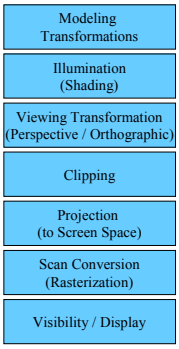
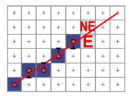
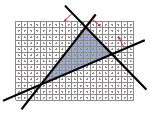
Acceleration Data Structures for Ray Tracing

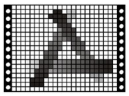
Last Time:



- Graphics Pipeline!!
- Clipping
- Line & Polygon Rasterization
  - Bresenham (DDA)
- Visibility
  - Depth Buffer (z buffer)







Schedule

- Wed Oct 13th Assignment 4 due (Shadows, Reflection, & Refraction)
- Wed Oct 20th Assignment 5 due (Voxel Rendering)

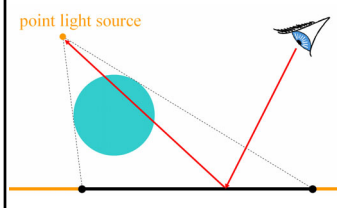
- Review Session for Quiz 1
  - Monday 25th, 7:30 – 9pm, room TBA
- Tuesday October 26th, in class: Quiz 1

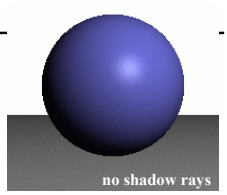
Today

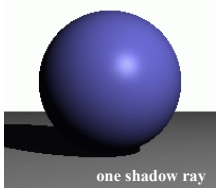
- Motivation – Distribution Ray Tracing
  - Soft shadows
  - Antialiasing (getting rid of jaggies)
  - Glossy reflection
  - Motion blur
  - Depth of field (focus)
- Bounding Boxes
- Spatial Acceleration Data Structures
- Flattening the Transformation Hierarchy

Shadows

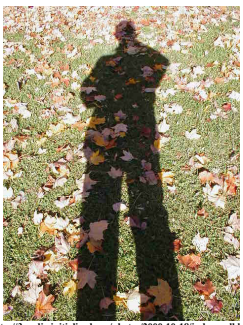
- one shadow ray per intersection per point light source





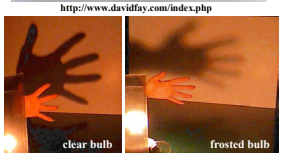


Shadows & Light Sources



http://3media.initialized.org/photos/2000-10-18/index_gall.htm

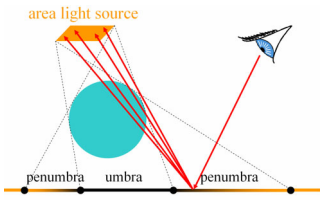
http://www.davidfay.com/index.php

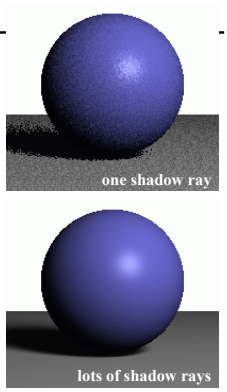


http://www.pa.uky.edu/~sciworks/light/preview/bulb2.htm

Soft Shadows

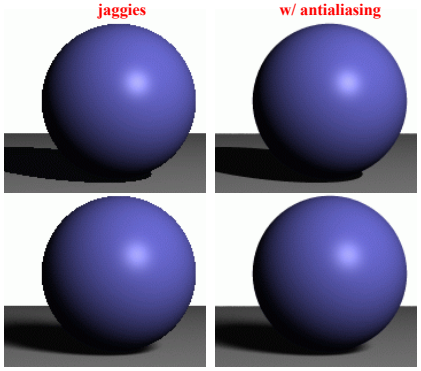
- multiple shadow rays to sample area light source





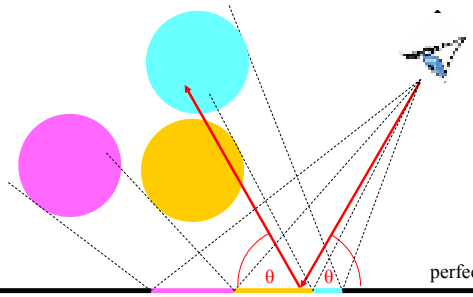
Antialiasing – Supersampling

- multiple rays per pixel



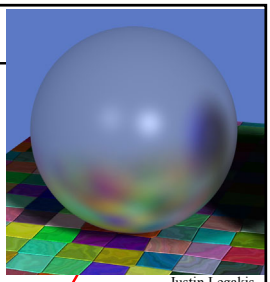
Reflection

- one reflection ray per intersection

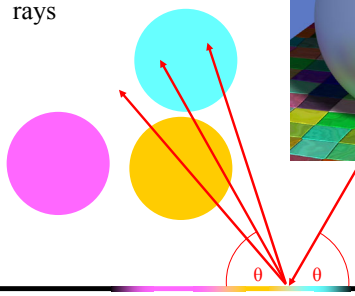


Glossy Reflection

- multiple reflection rays

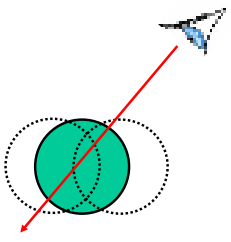


Justin Legakis



Motion Blur

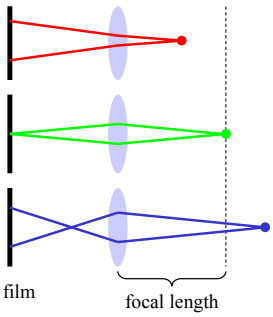
- Sample objects temporally

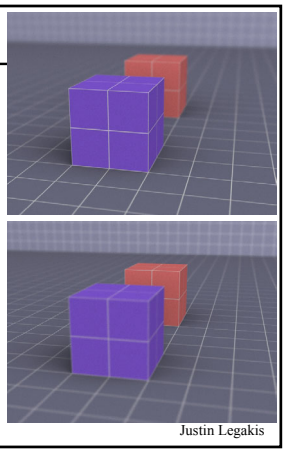


Rob Cook

Depth of Field

- multiple rays per pixel





Justin Legakis

Ray Tracing Algorithm Analysis

- Ray casting
- Lots of primitives
- Recursive
- Distributed Ray Tracing Effects
  - Soft shadows
  - Anti-aliasing
  - Glossy reflection
  - Motion blur
  - Depth of field

cost ≈ height * width *
num primitives *
intersection cost *
size of recursive ray tree *
num shadow rays *
num supersamples *
num glossy rays *
num temporal samples *
num focal samples *
. . .

can we reduce this?

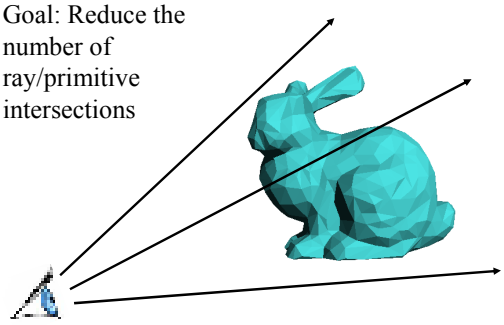
Questions?

Today

- Motivation – Distribution Ray Tracing
- Bounding Boxes
  - of each primitive
  - of groups
  - of transformed primitives
- Spatial Acceleration Data Structures
- Flattening the Transformation Hierarchy

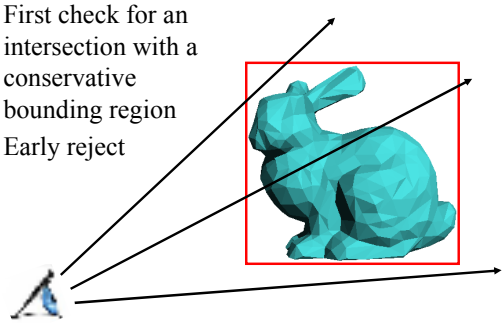
Acceleration of Ray Casting

- Goal: Reduce the number of ray/primitive intersections



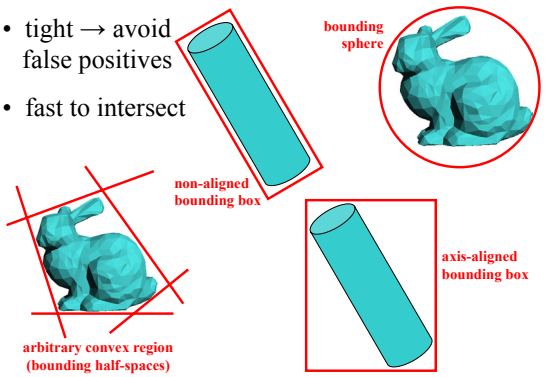
Conservative Bounding Region

- First check for an intersection with a conservative bounding region
- Early reject



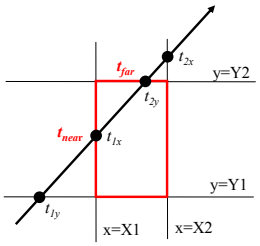
Conservative Bounding Regions

- tight → avoid false positives
- fast to intersect



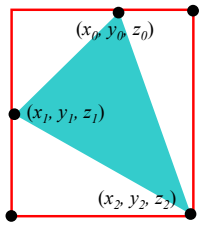
Intersection with Axis-Aligned Box

From Lecture 2,
Ray Casting II



- For all 3 axes, calculate the intersection distances t_1 and t_2
- $t_{near} = \max(t_{1x}, t_{1y}, t_{1z})$
  $t_{far} = \min(t_{2x}, t_{2y}, t_{2z})$
- If $t_{near} > t_{far}$, box is missed
- If $t_{far} < t_{min}$, box is behind
- If box survived tests, report intersection at t_{near}

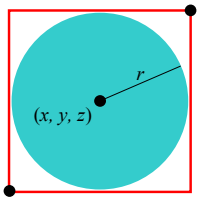
Bounding Box of a Triangle



$(x_{max}, y_{max}, z_{max})$
$= (\max(x_0,x_1,x_2),$
$\max(y_0,y_1,y_2),$
$\max(z_0,z_1,z_2))$

$(x_{min}, y_{min}, z_{min})$
$= (\min(x_0,x_1,x_2),$
$\min(y_0,y_1,y_2),$
$\min(z_0,z_1,z_2))$

Bounding Box of a Sphere

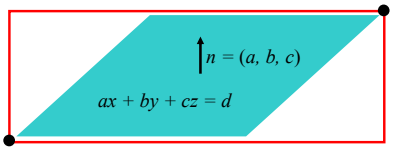


$(x_{max}, y_{max}, z_{max})$
$= (x+r,\ y+r,\ z+r)$

$(x_{min}, y_{min}, z_{min})$
$= (x-r,\ y-r,\ z-r)$

Bounding Box of a Plane

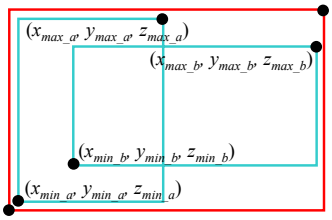
$(x_{max}, y_{max}, z_{max})$
$= (+\infty, +\infty, +\infty)^*$



$(x_{min}, y_{min}, z_{min})$
$= (-\infty, -\infty, -\infty)^*$

* *unless n is exactly perpendicular to an axis*

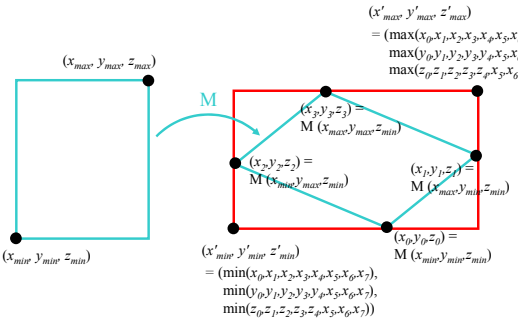
Bounding Box of a Group



$(x_{max}, y_{max}, z_{max})$
$= (\max(x_{max\_a},x_{max\_b}),$
$\max(y_{max\_a},y_{max\_b}),$
$\max(z_{max\_a},z_{max\_b}))$

$(x_{min}, y_{min}, z_{min})$ $= (\min(x_{min\_a},x_{min\_b}),$
$\min(y_{min\_a},y_{min\_b}),$
$\min(z_{min\_a},z_{min\_b}))$

Bounding Box of a Transform

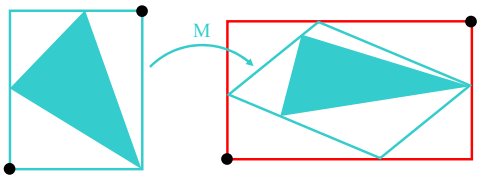


$(x'_{max}, y'_{max}, z'_{max})$
$= (\max(x_0,x_1,x_2,x_3,x_4,x_5,x_6,x_7),$
$\max(y_0,y_1,y_2,y_3,y_4,y_5,x_6,x_7),$
$\max(z_0,z_1,z_2,z_3,z_4,x_5,x_6,x_7))$

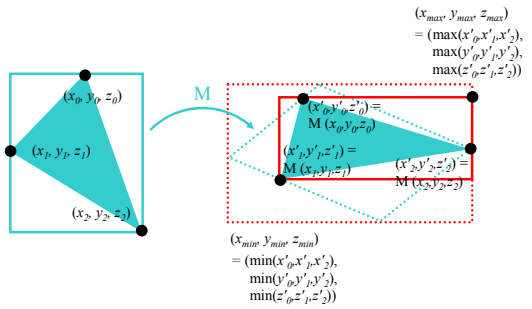
$(x'_{min}, y'_{min}, z'_{min})$
$= (\min(x_0,x_1,x_2,x_3,x_4,x_5,x_6,x_7),$
$\min(y_0,y_1,y_2,y_3,y_4,x_5,x_6,x_7),$
$\min(z_0,z_1,z_2,z_3,z_4,x_5,x_6,x_7))$

Special Case: Transformed Triangle

Can we do better?



Special Case: Transformed Triangle

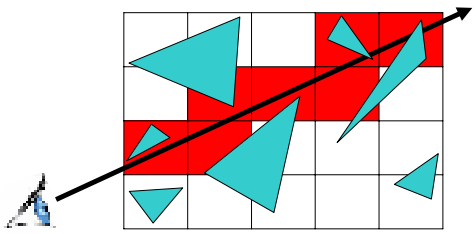


Questions?

Today

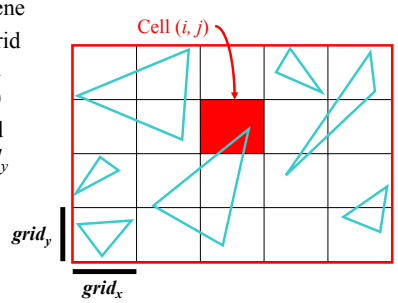
- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
  - Regular Grid
  - Adaptive Grids
  - Hierarchical Bounding Volumes
- Flattening the Transformation Hierarchy

Regular Grid



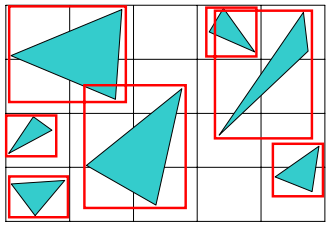
Create Grid

- Find bounding box of scene
- Choose grid resolution (n_x, n_y, n_z)
- $grid_x$ need not = $grid_y$



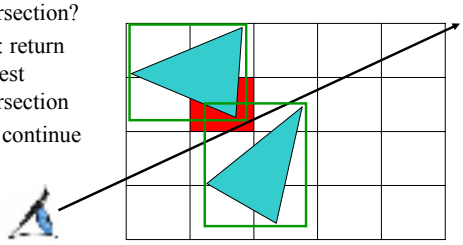
Insert Primitives into Grid

- Primitives that overlap multiple cells?
- Insert into multiple cells (use pointers)



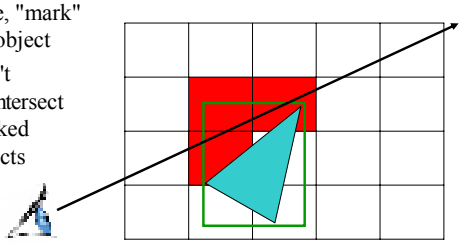
For Each Cell Along a Ray

- Does the cell contain an intersection?
- Yes: return closest intersection
- No: continue



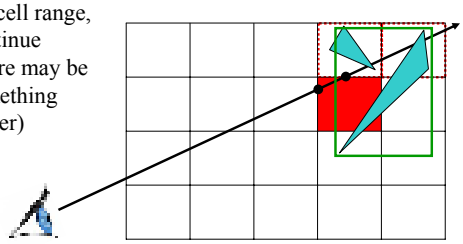
Preventing Repeated Computation

- Perform the computation once, "mark" the object
- Don't re intersect marked objects



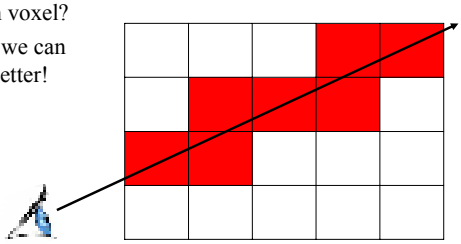
Don't Return Distant Intersections

- If intersection t is not within the cell range, continue (there may be something closer)



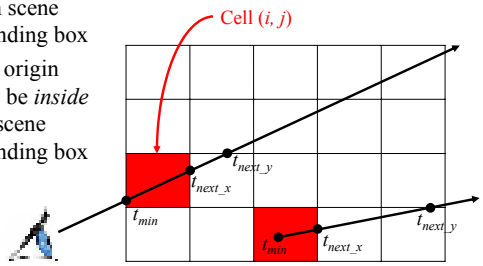
Which Cells Should We Examine?

- Should we intersect the ray with each voxel?
- No! we can do better!



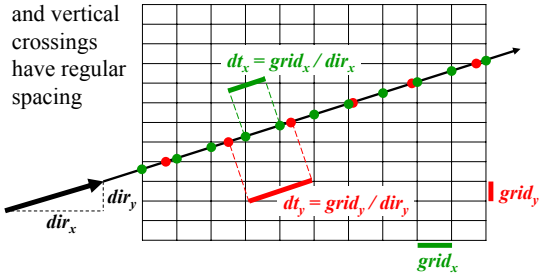
Where Do We Start?

- Intersect ray with scene bounding box
- Ray origin may be *inside* the scene bounding box



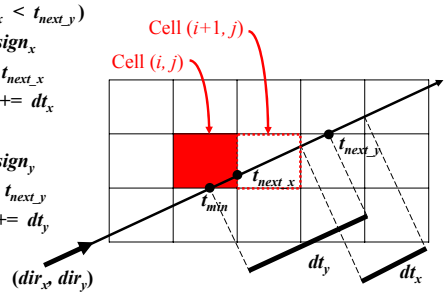
Is there a Pattern to Cell Crossings?

- Yes, the horizontal and vertical crossings have regular spacing



What's the Next Cell?

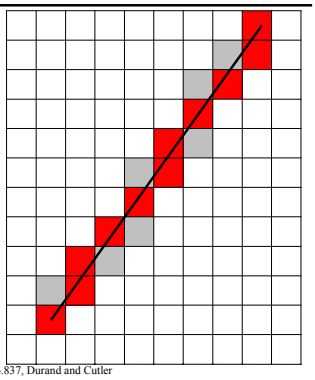
if ($t_{next\_x} < t_{next\_y}$)
  i += $sign_x$
  $t_{min} = t_{next\_x}$
  t_{next_x} += dt_x
else
  j += $sign_y$
  $t_{min} = t_{next\_y}$
  t_{next_y} += dt_y



if ($dir_x > 0$) $sign_x = 1$ *else* $sign_x = -1$
if ($dir_y > 0$) $sign_y = 1$ *else* $sign_y = -1$

What's the Next Cell?

- 3DDDA – Three Dimensional Digital Difference Analyzer
- Similar to Bresenham's Line Rasterization!



Pseudo-Code

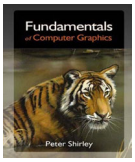
```
create grid
insert primitives into grid
for each ray r
  find initial cell c(i,j), t_min, t_next_x & t_next_y
  compute dt_x, dt_y, sign_x and sign_y
  while c != NULL
    for each primitive p in c
      intersect r with p
      if intersection in range found
        return
    c = find next cell
```


Regular Grid Discussion

- Advantages?
  - easy to construct
  - easy to traverse
- Disadvantages?
  - may be only sparsely filled
  - geometry may still be clumped

A Note about Typos

- Typos happen in lecture notes
  - Don't be afraid of thinking and asking questions
  - Please tell us about any typos you find & we'll fix them ASAP
- Typos happen in textbooks
  - The pseudocode for the 3DDDA ray/grid marching in Shirley is buggy
  - Think, don't just copy directly



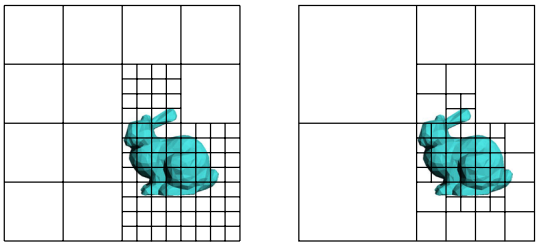
Questions?

Today

- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
  - Regular Grid
  - Adaptive Grids
  - Hierarchical Bounding Volumes
- Flattening the Transformation Hierarchy

Adaptive Grids

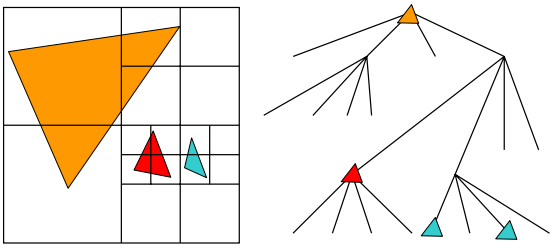
- Subdivide until each cell contains no more than *n* elements, or maximum depth *d* is reached



Nested Grids Octree/(Quadtree)

Primitives in an Adaptive Grid

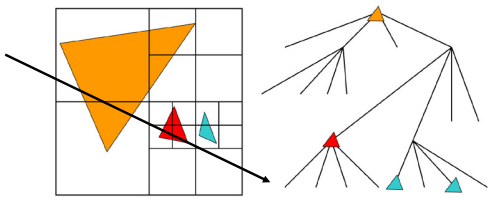
- Can live at intermediate levels, or be pushed to lowest level of grid



Octree/(Quadtree)

Adaptive Grid Discussion

- Advantages?
  - grid complexity matches geometric density
- Disadvantages?
  - more expensive to traverse (especially octree)

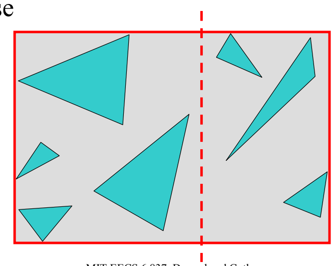


Today

- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
  - Regular Grid
  - Adaptive Grids
  - Hierarchical Bounding Volumes
- Flattening the Transformation Hierarchy

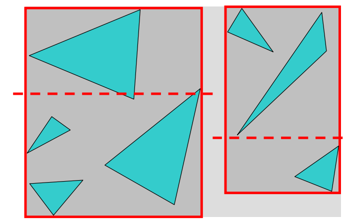
Bounding Volume Hierarchy

- Find bounding box of objects
- Split objects into two groups
- Recurse



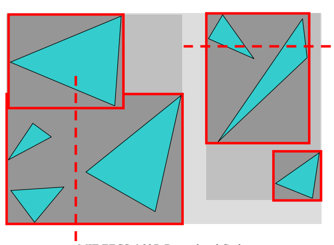
Bounding Volume Hierarchy

- Find bounding box of objects
- Split objects into two groups
- Recurse



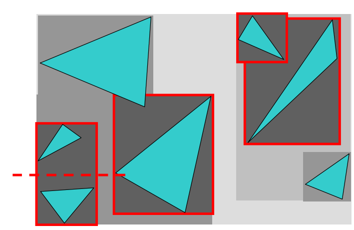
Bounding Volume Hierarchy

- Find bounding box of objects
- Split objects into two groups
- Recurse



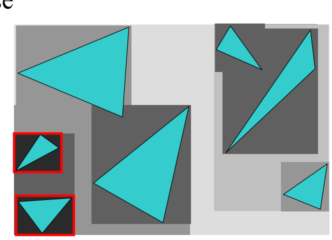
Bounding Volume Hierarchy

- Find bounding box of objects
- Split objects into two groups
- Recurse



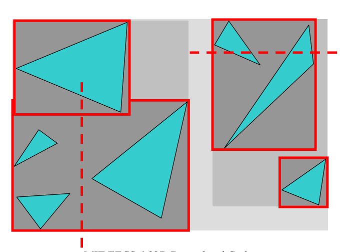
Bounding Volume Hierarchy

- Find bounding box of objects
- Split objects into two groups
- Recurse



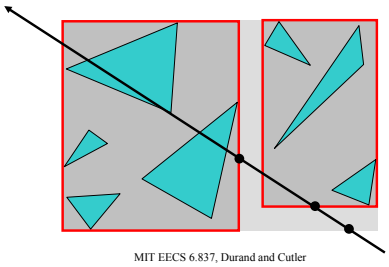
Where to split objects?

- At midpoint *OR*
- Sort, and put half of the objects on each side *OR*
- Use modeling hierarchy



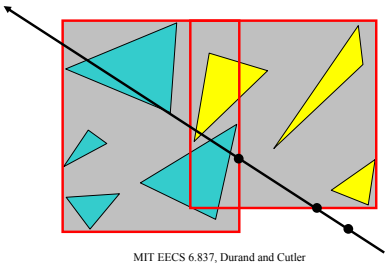
Intersection with BVH

- Check sub-volume with closer intersection first



Intersection with BVH

- Don't return intersection immediately if the other subvolume may have a closer intersection



Bounding Volume Hierarchy Discussion

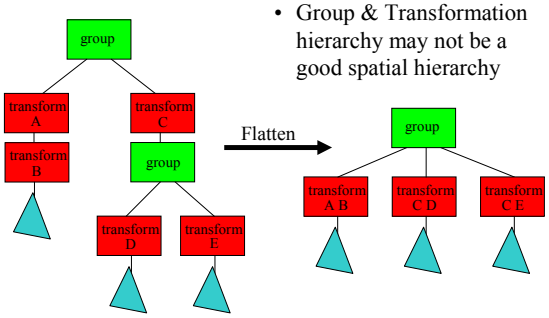
- Advantages
  - easy to construct
  - easy to traverse
  - binary
- Disadvantages
  - may be difficult to choose a good split for a node
  - poor split may result in minimal spatial pruning

Today

- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
- Flattening the Transformation Hierarchy

Transformation Hierarchy

- Group & Transformation hierarchy may not be a good spatial hierarchy

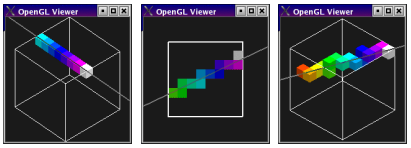


Assignments 5 & 6

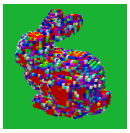
- Assignment 5: Voxel Rendering
  - Bounding boxes for primitives
  - Sphere voxelization
  - Regular grid data structure
  - Fast ray-grid intersection
  - Flatten the transformation hierarchy
- Assignment 6: Grid Acceleration & Solid Textures
  - Accelerated ray tracing (6)
  - Analyze ray tracing statistics (average # of rays, intersections, etc. per pixel)
  - Solid textures (next time)
  - Extra Credit: Distribution Ray Tracing

Ray Marching Visualization

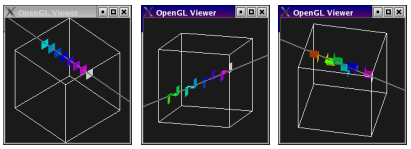
sphere voxelization



cells traversed



primitive density



entered faces

Next Time:

Texture Mapping