Large-Scale Polygon Rendering

Solutions
- Decimation
- Visibility Culling
- Parallel Rendering
- Others

Surface Decimation
Use a smaller number of triangles to represent the same surface -> preserve the topology
- Removing vertices
- Contracting edges
- Etc.

Visibility Culling
Do not display polygons that are not visible to the eyes

Parallel Rendering

3D Rendering Pipeline
Parallel Rendering

Geometry processing: parallelized by assigning each processor a subset of primitives (polygons)

Rasterization: parallelized by assigning each processor a portion of pixel calculations

What is rendering anyway?
Calculate the effect of each primitive on each pixel

During transformation, a primitive can go anywhere on the screen

Rendering can be seen as sorting primitives to the screen

Rendering as a sorting process

Where to sort?
Sorting can pretty much happen anywhere

Where/When to sort will affect the structure of the parallel rendering system

Sort-First Parallel Rendering

Graphics database (arbitrarily partitioned)
Redistribute “row” primitives
Geometry processing
Rasterization
Display
**Sort-First Parallel Rendering**

Distribute the primitives early in the rendering pipeline

1. Subdivide the screen
2. “Pre-transform” the primitives
3. Distribute the primitives
4. Each processor render its own primitives
5. No communication needed afterwards

![Sort-First Parallel Rendering Diagram](image)

**Sort-Middle Parallel Rendering**

Distribute the primitives in the middle of pipeline

1. Arbitrary assignment
2. Geometry processing
3. Sorting
4. Rasterization

![Sort-Middle Parallel Rendering Diagram](image)

**Sort-Last Parallel Rendering**

Distribute the primitives in the end of pipeline

1. Arbitrary assignment
2. Geometry processing
3. Rasterization

Sort images with z
Compositing/z-buffer

![Sort-Last Parallel Rendering Diagram](image)