Advanced Illumination Techniques for GPU-Based Volume Raycasting

Markus Hadwiger
VRVis Research Center
Vienna, Austria

Patric Ljung
Siemens Corporate Research
Princeton, NJ, USA

Christof Rezk Salama
Computer Graphics Group
Institute for Vision and Graphics
University of Siegen, Germany

Timo Ropinski
Visualization and Computer Graphics Research Group,
University of Münster, Germany
Ray Casting Basics

Markus Hadwiger
VRVis Research Center
Vienna, Austria

Patric Ljung
Siemens Corporate Research
Princeton, NJ, USA

Christof Rezk Salama
Computer Graphics Group
Institute for Vision and Graphics
University of Siegen, Germany

Timo Ropinski
Visualization and Computer Graphics Research Group,
University of Münster, Germany
ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

CT Human Head: Visible Human Project, US National Library of Medicine, Maryland, USA

CT Angiography: Dept. of Neuroradiology, University of Erlangen, Germany
Engineering

Computational Fluid Dynamics (CFD)

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

Eurographics 2009
Materials Science, NDT

Micro CT, Compound Material,
Material Science Department,
University of Erlangen

Biology

biological sample of the soil, CT,
Virtual Reality Group,
University of Erlangen

Hinge Bearing,
Austrian Foundry Research Institute

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

Eurographics 2009
Archaeology

Hellenic Statue of Isis
3rd century B.C.
ARTIS, University of Erlangen-Nuremberg, Germany

Sotades Pygmaios Statue,
5th century B.C
ARTIS, University of Erlangen-Nuremberg, Germany
Special Effects and Games

Clouds and Atmospheric Scattering

Dobashi et al.

Fire and Explosions

Krüger and Westermann
Advanced Lighting

- Shadows and scattering
Advanced Lighting

MRI Brain

- no shading
- gradient shading
- shadows + scattering
Advanced Lighting
Advanced Lighting

Industrial CT

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

Eurographics 2009
Ray Integration

How do we determine the radiant energy along the ray?

**Physical model:** emission and absorption, no scattering

$$I(s) = I(s_0) e^{-\tau(s_0, s)}$$

- **Initial intensity** at $s_0$
- **Absorption along the ray segment** $s_0 - s$
- **Extinction** $\tau$
- **Absorption** $\kappa$

Without absorption all the initial radiant energy would reach the point $s$. 

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

Eurographics 2009
How do we determine the radiant energy along the ray?

*Physical model: emission and absorption, no scattering*

Every point along the viewing ray emits additional radiant energy.

\[
I(s) = I(s_0) e^{-\tau(s_0,s)} + \int_{s_0}^{s} q(\tilde{s}) e^{-\tau(\tilde{s},s)} d\tilde{s}
\]
Volume Rendering

*Image order approach:*

*For each pixel {*
  *
calculate color of the pixel
*
}
Volume Rendering

Object order approach:

For each pixel {
    calculate color of the pixel
}

Image Plane

Data Set

Eye

For each slice {
    For each pixel {
        calculate contribution to the image
    }
    calculate color of the pixel
}
Why Ray Casting on GPUs?

- Most GPU rendering is object-order (rasterization)

- Image-order is more “CPU-like”
  - Simpler to implement
  - Very flexible
    (adaptive sampling, ...)

- Correct perspective
- Single pass ray casting
- 32-bit compositing
Recent GPU Approaches

- **Rectilinear grids**
  - [Krüger and Westermann, 2003]
  - [Röttger et al., 2003]
  - [Green, 2004] (in NVIDIA SDK)
  - [Stegmaier et al., 2005]
  - [Scharsach et al., 2006]
  - [Gobbetti et al., 2008]

- **Unstructured (tetrahedral) grids**
  - [Bernardon et al., 2004]
  - [Callahan et al., 2006]
  - [Muigg et al., 2007]
Correct Perspective

- Entering the volume
- Wide field of view
- Fly-throughs
- Virtual endoscopy
- Integration into perspective scenes: games, ...
Single-Pass Ray Casting

- Enabled by conditional loops in fragment shaders (Shader Model 3.0 and higher)
- Substitute multiple passes and early-z testing by single loop and early loop exit
- No compositing buffer: full 32-bit precision!

- NVIDIA SDK example: compute ray intersections with bounding box, march along rays and composite

- Volume rendering example in NVIDIA CUDA SDK
Basic Ray Setup / Termination

Two main approaches:

- Procedural ray/box intersection
  [Röttger et al., 2003], [Green, 2004]

- Rasterize bounding box
  [Krüger and Westermann, 2003]

Either:

- Ray start position and exit check
- Ray start position and exit position
- Ray start position and direction vector
Procedural Ray Setup / Term.

- Procedural ray / box intersection
  - Everything handled in fragment shader

- Ray given by camera position and volume entry position
- Exit criterion needed

- Pro: simple and self-contained
- Con: full load on fragment shader
Fragment Shader

- Rasterize front faces of bounding box
- Texcoords are volume position in [0,1]
- Subtract camera pos
- Accumulate/composite
- Repeatedly check for exit of bounding box
"Image-Based" Ray Setup / Term.

- Rasterize bounding box front faces and back faces
- Ray start positions: front faces
- Direction vectors: back faces – front faces
- Independent of projection (orthogonal/perspective)
Early ray termination
- Isosurfaces:
  stop when surface hit
- Direct volume rendering:
  stop when opacity >= threshold

Several possibilities
- Older GPUs (before shader model 3):
  multi-pass rendering with early-z test
- Shader model 3: break out of ray-casting loop
- Current GPUs: early loop exit works well
Empty space skipping
- Skip transparent samples
- Depends on transfer function
- Start casting close to first hit

Several possibilities
- Per-sample check of opacity (expensive)
- Traverse regular grid or hierarchy (e.g., octree with stack-less traversal [Gobbetti et al., 2008])

These are image-order:
what about object-order?
Modify initial rasterization step

rasterize bounding box  rasterize "tight" bounding geometry
Store min-max values of volume blocks
- Cull blocks against transfer function or isovalue
- Rasterize front and back faces of active blocks

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING
Object-Order Empty Space Skip. (3)

- Rasterize front and back faces of active min-max blocks
- Start rays on block front faces
- Terminate when
  - Full opacity reached, or
  - Back face reached
Object-Order Empty Space Skip. (3)

- Rasterize front and back faces of active min-max blocks
- Start rays on block front faces
- Terminate when
  - Full opacity reached, or
  - Back face reached

- Not all empty space is skipped
Scene Integration (1)

- Build on image-based ray setup
- Allow viewpoint inside the volume

- Intersect polygonal geometry
Scene Integration (2)

- Near clipping plane clips into front faces

- Fill in holes with near clipping plane
- Can use depth buffer [Scharsach et al., 2006]
Scene Integration (3)

1. Starting position computation
   ⇒ Ray start position image

2. Ray length computation
   ⇒ Ray length image

3. Render polygonal geometry
   ⇒ Modified ray length image

4. Raycasting
   ⇒ Compositing buffer

5. Blending
   ⇒ Final image
Virtual Endoscopy

- Viewpoint inside the volume with wide field of view
  - E.g.: virtual colonoscopy

- Hybrid isosurface rendering / direct volume rendering
  - E.g.: colon wall and structures behind
Virtual Colonoscopy

- First find isosurface; then continue with DVR
Virtual Colonoscopy

First find isosurface; then continue with DVR
Isosurfaces/Level Sets
- scanned data
- distance fields
- CSG operations
- level sets: surface editing, simulation, segmentation, …

Isosurface Ray Casting
Intersection Refinement (1)

- Fixed number of bisection or binary search steps
- Virtually no impact on performance

- Refine already detected intersection

- Handle problems with small features / at silhouettes with adaptive sampling
Intersection Refinement (2)

without refinement

with refinement

sampling rate 1/5 voxel (no adaptive sampling)
Intersection Refinement (3)

Sampling distance 1.0, 24 fps

Sampling distance 5.0, 66 fps

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING

Eurographics 2009
Deferred Isosurface Shading

- Shading is expensive
- Full ray casting step computes only intersection image
Memory Management

What happens if data set is too large to fit into local GPU memory?

Divide data set into smaller chunks (bricks)

One plane of voxels must be duplicated for correct interpolation across brick boundaries.

incorrect interpolation!
Bricking

- Combine bricks for memory management
- Smaller blocks for object-order empty space skipping
Bricked Single-Pass Casting (1)

- Duplicate neighbor voxels for filtering
- Store $n^3$ bricks as $(n+1)^3$
  - 10% overhead with $32^3$ bricks
- Pack needed bricks into single 3D texture
Bricked Single-Pass Casting (2)

- Layout/index texture for addr. translation
- Supports multi-resolution rendering
- Map virtual volume coords to physical tex
## Flat vs. Hierarchical Bricking

<table>
<thead>
<tr>
<th></th>
<th>Flat</th>
<th>Hierarchical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of bricks</td>
<td>↔</td>
<td>↓</td>
</tr>
<tr>
<td>Texture size of brick</td>
<td>↓</td>
<td>↔</td>
</tr>
<tr>
<td>Physical extent of brick</td>
<td>↔</td>
<td>↑</td>
</tr>
</tbody>
</table>
Shadow Memory Management (1)
Shadow Memory Management (2)
Shadow Memory Management (3)
Adaptive Volume Sampling

ADVANCED ILLUMINATION TECHNIQUES FOR GPU-BASED VOLUME RAYCASTING
Adaptive Volume Sampling

Full density

Adaptive density

Speed-up: 2.6-2.8
Conclusions

- Ray casting has become the most important GPU volume rendering technique
  - Very flexible and easy to implement
  - Now with advanced lighting in real time

- Mixing image-order and object-order approaches is well suited to GPUs

- Flexible memory management for both rendering and lighting
Thank You!

Acknowledgments

Christof Rezk-Salama, Patric Ljung, Henning Scharsach, Daniel Weiskopf