Spatial Data Structures

Hierarchical Bounding Volumes
Regular Grids
Octrees
BSP Trees
Constructive Solid Geometry (CSG)

November 6, 2003
Doug James
Carnegie Mellon University

http://www.cs.cmu.edu/~d james/Fall03/15-462
Announcements

• Ray tracing handouts online (CMU access only)
  – Supplements text
• Last written assignment out tomorrow
  – Due Tuesday, Nov 18.
• Last programming assignment
  – Ray tracing
  – Out Tuesday, Nov 18.
  – Due Tuesday, Dec 2.
Ray Tracing Acceleration

• Faster intersections
  – Faster ray-object intersections
    • Object bounding volume
    • Efficient intersectors
  – Fewer ray-object intersections
    • Hierarchical bounding volumes (boxes, spheres)
    • Spatial data structures
    • Directional techniques

• Fewer rays
  – Adaptive tree-depth control
  – Stochastic sampling

• Generalized rays (beams, cones)
Spatial Data Structures

• Data structures to store geometric information
• Sample applications
  – Collision detection
  – Location queries
  – Chemical simulations
  – Rendering
• Spatial data structures for ray tracing
  – Object-centric data structures (bounding volumes)
  – Space subdivision (grids, octrees, BSP trees)
  – Speed-up of 10x, 100x, or more
Bounding Volumes

• Wrap complex objects in simple ones
• Does ray intersect bounding box?
  – No: does not intersect enclosed objects
  – Yes: calculate intersection with enclosed objects
• Common types
  – Boxes, axis-aligned
  – Boxes, oriented
  – Spheres
  – Finite intersections or unions of above
Selection of Bounding Volumes

• Effectiveness depends on:
  – Probability that ray hits bounding volume, but not enclosed objects (tight fit is better)
  – Expense to calculate intersections with bounding volume and enclosed objects

• Amortize calculation of bounding volumes

• Use heuristics
Hierarchical Bounding Volumes

• With simple bounding volumes, ray casting still has requires $O(n)$ intersection tests
• Idea: use tree data structure
  – Larger bounding volumes contain smaller ones etc.
  – Sometimes naturally available (e.g. human figure)
  – Sometimes difficult to compute
• Often reduces complexity to $O(\log(n))$
Ray Intersection Algorithm

- Recursively descend tree
- If ray misses bounding volume, no intersection
- If ray intersects bounding volume, recurse with enclosed volumes and objects
- Maintain near and far bounds to prune further
- Overall effectiveness depends on model and constructed hierarchy
Spatial Subdivision

- Bounding volumes enclose objects, recursively
- Alternatively, divide space
- For each segment of space keep list of intersecting surfaces or objects
- Basic techniques
  - Regular grids
  - Octrees (axis-aligned, non-uniform partition)
  - BSP trees (recursive Binary Space Partition, planes)
Grids

- 3D array of cells (voxels) that tile space
- Each cell points to all intersecting surfaces
- Intersection alg steps from cell to cell
Caching Intersection points

- Objects can span multiple cells
- For A need to test intersection only once
- For B need to cache intersection and check next cell for closer one
- If not, C could be missed

(yellow ray)
Assessment of Grids

- Poor choice when world is non-homogeneous
- Size of grid
  - Too small: too many surfaces per cell
  - Too large: too many empty cells to traverse
  - Can use alg like Bresenham’s for efficient traversal
- Non-uniform spatial subdivision more flexible
  - Can adjust to objects that are present
Outline

- Hierarchical Bounding Volumes
- Regular Grids
- Octrees
- BSP Trees
- Constructive Solid Geometry (CSG)
Quadtrees

- Generalization of binary trees in 2D
  - Node (cell) is a square
  - Recursively split into 4 equal sub-squares
  - Stop subdivision based on number of objects
- Ray intersection has to traverse quadtree
- More difficult to step to next cell
Octrees

- Generalization of quadtree in 3D
- Each cell may be split into 8 equal sub-cells
- Internal nodes store pointers to children
- Leaf nodes store list of surfaces
- Adapts well to inhomogeneous scenes
Assessment for Ray Tracing

- **Grids**
  - Easy to implement
  - Require a lot of memory
  - Poor results for inhomogeneous scenes
- **Octrees**
  - Better on most scenes (more adaptive)
- **Alternative:** nested grids
- **Spatial subdivision** expensive for animations
- **Hierarchical bounding volumes**
  - Natural for hierarchical objects
  - Better for dynamic scenes
Other Spatial Subdivision Techniques

- Relax rules for quadtrees and octrees
- k-dimensional tree (k-d tree)
  - Split at arbitrary interior point
  - Split one dimension at a time
- Binary space partitioning tree (BSP tree)
  - In 2 dimensions, split with any line
  - In k dims. split with k-1 dimensional hyperplane
  - Particularly useful for painter’s algorithm
  - Can also be used for ray tracing [see handout]
Outline

• Hierarchical Bounding Volumes
• Regular Grids
• Octrees
• BSP Trees
• Constructive Solid Geometry (CSG)
BSP Trees

• Split space with any line (2D) or plane (3D)
• Applications
  – Painters algorithm for hidden surface removal
  – Ray casting
• Inherent spatial ordering given viewpoint
  – Left subtree: in front, right subtree: behind
• Problem: finding good space partitions
  – Proper ordering for any viewpoint
  – Balance tree
• For details, see http://reality.sgi.com/bspfaq/
Building a BSP Tree

- Use hidden surface removal as intuition
- Using line 1 or line 2 as root is easy

A BSP tree using 2 as root

The subdivision of space it implies
Splitting of surfaces

• Using line 3 as root requires splitting
Building a Good Tree

- Naive partitioning of $n$ polygons yields $O(n^3)$ polygons (in 3D)
- Algorithms with $O(n^2)$ increase exist
  - Try all, use polygon with fewest splits
  - Do not need to split exactly along polygon planes
- Should balance tree
  - More splits allow easier balancing
  - Rebalancing?
Painter’s Algorithm with BSP Trees

• Building the tree
  – May need to split some polygons
  – Slow, but done only once
• Traverse back-to-front or front-to-back
  – Order is viewer-direction dependent
  – What is front and what is back of each line changes
  – Determine order on the fly
Details of Painter’s Algorithm

• Each face has form $Ax + By + Cz + D$
• Plug in coordinates and determine
  – Positive: front side
  – Zero: on plane
  – Negative: back side
• Back-to-front: inorder traversal, farther child first
• Front-to-back: inorder traversal, near child first
• Do backface culling with same sign test
• Clip against visible portion of space (portals)
Clipping With Spatial Data Structures

• Accelerate clipping
  – Goal: accept or rejects whole sets of objects
  – Can use an spatial data structures

• Scene should be mostly fixed
  – Terrain fly-through
  – Gaming

Hierarchical bounding volumes

Octrees
Data Structure Demos

- BSP Tree construction
  http://symbolcraft.com/graphics/bsp/index.html
- KD Tree construction
  http://www.rolemaker.dk/nonRoleMaker/uni/algogem/kdtree.htm
Real-Time and Interactive Ray Tracing

• Interactive ray tracing via space subdivision
  http://www.cs.utah.edu/~reinhard/egwr/

• Interactive ray tracing with good hardware
  http://www.cs.utah.edu/vissim/projects/raytracing/
Outline

- Hierarchical Bounding Volumes
- Regular Grids
- Octrees
- BSP Trees
- **Constructive Solid Geometry (CSG)**
Constructive Solid Geometry (CSG)

- Generate complex shapes with simple building blocks (boxes, spheres, cylinders, cones, ...)
- Particularly applicable for machined objects
- Efficient with ray tracing
Example: A CSG Train

Brian Wyvill et al., U. of Calgary
Boolean Operations

- Intersection and union
- Subtraction
  - Example: drilling a hole
CSG Trees

• Set operations yield tree-based representation

• Use these trees for ray/objects intersections
• Think about how!
Implicit Functions for Booleans

- Solid as implicit function, $F(x, y, z)$
  - $F(x, y, z) < 0$ interior
  - $F(x, y, z) = 0$ surface
  - $F(x, y, z) > 0$ exterior
- For CSG, use $F(x, y, z) \in \{-1, 0, 1\}$
- $F_A \circlearrowleft B (p) = \max(F_A(p), F_B(p))$
- $F_{A \cup B} (p) = \min(F_A(p), F_B(p))$
- $F_{A - B} (p) = \max(F_A(p), -F_B(p))$
Summary

• Hierarchical Bounding Volumes
• Regular Grids
• Octrees
• BSP Trees
• Constructive Solid Geometry (CSG)
Preview

• Radiosity
• Ray tracing handouts online (CMU access only)
  – Supplements text
• Last written assignment out tomorrow
  – Due Tuesday, Nov 18.