Advanced Visual Medicine: Techniques, Applications and Software

Foundations of Rendering

Klaus Mueller

Department of Computer Science
Stony Brook University, USA
mueller@cs.sunysb.edu

Overview

Rendering Modes: X-ray, MIP, shaded displays

Basic volumetric rendering: using raycasting

Transfer functions: mapping raw data to visuals

Rendering quality: post- vs. pre-shaded rendering

Controlling rendering effort: occlusions, importance

Rendering acceleration: rendering on GPUs

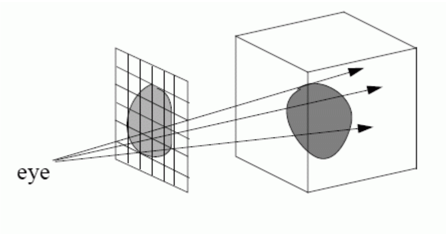
Navigation techniques: e.g., virtual colonoscopy

more info in [Kaufman 05]

Raycasting: Concept

Most intuitive rendering technique

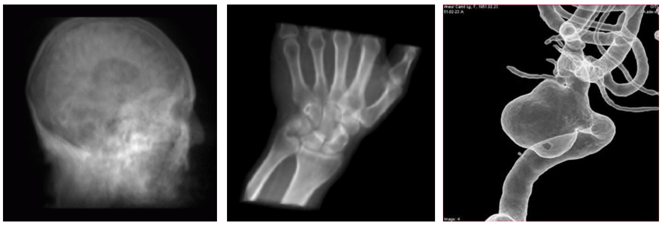
- shoot rays into the scene starting from the eye
- the “gold standard” of volume rendering
- use it to derive the fidelity of other paradigms



Volume Rendering Modes: X-Ray

Rays simply sum everything up that falls into their path

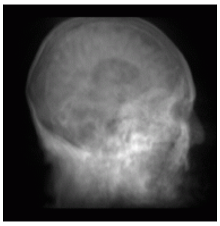
- good for first overview, since no data is culled away
- but overdraw and clutter can be a problem



Volume Rendering Modes: X-Ray

Rays simply sum everything up that falls into their path

- good for first overview, since no data is culled away
- but overdraw and clutter can be a problem

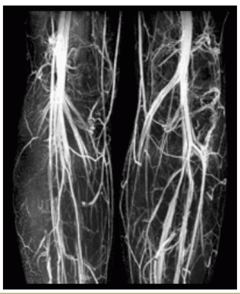



Volume Rendering Modes: MIP (1)

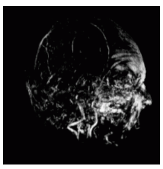
MIP = Maximum Intensity Projection

- a little bit more restrictive than X-ray
- keeps only the maximum value along the ray
- assumes that the maximum value is also the most important
- often used by doctors to get a first look at the data
- great for angiography visualization

legs



head



Volume Rendering Modes: MIP (2)

Various types of MIP [Yen 97] [Shareef 02]

- original MIP (OM)

  $$I(p) = \max_{0 \le z \le D}(f(z))$$

- thin slab MIP (TM)

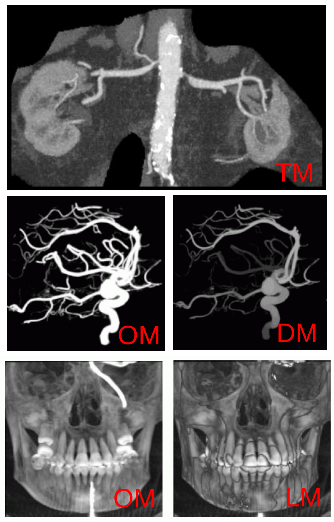
  $$I(p) = \max_{d1 \le z \le d2}(f(z))$$

- depth-shaded MIP (DM)

  $$I(p) = \max_{0 \le z \le D}(d(z) \cdot f(z))$$

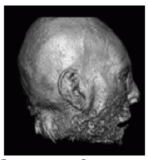
- local maximum MIP (LM)

  $$I(p) = \min_{0 \le z \le D}(z \mid f(z) \ge t \wedge LMAX(z))$$



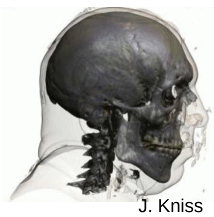
Shaded Volume Rendering

Identify actual surfaces and accentuate them by modeling the reflection of light



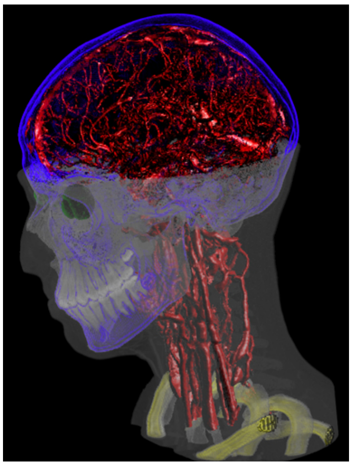
- require surface gradients
- need to map densities to color
- this mapping is done via *transfer functions*

There can be multiple nested surfaces



- to see all one needs to make front surfaces *semi-transparent* and *composite colors*
- achieve this mapping with a transfer function

All Combined: Two-Level Volume Rendering



Skin and teeth: MIP with different intensity ramps

Blood vessels: shaded volume rendering

Skull: contour rendering

Vertebrae: gradient magnitude-weighted transfer function with shaded volume rendering

A clipping plane has been applied to the skin object

[Hauser 01]

Shaded Volume Rendering: Overview

There are two approaches for this:

- polygonal rendering
- direct volume rendering

Shaded Volume Rendering: Overview

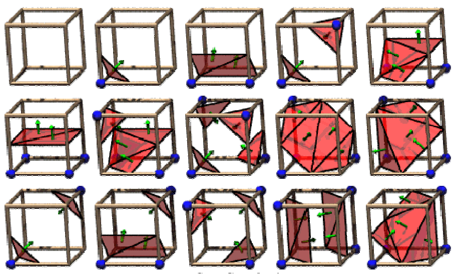
There are two approaches for this:

- polygonal rendering (somewhat dated)
- direct volume rendering (preferred)

Rendering With Polygons: Mesh Generation

Extraction of polygon mesh with *Marching Cubes*

- Set iso-value *iso* to boundary
- Label all voxels below *iso* as “*out*”, else as ”*in*”
- Then each voxel 8-cell fits one of 15 base cases:



The 15 Cube Combinations

[Lorensen 87]

Rendering With Polygons: Mesh Generation

Assemble mesh given the extracted polygons

Render with polygon graphics hardware

Rendering With Polygons: Mesh Generation

Assemble mesh given the extracted polygons

Render with polygon graphics hardware

Problem:

- will likely get very large meshes
- graphics pipeline will be overwhelmed
- rendering will not be interactive

Rendering With Polygons: Mesh Generation

Assemble mesh given the extracted polygons

Render with polygon graphics hardware

Problem:

- will likely get very large meshes
- graphics pipeline will be overwhelmed
- rendering will not be interactive

Solution:

- simplify mesh - undesirable since loss of detail

Rendering With Polygons: Mesh Generation

Assemble mesh given the extracted polygons

Render with polygon graphics hardware

Problem:

- will likely get very large meshes
- graphics pipeline will be overwhelmed
- rendering will not be interactive

Solution:

- simplify mesh - undesirable since loss of detail
- do Direct Volume Rendering (DVR)

Rendering With Polygons: Mesh Generation

Assemble mesh given the extracted polygons

Render with polygon graphics hardware

Problem:

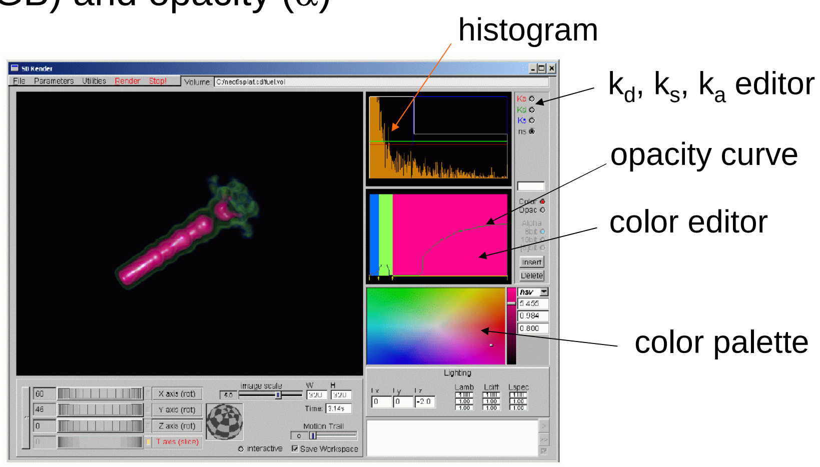
- will likely get very large meshes
- graphics pipeline will be overwhelmed
- rendering will not be interactive

Solution:

- simplify mesh - undesirable since loss of detail
- do Direct Volume Rendering (this tutorial!)

Transfer Functions: Basics

Map the volume scalar data (densities) into color (RGB) and opacity (α)



Back to Volume Rendering

Two modes of rendering

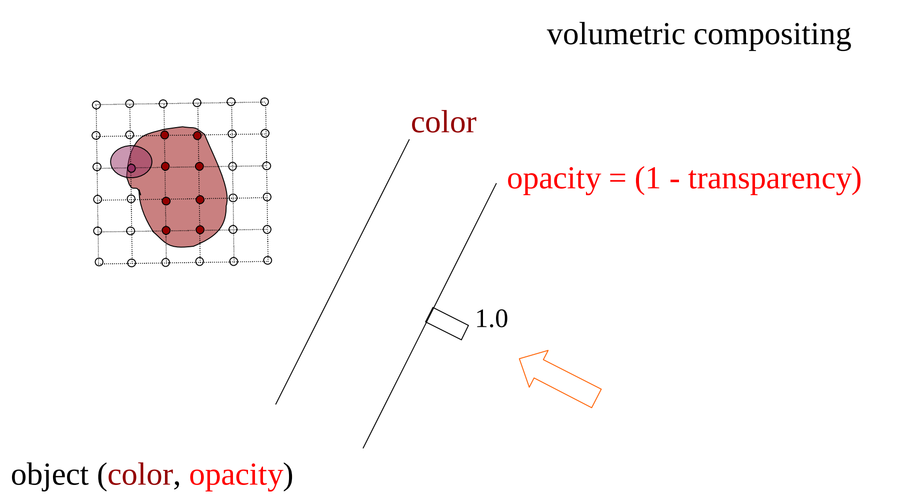
Pre-classified rendering: map first and render later

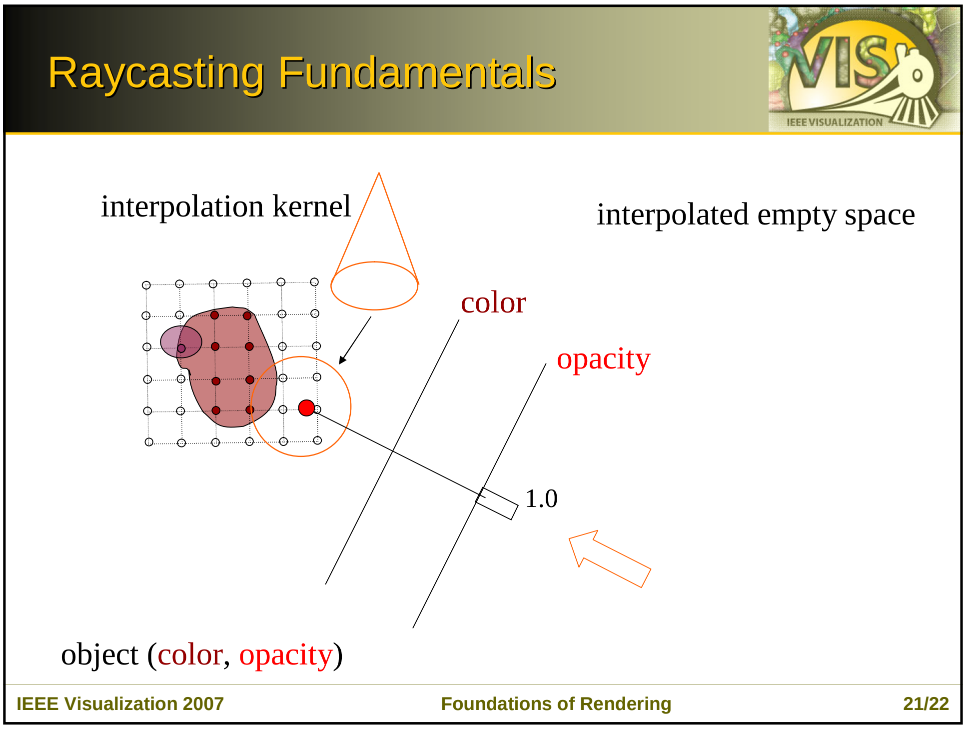
- now the voxels are 4-tuples: RGB color and opacity (=1-transparency)
- also known as *pre-shaded rendering*
- we shall look at this next

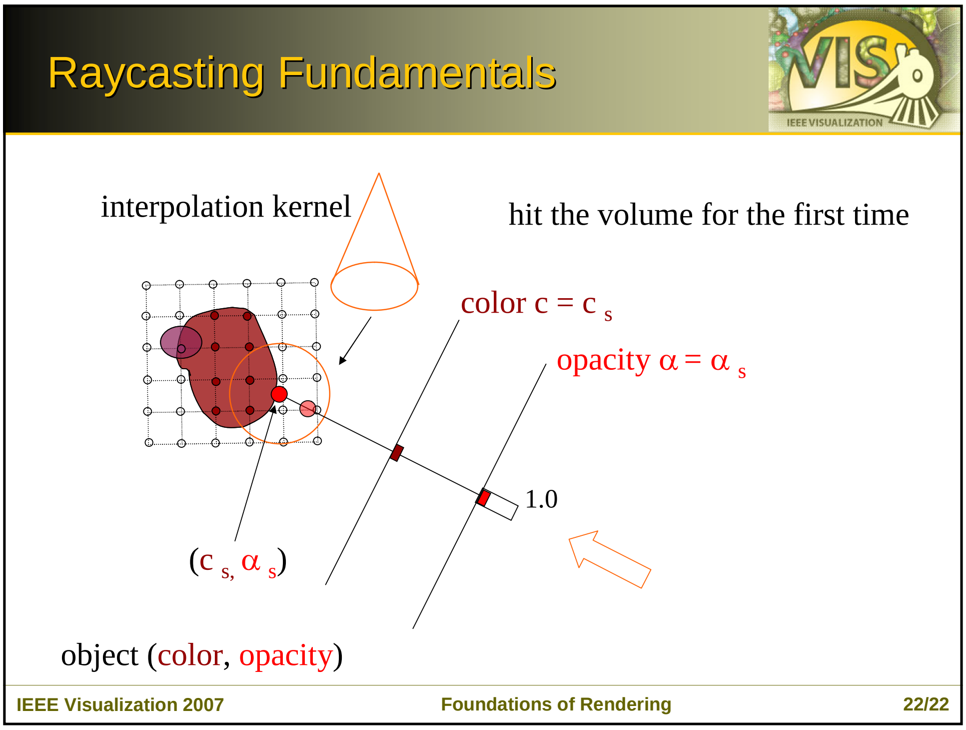
Post-classified rendering: render first and map later

- also known as *post-shaded rendering*
- we shall look at this later

Raycasting Fundamentals

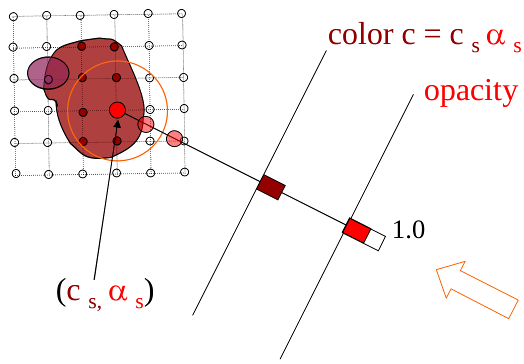



Raycasting Fundamentals
IEEE VISUALIZATION
interpolation kernel
interpolated empty space
color
opacity
1.0
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
21/22


Raycasting Fundamentals
IEEE VISUALIZATION
interpolation kernel
hit the volume for the first time
color c = c s
opacity α = α s
1.0
(c s, α s)
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
22/22

Raycasting Fundamentals

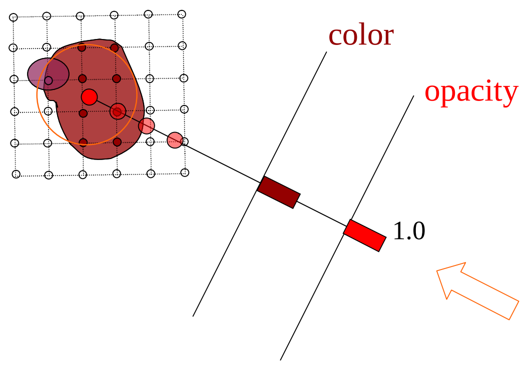
volumetric compositing



object (color, opacity)

Raycasting Fundamentals

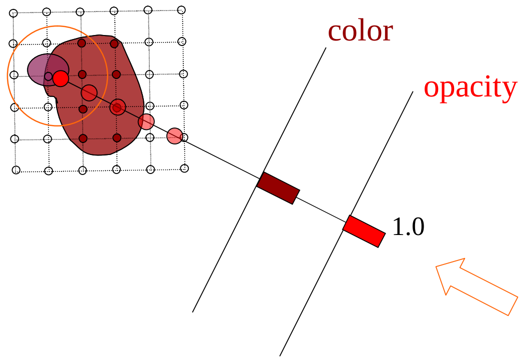
volumetric compositing



object (color, opacity)

Raycasting Fundamentals

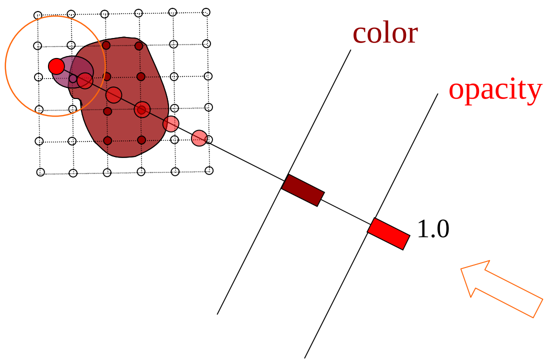
volumetric compositing



object (color, opacity)

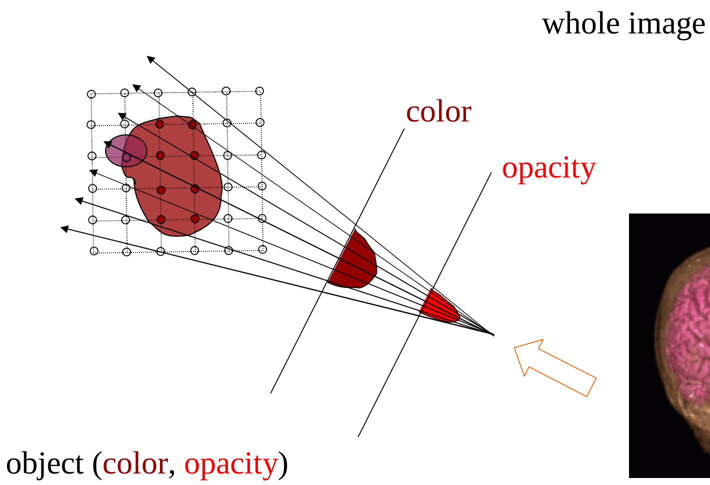
Raycasting Fundamentals

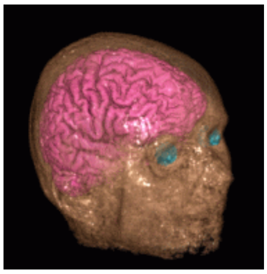
volumetric compositing



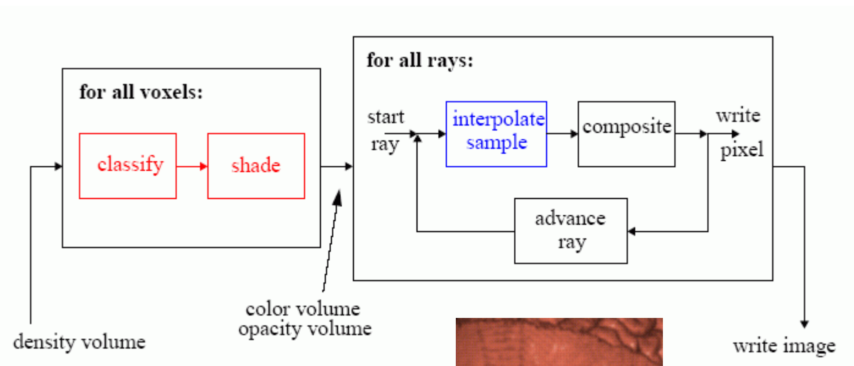
object (color, opacity)

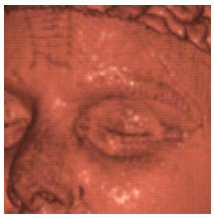
Raycasting Fundamentals



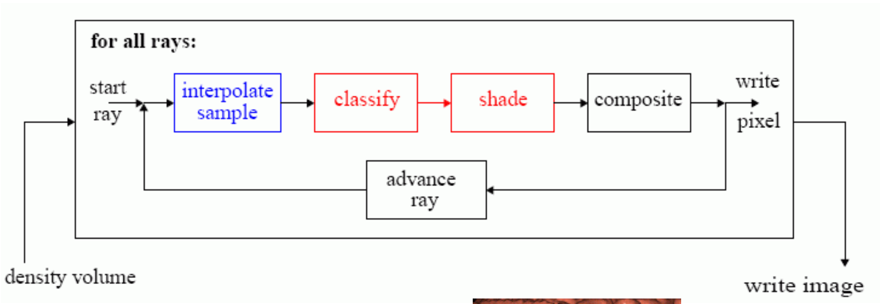


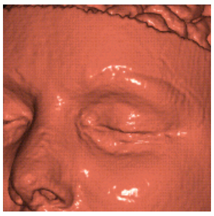
Pre-Shaded Rendering Pipeline



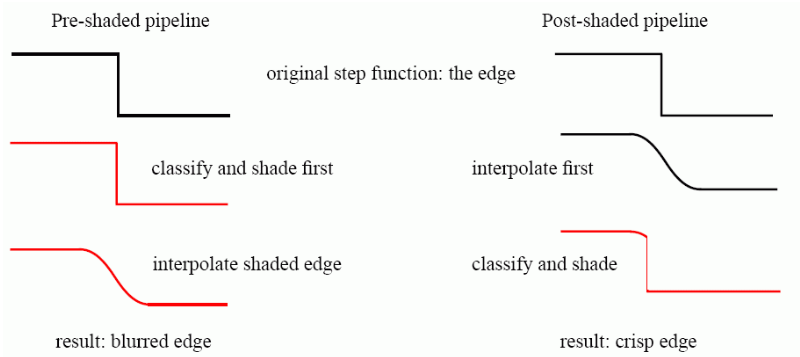


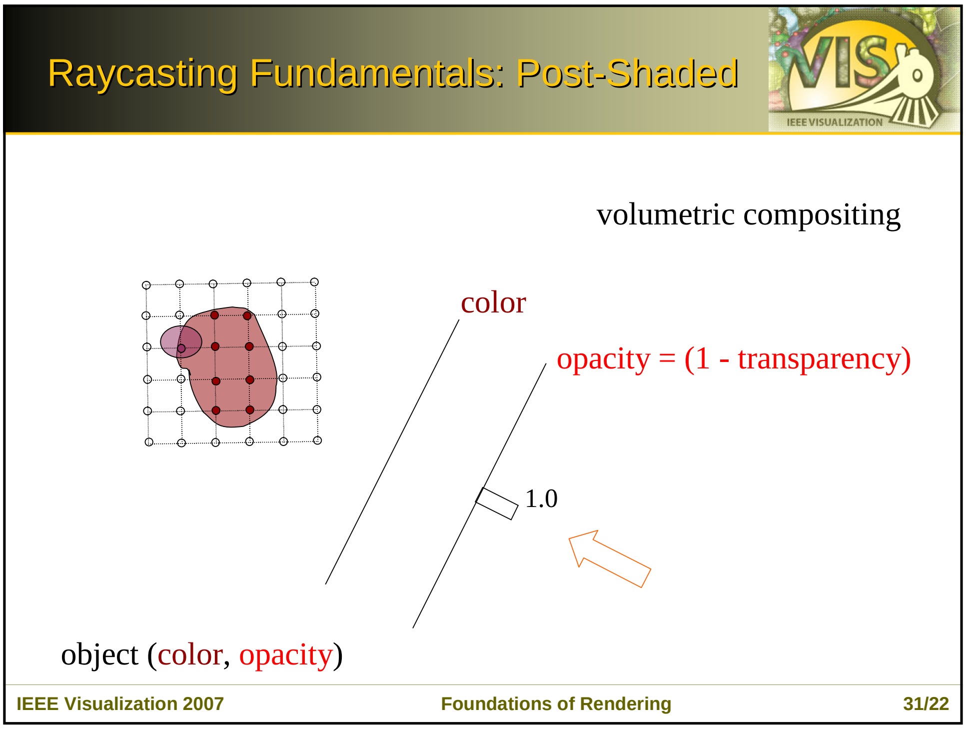
Post-Shaded Rendering Pipeline

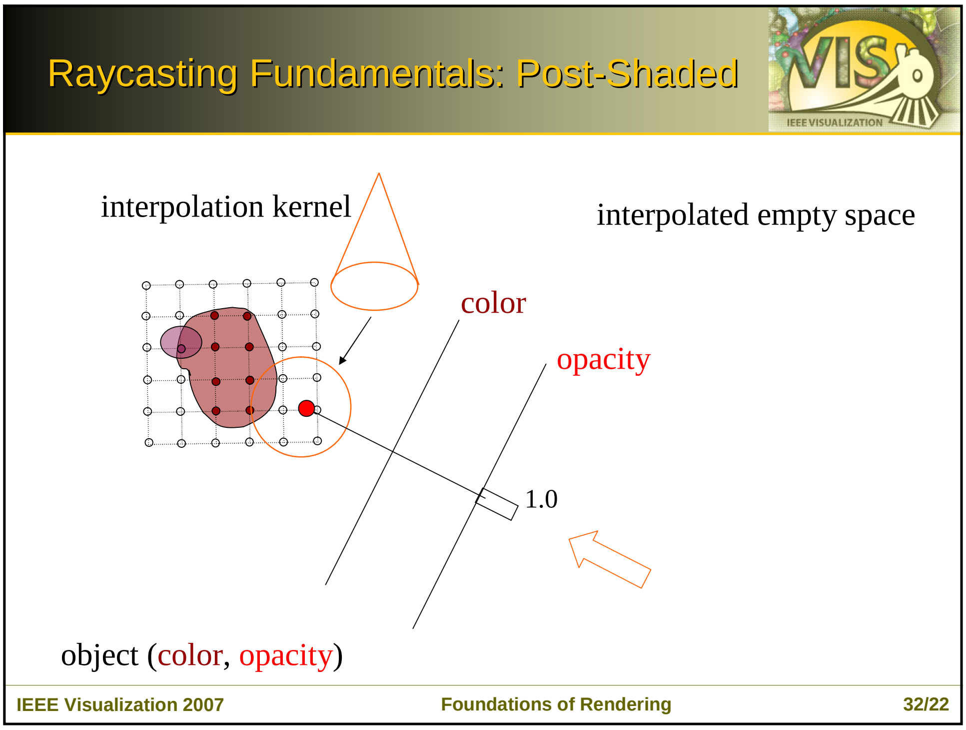


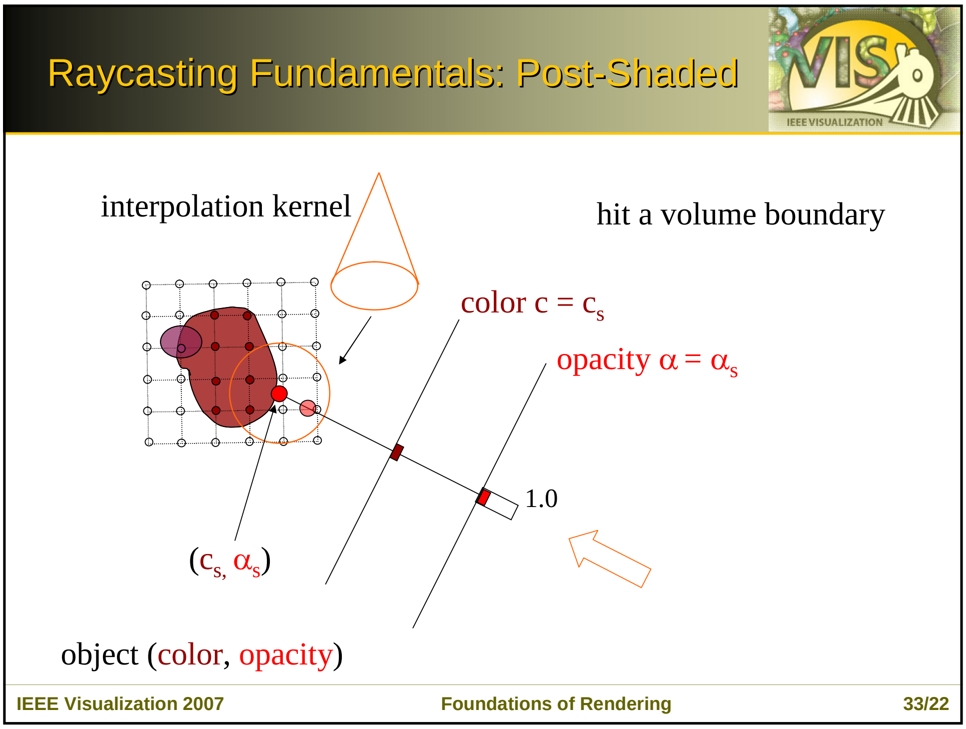


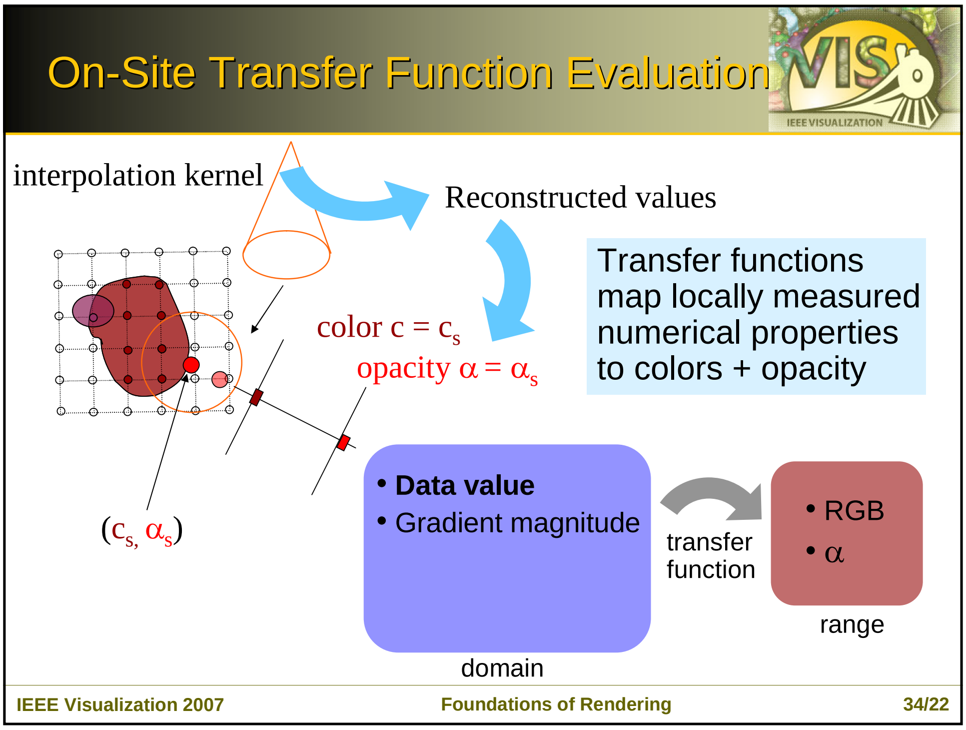
Pre- vs. Post-Shaded Rendering




Raycasting Fundamentals: Post-Shaded
IEEE VISUALIZATION
volumetric compositing
color
opacity = (1 - transparency)
1.0
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
31/22

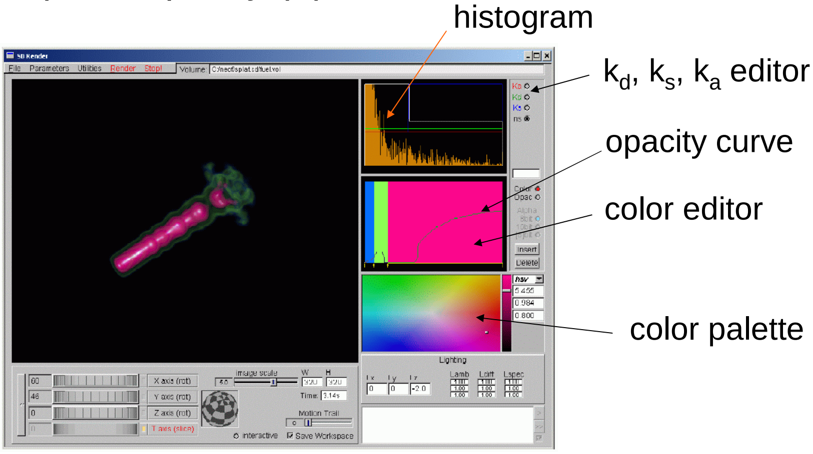

Raycasting Fundamentals: Post-Shaded
IEEE VISUALIZATION
interpolation kernel
interpolated empty space
color
opacity
1.0
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
32/22


Raycasting Fundamentals: Post-Shaded
IEEE VISUALIZATION
interpolation kernel
hit a volume boundary
color c = c$_s$
opacity α = α$_s$
1.0
(c$_s$, α$_s$)
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
33/22


On-Site Transfer Function Evaluation
IEEE VISUALIZATION
interpolation kernel
Reconstructed values
Transfer functions map locally measured numerical properties to colors + opacity
color c = c$_s$
opacity α = α$_s$
(c$_s$, α$_s$)
• Data value
• Gradient magnitude
domain
transfer function
• RGB
• α
range
IEEE Visualization 2007
Foundations of Rendering
34/22

Transfer Functions: Basics

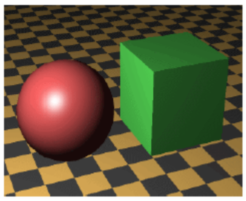
Map the volume scalar data (densities) into color (RGB) and opacity (α)

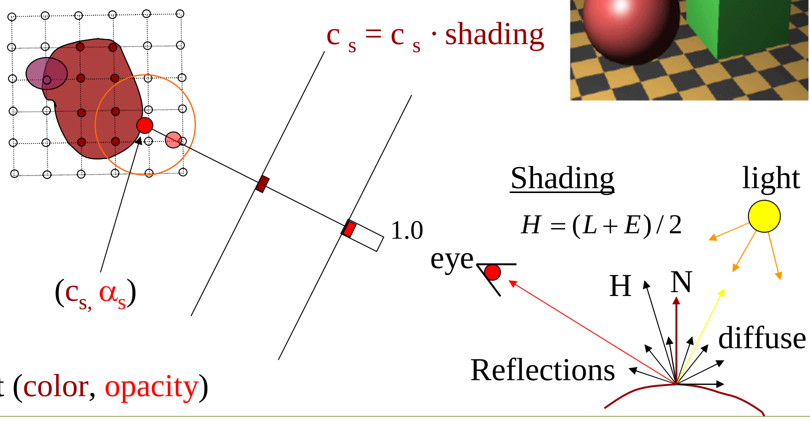


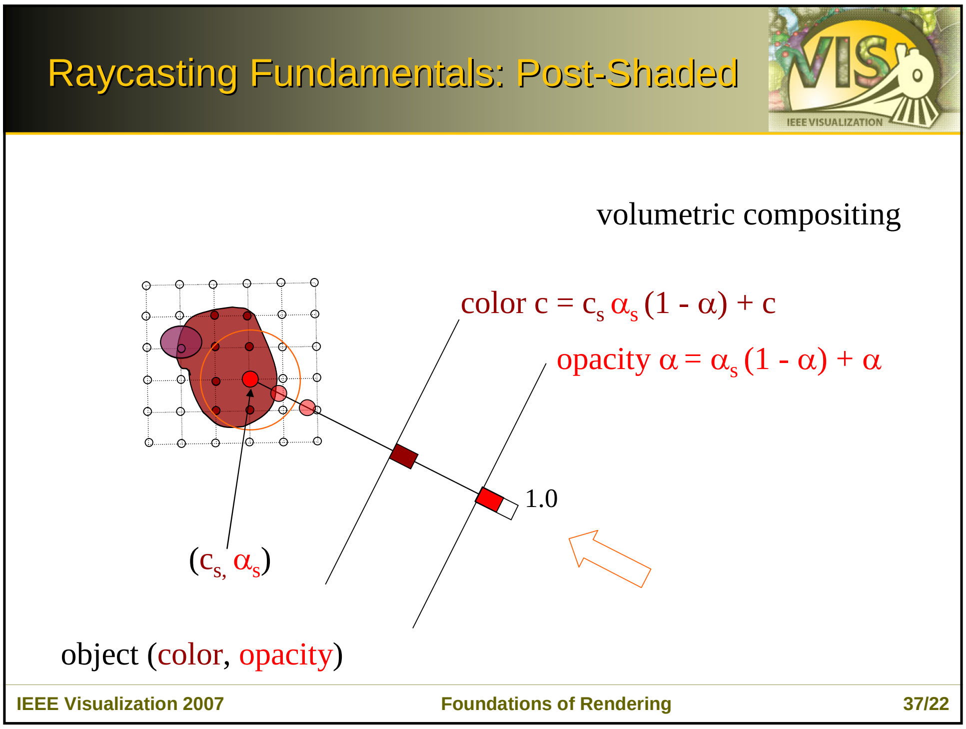
Raycasting Fundamentals: Post-Shaded

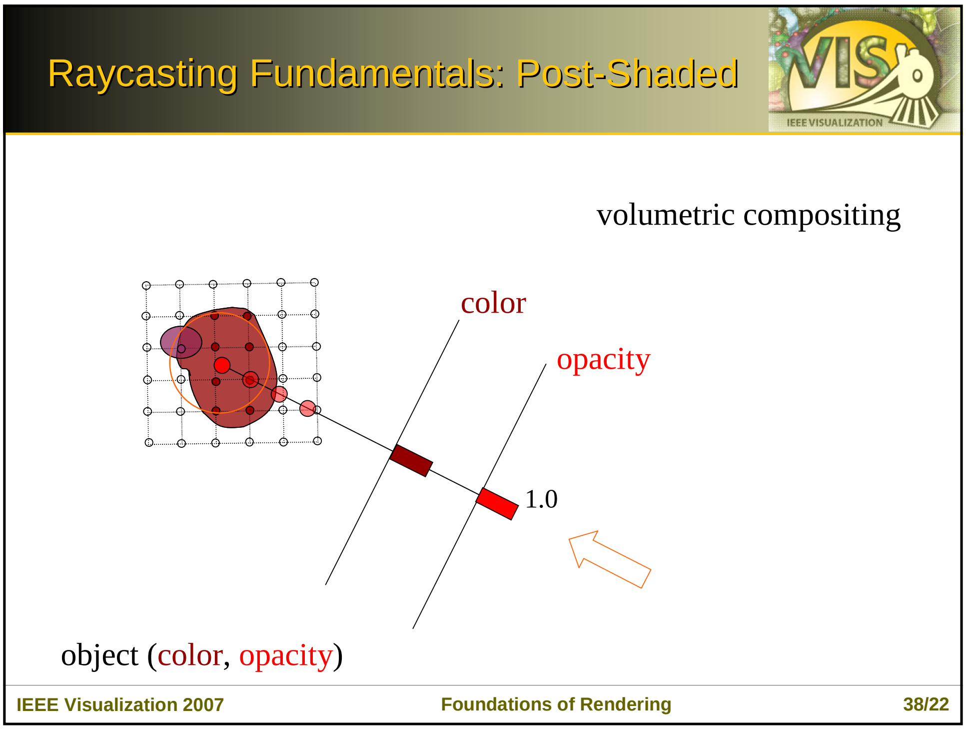
$$shading = c_s(k_a I_a + k_d I_L N \cdot L) + k_s I_L (H \cdot N)^{ns}$$

use unit vectors



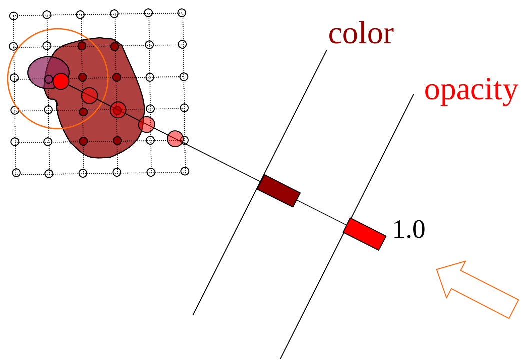



Raycasting Fundamentals: Post-Shaded
IEEE VISUALIZATION
volumetric compositing
color c = c_s α_s (1 - α) + c
opacity α = α_s (1 - α) + α
1.0
(c_s, α_s)
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
37/22


Raycasting Fundamentals: Post-Shaded
IEEE VISUALIZATION
volumetric compositing
color
opacity
1.0
object (color, opacity)
IEEE Visualization 2007
Foundations of Rendering
38/22

Raycasting Fundamentals: Post-Shaded

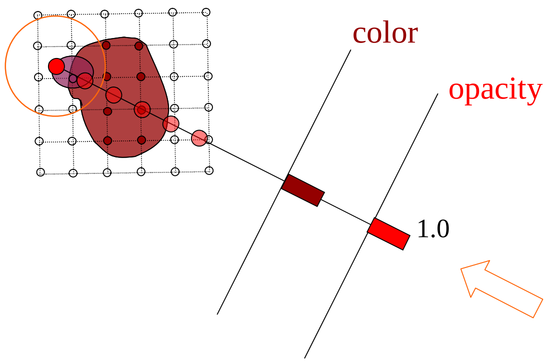
volumetric compositing



object (color, opacity)

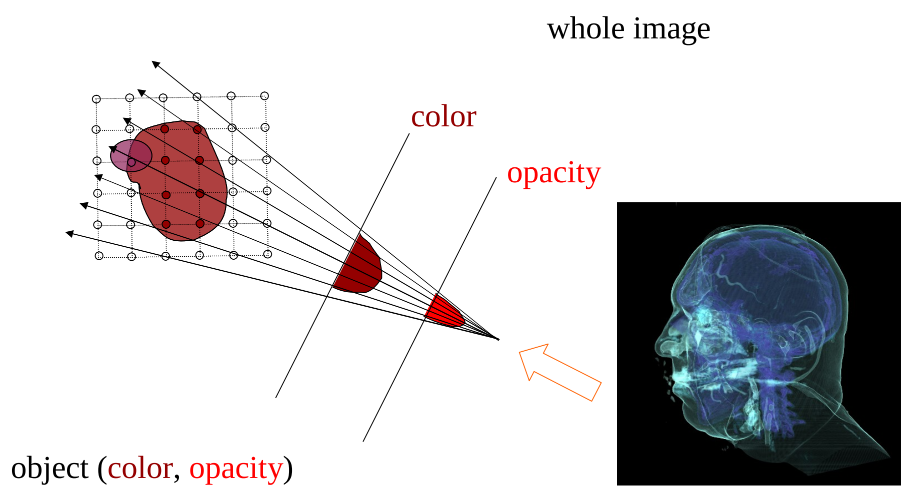
Raycasting Fundamentals: Post-Shaded

volumetric compositing

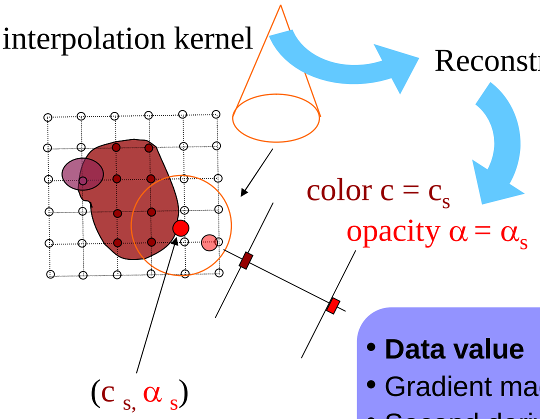


object (color, opacity)

Raycasting Fundamentals: Post-Shaded



Back To Transfer Functions…



Transfer functions map locally measured numerical properties to colors + opacity

- **Data value**
- Gradient magnitude
- Second derivative
- Surface curvature

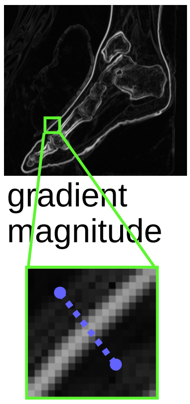
domain

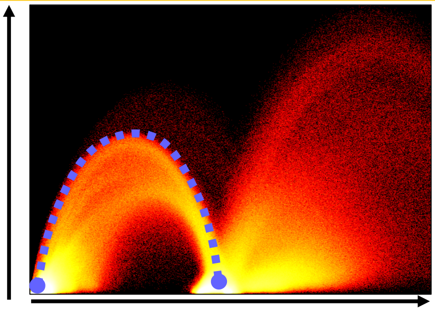
transfer function

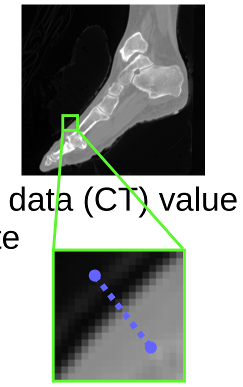
- RGB
- α

range

Transfer Functions: Multi-Dimensional

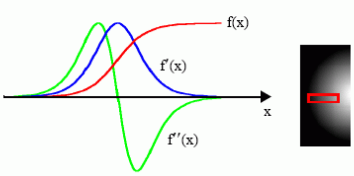




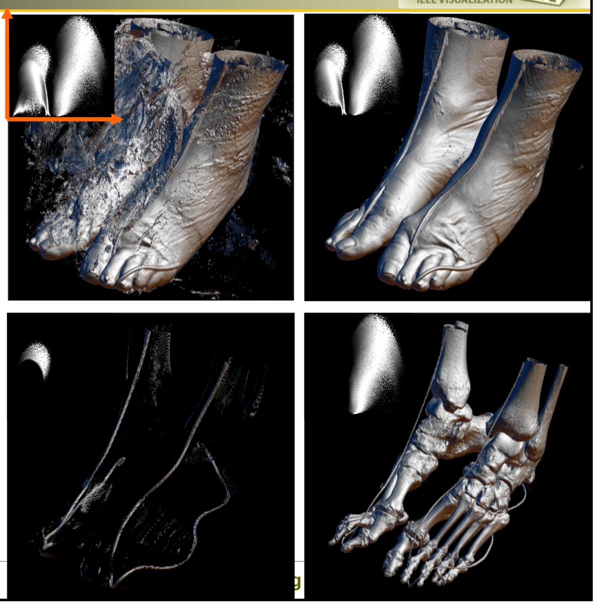


- **Boundaries** in volume create **arches** in (value,gradient) domain [Kindlmann 98]
- Arches guide placement of opacity to emphasize material interfaces [Kniss 01]

Transfer Functions: Multi-Dimensional



- Boundaries can be described in terms of:
  - – maximum in 1st derivative
  - – zero-crossing in 2nd derivative
- Semi-automatic classification possible in clean data



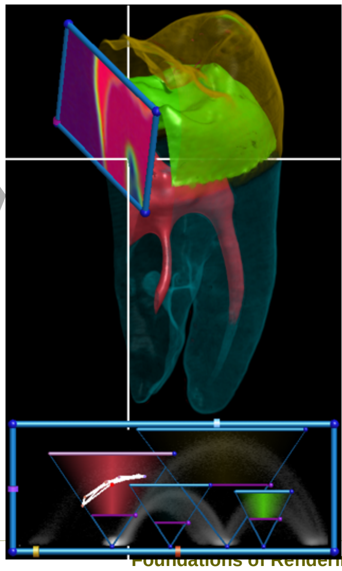
Transfer Functions: Multi-Dimensional

Dual-domain interaction:

[Kniss 01]

Make features opaque by pointing at them

New Rendering

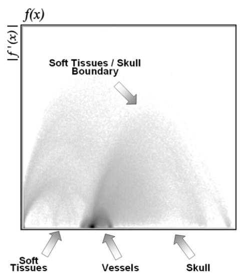


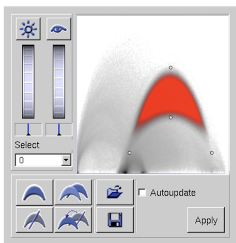
Actions in spatial domain

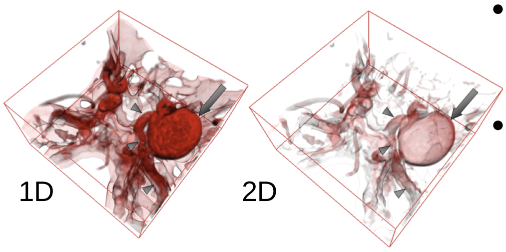
Changes to transfer function

New transfer function

Multi-D TFs: Refined and Applied







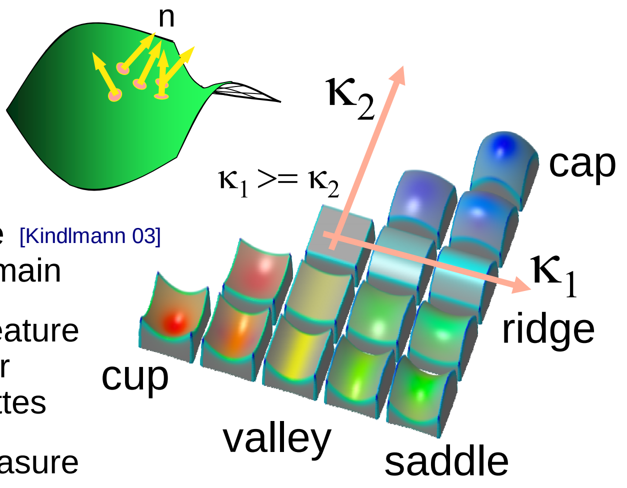
- Higeura et al. 04: Automatic Adjustment of Bidimensional Transfer Functions for Direct Volume Visualization of Intracranial Aneurysms
- Similar scans have similar histograms
- Registration between histograms: map standard TF to new scan

Transfer Functions: Multi-Dimensional

Curvature: how change in surface position changes surface normal (**n**)

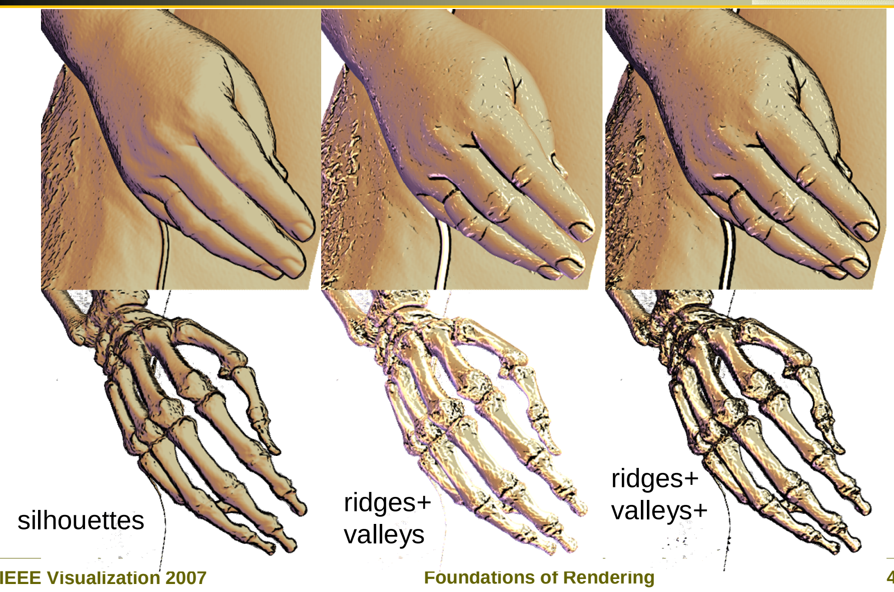
Principal curvatures (κ_1,κ_2) form possible transfer function domain [Kindlmann 03]

- Enables surface feature enhancement, better control over silhouettes
- Convolution to measure 1st and 2nd derivatives



IEEE Visualization 2007 Foundations of Rendering 47/22

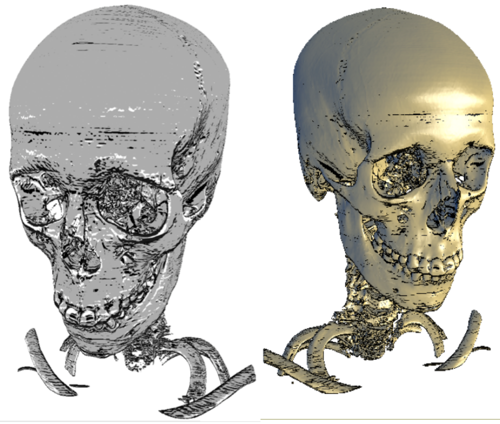
Transfer Functions: Multi-Dimensional



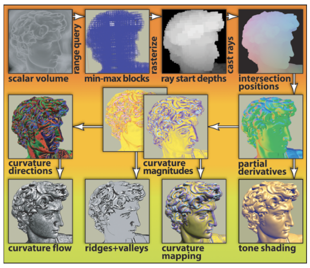
IEEE Visualization 2007 Foundations of Rendering 48/22

Curvature Based TFs: Interactive

- Hadwiger et al. 05: Real-Time Ray-Casting and Advanced Shading of Discrete Isosurfaces
- Cubic-spline Convolution-based Curvature on the GPU



- Interactive exploration with NPR effects



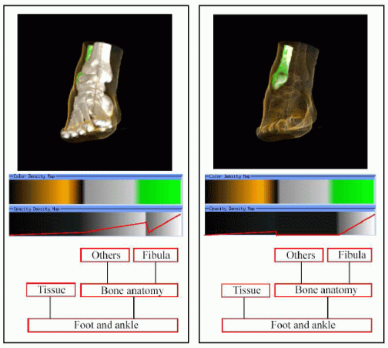
Transfer Functions: Clinical Practice

For medical visualization it is advisable to keep the interaction with transfer functions at a minimum

Doctors (unlike scientists) do not have the time (nor desire) to play with complex transfer function editors

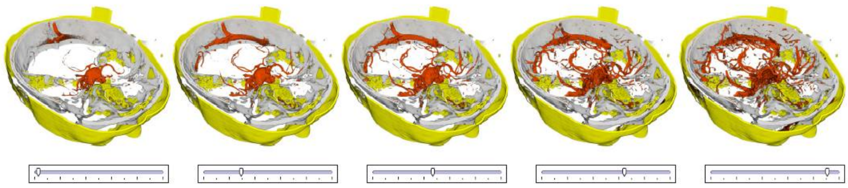
Better approach: [Mueller 05]

- simplify
- make task-oriented
- automate
- include semantics



Transfer Functions: Clinical Practice

A single slider bar is most appreciated [Rezk-Salama Vis06]

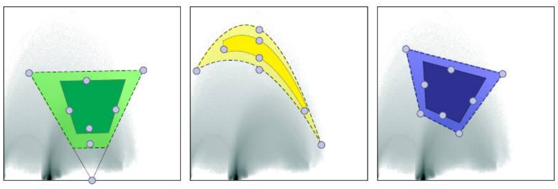


Enables doctors to quickly fine-tune the transfer function for specific objects

- works since in CT usually only small deviations exist
- but these require complex interactions in the transfer function domain

Parameter Mapping Approach (1)

Typical transfer function parameterization:



Datasets typically only deviate modestly from this

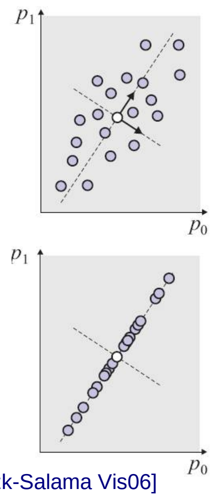
- but in complex ways
- meaning, lots of tweaking is required

[Rezk-Salama Vis06]

Parameter Mapping Approach (2)

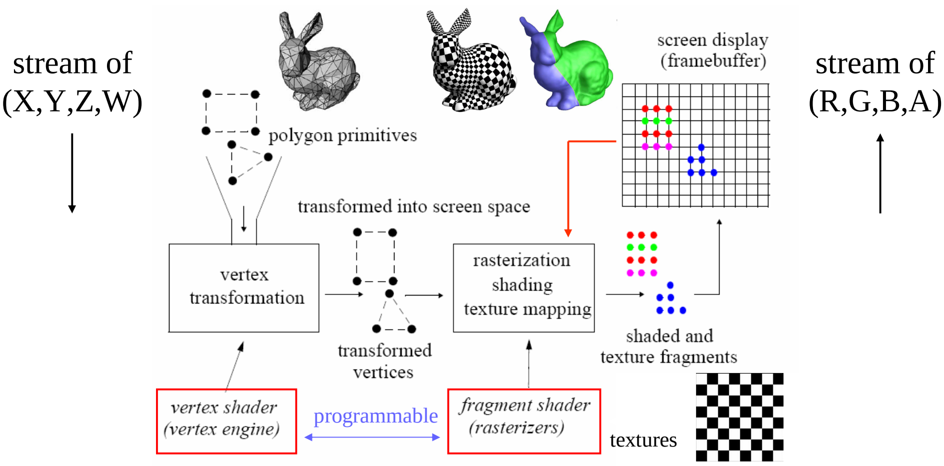
We can learn these deviations by observing a few datasets

- encode the parameters into an N-D vector
- find the principal component of the vectors (the main Eigenvector)
- project all other vectors onto this Eigenvector
- the min and max then represent the min and max of the slider



[Rezk-Salama Vis06]

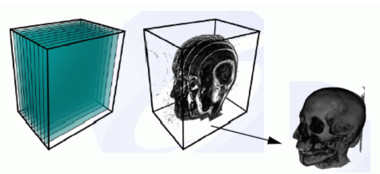
Accelerated Rendering on GPUs



Accelerated Rendering on GPUs: Concept

Simplest approach [Rezk-Salama 01]

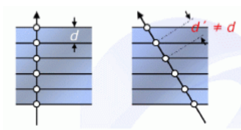
- represent the volume as a stack of axis-aligned “proxy polygons”
- texture-map volume slices onto corresponding proxy polygons
- render polys to screen, properly shifted according to viewing direction, front-to-back
- shade and composite slice by slice



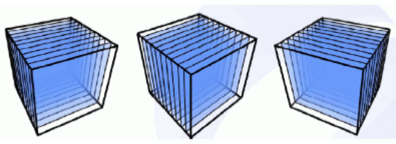
Accelerated Rendering on GPUs: Main Issues

There are two main disadvantages with this approach:

- sampling distance *d* is larger than 1.0 for off-axis viewing directions



- need three stacks of volumes, one for each major viewing direction

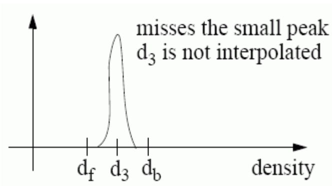


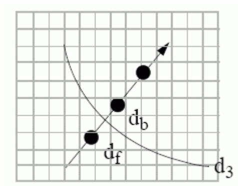
from: Engel, Kraus, Ertl, Graphics Hardware Workshop 2001

Pre-Integrated Volume Rendering (1)

Designed to overcome artifacts due to:

- too large sampling intervals (rays, 2D textures)
- transfer functions with high frequency features, which may not be captured by two consecutively interpolated densities





Pre-Integrated Volume Rendering (2)

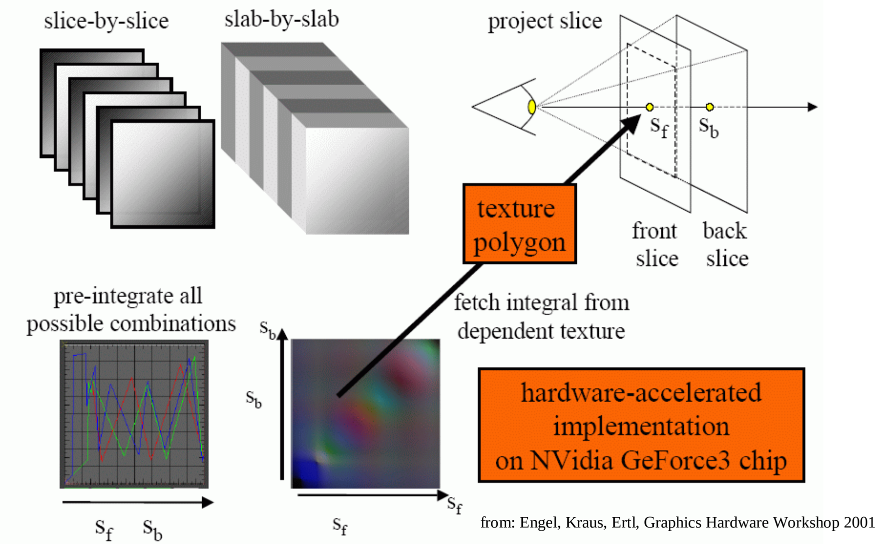
Solution:

- pre-compute color and opacity integrals for all possible front- and back density pairs, d_f and d_b
- gives rise to a 2D table, indexed by interpolated d_f and d_b, assuming piecewise linear densities:

$$c(d_f, d_b) = \int_0^1 \mu((1-s)d_f + sd_b) \cdot c((1-s)d_f + sd_b) e^{-\int_0^s \mu((1-s)d_f + sd_b) dt} ds$$

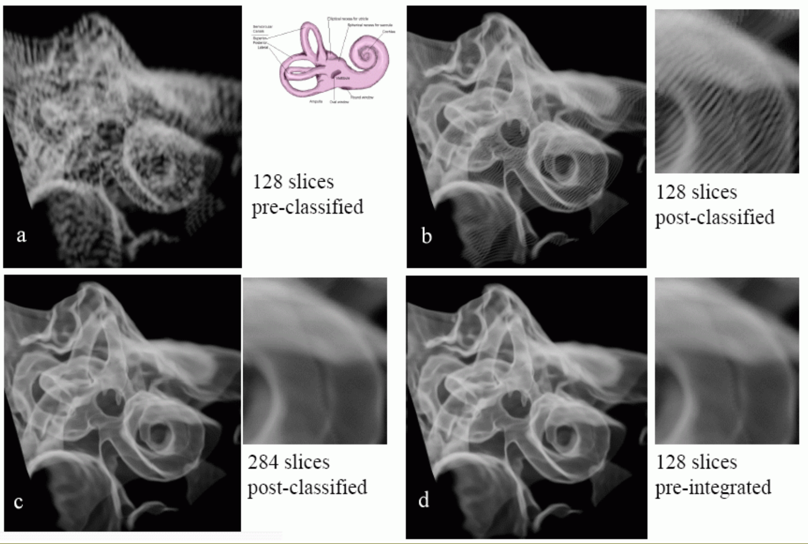
- pre-integrated opacities computed similar

Pre-Integrated Volume Rendering (3)



from: Engel, Kraus, Ertl, Graphics Hardware Workshop 2001

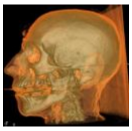
Pre-Integrated Volume Rendering (4)

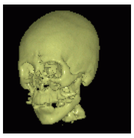


Accelerated Rendering on GPUs: Conclusions

Using the texture mapping hardware approach allows interactive frame rates with practical-sized datasets

More advanced GPU-based renderer offer:





- raycasting (more natural than textures)
- empty-space skipping [Stegmaier 05] [Leung 06]
- early ray termination, occlusion culling
- advanced rendering effects (shadows, translucencies, advanced lighting, etc.)

These offer advantages in speed, quality, flexibility

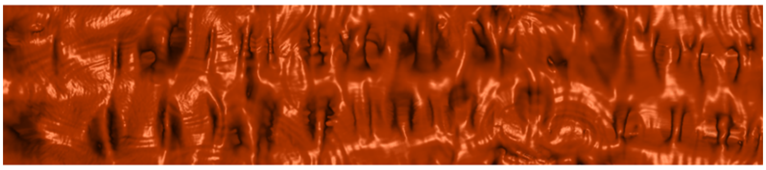
Alternative To 3D Viewing: Intro

Let's have a look at virtual colonoscopy

- complex structure requires fly-throughs

Alternative to 3D Viewing: Practice

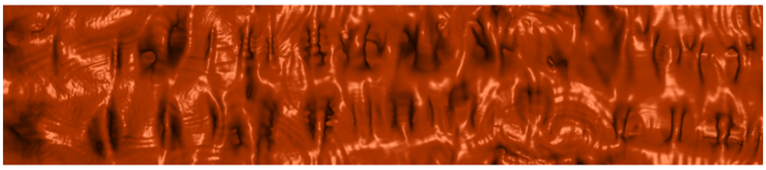
Surface flattening using conformal mapping [Hong Vis06]



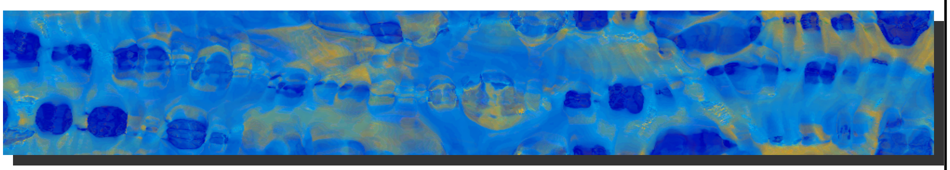
Now can visualize the entire colon as one image

Alternative to 3D Viewing: Practice

Surface flattening using conformal mapping:

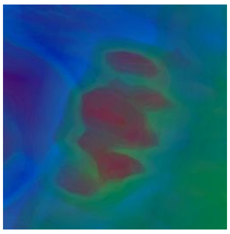


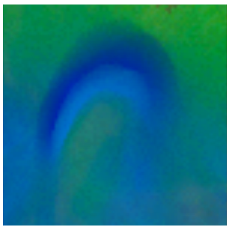
Rendered in translucent mode: [Hong Vis06]



Enables Quick Diagnosis

Only local inspection:

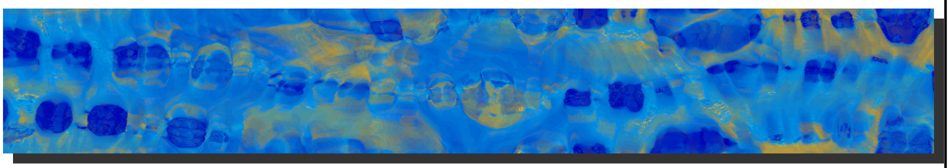




Hyperplastic polyp   Adenoma

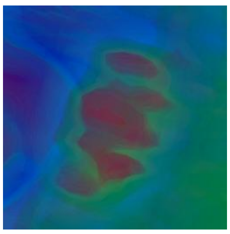
[Hong Vis06]

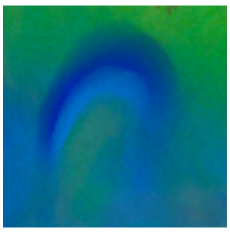
Flattening enables a *global* overview of pathology:



Enables Quick Diagnosis

Only local inspection:





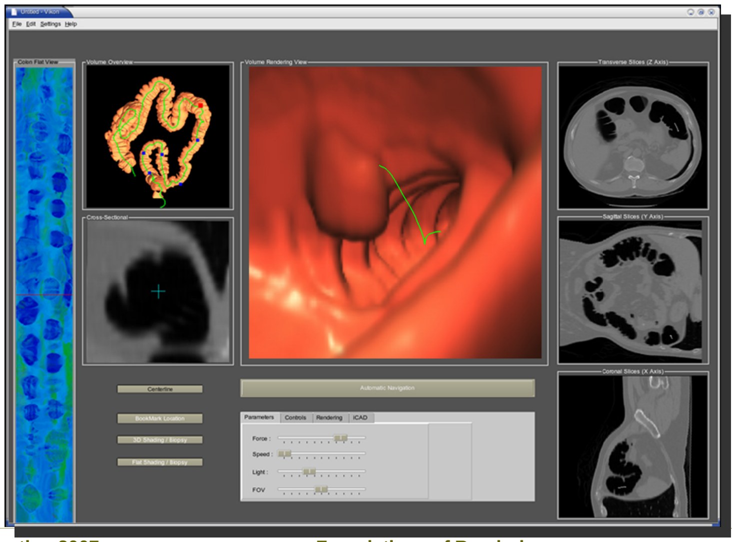
Hyperplastic polyp   Adenoma

[Hong Vis06]

CAD (Computer Aided Diagnosis):

Marked: suspected polyps

Actual Interface For Clinical Use



References (1)

[Bitter 00] I. Bitter, M. Sato, M. Bender, K. McDonnell, A. Kaufman, and M. Wan, "CEASAR: a smooth, accurate and robust centerline extraction algorithm," IEEE Visualization '00, pp. 45-52, 2000.

[Cohen 94] D. Cohen and Z. Sheffer, "Proximity clouds - an acceleration technique for 3D grid traversal," The Visual Computer, vol. 10, no. 11, pp. 27-38, 1994.

[Danskin 92] J. Danskin and P. Hanrahan, "Fast algorithms for volume ray tracing," Symp. Volume Visualization '92, pp. 91-98, 1992.

[Engel 01] K. Engel, M. Kraus, and T. Ertl, "High-Quality Pre-Integrated Volume Rendering Using Hardware-Accelerated Pixel Shading," Siggraph/Eurographics Workshop on Graphics Hardware 2001.

[Grimm 04] S. Grimm, S. Bruckner, A. Kanitsar, E. Gröller, "Memory Efficient Acceleration Structures and Techniques for CPU-based Volume Raycasting of Large Data," Symposium on Volume Visualization and Graphics, pp. 1-8, 2004.

[Hauser 01] H. Hauser, L. Mroz, G. Bischi, and E. Gröller, "Two-Level Volume Rendering," IEEE Trans. on Visualization Computer Grapics, vol. 7, no. 3, pp. 242-252, 2001.

[Hong 97] L. Hong, S. Muraki, A. E. Kaufman, D. Bartz, T. He, "Virtual voyage: interactive navigation in the human colon, SIGGRAPH '97, pp. 27-34, 1997.

References (2)

[Kaufman 05] A. Kaufman, S. Lakare, K. Kreeger, and I. Bitter, "Virtual colonoscopy," Communications of the ACM, vo. 48, no. 2, pp. pp. 37-41, 2005.

[Kaufman 05] A. Kaufman and K. Mueller, "Overview of volume rendering", The Visualization Handbook, C. Johnson and C. Hansen, editors, Academic Press, 2005

[Kindlmann 98] G. Kindlmann and J. Durkin, "Semi-automatic generation of transfer functions for direct volume rendering," Symp. Volume Visualization '98, pp. 79-86, 1998

[Kniss 02] J. Kniss, G. Kindlmann, and C. Hansen, "Multidimensional transfer functions for interactive volume rendering," IEEE Trans. Visualization and Computer Graphics, vol. 8, no. 3, pp. 270-285, 2002.

[Lakare 00] S. Lakare, M. Wan, M. Sato, and A. Kaufman, "3D digital cleansing using segmentation rays," IEEE Visualization '00, pp. 37-44, 2000.

[Lakare 03] S. Lakare, A. Kaufman, "Anti-Aliased Volume Extraction," Eurographics - IEEE TCVG Symposium on Visualization, 2003.

[Leung 06] W. Leung, N. Neophytou, and K. Mueller, "SIMD-aware raycasting," Volume Graphics 2006, pp. 59-62, August 2006.

[Li 03] W. Li, K. Mueller, and A. Kaufman, "Empty-space skipping and occlusion clipping for texture-based volume rendering," IEEE Visualization '03, pp. 317-325, October 2003.

References (3)

[Lorensen 87] W. Lorensen and H. Cline, "Marching Cubes: A high resolution 3D surface construction algorithm," Siggraph '87, pp. 163-169, 1987.

Mueller 99] K. Mueller, T. Möller, and R. Crawfis, "Splatting without the blur," Proceedings Visualization '99, pp. 363-371, 1999.

[Mueller 05] K. Mueller, S. Lakare, and A. Kaufman, "Volume exploration made easy using feature maps," Scientific Visualization: Extracting Information and Knowledge from Scientific Data Sets, G. Nielson, G.-P. Bonneau and T. Ertl, eds, Springer-Verlag, Heidelberg, Germany.

[Pfister 99] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler, "The VolumePro real-time ray-casting system," SIGGRAPH 1999, pp. 251-260, 1999.

[Rezk-Salama 00] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl, "Interactive Volume Rendering on Standard PC Graphics Hardware Using Multi-Textures and Multi-Stage-Rasterization," Eurographics/SIGGRAPH Workshop on Graphics Hardware '00, pp. 109-118, 2000.

[Srámek 00] M. Srámek and A. E. Kaufman, "Fast ray-tracing of rectilinear volume data using distance transforms," IEEE Trans. on Visualization and Computer Graphics, vol. 6, no. 3, pp. 236-252, 2000.

[Stegmaier 05] S. Stegmaier, M. Strengert, T. Klein, T. Ertl, "A Simple and Flexible Volume Rendering Framework for Graphics-Hardware-based Raycasting," Volume Graphics 2005, pp. 187-195, August 2005.

References (4)

[Rezk-Salama Vis 06] C. Rezk-Salama M. Keller, and P. Kohlmann, “High-level user interfaces for transfer function design with semantics," IEEE Visualization ’06 (IEEE Trans. Visualization and Computer Graphics), 2006.

[Hong Vis 06] W. Hong. F. Qiu, and A. Kaufman, “A pipeline for computer-aided polyp detection,” Splatting without the blur," IEEE Visualization ’06 (IEEE Trans. Visualization and Computer Graphics), 2006.

Acknowledgments

Funding provided by: NIH, NSF, ONR, DOE