

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

Pradeep Sen
Advanced Graphics Lab

Class 5
January 31, 2007

Pop-quiz!

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Last time

- Application-side optimizations
- Bounding box hierarchy
- BSP Trees
- Started talking about cells and portals

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

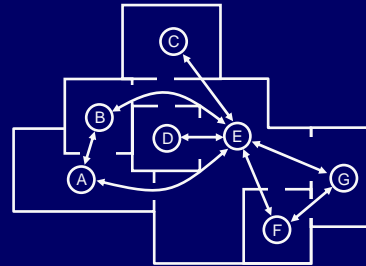
Today

- More on portals
- More occlusion testing
- Scene graphs
- Laying down the foundation of a rendering engine

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Cells and portals example



AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Room drawing algorithm

```
void drawRoom(Room *room, Clip planes) {  
    if (room->visited)  
        return; // this room has been drawn  
    room->visited = true;  
    for each portal in room {  
        if isVisible(portal) {  
            new_planes = clipPortal(planes,  
                                   portal->clipPlanes);  
            drawRoom(portal->room, new_planes);  
        }  
    }  
    draw(room->walls, planes);  
    draw(room->objects, planes);  
    room->visited = false;  
}
```

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Portal culling



Quake4 - id (2005)

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Portal culling

- In other words, it is a form of occlusion culling
- Objects are hidden by the frame of the portal and walls of the room
- If you want extreme efficiency, then a good portion of your code will be dedicated to efficient clipping

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Portal engines

- Are not suitable for outdoor scenes...



Command & Conquer: Generals - Electronic Arts (2003)

- Where do you put the portal?

AGL Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Hierarchical z-buffer

- With a normal z-buffer, if you are drawing a screen-sized quad, you will rasterize and compare depth at every pixel
- For a 1024 x 1024 image, that is a million comparisons per quad
- What if we had a hierarchical representation that would allow us to do the comparisons more efficiently?

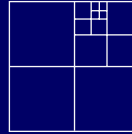


Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Hierarchical z-buffer

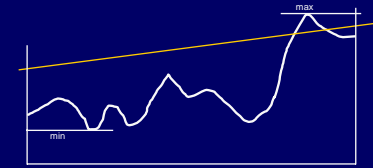
- Quad tree structure
- Divides the screen recursively into four children
- At each node store zmin, zmax



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

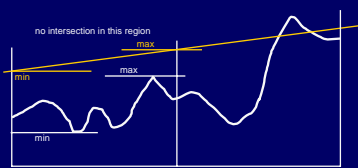
2-D example



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

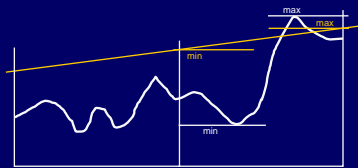
2-D example



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

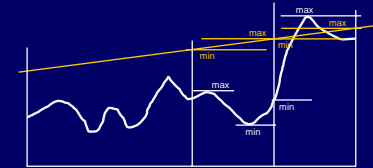
2-D example



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

2-D example



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Hierarchical z-buffer

- Let's see the size of the structure
- Depth buffer resolution: NVIDIA & ATI use 3 bytes (24 bits/pixel)
- 1024 x 1024 image is 3MB for standard (flat) depth buffer
- Calculation for quad-tree depth buffer size
- Hierarchical z-buffer: 2MB of additional data, not too much worse than the original size...

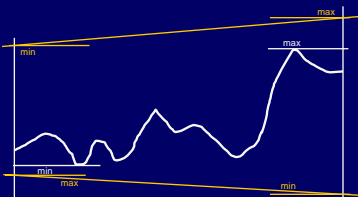


Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Hierarchical z-buffer

- Best time lookup:



Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Hierarchical z-buffer

- ATI's Hyper-Z
- One-level hierarchical z-buffer



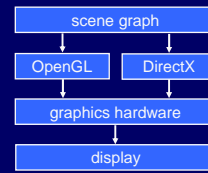
Real-time Rendering & Graphics Hardware
Pradeep Sen

Class 5 - January 31, 2007

Occlusion queries

- Want a hardware-based test that tells you: would this triangle show up if I rendered it?
- `GL_HP_occlusion_test`
- Blocking call, bad for performance
- No feedback on how many pixels are visible
- `nv_occlusion_query`
- Non-blocking
- Returns the number of pixels written

Scene graphs



Scene Graph

- Tree structure that keeps track of objects in the scene
- Because rendering engines allow instancing, these are technically DAGs
- Transformations are inherited

Scene Graphs

