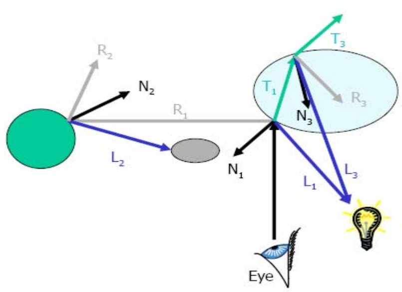
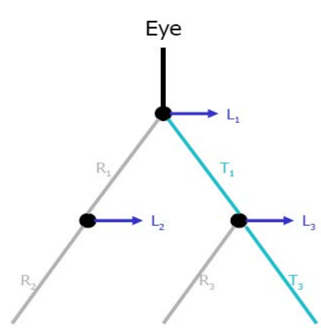
Advanced Ray Tracing

Thanks to Fredo Durand and Barb Cutler

The Ray Tree





Ni surface normal

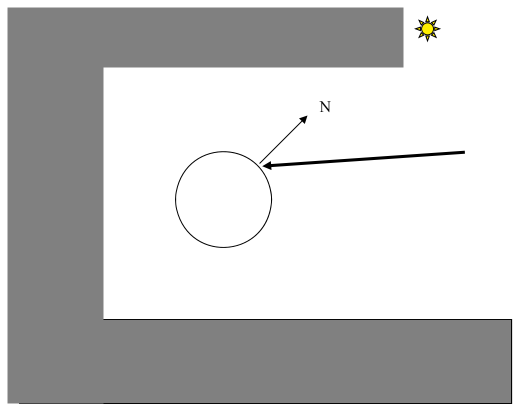
Ri reflected ray

Li shadow ray

Ti transmitted (refracted) ray

Ray Tree

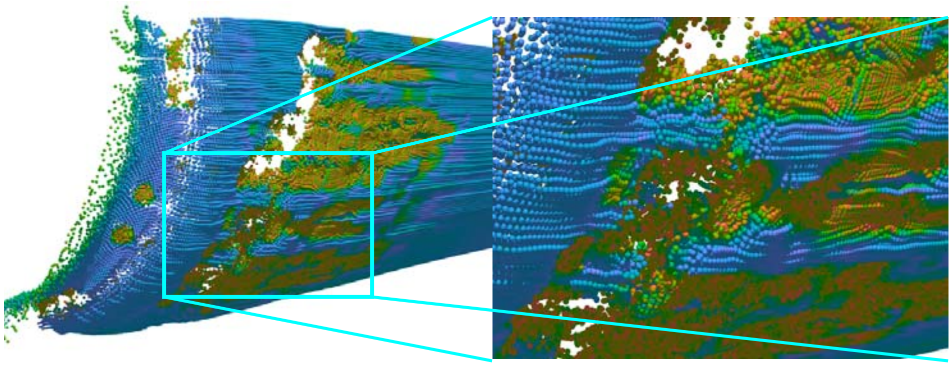
Draw the ray-tree to depth 3 for the following initial ray
(boxes are solid plastic, sphere is glass):



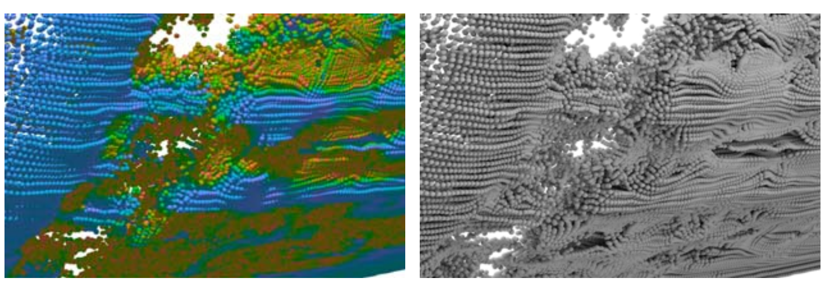
Basic Ray Tracing: Notes

- Intersection calculations are expensive, and even more so for more complex objects
  - Not currently suitable for real-time (i.e., games)
- Only global illumination effect is purely specular reflection/transmission
  - No "diffuse reflection" from other objects ! Still using ambient term
  - One remedy is **radiosity** (slow, offline, precompute
  - Ambient Occlusion
- Shadows have sharp edges, which is unrealistic – want to soften (coming up)

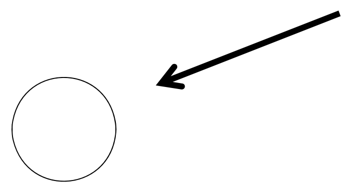
Phong shading



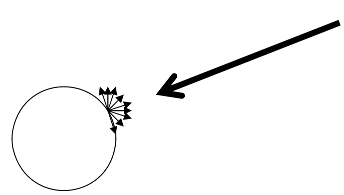
Ambient Occlusion



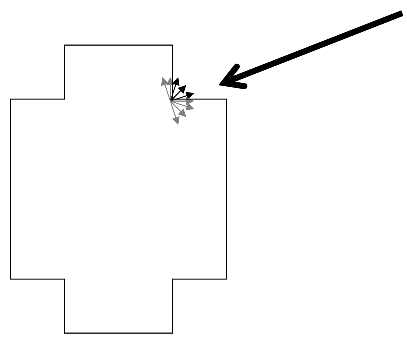
Ambient Occlusion



Ambient Occlusion



Ambient Occlusion

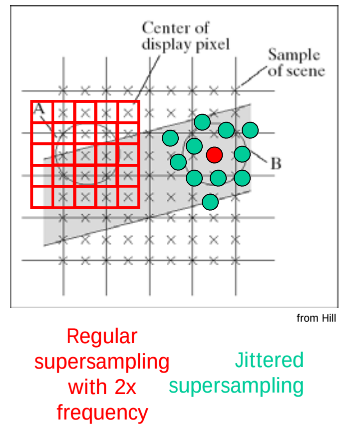


Ray Tracing: Improvements

- Image quality: Anti-aliasing
  - Supersampling: Shoot multiple rays per pixel (grid or jittered)
    - Adaptive: More rays in areas where image is changing more quickly
- Efficiency: Bounding extents
  - Idea: Enclose complex objects in shapes (e.g., sphere, box) that are less expensive to test for intersection
  - Comping up

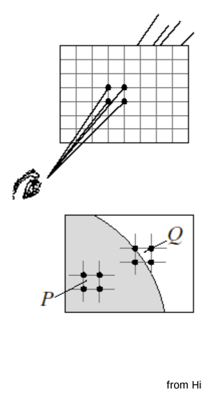
Supersampling

- Rasterize at higher resolution
  - Regular grid pattern around each "normal" image pixel
  - Irregular **jittered** sampling pattern reduces artifacts
- Combine multiple samples into one pixel via **weighted average**
  - "Box" filter: All samples associated with a pixel have equal weight (i.e., directly take their average)
  - Gaussian/cone filter: Sample weights inversely proportional to distance from associated pixel

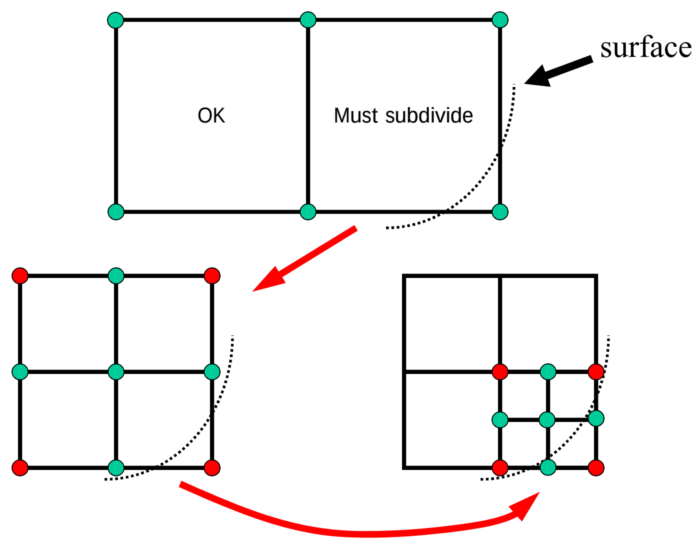


Adaptive Supersampling (Whitted's method)

- Shoot rays through 4 pixel corners and collect colors
- Provisional color for entire pixel is **average** of corner contributions
  - If you stop here, the only overhead vs. center-of-pixel ray-tracing is another row, column of rays
- If any corner's color is too different, **subdivide** pixel into quadrants and recurse on quadrants



Adaptive Supersampling: Details

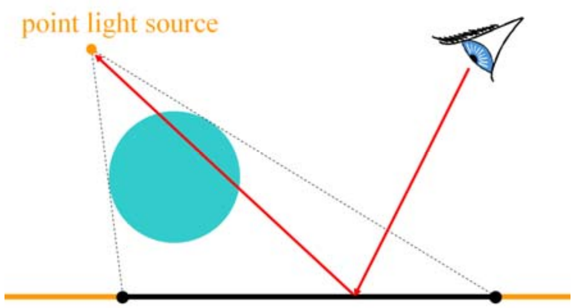


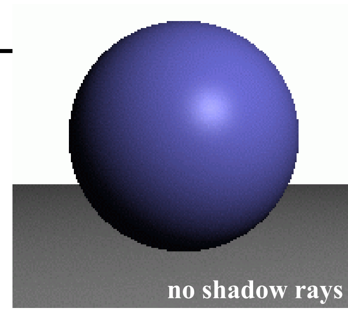
Ray-tracing Acceleration

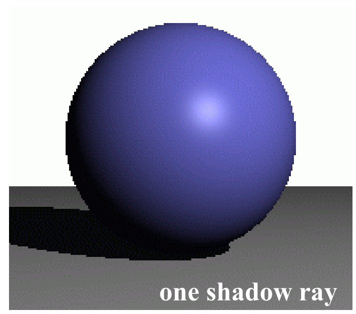
- Motivation – Distribution Ray Tracing
  - Soft shadows
  - Antialiasing (getting rid of jaggies)
  - Glossy reflection
  - Motion blur
  - Depth of field (focus)
- Bounding Boxes
- Spatial Acceleration Data Structures
- Flattening the Transformation Hierarchy

Shadows

- one shadow ray per intersection per point light source



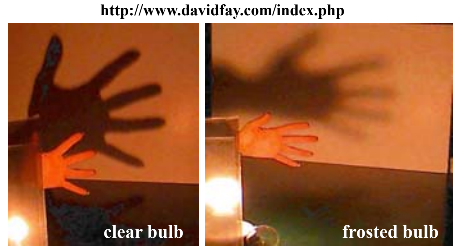




Shadows & Light Sources

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

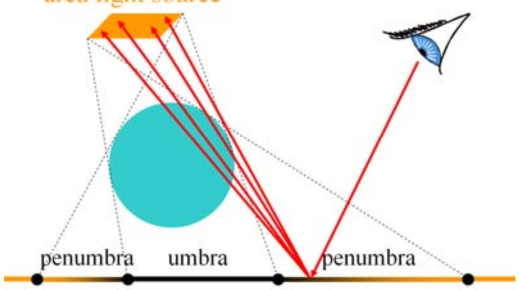
http://www.davidfay.com/index.php

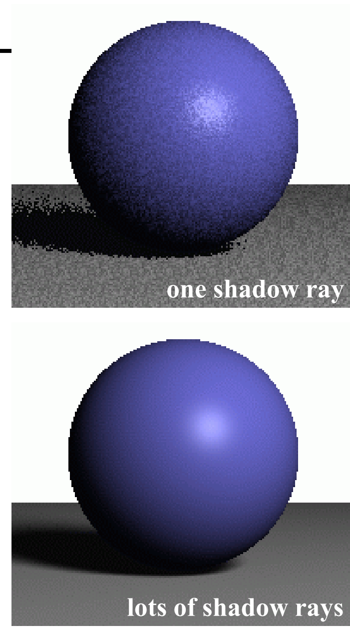


http://www.pa.uky.edu/~sciworks/light/preview/bulb2.htm

Soft Shadows

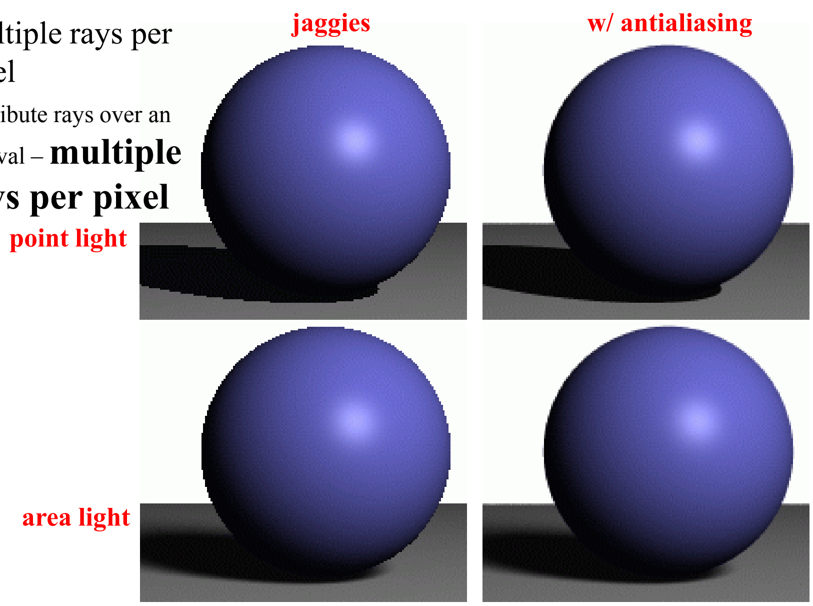
- Multiple shadow rays to sample area light source
- Distribute rays over an interval – **light source**





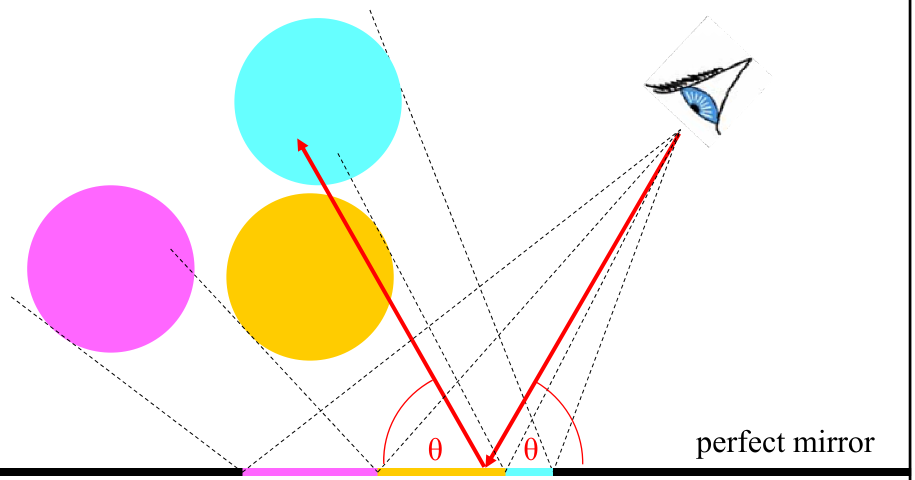
Antialiasing – Supersampling

- Multiple rays per pixel
- Distribute rays over an interval – **multiple rays per pixel**



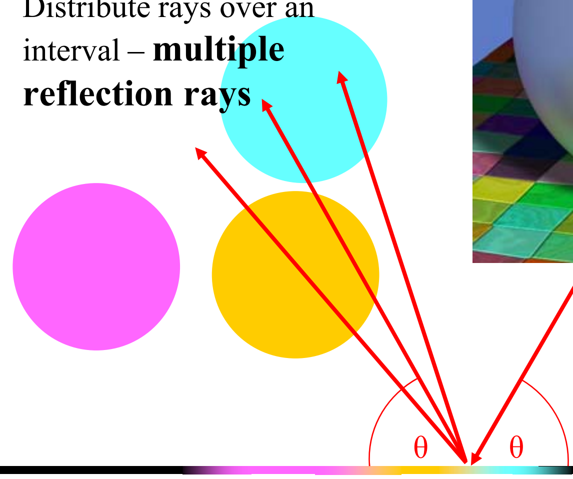
Reflection

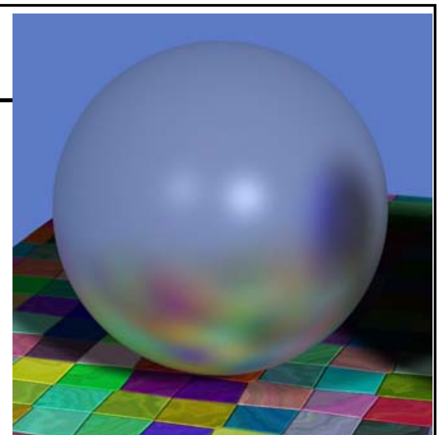
- one reflection ray per intersection



Glossy Reflection

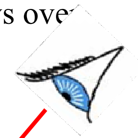
- multiple reflection rays
- Distribute rays over an interval – **multiple reflection rays**

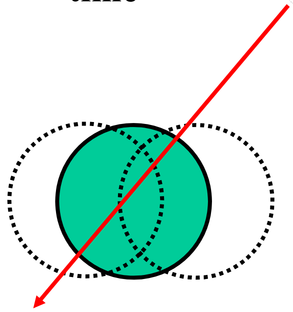



Justin Legakis

Motion Blur

- Sample objects temporally
- Distribute rays over an interval – **time**

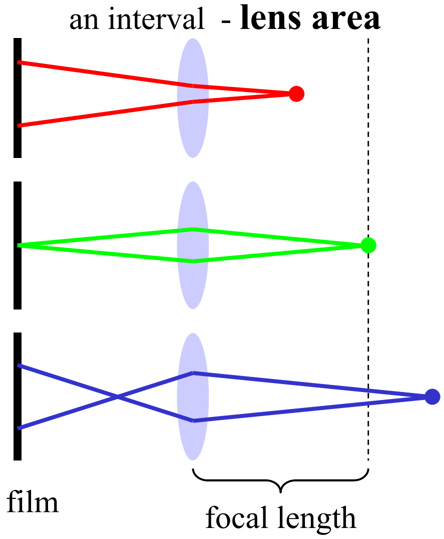


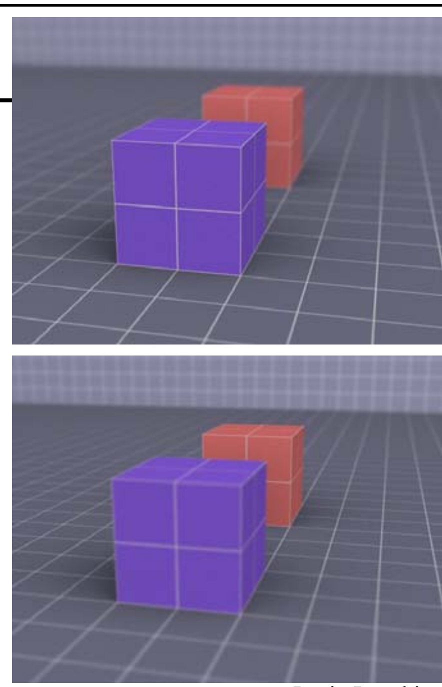


Rob Cook

Depth of Field

- Distribute rays over an interval - **lens area**





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?

Ray-Tracing Taxonomy

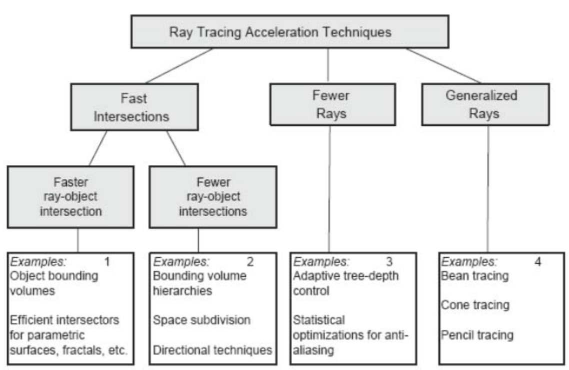


Figure 1: A broad classification of acceleration techniques.

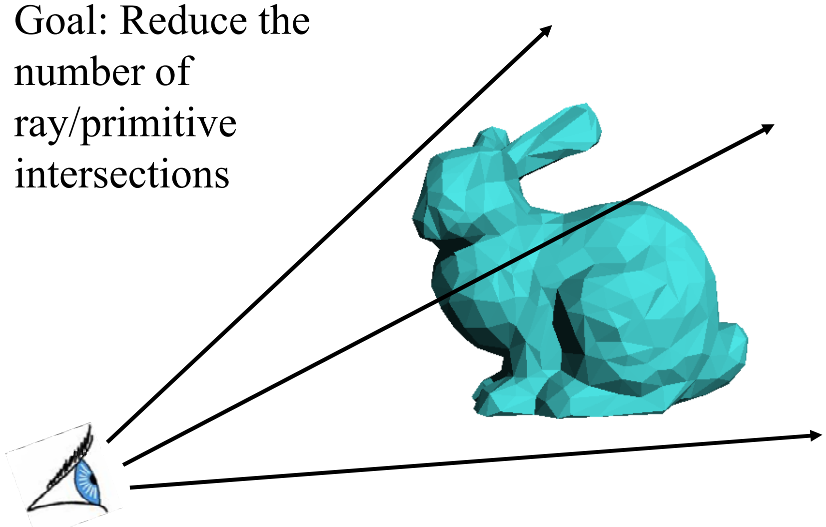
by James Arvo and David Kirk

Ray-tracing Acceleration

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

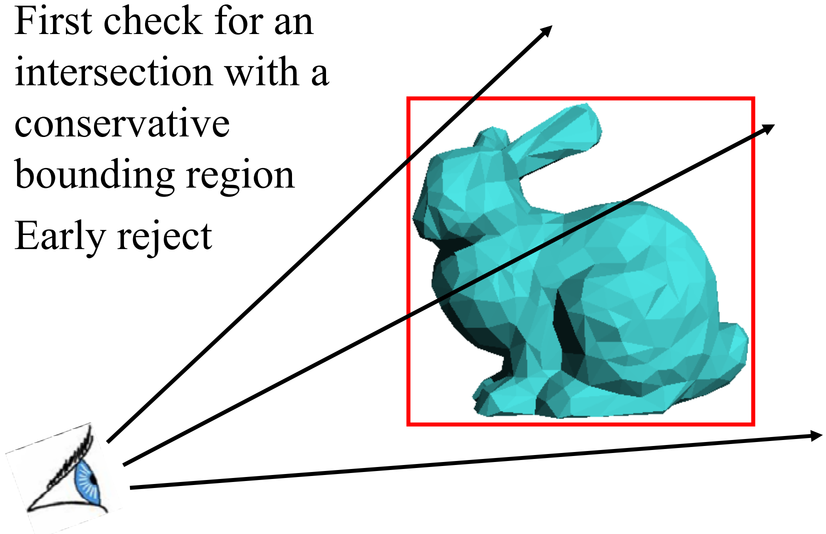
Acceleration of Ray Casting

- Goal: Reduce the number of ray/primitive intersections



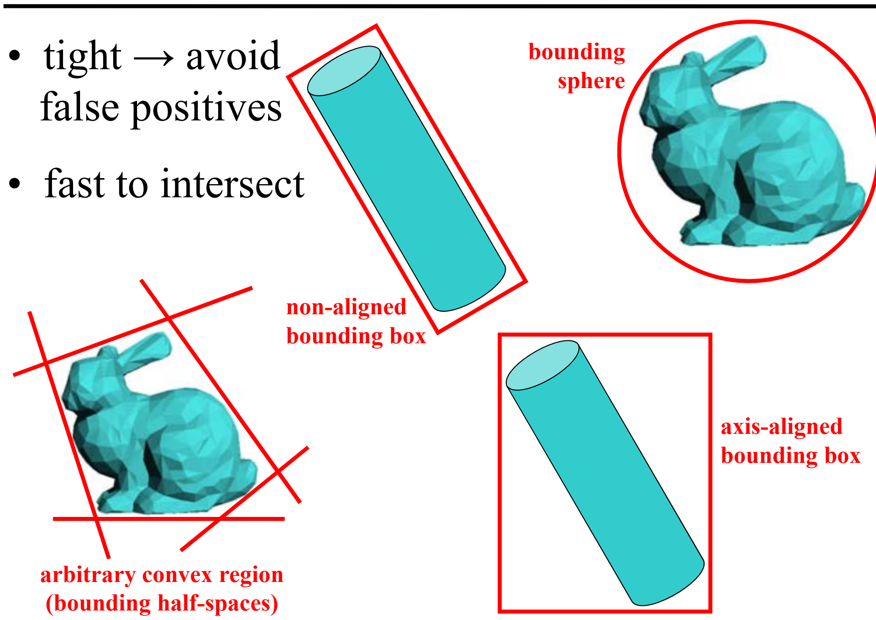
Conservative Bounding Region

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

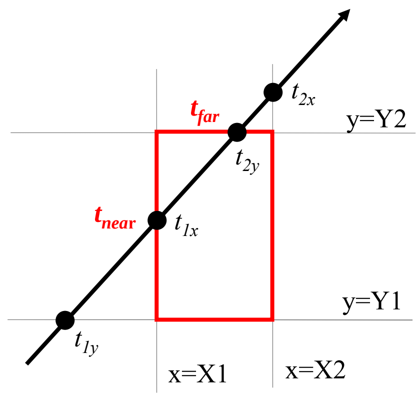


Conservative Bounding Regions

- tight → avoid false positives
- fast to intersect

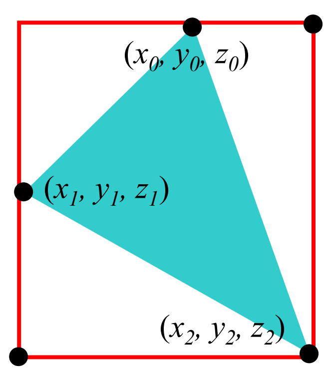


Intersection with Axis-Aligned Box



- 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}

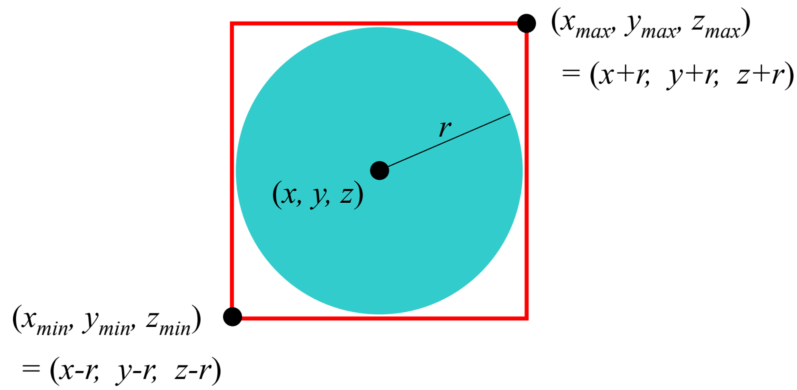
Bounding Box of a Triangle



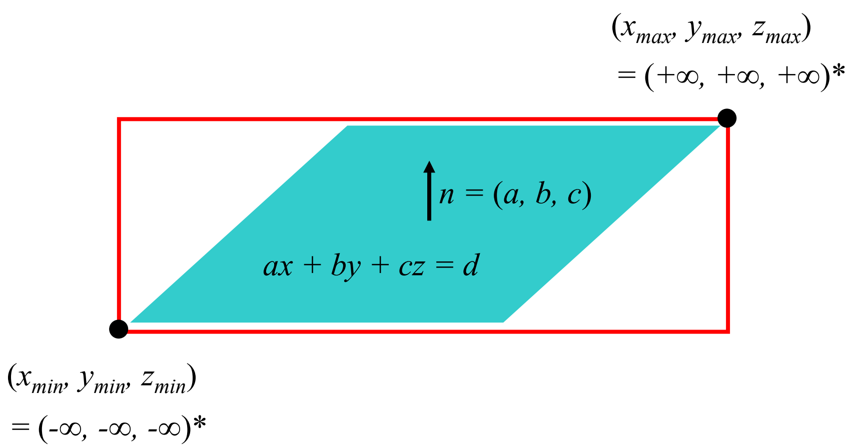
$(x_{max}, y_{max}, z_{max})$
$= (\max(x_0,x_1,x_2),$
$\max(y_0,y_1,y_2),$
$\max(z_0,z_1,z_2))$

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

Bounding Box of a Sphere

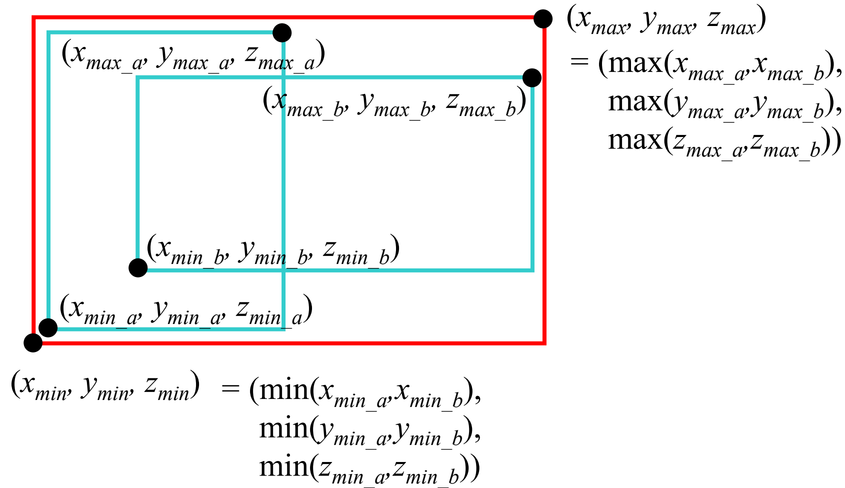


Bounding Box of a Plane

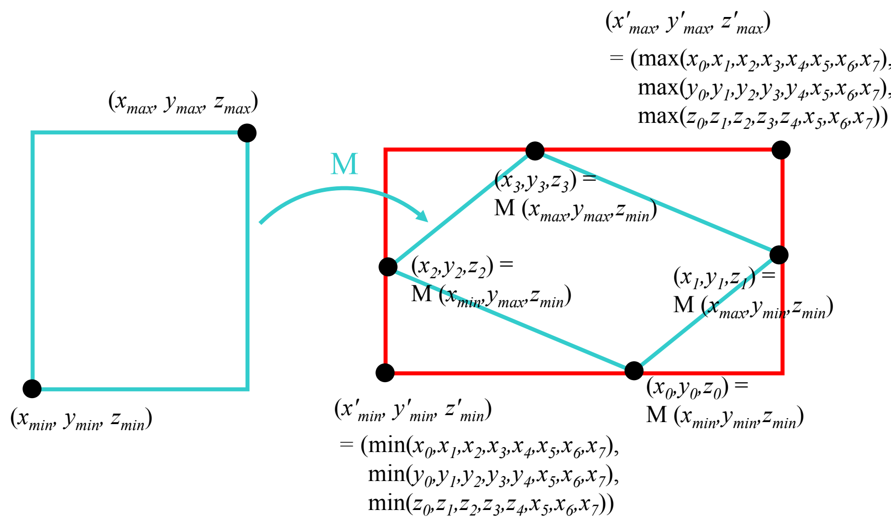


* *unless n is exactly perpendicular to an axis*

Bounding Box of a Group

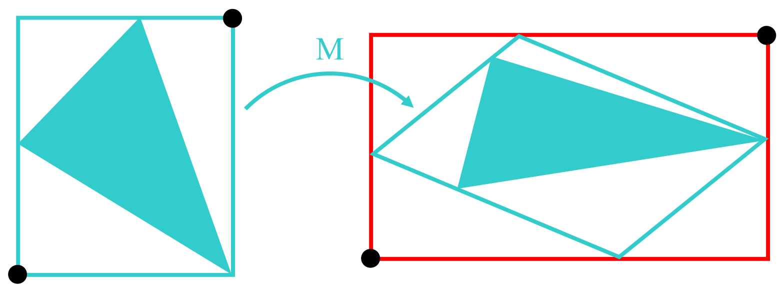


Bounding Box of a Transform

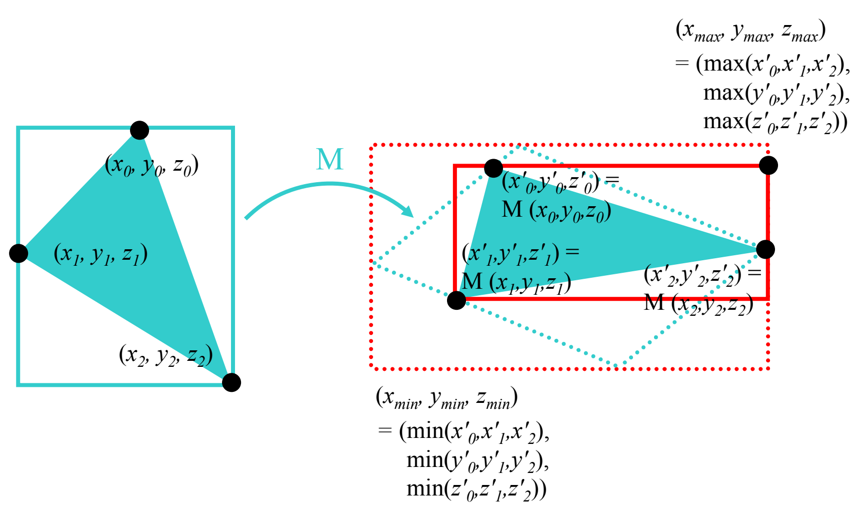


Special Case: Transformed Triangle

Can we do better?



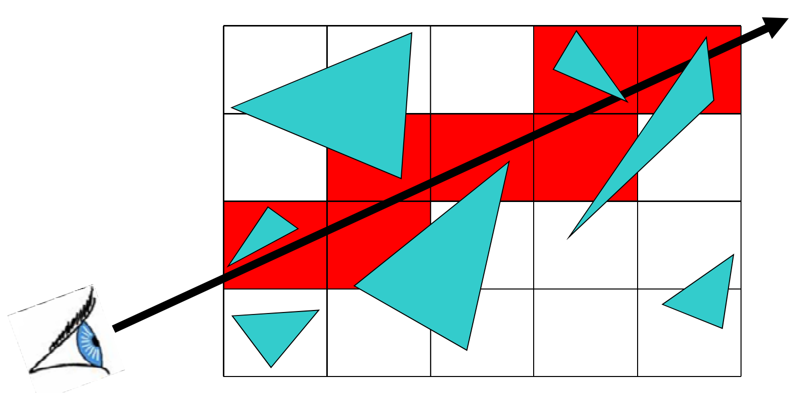
Special Case: Transformed Triangle



Ray-tracing Acceleration

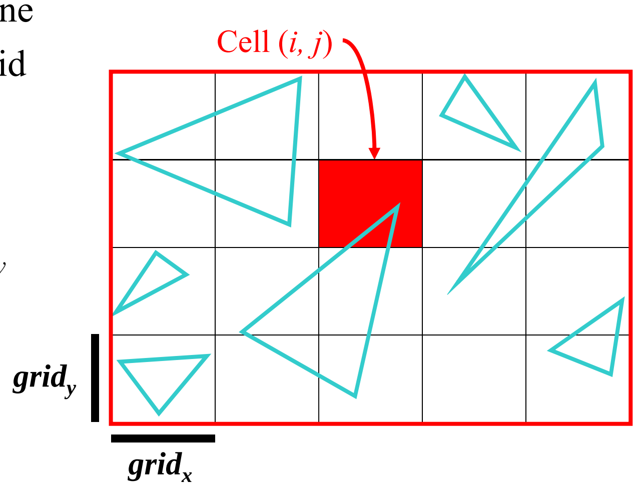
- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
  - Regular Grid
  - Adaptive Grids
  - Hierarchical Bounding Volumes
- Flattening the Transformation Hierarchy

Regular Grid



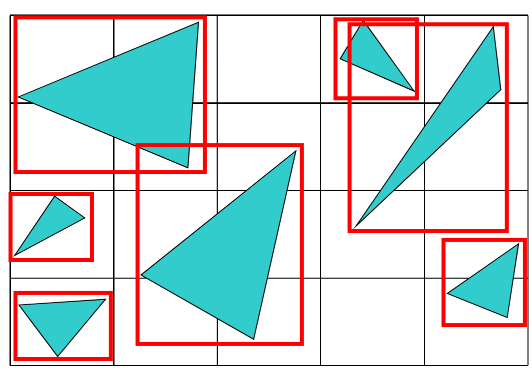
Create Grid

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



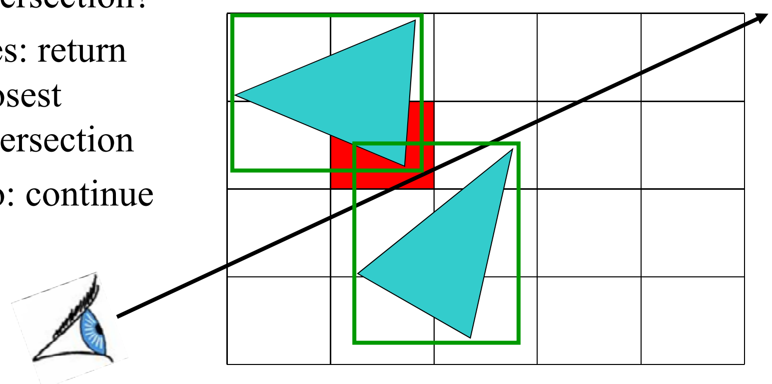
Insert Primitives into Grid

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



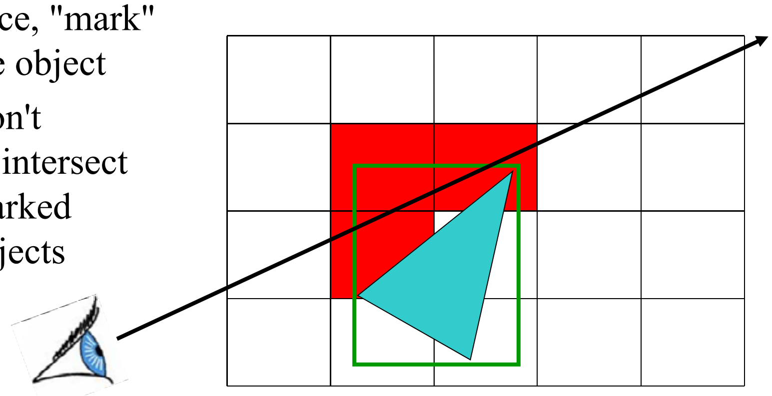
For Each Cell Along a Ray

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



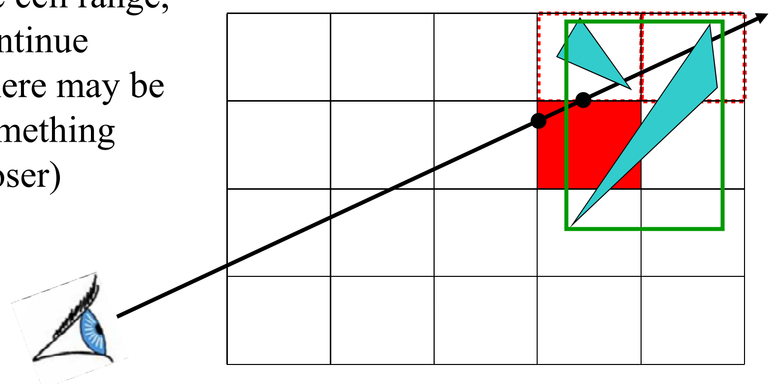
Preventing Repeated Computation

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



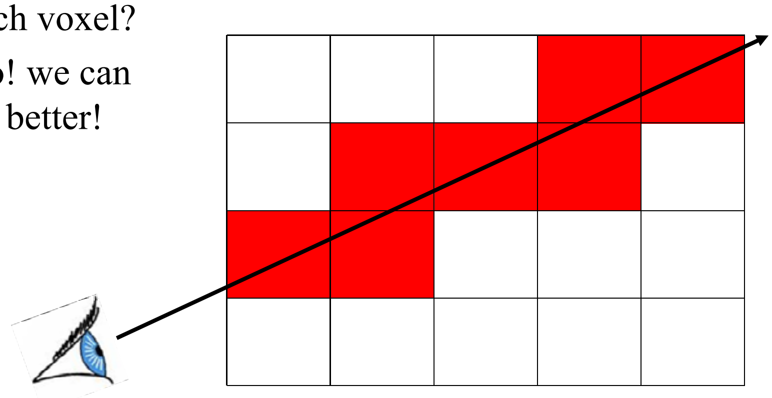
Don't Return Distant Intersections

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



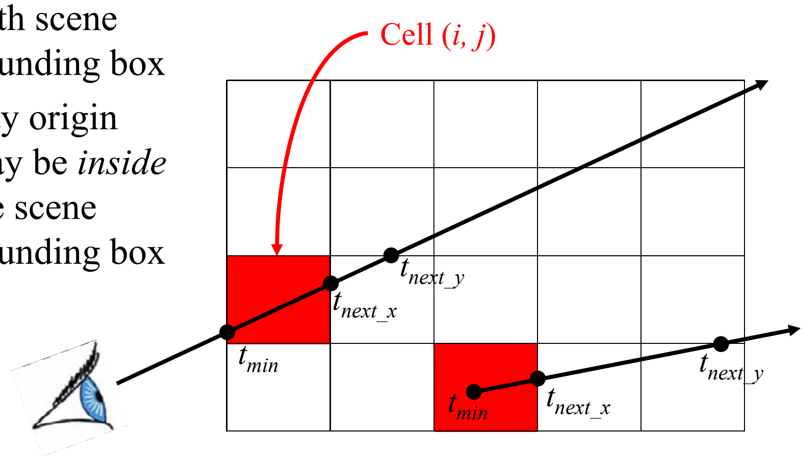
Which Cells Should We Examine?

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



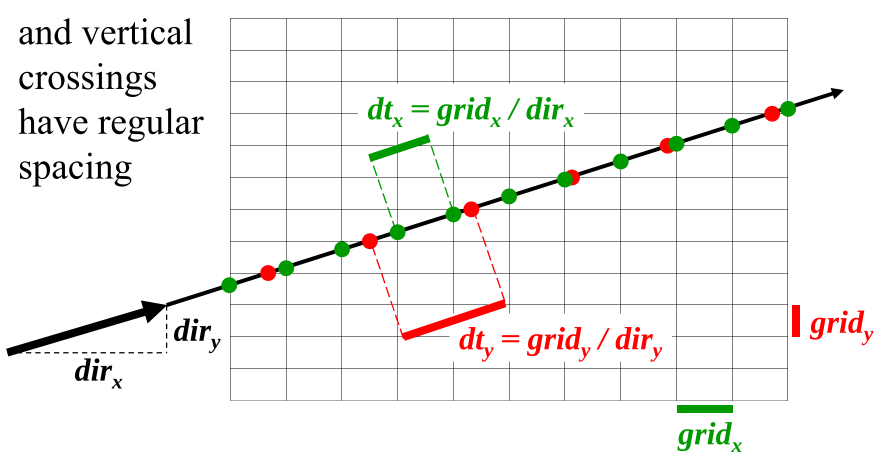
Where Do We Start?

- 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 += $sign_x$

  $t_{min} = t_{next\_x}$

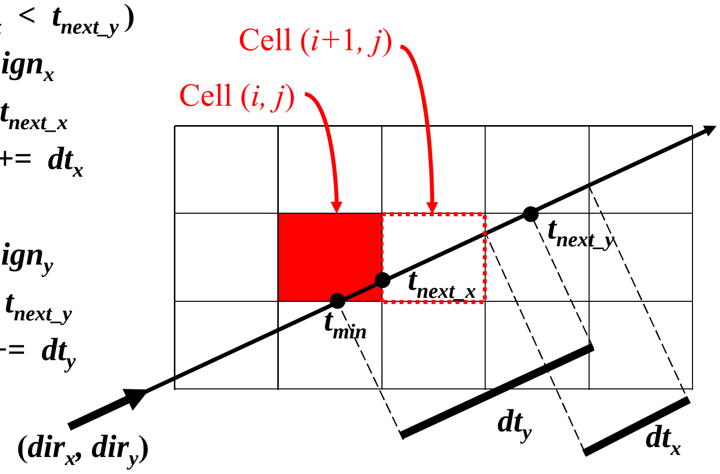
  t_{next_x} += dt_x

else

  j += $sign_y$

  $t_{min} = t_{next\_y}$

  t_{next_y} += dt_y

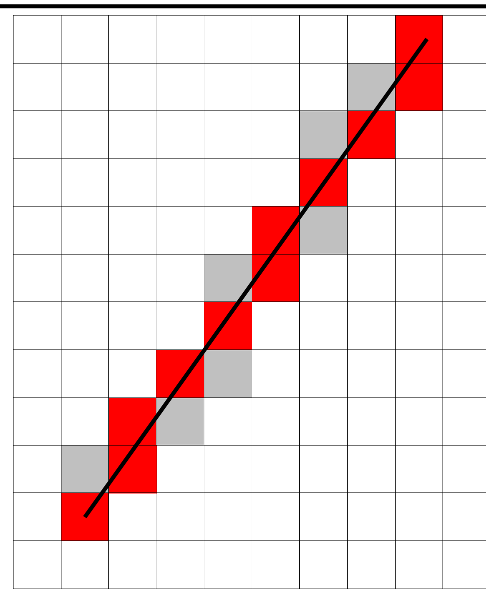


if ($dir_x > 0$) $sign_x = 1$ *else* $sign_x = -1$

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

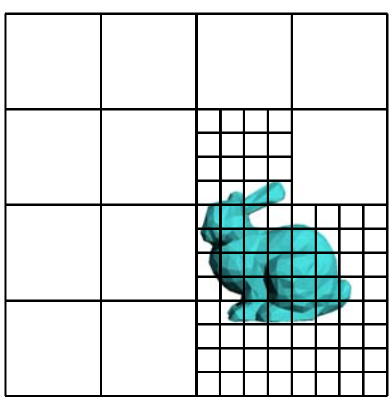
- Advantages?
  - easy to construct
  - easy to traverse
- Disadvantages?
  - may be only sparsely filled
  - geometry may still be clumped

Ray-tracing Acceleration

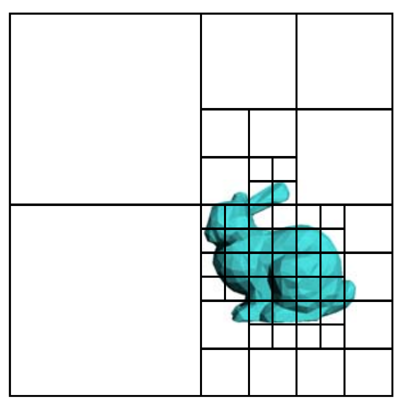
- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
  - Regular Grid
  - Adaptive Grids
  - Hierarchical Bounding Volumes
- Flattening the Transformation Hierarchy

Adaptive Grids

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



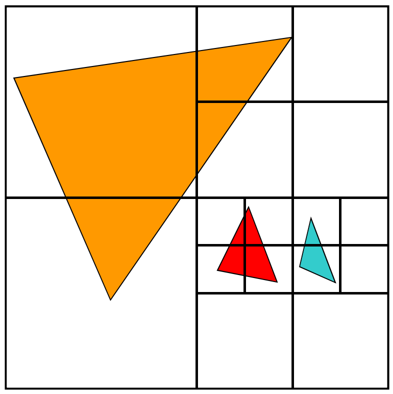
Nested Grids



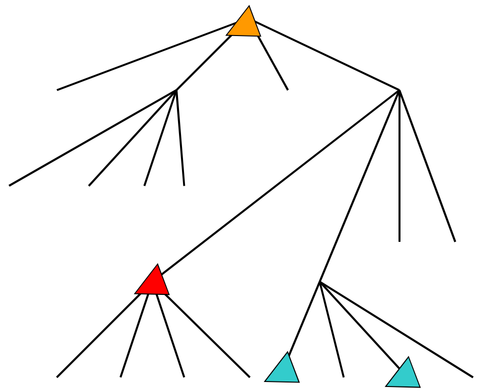
Octree/(Quadtree)

Primitives in an Adaptive Grid

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

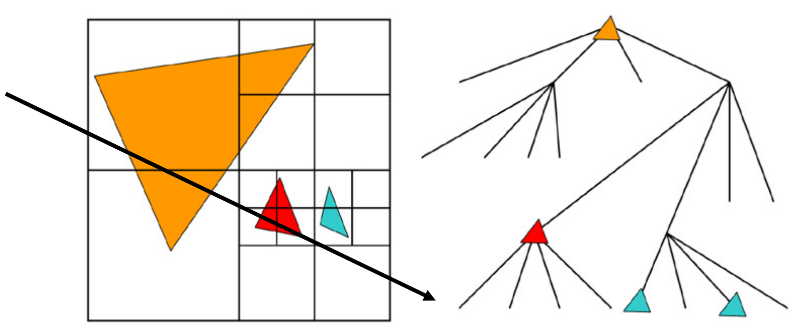


Octree/(Quadtree)



Adaptive Grid Discussion

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

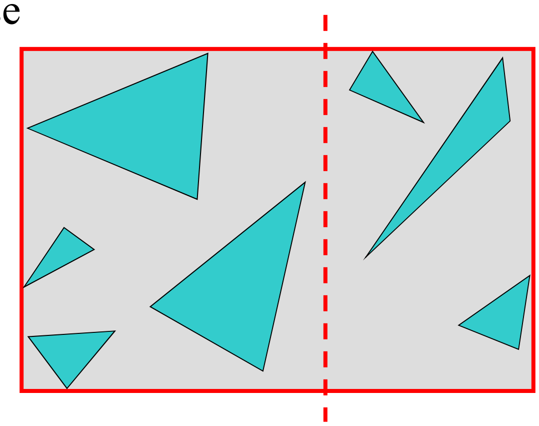


Ray-tracing Acceleration

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

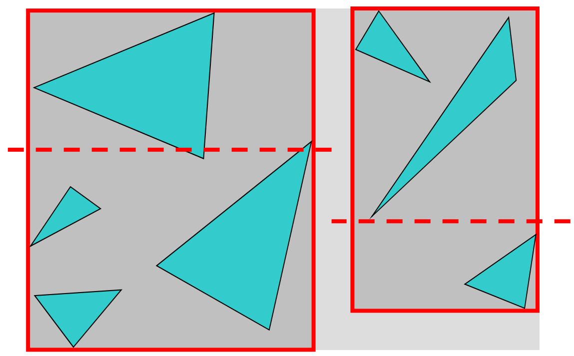
Bounding Volume Hierarchy

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



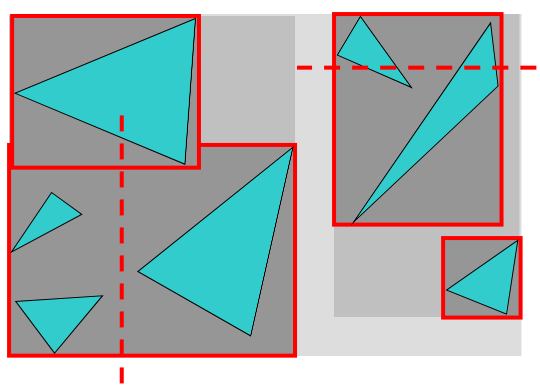
Bounding Volume Hierarchy

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



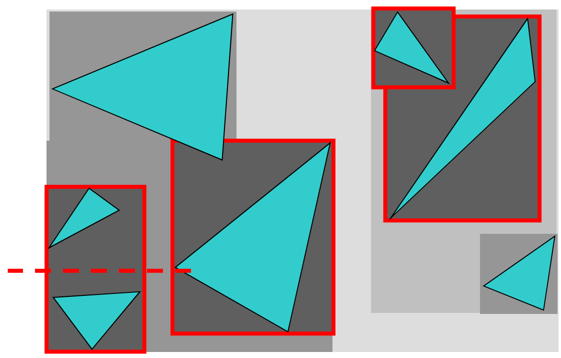
Bounding Volume Hierarchy

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



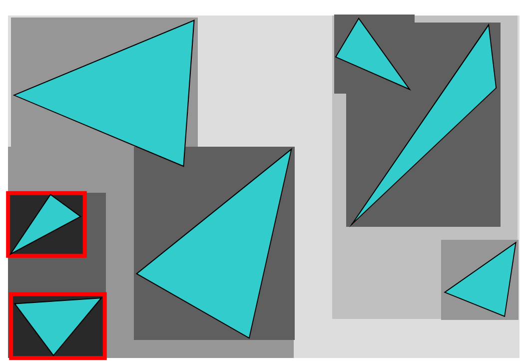
Bounding Volume Hierarchy

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



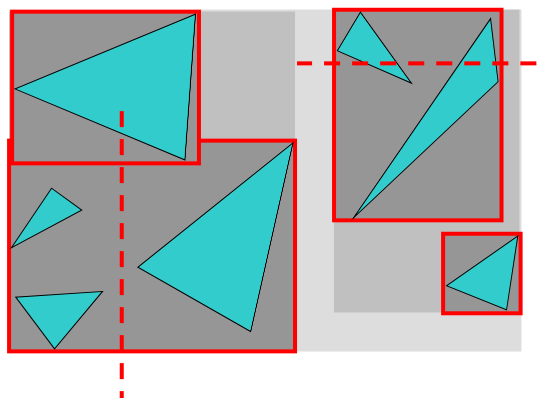
Bounding Volume Hierarchy

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



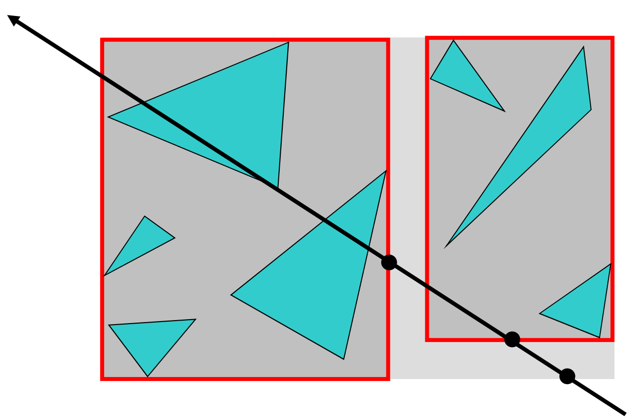
Where to split objects?

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



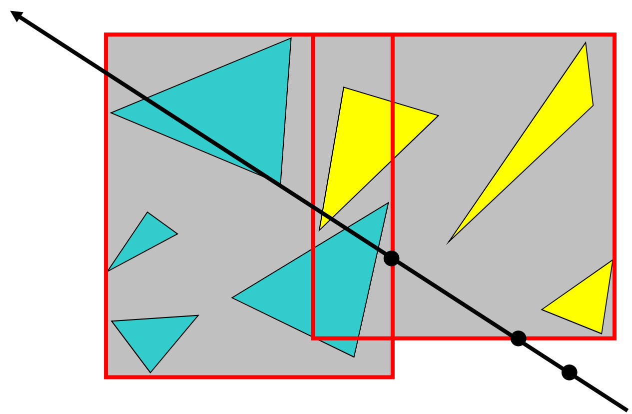
Intersection with BVH

- 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

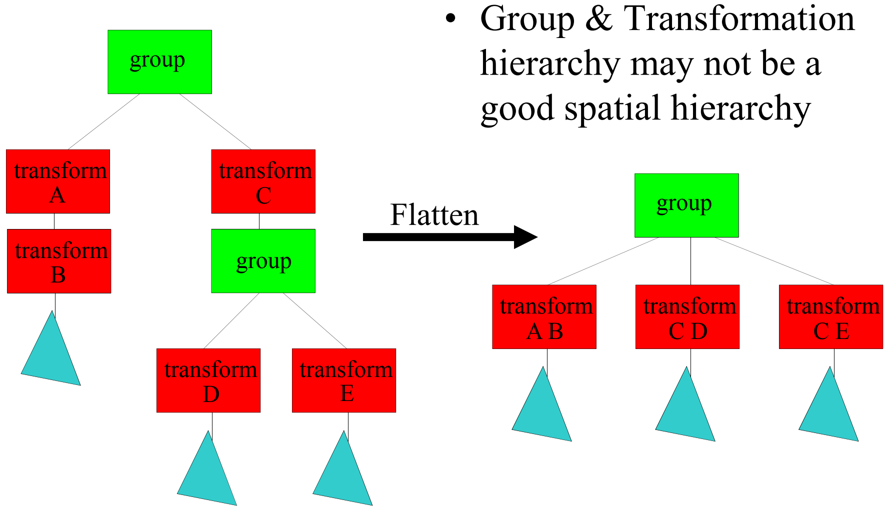
- Advantages
  - easy to construct
  - easy to traverse
  - binary
- Disadvantages
  - may be difficult to choose a good split for a node
  - poor split may result in minimal spatial pruning

Ray-tracing Acceleration

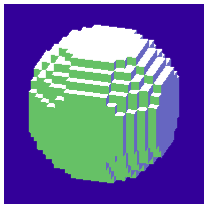
- Motivation – Distribution Ray Tracing
- Bounding Boxes
- Spatial Acceleration Data Structures
- Flattening the Transformation Hierarchy

Transformation Hierarchy

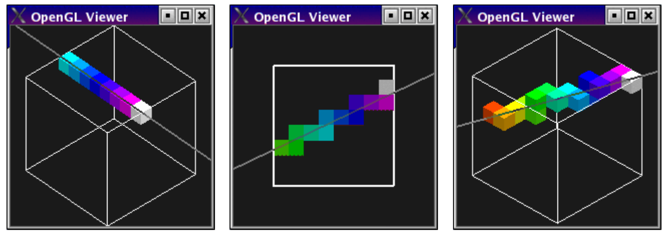
- Group & Transformation hierarchy may not be a good spatial hierarchy



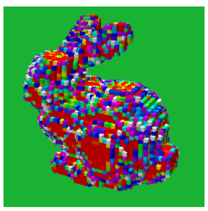
Ray Marching Visualization



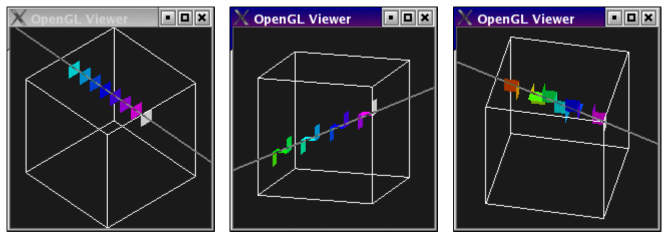
sphere voxelization



cells traversed



primitive density



entered faces

- Next time: ray-tracing at Pixar (or cancel class)