Hierarchical Occlusion Map

Zhang et al
SIGGRAPH 98

Basic Ideas
- Choose a set of graphics objects from the scene as Occluder
- Use the occluder to define Occlusion Map (hierarchically)
- Compare the rest of scene against the occlusion map

Example
Blue: Occluders
Red: Occludees

2-Step Occlusion Test
1. Overlap Test
2. Overlap Test
Overlap + Depth = Occlusion

Why decomposition?
- The occlusion test is done approximately (conservatively)
- We can afford to be more conservative in depth test than overlap test
Why Decomposition?

Overlap Test – Occlusion Map
- Representation of projection for overlap test: occlusion map
- A gray scale image - each pixel represents one block of screen region
- Generate by rendering occluders

Occlusion Map (OM)
- Each pixel of the occlusion map has an opacity, which represents the ratio of the sum of the opaque areas in the block to the total area.
- If fully covered, $p = 1$, if anti-aliased pixel, $p < 1$)
- Occlusion map: the alpha channel of an image

Overlap Test using OM
- For each potential occludee, we can scan-convert it and compare against the opacity of the pixels it overlaps — Expensive!!
  - Conservative Approximation: use the screen-space bounding box of the occludee (a superset of the actual covered pixels)
  - If all the pixels inside the bounding box are opaque, the object is occluded.

Hierarchical Occlusion Map
- Like hierarchical Z-buffer, we can create a hierarchy to speed up the comparison (for large objects)

Overlap Test using HOM
- Basic Algorithm
  1. Start from the lowest resolution
  2. If the pixel cover the bounding rectangle has a value 1, the object is occluded
  3. Otherwise traverse down the hierarchy:
     - If all children =1: occluded
     - If all children =0: not occluded
     - Otherwise, traverse down further
**Approximate Overlap Test**
- Instead of concluding an object is occluded only when the bounding box is within pixels with opacity 1, we can use a threshold between \([0,1]\).
- Early termination in the high level of the hierarchy.
- What does it mean when a block has high opacity but not one?

*This is the unique feature of HOM!!*

**Depth Test (1)**

Approximate Z (depth) test:
- A single Z Plane

**Depth Test (2)**

Break the screen into small regions
- Build at each frame
- Instead of using Z-buffer, use the occluder’s bounding volume’s farthest Z
- Compare each potential occludee’s nearest Z (conservative test)

**Occluder Selection**

Ideal occluder: the visible objects - it’s a joke
View-dependent occluder: too expensive \((m^6 \log m)\) - Wow!

Solution: Estimate and build an occluder database
Discard objects that do not serve as good occluders

**Occluder Selection (2)**

- Size: not too small
- Redundant: detail polygons (clock on the wall)
- Complexity: Complex polygons are not preferred (why?)
- Done at run time - sort the occluders in depth, add them in order until reach the polygon count.

**Discussion**

- Robust
- Very little assumption about hardware
- Need high depth complexity
- Adaptive occlusion culling
Hardware Support

- Most of the aforementioned algorithms need to be done by applications (software)
- Hardware support is emerging –
  - Example HP Visualize fx board (1997)

HP hardware occlusion

- Extend the OpenGL – add a OCCLUSION_MODE
- The bounding box of an object is scan converted first
- A flag is set if any pixel of the BB faces is visible
- Only need to read back one flag, instead of the entire frame buffer
- Tradeoff - valuable rendering time is used to render useless BB faces (need to be used wisely)
- Reportedly 25%-100% speedup were observed