CS559: Computer Graphics

Lecture 36: Raytracing

Li Zhang

Spring 2008

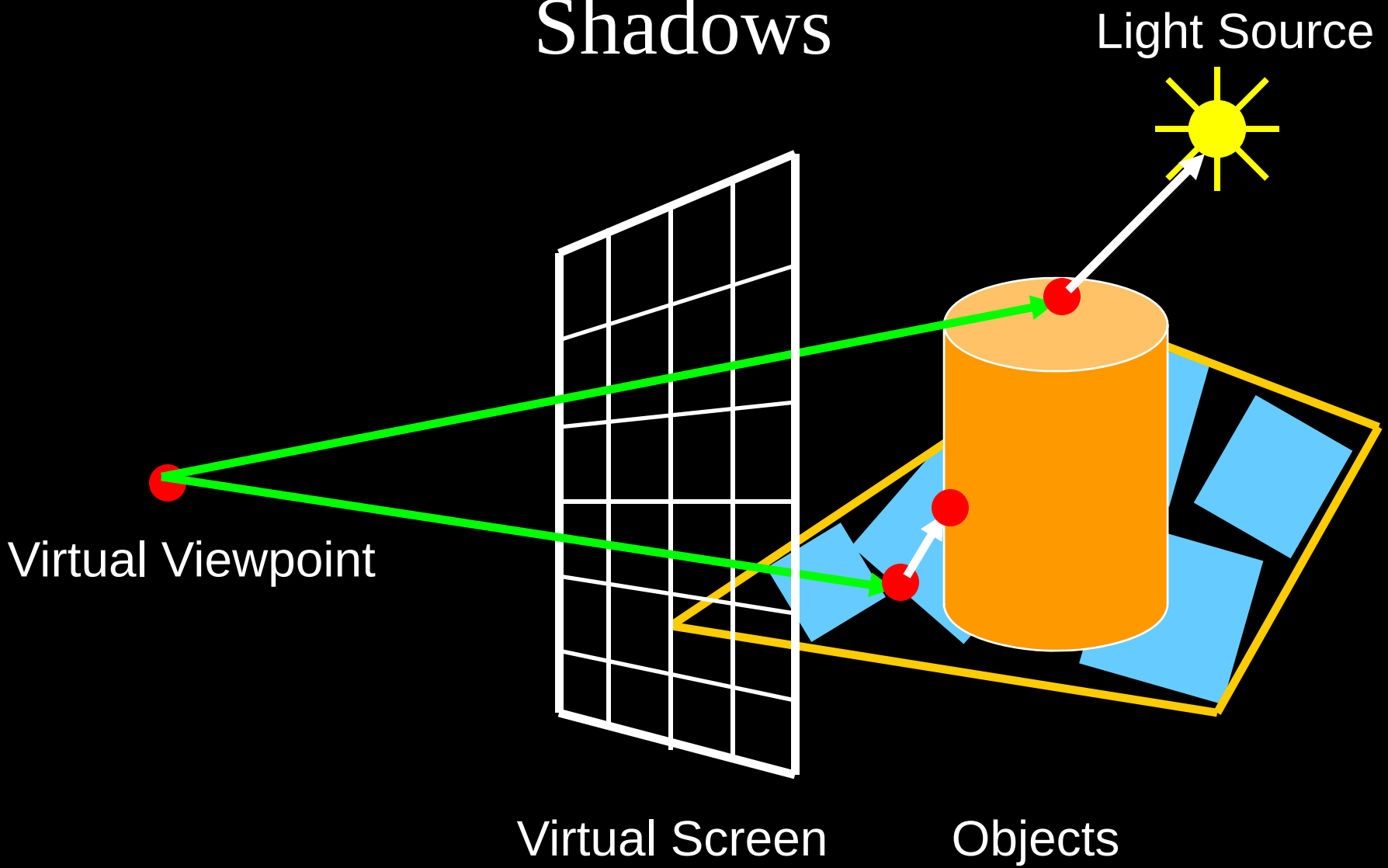
Many Slides are from Ravi Ramamoorthi at Columbia U, and Ronen Barzel at UCSD

Today

- ray tracing, image based rendering

- Reading
  - Shirley Ch 10 on ray tracing, except for ch 10.10
  - Shirley Ch 25 on image based rendering
  - (Optional) Levoy and Hanrahan, Light Field Rendering, SIGGRAPH 1996, http://portal.acm.org/citation.cfm?id=237199

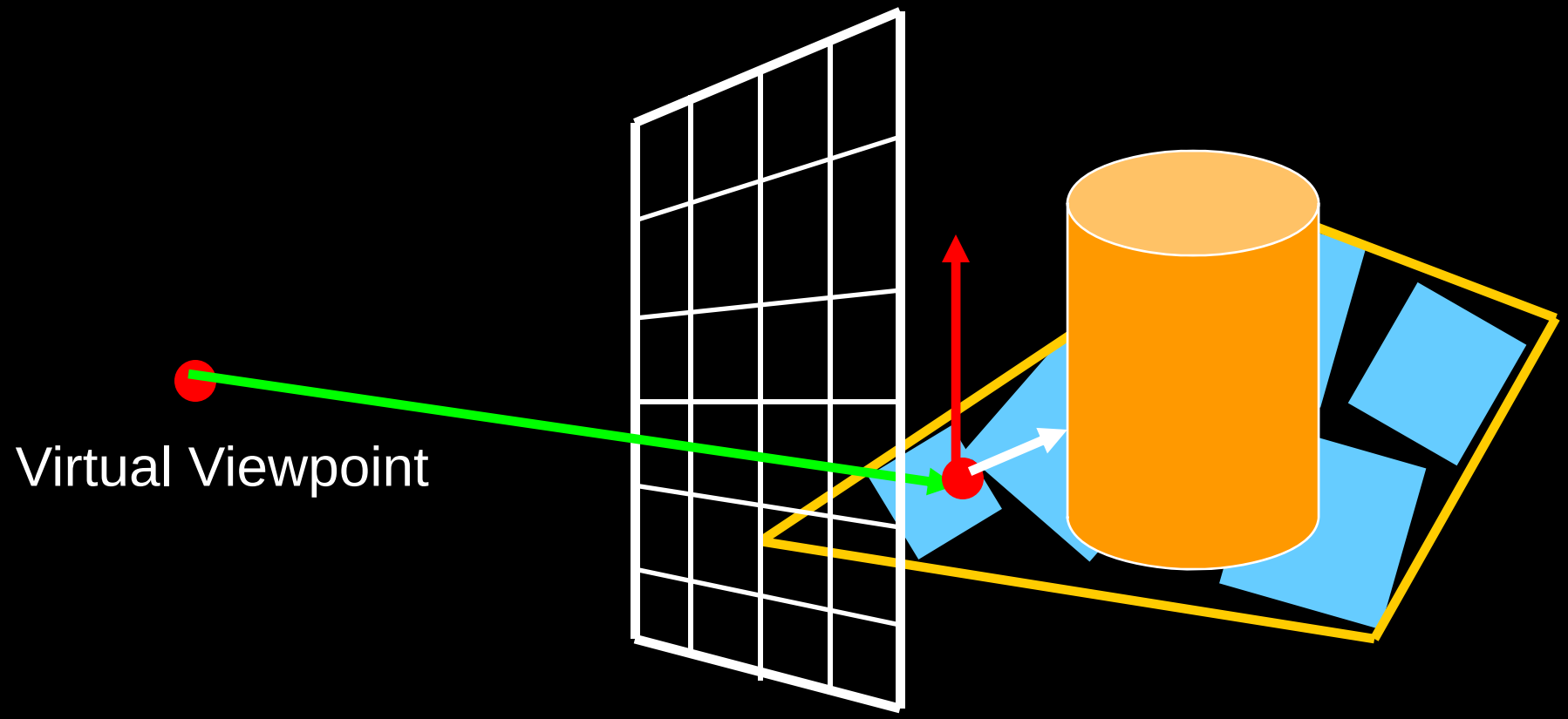
Shadows



Shadow ray to light is blocked by object → shadow

10.5 in textbook

Mirror Reflections/Refractions



Generate reflected ray in mirror direction,
Get reflections and refractions of objects

10.6 in textbook

Recursive Ray Tracing (Core Idea)

For each pixel

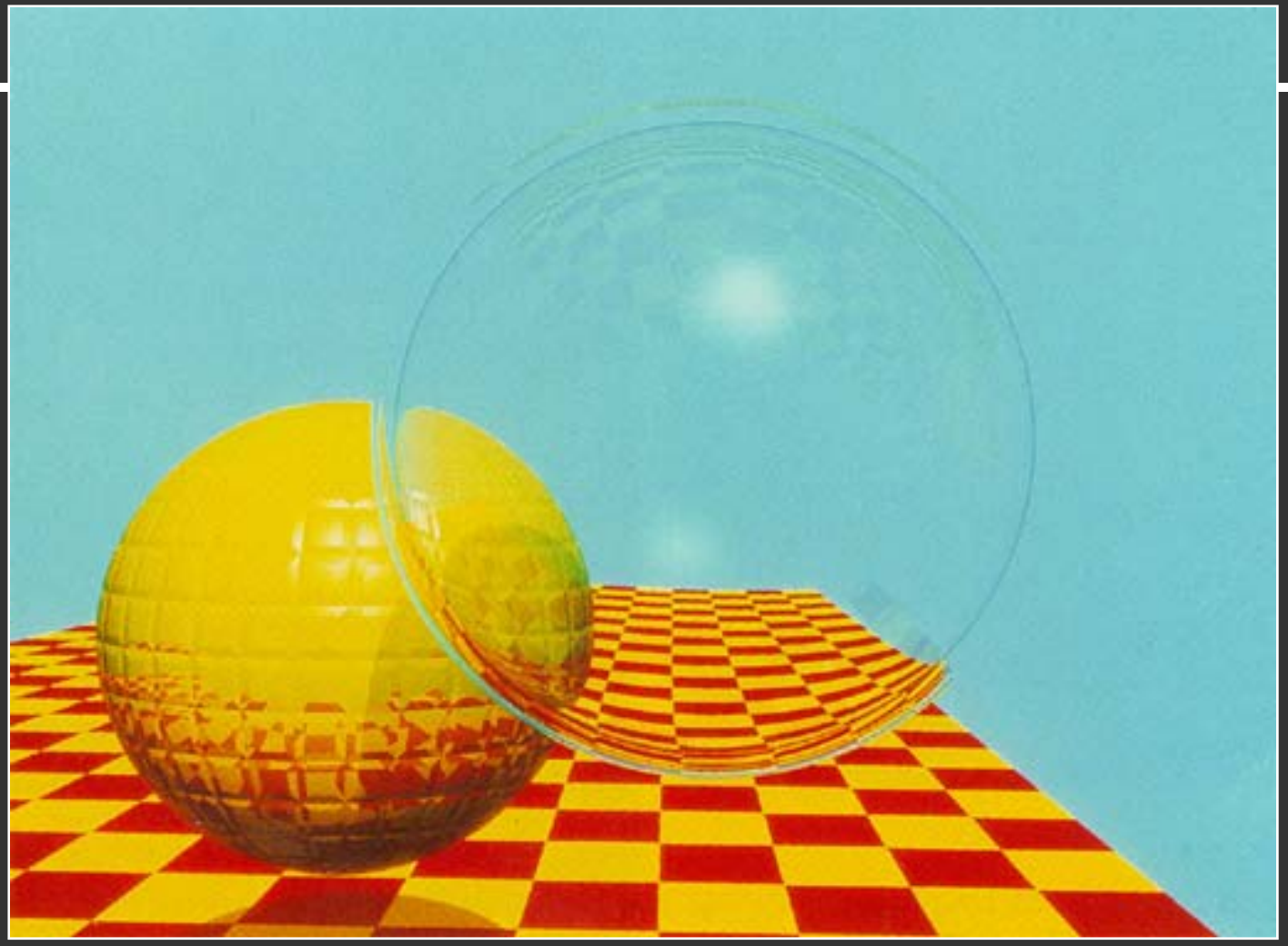
- Trace Primary Eye Ray, find intersection
- Trace Secondary Shadow Ray(s) to all light(s)
  - Color = Visible1 ? Illumination Model(light1) : 0 ;
  - Color += Visible2 ? Illumination Model(light2) : 0 ;
  - …
- Trace Reflected Ray
  - Color += reflectivity * Color of reflected ray
- Trace Refracted Ray
  - Color += transparency * Color of refracted ray

Recursive function Calls

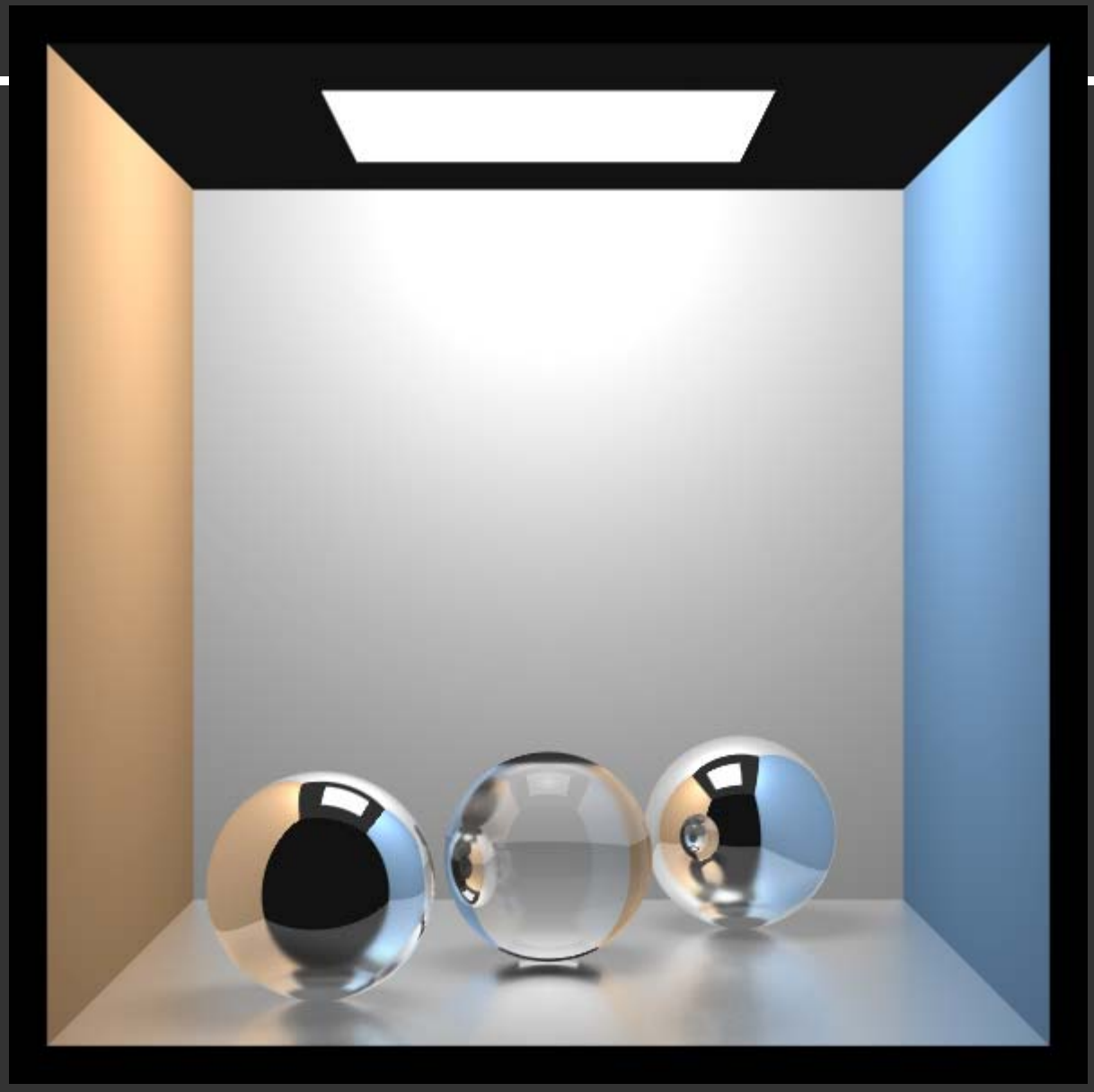
Also see section 10.4 in text

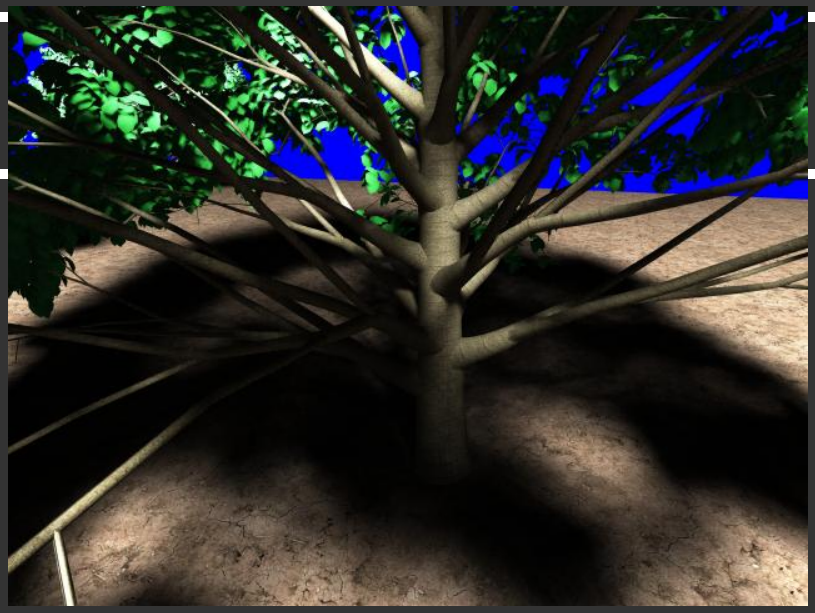
Problems with Recursion

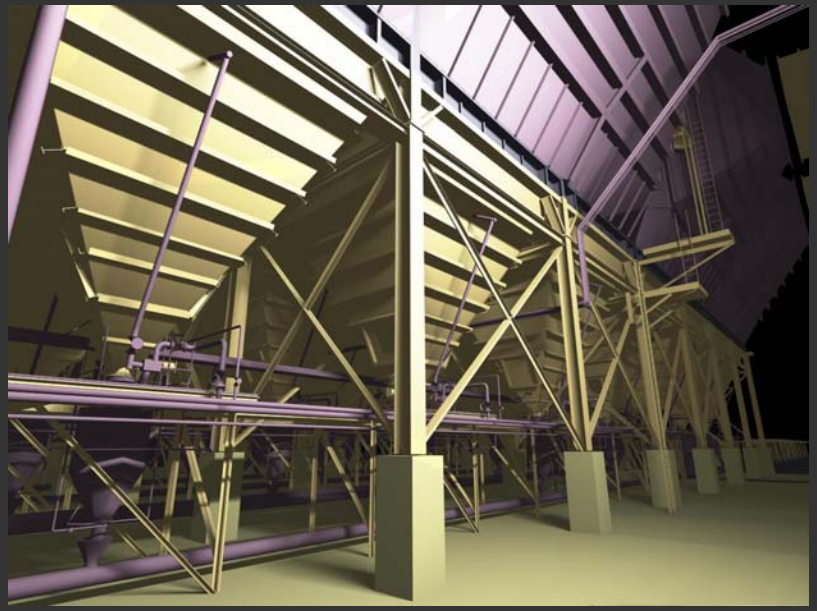
- Reflection rays may be traced forever
- Generally, set maximum recursion depth



Turner Whitted 1980







Effects needed for Realism

- (Soft) Shadows
- Reflections (Mirrors and Glossy)
- Transparency (Water, Glass)
- Interreflections (Color Bleeding)
- Complex Illumination (Natural, Area Light)
- Realistic Materials (Velvet, Paints, Glass)

Discussed in this lecture

Not discussed so far but possible with distribution ray tracing (10.11)

Hard (but not impossible) with ray tracing; radiosity methods

How to implement Ray tracing?

- Ray parameterization
- *Ray-Surface Intersection*

Ray/Object Intersections

- Heart of Ray Tracer
  - One of the main initial research areas
  - Optimized routines for wide variety of primitives
- Various types of info
  - Shadow rays: Intersection/No Intersection
  - Primary rays: Point of intersection, material, normals, Texture coordinates

Example

- Sphere
  - How to decide there is an intersection?
- Triangle
  - How to decide the intersection is inside?
- Polygon
  - How to decide the intersection is inside?

- How about an ellipsoid?

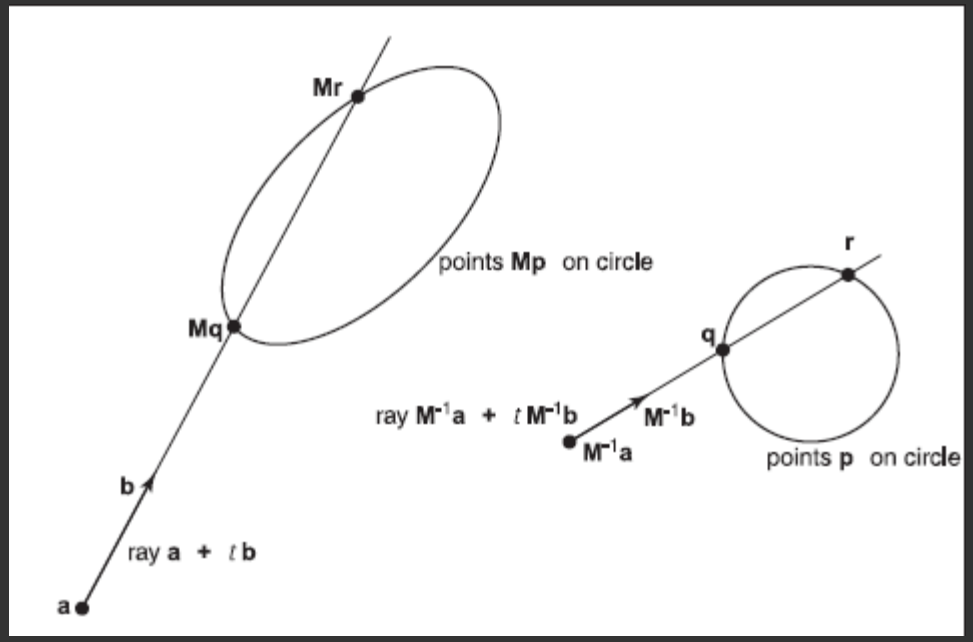
Ray-Tracing Transformed Objects

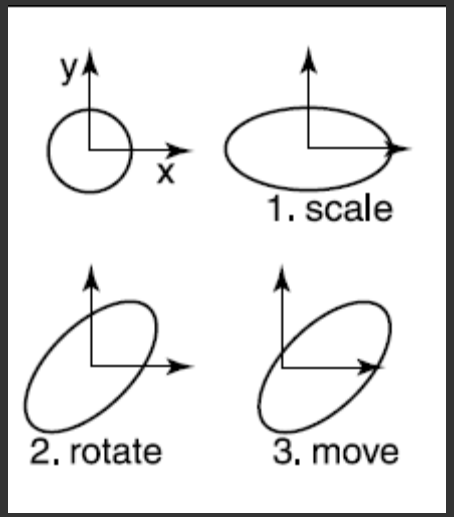
We have an optimized ray-sphere test

- But we want to ray trace an ellipsoid…

Solution: Ellipsoid transforms sphere

- Apply inverse transform to ray, use ray-sphere





Section 10.8 of text

Acceleration

Testing each object for each ray is slow

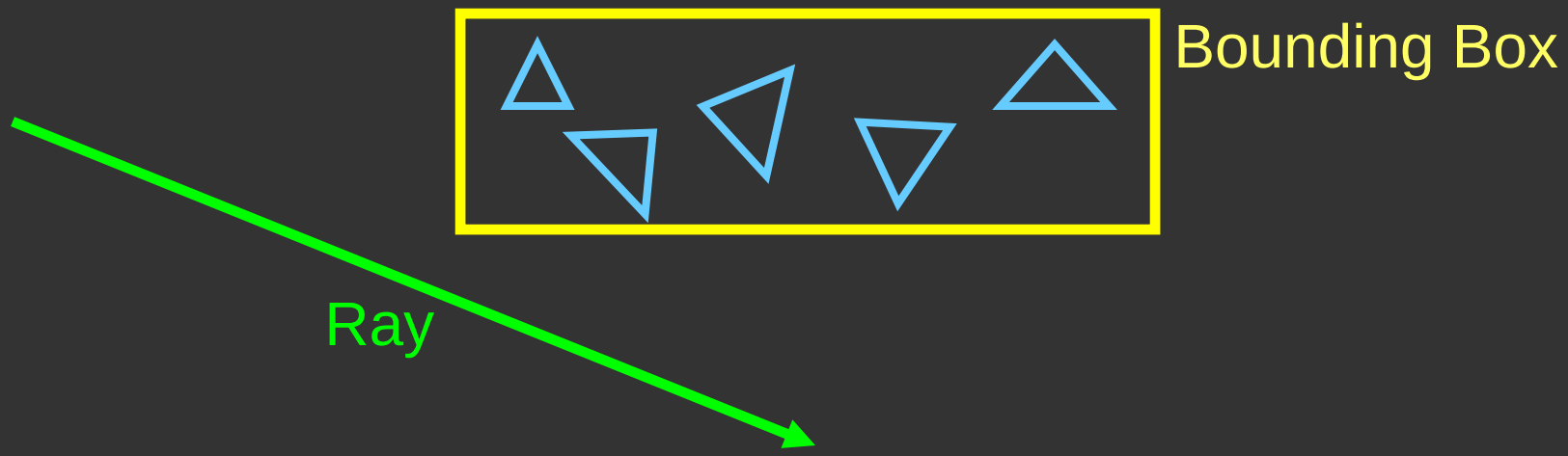
- Faster Intersections
  - Optimized Ray-Object Intersections
  - ***Fewer Intersections***

Section 10.9 goes into more detail

Acceleration Structures

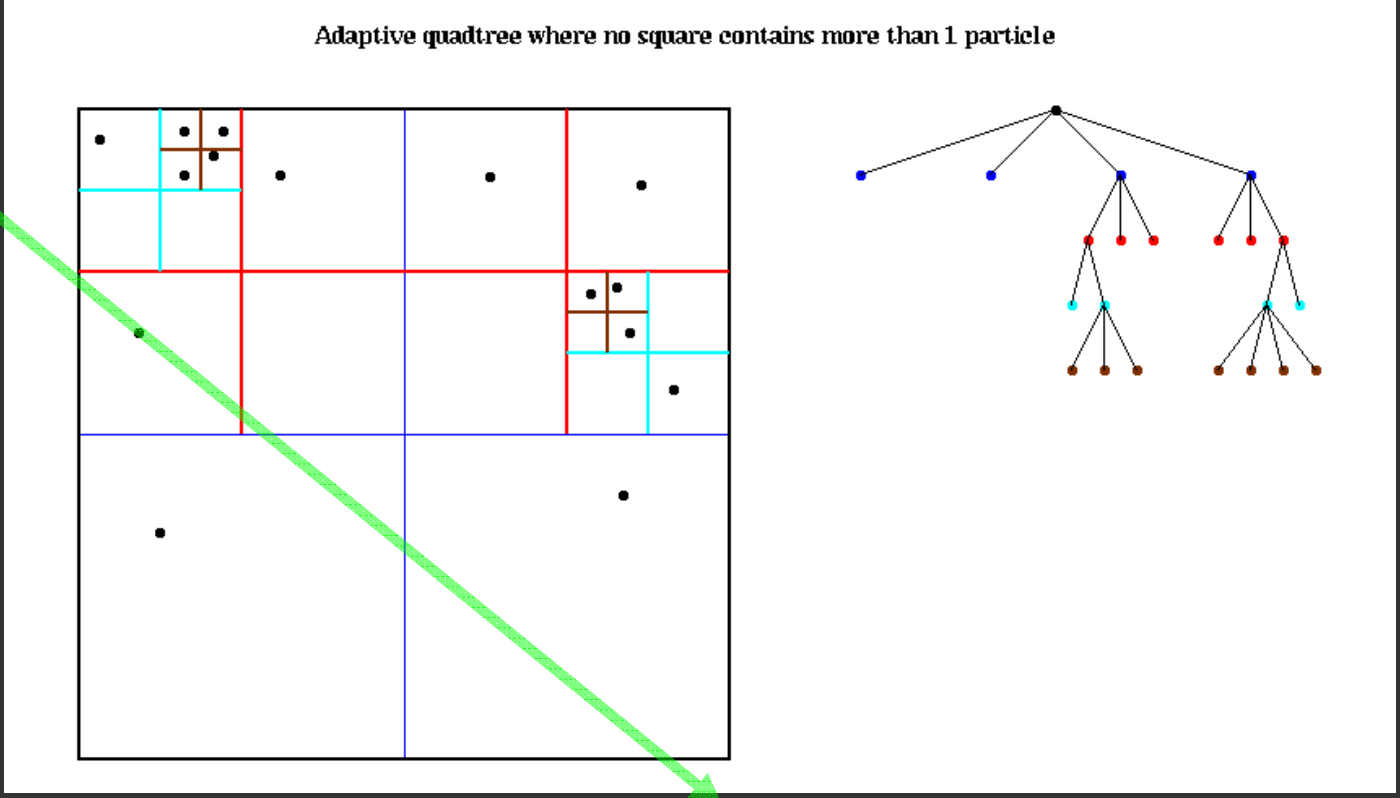
Bounding boxes (possibly hierarchical)

If no intersection bounding box, needn't check objects

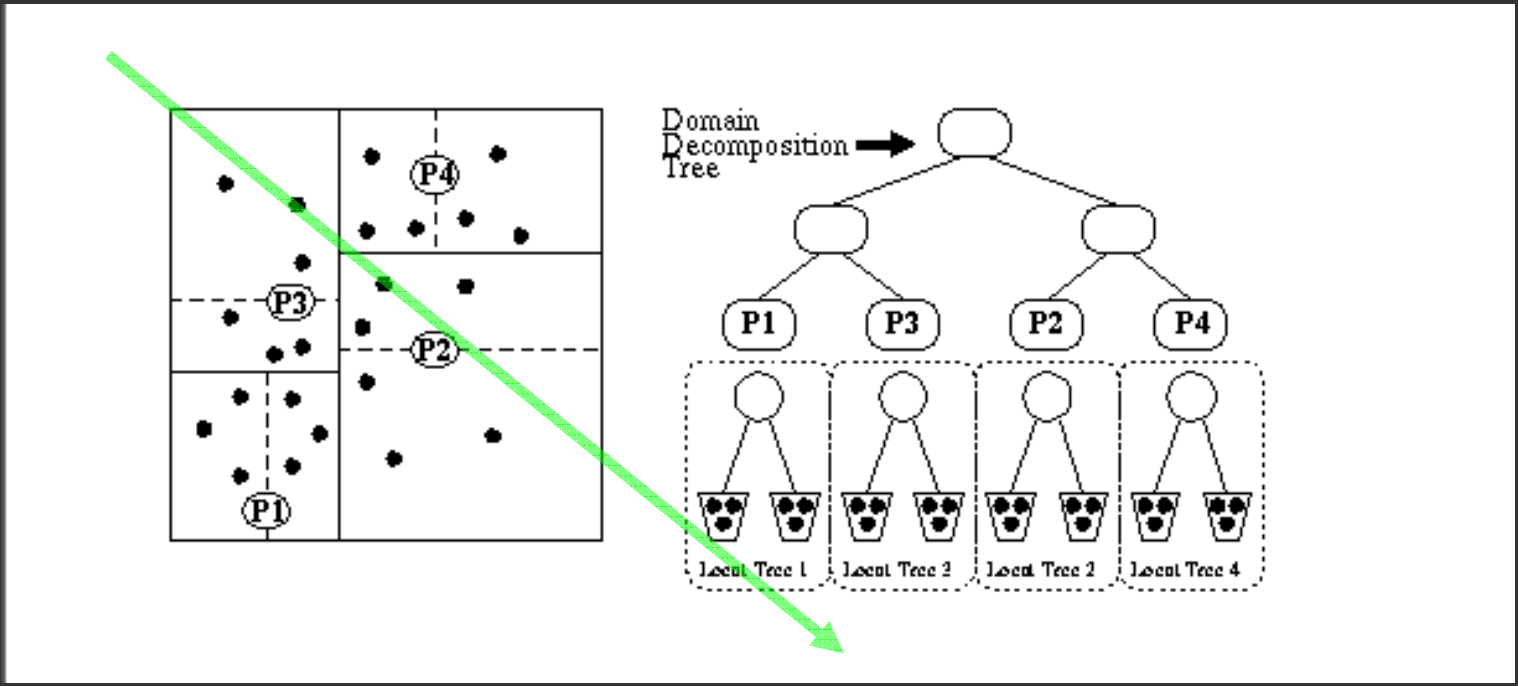


Different Spatial Hierarchies (Oct-trees, kd trees, BSP trees)

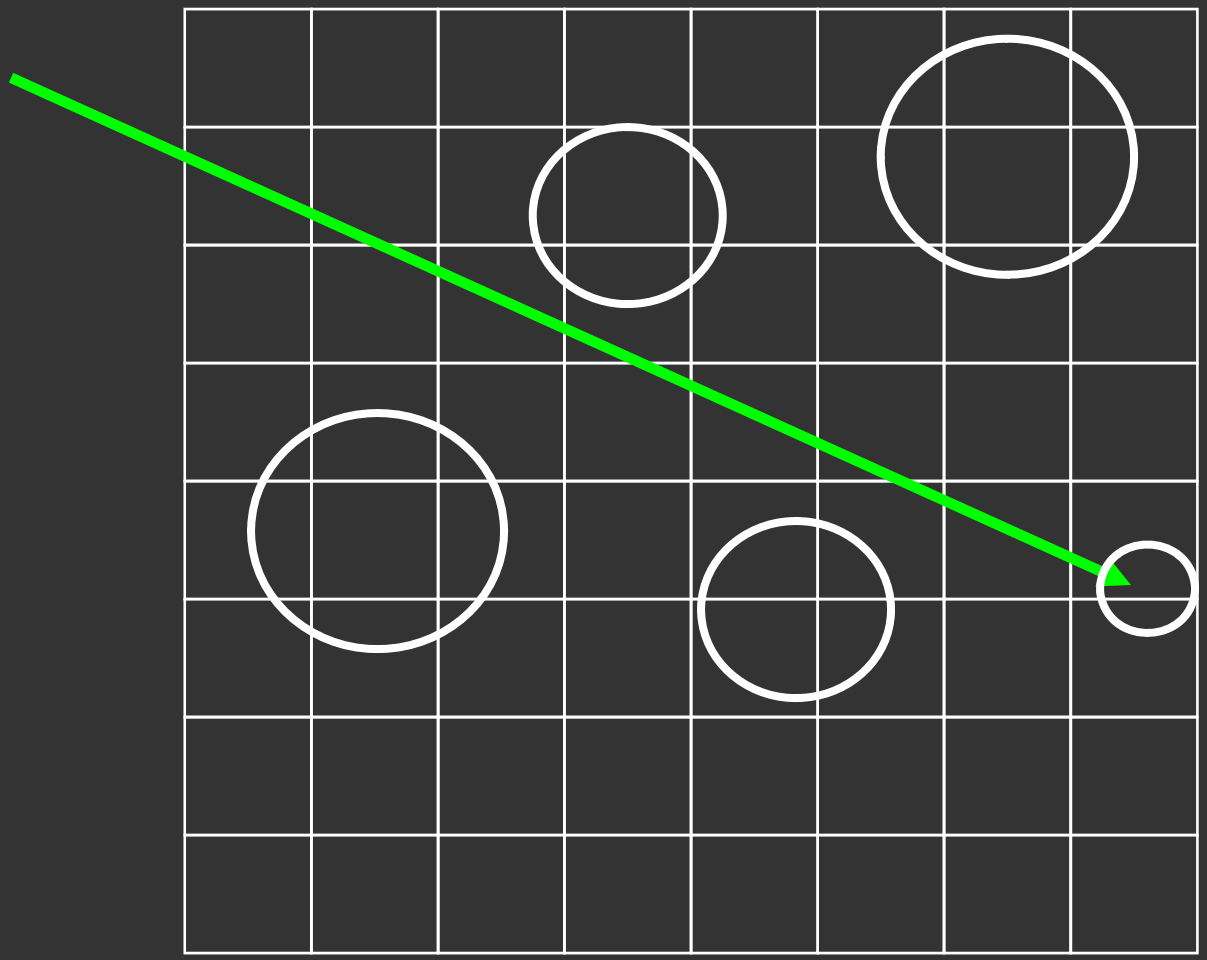
Octree


Adaptive quadtree where no square contains more than 1 particle

K-d tree

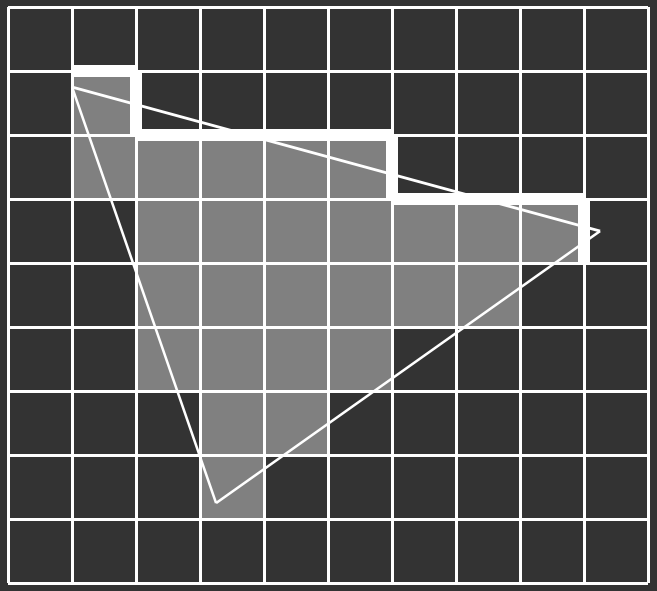

P4
P3
P2
P1
Domain
Decomposition
Tree
P1
P3
P2
P4
Local Tree 1
Local Tree 2
Local Tree 3
Local Tree 4

Acceleration Structures: Grids



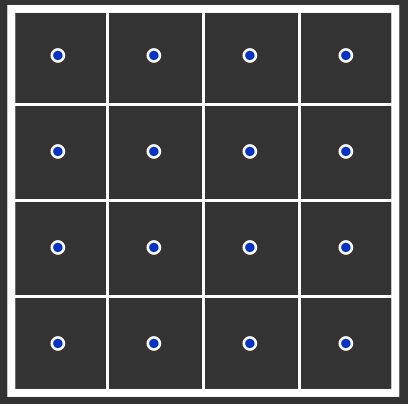
Anti-aliasing

- Aliasing when drawing a diagonal on a square grid:
  - *stairstepping*
  - AKA *jaggies*
- Especially noticeable:
  - high-contrast edges
  - near horizontal or near vertical
  - As line rotates (in 2D)
    - steps change length
    - corners of steps slide along the edge
    - known as *crawlies*



Supersampling

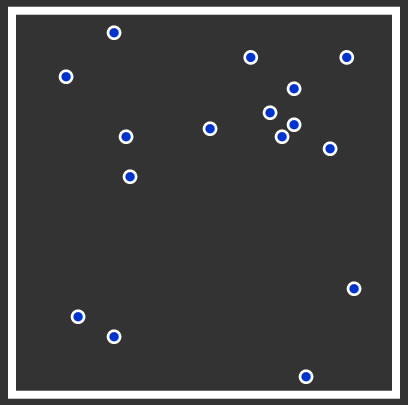
- A more popular method (although less elegant) is *supersampling:*
  - Point sample the pixel at several locations
  - Combine the results into the final pixel color
- By sampling more times per pixel:
  - Raises the sampling rate
  - Raises the frequencies we can capture
- Commonly use 16 or more samples per pixel
  - Requires potentially 16 times as much work to generate image
  - 16 times Memory?
- A brute-force approach
  - But straightforward to implement
  - Very powerful



Moiré Artifact

Random Sampling

- Supersample at several randomly located points
- Breaks up repeating signals
  - Eliminates Moiré patterns
  - Instead of aliasing, frequencies greater than 1 pixel appear as *noise* in the image
- Noise tends to be less objectionable to the viewer than jaggies or Moiré patterns
  - The human eye is pretty good at filtering out noise
- But suffers from potential clustering and gaps
  - Result is not necessarily accurate
  - Too much noise.



Jittered Sampling

- AKA *stratified sampling*,
- Divide pixel into a grid of *subpixels*
  - Sample each subpixel at a random location
- Combines the advantages of both uniform and random sampling
  - filters high frequencies
  - frequencies greater than subpixel sampling rate turned into noise
- Commonly used