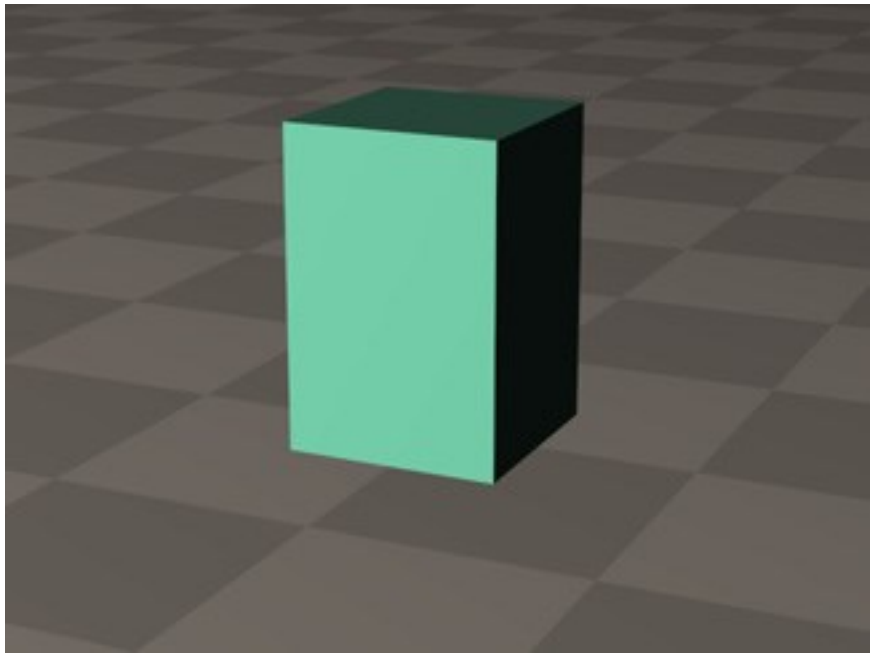
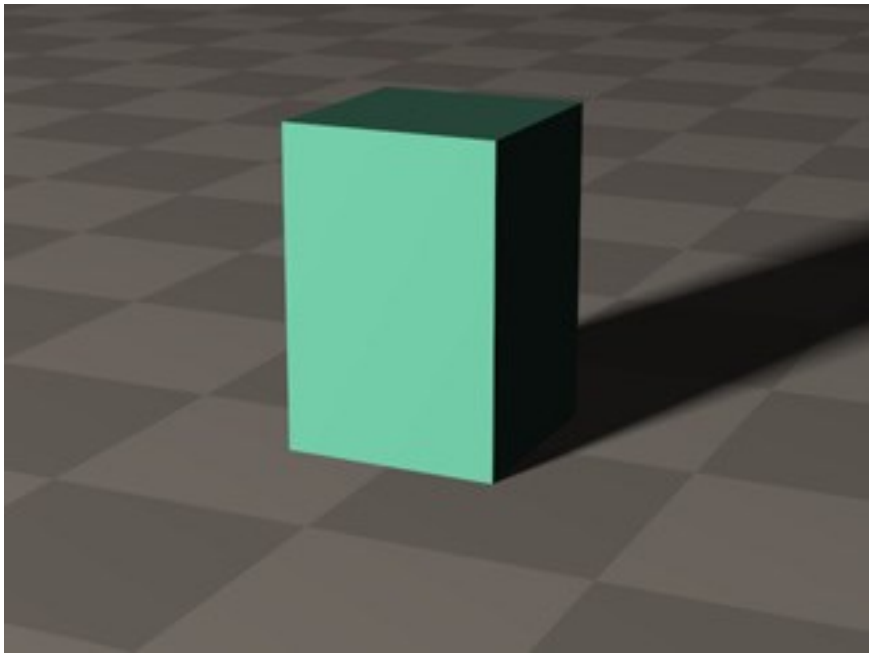
CS100433
Shadows

Junqiao Zhao 赵君峤
Department of Computer Science and Technology
College of Electronics and Information Engineering
Tongji University

Why shadows?

- 3D cueing
  - location, even motion

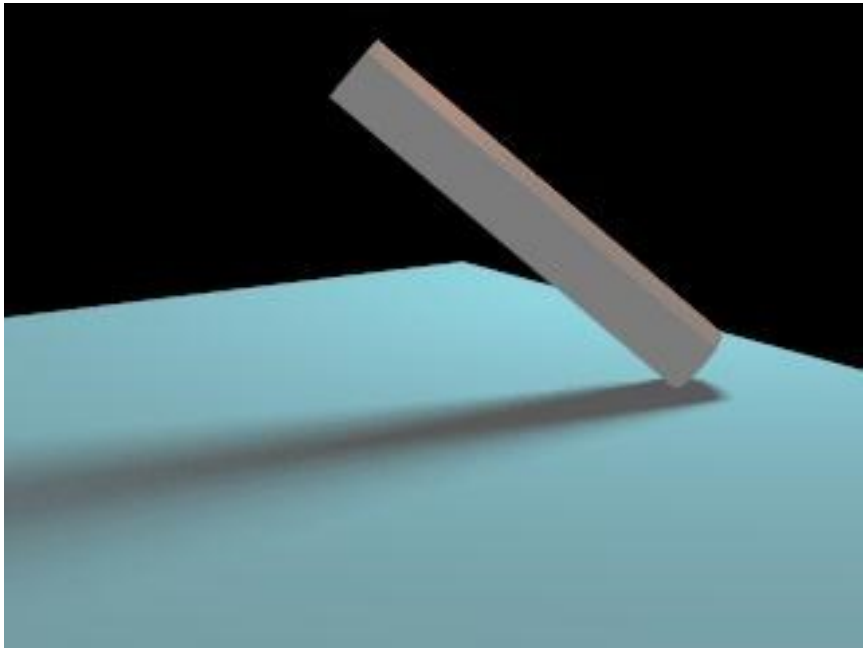




[Tom Thorne, Edinburgh]

How shadows are generated?

- Areas hidden from the light source by occlusion cause by objects
- Hard shadows and soft shadows



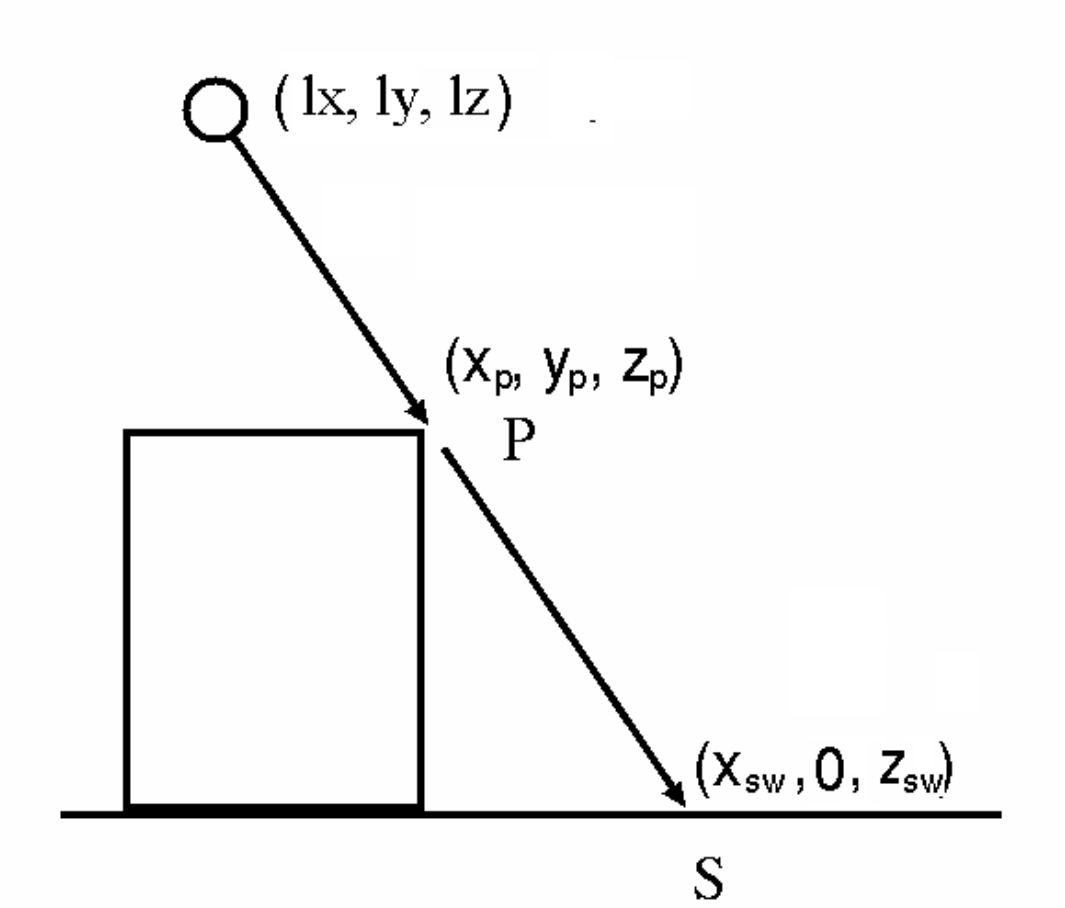
[Tom Thorne, Edinburgh]

Shadow techniques

- Ground shadow
- Shadow texture
- Shadow map
- Shadow volume

Ground shadow

- Shadow cast by objects onto the ground (y=0)



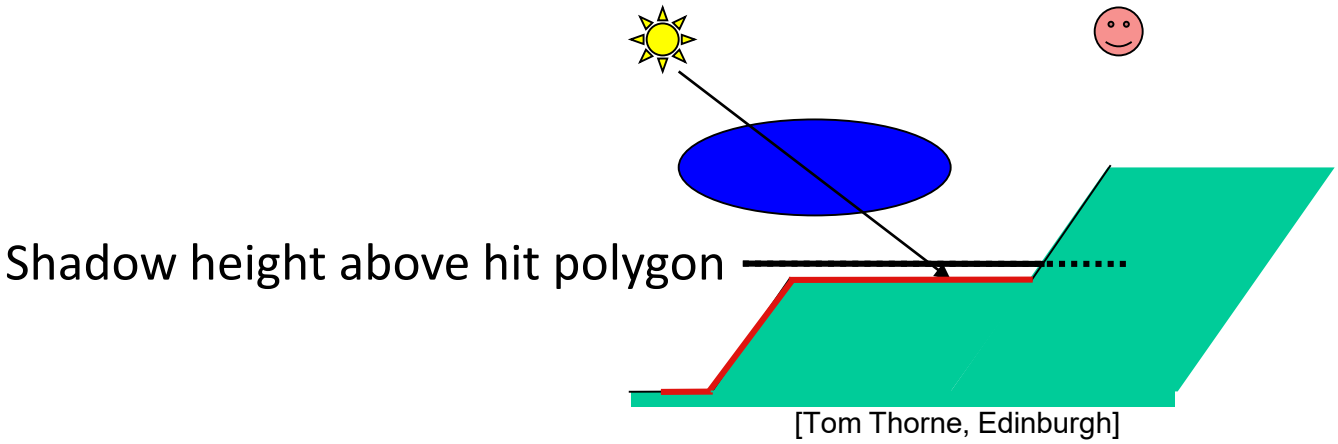
Point light ground shadows

- Blinn 88
- The matrix transform the object onto the ground

$$\begin{bmatrix} x_{sw} \\ 0 \\ z_{sw} \\ 1 \end{bmatrix} = \begin{bmatrix} l_y & -l_x & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -l_z & l_y & 0 \\ 0 & -1 & 0 & l_y \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

Draw the ground shadow

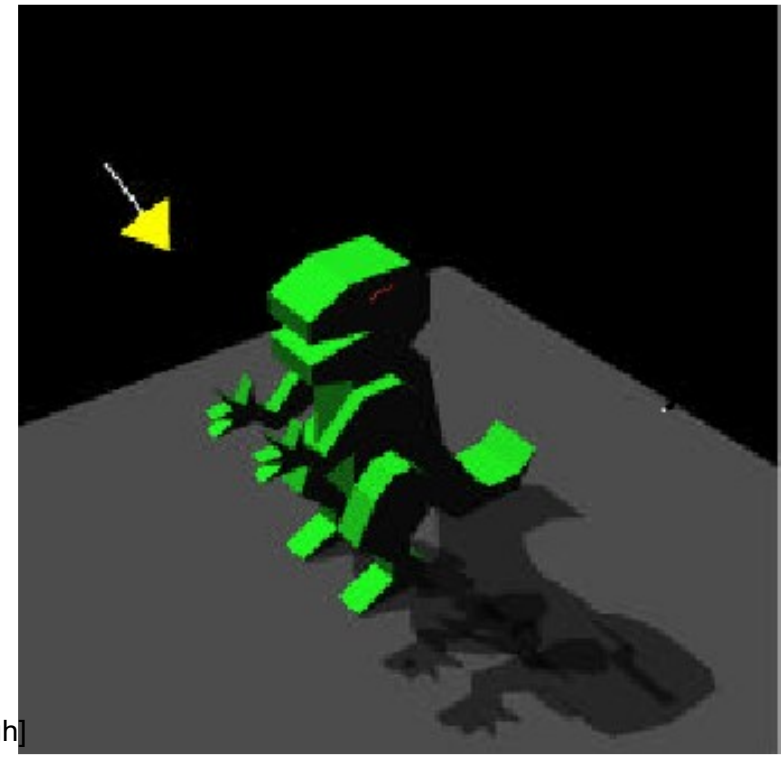
- The matrix transform the object onto the ground
- Thus
  - Draw the object
  - Multiply the shadow matrix
  - Redraw the object in grey



[Tom Thorne, Edinburgh]

Problem of ground shadow

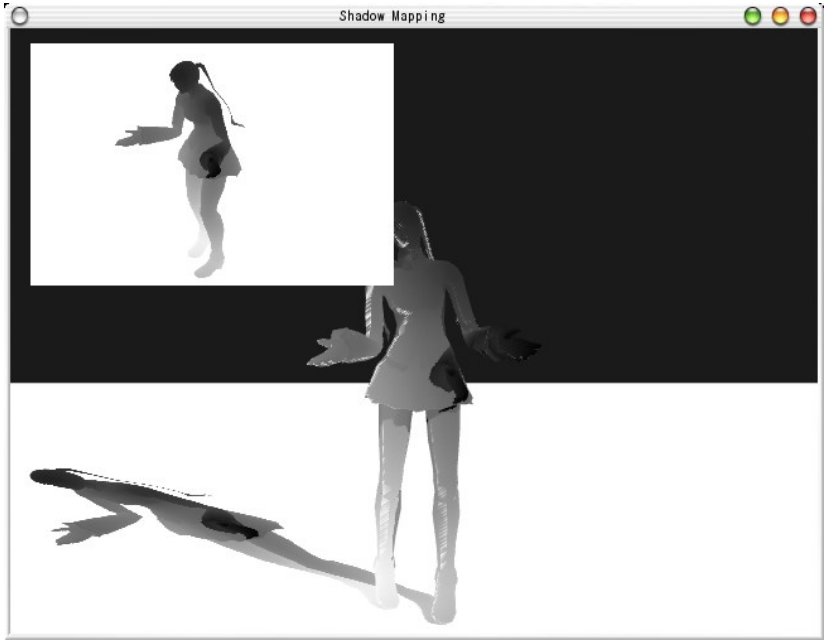
- The shadow only cast onto (ground) planes
- The shadow is hard shadow
- The performance is not optimal in static scene



[Tom Thorne, Edinburgh]

Shadow texture

- Using a shadow image as a texture
  - Occluder from light’ s view
- Project the image onto the object
  - Can be curved surface or other objects



[Tom Thorne, Edinburgh]

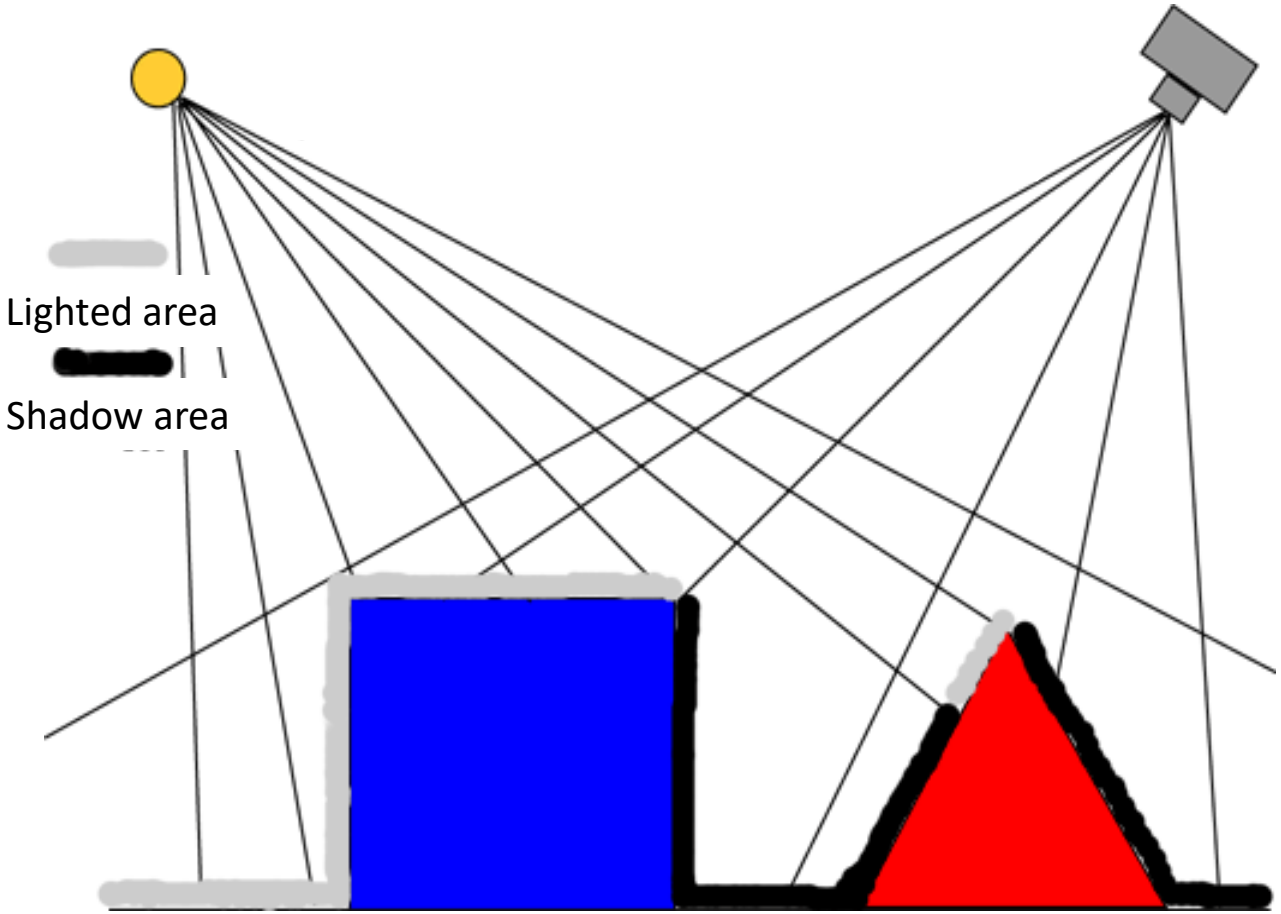
Shadow map

- Preparation
  - Prepare a depth buffer for each light
  - Render the scene from the light position
  - Save the depth information in the Depth buffer
- Rendering the scene
  - Render the objects; whenever rendering an object, check if it is shadowed or not by transforming its coordinate into the light space
  - After the transformation, if the depth value is larger than that in the light's depth buffer it should be shadowed

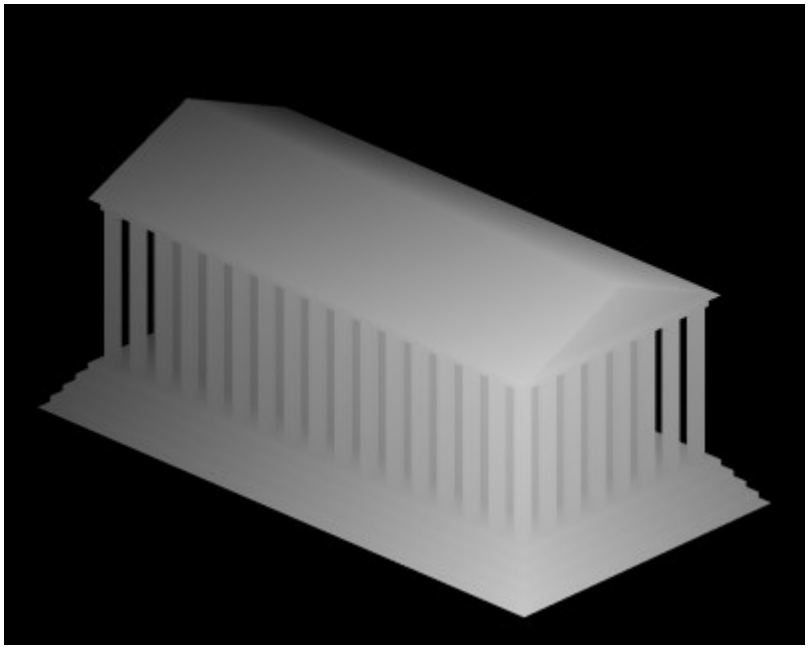
https://learnopengl.com/code_viewer_gh.php?code=src/5.advanced_lighting/3.1.2.shadow_mapping_base/shadow_mapping_base.cpp

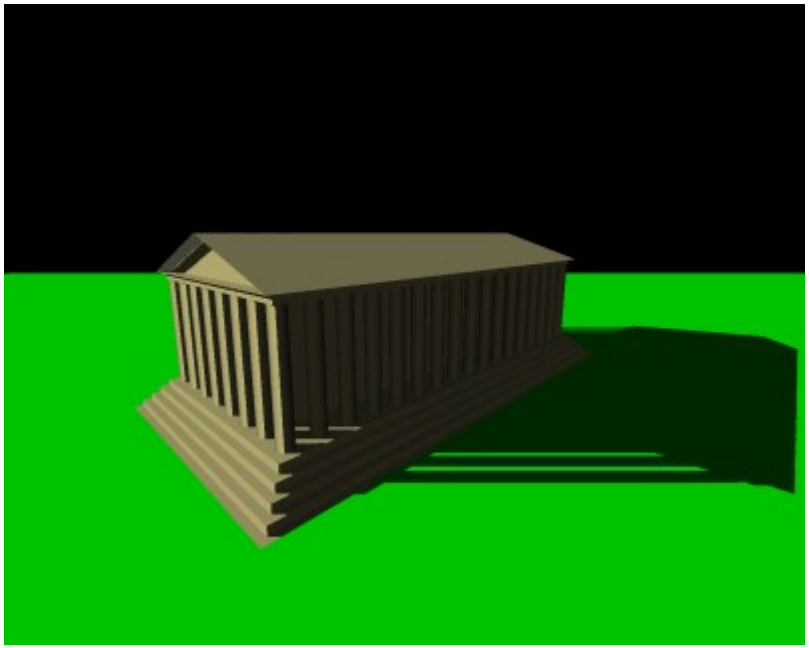
Shadow map

- Using Depth buffer



http://www.it.hiof.no/~borres/j3d/explain/shadow/bilder/shadowmap.gif





[Wikipedia]

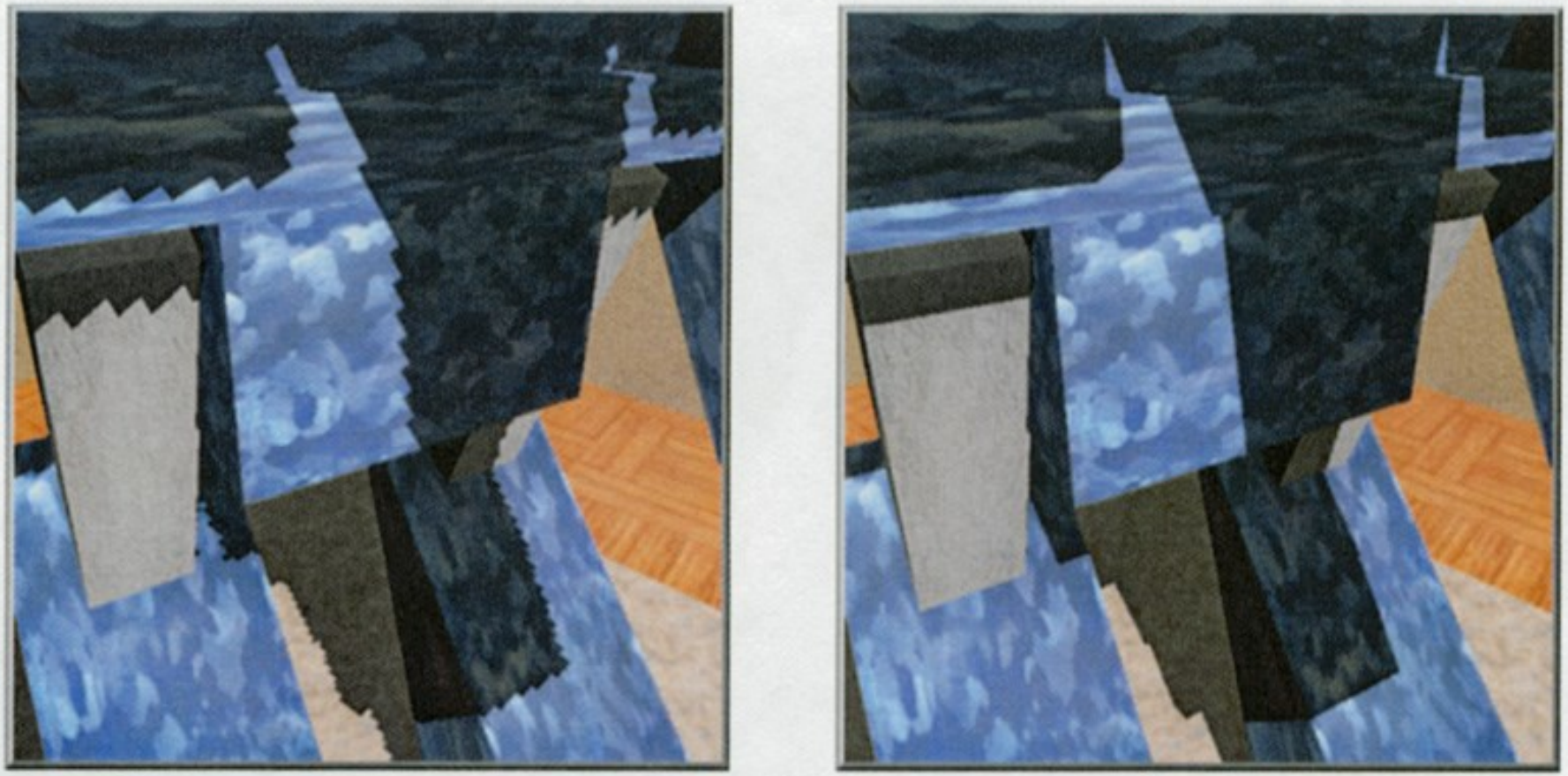


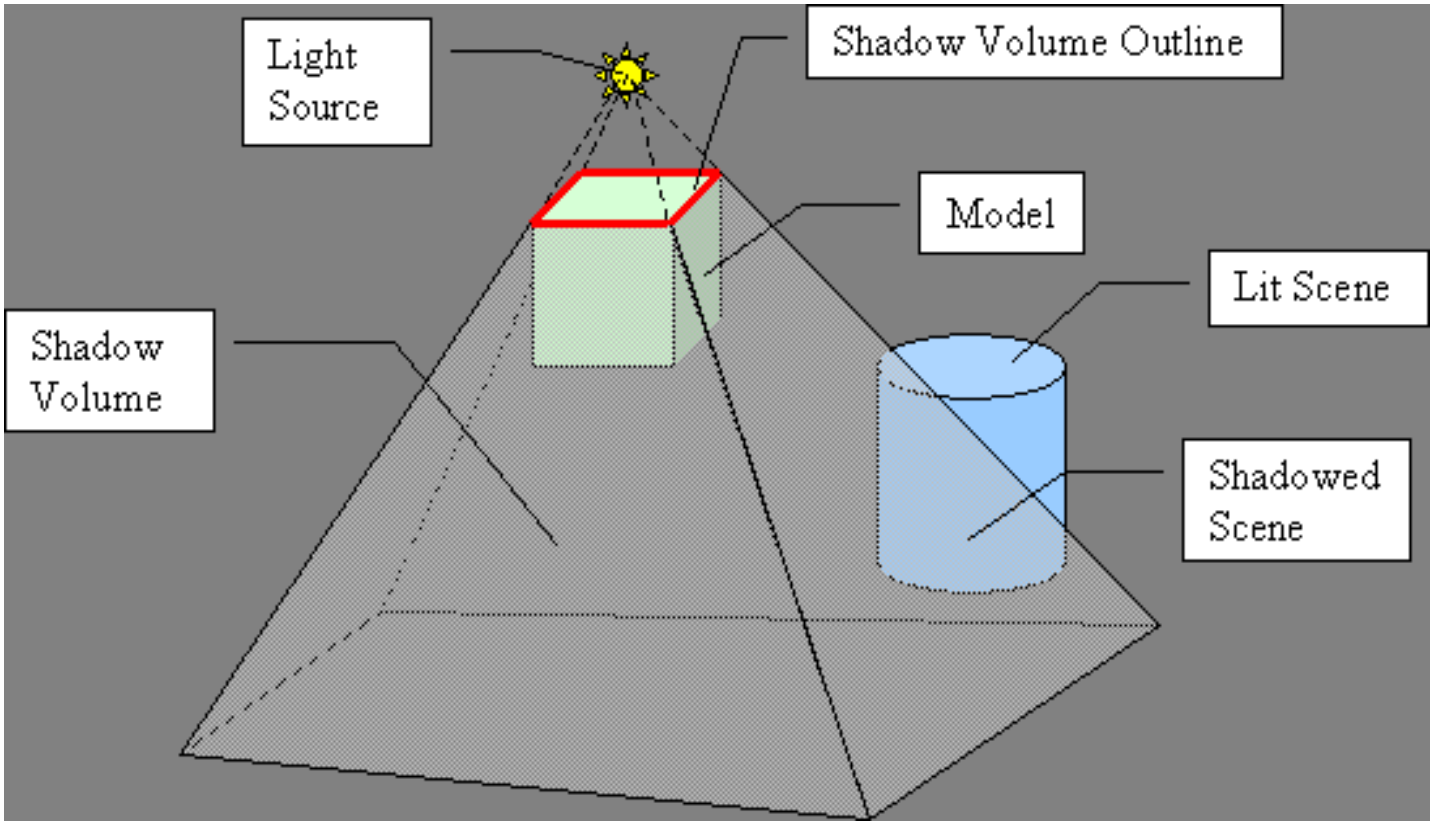
Figure 2: A conventional 2,048×2,048 pixel shadow map (left) compared to a 16 MB ASM (right). EFFECTIVE SHADOW MAP SIZE: 65,536×65,536 PIXELS.

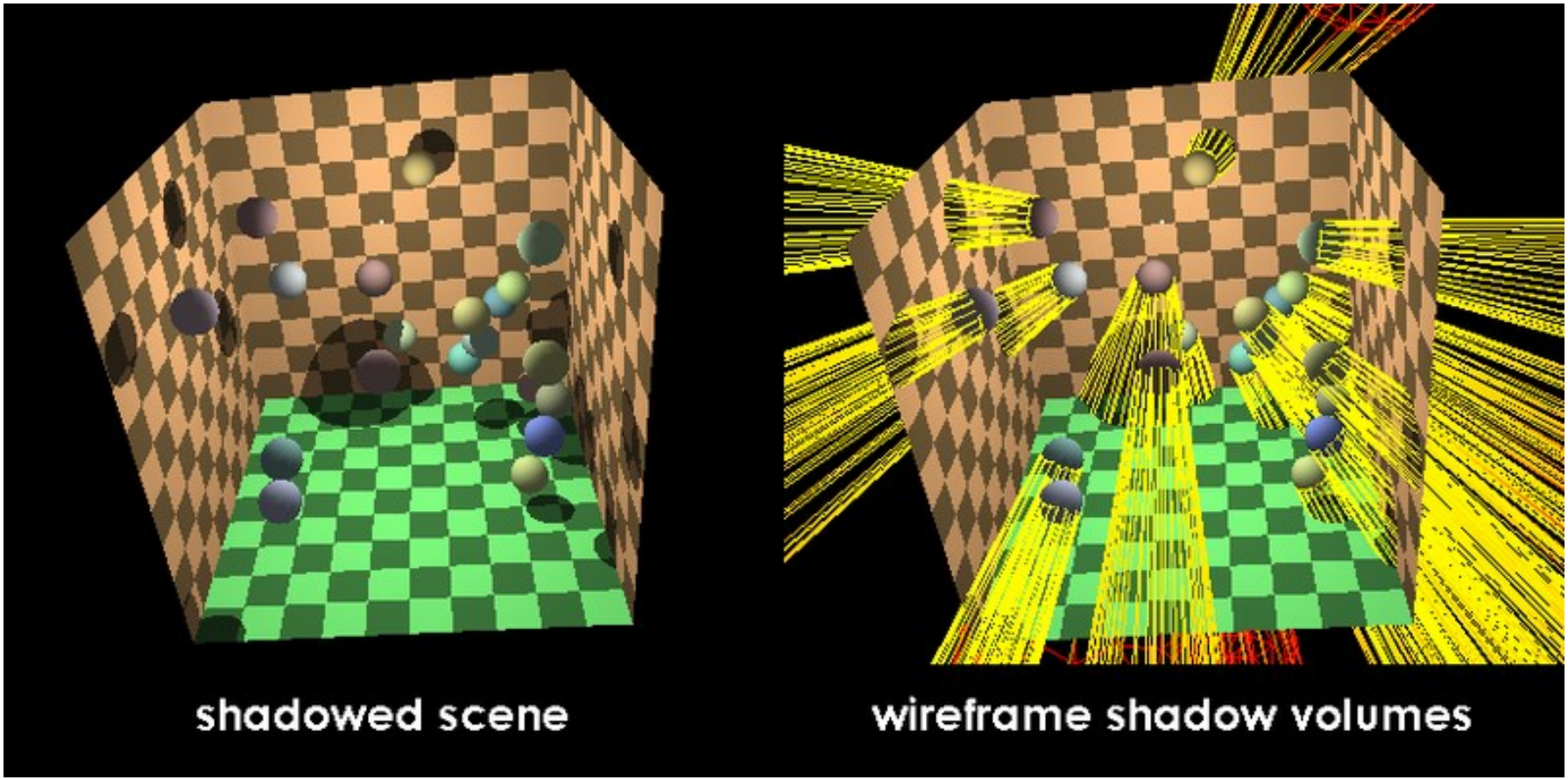
Shadow volume

- In the reality, the shadow cast by an object blocking light is a volume which is 3D!
- Project a ray from the light source through each vertex/silhouette in the shadow caster to infinity
- Any objects intersecting with the volume will get shadow on them.
  - Self-cast shadow
  - General-purpose

Shadow volumes

- Cast by sihouette

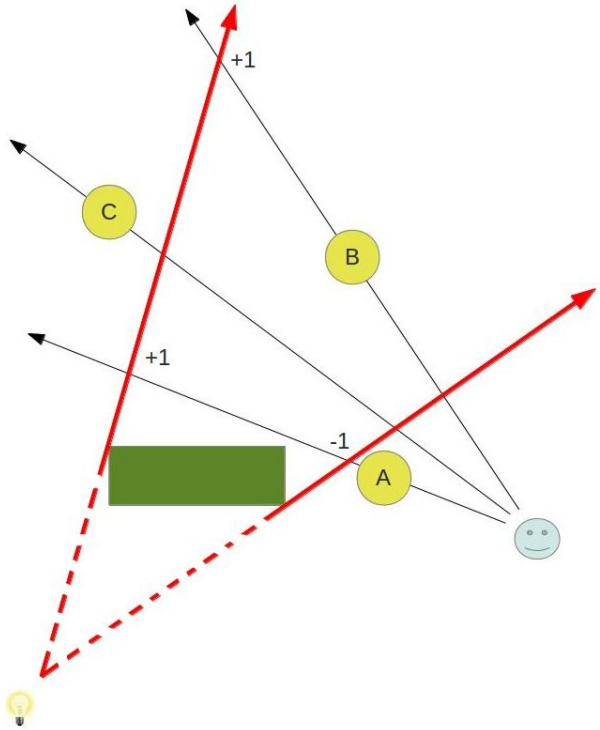




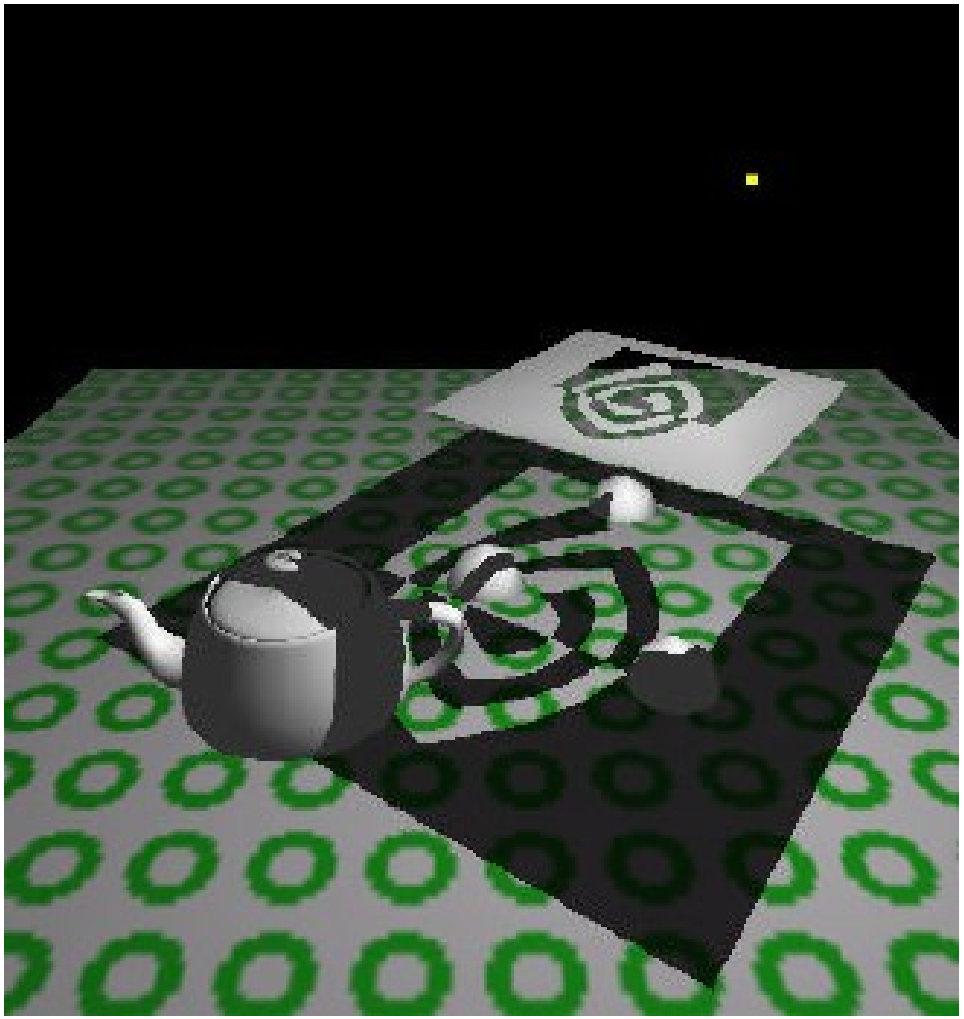
https://en.wikipedia.org/wiki/Shadow_volume

Stencil buffer

- Stencil buffer
  - A data buffer
  - Used as a Stencil
  - An integer per pixel
- Depth fail method
  - http://ogldev.atspace.co.uk/www/tutorial40/tutorial40.html
  - 1 Render the scene into Depth buffer
  - 2 Create shadow volume
  - 3 Render the volume in Stencil buffer following rules
  - 4 Render the scene under Stencil test



- Computation intensive
- Hard shadow

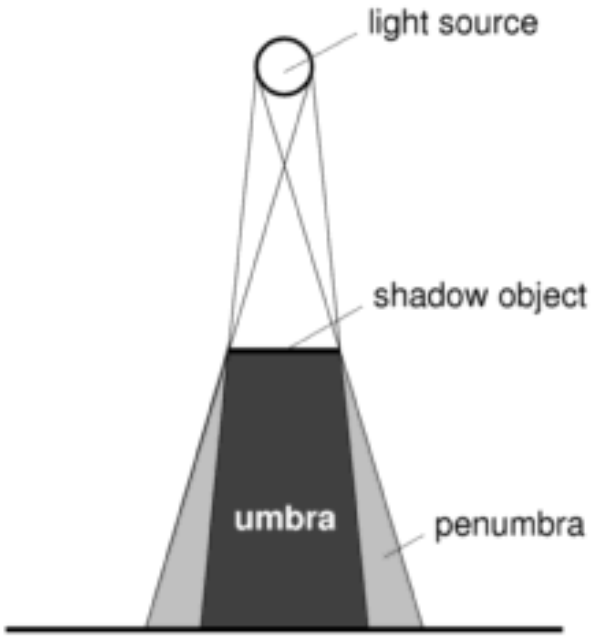


Stencil buffer based implementation

- Doom3 Made it popular

Soft shadows

- Made by area light
  - umbra – totally blocked from the light source
  - Penumbra – partially blocked from the light source
- Can be modelled by a collection of point light sources



Shadow map vs Shadow volume

- Faster than shadow volume
  - GPU based
- Less accurate because the resolution of the depth buffer
  - Aliasing at edges

- Questions?

CS100433
Ray tracing

Junqiao Zhao 赵君峤

Department of Computer Science and Technology

College of Electronics and Information Engineering

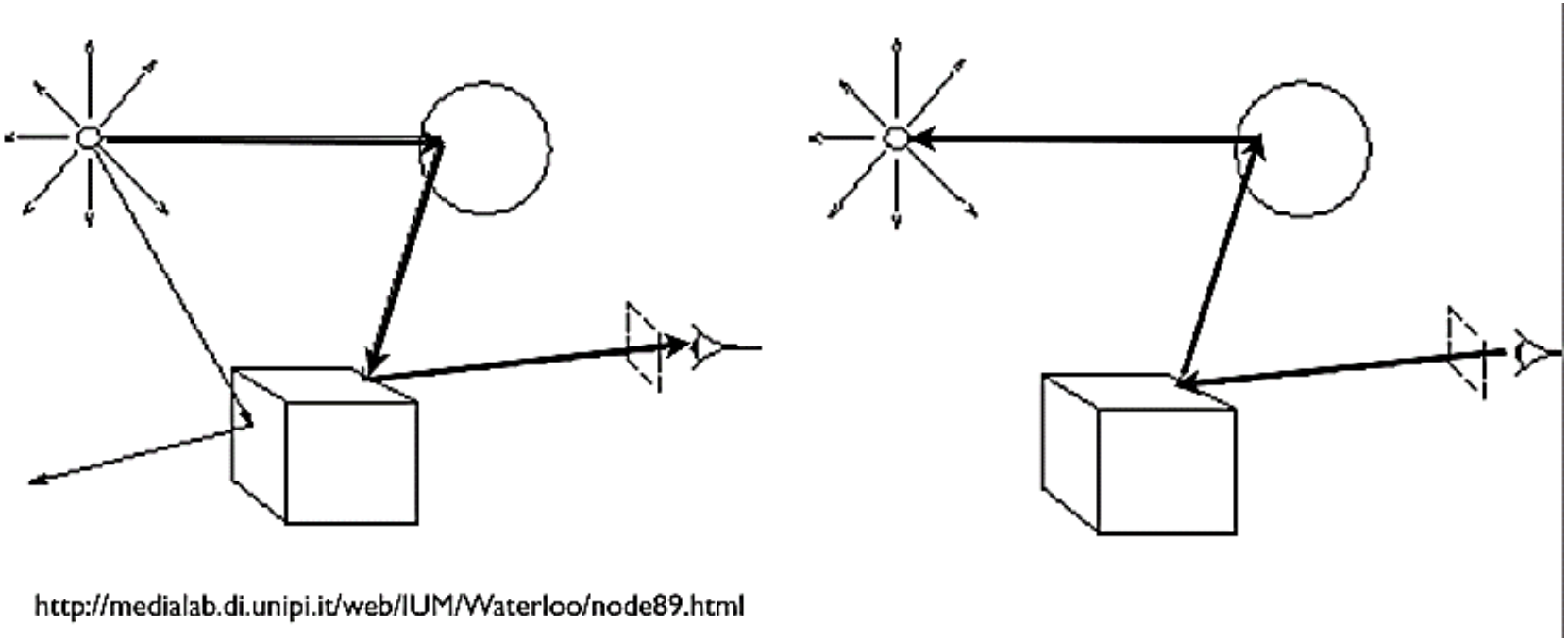
Tongji University

OpenGL vs Ray tracing

- OpenGL is based on a pipeline model in which primitives are rendered one at time
  - No shadows (shadow maps etc.)
  - No multiple reflections (environment maps etc.)
- Global approaches
  - Rendering equation
  - Ray tracing
  - Radiosity
  - Easily making effects: shadows, reflections, transparency

OpenGL vs Ray tracing

- Based on following light rays through the scene
- Iterate over pixels instead of primitives



http://medialab.di.unipi.it/web/IUM/Waterloo/node89.html

Ray tracing

For each pixel {

Shoot a ray from camera to pixel; //Perspective

for all objects in scene

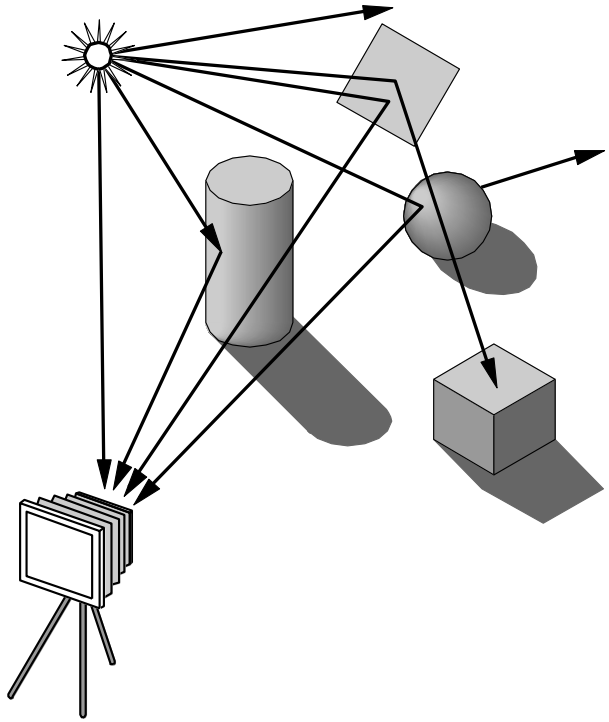
Compute intersection with ray

Find object with closest intersection

Recursively shoot rays from the intersection point

Display color using object + light property

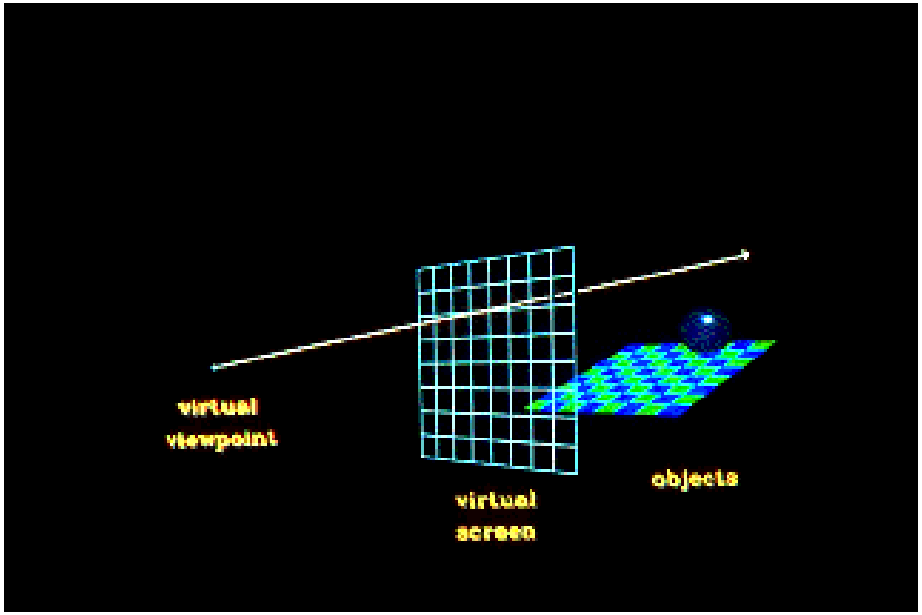
}

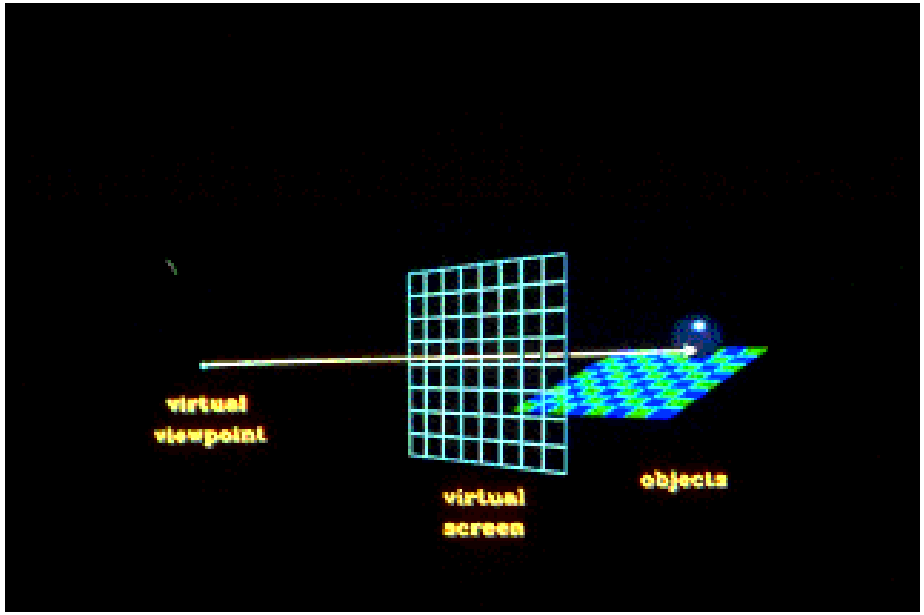


(Ed Angel, NewMexico)

Ray tracing

- Sometimes the ray misses all of the objects
- and sometimes the ray will hit an object

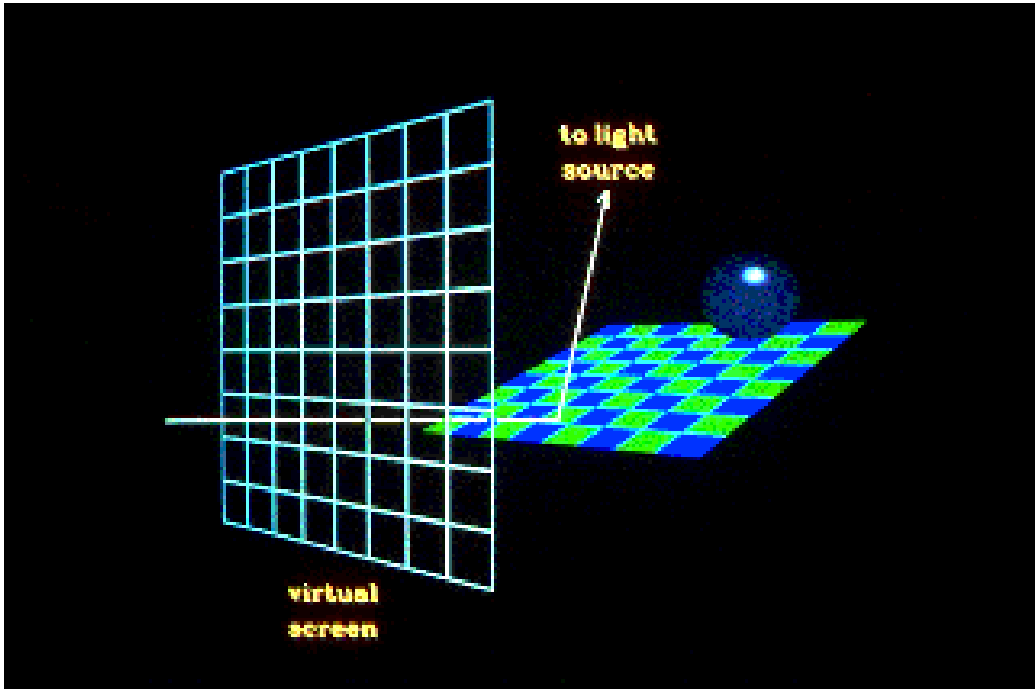




(Siggraph Education: Overview of Ray Tracing)

Ray tracing

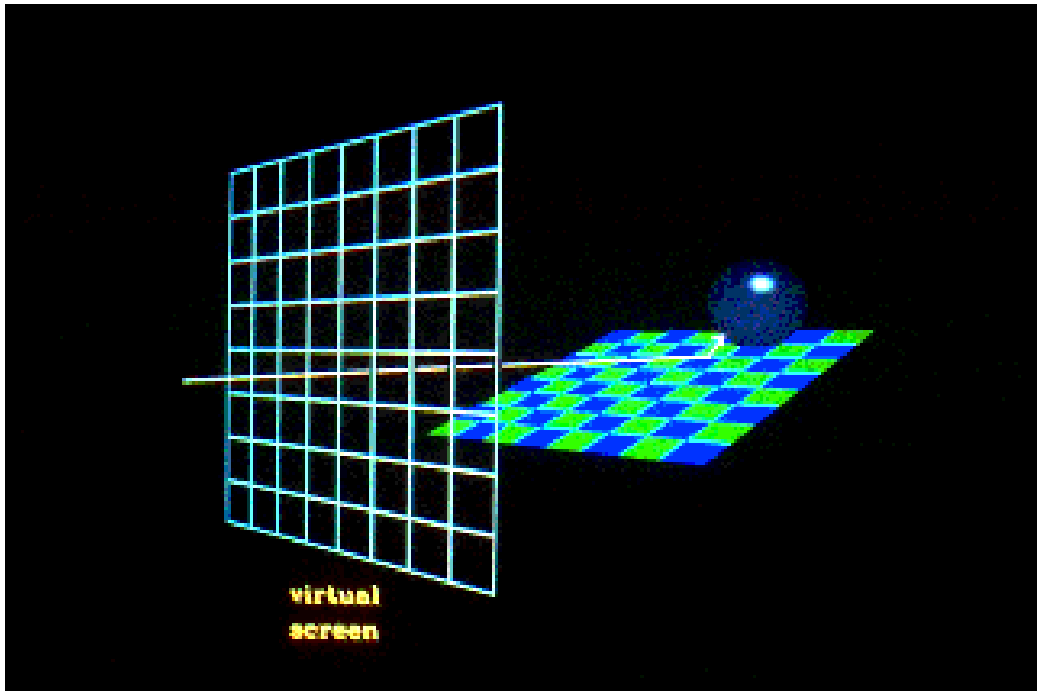
- If the ray hits an object, we want to know if that point on the object is in a shadow.
- So, when the ray hits an object, a secondary ray, called a **"shadow" ray**, is shot towards the light sources.



(Siggraph Education: Overview of Ray Tracing)

Ray tracing

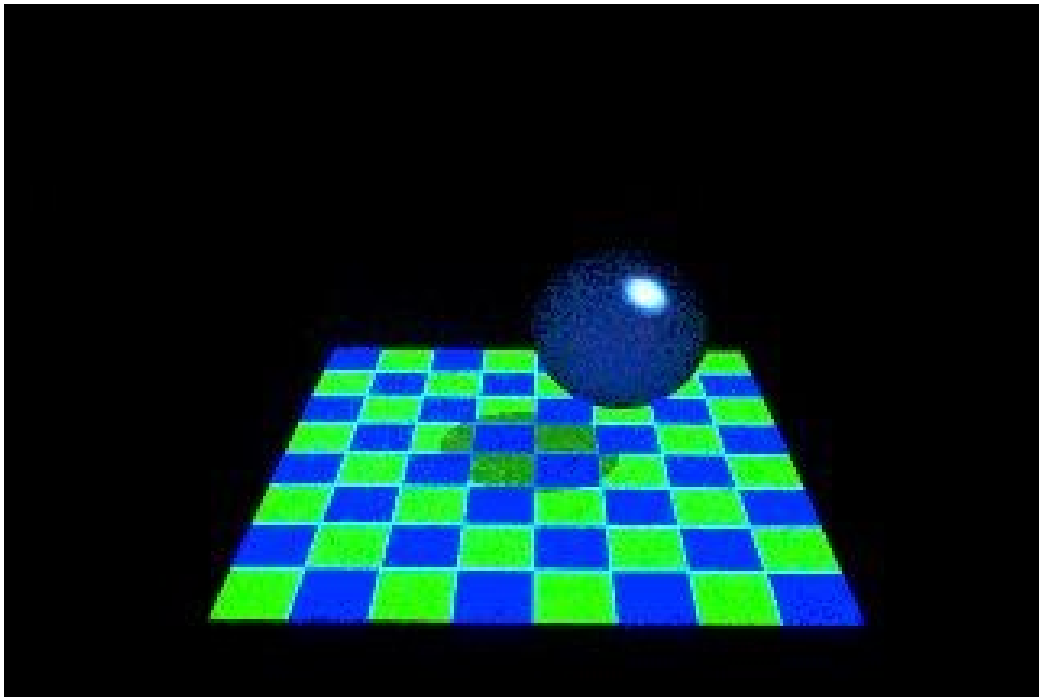
- If this shadow ray hits another object before it hits a light source, then the first intersection point is in the shadow of the second object.
- For a simple illumination model this means that we only apply the ambient term for that light source.



(Siggraph Education: Overview of Ray Tracing)

Ray tracing

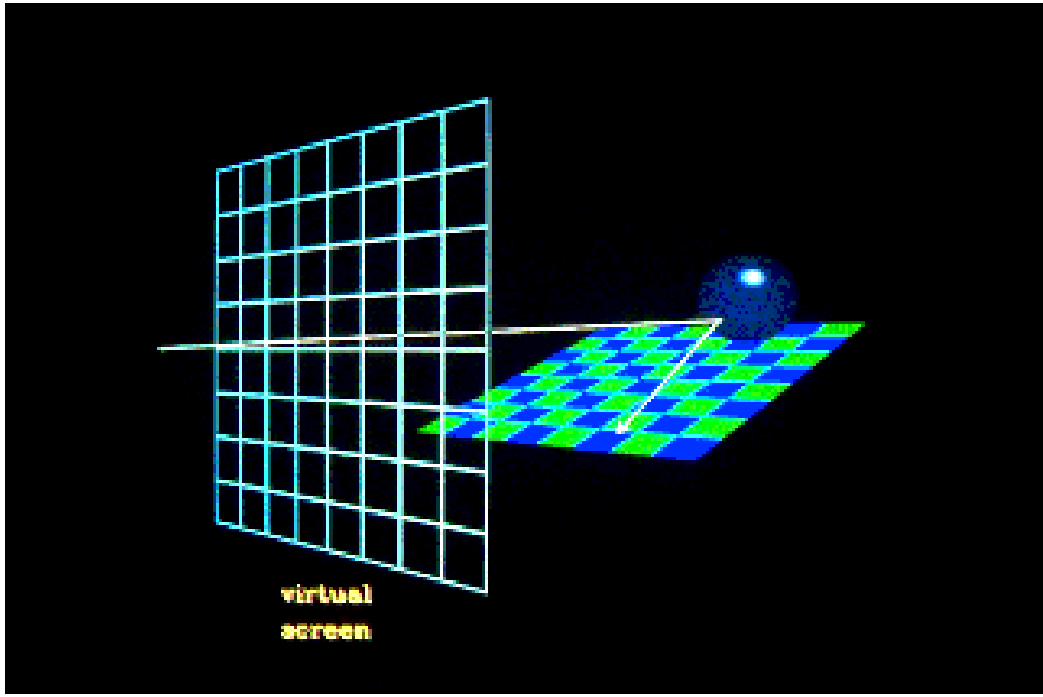
- If this shadow ray hits another object before it hits a light source, then the first intersection point is in the shadow of the second object.
- For a simple illumination model this means that we only apply the ambient term for that light source.



(Siggraph Education: Overview of Ray Tracing)

Ray tracing

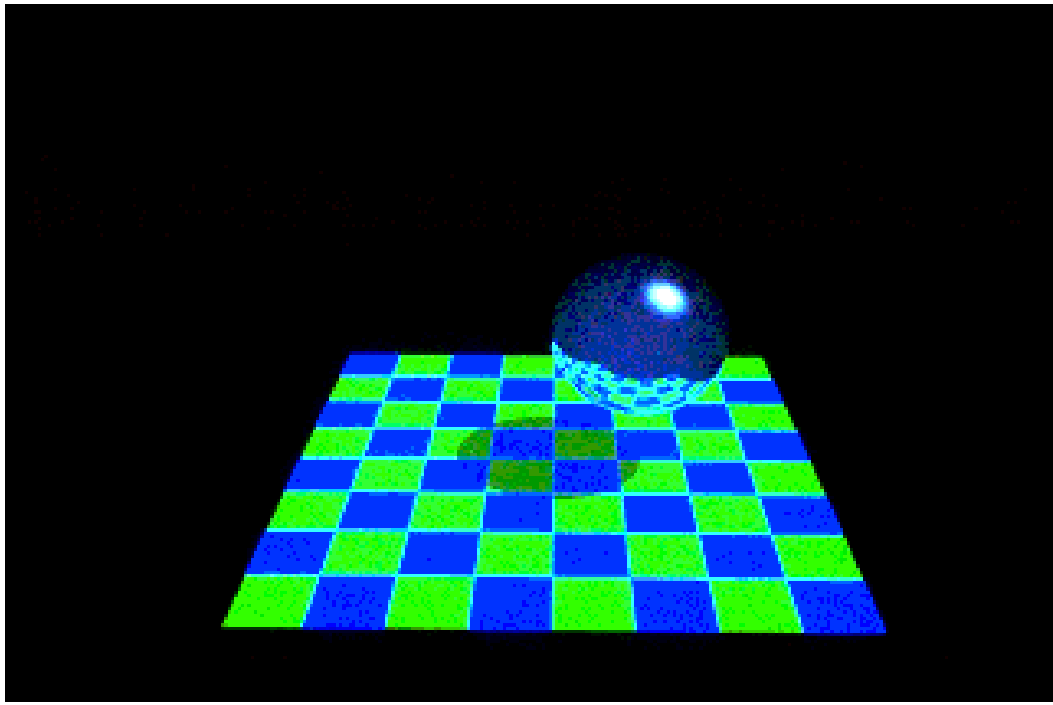
- Also, when a ray hits an object, a **reflected ray** is generated which is tested against all of the objects in the scene.



(Siggraph Education: Overview of Ray Tracing)

Ray tracing

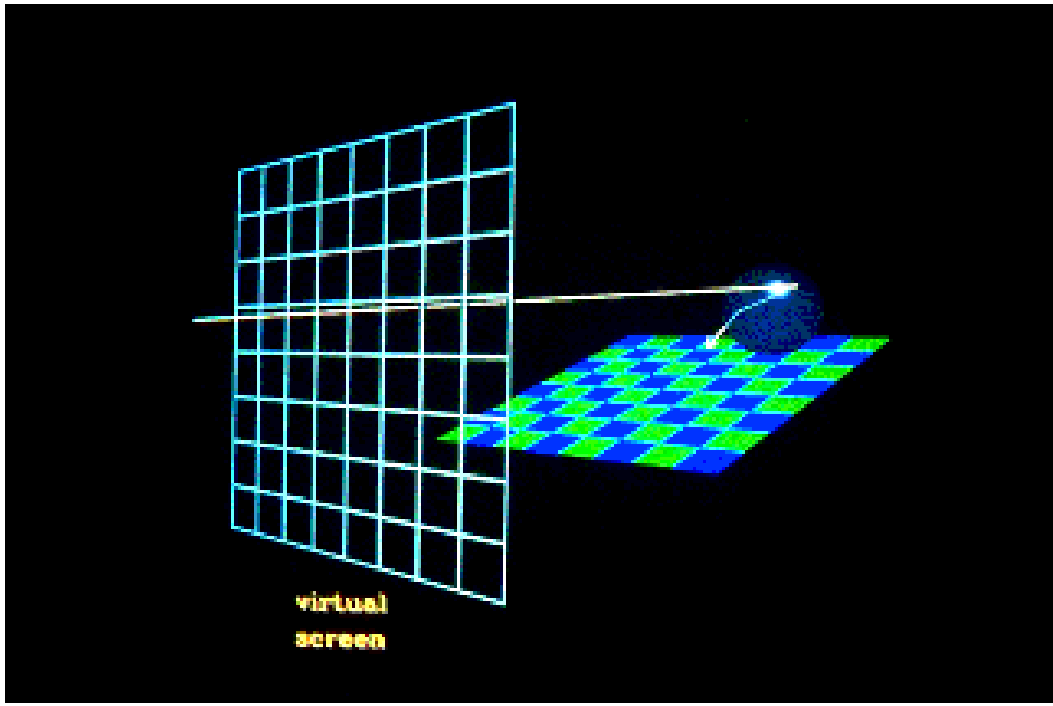
- If the reflected ray hits an object then a local illumination model is applied at the point of intersection and the result is carried back to the first intersection point.



(Siggraph Education: Overview of Ray Tracing)

Ray tracing

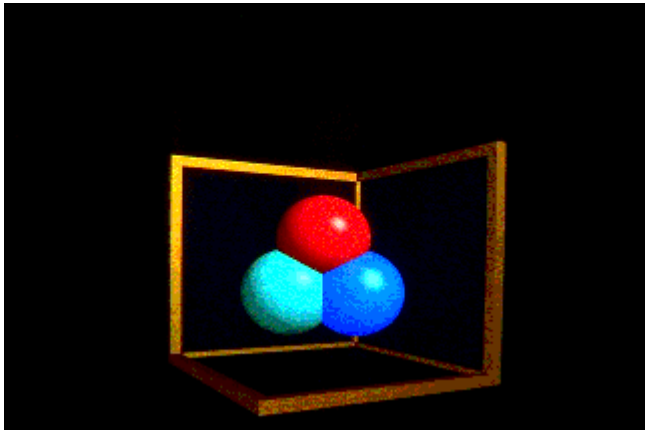
- If the intersected object is transparent, then a **transmitted ray** is generated and tested against all the objects in the scene.

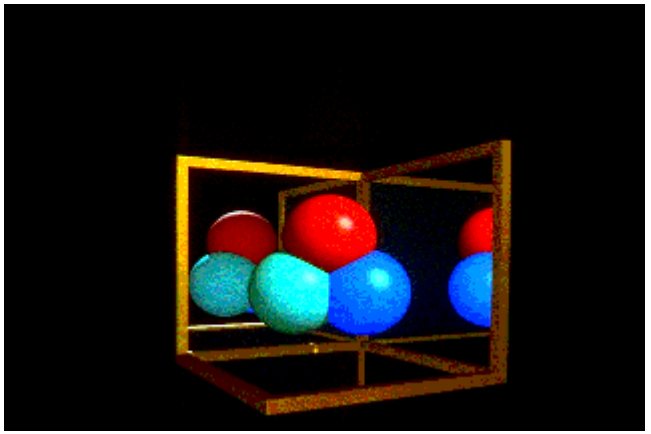


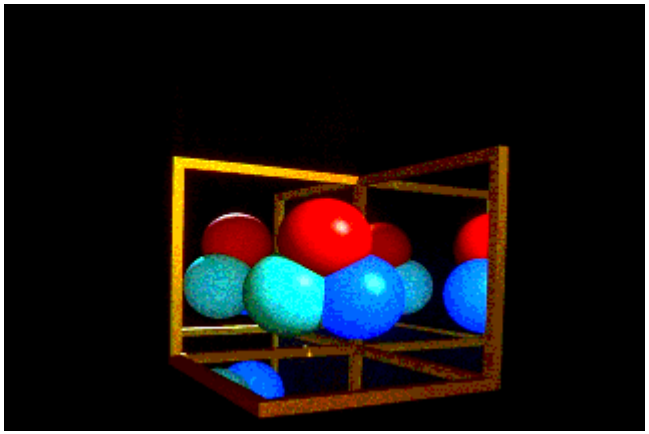
(Siggraph Education: Overview of Ray Tracing)

Ray tracing

- The reflection ray can be implemented recursively
- There can be no reflection or multiple reflection

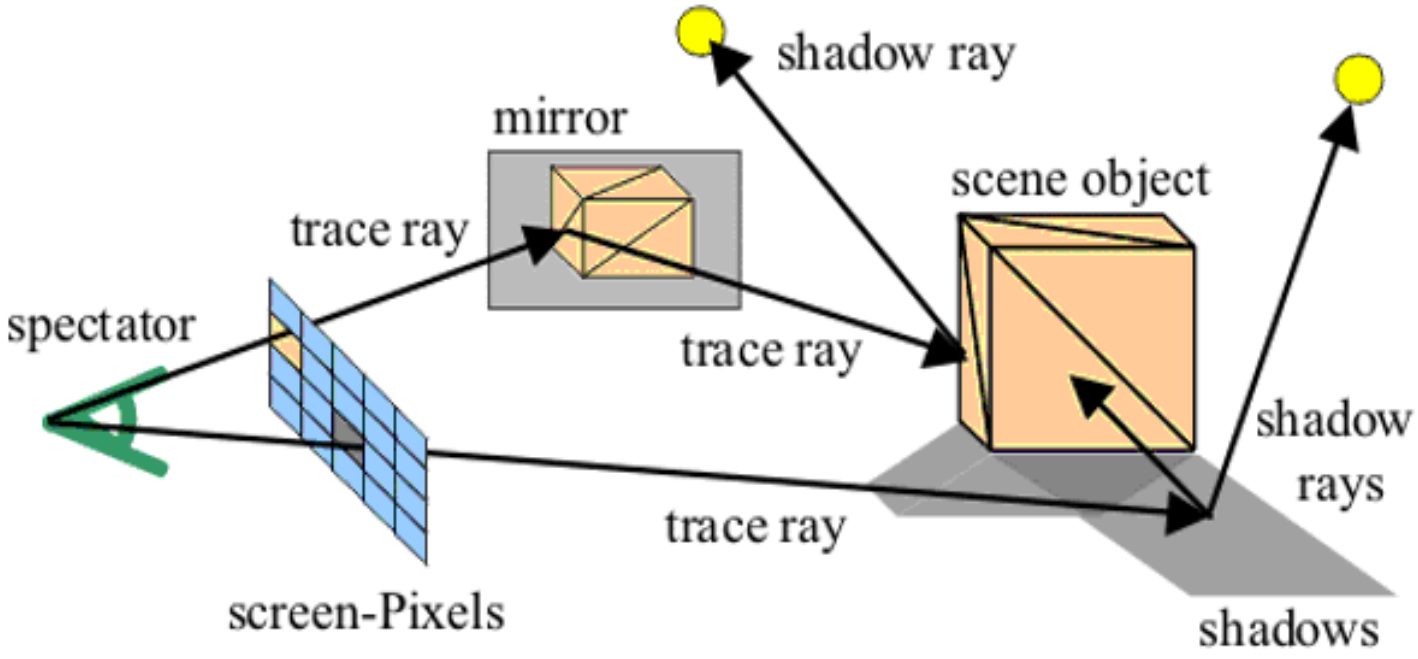






http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtrace0.htm

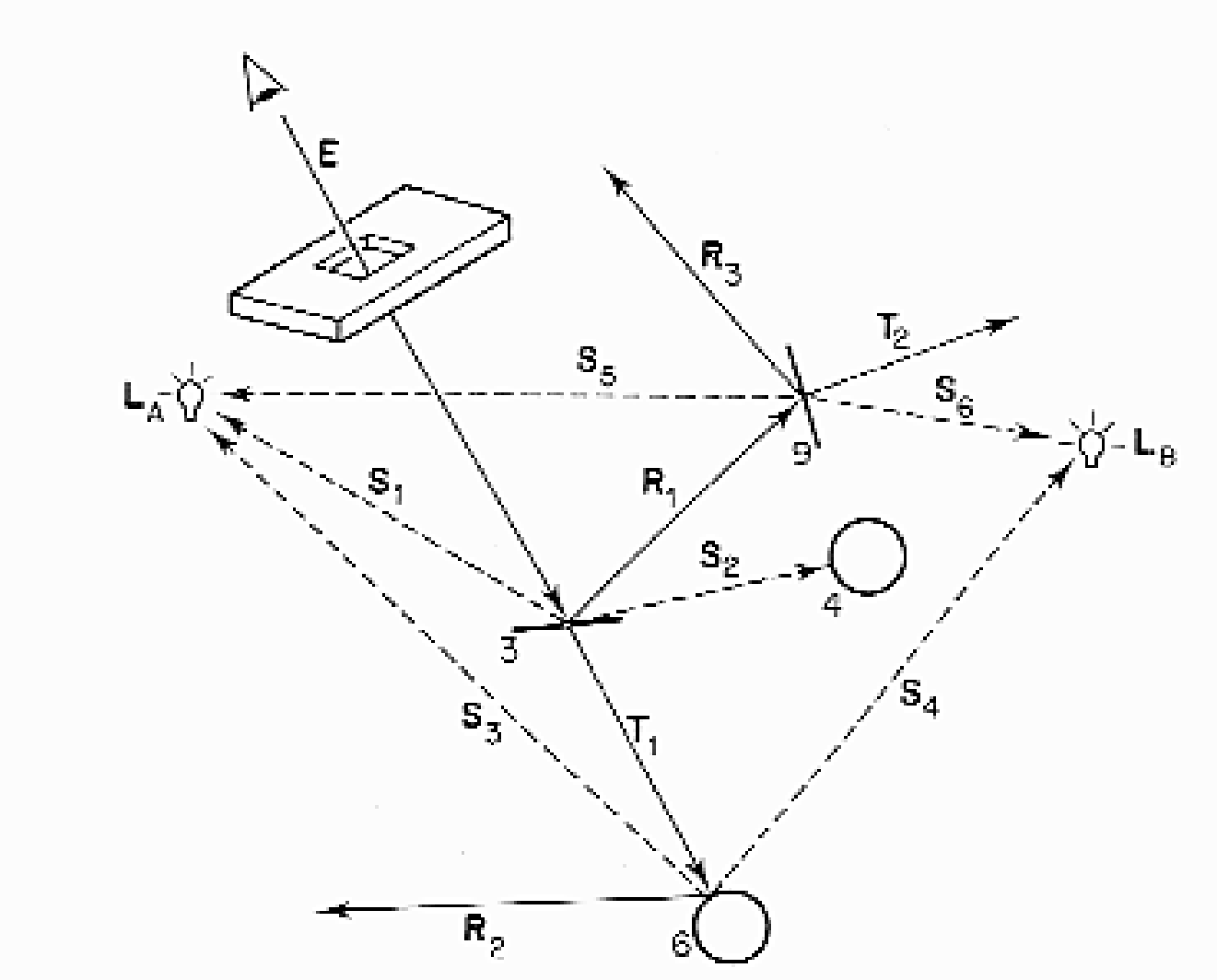
Rays



(http://www.ice.rwth-aachen.de/research/tools-projects/grace/ray-traycing/)

Ray Tree

- Rays are generated recursively



Ray Tree

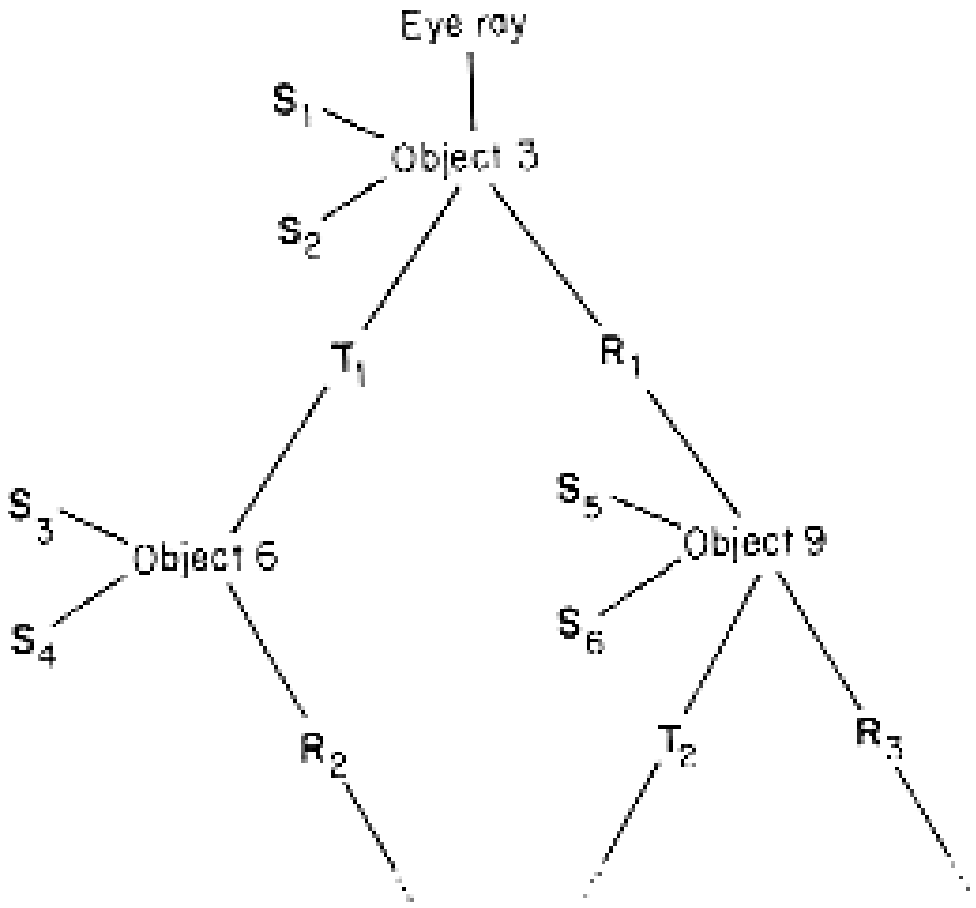
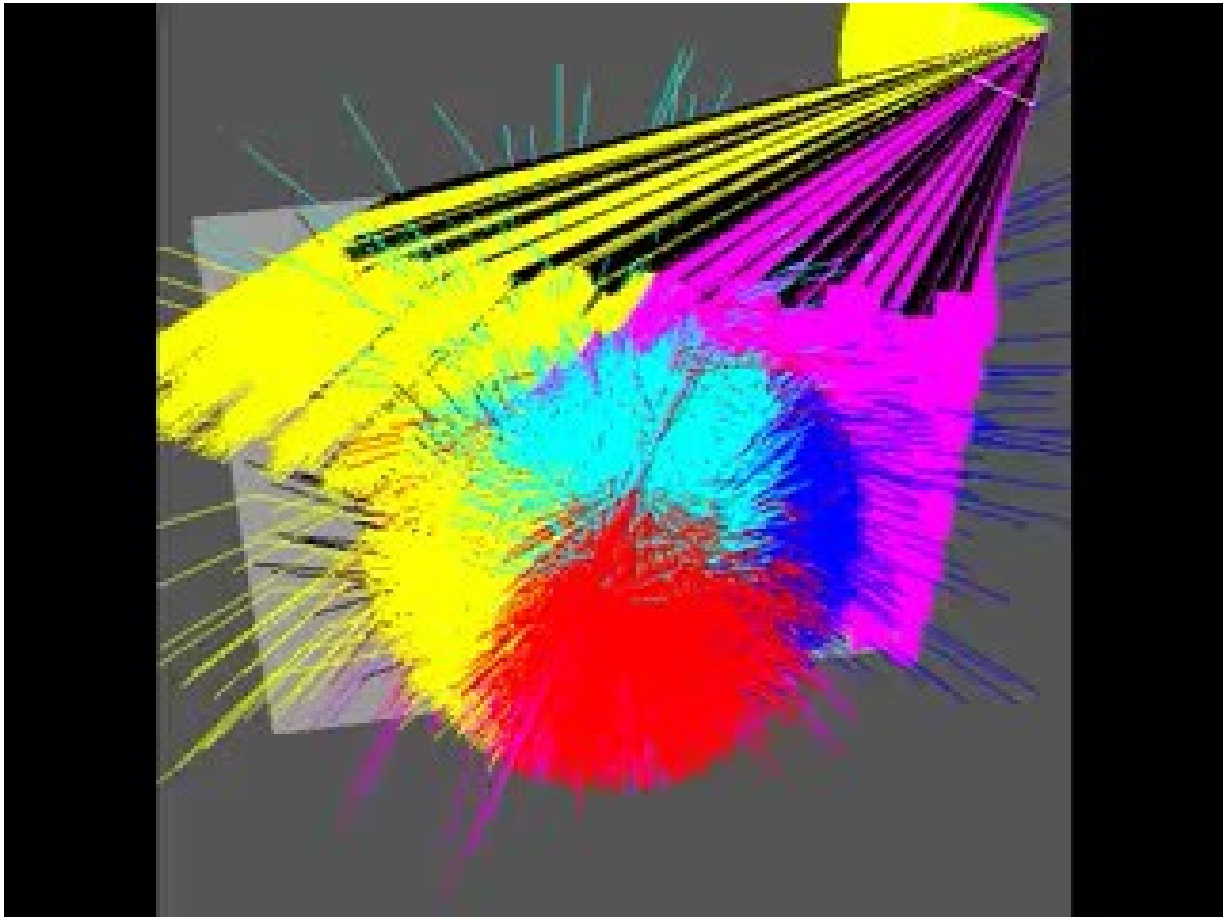


Fig. 12. The ray tree in schematic form.

Visualization of rays



https://www.youtube.com/watch?v=aKqxonOrl4Q

Ray tracing

- Should be able to handle all physical interactions
- Ray tracing paradigm is computation intensive
- Most rays do not affect what we see
- Scattering produces many (infinite) additional rays
- Alternative: ray casting

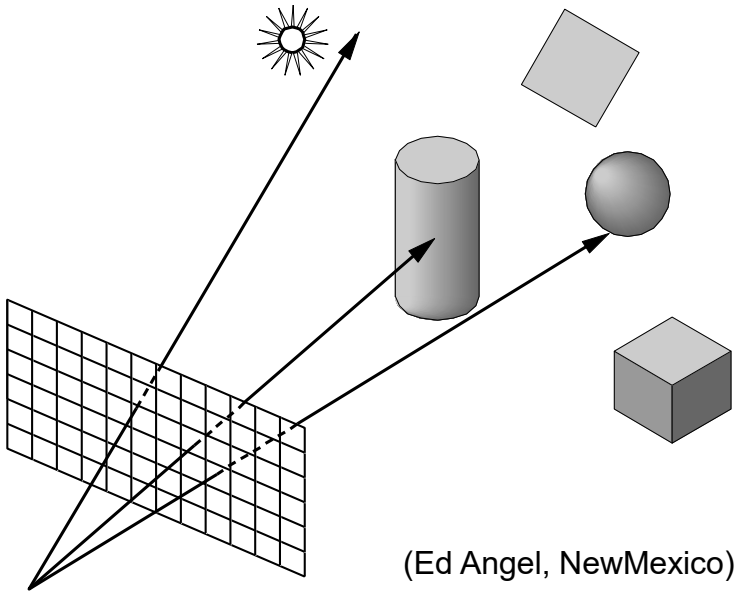
Diffuse Surfaces

- Theoretically the scattering at each point of intersection generates an infinite number of new rays that should be traced
- We could only trace the transmitted and reflected rays but use the Phong model to compute shade at point of intersection
- We can also sample the scattering rays by using Monto-carlo

Ray casting

- Only rays that reach the eye matter
- Reverse direction and cast rays
- Need at least one ray per pixel

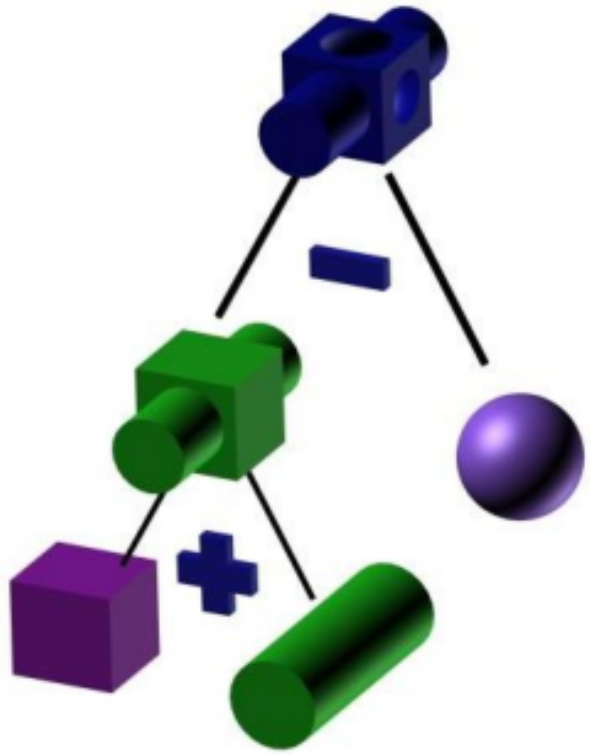
```
For each pixel {
    Shoot a ray from camera to pixel;
    //Perspective

    for all objects in scene
        Compute intersection with ray

    Find object with closest intersection

    Display color using object + light
    property
}
```



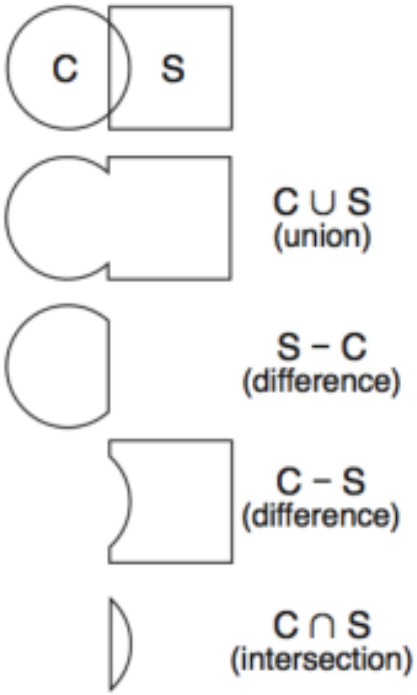
(Ed Angel, NewMexico)

Ray casting for CSG

- CSG: Using set operations for solid shapes



http://escience.anu.edu.au/lecture/cg/surfaceModeling/CSG_tree.en.html

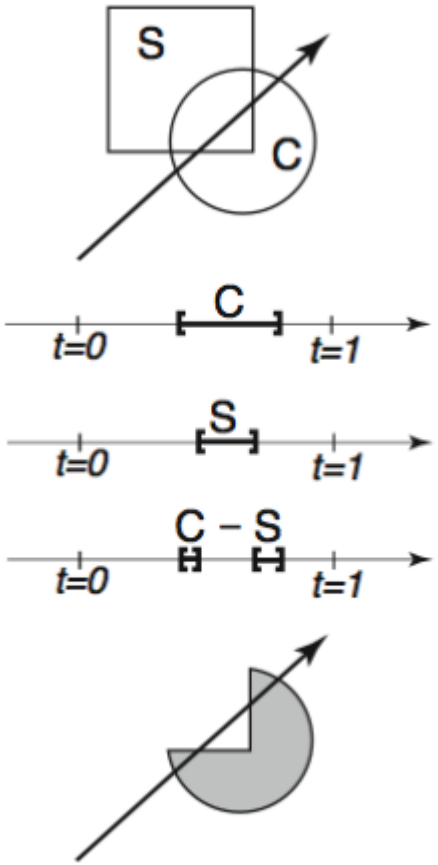


http://www.cs.utah.edu/~shirley/fcg/figures/ray/

Ray casting for CSG

- If only an image is needed do not need to change the models **Boolean operation**

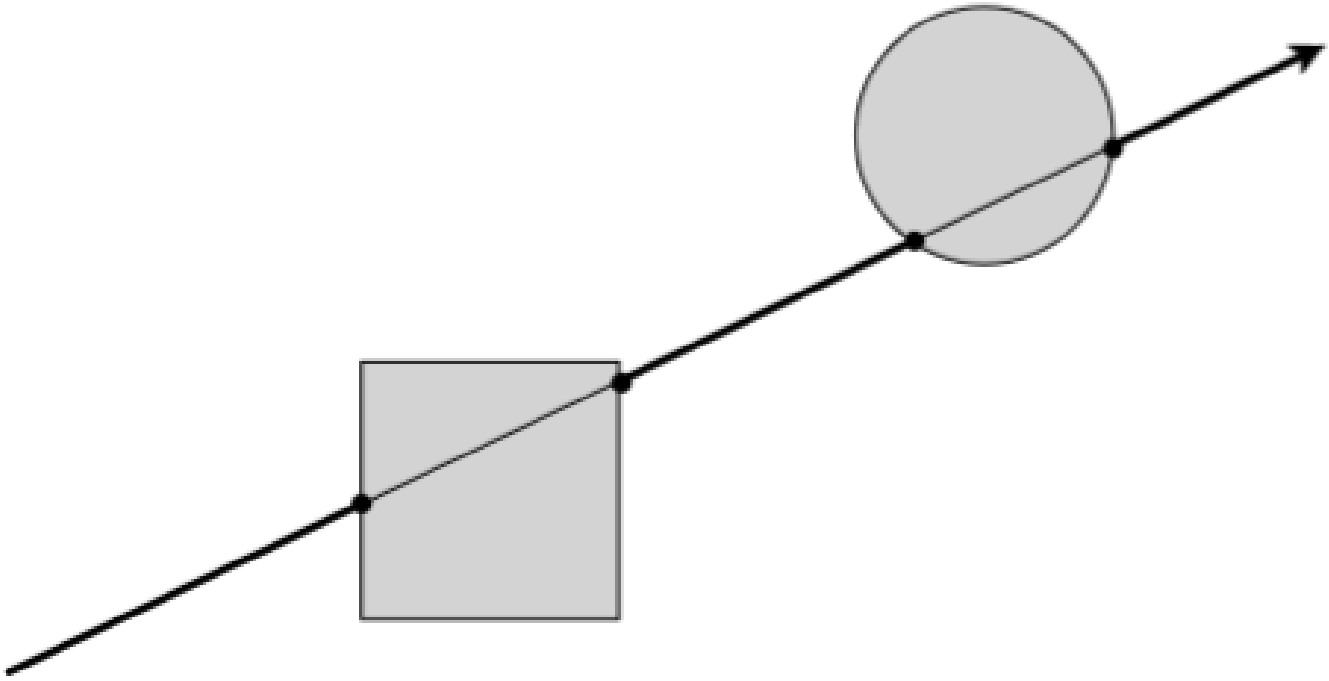
- Ray casting
  - Find all intersections with a model, instead of just the closest.
  - find intervals where a ray is inside the object
  - Do set operations on intervals



http://www.cs.utah.edu/~shirley/fcg/figures/ray/

Intersection and shading

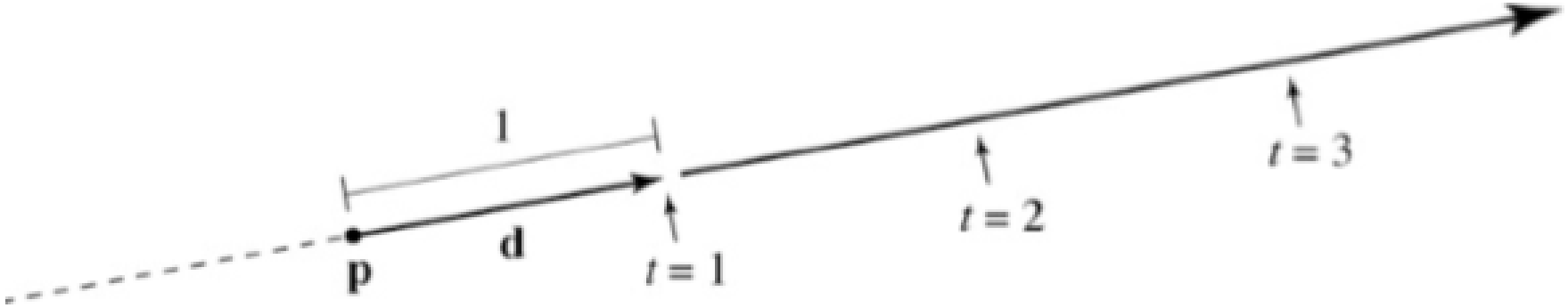
- Ray intersection



(Steve Marschner, MIT)

Ray

- Standard representation: $r(t) = p + td$ where p is the start point and d is the direction and the parameter t >0



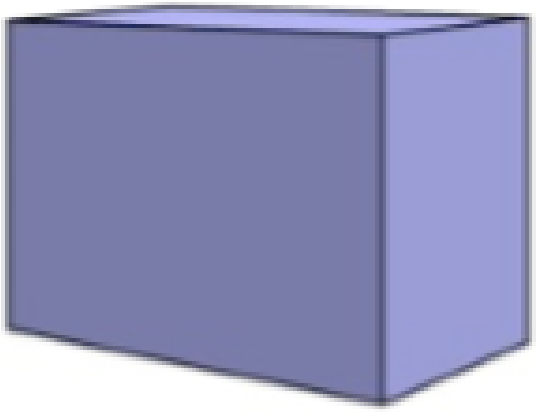
(Steve Marschner, MIT)

Ray-sphere intersection

- Solving algebra:
- $r(t) = p + td$
- $(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2$
- Substitute:
- $(r_x(t) - x_0)^2 + (r_y(t) - y_0)^2 + (r_z(t) - z_0)^2 = R^2$
- Quadratic formula to solve *t*
  - There will be two *t* values

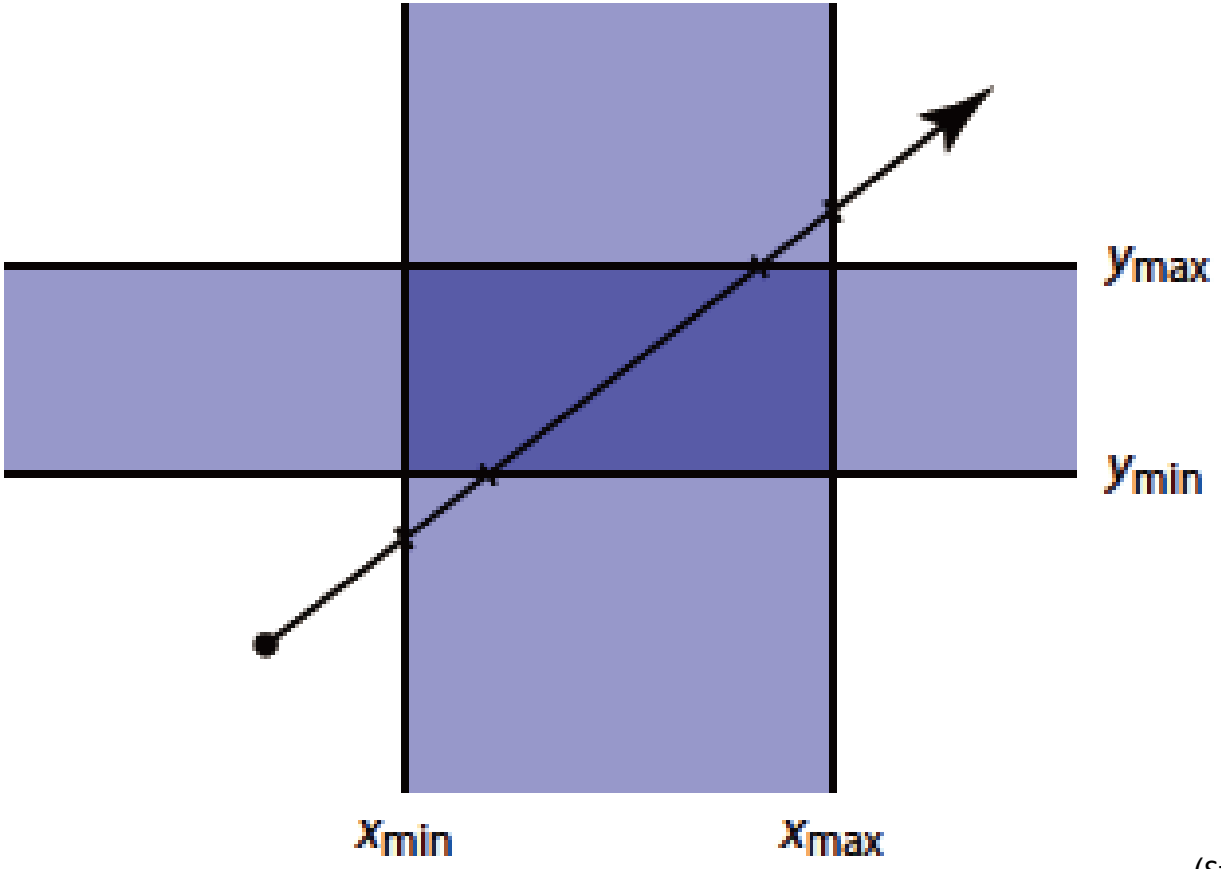
Ray-box intersection

- Intersect with 6 faces
- Again solving algebra
  - Ray equation
  - Plane equation
  - Boundary check
- Or intersect with boundaries



Ray-box intersection

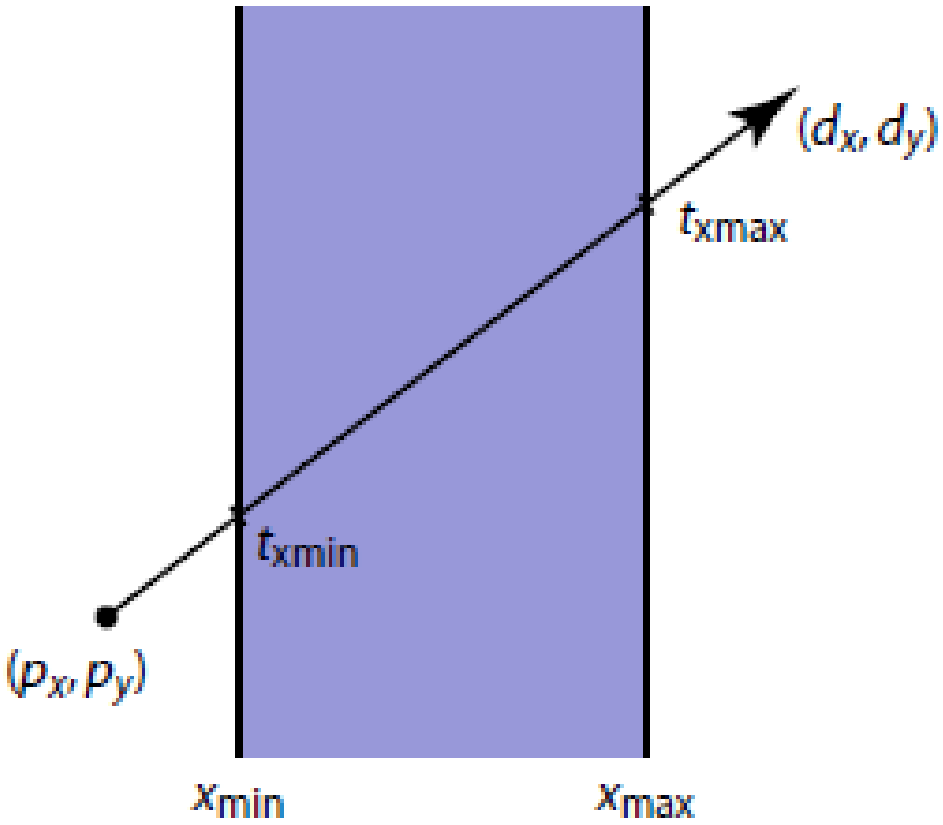
- Similar to clipping algorithm



(Steve Marschner, MIT)

Ray-box intersection

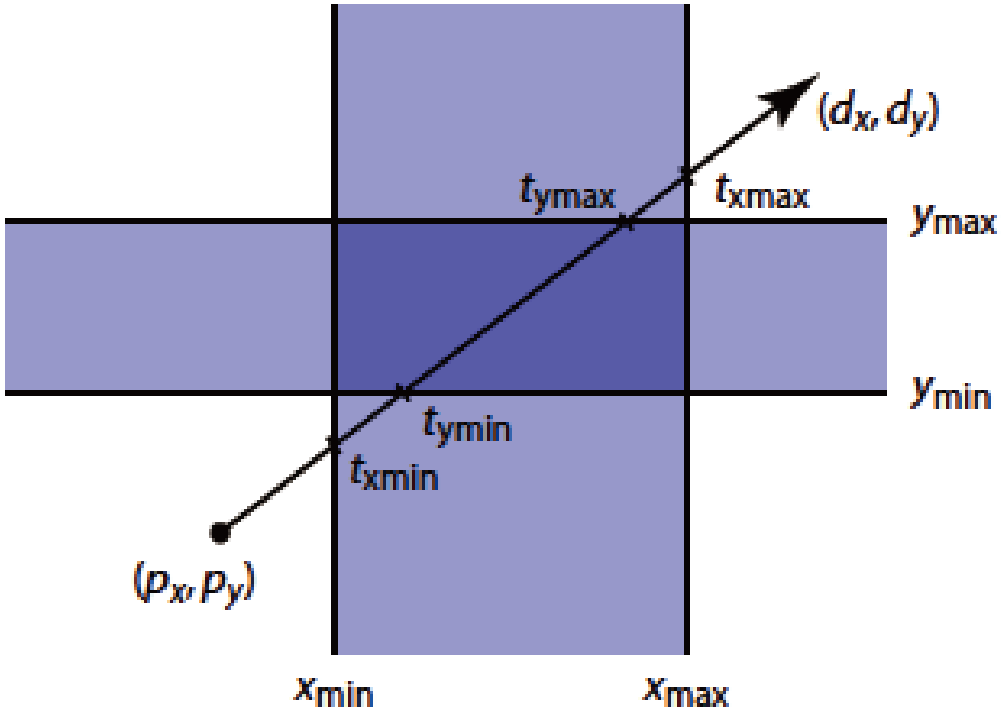
- Similar to clipping algorithm
- $p_x + t_{xmin} d_x = x_{min}$
- $p_x + t_{xmax} d_x = x_{max}$



(Steve Marschner, MIT)

Ray-box intersection

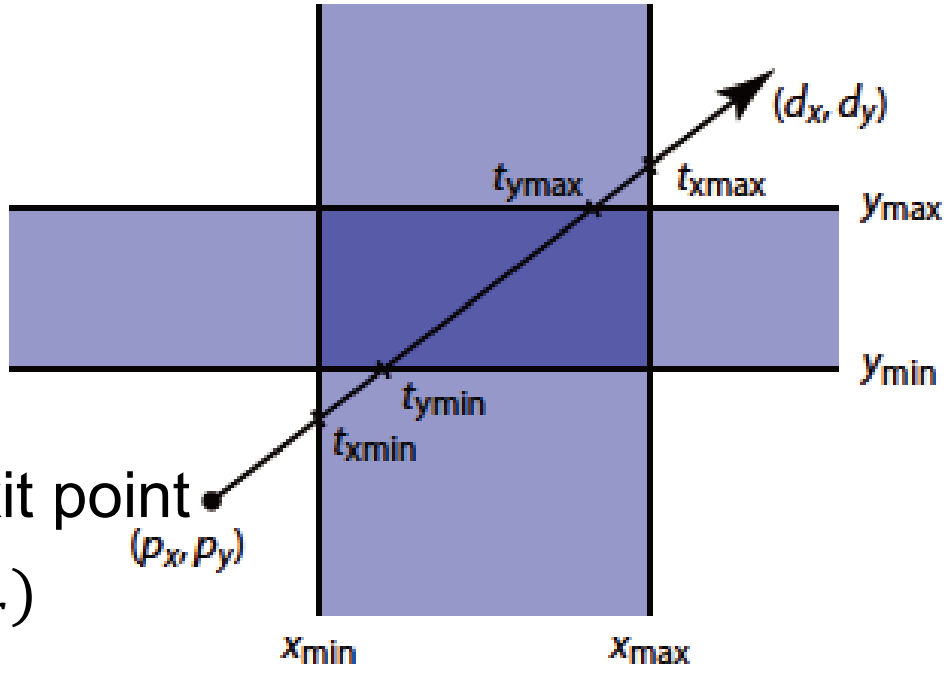
- Similar to clipping algorithm
- $p_x + t_{xmin} d_x = x_{min}$
- $p_x + t_{xmax} d_x = x_{max}$
- $p_y + t_{ymin} d_y = y_{min}$
- $p_y + t_{ymax} d_y = y_{max}$



(Steve Marschner, MIT)

Ray-box intersection

- Find the enter point and exit point
- $t_{xenter} = \min(t_{xmin}, t_{xmax})$
- $t_{yenter} = \min(t_{ymin}, t_{ymax})$
- $t_{xexit} = max(t_{xmin}, t_{xmax})$
- $t_{yexit} = \max(t_{ymin}, t_{ymax})$

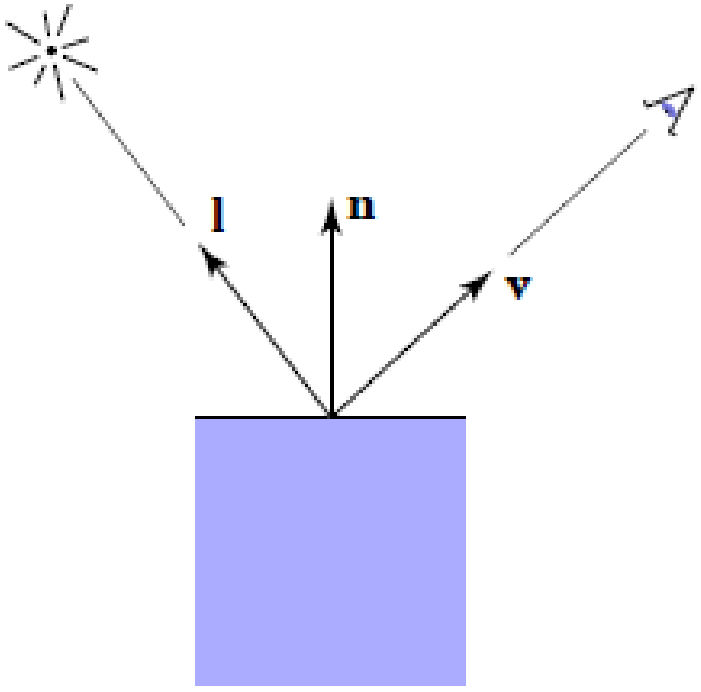


- Last enter point and first exit point
- $t_{enter} = \max(t_{xenter}, t_{yenter})$
- $t_{exit} = \min(t_{xexit}, t_{yexit})$

(Steve Marschner, MIT)

Shading

- Compute illumination
- Inputs:
  - Eye direction
  - Light source directions
  - Surface normal
  - Surface material
- Can be the same as Phong reflection model!
- $I = L_a k_a + L_d k_d \max(0, l \cdot n) + L_s k_s (r \cdot v)^{\alpha}$



(Steve Marschner, MIT)

Real-time ray tracing

https://www.youtube.com/watch?v=aKqxonOrl4Q

Ray tracing acceleration

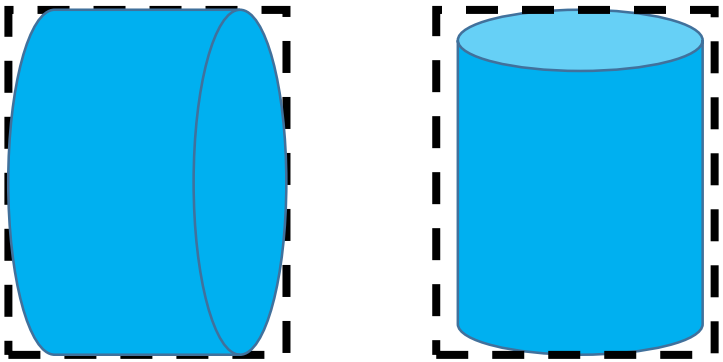
- Ray tracer spends most of the time in ray-surface intersection
- To improve
  - Make intersection more efficient
  - Only do intersection when necessary
  - Trace in parallel
  - …

Intersect when necessary

- Try to avoid meaningless intersections by using auxiliary data structures
  - BVH
  - Octree
  - BSPtree
  - Kdtree
  - Etc.

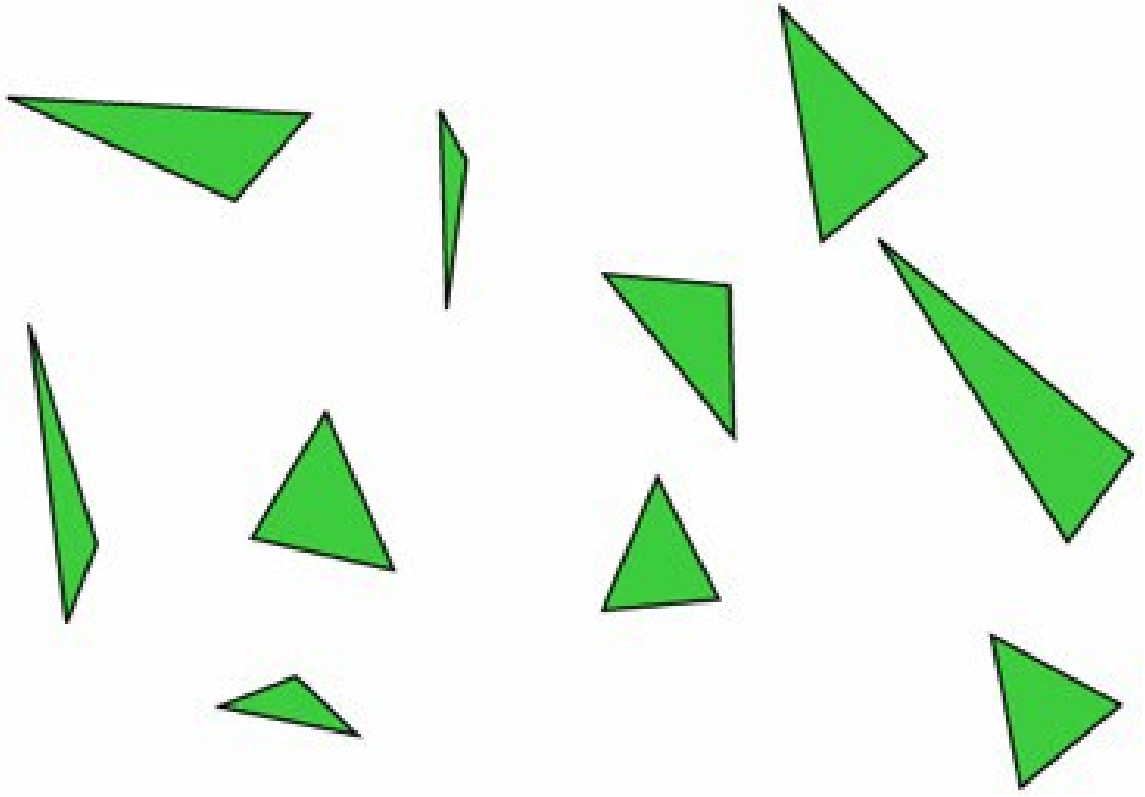
Bounding volume hierarchy (BVH)

- Object is contained within an volume
- First test the ray with the volume, then test the including objects when it hits the volume
- The volume should be simple (AABB boxes in most of time)
- Volume should fit the object tightly



Bounding volume hierarchy (BVH)

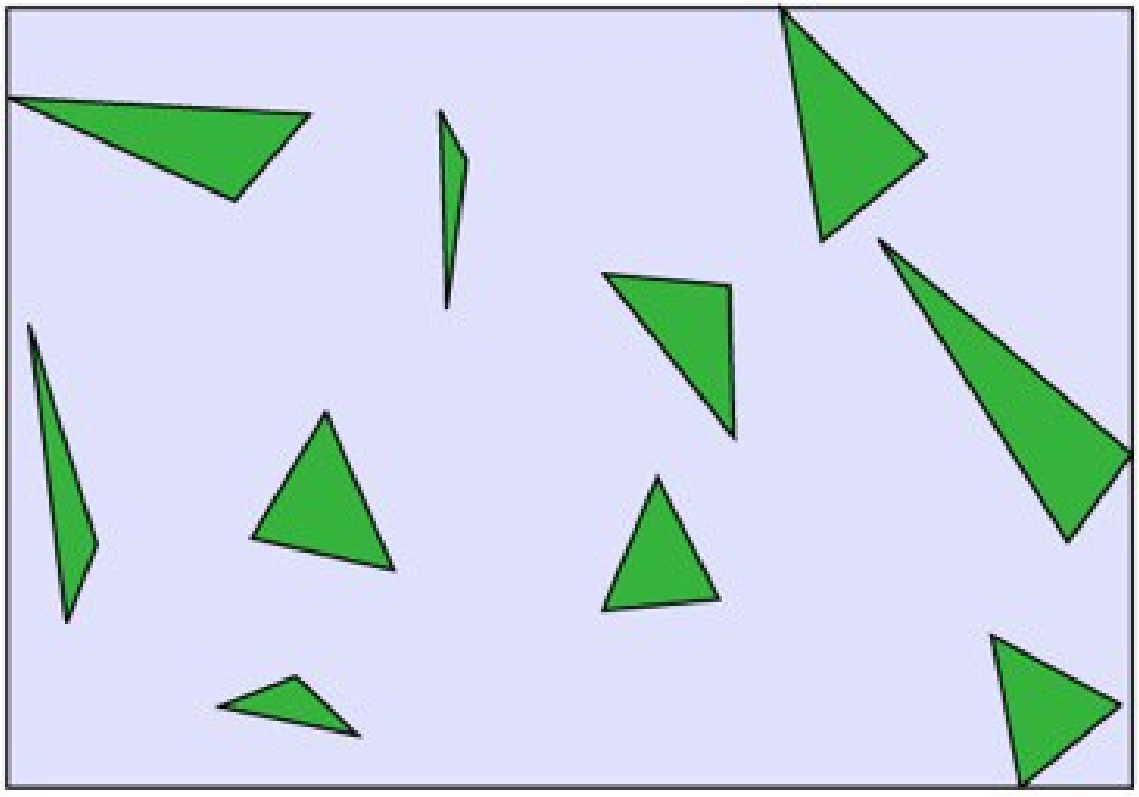
- Hierarchy of bounding volumes
- A tree



(Steve Marschner, MIT)

Bounding volume hierarchy (BVH)

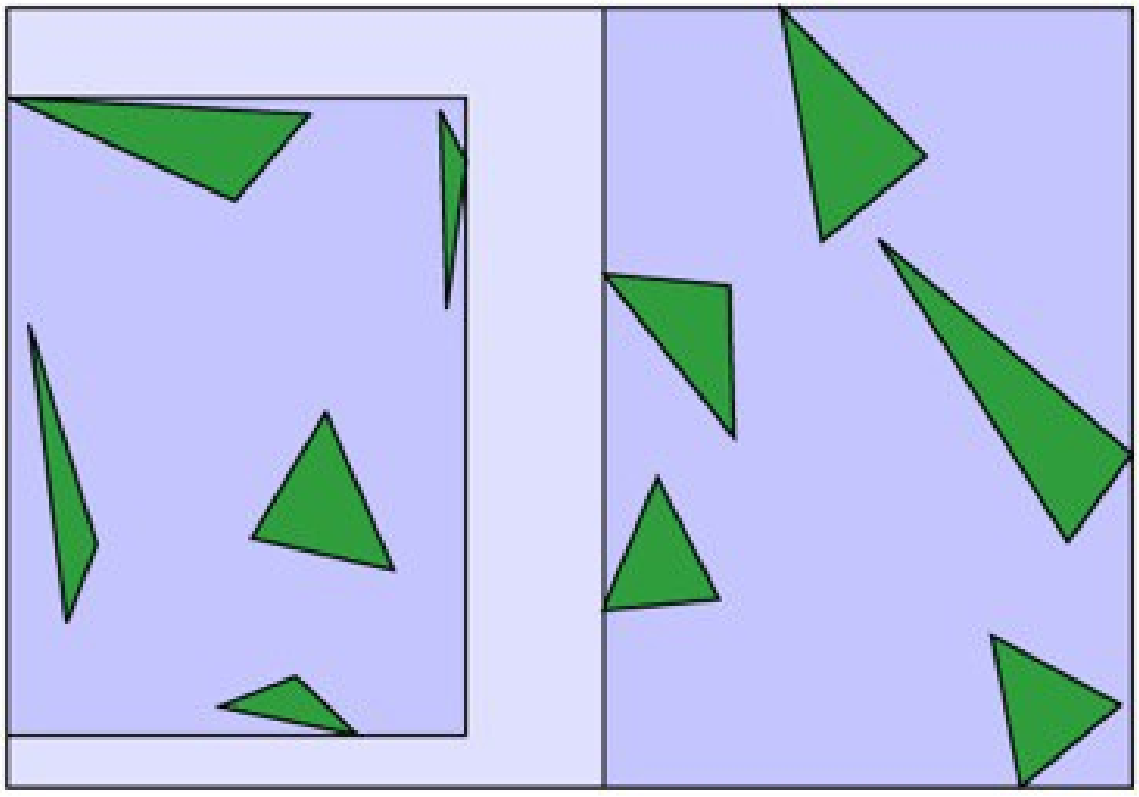
- Hierarchy of bounding volumes
- A tree



(Steve Marschner, MIT)

Bounding volume hierarchy (BVH)

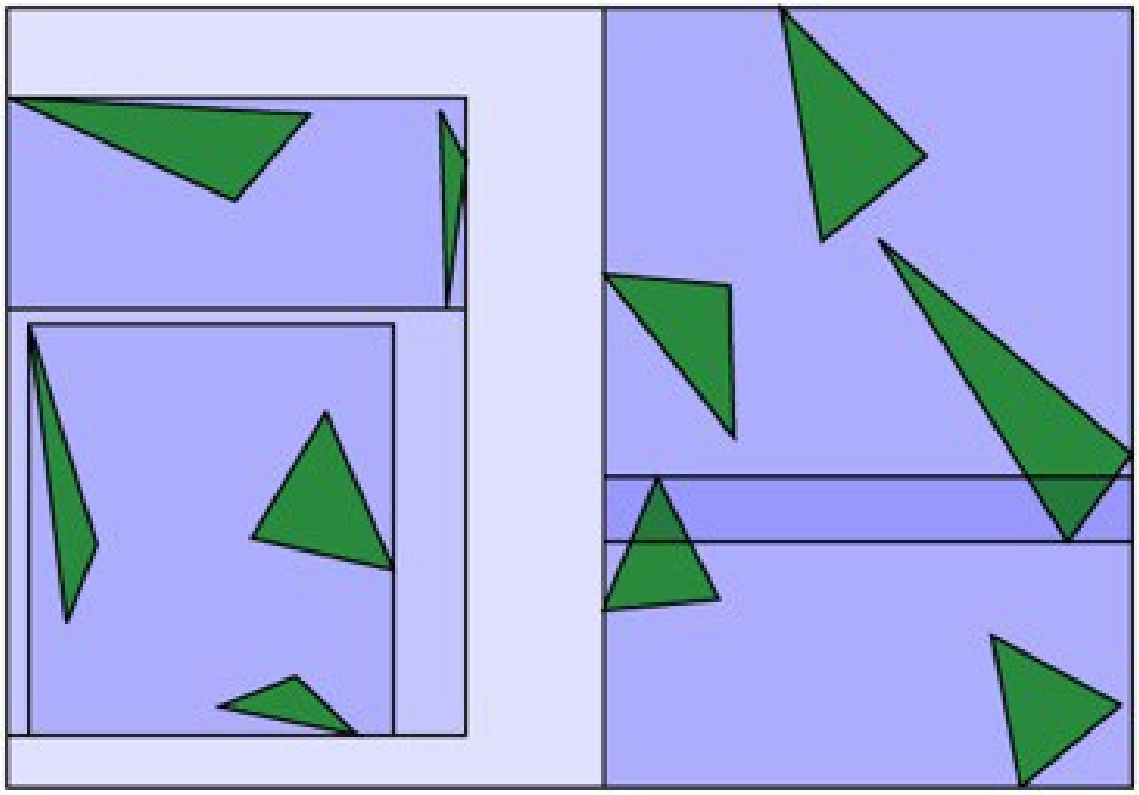
- Hierarchy of bounding volumes
- A tree



(Steve Marschner, MIT)

Bounding volume hierarchy (BVH)

- Hierarchy of bounding volumes
- A tree



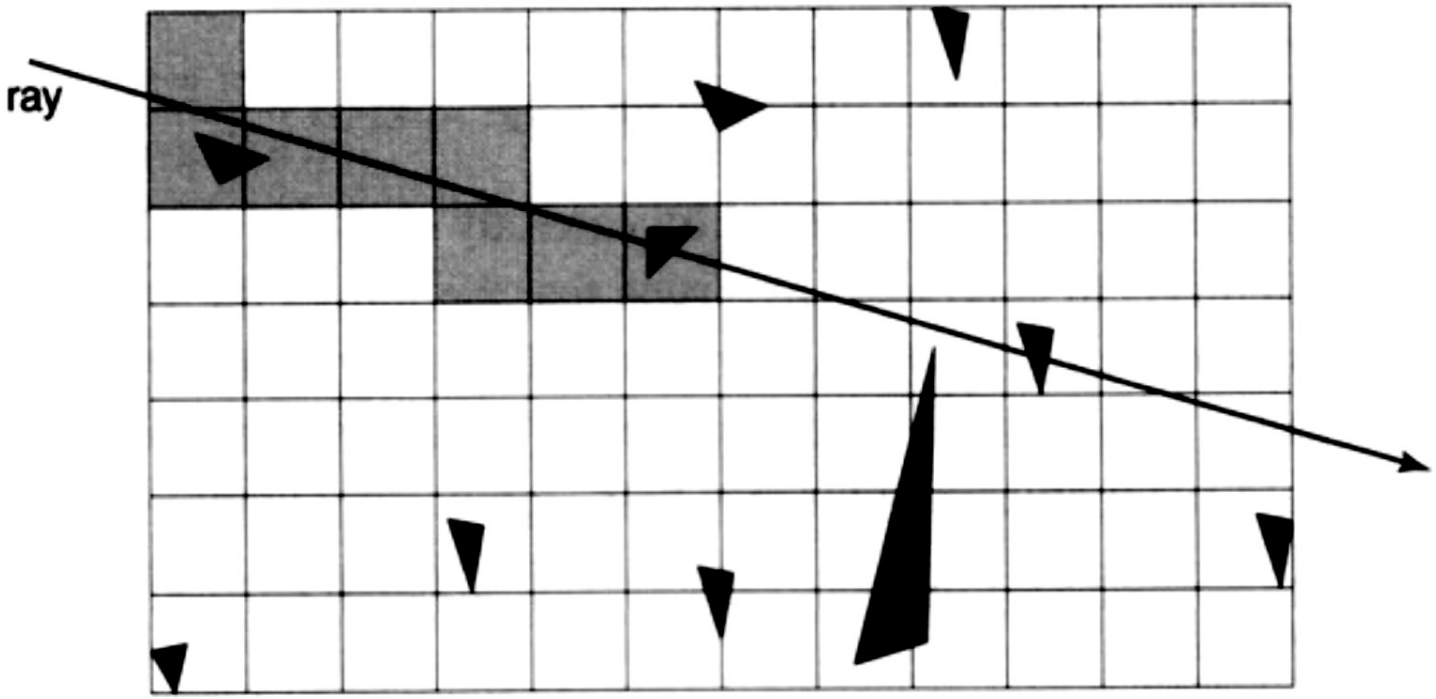
(Steve Marschner, MIT)

Bounding volume hierarchy (BVH)

- Building a BVH
- Partition smartly
- Balance the tree will generally improve the efficiency
- Expectation of the intersection cost

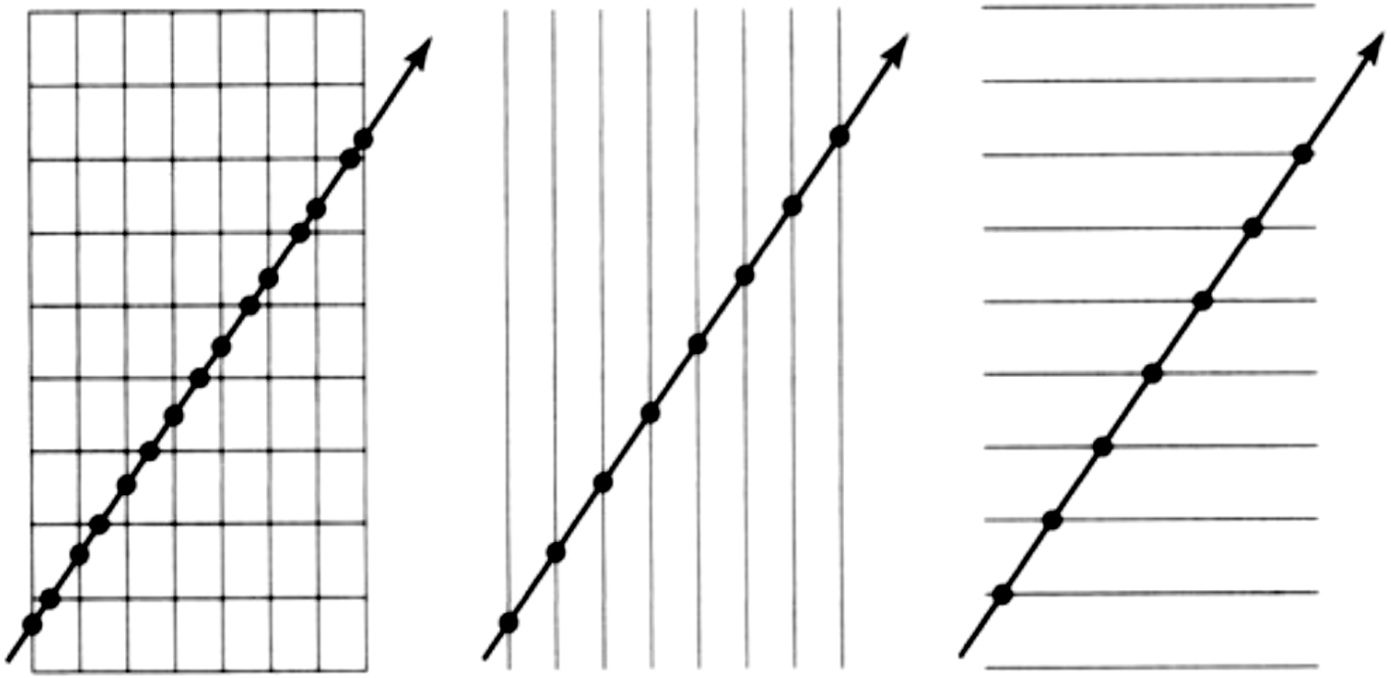
Octree

- Analogy to Grids in 3D
- Not group objects, but divide space, how?



Octree

- Traverse the grid using enter point and exit point



Ray Tracing in One Weekend

- *Peter Shirley* https://raytracing.github.io/books/RayTracingInOneWeekend.html

References

- Steve Marschner, CS4620/5620 Computer Graphics, Cornell
- Ed Angel, CS/EECE 433 Computer Graphics, University of New Mexico
- Elif Tosun, Computer Graphics, The University of New York

- Questions?