

Real-Time Ray-Casting



Hanspeter Pfister

MERL - A Mitsubishi Electric Research Laboratory

201 Broadway

Cambridge, MA 02139

Email: pfister@merl.com

WWW: <http://www.merl.com/people/pfister/>

1

Outline



▶ Introduction

- Down the Ray-Casting Pipeline
 - ◆ Traversal order and memory organization
 - ◆ Processing units
- Case Study
 - ◆ The VolumePro Ray-Casting System

2

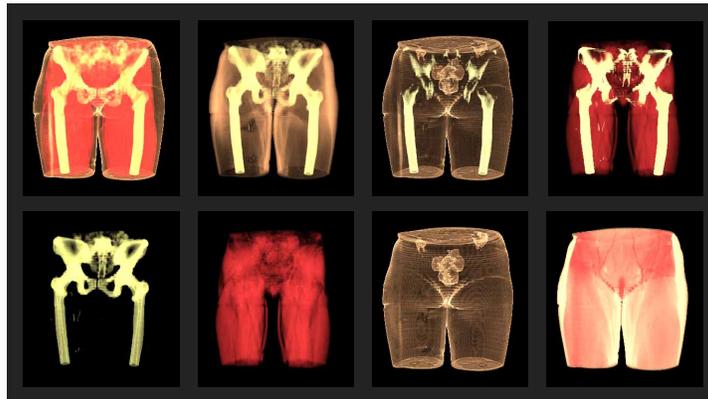
Why Real-Time Rendering?

- Fluid motion, increased depth perception
 - ◆ Movies: 24 frames / sec
 - ◆ Video: 30 frames / sec
- Real-time response to parameter changes
 - ◆ Makes it easier to deal with complicated visualization parameters such as transfer functions for color and opacity
- Real-time response to data changes
 - ◆ Interactive acquisition / simulation / modeling

3

Real-Time Parameter Changes

- Same data, different transfer functions

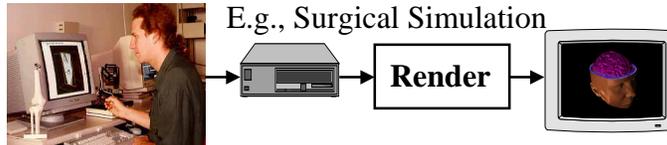


4

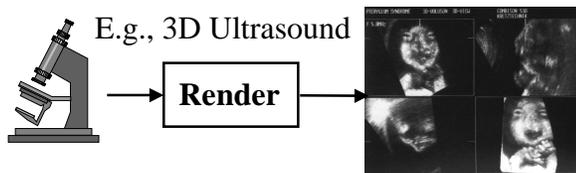
Real-Time Data Changes



- Simulation / visualization



- Acquisition / visualization



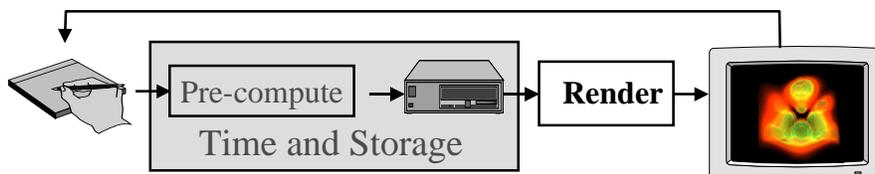
5

Software Rendering on PCs



- Around 10 M tri-linear samples / sec
- Speed through pre-computation and additional data storage
- Pre-computation prohibits real-time interaction with the data

Delayed Visual Feedback



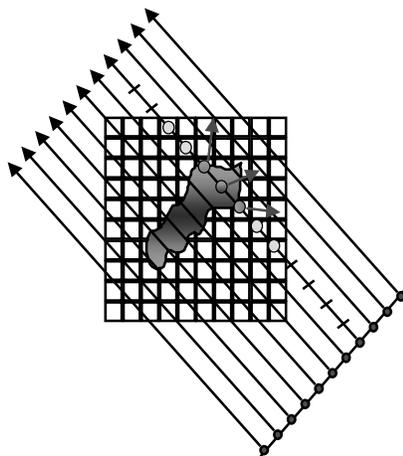
6

Why Ray-Casting?

- Best image quality of all direct volume rendering methods
 - ✦ Easily accommodates super-sampling
- Well understood
 - ✦ Parallel and perspective projections use the same framework
 - ✦ Many acceleration techniques such as early ray termination, adaptive sampling, space leaping, etc.
- Embarrassingly parallel

7

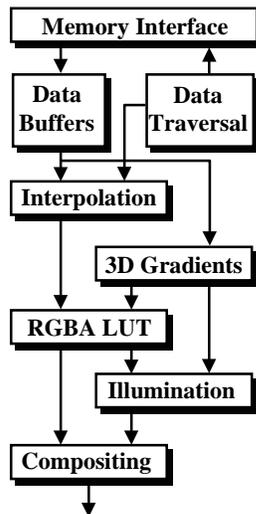
Volume Ray-Casting



- Data traversal
 - ✦ For each pixel, step along a ray
- Resampling
 - ✦ Tri-linear interpolation
- Classification
 - ✦ Assign RGBA to each sample
- Shading
 - ✦ Estimate gradients (normals)
 - ✦ Per-sample illumination
- Compositing
 - ✦ Blend samples into pixel color

8

The Ray-Casting Pipeline



- Data traversal
 - ✦ For each pixel, step along a ray
- Resampling
 - ✦ Tri-linear interpolation
- Classification
 - ✦ Assign RGBA to each sample
- Shading
 - ✦ Estimate gradients (normals)
 - ✦ Per-sample illumination
- Compositing
 - ✦ Blend samples into pixel color

9

Special-Purpose Hardware

- Research Projects
 - ✦ VIRIM - University of Mannheim, Germany
 - ✦ VOGUE - University of Tübingen, Germany
 - ✦ VIZARD - University of Tübingen, Germany
 - ✦ Cube - SUNY Stony Brook, USA
 - ✦ EM-Cube - MERL, USA
- Commercial Products
 - ✦ 3D Texture Hardware - OpenGL 1.1
 - ✦ VolumePro - Mitsubishi Electric

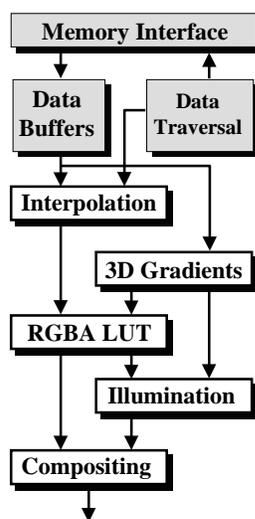
10

Outline

- Introduction
- ▶ Down the Ray-Casting Pipeline
 - Traversal order and memory organization
 - ◆ Processing units
- Case Study
 - ◆ The VolumePro Ray-Casting System

11

Down the Ray-Casting Pipeline



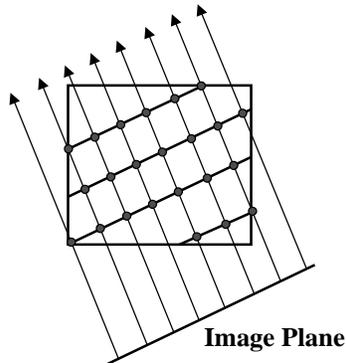
- It's the memory, stupid!
- The order of data access determines performance
 - ◆ Random access is deadly
- The memory organization should match the traversal order
- Buffering and reuse of data (at several stages in the pipeline) is crucial
- Ideally, access each voxel only once per frame

12

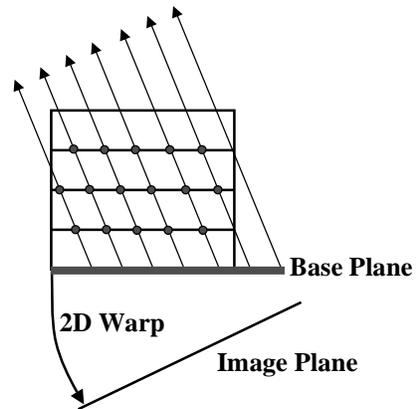
Data Traversal



Image-order

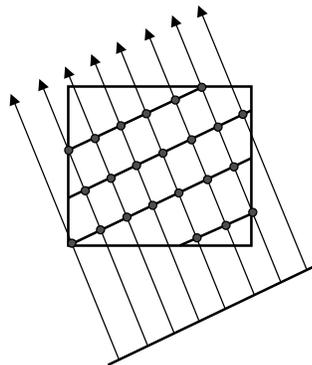


Object-order



13

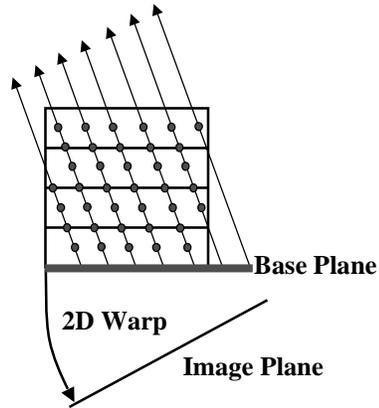
Image-Order Texture Slicing



- Volume is a 3D texture
- Generate polygons perpendicular to viewing direction
- Clip against volume bounding box
- Assign 3D texture coordinates to each vertex of the clipped polygons
- Project back-to-front using OGL blending operations

14

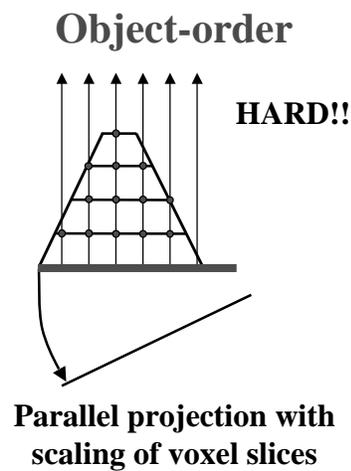
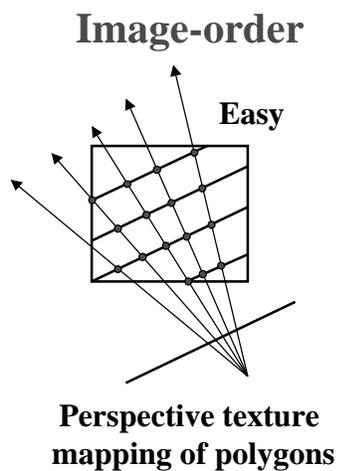
VolumePro Object-Order Ray-Casting



- Efficient object-order traversal of the data
- Cast rays from base plane pixels (not from image plane)
- Resample dataset on slices parallel to base plane
- Generate additional samples between voxel slices
- Final 2D warp onto image plane

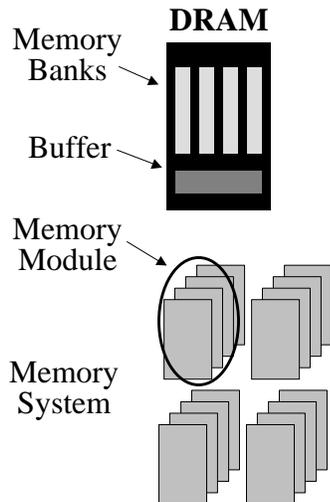
15

Perspective Projections



16

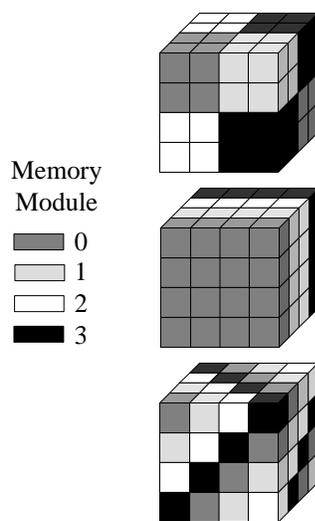
Anatomy of a Memory System



- Parallelism at several levels
 - ✦ Multiple banks per DRAM
 - ✦ Multiple DRAMs per Memory Module
 - ✦ Multiple Memory Modules per Memory System
- Use of fast internal DRAM buffers to “cache” data
- Problem: How to distribute the volume data across banks / DRAMs / modules?

17

Memory Organization

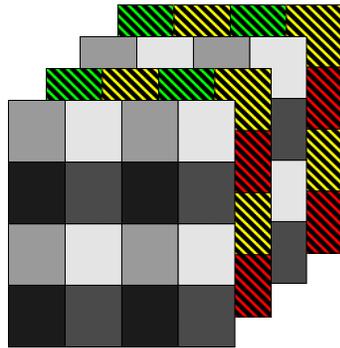


- 3D blocks of voxels
 - ✦ Typically 8 voxels of a tri-linear cell
- 2D slices of voxels
 - ✦ Access conflicts in one dimension
- 1D scanlines of voxels
 - ✦ Vector memory organization

18

8-way Interleaving

3D Texture Memory



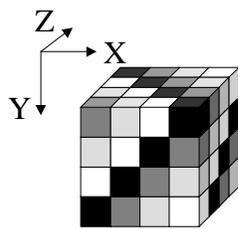
Memory Module

	0		4
	1		5
	2		6
	3		7

- Used in 3D texturing
- Texture slices are stored interleaved in memory
- Fast access to arbitrary 3D tri-linear cells
- Works well for image-order traversal
- Problem: Not scalable; more bandwidth requires memory replication

19

Vector Memory (3D Skewing)



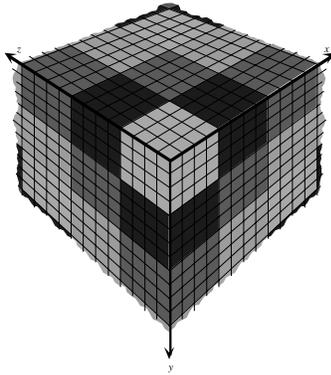
Memory Module

	0		2
	1		3

- Module # = $(x+y+z) \bmod n$
 (x,y,z) = Voxel coordinates
 n = Number of memory modules
- Conflict free access to scanlines of voxels
- Works well for object-order traversal
- Available bandwidth scales with number of memory modules
- Problem: Random access to voxels in each module

20

3D Skewing of Blocks



Memory Module



- Used in VolumePro
- Store voxels inside a block consecutively in memory
- Store blocks of voxels skewed in memory
- Adjacent blocks are always in different memory modules, independent of view-direction

21

Summary

Image-order

- Used in 3D texturing
- Perspective is easy
- No post filtering
- Random access to 3D tri-linear cells
- Typically 8-way interleaved memory
- More bandwidth through replication

Object-order

- Used in VolumePro
- Perspective is hard
- Post 2D warp
- Fast, streaming access to voxel scanlines
- Typically vector memory and skewing
- Bandwidth scales with number of modules

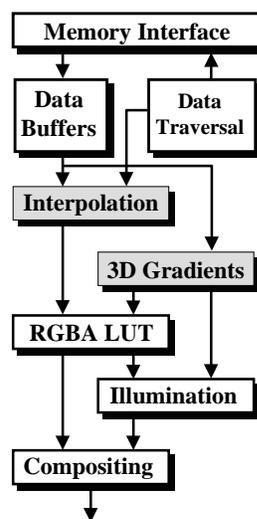
22

Outline

- Introduction
- ▶ Down the Ray-Casting Pipeline
 - ◆ Traversal order and memory organization
 - Processing units
- Case Study
 - ◆ The VolumePro Ray-Casting System

23

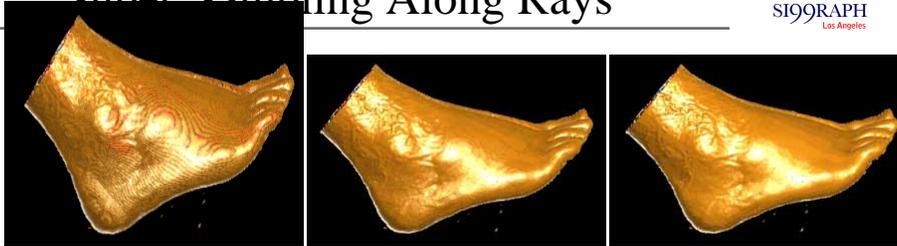
Down the Ray-Casting Pipeline



- Tri-linear interpolation is sufficient for α -blending
- High-quality iso-surfaces require root finding
 - ◆ Intersection of a ray with the cubic tri-linear function
- Central difference gradients are standard
 - ◆ Linear operations can be exchanged (interpolation \leftrightarrow central differences)

24

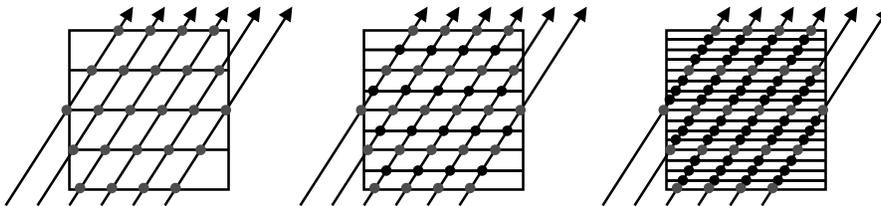
Super Sampling Along Rays



SS = 1

SS = 2

SS = 4



25

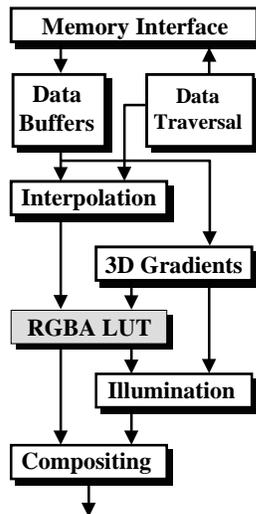
Gradient Computation



- 3D Texturing
 - ◆ Store additional data (e.g., store RGBA or an additional volume with quantized gradients)
 - ◆ Calculate gradients using multiple passes
- VolumePro
 - ◆ Calculate per-sample gradients in hardware
 - ◆ In Z: Central difference between voxels, tri-linear interpolation of gradient to the sample position
 - ◆ In X and Y: Tri-linear interpolation of voxels, central differences between tri-linear samples

26

Down the Ray-Casting Pipeline



- Order matters
 - ✦ Classification is non-linear
- Interpolate, then classify
 - ✦ Simpler, faster interpolation
 - ✦ It is better to first interpolate and *then* map to color
- Classify, then interpolate
 - ✦ For pre-classified data
- Lookup can be based on more than scalar sample values

27

Classification

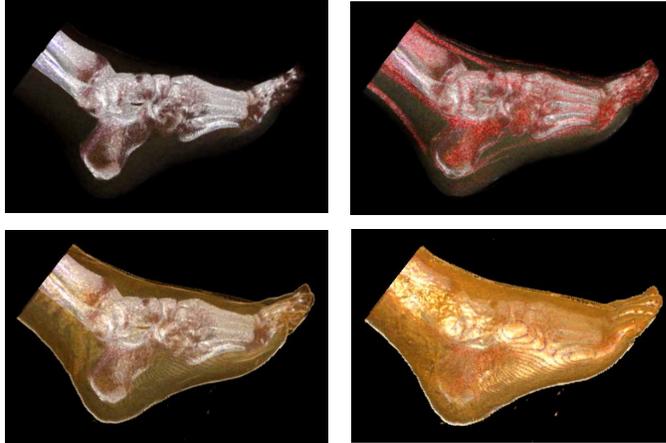
- 3D Texturing
 - ✦ Pre-classified RGBA volumes or RGBA color lookup before 3D texture is loaded
 - ✦ Texture slicing on RGBA textures is inaccurate for non-linear color mapping
- VolumePro
 - ✦ RGBA lookup based on interpolated samples
 - ✦ 12-bit sample \Rightarrow 24-bit RGB, 12-bit α
 - ✦ Double buffered 4k x 36-bit lookup tables

28

Real-Time Classification

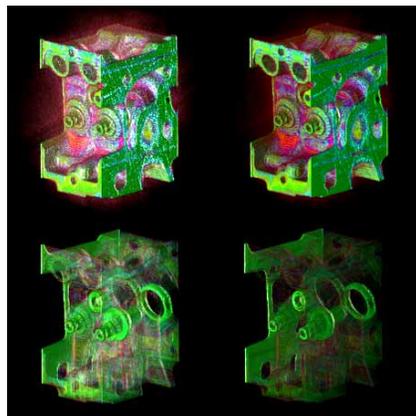


Interactive design of color and opacity transfer functions



29

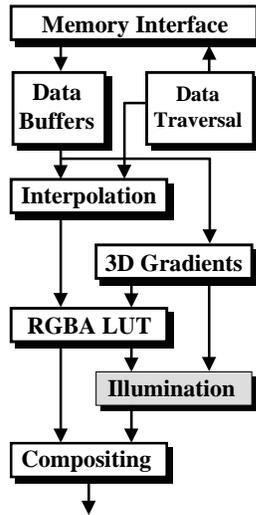
Gradient Magnitude Modulation



- Modulate opacity and illumination with the gradient magnitude
- Attenuate noisy or homogeneous regions with low gradient magnitudes
- Enhance areas of fast data changes with high gradient magnitudes

30

Down the Ray-Casting Pipeline



- Illumination greatly enhances depth perception
- Directional lighting requires 3D gradients
- Parallel light sources
 - ✦ Single light source at ∞
 - ✦ Light sources at 0° and 45°
 - ✦ Arbitrary light placement

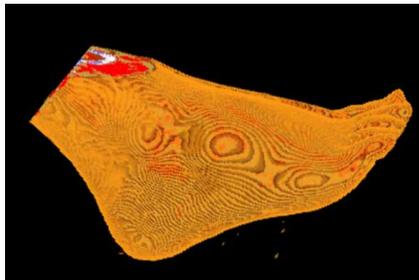
31

The Phong Illumination Model

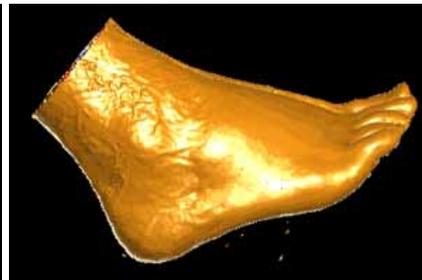
Emissive Diffuse Specular

$$\text{Color} = (k_e + k_d I_d) \text{ SampleColor} + k_s I_s \text{ SpecularColor}$$

No Illumination



Phong Illumination



32

3D Texture Hardware



$$I_d = (G \cdot L)$$



Courtesy of David Heath, Johns Hopkins University

- Gradient shading
 - ◆ Diffuse illumination with one parallel light source
 - ◆ Transform gradients to screen space
 - ◆ Tricks with color matrix
- Gradientless shading
 - ◆ Forward differences with respect to light direction
- Multi-pass methods

33

VolumePro Reflectance Maps



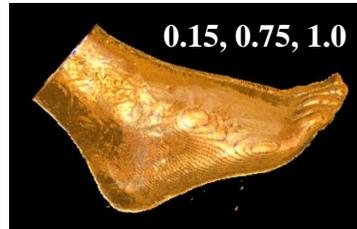
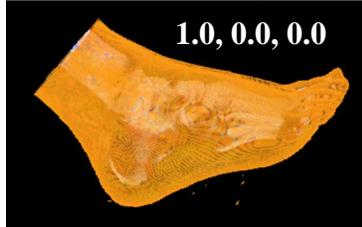
- Environment maps for pre-computed illumination values
 - ◆ I_d = diffuseLUT [Gradient]
 - ◆ I_s = specularLUT [Reflectance Vector]
- Illumination mapped on a unit cube
- Unlimited number of light sources
- Reflectance vector compute in hardware
- The diffuse and specular reflectance maps need to be recomputed if the lights change

34

VolumePro Phong Illumination

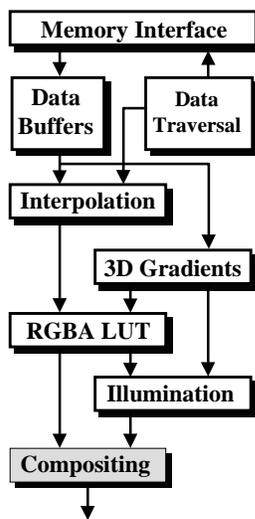


k_e, k_d, k_s



35

Down the Ray-Casting Pipeline



- Alpha blending (FTB)

$$C_{acc} += (1 - \alpha_{acc}) (\alpha_{samp} C_{samp})$$

$$\alpha_{acc} += (1 - \alpha_{acc}) \alpha_{samp}$$
- Maximum Intensity Projection (MIP)

if (sampleValue > maxValue)

$$C_{acc} = C_{samp}$$

$$maxValue = sampleValue$$
- Weighted sum / average

36

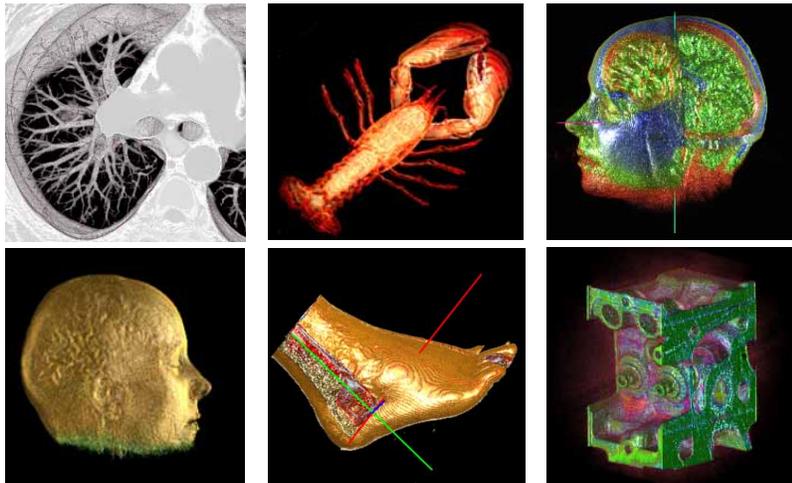
Outline

- Introduction
- Down the Ray-Casting Pipeline
 - ◆ Traversal order and memory organization
 - ◆ Processing units
- ▶ Case Study
 - ◆ The VolumePro Ray-Casting System

37

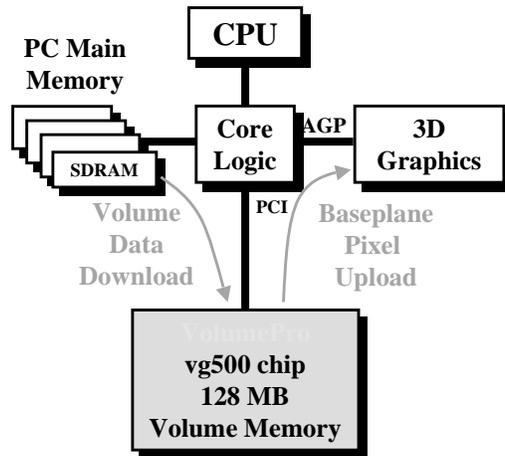
VolumePro

Single-chip real-time ray-casting engine for the PC



38

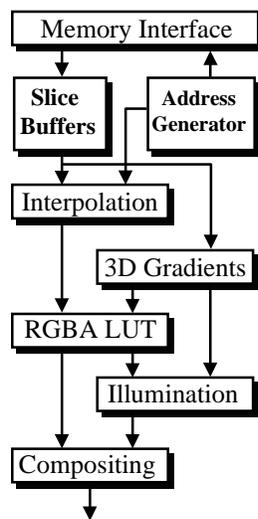
VolumePro System



- PCI card
- Requires companion 3D graphics card (AGP or PCI)
- Baseplane pixels are transferred to 3D card
- Post-warp and display using texture mapping on the 3D card

39

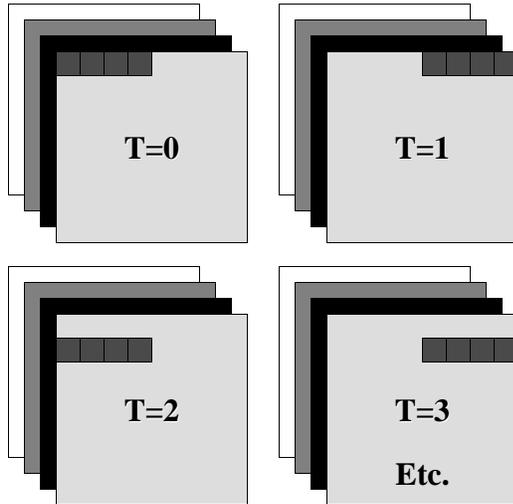
The vg500 Rendering Pipeline



- Four ray-casting pipelines on chip running @ 130 MHz
- Orthographic projections only
- 12-bit datapaths (16-bit memory interface)
- 4k RGB and opacity LUT
- Per-sample Phong illumination
- 12-bit front-to-back α -blending, MIP, MinIP
- 500 million samples / sec

40

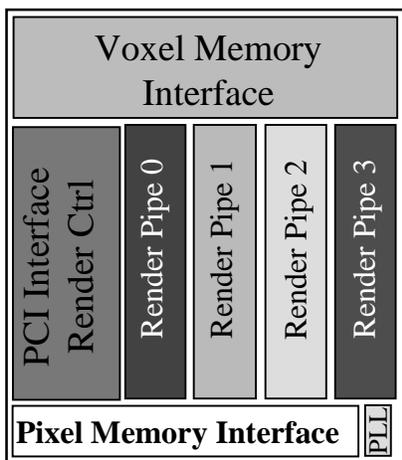
Parallel Slice-by-slice Processing



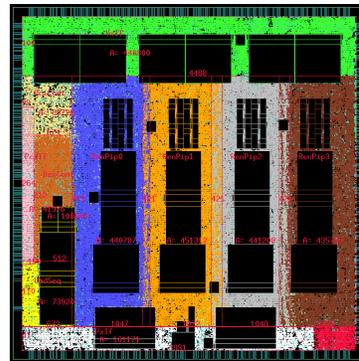
- Within a voxel scanline, four voxels are processed in parallel
- Within a slice, voxel scanlines are processed top to bottom
- Slices are processed front to back

41

vg500 Volume Rendering ASIC



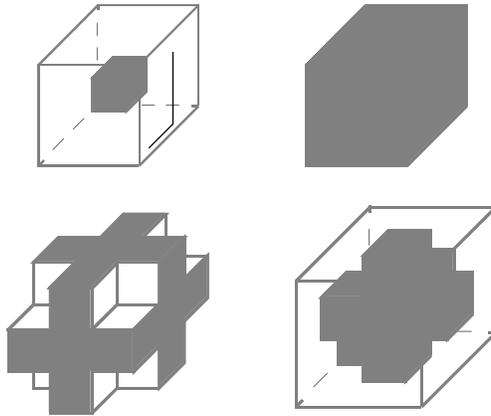
Four pipelines at 133 MHz



ASIC Floorplan

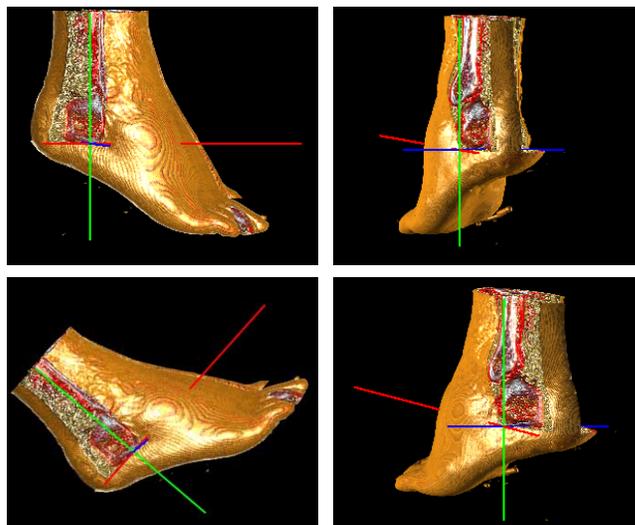
42

Different Cropping Modes



47

3D Line Cursor and Cropping



48

VLI - Volume Library Interface



- A set of C++ classes that provide full access to the vg500 chip features
 - ✦ VLILookupTable, VLICamera, VLICrop, VLICutPlane, VLILight, VLICursor, etc.
 - ✦ VLIVolume manages voxel data storage, data formats, and transformations
 - ✦ VLIContext is a container object for all attributes needed to render the volume
 - ✦ α correction, anisotropy and gantry tilt, super-volumes, partial volume updates, etc.

49

Summary



- Real-Time Ray-Casting is here!
 - ✦ 3D Texture Mapping
 - ✦ VolumePro
- There are major differences in both approaches
 - ✦ Different memory organizations
 - ✦ Different rendering pipelines
 - ✦ Different APIs
- Volume Graphics will become an integral part of 3D graphics systems

50



More Information

- SGI Volumizer
 - ◆ <http://www.sgi.com/software/volumizer/>
- VolumePro
 - ◆ <http://www.3dvolumegraphics.com/>
 - ◆ 3dvolumegraphics@mea.com

51



Many thanks to...

- Forrester Cole, Harvard University
- Jan (“YON”) Hardenbergh, MERL
- Ray Jones, MERL
- Arie Kaufman, SUNY Stony Brook
- The VolumePro Development Team

52

Bibliography



Volume Rendering Hardware Overviews

- H. Ray, H. Pfister, D. Silver, T. A. Cook, "Ray-Casting Architectures for Volume Visualization", To appear in *IEEE Transactions on Visualization and Computer Graphics*, 1999.
- H. Pfister, "Architectures for Real-Time Volume Rendering", *Journal of Future Generation Computer Systems* (FGCS), pages 1-9, Vol. 15, 1999, Elsevier Science.
- J. Hesser, R. Maenner, G. Knittel, W. Strasser, H. Pfister, and A. Kaufman. "Three architectures for volume rendering". In *Proceedings of Eurographics '95*, pages C-111-C-122, Maastricht, The Netherlands, 1995.

Volume Rendering with 3D Texture Mapping

- A. van Gelder, K. Kim, "Direct Volume rendering with Shading via Three-Dimensional textures", In *1996 Workshop on Volume Visualization*, pages 23-30, San Francisco, CA, 1996.
- K. Akeley. "RealityEngine Graphics". In *Computer Graphics*, Proceedings of SIGGRAPH 93, pages 109-116, 1993.
- B. Cabral, N. Cam, and J. Foran. "Accelerated volume rendering and tomographic reconstruction using texture mapping hardware". In *1994 Workshop on Volume Visualization*, pages 91-98, Washington, DC, 1994.
- T. J. Cullip and U. Neumann. "Accelerating volume reconstruction with 3D texture mapping hardware". Technical Report TR93-027, Department of Computer Science at the University of North Carolina, Chapel Hill, 1993.

53

Bibliography (cont.)



VolumePro

- H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler, The VolumePro Real-Time Ray-Casting System, To Appear in *Proceedings of SIGGRAPH 1999*.
- R. Osborne, H. Pfister, H. Lauer, N. McKenzie, S. Gibson, W. Hiatt, and T. Ohkami. "EM-Cube: An architecture for low-cost real-time volume rendering". In *Proceedings of the Siggraph/Eurographics Workshop on Graphics Hardware*, August 1997.
- H. Pfister, "Architectures for Real-Time Volume Rendering", Ph.D. Thesis, State University of New York at Stony Brook, December 1996.
- H. Pfister and A. Kaufman. "Cube-4 - A scalable architecture for real-time volume rendering". In *1996 ACM/IEEE Symposium on Volume Visualization*, pages 47-54, San Francisco, CA, October 1996.
- R. Yagel and A. Kaufman. "Template-based volume viewing". *Computer Graphics Forum, Proceedings Eurographics*, 11(3):153-167, September 1992.
- A. Kaufman and R. Bakalash. "Memory and processing architecture for 3D voxel-based imagery". *IEEE Computer Graphics & Applications*, 8(6):10-23, November 1988.

54