BSP Trees

• Binary Spatial Partitioning (BSP) means:
  – Partition (or split) a space into Binary (two) parts using a separating plane.
  – Repeat the process for both resulting subspaces and you will get a BSP tree.
Occlusion

Objects “behind” the splitting plane cannot hide objects “in front” of the plane, regardless the relative location of the observer.
Splitting Planes

- What we see in this example is a simple model with four polygons.
- We choose the splitting planes so that they lay on a one of the polygon of the model.
Creating the tree

We choose a splitting plane that splits the space in two.

- **Associated Splitting Plane**
- **Root Node**
- **L Node**
- **R Node**

*Chosen Splitting Plane*
Creating the tree
Creating the tree

Chosen Splitting Plane
Creating the tree

Chosen Splitting Plane
Creating the tree

The leaves of the tree are convex regions.
We want to render the scene from this point of view. In what order should we render the regions?
Traversing the Tree

Test against the splitting plane

Traverse the **R** subtree **before** the **L** subtree

Rendering order:
Creating the tree

Test against the splitting plane

Traverse the L subtree before the R subtree

Rendering order:  

Test against the splitting plane
Creating the tree

Traverse the L subtree before the R subtree

Rendering order:  

Test against the splitting plane
Creating the tree

Test against the splitting plane

R subtree before L subtree

Rendering order:  ■  ■
Creating the tree

Test against the splitting plane

Traverse the L subtree before the R subtree

Test against the splitting plane

Rendering order:  ▢ ▢ ▢ ▢
Convex Cells

The cells can be ordered back to front, or front to back.
F2B Order
Hidden Surface Removal

- Construct a BSP tree:
  - Pick a polygon, let its supporting plane be the root of the tree.
  - Create two lists of polygons: these in front, and those behind (splitting polygons as necessary).
  - Recurse on the two lists to create the two sub-trees.

Display:
- Traverse the BSP tree back to front, drawing polygons in the order they are encountered in the traversal.
BSP Construction
BSP Trees

BSP-Trees are view-independent

A good splitting plane minimize the number of polygon intersections, and aims at a balanced tree.

How to choose the order of splitting planes during construction?
Point Location

Given p, in which cell it resides?
Given R, which cells, and in which order it traverses?