Collision Detection, Part II

Kim Steenstrup Pedersen

kimstp@itu.dk

www.itu.dk/courses/MEP/E2006/

The IT University of Copenhagen

Plan for today

- OBB's
- Broad phase techniques
- Bounding volume (BV) hierarchies
- Program design II.

Bounding Volumes: Oriented Bounding Box (OBB)

Representation:

- OBB center point c.
- Local unit axis vectors $\mathbf{A}^i$, $i = 1, 2, 3$.
- Radius vector $\mathbf{a} = (a_1, a_2, a_3)^T$.

Intersection test:

- Naive test: Total tests required 12 edges $\times$ 6 faces $\times$ 2 cubes = 144 tests!
- We can do better by using method due to Gottschalk et al. [1996].

Separating Axis Theorem:
Two disjoint boxes can always be separated by a plane which is parallel to a face of either boxes, or parallel to an edge from each box. The separating plane normal forms a separating axis. Projecting the boxes onto this axis produces disjoint intervals.

Bounding Volumes: Oriented Bounding Box (OBB)

- Using the separating axis theorem we can reduce the total number of tests to: 3 faces + 3 faces + 3 $\times$ 3 edges = 15
- If any one of the 15 tests leads to disjoint intervals then the two OBB's are disjoint and no more tests need to be done. If all tests indicate intersection then the two OBB's are colliding.

Center displacement: $\mathbf{T} = c_B - c_A$
Unit separating axis: $\mathbf{L}$
Projected radius: $r_A = \sum_{i=1}^{3} |a_i \mathbf{A}^i \cdot \mathbf{L}|$

Intersection test: Intervals are disjoint iff

$$|\mathbf{T} \cdot \mathbf{L}| > \sum_{i=1}^{3} |a_i \mathbf{A}^i \cdot \mathbf{L}| + \sum_{i=1}^{3} |b_i \mathbf{B}^i \cdot \mathbf{L}|$$

- Various simplifications of the test can be made for the individual separating axes $\mathbf{L}$.

Bounding Volumes: Oriented Bounding Box (OBB)

Construction:

- By design. The OBB may be specified by hand as part of the design of the model/object.
- Automatic method by Gottschalk et al. [1996] based on a principal component analysis (PCA) of the model geometry.

Gottschalk et al. OBB:
Assume object consists of a triangle soup (unordered set of triangles). Let the ith triangle be represented by vertices $\mathbf{p}^i, \mathbf{q}^i, \mathbf{r}^i$, with $i = 1, \ldots, n$.

Mean of triangle centers

$$\mu = \frac{1}{3n} \sum_{i=0}^{n} (\mathbf{p}^i + \mathbf{q}^i + \mathbf{r}^i)$$

Covariance matrix $C \in \mathbb{R}^{3\times 3}$ of triangle centers

$$C_{jk} = \frac{1}{3n} \sum_{i=0}^{n} (\bar{\mathbf{p}}_j^i \bar{\mathbf{p}}_k^i + \bar{\mathbf{q}}_j^i \bar{\mathbf{q}}_k^i + \bar{\mathbf{r}}_j^i \bar{\mathbf{r}}_k^i) \ , \ \bar{\mathbf{p}}^i = \mathbf{p}^i - \mu, \text{ etc.}$$

Bounding Volumes: Oriented Bounding Box (OBB)

Constructing the OBB from C and μ:

- Form the OBB axis from the 3 normalized orthogonal (due to symmetry) eigenvectors of the covariance matrix C.
- The center of the OBB is given by μ.
- Find extremal vertices along each axis to find the size of the OBB.

Aside:
The vector $\mathbf{x}$ is an eigenvector of the matrix A if

$$A\mathbf{x} = \lambda\mathbf{x}$$

and λ is called the eigenvalue for $\mathbf{x}$. For a covariance matrix λ is the variance along the axis $\mathbf{x}$.

Efficient numerical methods exist for computing eigenvectors.

Bounding Volumes: Oriented Bounding Box (OBB)

Issues with the method by Gottschalk et al.:

Problem:

- Dense planar cluster of interior vertices may misalign the OBB.

Solution:

- Form the convex hull of the model and use triangles on the hull to align the OBB.

Problem:

- Dense planar cluster on the convex hull boundary may misalign the OBB.

Solution:

- Weigh the covariances and mean by the area of the triangles. Modifies the equations from last slide slightly.

Computational complexity:
$O(n)$ without convex hull and $O(n \log n)$ including convex hull computations. n is the number of triangles.

Other narrow phase techniques

- Scene geometry vs scene geometry (Exact collision detection):
  Example: Two convex polyhedra X and Y:
  1. Test: Vertices of X within Y and vice versa
  2. Test: Edges of X penetrating Y and vice versa
  3. Test: Face centroid of X within Y (Identical and aligned objects)
- Rays vs bounding volumes or scene geometry: Useful for bullets, line-of-sight determination, object-ground collision.
- BV or scene geometry vs planes.

Broad phase strategies

Bounding volumes does not reduce the number of pairwise tests!

Possible broad phase strategies:

- Bounding Volume Hierarchies (BVH)
- Spatial partitioning:
  - Grid-based methods (e.g. quad- and octrees)
  - BSP trees
- Temporal coherence (swept volume): Check 4D space-time objects, estimate next time for collision. Test only objects that move!

A two phase (broad and narrow phase) strategy would use e.g. the BVH to do a broad phase detection at internal nodes. At leaf nodes we could do a narrow phase BV vs. BV detection and continue with an exact detection based on the scene geometry.

Bounding Volume Hierarchies (BVH)

- A hierarchical data structure which represent a collection of BV's and their hierarchical relationships.
- Useful when doing collision detection between complex arbitrarily shaped objects.
- Usual data structures are trees.
- Works well for static objects and dynamic objects where the object sub-parts stay together.

Desired characteristics:

- The nodes contained in any given subtree should be near each other.
- Each node should have minimal volume.
- The sum of all bounding volumes should be minimal.
- The volume overlap of siblings should be minimal.
- The hierarchy should be balanced to achieve efficient pruning during traversal.

Bounding Volume Hierarchies (BVH)

Construction:

- Top-down: Form BV incompasing the whole object and start to split into sub BV's.
- Bottom-up: Merge BV's from primitive BV's.
- Insertion: Insert BV's following some optimality criterion.

Collision queries by traversing hierarchies:

- Object A and B represented by each their BVH.
- Collision query: Check object A's BVH against object B's by simultaneously traversing the two BVH's.
- Visiting strategy: breadth first, depth first, etc.

Granularity is chosen by setting the size of the leaf BV's. Does not have to follow scene geometry. We can build BVH's for a complete scene or individual objects.

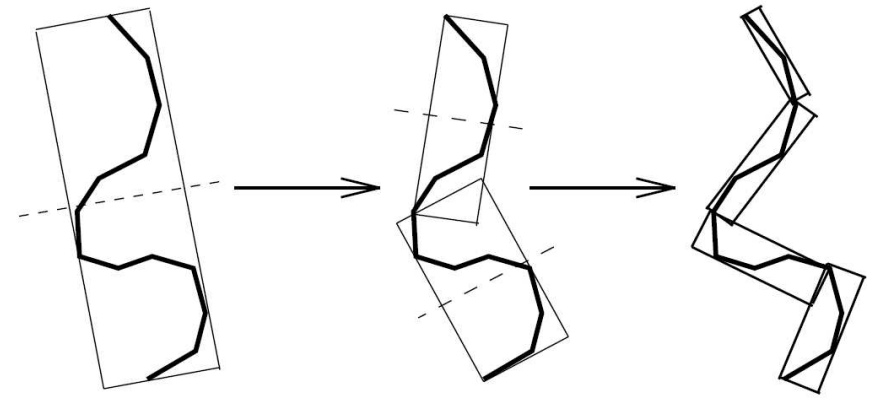
Bounding Volume Hierarchies (BVH): OBB tree

Gottschalk et al. [1996] proposed a top-down method for building an OBB tree.

1. Compute OBB for object.
2. Split object polygon set along longest OBB axis (largest eigenvalue). Splitting plane orthogonal to axis and going through mean value μ.
3. Polygons are sorted after which side their center lie according to splitting plane. Go to step 1 for each side of the splitting plane.
4. Continue subdivision until one polygon per OBB or polygon set cannot be subdivided.

This subdivision can be stopped earlier given some stopping criterion, e.g. stop when less than 10 polygons per OBB.

Computational complexity:
$O(n \log n)$ without convex hull and $O(n \log^2 n)$ including convex hull computations. n is the number of triangles.



Collision detection system program design: Tips for BVH

Scene geometry should be augmented with collision geometry. This could be done by augmenting your scene graph data structure with collision data. Each geometry node could store a bounding volume.

But not always a good idea to make your BVH equal to the scene graph!

This works for sub-graphs representing complex objects (with sub-parts being in close proximity of each other).

Alternative: Maintain two graphs: A scene graph and a BVH tree.

By clever design you can reuse the core scene graph data structure for the BVH tree. Each BVH node need to know about the scene geometry, so a pointer to scene graph nodes is probably needed.

Bounding Volume Hierarchies (BVH): Sphere tree

Simple sphere tree based on scene graph:
Building a sphere tree bottom-up by merging sub-spheres to form a new sphere.

Given a scene graph for an object.

1. Start at the leaves by computing bounding spheres of objects.
2. Move to parent scene graph node and compute bounding sphere that circumscribes the bounding spheres of the children.
3. Add new bounding sphere to sphere tree by making it the new root node.
4. Go to step 2, until scene graph root has been reached.

Merging two spheres: $A(c_A, r_A), B(c_B, r_B), d = |c_A - c_B|$

- Case 1: Sphere A is completely enclosed by sphere B. Use sphere B as new bounding sphere. True if, $|r_A - r_B| \geq d$.
- Case 2: Spheres A and B are partially overlapping or disjoint. New radius $r = (d + r_A + r_B)/2$, new center $c = c_A - r_A\mathbf{T} + r\mathbf{T}$ with $\mathbf{T} = (c_A - c_B)/|c_A - c_B|$.

Other more efficient methods exist!

General problems in collision detection

Collsion detection is never perfect!

Objects may fall through cracks in polygon meshes and collision may be missed! Usually the case for arbitrary polgyon meshes created by a content-creation tool.

Emergency plans for such situations can be made:

- Use an infinite plane underneath a hill terrain. Will catch an object that falls through.
- Can be prevented to some extend by fixing the mesh by detecting cracks etc. Can be done by hand or automatically.

Numerical errors such as round-off errors can also give incorrect collision results. Care must be taken when programming the various computations (e.g. temporarily change to higher precision, reorder terms, etc.).