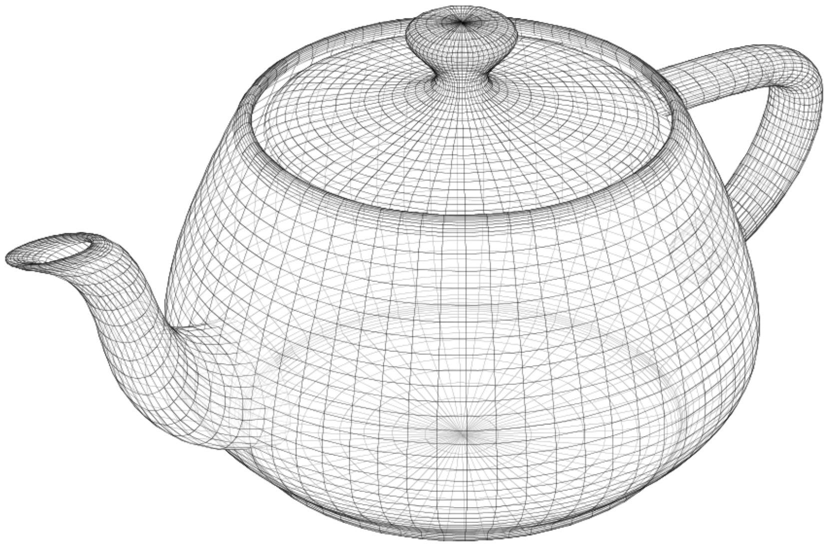


Collision Detection

Interactive Computer Graphics

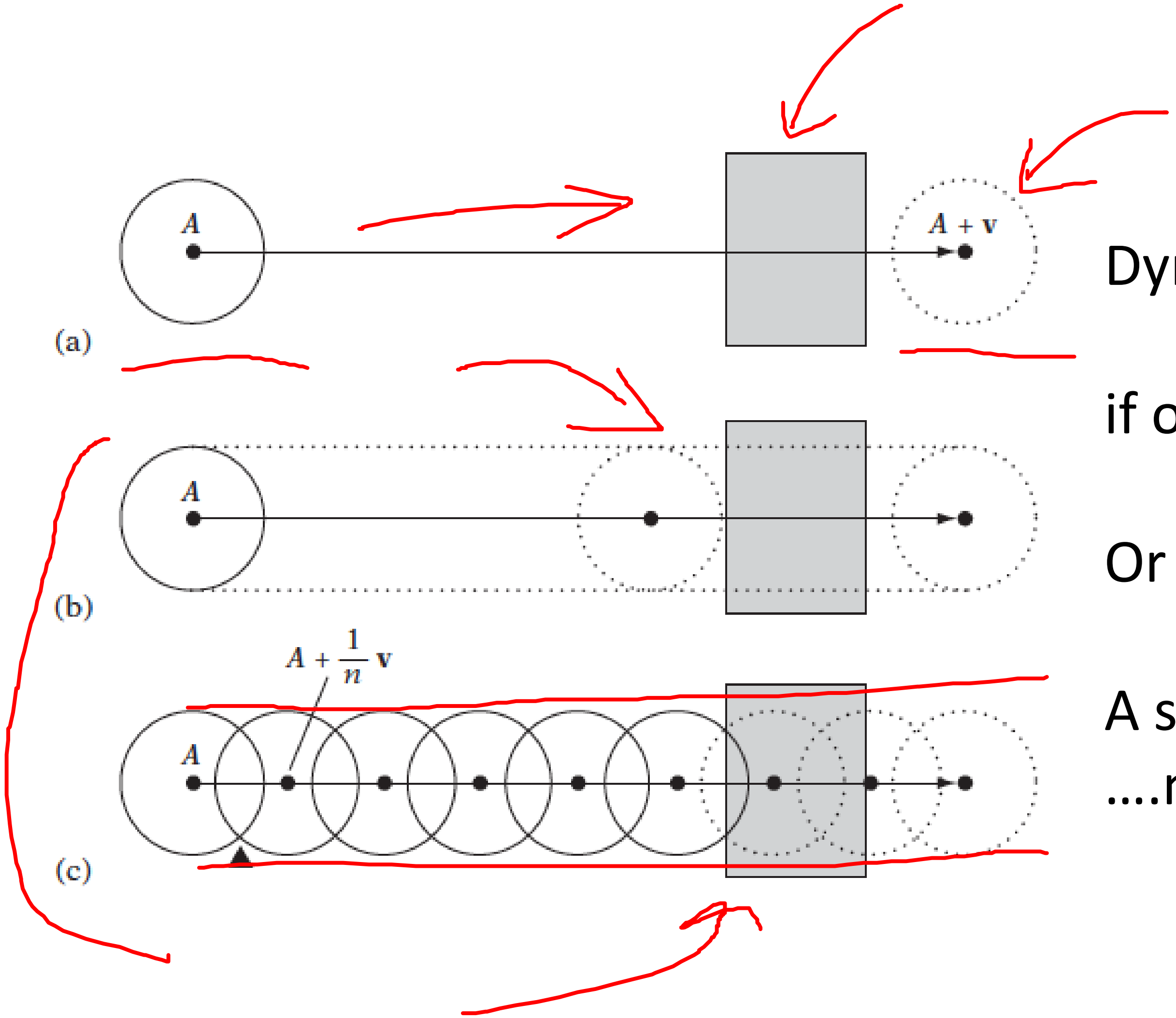
Professor Eric Shaffer

ILLINOIS

Collision Detection

- Surprisingly complex topic
  - Even a high-quality engine like Unity has issues
- We will discuss how to simulate two types of collision
  - Sphere-Wall
  - Sphere-Sphere
- We will discuss using bounding volumes to accelerate collision detection

Dynamic Collision Detection



Dynamic collision tests an exhibit *tunneling*

if only the final positions of the objects are tested (a)

Or even if the paths of the objects are sampled (c)

A sweep test assures detection
….may not be computationally feasible.

Sphere-Plane Collision

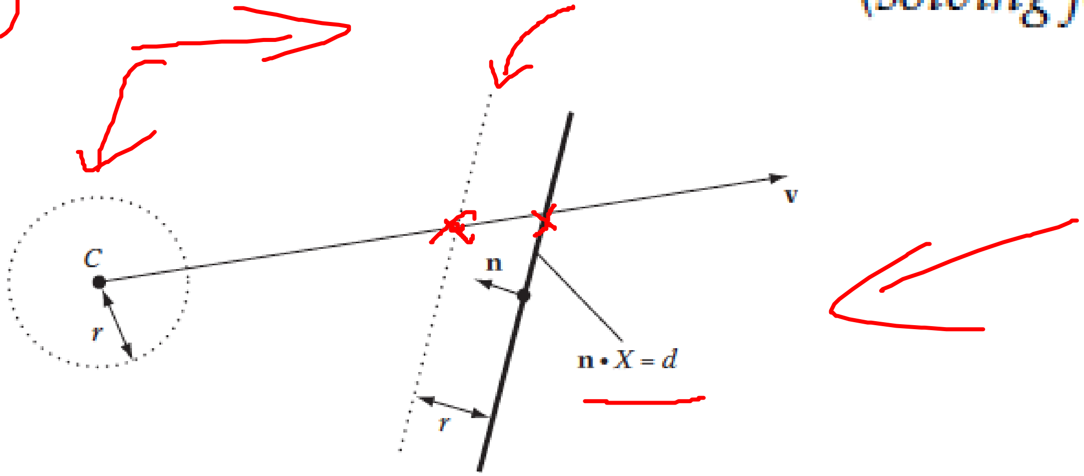
$(\mathbf{n} \cdot X) = d \pm r \Leftrightarrow$ *(plane equation for plane displaced either way)*

$\mathbf{n} \cdot (C + t\mathbf{v}) = d \pm r \Leftrightarrow$ *(substituting $S(t) = C + t\mathbf{v}$ for X)*

$(\mathbf{n} \cdot C) + t(\mathbf{n} \cdot \mathbf{v}) = d \pm r \Leftrightarrow$ *(expanding dot product)*

$t = (\pm r - ((\mathbf{n} \cdot C) - d))/(\mathbf{n} \cdot \mathbf{v})$ *(solving for t)*

Why is it $\pm r$?

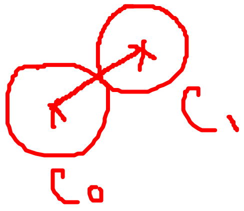


Sphere-Sphere Collision Detection

The vector $\mathbf{d}$ between the sphere centers at time t is given by

$s + t\,v$

$$\mathbf{d}(t) = (C_0 + t\mathbf{v}_0) - (C_1 + t\mathbf{v}_1) = (C_0 - C_1) + t(\mathbf{v}_0 - \mathbf{v}_1)$$



$r = r_0 + r_1$

$$\mathbf{d}(t) \cdot \mathbf{d}(t) = (r_0 + r_1)^2 \Leftrightarrow \qquad \text{(original expression)}$$

$$(\mathbf{s} + t\,\mathbf{v}) \cdot (\mathbf{s} + t\,\mathbf{v}) = r^2 \Leftrightarrow \qquad \text{(substituting } \mathbf{d}(t) = \mathbf{s} + t\,\mathbf{v}\text{)}$$

$$(\mathbf{s} \cdot \mathbf{s}) + 2(\mathbf{v} \cdot \mathbf{s})t + (\mathbf{v} \cdot \mathbf{v})t^2 = r^2 \Leftrightarrow \qquad \text{(expanding dot product)}$$

$$\underbrace{(\mathbf{v} \cdot \mathbf{v})}_{a}t^2 + 2\underbrace{(\mathbf{v} \cdot \mathbf{s})}_{b}t + \underbrace{(\mathbf{s} \cdot \mathbf{s} - r^2)}_{c} = 0 \qquad \text{(canonic form for quadratic equation)}$$

This is a quadratic equation in t. Writing the quadratic in the form $at^2 + 2bt + c = 0$, with $a = \mathbf{v} \cdot \mathbf{v}$, $b = \mathbf{v} \cdot \mathbf{s}$, and $c = \mathbf{s} \cdot \mathbf{s} - r^2$ gives the solutions for t as

$$\boxed{t = \frac{-b \pm \sqrt{b^2 - ac}}{a}.}$$

ILLINOIS

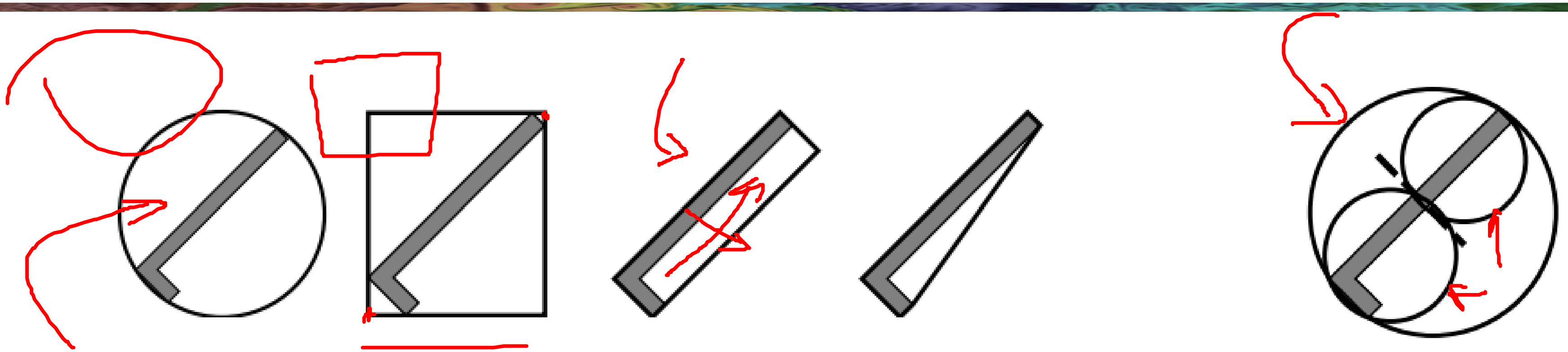
General Collision Detection

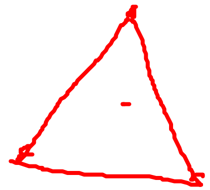
Often requires use of bounding volumes or spatial data structures

1. **Broad Phase:** Uses bounding volumes to quickly determine which objects need to be checked closely for collision

2. **Narrow Phase:** Perform careful, expensive collision checks, such as triangle-triangle intersection for meshes

Bounding Volumes





Sphere AABB OOBB Convex Hull Hierarchical (BVH)

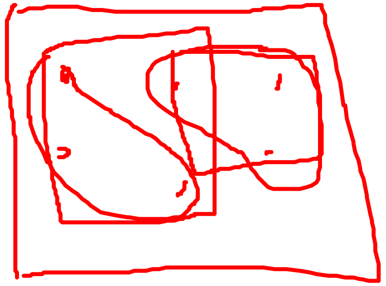
AABB=Axis Aligned Bounding Box

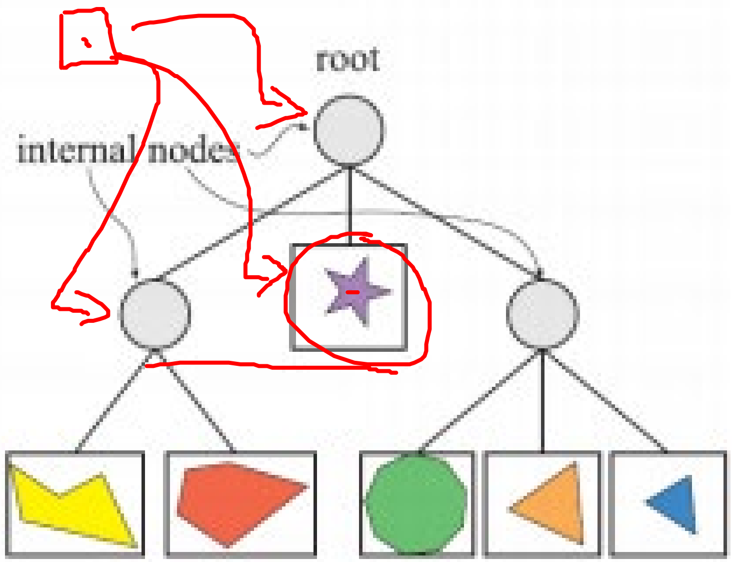
OOBB = Object-Oriented Bounding Box

BVH = Bounding Volume Hierarchy

- How much computation is needed to build the BV?
- How much computation is needed to check for BV intersections
- How many false positive collisions do the BVs generate?

BVH Construction



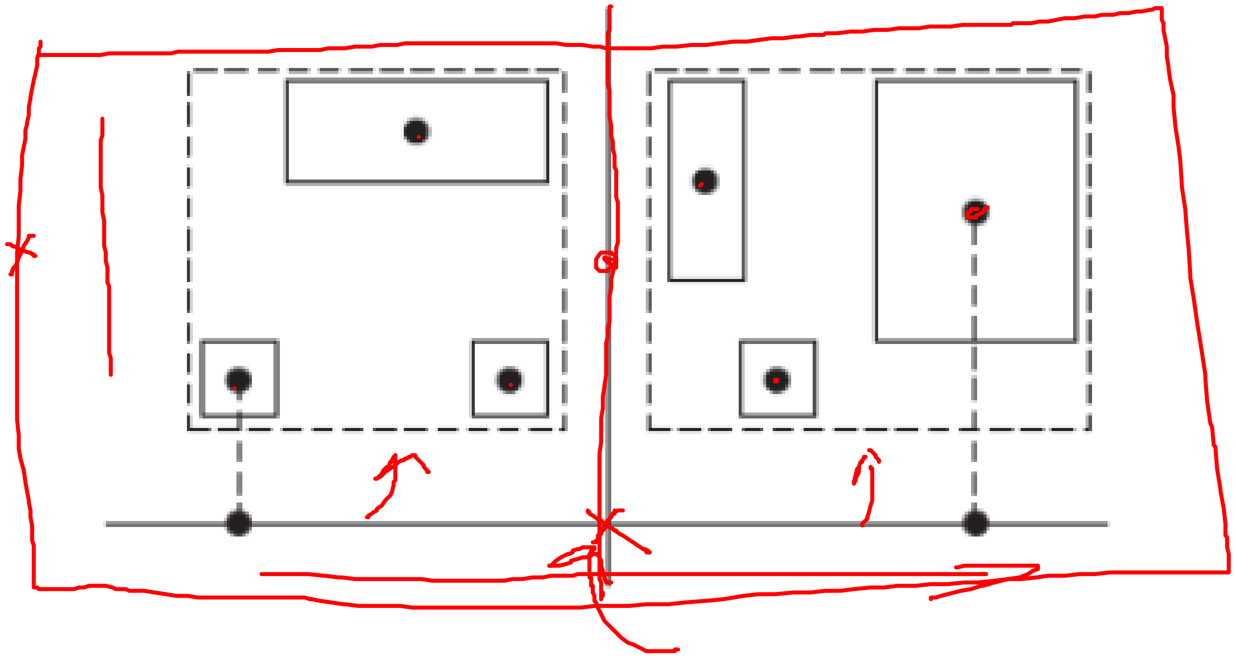


Can be constructed top down:

1. Compute bounding volume enclosing all of the geometry
2. Split the geometry into two or more groups
3. Compute bounding volume for each group
4. Recurse
5. Leaf nodes will enclose only one geometric primitive

Can also be built by bottom up merging which offers better parallelism

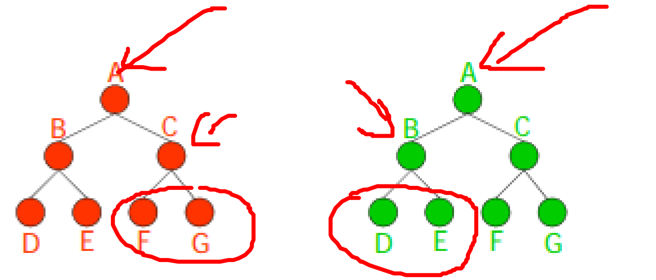
BVH: How to Split



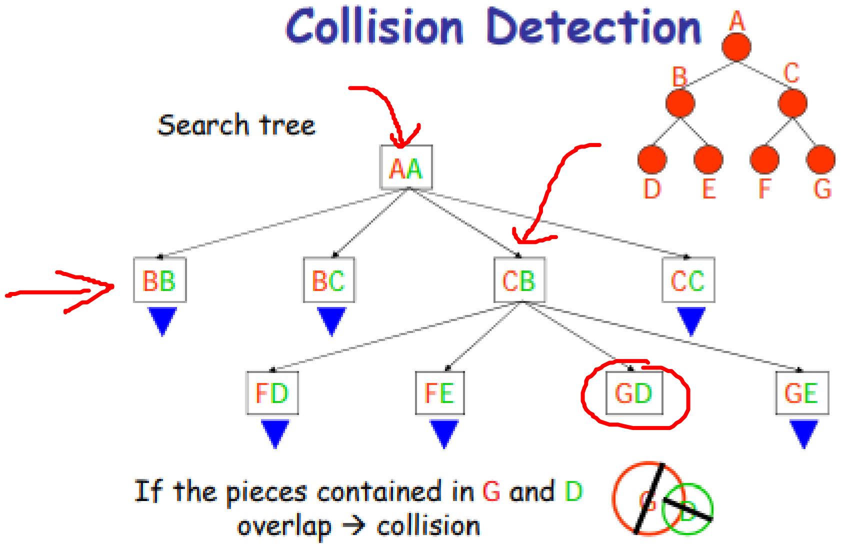
- Can compute a centroid for each geometric primitive
  - Split on median centroid, along longest axis
  - Split on average centroid, along longest axis
- More sophisticated splitting criteria can be used
  - E.g. Surface Area Heuristic used on BVHs for ray-tracing

BVH: How to Collide

- In a single BVH for scene
  - Two geometric primitives can overlap only if their volumes overlap
- Or, BVHs can be used for each composite object (e.g. mesh)
  - A search tree is constructed that records descent into each BVH
  - Determines if any cells overlap



Two objects described by their precomputed BVHs



If the pieces contained in G and D overlap → collision