

ECE 595 / CS 491 / CS 591
**Real-Time Rendering &
Graphics Hardware**

Pradeep Sen
Advanced Graphics Lab

Class 6
February 5, 2007

Announcements

- Intro project due Wednesday by midnight. E-mail me a zipped version of your code along with the project write-up.
- Graphics Hardware 2007, San Diego (deadline April 9th)
- Eurographics Symposium on Rendering, Grenoble, France (deadline April 10th)

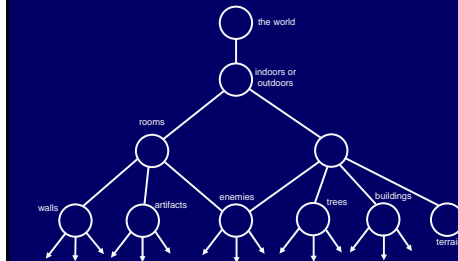
Last time

- Portals & occlusion testing
- Scene graphs

Today

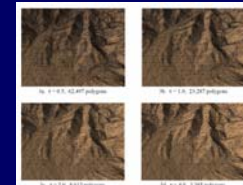
- Graphics engines
- Real-time graphics APIs (OpenGL)

Scene Graphs



Terrain

- Stored as a height-field
- Can be rendered using quad trees
- Level-of-detail algorithms



Trees

- Can use billboard techniques



Game engines

- Typically incorporate the following elements:
 - Rendering
 - Visibility
 - Collision detection
 - AI
 - Networking
 - File/asset management
 - Physics

Game engine rendering

- PVS determination, culling and LOD
- Surface shading (vertex and fragment)
- Per-pixel lighting
- Bumpmapping
- Scene Lighting
- Shadows
- Translucency
- Glowing and other post-processing effects

Game engines

- If you work in game development, you won't be creating a game engine from scratch
- Instead, you will probably be working a commercial engine:
 - Ogre
 - Torque
 - Gamebryo
 - Unreal

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007


Ogre engine

- Not a game engine, but a 3D rendering engine
- Free, open source! (GNU Lesser General Public License)
- Supports vertex & fragment programs in assembler as well as high-level languages
- Hierarchical scene graph structure
- Full-screen post-processing effects
- Transparent objects arranged in order
- Particle systems

<http://www.ogre3d.org/>

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007


Ogre screenshots



Pacific Storm

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007

Ogre screenshots



Pacific Storm

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007


Ogre screenshots



Ankh, Heart of Osiris

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007

Ogre screenshots



Stencil shadow demonstration

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007


Torque engine

- Distributed by GarageGames
- \$150 indie, \$750 commercial licenses
- Write in a TorqueScript language
- TorqueGameEngine and TorqueGameEngineAdvanced
- TorqueNet network architecture (supports up to 128 players)

<http://www.garagegames.com/>

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007

Torque screenshots



From Torque website

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007

Gamebryo engine

- \$50,000!
- Completely customizable rendering pipeline
- Default rendering path includes dark maps, decals, projective lights & shadows, bump maps, environment maps
- Render-to-texture capabilities for TV screens, shadows, etc.
- HDR capabilities
- Optimized for XBOX 360, Playstation3, etc.

<http://www.emergent.net/>

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 – February 5, 2007

Gamebryo screenshots



Sid Meier's Civilization IV - Firaxis Games (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Gamebryo screenshots



Freedom Force vs the Third Reich - Irrational Games (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007


Unreal 3

- Over \$700,000!!
- 64-bit HDR rendering pipeline
- Support for all modern lighting techniques, including spherical harmonic maps
- Dynamic stencil-buffered shadows
- Soft shadows using filtered shadow buffers
- Seamless interconnected interior/exterior

<http://www.unrealtechnology.com>

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Unreal3 screenshots (?)



Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007


Unreal3 screenshots (?)



Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Unreal3 screenshots



Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Unreal3 screenshots



Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Unreal3 screenshots



Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Field Trip!

- Unreal2 rendering engine

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Improvement over time

Unreal Engine 1 2 3

Unreal3 Engine - Unreal (2006)

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Real-time APIs

- Rendering engine communicates to the hardware via a real-time API
- There are currently two popular real-time APIs:
 - OpenGL
 - DirectX

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Real-time APIs

```

    graph TD
      A[scene graph] --> B[OpenGL]
      A --> C[DirectX]
      B --> D[graphics hardware]
      C --> D
      D --> E[display]
    
```

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

OpenGL

- Design led by Kurt Akeley, one of the founders of Silicon Graphics
- Expose the graphics hardware as a state machine
- Specification controlled by the "ARB": Architecture Review Board
<http://www.opengl.org>

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

OpenGL state machine

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Issuing geometry

- There are two ways to pass geometry down to the hardware in OpenGL:
 - Immediate mode
 - Display lists

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Immediate mode

- Vertices are passed down as soon as they are specified

```

for each vertex i in geometry{
  // specify the vertex normal
  glVertexNormal3f (n[i][0],n[i][1],n[i][2]);
  // specify the vertex color
  glColor4f (c[i][0],c[i][1],c[i][2],c[i][3]);
  // specify the texture coordinates
  glTexCoord2f (tex[i][0], tex[i][1]);
  // specify the vertex position
  glVertex3f (v[i][0], v[i][1], v[i][2]);
}
    
```

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Immediate mode

- Note that there are various commands to allow different format types:


```
void glVertex{234}{sifd}{v}(TYPE coords);
```
- Examples:


```

glVertex2d(3,4);
glVertex2f(3.1,4.2);

float v[3] = {1.0, 2.9, 3.5};
glVertex3f(v);
            
```

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Immediate mode

- The glBegin() statement indicates what kind of primitive is being drawn


```

glBegin(GL_TRIANGLES);
  glVertex2d(0,0);
  glVertex2d(2,3);
  glVertex2d(-1,5);
glEnd();
            
```
- Some others include GL_POINTS, GL_LINES, GL_QUADS, GL_TRIANGLE_STRIP, etc.

AGL Real-time Rendering & Graphics Hardware Pradeep Sen Class 6 - February 5, 2007

Vertex arrays

- Instead of performing individual calls per vertex, pack them into an array and call it once
- First we must enable the “vertex arrays”

```
void glEnableClientState(GLenum array);
```
- Where array can be:
 - GL_VERTEX_ARRAY
 - GL_COLOR_ARRAY
 - GL_TEXTURE_COORD_ARRAY
 - etc



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Vertex arrays

- Provide a pointer to the array that contains the appropriate information

```
void glVertexPointer(int size, GLenum type,  
GLsizei stride,  
const GLvoid *ptr);
```

- Similar calls exist for color, texture coordinates, etc.



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Vertex arrays

- To draw, use

```
void glArrayElement(int i);
```

- To send down the i^{th} element, or

```
void glDrawArrays(GLenum mode, int first,  
GLsizei count);
```

- To draw a group of elements

```
int i;  
glBegin(mode);  
for (i=0; i < count; i++)  
    glArrayElement(first+i);  
glEnd();
```



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Display lists

- Provides a cache of OpenGL commands

```
glNewList(1, GL_COMPILE);  
drawScene();  
glEndList();
```

```
glLoadMatrix(M);  
glCallList(1);
```

- Cannot be modified!



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

OpenGL Extensions

- OpenGL was meant to be dynamic and to evolve as hardware capabilities changed
- Designers developed an extension-based architecture, where the driver reports to OpenGL what extensions it exposes
- Complete list of extensions can be seen in the extension registry:

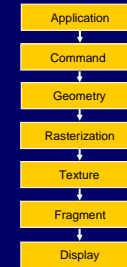
<http://www.opengl.org/registry/>



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

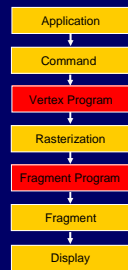
GPU Programmability



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

GPU Programmability



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Vertex program

- Extension GL_ARB_vertex_program



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Sample program



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 6 – February 5, 2007

Reading

- ARB_vertex_program and ARB_fragment_program specifications (become familiar with the instruction sets)