Managing Scene Complexity

Thomas Funkhouser
Interactive Visualization

Render images with interactive control of viewpoint

Boeing
Mechanical CAD

SGI
Medicine

E&S
Driving Simulation

Disney
Entertainment

Avery Fisher
Architectural CAD

UBC
Education
Complexity of Massive Models

Examples:

- Automobile: ~20,000 parts
- Boeing Airplane: ~2,000,000 parts
- Aircraft Carrier: ~20,000,000 parts
- Outdoor Environments
Prototype Models

Cassini spacecraft

349,281 faces
127 objects
Architectural Models

Soda Hall Model (7.6M polygons)
Coal-Fired Powerplant: 15 million triangles
Mechanical CAD Models

Submarine Torpedo Room (850K polygons)
Mechanical CAD Models

82 million triangles; 126,000 objects

Newport News Shipbuilding
Interactive Visualization Systems

Frame rate:
- >10-60 frames per second

Latency
- <10-500 milliseconds response delay

Realism
- Realistic enough to convey information
Rendering Acceleration Techniques

Visibility Culling

• Backface culling, view-frustum culling, potentially-visible-sets, occlusion culling, ...

Detail Elision

• Levels of detail, multiresolution, ...

Image-Based Rendering

• Texture-maps, imposters, light fields, ...
Visibility Culling

Quickly eliminate large portions of the scene that will not be visible in the final image

- Not the exact visibility solution, but a quick and conservative test to reject primitives that are not visible
  - Trivially reject stuff that is obviously not seen
  - Use Z-buffer and clipping for the exact solution
Visibility Culling

Basic idea: don’t render what can’t be seen

- Off-screen: view-frustum culling
- Occluded by other objects: occlusion culling
View-Frustum Culling

Don’t draw primitives outside the view frustum

• Organize primitives into clumps
• Before rendering the primitives in a clump, test their bounding volume against the view frustum
Hierarchical bounding volumes

- If a clump is entirely outside or entirely inside view frustum, no need to test its children
Uniform Grid Subdivision

View Frustum
Octree Subdivision
View-Frustum Culling

Hierarchical bounding volumes

- If a clump is entirely outside or entirely inside view frustum, no need to test its children
Hierarchical View Frustum Culling

Use hierarchy to accelerate rendering

- Check contents visibility only if bounding volume is visible
Hierarchical View-Frustum Culling

What shape should the bounding volumes be?

- Spheres and axis-aligned bounding boxes:
  - Simple to calculate/test
  - May be poor approximation
- Convex hulls:
  - More complex to calculate/test
  - Tighter approximation
Occlusion Culling

Blue parts: occluders
Red parts: occludees
Occlusion Culling

Object-precision
- Cells and portals
- Shadow volumes

Image-precision
- OpenGL occlusion test
- Hierarchical Z-buffer
- Hierarchical occlusion maps
Cells and Portals

Subdivide space into cells.

Cluster polygons into objects.

Build an index of objects incident upon each cell.
Cells and Portals

Source Cell

Visible cells
(All cells reached by sightline from region)

Visible Objects
(All objects incident upon brown visibility beams)
Cells and Portals

Visible Cells

Visible Objects

Observer
Occlusion Culling

Object-precision
- Cells and portals
- Shadow volumes

Image-precision
- OpenGL occlusion test
- Hierarchical Z-buffer
- Hierarchical occlusion maps
OpenGL Occlusion Test

Hardware returns how many z-buffer tests pass.
Hierarchical Z-Buffer

Store z-buffer as pyramid and test depth hierarchically.
Rendering Acceleration Techniques

Visibility Culling
- Backface culling, view-frustum culling, potentially-visible-sets, occlusion culling, ...

Detail Elision
- Levels of detail, multiresolution, ...

Image-Based Rendering
- Texture-maps, imposters, light fields, ...
Detail Elision

Triangles:
41,855
27,970
20,922
12,939
8,385
4,766

courtesy of Division and Viewpoint
Levels of Detail

Pre-process
• Generate discrete set of independent levels of detail

Run-time
• Select level of detail according to viewpoint

Advantages
• Fairly efficient storage (2x original)
• No significant run-time overhead

Disadvantages
• Requires per-object simplification
• Not good for spatially large objects
Levels of Detail

597 Polygons
211 Polygons
51 Polygons
28 Polygons

889 Polygons
241 Polygons
97 Polygons
40 Polygons
Levels of Detail

69 k tris

11 k tris

2 k tris

575 tris
**Multiresolution Surfaces**

**Pre-process**
- Generate tree of simplification operations

**Run-time**
- Refine/coarsen current model according to viewpoint

**Advantages**
- Allows finer control of tessellation

**Disadvantages**
- More run-time computation and complexity
- Difficult for retained-mode graphics
Selecting Levels of Detail

Two possibilities:

• Guarantee quality, maximize frame rate
• Guarantee frame rate, maximize quality
Selecting Levels of Detail

\[ p = \frac{\varepsilon r}{w} = \frac{\varepsilon r}{2d \tan\left(\frac{\theta}{2}\right)} \]
Guaranteeing Frame Rate

No Detail Elision
0.22 Seconds
(19,881 Polygons)

Optimization Detail Elision
0.05 Seconds
(3,568 Polygons)
Guaranteeing Frame Rate

Objects Shaded by LOD
(Higher LODs appear darker)

Pixel-by-Pixel Differences
(Larger differences appear brighter)
Image-Based Representations
Interactive Visualization System

Database Management
- Lookahead Pre-Fetch
- Cache Management

Database Organization
- I/O Operations

Display Management
- Visibility Determination
- Detail Elision

User Interface

Mouse

Monitor

Database
## Interactive Visualization Results

<table>
<thead>
<tr>
<th>Cull Method</th>
<th># Polygons</th>
<th>% Model</th>
<th>Frame Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Avg</td>
<td>Max</td>
<td>Avg</td>
</tr>
<tr>
<td>Entire Model</td>
<td>1,418,807</td>
<td>1,418,807</td>
<td>100.0%</td>
</tr>
<tr>
<td>Visible Objects</td>
<td>5,002</td>
<td>45,828</td>
<td>0.35%</td>
</tr>
<tr>
<td>Detail Elision</td>
<td>2,534</td>
<td>6,912</td>
<td>0.18%</td>
</tr>
</tbody>
</table>

Per frame statistics collected during walk along test path through Soda Hall.