OBBTree: A Hierarchical Structure for Rapid Interference Detection

S. Gottschalk, M.C. Lin, D. Manocha

Presented by Jeff Wang

Overview

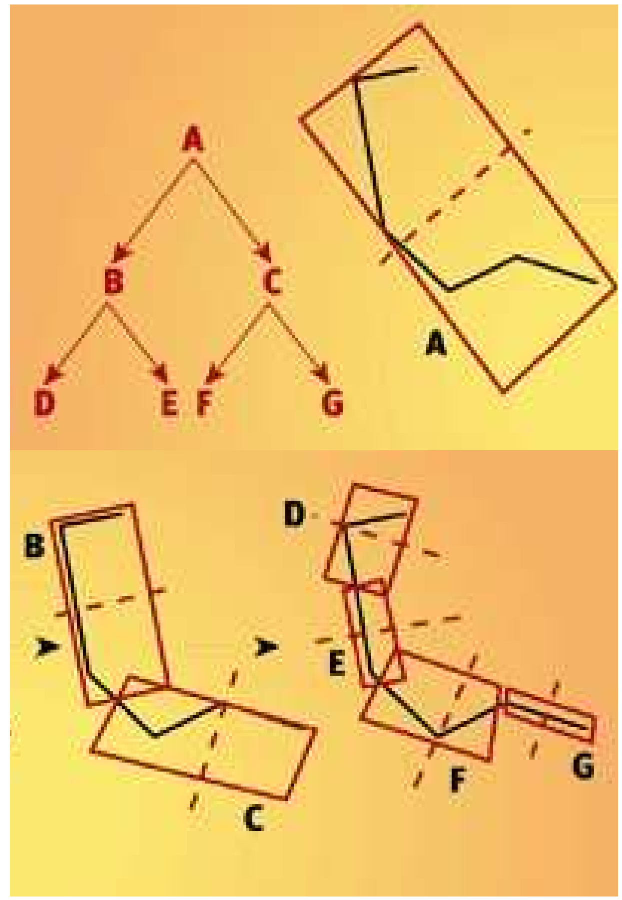
- Introduction - research goals
- Previous Work
- OBBTree Construction
- Collision Testing

Requirements for Collision Detection Algorithms

- Model Complexity
- Unstructured Representation
- Close Proximity & Multiple Contact Pts
- Accurate Contact Determination

Previous Work

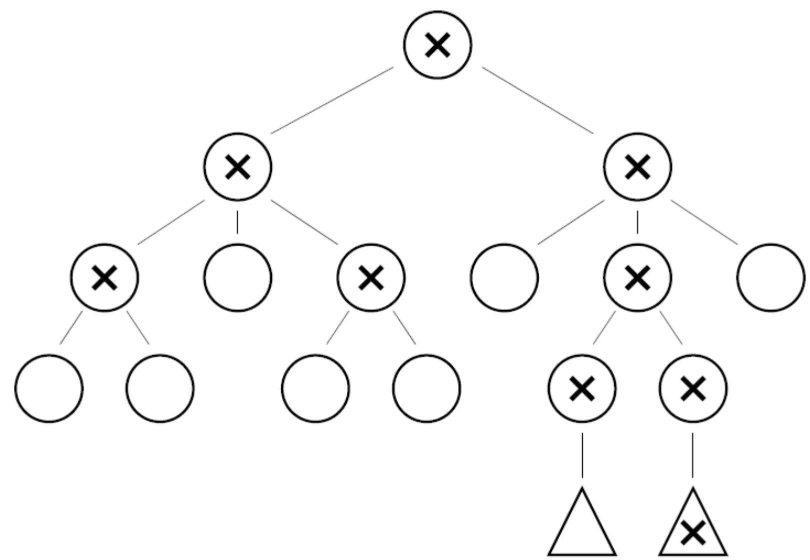
- Hierarchical Methods
  - Model represented as hierarchical bounding volumes
  - At each internal node: BV that encloses all children in its volume
  - At each leaf: one or more prims
  - Bottom-up or top-down



[gamasutra.com - Adv col det techniques]

Testing Using Hierarchy Trees

- Take two trees see if they overlap
- If yes then recursively test subtrees
  - If subtrees do not intersect, then stop
  - If subtrees do intersect, process until leaf node to find out location



[Gottenschalk PhD Thesis, UNC]

Cost Function

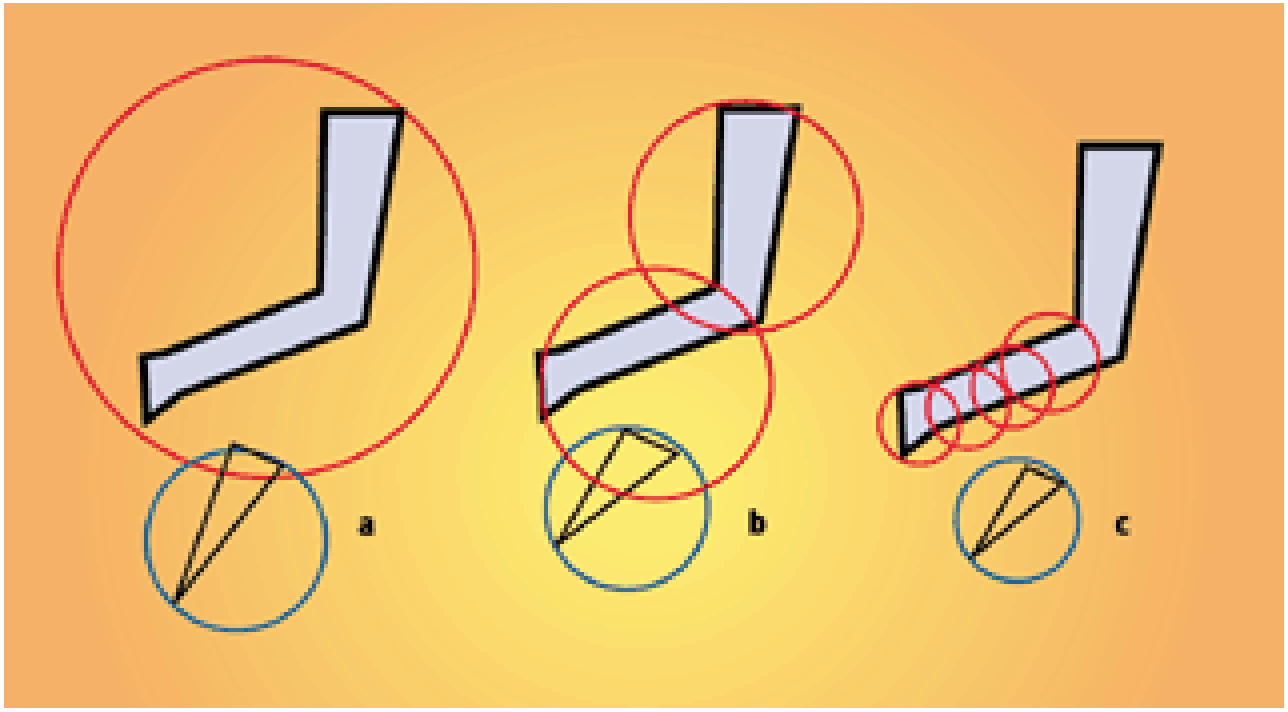
- Use to analyze cost of hierarchical structures:

$$T = N_v \times C_v + N_p \times C_p$$

N_v & C_v: number and cost for BV tests

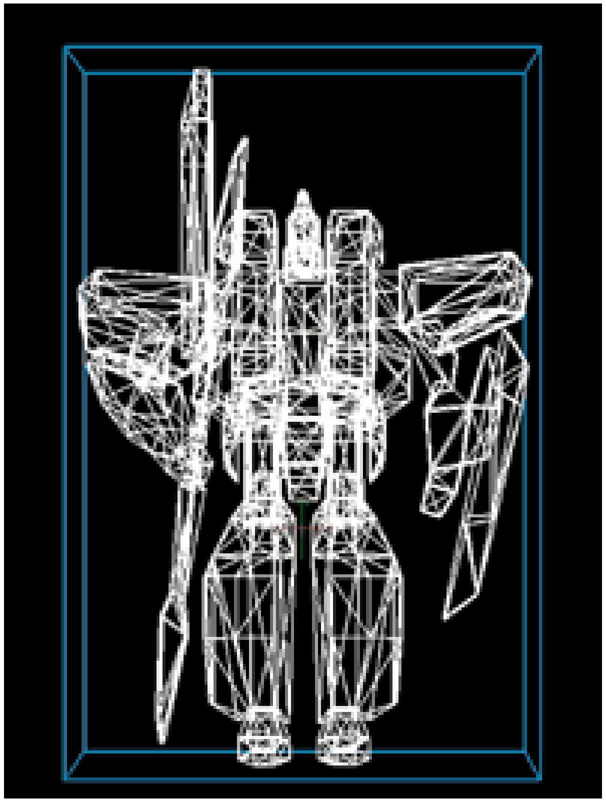
N_p & C_p: number and cost for primitive pair tests

Sphere Trees



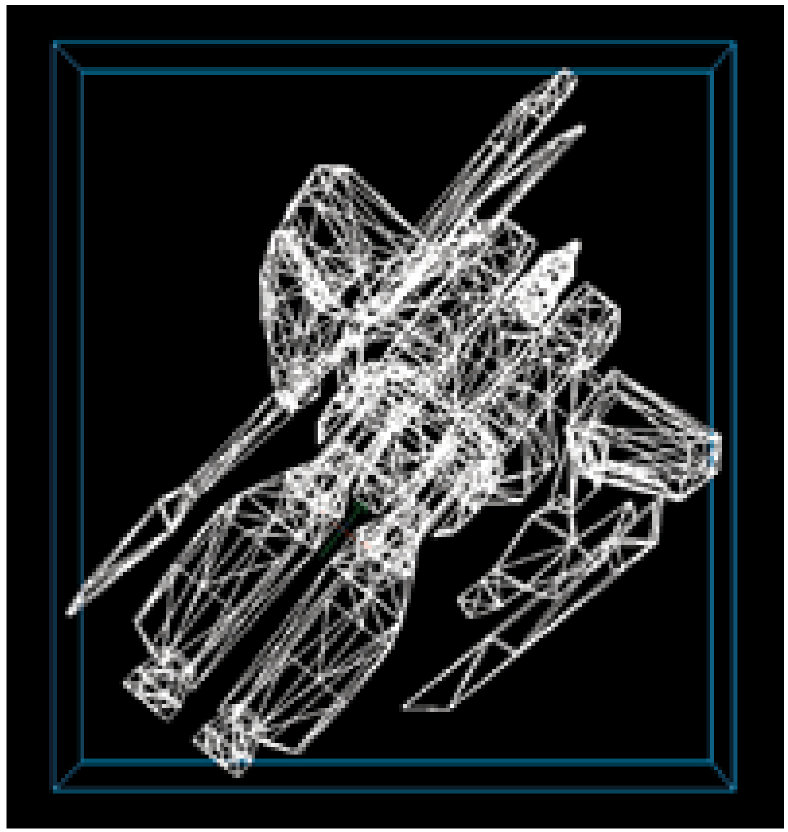
[gamasutra.com - Adv col det techniques]

AABB Trees



[gamasutra.com - When two hearts collide]

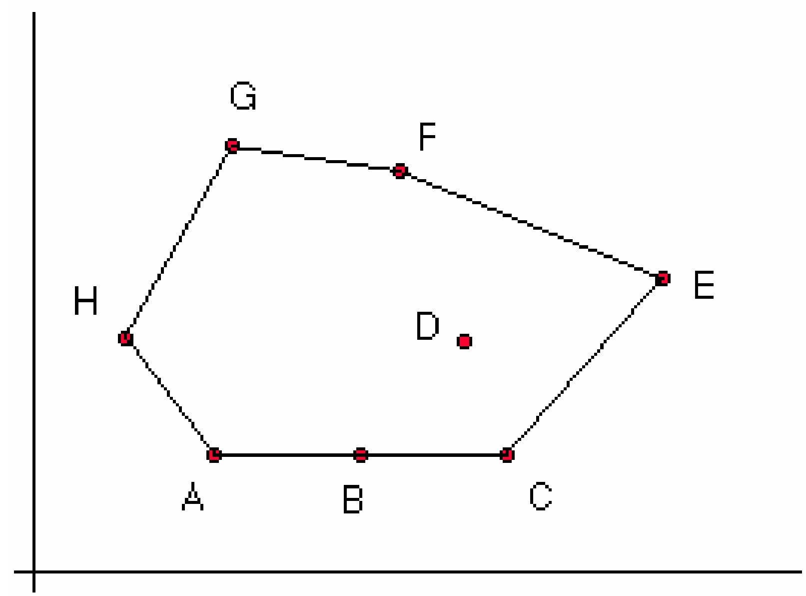
AABB Trees (cont'd)



[gamasutra.com - When two hearts collide]

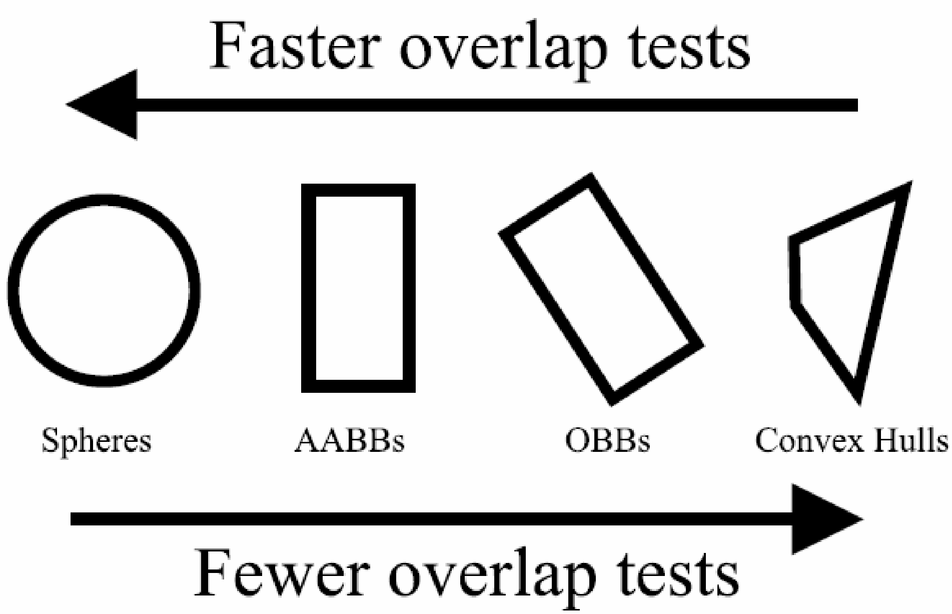
Convex Hulls

- Basically, the tightest polygon that contains a set of points



[www.cs.oberlin.edu/classes/dragn/labs/compgeom/compgeom1.html]

Summary of Bounding Volumes



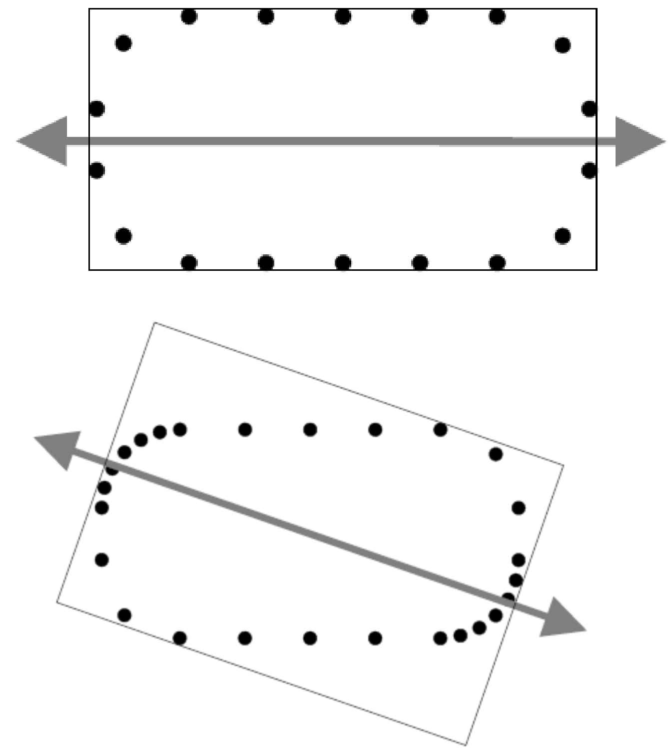
[Gottenschalk PhD Thesis, UNC]

Construction of OBBTree

- Two main steps
  - Place tight fitting OBB around a collection of polygons
  - Grouping of nested OBBs into a tree hierarchy

Bounding Box Placement

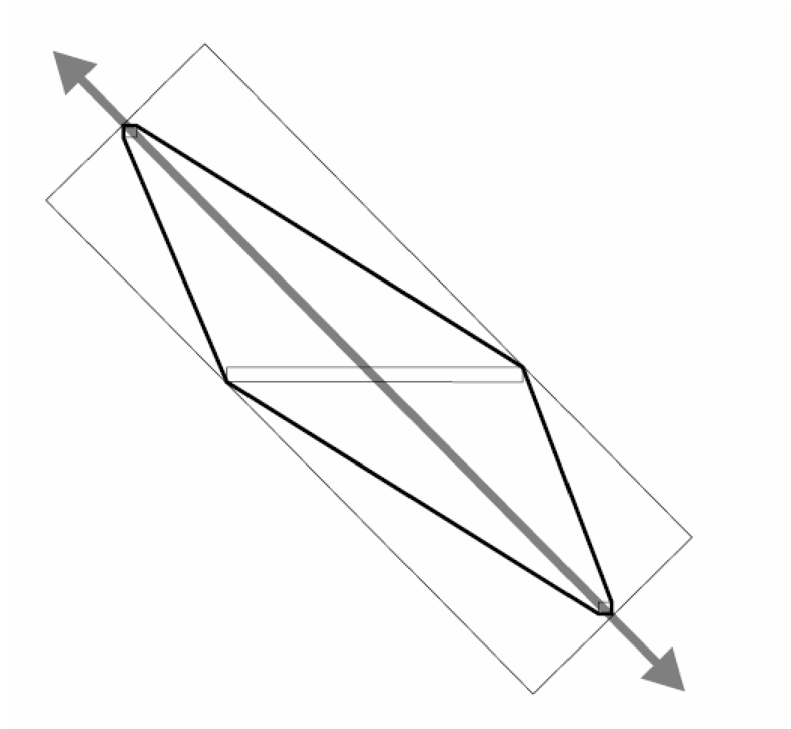
- Use statistical distribution of geometry
  - Cannot just use points
  - Sample infinitely densely by integrating over surface of convex hull



[Gottenschalk PhD Thesis, UNC]

Bounding Box Placement

- Use statistical distribution of geometry
  - Cannot just use points
  - Sample infinitely densely by integrating over surface of convex hull



[Gottenschalk PhD Thesis, UNC]

Bounding Box Placement

- Parameterize triangle using *s,t*

$$\mathbf{x}^i = \mathbf{p}^i + s(\mathbf{q}^i - \mathbf{p}^i) + t(\mathbf{r}^i - \mathbf{p}^i), \qquad s,t \in [0,1]$$

  - p^i, s^i, r^i: points on i^{th} triangle

- Mean point of convex hull

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{m^i}\int_0^1\int_0^{1-t}\mathbf{x}^i\,ds\,dt\right) = \frac{1}{6n}\sum_{i=1}^{n}\frac{1}{m^i}(\mathbf{p}^i+\mathbf{q}^i+\mathbf{r}^i)$$

  - n: number of triangles
  - m^i: area of i^{th} triangle

Bounding Box Placement

- Covariance Matrix
  - Describes shape of point cloud
  - Equivalent to 2D normal Gaussian distribution curve on a number line

$$\mathbf{C}_{jk} = \frac{1}{24n} \sum_{i=1}^{n} m^i [(\overline{\mathbf{p}}_j^i + \overline{\mathbf{q}}_j^i + \overline{\mathbf{r}}_j^i)(\overline{\mathbf{p}}_k^i + \overline{\mathbf{q}}_k^i + \overline{\mathbf{r}}_k^i) + \overline{\mathbf{p}}_j^i \overline{\mathbf{p}}_k^i + \overline{\mathbf{q}}_j^i \overline{\mathbf{q}}_k^i + \overline{\mathbf{r}}_j^i \overline{\mathbf{r}}_k^i], \qquad 1 \le j,k \le 3$$

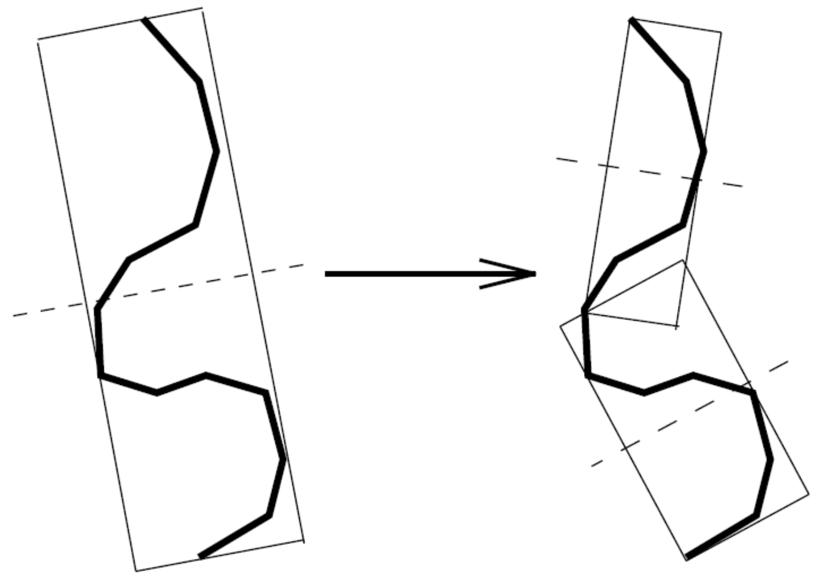
where $\overline{\mathbf{p}}^i = \mathbf{p}^i - \mu$, $\overline{\mathbf{q}}^i = \mathbf{q}^i - \mu$, and $\overline{\mathbf{r}}^i = \mathbf{r}^i - \mu$.

Bounding Box Placement

- Actually make the box
  - Normalize eigenvectors to use as a basis
  - Find extreme verts along each axis of basis
  - Size and orient bounding box
  - Bind to extreme verts

Hierarchy Construction

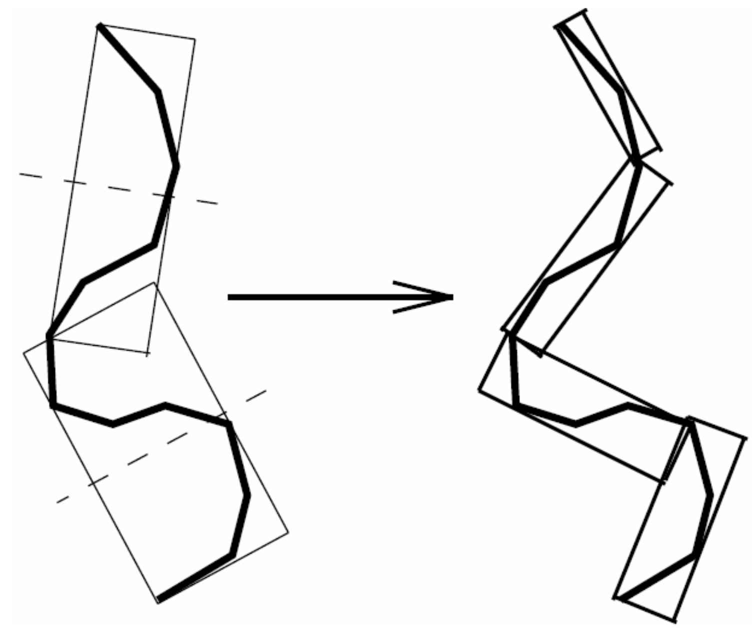
- Top Down Subdivision
  - Split longest axis of box with plane orthogonal to one of its axes
  - Partition based on which side of plane center pt lies on



[Gottenschalk et al, SIGGRAPH ‘98]

Hierarchy Construction

- If you can’t split along longest axis, subdivide along 2nd longest
- Continue until you can’t split volume anymore



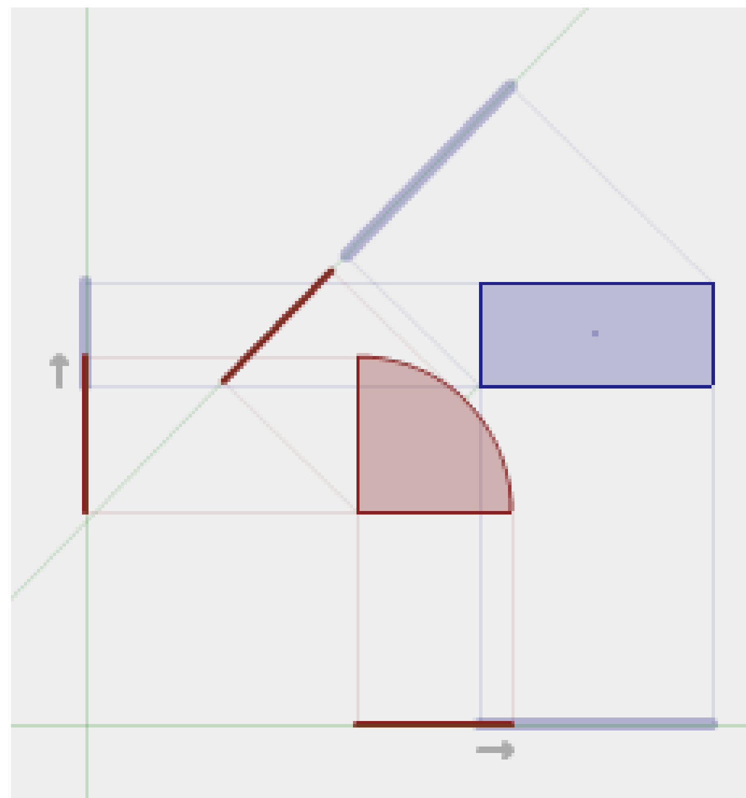
[Gottenschalk et al, SIGGRAPH ‘98]

Overlap Testing for OBBs

- Previous algorithms
  - Simple edge testing (144 inequalities)
  - Linear programming
  - Closest features computation
- This paper: separating axis theorem

Separating Axis Theorem

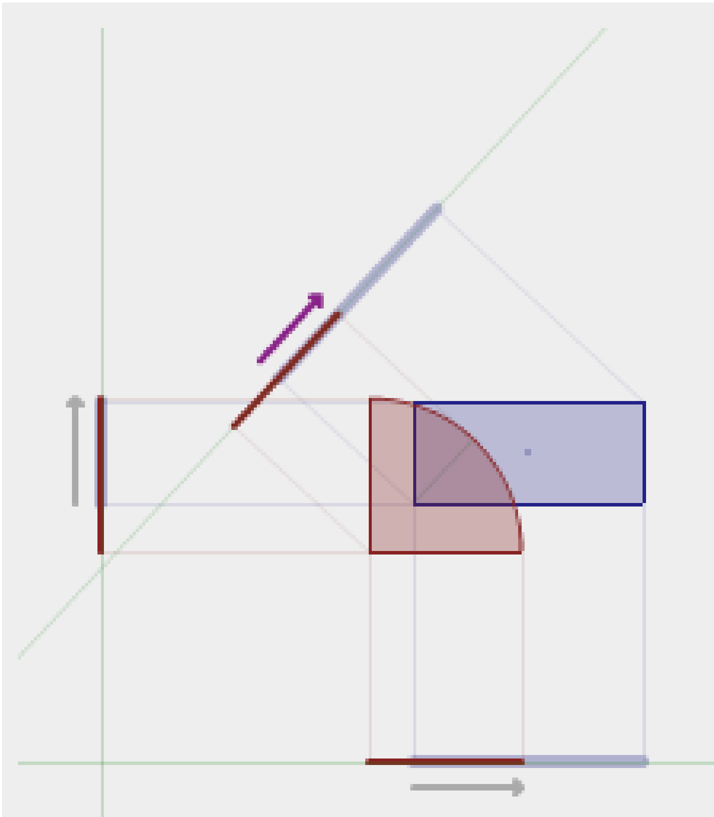
- Two polytopes are disjoint iff there exists a separating axis which is either perpendicular to a face of one of the polytopes or perpendicular to an edge from each



[www.harveycartel.org/metanet/tutorials/tutorialA.html]

Separating Axis Theorem

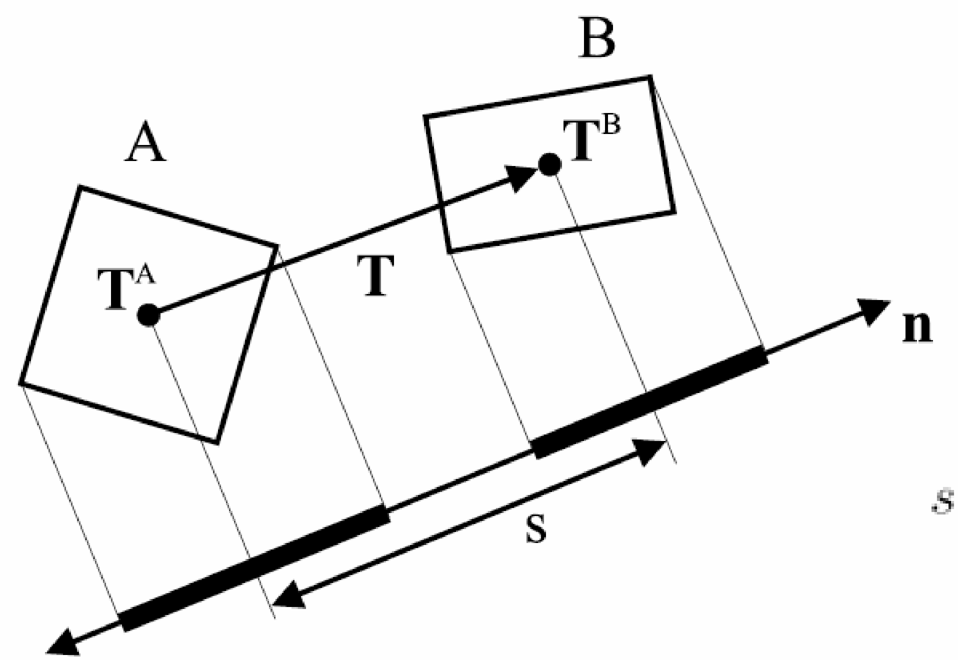
- Two polytopes are disjoint iff there exists a separating axis which is either perpendicular to a face of one of the polytopes or perpendicular to an edge from each



[www.harveycartel.org/metanet/tutorials/tutorialA.html]

Testing Procedure

- Project centers of the boxes onto the axis

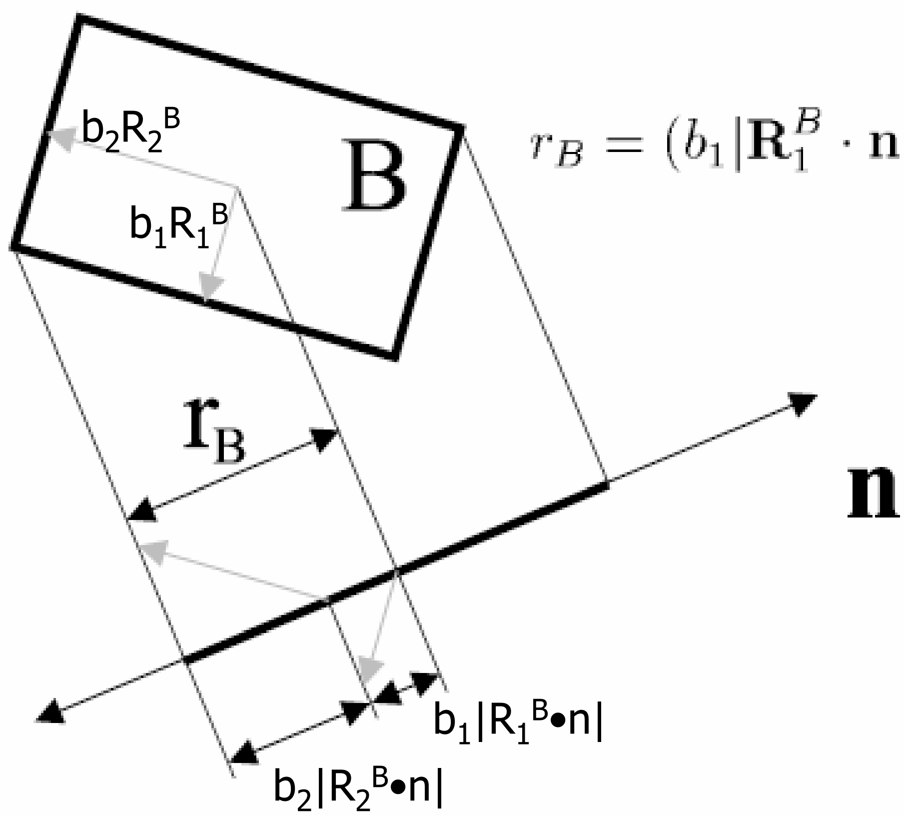


$$s = |(\mathbf{T}^A - \mathbf{T}^B) \cdot \mathbf{n}| / |\mathbf{n}|$$

[Gottenschalk PhD Thesis, UNC]

Testing Procedure

- Compute radius of intervals

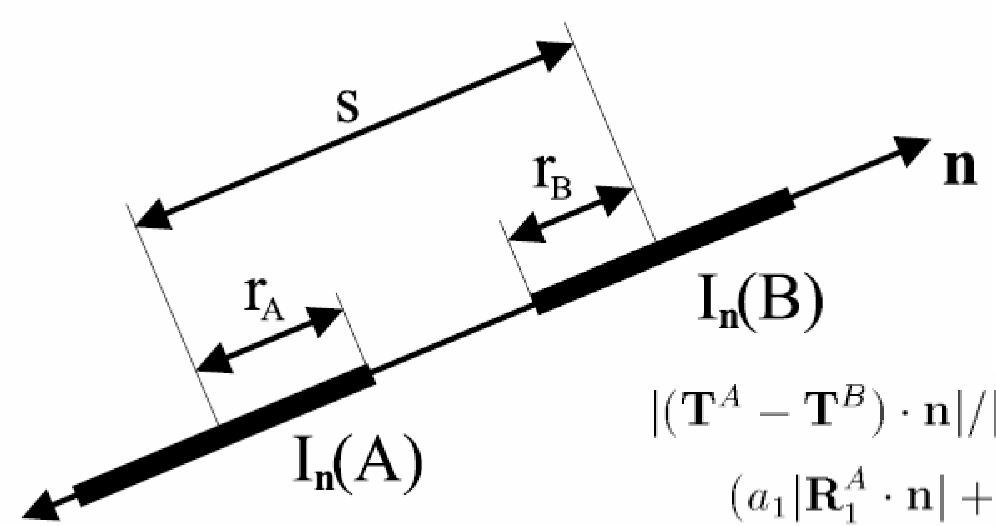


$$r_B = (b_1|\mathbf{R}_1^B \cdot \mathbf{n}| + b_2|\mathbf{R}_2^B \cdot \mathbf{n}| + b_3|\mathbf{R}_3^B \cdot \mathbf{n}|)/|\mathbf{n}|$$

[Gottenschalk PhD Thesis, UNC]

Testing Procedure

- Two intervals are disjoint iff the separation of their midpoints is greater than sum of half-widths



$$
\begin{aligned}
&|(\mathbf{T}^A - \mathbf{T}^B) \cdot \mathbf{n}|/|\mathbf{n}| > \\
&\quad (a_1|\mathbf{R}_1^A \cdot \mathbf{n}| + a_2|\mathbf{R}_2^A \cdot \mathbf{n}| + a_3|\mathbf{R}_3^A \cdot \mathbf{n}|)/|\mathbf{n}| + \\
&\quad (b_1|\mathbf{R}_1^B \cdot \mathbf{n}| + b_2|\mathbf{R}_2^B \cdot \mathbf{n}| + b_3|\mathbf{R}_3^B \cdot \mathbf{n}|)/|\mathbf{n}|
\end{aligned}
$$

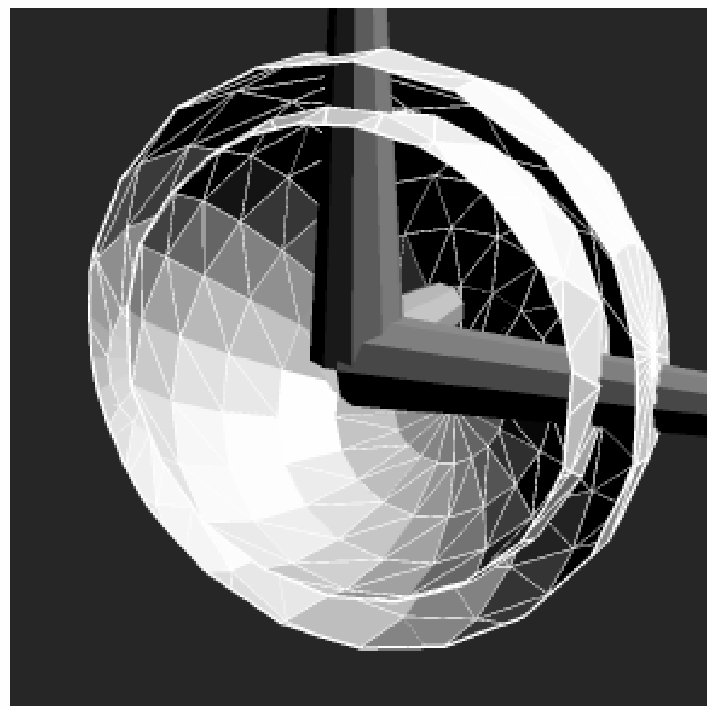
[Gottenschalk PhD Thesis, UNC]

Results: It's Fast!

- OBB vs. AABB
  - OBB: 1/7 - 1/5 sec
  - AABB: 1/25 - 1/75 sec
- OBB Overlap Testing
  - Separating Axis Theorem: 5-7 μs
  - Closest Features: 45-105 μs
  - Linear Programming: 180-230 μs

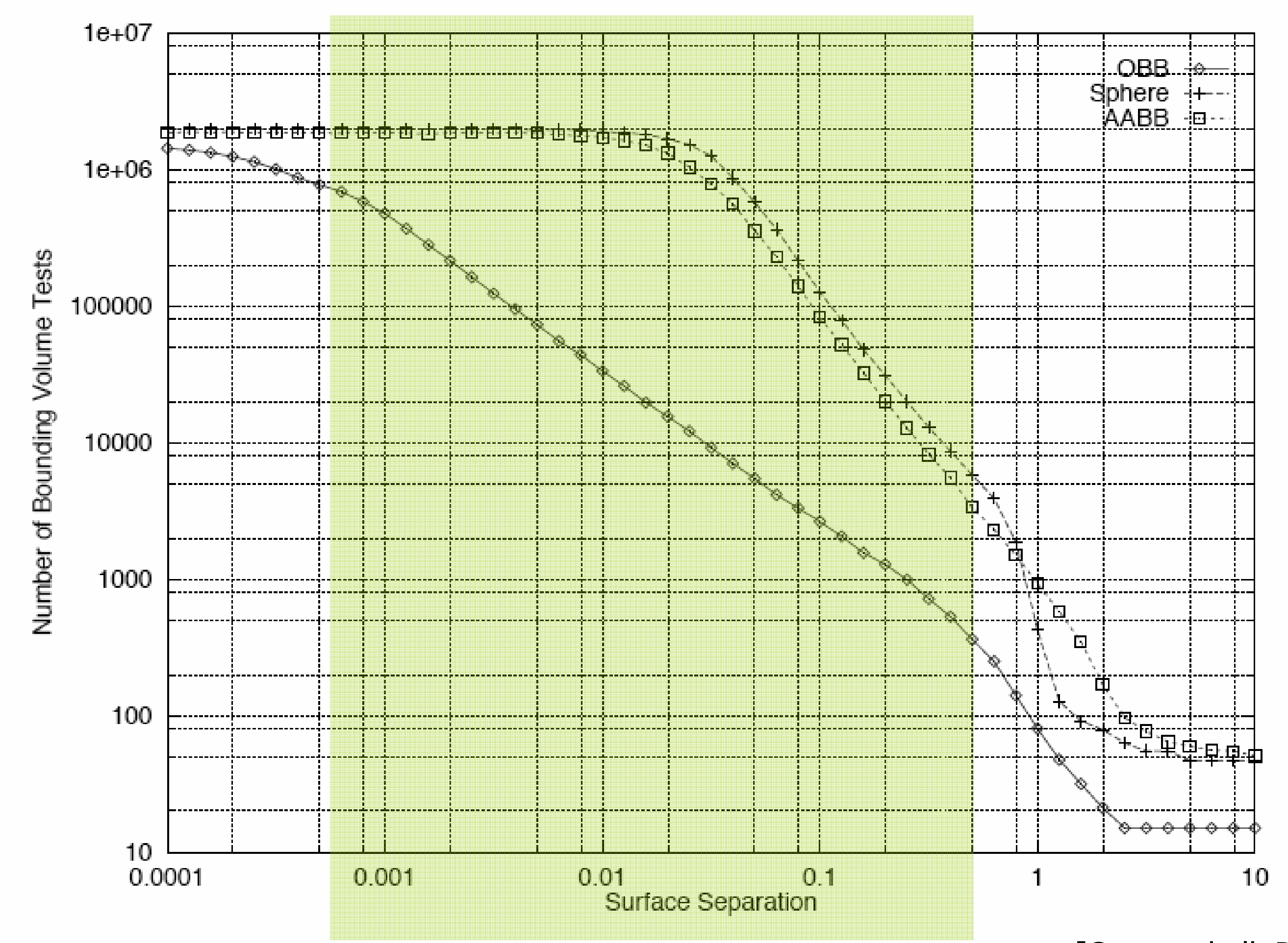
Parallel Close Proximity

- Every pt on obj is the same distance from other object
- Approximates two smooth, highly tesselated surfaces



[Gottenschalk PhD Thesis, UNC]

Parallel Close Proximity



[Gottenschalk PhD Thesis, UNC]

Critical Analysis

- No bounding volume type works best for all occasions
- OBBTrees are better for collision detection in parallel close proximity

Advantages of OBBTree

- Independent of input geometry
  - Tight fit in all scenarios leads to lower number of tests
    - Exceptions
      - Coarsely tessellated surfaces
      - Parallel close proximity does not exist

Requirements for Collision Detection Algorithms

- Model Complexity
- Unstructured Representation
- Close Proximity & Multiple Contact Pts
- Accurate Contact Determination

Revised Requirements for Collision Detection Algorithms

- Model Complexity
- Unstructured Representation
- Close Proximity & Multiple Contact Pts
- Accurate Contact Determination
- *Run-time Hierarchy*

Disadvantages of OBBTree

- Deforming objects must be computed at run-time
- Only considers polygons. What about curved surfaces?