Visible Surface Determination
Part II

Foley & Van Dam, Chapter 15
Visible Surface Determination

- Depth Sort (Painter Algorithm)
- Binary Space Partitioning Tree
- Area Subdivision Algorithms (Warnock’s)
Depth Sort (Painter Algorithm)

- **Algorithm**:
  - Sort the polygons in the scene by their depth
  - Draw them back to front

- **Problem**: Unless all polygons have constant z, a strict depth ordering may not exist

**Note**: Constant z case is important in VLSI design
Depth Sort (Painter Algorithm)

• **General Case:** Given two polygons P and Q, an order may be determined between them, if at least one of the following holds:
  1) z values of P and Q do not overlap
  2) The bounding rectangle in the x, y plane for P and Q do not overlap
  3) P is totally on one side of Q’s plane
  4) Q is totally on one side of P’s plane
  5) The bounding rectangles of Q and P do not intersect in the projection plane
Depth Sort (Painter Algorithm)

- If all the above conditions do not hold, P and Q may be split along intersection edge into two smaller polygons.
Binary Space Partitioning Tree

- Interior nodes correspond to partitioning planes
- Leaf nodes correspond to convex regions of space
Binary Space Partitioning Tree

- Tests 3 and 4 in Depth Sort technique can be exploited efficiently with BSP-Trees:
  - Let $L_p$ be the plane $P$ lies in. The 3D space may be divided into the following three groups:
    - Polygons in front of $L_p$
    - Polygons behind $L_p$
    - Polygons intersecting $L_p$
  - Polygons in the third class are split, and resulting polygons are classified into the first two classes
  - As a result of the subdivision with respect to $L_p$:
    - The polygons behind $L_p$ cannot obscure $P$, so we can draw them first
    - $P$ cannot obscure the polygons in front of $L_p$ so we can draw $P$ second
    - Finally we draw the polygons in front of $P$
Binary Space Partitioning Tree

• **BSP Tree Algorithm:**
  • Construction of the BSP tree:
    – Pick a polygon, let its supporting plane be the root of the tree
    – Create two lists of polygons: those in front, and those behind (splitting polygons if necessary)
    – Recurse on the two lists to create two sub-trees
  • Display:
    – Traverse the BSP tree back to front, drawing polygons in the order they are encountered in the traversal
Binary Space Partitioning Tree

• Example:
Binary Space Partitioning Tree

• **Properties:**
  • The BSP tree is **view independent**
  • The BSP tree is constructed using the geometry of the object only
  • The tree can be used for hidden surface removal at an arbitrary direction
  • BSP tree is an **object-precision** algorithm
Area Subdivision Algorithms

- **Warnock’s Algorithm:**
  - Subdivide screen area recursively, until visible surfaces are easy to determine
  - Each polygon has one of four relationships to the area of interest:
    - Surrounding
    - Intersecting
    - Contained
    - Disjoint
Area Subdivision Algorithms

- **Warnock’s Algorithm:**
  - If all polygons are disjoint from the area, fill area with background color
  - Only one intersecting or contained polygon: First fill with background color, then scan convert polygon
  - Only one surrounding polygon: Fill area with polygon’s color
  - More than one polygon is surrounding, intersecting, or contained, but one surrounding polygon is in front of the rest: Fill area with polygon’s color
  - If none of the above cases occurs: Subdivide area into four parts, and recurse
Area Subdivision Algorithms

• Warnock’s Algorithm:
  • When the resolution of the image is reached, polygons are sorted by their Z-values at the center of the pixel, and the color of the closest polygon is used

• Image-precision technique