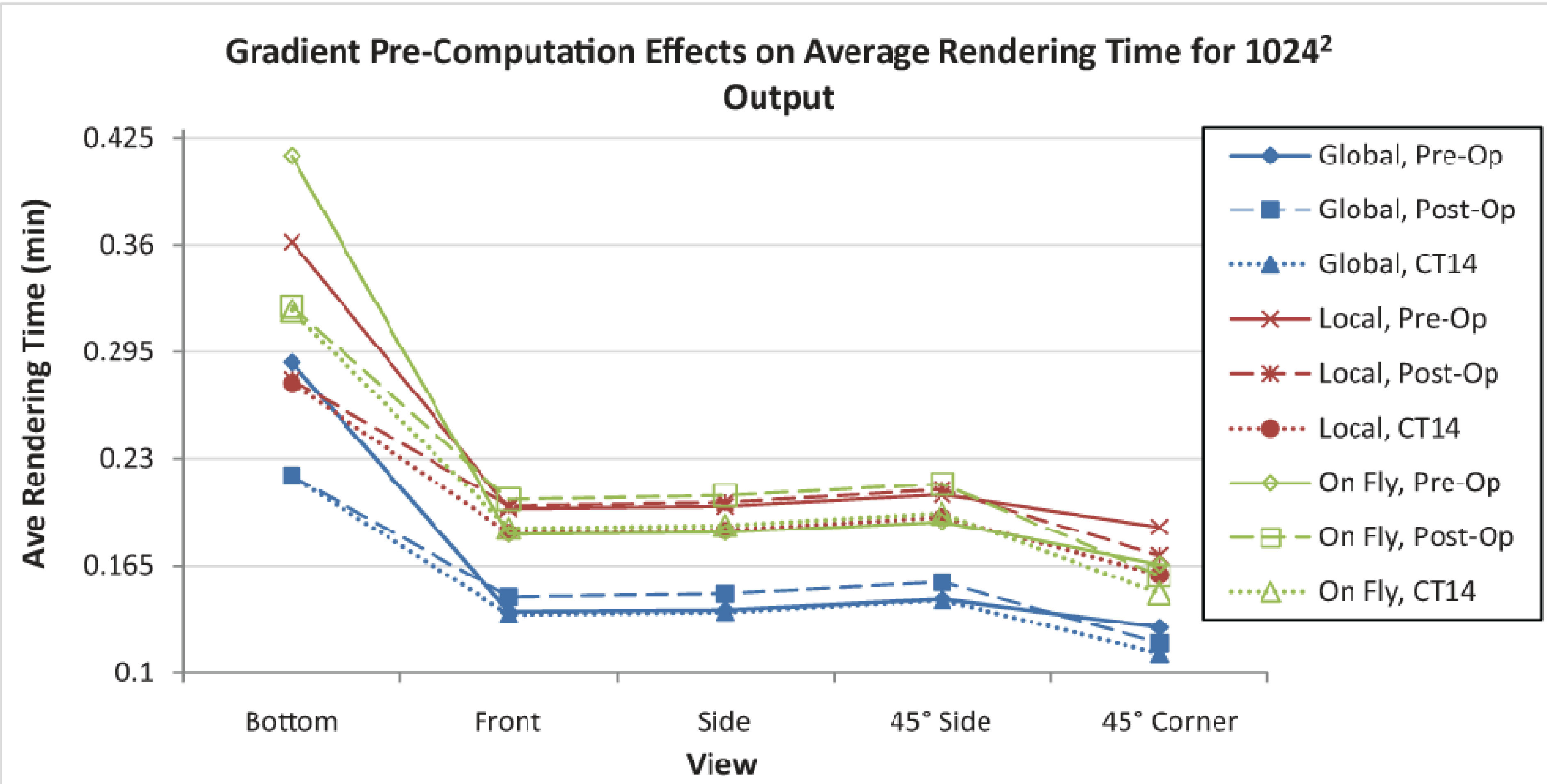
- cubed bricks
  - consistent rendering times across different views
  - lower rendering times without sub-bricks
- choosing size
  - dimension power of 2 for voxel index computation
  - 32^3 best

	Voxel Mem (KB)	Voxel & Gradient Mem (KB)	Rendering time (s)
32^3	64	448	4.9
32 x 32 x 32	64	448	9.3
64^3	512	3,584	5.4
64 x 64 x 64	512	3,584	17.0

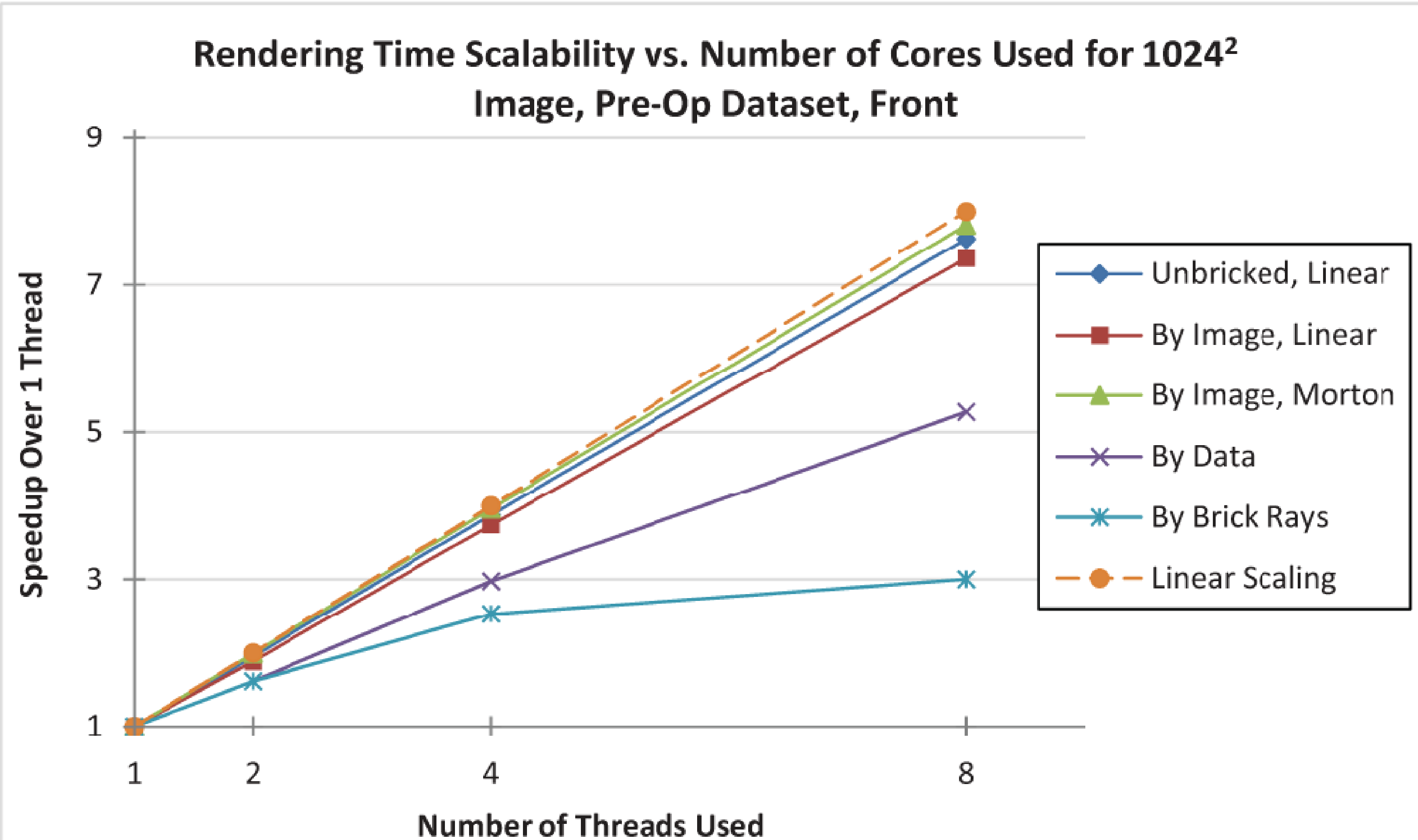
Results: Single-Core: Data Storage Order



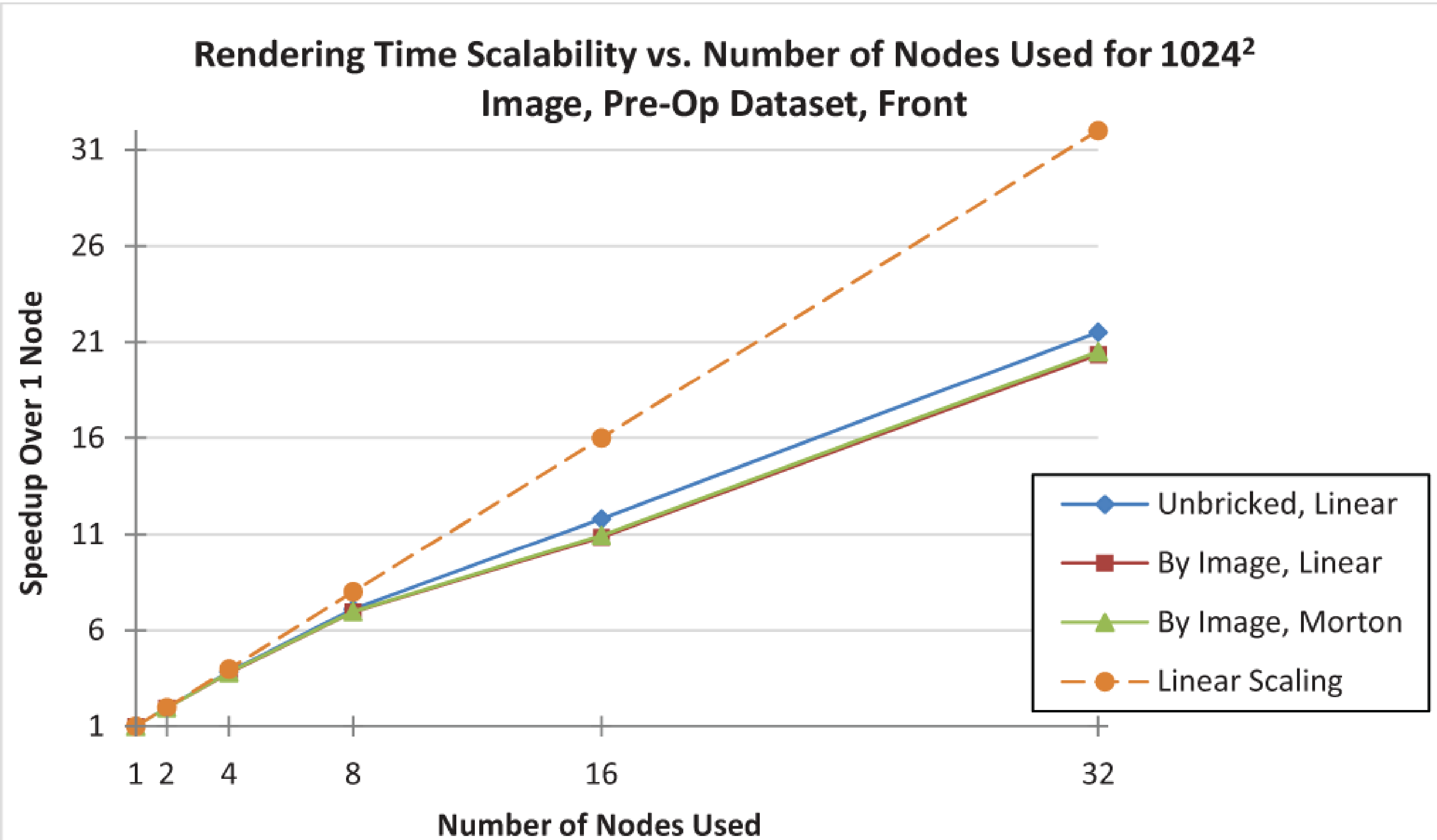
Results: Single-Core: Gradient Computation



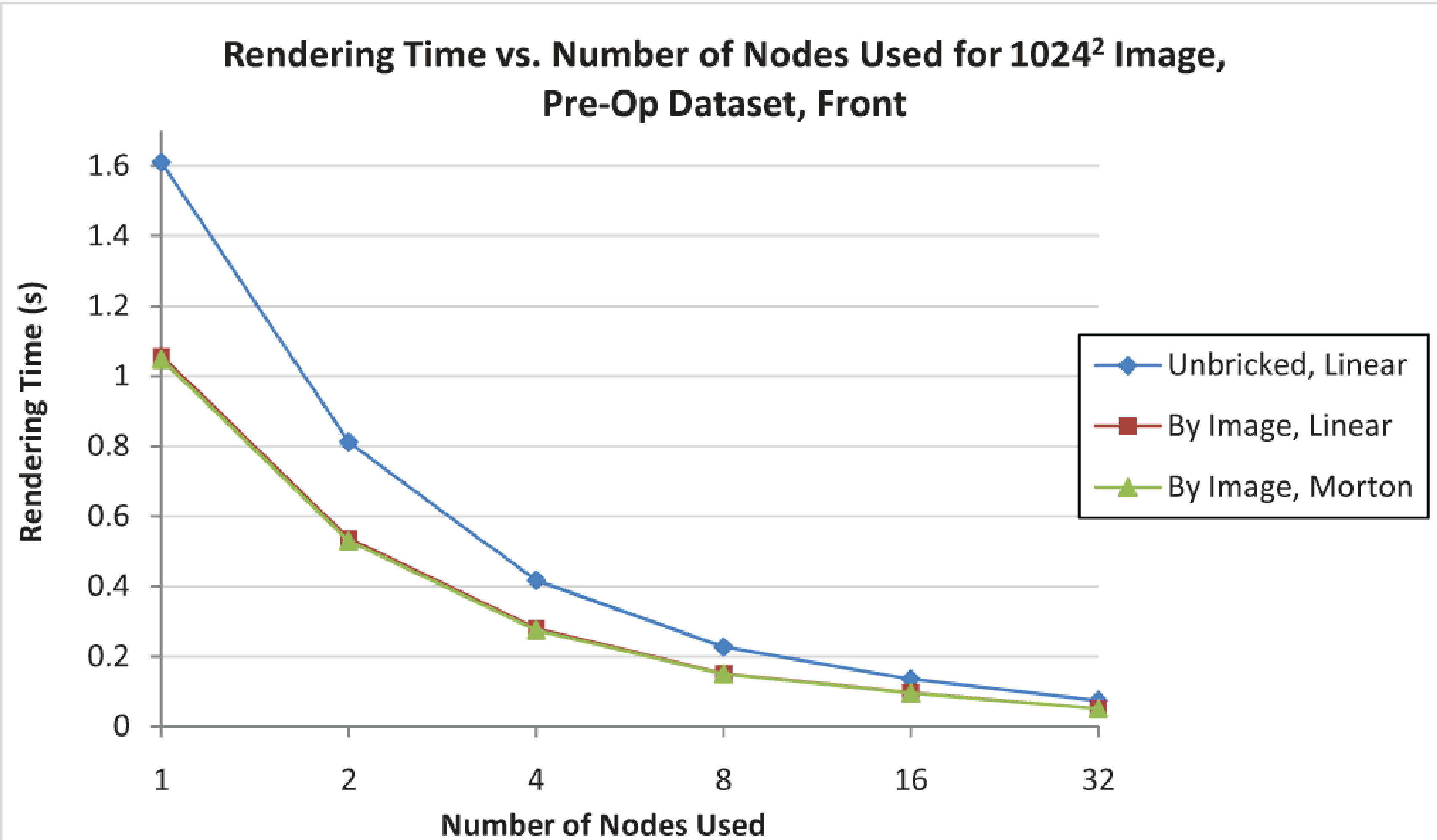
Results: Parallelizations: Number of Cores



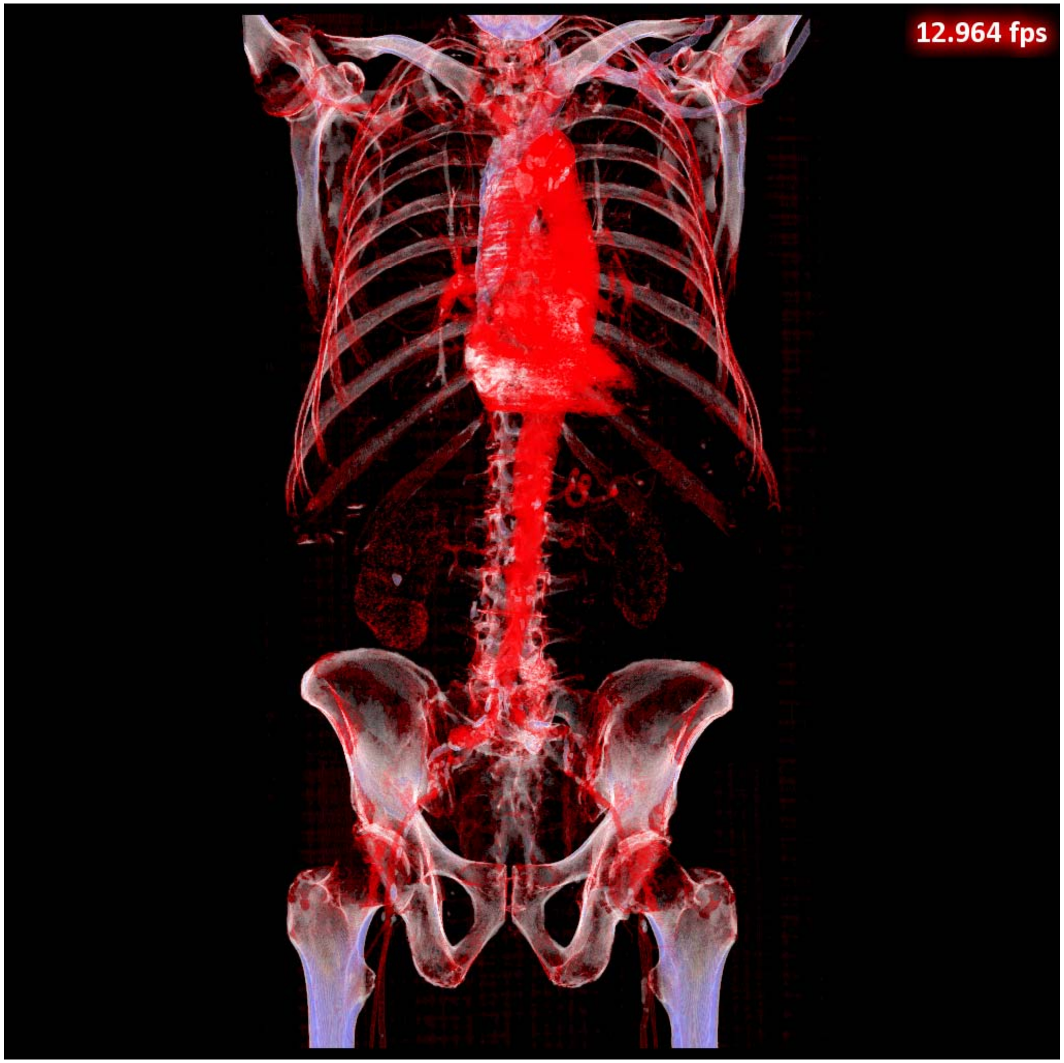
Results: Parallelizations: Number of Nodes



Results: Parallelizations: Number of Nodes



Results: Pre-Operation Movie



Results: Post-Operation Movie

Conclusion: Accelerations

- bricking improves rendering times by 33.7%
- Morton data storage 2% faster than linear (32 nodes)
- global gradient cache fastest but high memory cost

Conclusion: Scalability

- scalability of bricked raycaster
  - great for multi-core: 7.87 for 8 cores
  - room for improvement for multi-node:
    - 3.75 for 4 nodes
    - 6.9 for 8 nodes
    - 20.6 for 32 nodes
- 32 nodes can render 1024^2 images at 10-20 fps
- general algorithm independent of specialized hardware (reduces cost, easily extendable)

Future Work

- time-dependent datasets
  - improve realism by matching to heart-beat
- application in an operating environment
  - highly distributed system
  - once internet bandwidth is high enough, computation located outside of hospital

Acknowledgements

- Advisors: Don Greenberg, Alex Vladimirsky
- Kavita Bala, Steve Marschner, Doug James, Bruce Walter, Lars Schumann
- Linda Stephenson, Hurf Sheldon, Peggy Andersen
- Edgar Velazquez, Brendan Holt, Todd Harvey, David Kaplan, Matt Low, Peter Kung, Adam Abree, Milos Hasan
- Program of Computer Graphics, Dept. of Computer Science, Dept. of Architecture, National Science Foundation ITR / AP: CCF-0205438

End

References

- [Lorensen, Cline ‘87] – W. Lorensen, H. Cline “Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm.” *ACM Computer Graphics*, 21(4):163-169, July 1987
- [Levoy ‘88] – M. Levoy “Display of Surfaces from Volume Data.” *IEEE Computer Graphics and Applications*, 8(3):29-37, May 1988
- [Levoy ‘90] – M. Levoy “Efficient Ray Tracing of Volume Data.” *ACM Trans. on Graphics*, 9(3):245-261, July 1990
- [Lacroute, Levoy ‘94] – P. Lacroute, M. Levoy “Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation.” In *SIGGRAPH ‘94: Proc. of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pg 451-458, 1994.
- [Sweeney, Mueller ‘02] – J. Sweeney, K. Mueller “Shear-Warp Deluxe: The Shear-Warp Algorithm Revisited.” In *VISSYM ‘02: Proc. of the Symposium on Data Visualization 2002*, pg 95-104, 2002

References

- [Westover '90] – L. Westover "Footprint Evaluation for Volume Rendering." *SIGGRAPH Computer Graphics*, 24(4):367-376, August 1990.
- [Mueller, Crawfis '98] – K. Mueller, R Crawfis "Eliminating Popping Artifacts in Sheet Buffer-Based Splatting." In *VIS '98: Proc. of the Conference on Visualization '98*, pg 239-245, October 1998.
- [Zwicker et. al. '04] – M. Zwicker, J. Rasanen, M. Botsch, C. Dachsbacher, M. Pauly "Perspective Accurate Splatting." In *GI '04: Proc. on Graphics Interface 2004*, pg 247-254, May 2004.
- [Kreeger, Kaufman '99] – K. Kreeger, A. Kaufman "Hybrid Volume and Polygon Rendering with Cube Hardware." In *HWWS '99: Proc. of the ACM SIGGRAPH / EUROGRAPHICS Workshop on Graphics Hardware*, pg 15-24, 1999.
- [Knittel, Strasser '97] – G. Knittel, W. Strasser "Vizard – Visualization Accelerator for Realtime Display." In *HWWS '97: Proc. of the ACM SIGGRAPH / EUROGRAPHICS Workshop on Graphics*

References

- [Meissner et. al. ‘02] – M. Meissner, U. Kanus, G. Wetekam, J. Hirche, A. Ehlert, W. Strasser, M. Doggett, P. Forthmann, R. Proska “Vizard II: A Configurable Interactive Volume Rendering System.” In *HWWS ‘02: Proc. of the ACM SIGGRAPH / EUROGRAPHICS Workshop on Graphics Hardware*, pp 137-146, 2002.
- [Pfister et. al. ‘99] – H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, L. Seiler “The VolumePro Real-Time Ray-Casting System.” In *SIGGRAPH ‘99: Proc. of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pg 251-260, 1999.
- [Kruger, Westermann ’03] – J. Kruger, R. Westermann “Acceleration Techniques for GPU-Based Volume Rendering.” In *VIS ‘03: Proc. of the 14th IEEE Visualization 2003*, pg 287-292, 2003

End