CPSC 436D
Video Game Programming

Collisions

Collision Configurations?

To detect collisions between polygons it is enough to test if their edges intersect

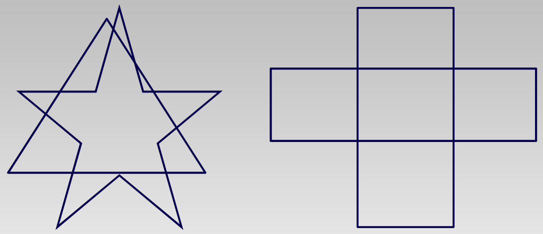
A. True

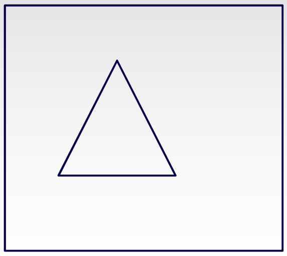
B. False

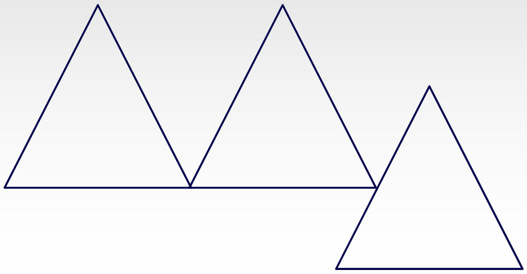
Collision Configurations?

Collision Configurations?

- Segment/Segment Intersection
  - *Point* ***on*** *Segment*
- Polygon inside polygon





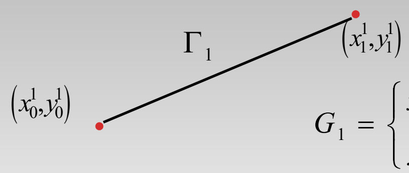


Resources

http://www.realtimerendering.com/intersections.html

Lines & Segments

Segment Γ_1 ***from*** $P_0 = (x_0^1, y_0^1)$ ***to*** $P_1 = (x_1^1, y_1^1)$



$$G_1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} t \in [0,1]$$

Line through $P_0 = (x_0^1, y_0^1)$ ***and*** $P_1 = (x_1^1, y_1^1)$

- Parametric $G_1(t), t \in (-\infty, -\infty)$
- Implicit $Ax+By+C=0$
  - *Solve 2 equations in 2 unknowns (set $A^2+B^2=1$)*

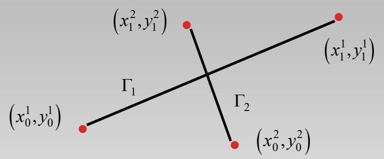
Inside Test?

- How to test if one poly is inside another?
- Use inside test for point(s)
- How?
  - *Convex Polygon*
    - Same side WRT to line equation (all sides)
  - *Non-Convex*
    - Subdivide=triangulate
    - How?
    - Shoot rays (beware of corners and special cases)

Point vs Line

- Point P=(P_x,P_y)
- Use implicit equation to determine coincidence & side
  - *Implicit* $Ax+By+C=0$
  - *Solve 2 equations in 2 unknowns (set* $A^2+B^2=1$*)*
  - *On:* $AP_x + B\,P_y + C=0$
  - *Use same orientation to get consistent left/right orientation for inside test for lines defining CONVEX polygon*
    - Same sign implies inside
    - *Eg.* ***ALL*** $AP_x + B\,P_y + C < 0$

Line-Line Intersection



$$G_1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} \quad t \in [0,1] \qquad G_2 = \begin{cases} x^2(r) = x_0^2 + (x_1^2 - x_0^2)r \\ y^2(r) = y_0^2 + (y_1^2 - y_0^2)r \end{cases} \quad r \in [0,1]$$

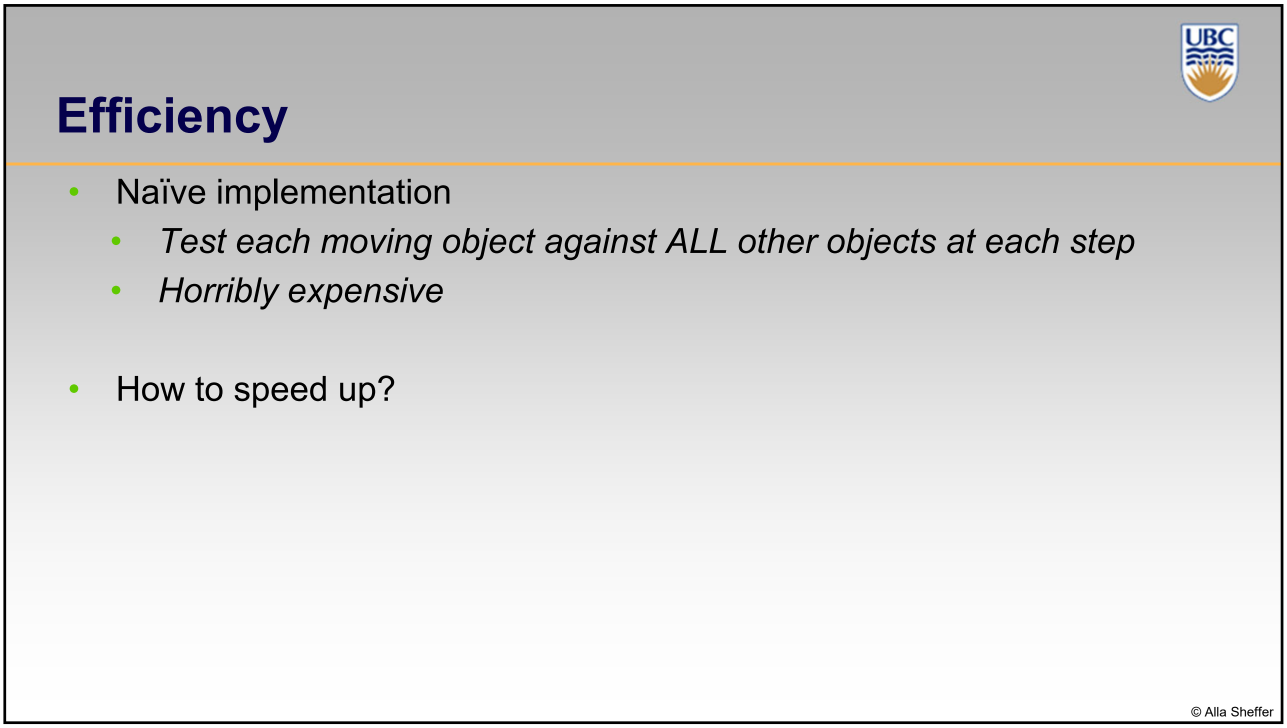
Intersection: *x* & *y* values equal in both representations - two linear equations in two unknowns (*r,t*)

$$\begin{aligned} x_0^1 + (x_1^1 - x_0^1)t &= x_0^2 + (x_1^2 - x_0^2)r \\ y_0^1 + (y_1^1 - y_0^1)t &= y_0^2 + (y_1^2 - y_0^2)r \end{aligned}$$

Question: What is the meaning of *r,t* < 0 or *r,t* >1 ?

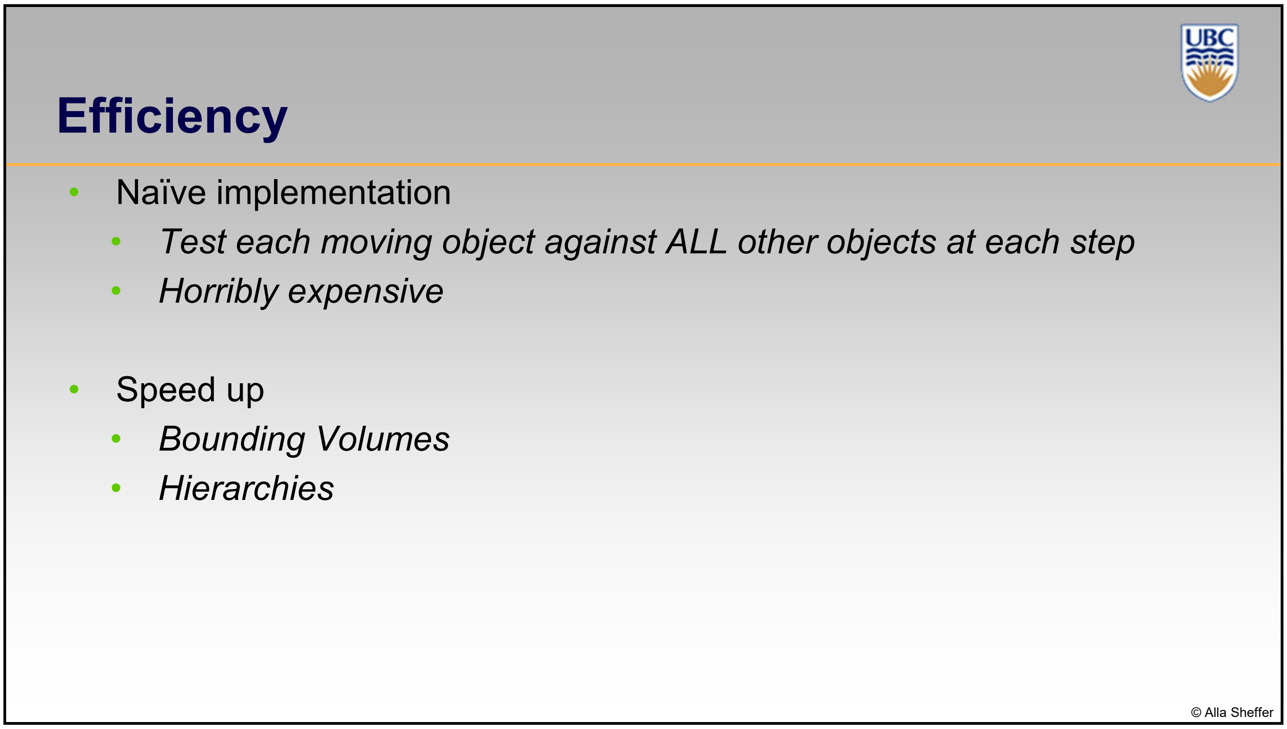
Question: What is the meaning of *r,t* < 0 or *r,t* >1 ?

A. They still collide
B. They do not collide
C. They may or maynot collide – need more testing



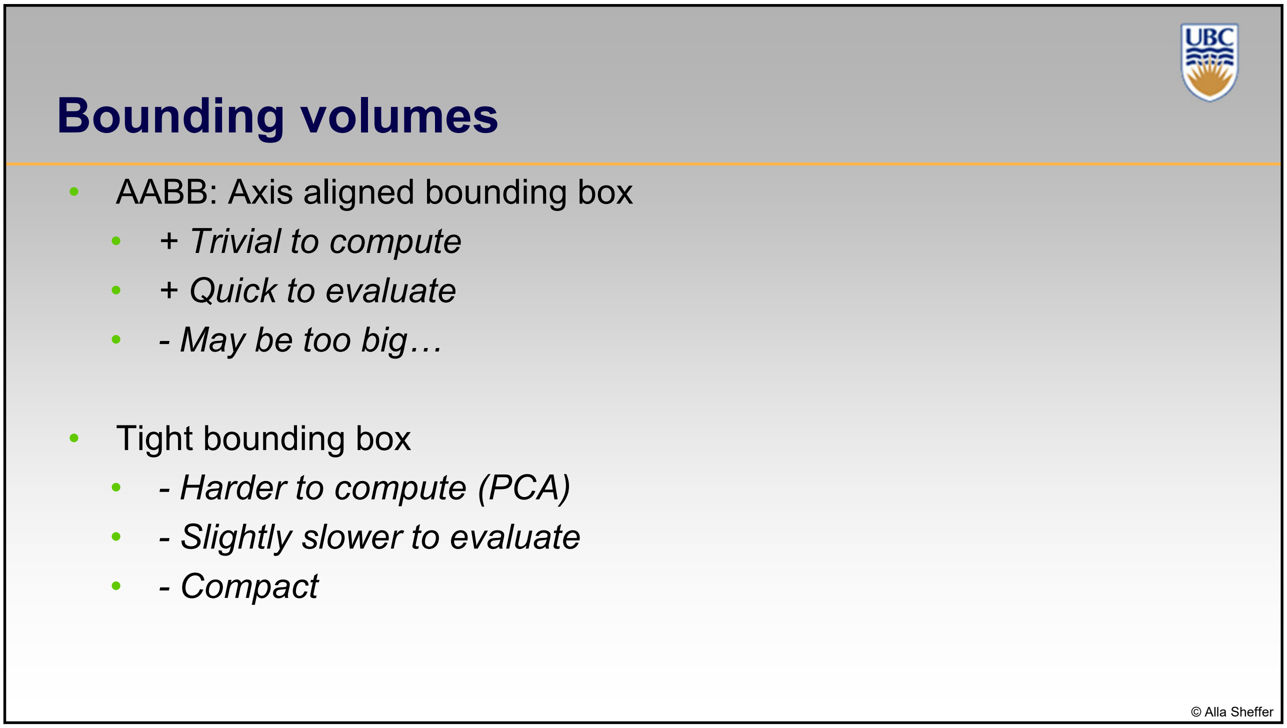
Efficiency

- Naïve implementation
  - *Test each moving object against ALL other objects at each step*
  - *Horribly expensive*
- How to speed up?



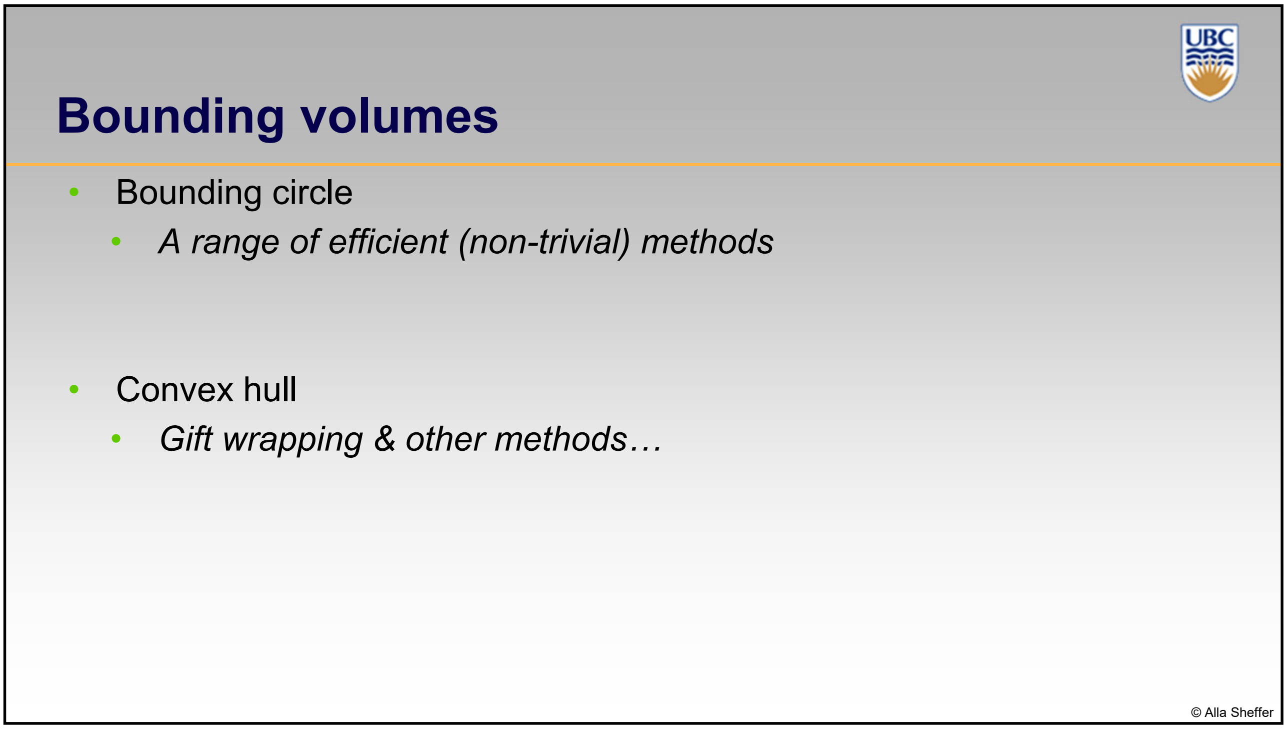
Efficiency

- Naïve implementation
  - *Test each moving object against ALL other objects at each step*
  - *Horribly expensive*
- Speed up
  - *Bounding Volumes*
  - *Hierarchies*



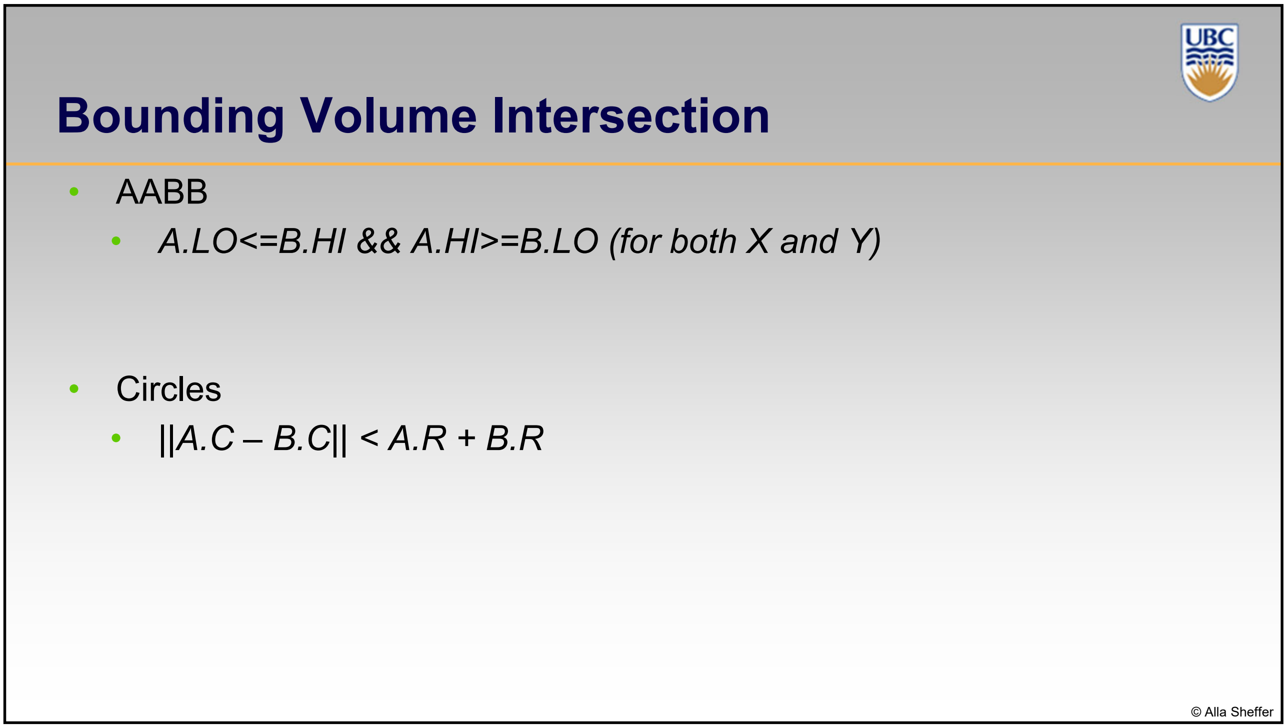
Bounding volumes

- AABB: Axis aligned bounding box
  - *+ Trivial to compute*
  - *+ Quick to evaluate*
  - *- May be too big…*
- Tight bounding box
  - *- Harder to compute (PCA)*
  - *- Slightly slower to evaluate*
  - *- Compact*



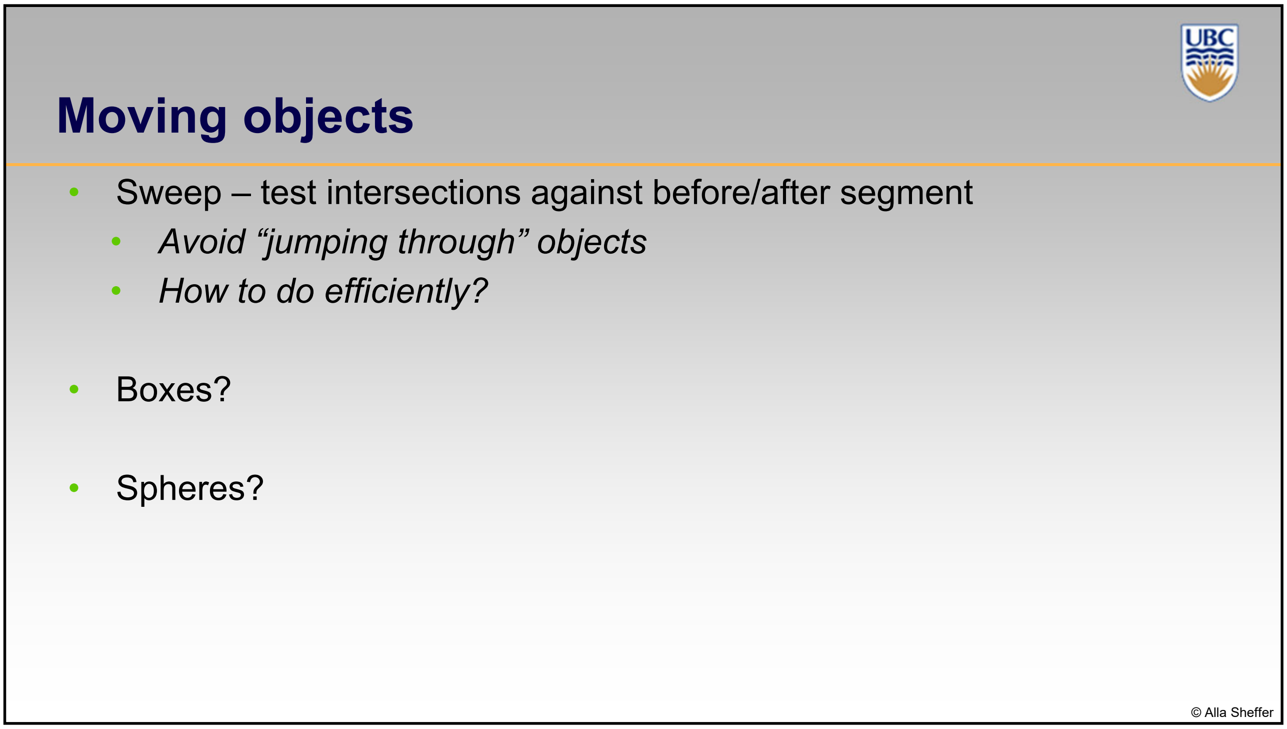
Bounding volumes

- Bounding circle
  - *A range of efficient (non-trivial) methods*
- Convex hull
  - *Gift wrapping & other methods…*



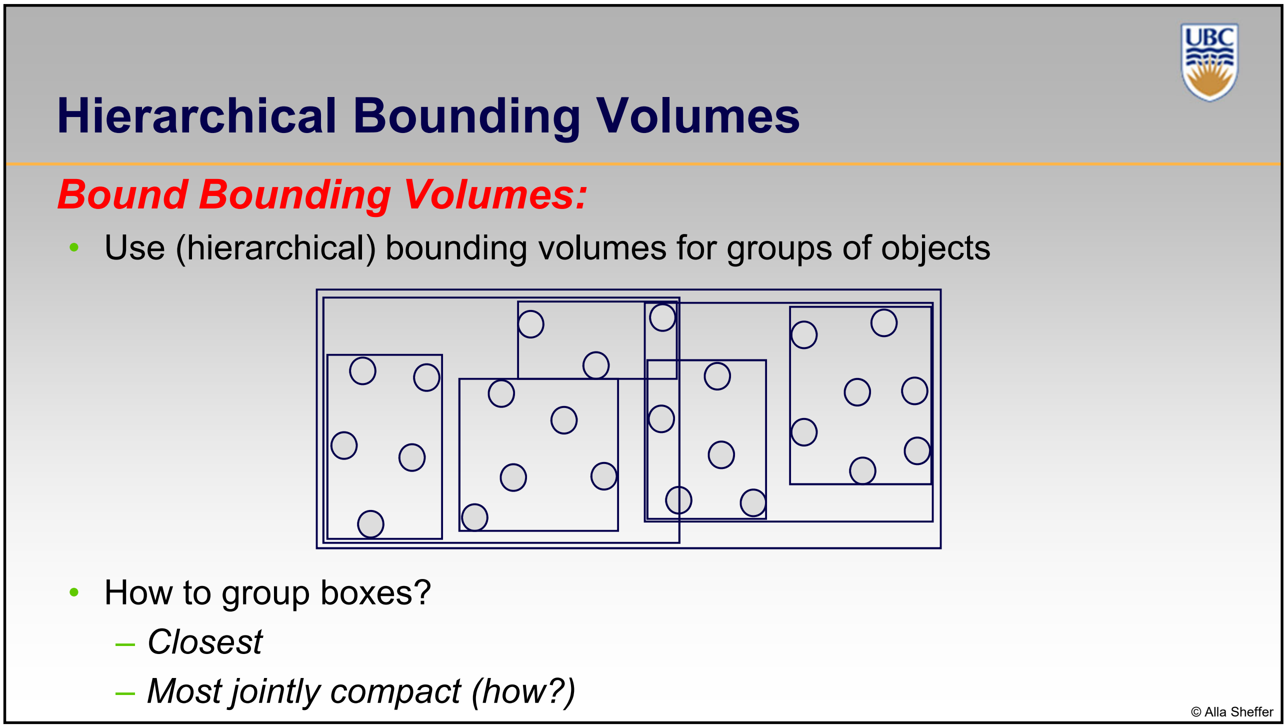
Bounding Volume Intersection

- AABB
  - *A.LO<=B.HI && A.HI>=B.LO (for both X and Y)*
- Circles
  - $||A.C - B.C|| < A.R + B.R$



Moving objects

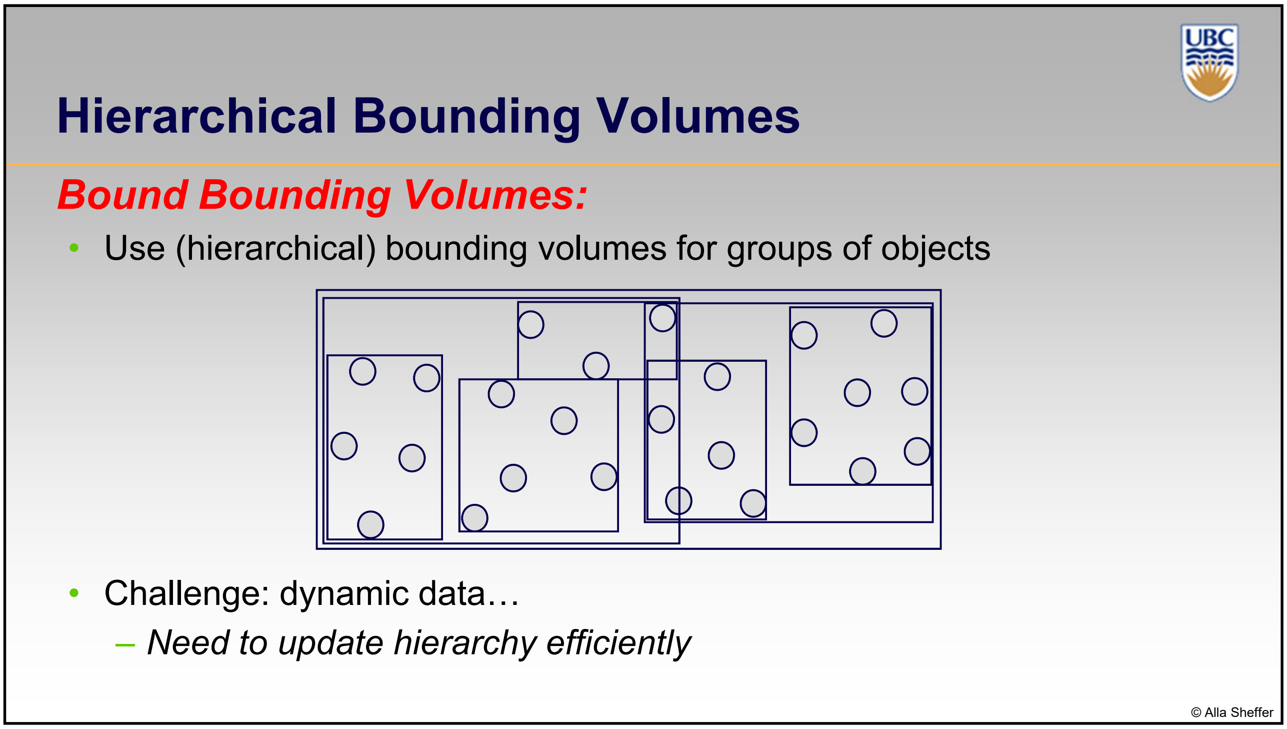
- Sweep – test intersections against before/after segment
  - *Avoid “jumping through” objects*
  - *How to do efficiently?*
- Boxes?
- Spheres?



Hierarchical Bounding Volumes

Bound Bounding Volumes:

- Use (hierarchical) bounding volumes for groups of objects
- How to group boxes?
  - *Closest*
  - *Most jointly compact (how?)*



Hierarchical Bounding Volumes

Bound Bounding Volumes:

- Use (hierarchical) bounding volumes for groups of objects
- Challenge: dynamic data…
  - *Need to update hierarchy efficiently*

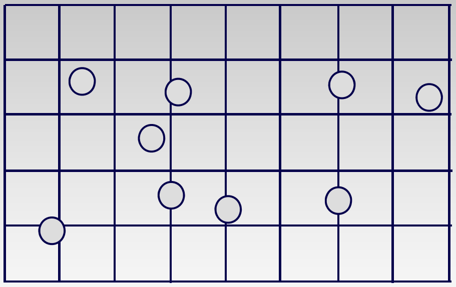
Spatial Subdivision DATA STRUCTURES

- Subdivide space (bounding box of the “world”)
- Hierarchical
  - *Subdivide each sub-space (or only non-empty sub-spaces)*
- Lots of methods
  - *Grid, Octree, k-D tree, (BSP tree)*

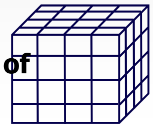
Regular Grid

Subdivide space into rectangular grid:

- Associate every object with the cell(s) that it overlaps with
- Test collisions only if cells overlap



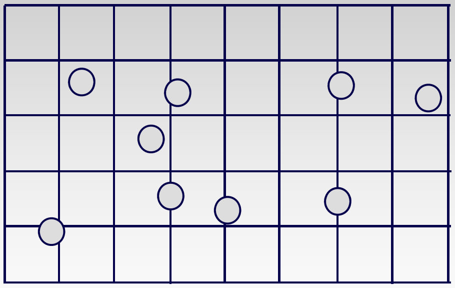
In 3D: regular grid of cubes (voxels):



Creating a Regular Grid

Steps:

- Find bounding box of scene
- Choose grid resolution in x, y, z
- Insert objects
- Objects that overlap multiple cells get referenced by all cells they overlap



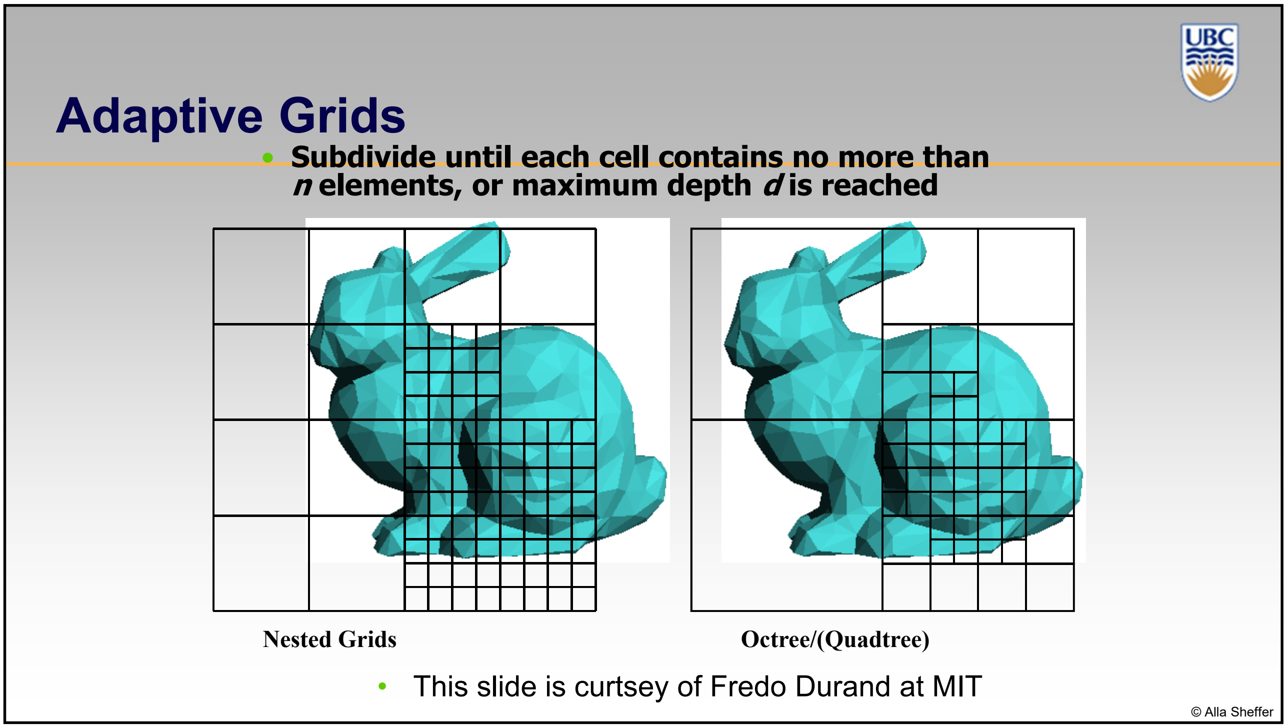
Regular Grid Discussion

Advantages?

- Easy to construct
- Easy to traverse

Disadvantages?

- May be only sparsely filled
- Geometry may still be clumped



Adaptive Grids

- Subdivide until each cell contains no more than *n* elements, or maximum depth *d* is reached

Nested Grids

Octree/(Quadtree)

- This slide is curtsey of Fredo Durand at MIT