Announcements

- Second midterm grades will be on-line by December 2\textsuperscript{nd}.
- Please check Gradesource for accuracy. Homework assignments 1-6 and midterm #1 should be complete.
- Final project to be presented on \textbf{Friday, Dec 3\textsuperscript{rd}, between 2 and 4pm in room 4140}
  - No late submissions accepted.
Lecture Overview

- Shadow Volumes
- Volume Rendering
Shadow Volumes

NVIDIA md2shader demo
A single point light source splits the world in two:
- Shadowed regions
- Unshadowed regions
- Volumetric shadow technique

A shadow volume is the boundary between these shadowed and unshadowed regions.
- Determine if an object is inside the boundary of the shadowed region and know the object is shadowed.
Shadow Volumes

- Many variations of the algorithm exist
- Most popular ones use the stencil buffer
  - Depth Pass
  - Depth Fail (a.k.a. Carmack’s Reverse, developed for Doom 3)
  - Exclusive-Or (limited to non-overlapping shadows)
- Most algorithms designed for hard shadows
- Algorithms for soft shadows exist
Shadow Volumes

- Shadowing object
- Light source
- Surface outside shadow volume (illuminated)
- Surface inside shadow volume (shadowed)
- Partially shadowed object
- Shadow volume (infinite extent)

Eye position (note that shadows are independent of the eye position)
Shadow Volume Algorithm

- High-level view of the algorithm
  - Given the scene and a light source position, determine the geometry of the shadow volume
  - Render the scene in two passes
    - Draw scene with the light enabled, updating only fragments in unshadowed region
    - Draw scene with the light disabled, updated only fragments in shadowed region
Shadow Volume Construction

- Need to generate shadow polygons to bound shadow volume
- Extrude silhouette edges from light source

Extruded shadow volumes
Shadow Volume Construction

- Done on the CPU
- Silhouette edge detection
  - An edge is a silhouette if one adjacent triangle is front facing, the other back facing with respect to the light
- Extrude polygons from silhouette edges
Stenciled Shadow Volumes

- **Advantages**
  - Support omnidirectional lights
  - Exact shadow boundaries

- **Disadvantages**
  - Fill-rate intensive
  - Expensive to compute shadow volume geometry
  - Hard shadow boundaries, not soft shadows
  - Difficult to implement robustly
Tagging Pixels as Shadowed or Unshadowed

- **The stenciling approach**
  - Clear stencil buffer to zero and depth buffer to 1.0
  - Render scene to leave depth buffer with closest Z values
  - Render shadow volume into frame buffer with depth testing but *without* updating color and depth, but *inverting* a stencil bit (Exclusive-Or method)
  - This leaves stencil bit set within shadow
Stencil Inverting of Shadow Volume

- Light source
- Eye position
- Shadowing object
- Two inverts, left zero
- One invert, left one
- Zero inverts, left zero
Visualizing Stenciled Shadow Volume Tagging

Shadowed scene

Stencil buffer contents

red = stencil value of 1
green = stencil value of 0

GLUT shadowvol example credit: Tom McReynolds
For Shadow Volumes With Intersecting Polygons

- Use a stencil enter/leave counting approach
  - Draw shadow volume twice using face culling
    - 1st pass: render *front* faces and *increment* when depth test passes
    - 2nd pass: render *back* faces and *decrement* when depth test passes
  - This two-pass way is more expensive than invert
  - Inverting is better if all shadow volumes have no polygon intersections
Increment/Decrement Stencil Volumes

Light source

Shadowing object

Eye position

0
+1
+2
+3
+2
+1

zero
zero
Shadow Volume Demo

- URL:
  
  http://www.paulsprojects.net/opengl/shadvol/shadvol.html
Resources

- Overview, lots of links
  http://www.realtimerendering.com/
- Basic shadow maps
- Avoiding sampling problems in shadow maps
  http://www.cg.tuwien.ac.at/research/vr/lispsm/
- Faking soft shadows with shadow maps
  http://people.csail.mit.edu/ericchan/papers/smoothie/
- Alternative: shadow volumes
  http://www.gamedev.net/reference/articles/article1873.asp
Lecture Overview

- Shadow Volumes
- Volume Rendering
What is Volume Rendering

- A *Volume* is a 3D array of voxels (volume elements, 3D equivalent of pixels)
- 3D images produced by CT, MRI, 3D mesh-based simulations are easily represented as volumes
- The *Voxel* is the basic element of the volume
  Typical volume size may be $128^3$ voxels, but any other size is acceptable.
- *Volume Rendering* means rendering the voxel-based data into a viewable 2D image.
Volume Data Types

- 3D volume data are represented by a finite number of cross-sectional slices (3D grid)
- Each voxel stores a data value
  - Single bit: binary data set
  - Typical: 8 or 16 bit integers
  - Simulations often generate floating point
  - Sometimes multi-valued (multiple data values per voxel), for instance RGB, multi-channel confocal microscopy
Applications: Medicine

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

CT Angiography:
Dept. of Neuroradiology
University of Erlangen, Germany
Applications: Geology

Deformed Plasticine Model,
Applied Geology,
University of Erlangen

Muschelkalk:
Paläontologie,
Virtual Reality Group,
University of Erlangen
Applications: 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
Applications

Material Science, Quality Control

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

Biology

Biological sample of soil, CT
Virtual Reality Group, University of Erlangen
Applications

Computational Science and Engineering
Methods of Representation

- Polygonal - Triangle Mesh
- Freeforms - parametric curves, patches...
- Solid Modelling - CGS (Constructive Solid Geometry)
- Direct Volume Rendering
Why Direct Volume Rendering?

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural representation of CT/MRI images</td>
<td>Huge data sets</td>
</tr>
<tr>
<td>Transparency effects (Fire, Smoke...)</td>
<td>Computationally expensive</td>
</tr>
<tr>
<td>High quality</td>
<td>Cannot be embedded easily into polygonal scene</td>
</tr>
</tbody>
</table>
Volume Rendering: Demo

Virvo URL:
http://www.calit2.net/~jschulze/projects/vox/
Volume Rendering Outline

Data Set → 3D Rendering → Classification

in real-time on commodity graphics hardware
Rendering Methods

There are two categories of volume rendering algorithms:

1. Ray casting algorithms (Object Order)
   - Basic ray-casting
   - Using octrees

2. Plane Composing (Image Order)
   - Basic slicing with 2D textures
   - Shear-Warp factorization
   - Translucent textures with image-aligned 3D textures
Ray Casting

- Software Solution

![Diagram](image)

- **Numerical Integration**
- **Resampling**
- **High Computational Load**
Next Lecture

- Midterm review
- Final project Q&A
- Volume Rendering Part 2