05: BROAD PHASE COLLISION DETECTION

01/02/2016
In order to address the above issues, collision detection is often broken down further (two phase collision detection).

Naïve collision detection problems [Hubbard 1993]

1. All-pairs weakness
2. Pair Processing weakness
3. Fixed timestep weakness

**Main Purpose:** quickly prune away pairs of objects from more detailed collision/contact processing

**Faster than** $O(N^2)$ **performance achieved by exploiting certain features typically present in animation data**

- Locality
- Coherency
- Kinematic knowledge

**Typical solutions**

I. Bounding volumes
II. Spatial subdivision
III. Sweep and sort / Sweep and Prune
IV. Space-Time Bounds
I. BOUNDING VOLUMES
**Key issues:** representation  |  construction  |  update  |  collision detection
**AXIS-ALIGNED BOUNDING BOX (AABB)**

**Definition:** Box with edges that always align to the major axes of the coordinate system

**Representation:** 6 floats (limits in each dimension)

**Creation:** Find min/max in x, y, z

**Collision check:** Get intervals of projections on each coordinate axis for every body and check if these overlap; 3 x 1d Interval testing

**Advantages:** Computationally efficient

**Disadvantages:**
- Unsatisfying fill efficiency
- Not invariant to rotation:
  - requires dynamic update of AABB OR use very large boxes enclosing object in all orientations

Spaceship image: free sprite by millionthvector (http://millionthvector.blogspot.ie/2013/07/free-alien-top-down-spaceship-sprites.html)
**Definition:** the smallest sphere that encloses object

**Representation:** 4 floats (centre, radius)

**Creation:** Find minimum enclosing sphere
- min+max (xyz); center; min radius

**Collision check**
- Let \( d = \text{distance between centres of 2 spheres} \)
- Then spheres are colliding \( IF (d < (r_A + r_B)) \)

**Advantages:**
- Invariant to rotation; computationally efficient
- Update is simply a translation

**Disadvantages:** Not good fit for long/flat objects
**Oriented Bounding Box (OBB)**

**Definition:** Box with edges that align to object such that it fits optimally in terms of fill efficiency

**Representation:** 15 floats (position, orientation, extents)

**Creation:** Manual or by PCA based fitting (see later)

**Collision test:** Map to box reference coords OR Use separating axis theorem

**Advantages:**
- Invariant to rotation
- Tighter bounds than AABB and spheres

**Disadvantages:**
- Computationally more expensive to generate
- Harder to implement
**SEPARATING AXIS THEOREM**

**Separating Axis:**
- An axis on which the projections of two polytopes don’t overlap.
  - But there are an infinite number of potential axes; we can’t try them all.

**Separating Axis Theorem:**
- Two convex polytopes are disjoint if and only if there exists a separating axis orthogonal to a face of either polytope or orthogonal to an edge from each polytope.
  - Reduces number of potential axes we need to test.
Two convex polytopes are disjoint \textbf{IFF} there exists a separating axis

- orthogonal to a face of either polytope

OR

- orthogonal to an edge from each polytope.
IMPLICATIONS OF THEOREM

- Given two generic polytopes, each with $E$ edges and $F$ faces, number of candidate axes to test is:

\[ 2F + E^2 \]

- OBBs have only $E = 3$ distinct edge directions, and only $F = 3$ distinct face normals. OBBs need at most 15 axis tests.

- Because edge directions and normals each form orthogonal frames, the axis tests are rather simple.
### Number of Axes to Test

<table>
<thead>
<tr>
<th>3D Objects</th>
<th>Face dirs (A)</th>
<th>Face dirs (B)</th>
<th>Edge dirs (AxB)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment–Tri</td>
<td>0</td>
<td>1</td>
<td>1x3</td>
<td>4</td>
</tr>
<tr>
<td>Segment–OBB</td>
<td>0</td>
<td>3</td>
<td>1x3</td>
<td>6</td>
</tr>
<tr>
<td>AABB–AABB</td>
<td>3</td>
<td>0(3)</td>
<td>0(3x0)</td>
<td>3</td>
</tr>
<tr>
<td>OBB–OBB</td>
<td>3</td>
<td>3</td>
<td>3x3</td>
<td>15</td>
</tr>
<tr>
<td>Tri–Tri</td>
<td>1</td>
<td>1</td>
<td>3x3</td>
<td>11</td>
</tr>
<tr>
<td>Tri–OBB</td>
<td>1</td>
<td>3</td>
<td>3x3</td>
<td>13</td>
</tr>
</tbody>
</table>

OBB OVERLAP TEST: AXIS TEST

- \( L \) is a separating axis IFF \( s > r_A + r_B \)
  - \( r_A \) and \( r_B \) are the half-length of interval spanned by the projection of the box onto \( L \) (somewhat like the radius)
OBB OVERLAP TEST: AXIS TEST DETAILS

- Box centers project to interval midpoints, so midpoint separation $s$ is length of vector $T$'s image.

\[ s = \left| (C_A - C_B) \cdot \hat{i} \right| \]

$\hat{i}$ is a normalised vector in direction of candidate axis $L$

$C_A$ and $C_B$ are the centres of respective OBB's
OBB OVERLAP TEST: AXIS TEST DETAILS

Half-length of interval is sum of box axis images.

\[ r_B = e_1^B |R_1^B \cdot \hat{l}| + e_2^B |R_2^B \cdot \hat{l}| + e_3^B |R_3^B \cdot \hat{l}| \]

OBB is typical stored as orientation matrix \( R \) and radius along each orthogonal axis \( e_1, e_2, e_3 \)

Note that \( R \) are the column vectors of \( R \)

The superscript \( B \) indicates that this is an element of body \( B \)
Table 4.1 The 15 separating axis tests needed to determine OBB-OBB intersection. Superscripts indicate which OBB the value comes from.

| $L$ | $|T \cdot L|$ | $r_A$ | $r_B$ |
|-----|----------------|--------|--------|
| $u_A^1$ | $t_1$ | $e_0^1$ | $e_0^1 \ |r_{01}| + e_1^1 \ |r_{01}| + e_2^1 \ |r_{02}|$ |
| $u_A^2$ | $t_2$ | $e_0^2$ | $e_0^2 \ |r_{20}| + e_1^2 \ |r_{21}| + e_2^2 \ |r_{22}|$ |
| $u_A^3$ | $t_3$ | $e_1^3$ | $e_1^3 \ |r_{01}| + e_1^3 \ |r_{11}| + e_1^3 \ |r_{12}|$ |
| $u_A^4$ | $t_4$ | $e_2^4$ | $e_2^4 \ |r_{20}| + e_2^4 \ |r_{21}| + e_2^4 \ |r_{22}|$ |
| $u_B^1$ | $t_1 + t_2 + t_3$ | $e_0^1 \ |r_{01}| + e_1^1 \ |r_{01}| + e_2^1 \ |r_{02}|$ |
| $u_B^2$ | $t_1 + t_3 + t_2$ | $e_1^2 \ |r_{01}| + e_1^3 \ |r_{11}| + e_1^4 \ |r_{12}|$ |
| $u_B^3$ | $t_2 + t_3 + t_2$ | $e_2^2 \ |r_{01}| + e_1^2 \ |r_{11}| + e_1^3 \ |r_{12}|$ |

N.B. the notation in this figure is from Ericson’s book and is slightly different from the previous slide.
K-DOP

Discrete oriented polytopes:

Essentially an extension of AABB/OBBS (Boxes are 6-dops)

- Similar to clipping corners of bounding box

Advantages

- Easy to compute
- Good fill efficiency
- Simple overlap test
  - generalisation of separating axis test for relevant value of k
K-DOP

6-DOP = AABB

14-DOP (corners)

18-DOP (edges)

26-DOP (edges and corners)
UNREAL 3 ENGINE EXAMPLE

http://udn.epicgames.com/Three/CollisionReference.html#K-DOP
e.g. Grid method

- Split scene up into uniform cells
- Each object keeps a record of grid cells that it overlaps with
- Only perform pair-wise collision test with objects in own cell or cell-neighbours on grid
SPATIAL HASHING

Project 2D/3D domain into 1D hashtable

- 2D/3D points used as keys
- Exploit locality

Optimization of Large-Scale, Real-Time Simulations by Spatial Hashing - E. Hastings, J. Mesit, R.K. Guha

Perfect Spatial Hashing – S. Lefebvre, H. Hoppe
(http://research.microsoft.com/en-us/um/people/hoppe/perfecthash.pdf)

© 2012 Taura J. Greig
http://openprocessing.org/sketch/60117
SPATIAL HASHING

http://users.design.ucla.edu/~mflux/p5/hashcollision2/applet/
SPATIAL SUBDIVISION

If grid size is too big in relation to objects: too many false positives in narrow phase

If grid size too small: too much work in broad phase

• single object could cover more than 1 grid position
HIERARCHICAL SPATIAL SUBDIVISION

Adaptive Grid Sizes:

- Boxes are tiled at the minimum resolution where the cells are enough to cover the bounding volume of the object.
- If $\text{res}(X)$ and $\text{res}(Y)$ are the tiling resolutions for $X$ and $Y$, then Boxes $X$ and $Y$ are close iff they overlap a common cell at resolution $\min(\text{res}(X), \text{res}(Y))$.

**Spatial Subdivision Schemes**

**Object Independent**
- Grid

**Object Dependent**
- Quadtree/Octree
- Kd-tree
- BSP Tree
III. SWEEP AND PRUNE
**Sweep and Prune**

**Create ABBB for each object**

- For each axis,
  - Create sorted list of start and end points of intervals for each box
- Traverse each list
  - Each time a startpoint is reached insert into active list
  - If endpoint is hit remove it from active list
  - If 2 or more objects are active at the same time they overlap in the specified dimension
  - Potential colliding pairs must overlap in all 3 lists

“Early out” if not overlapping in any direction

Exploit coherency e.g. insertion-sort to get log(N) performance
IV. SPACE TIME BOUNDS
RECALL: FIXED TIME-STEP WEAKNESS

Have objects collided?

Bullet through paper effect
SOLUTION: SPACE TIME BOUNDS

Continuous collision detection

Sweep tests

But this assumes constant linear velocity within the timestep
A volume encompassing all positions that a particle could be within after a given timestep $\Delta t$

- Central axis is calculated from velocity $x_2 = x_1 + v\Delta t$
- Radius is calculated from acceleration $R(t + \Delta t) \geq |a\Delta t|$
For a solid object

Parabolic Horn is grown by \( r \), the radius of bounding sphere.
Hyper-trapezoid approximation of the parabolic horn

More wasted space but quicker collision check
Further culling is possible if we take into account the direction of $a$.

- An example of exploiting kinematic information.
GJK FOR MOVING OBJECTS
FROM ERICSSON SIGGRAPH 04 TALK
SPHERE-SWEPT VOLUMES

The volume generated by sweeping a sphere over a primitive

- Effectively: Growing a primitive outward by distance $R$

- More formally: the minkowski sum of a sphere and relevant primitive

![Diagram of Point-swept sphere (PSS), Line-swept sphere (LSS), and Rectangle-swept sphere (RSS)]
CONTINUOUS COLLISION DETECTION

[Redon 2004]
**SSV DETAILS**

**Update standard: orientation + position update**

**Collision checks**

- Similar to the distance checking with the sweep primitive associated with the SSV (i.e. point, line-segment, quad) ....

- Is distance $(V_1, V_2) < r_1 + r_2$? (where $V_i$ is the sweep primitive i.e. point, line, rectangle; and $r_i$ the respective radii);

- Minkowski sum can be optimised on GPU
SSV GENERATION

Similar to OBB based upon Principle Component Analysis

- For PSS, use the largest dimension as the radius
- For LSS, use the two largest dimensions as the length and radius
- For RSS, use all three dimensions
Miguel Gomez - “Simple Intersection Tests For Games” - Gamasutra, Oct 18, 1999

Kenny Erleben et al “Physics Based Animation” (Book) Chapter 12: Broad Phase Collision Detection


