Hierarchical Bounding Volume

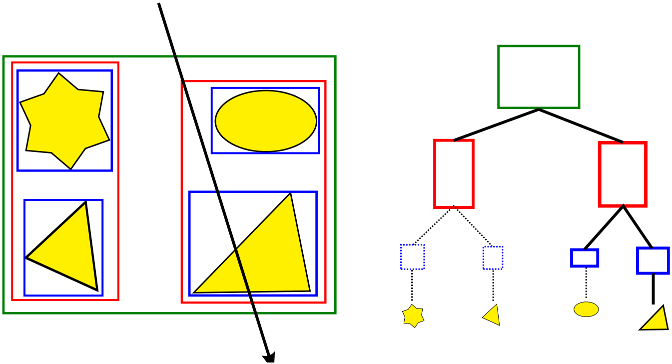
October 11, 2005

Outline

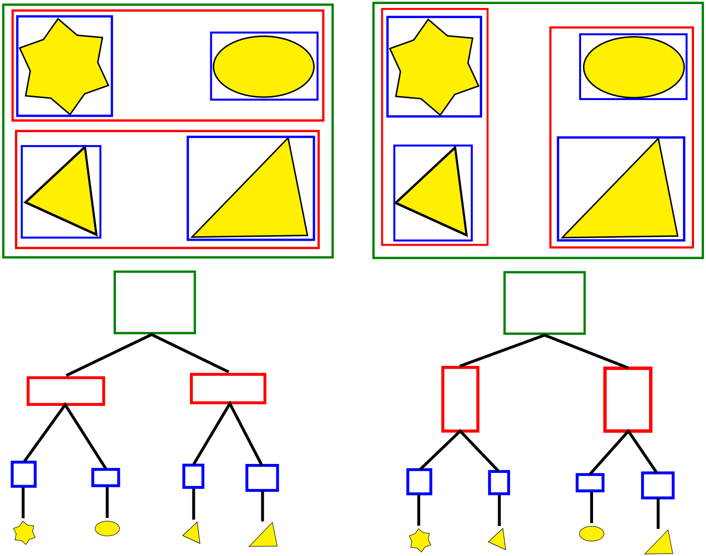
- Introduction to hierarchical bounding volume (HBV)
- Tree generation
- Other optimization issues

Introduction to HBV

- A HBV is a tree structure formed by bounding volumes.
- In intersection testing, child nodes are tested only if their parents intersect with the ray.



More than one way to build the tree...



What is a "good tree"? (1/2)

- We build the tree in order to reduce intersection tests.
- Thus, the quality of a tree can be represented by $E[X]$, where X denotes the number of intersection tests for each ray.

What is a "good tree"? (2/2)

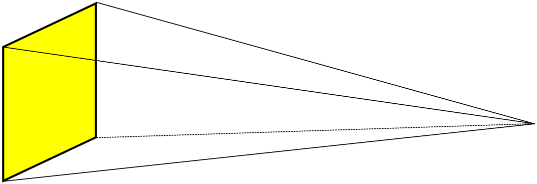
$E[X]$ can be expressed by:

$$E[X] = \sum_n a_n P(\text{ray hits } n \mid \text{ray hits the root})$$

where n denotes nodes in the tree, and a_n denotes the additional intersection tests when the ray hits n.
Actually, a_n is the number of child of n.

Approximation of the Conditional Probability

- Assuming uniform distribution for ray directions, the probability that the ray hits a bounding box is proportional to the solid angle spanned by it.



- At large distances, this is approximately proportional to the surface area of the bounding box.
- Therefore, the conditional probability can be expressed by:

$$P(\text{ray hits } n \mid \text{ray hits the root}) = \frac{\text{surface area of } n}{\text{surface area of the root}}$$

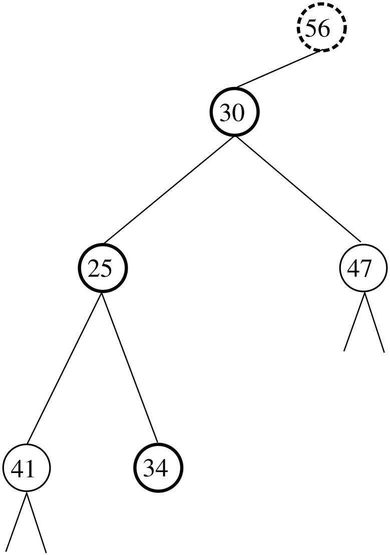
Tree Generation

- Finding the optimal tree may take too much computation.
- However, finding an suboptimal one is acceptable.
- Construct the suboptimal tree by heuristic search:
  1. Initialize the tree as empty.
  2. For each primitive, find a insertion point for it that the increased cost ($E[X]$) is minimalized.
  3. Repeat until all primitives are inserted.

Finding the Best Insertion Point

- We don't need to search the whole tree.
- For each internal node, we can prune its children which have higher insertion cost than others.
- Therefore, we only search nodes along a path from root to some leaf node.
- The time complexity is approximately $O(n \log n)$

Example

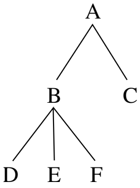


Other Optimization Issues

- Tree traversal optimization
- Intersection ray V.S. shadow ray
- Object cache

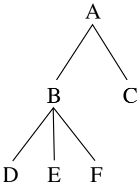
Tree Traversal Optimization

- In HBV algorithm, the tree is searched in a fixed order (usually DFS).

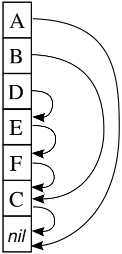


Tree Traversal Optimization

- In HBV algorithm, the tree is searched in a fixed order (usually DFS).



- We can pack the tree into an array with skip pointers, which reduce traversal time.



Intersection Ray V.S Shadow Ray

- We don't care about the intersection point of shadow rays.
- Different acceleration structures can be used for intersection rays and shadow rays separately.
- In general, grids are faster for intersection rays, and HBV are faster for shadow rays.

Object Cache

- Rays often have spatial coherency, and we can cache the object hit by the previous ray.
- For intersection rays, the ray will still need to be check against the environment.
  - Mailboxes are used to prevent duplicated checks.
- For shadow rays, we need to cache different objects for each light source.
- Reflected and transmitted rays are ignored in this scheme, because they are usually spatially different.

Thank you!