BSP-Trees

- Binary Space Partition Trees
  - Split space along planes
  - Allows fast queries of some spatial relations

- Simple construction algorithm
  - Select a plane as sub-tree root
  - Everything on one side to one child
  - Everything on the other side to other child
  - Use random polygon for splitting plane
BSP-Trees

BSP-Trees
BSP-Trees

- Visibility Traversal
  - Variation of in-order-traversal
    - Child one
    - Sub-tree root
    - Child two
  - Select “child one” based on location of viewpoint
    - Child one on same side of sub-tree root as viewpoint

Let’s consider an example…
BSP-Trees

Save nodes in order when popped
BSP-Trees

Remember to 'pop' before traversing children

BSP-Trees
BSP-Trees

21

BSP-Trees

22
This is a forward-to-backward traversal.
BSP-Trees

On to another useful space partition…
RayTrace(image)
  For ray in camera
    image[pixel] = Trace(ray)
Trace(ray)
  t_hit = infinity
  For object in scene
    t_hit = min(object.intersect(ray), t_hit)
    shade at t_hit
  possible calls to Trace(new_ray)
Your Ray Tracer

RayTrace(image)
  For ray in camera
    image[pixel] = Trace(ray)

Trace(ray)
  t_hit = infinity
  For object in scene
    t_hit = min(object.intersect(ray), t_hit)
  shade at t_hit
  possible calls to Trace(new_ray)

That's a lot of monkeys to test
Bounding Shapes

• Bounding shape completely encloses associated object
• Rays cannot hit object w/o intersecting bounding shape
• Two objects cannot collide if shapes don’t overlap
• Tradeoff: Simplicity -vs- tightness

Axis-Aligned Bounding Boxes

• Axis-aligned bounding box defined by min and max xyz
Axis-Aligned Bounding Boxes

Transform box
Not axis-aligned

Min/max of new points
Linear cost to compute

Why would we do this?

Axis-Aligned Bounding Boxes

Min/max of transformed BB points
Constant time
Adds slop
Cumulative slop if multiple transforms occur sequentially
### Tightness

<table>
<thead>
<tr>
<th>Method (1)</th>
<th>Method (2)</th>
</tr>
</thead>
</table>

### Box Intersection

- Entry y at entry y
- Exit x at exit x
- Exit y at exit y

Line goes through window if intervals overlap

\[
\begin{align*}
    t_{x1} & = \min (t_{x_{\text{exit}}}, t_{x_{\text{entry}}}) \\
    t_{x2} & = \max (t_{x_{\text{exit}}}, t_{x_{\text{entry}}}) \\
    t_{y1} & = \min (t_{y_{\text{exit}}}, t_{y_{\text{entry}}}) \\
    t_{y2} & = \max (t_{y_{\text{exit}}}, t_{y_{\text{entry}}})
\end{align*}
\]
AABB Trees

Ray Test Against Bound Tree

- `RayHitSubTree(&ray, node)`
  - If `RayHitBB(ray, node.xfBB)`
  - `ixfRay = Inverse(node.xf)*ray`
  - If `RayHitBB(ixfRay, node.BB)`
    - If node is group
      - Foreach child in node.children
        - `RayHitSubTree(ixfRay, child)`
    - else // node not group
      - `RayHitGeometry(ixfRay, node.geom)`
  - `ray.collisionInfo.update(ixfRay)`
Building the tree

• Sort (or QuickSelect) and split on one axis
• Repeat for the other axis
  - x,y,z

Other Schemes

• Uniform Grid/Octrees
• Spatial Hierarchies
• Etc
Voxels/Octree

- Fun Links
  - http://www.youtube.com/watch?v=6x8fYNFZsDQ
  - http://www.youtube.com/watch?v=Jl5dWZtmQOE