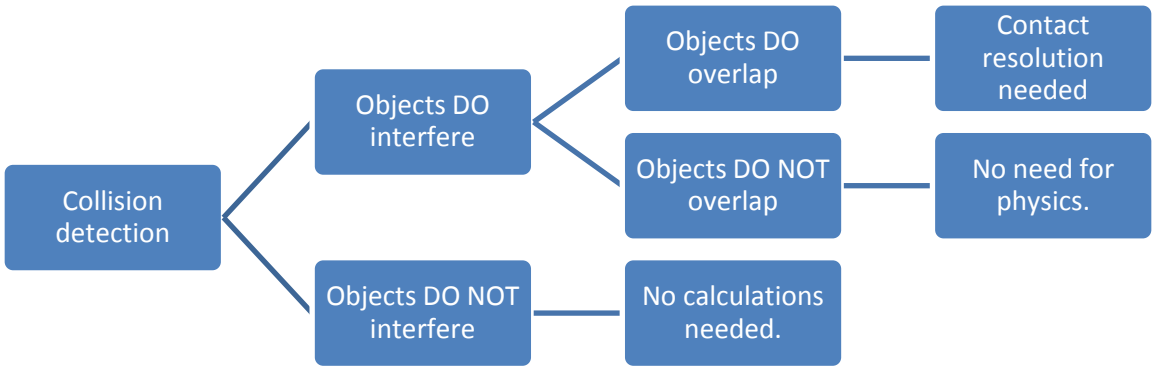
Physically Based Modeling for Interactive Simulation and Games
Scribe Notes for the lecture on February 23rd
Collision Detection
Spring 2011

Recep Doga Siyli

Collision detection involves "computational geometry". There are 2 main subject about collisions in a scene. Collision detection is the first one which takes the most of the calculation time and the other one is Collision response (in other words -> contact resolution) which involves "Physics".

1. Collision detection



2. Contact resolution

Collision can be in various ways --> point-point, many points. Also there are too many contact cases.

How to check collision? --> It depends on the objects in the scene [moving ones and stable (stationary) ones].

Physics is faster than collision detection.

There are a lot of methods for Contact resolution.

Nothing is really "rigid". So once a collision detection is detected, there is deformation. We need to do approximations for this.

Even if there is no collision, another geometric query is "how close are the two objects?".

Questions that we should ask for contact resolution are -->

1. Which triangles collide?

We can do brute force on all triangles :

a. Triangle vs. triangle

We can take both models and compare all the triangles.
For all triangles of object A
compare Tri_A_i with all triangles in object B.

b. Points vs triangle

For all points of object A
compare Point_A_i with all triangles in object B.

c. Edge vs edge

For all edges of object A
compare Edge_A_i with all edges in object B.

2. How much penetration do we have?

3. What is the smallest amount of translation needed for getting rid of penetration (Penetration depth query).

3. Classes of Objects

There are classes of objects. They have different complexities when we need collision detection.

a. Convex vs. non-convex

Convex algorithms are much faster than non-convex ones. Hence it is wise to make non-convex models --> convex models.

b. Polygonal vs. splines, curves

c. Geometric vs. Volumetric

Geometric models are like models with triangles. Volumetric models have implicit functions (level set functions) --> For example : $x^2+y^2+z^2-1=0$ is a Volumetric model of a unit sphere.

d. Rigid vs. non-rigid

For rigid bodies pre-computation is possible. We can pre-process the model for enhancing collision detection. For non-rigid object this is not available. Also for non-rigid objects "self-collisions" must be taken into account (e.g. ears of bunny hitting each other).

e. N-body vs. pair-wise

N-body--> 10000 bunnies.
Pair-wise--> Just 2 objects

f. Constructive solid geometry vs. B-splines

g. Static vs. dynamic

Houses in a scene is an example of static objects which makes collision detection simpler. Any moving abject in the scene is called dynamic.

4. Implicit functions

a. Distance Field

We can create distance fields of the objects. First we voxelize the object. For each voxel we just record the distance to surface.

5. Minkowski Sums

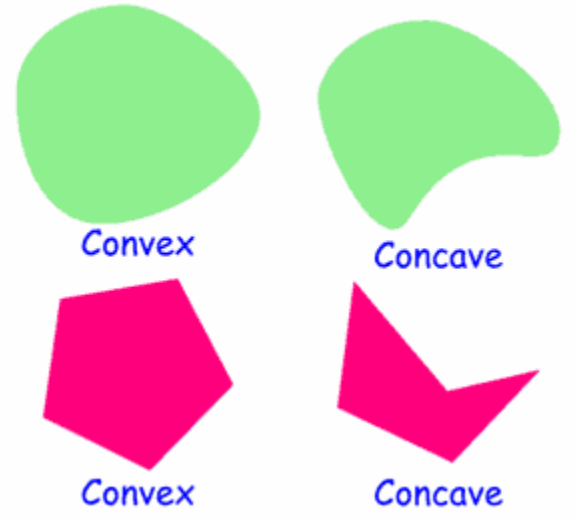
One of the easiest and most straightforward explanation for this subject is at : http://www.geometrylab.de/minkowski/index.html.en. The applets and flash on the page is very easy to understand.

6. Problem

When we want to compare 2 objects with both 'n' triangles, if they are general polyhedral then the complexity is $O(n^6)$. On the other hand if they are convex polyhedral the complexity is $O(n^2)$.

a. What is convex?

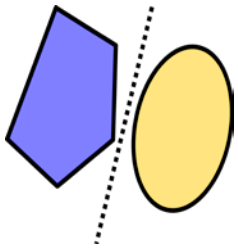
The word convex means curving out or bulging outward, as opposed to concave.



b. Separating Hyper-plane Theorem?

The theorem mainly discusses about obtaining contact information between two objects if they are both convex.

Two disjoint convex sets are separable by a hyper-plane -->



Concave objects not covered -->

More information can be found at :

http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBQQFjAA&url=http%3A%2F%2Frealtimecollisiondetection.net%2Fpubs%2FGDC07_Ericson_Physics_Tutorial_SAT.ppt&rct=j&q=separating%20hyperplane%20theorem%20GDC%20CONF&ei=cPeATZrhLY_msQOpoYmVBg&usg=AFQjCNF0fjhnhVuUali_aXWmMdb6kdLx0Q&sig2=MX8kd_H3LKlOt7Bhr9VyDg

or within the zip folder --> GDC07_Ericson_Physics_Tutorial_SAT.ppt

For 2D calculations we look for a "separating axis" whereas for 3D calculations we look for "separating plane".