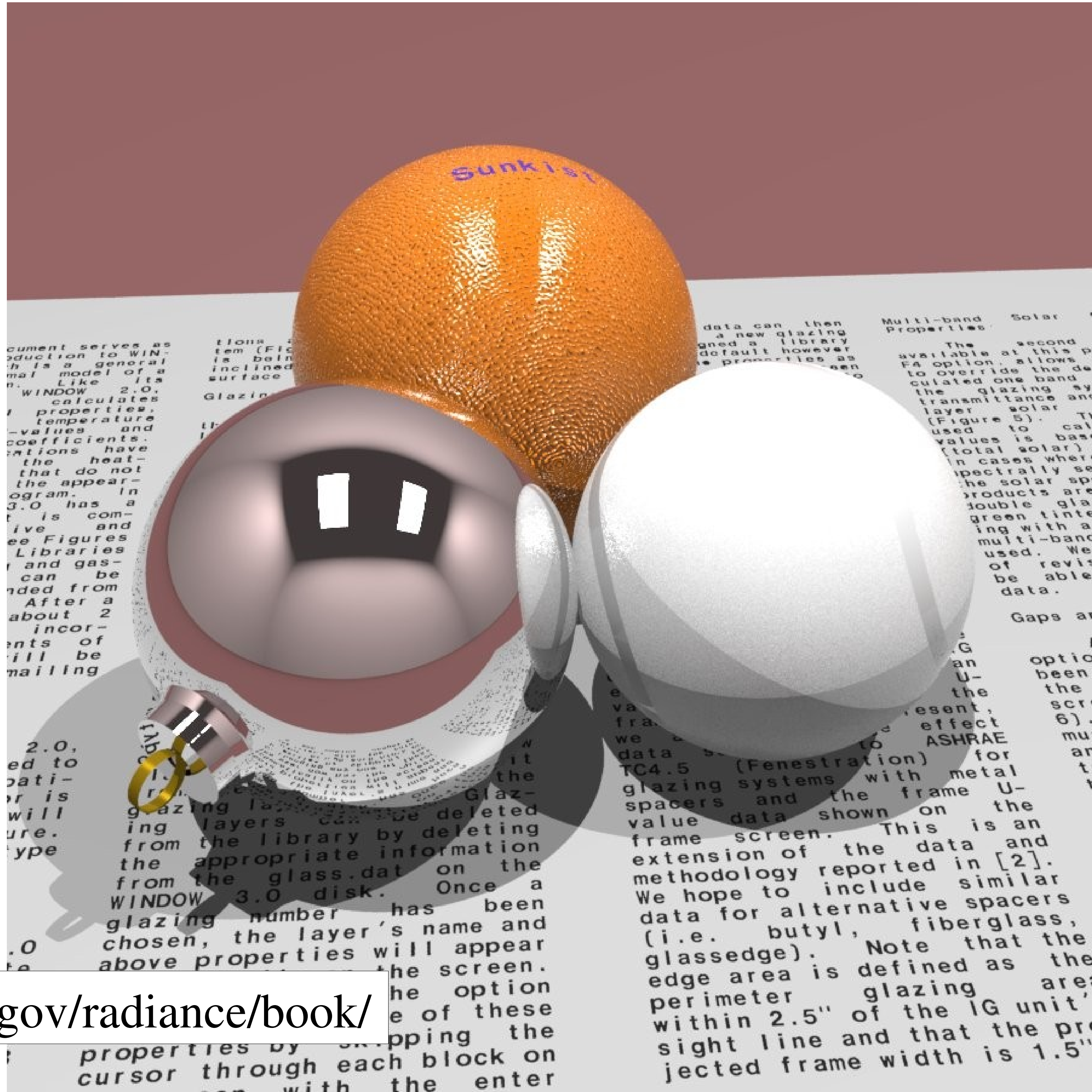
Ray tracing

EECS 487

November 20, 2006



http://radsite.lbl.gov/radiance/book/

Conventional pipeline (rasterization)

- For each triangle
  - Compute lighting at vertices
  - For each pixel within triangle
    - Compute interpolated color and depth
    - Write pixel if depth test passes

- Q: the above description is somewhat “old style”
  - how have things changed lately?

Advantages of conventional pipeline

- Simple
- Can be implemented in HW
  - Parallel processing (SIMD)
  - Vertices
  - Pixels
- Visibility determination is fast
  - z-buffer

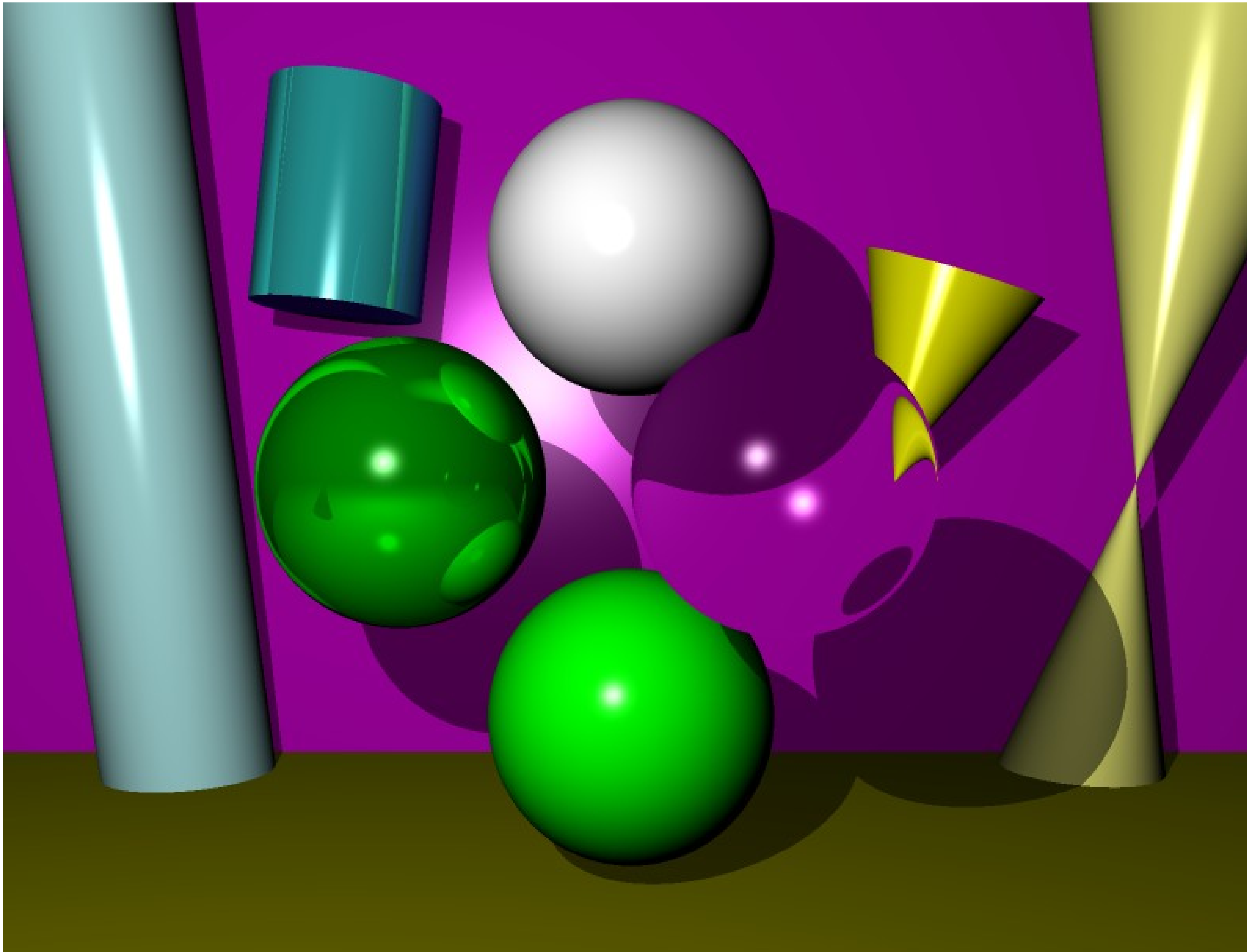
Disadvantages

- Missing effects
  - namely?

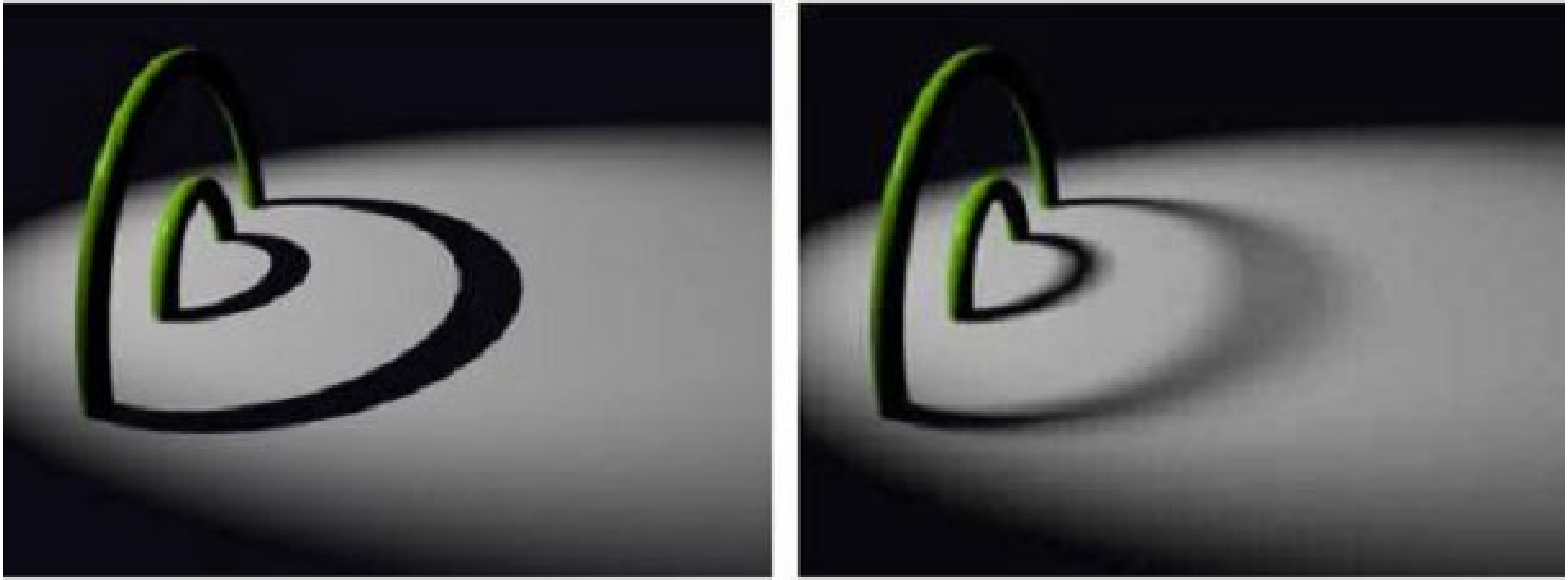
Disadvantages

- Missing effects
  - Shadows
  - Reflection
  - color bleeding
  - Depth of field
  - Motion blur
  - Aliasing

Refraction, hard shadows, reflection



Soft shadows



http://www-csl.csres.utexas.edu/users/billmark/teach/cs384g-05-fall/projects/ray/ray_examples/

Caustics

Motion blur

Ray-casting

- For each *pixel*
  - Compute ray into scene
  - Find intersection with nearest object
    - Compute lighting (via position, normal)

Advantages of ray-casting

- Simple

Advantages of ray-casting

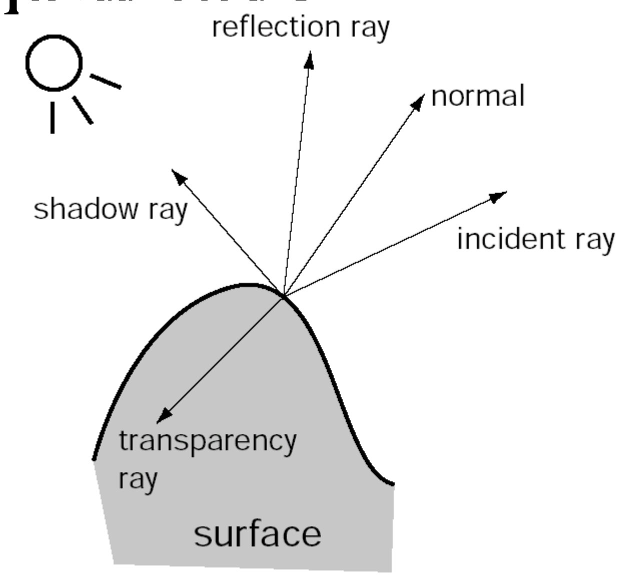
- Simple
- Can be extended to include global illumination effects:
  - Reflections (specular, glossy)
  - Shadows (hard, soft)
  - Depth of field
  - Motion blur
- Then it's called *ray-tracing*

Disadvantages of ray-tracing

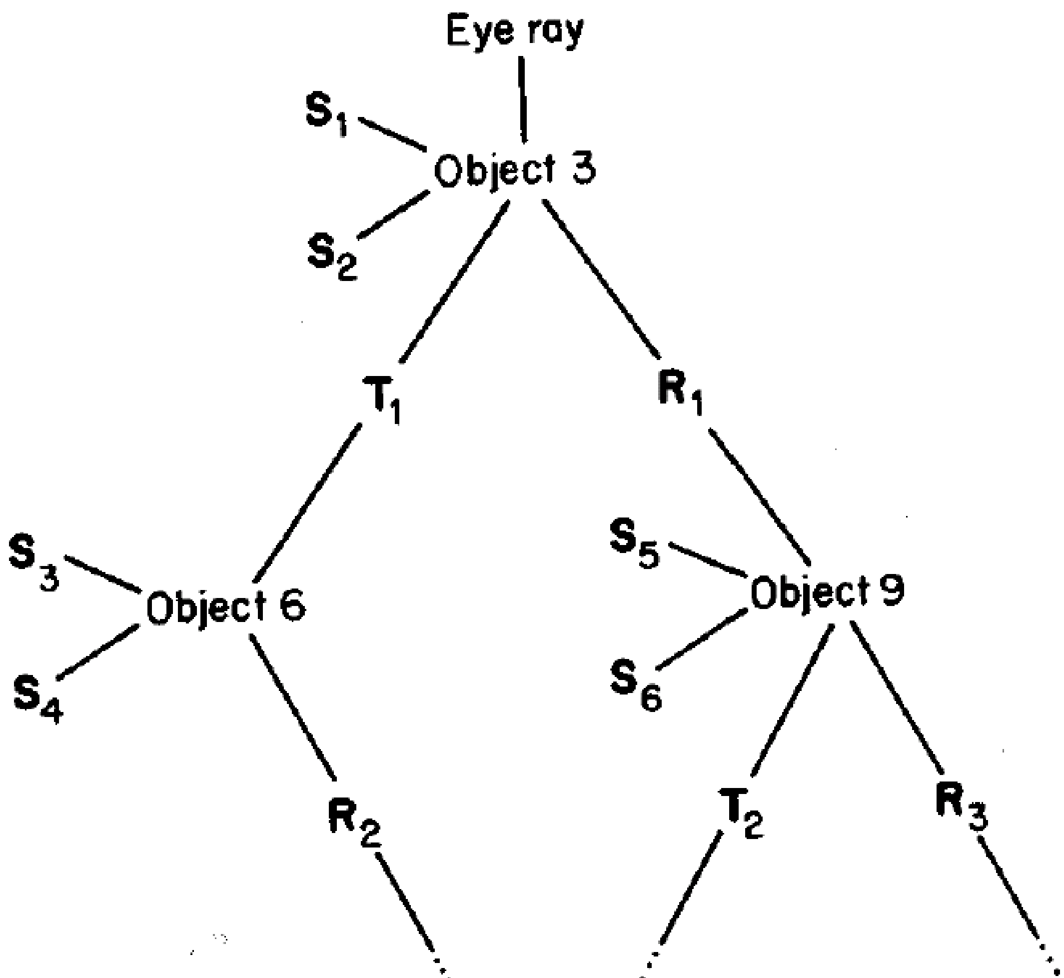
- Done in software: slower
- Adding realism can increase computations exponentially (distribution ray tracing)

Kinds of rays (basic ray tracing)

- Primary ray
  - leaves the eye and travels out to the scene
- When hit - spawn three new rays to “collect light”
  - shadow ray
    - towards light
  - reflection ray
  - transparency ray



The ray tree



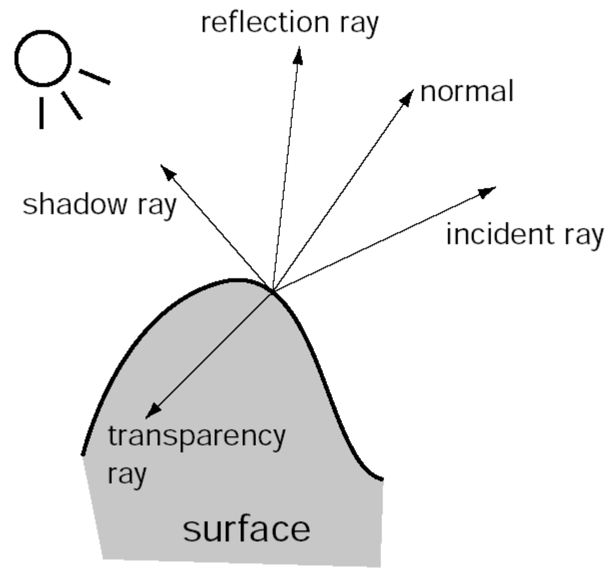
Raytracing is …

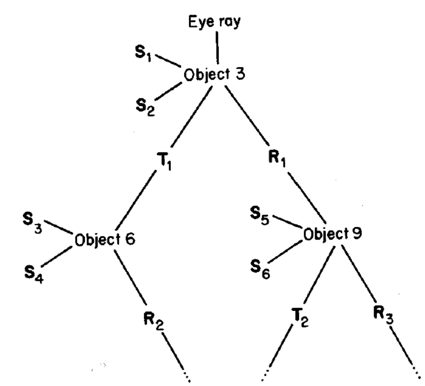
- **recursive**

- I(incident-out) =
  I(shadow-local-in)
  + Kr * I(reflection-in)
  + Kt * I(transparent-in)

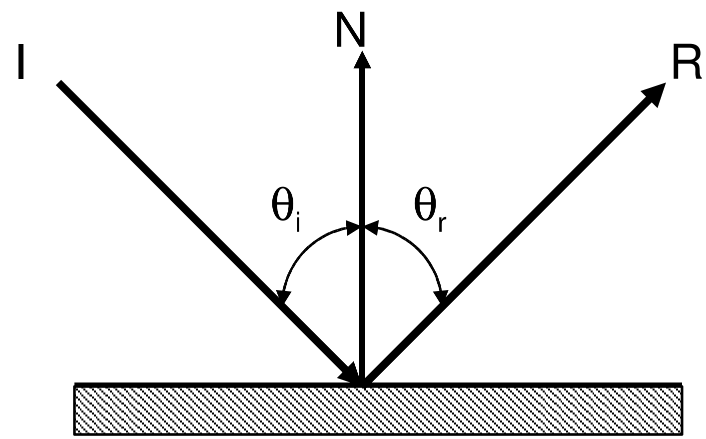
  – what is a range of Kr and Kt?

- Without recursion we have raycasting





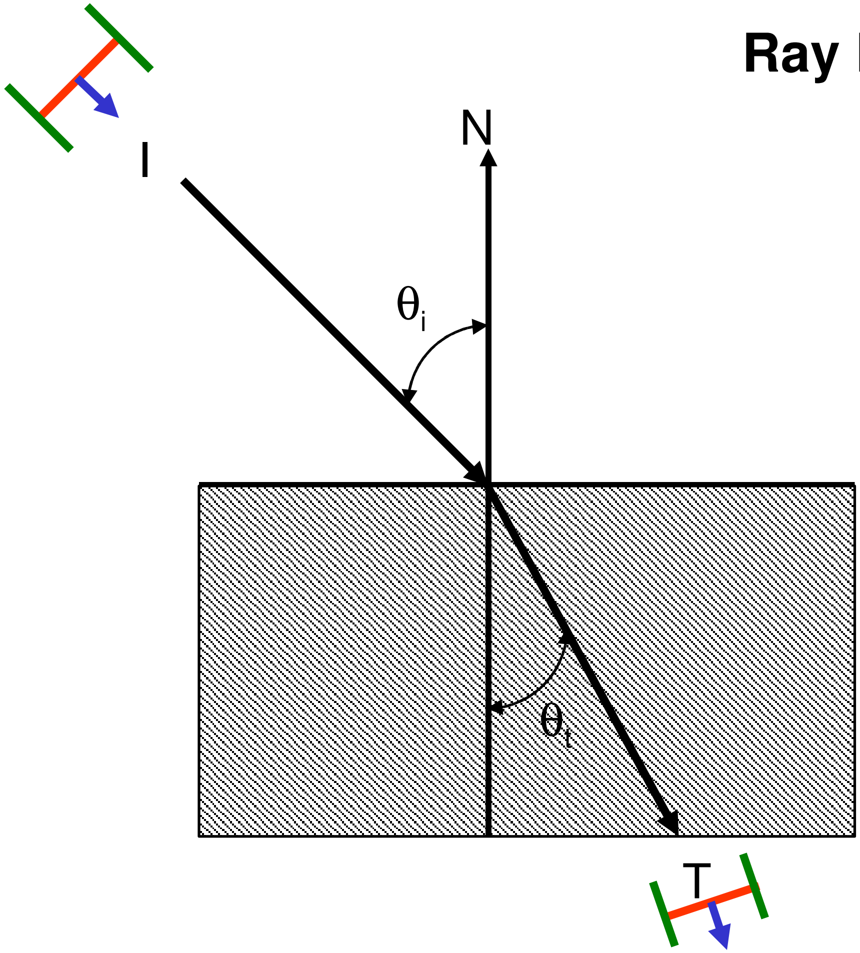
Ray Reflections



$\theta_I = \theta_r$

$$\vec{R} = \vec{I} - 2(\vec{N} \bullet \vec{I})\,\vec{N}$$

Ray Refraction



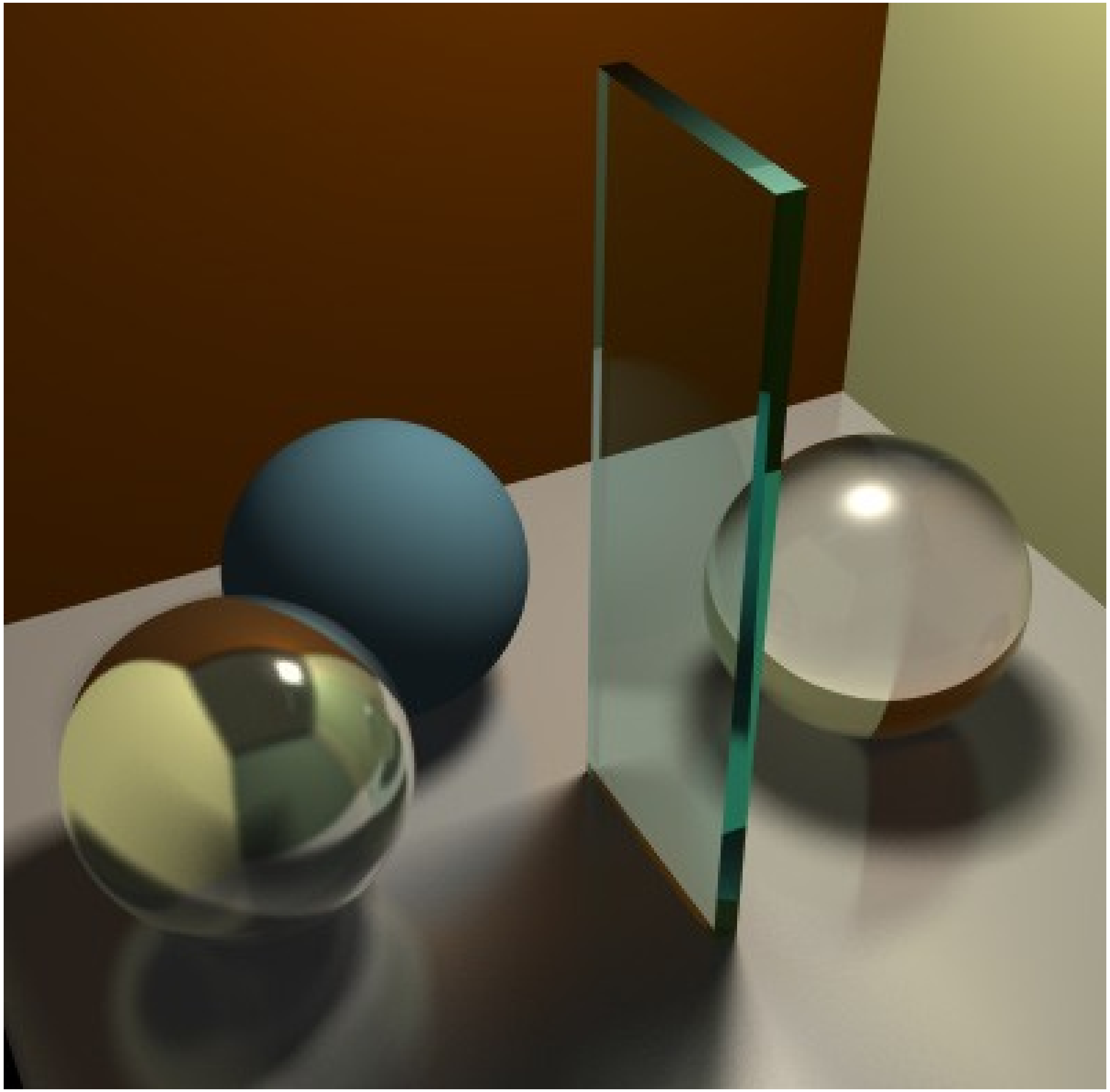
Snell's Law

$$\frac{\sin(\theta_i)}{\sin(\theta_t)} = \eta_{21} = \frac{\eta_2}{\eta_1}$$

Index of refraction: ratio of speed of light in a vacuum to speed in the material

Light Attenuation

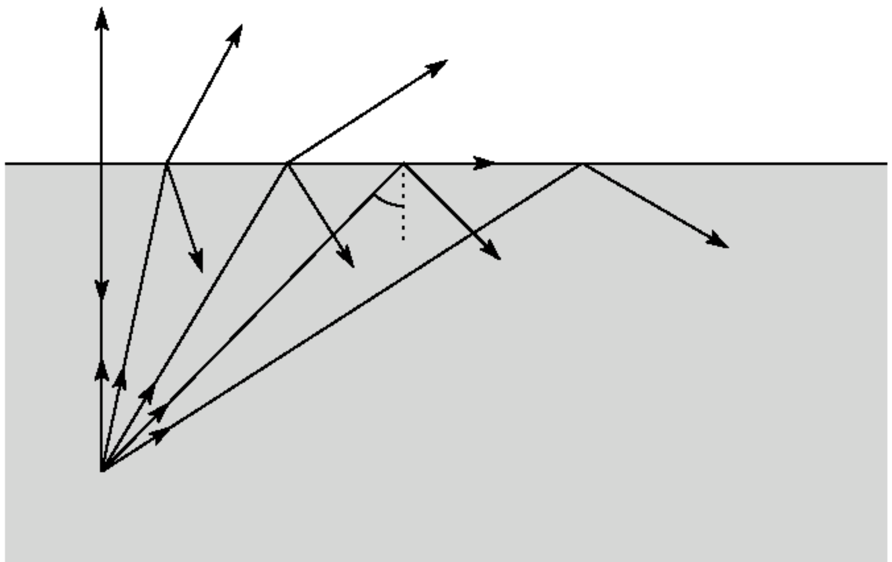
- Light may lose intensity and shift color
  - effect increases with distance
- Beer's law
  - Fall-off is exponential w/ distance
  - r, g, b components computed separately
  - text has details



http://www.jasonwaltman.com/graphics/rt-soft-fuzzy.html

Watch out for…

- Total internal refraction
  - light may not get through the interface



Computing intersections

- Crucial computation (inner loop)
- Spheres
- Planes
- CSGs

Speed-up techniques

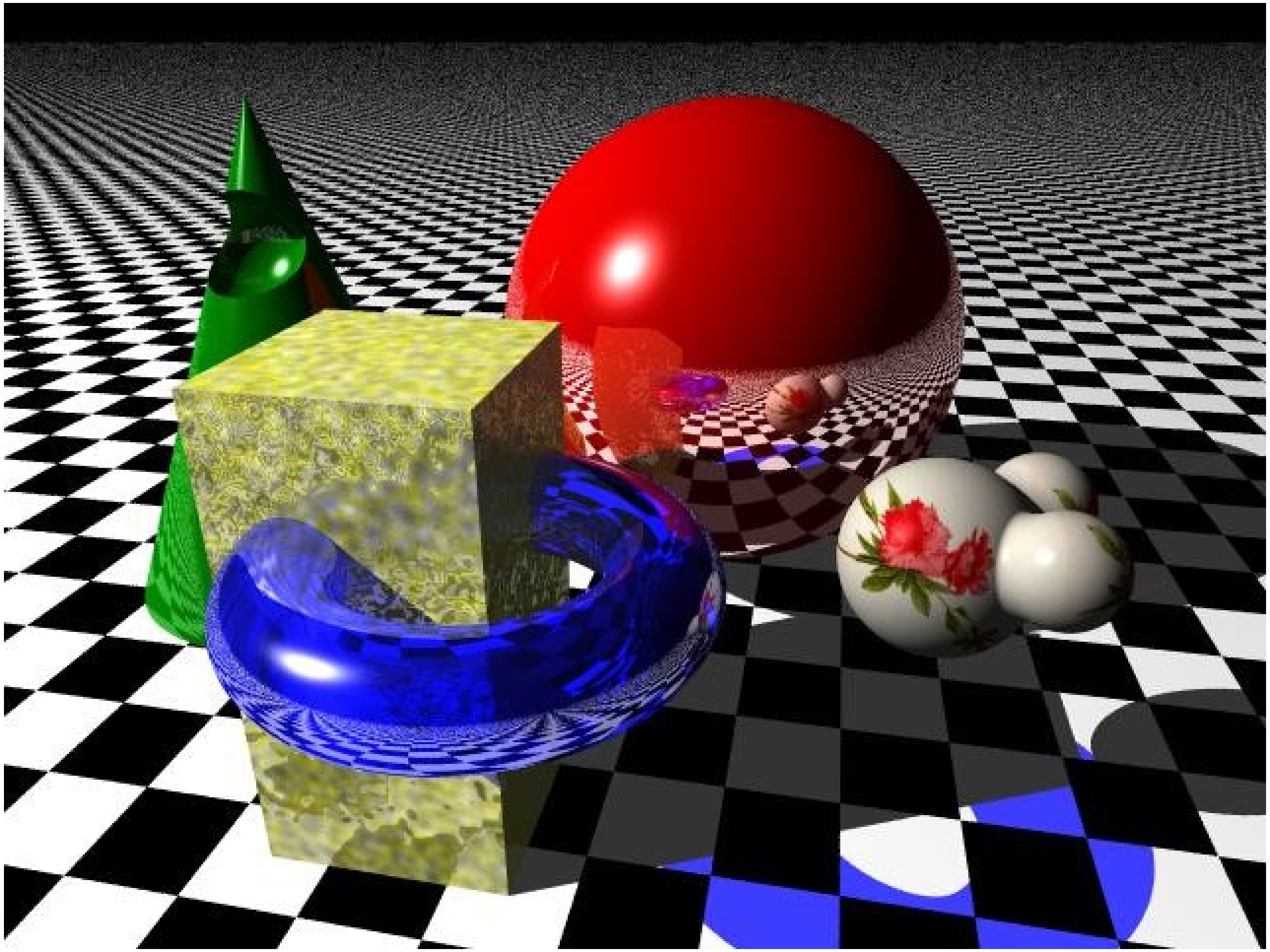
- Bounding volumes
  - Spheres
  - Boxes
- Uniform spatial subdivision
- Hierarchical bounding boxes

Using hierarchical bounding boxes

To check for intersections w/ objects in box:

- if ray misses box, return none
- if box is “leaf” test intersections w/ each triangle stored in the box, return closest
- else check for intersections w/ each child box, return closest

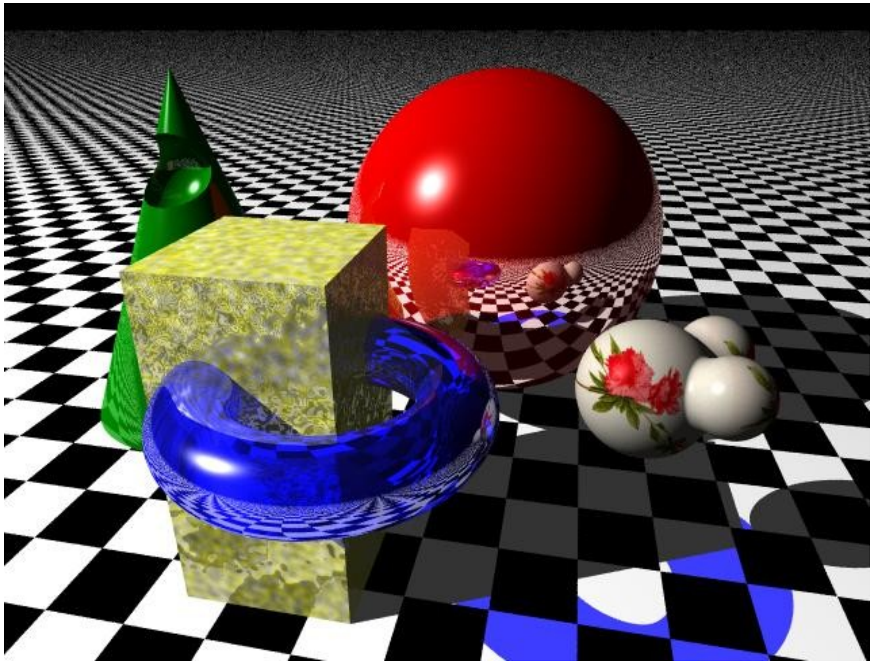
Problem (basic ray tracing):
images are too clean



http://www.tjhsst.edu/~dhyatt/supercomp/p501.html

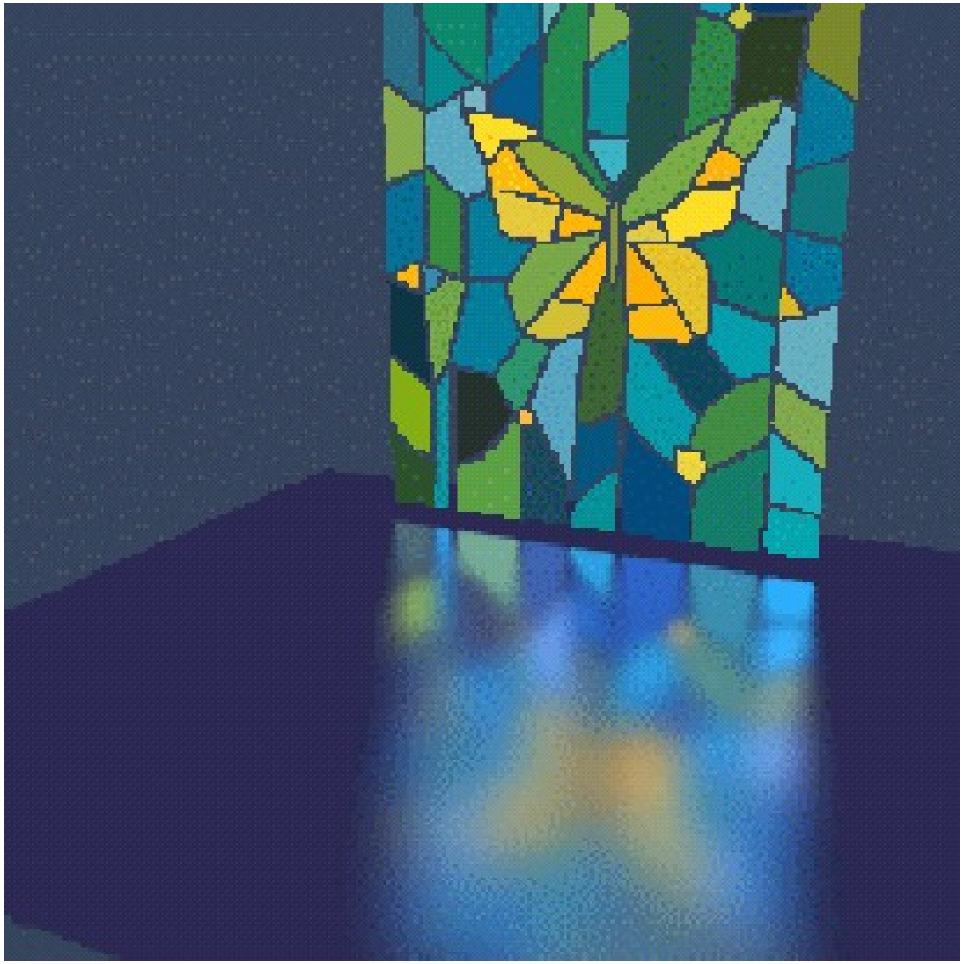
What's missing

- Reflections are perfect
- Shadows are hard
- Everything is in focus
- Shutter speed is infinite
- Prone to aliasing
  - Same as conventional pipeline



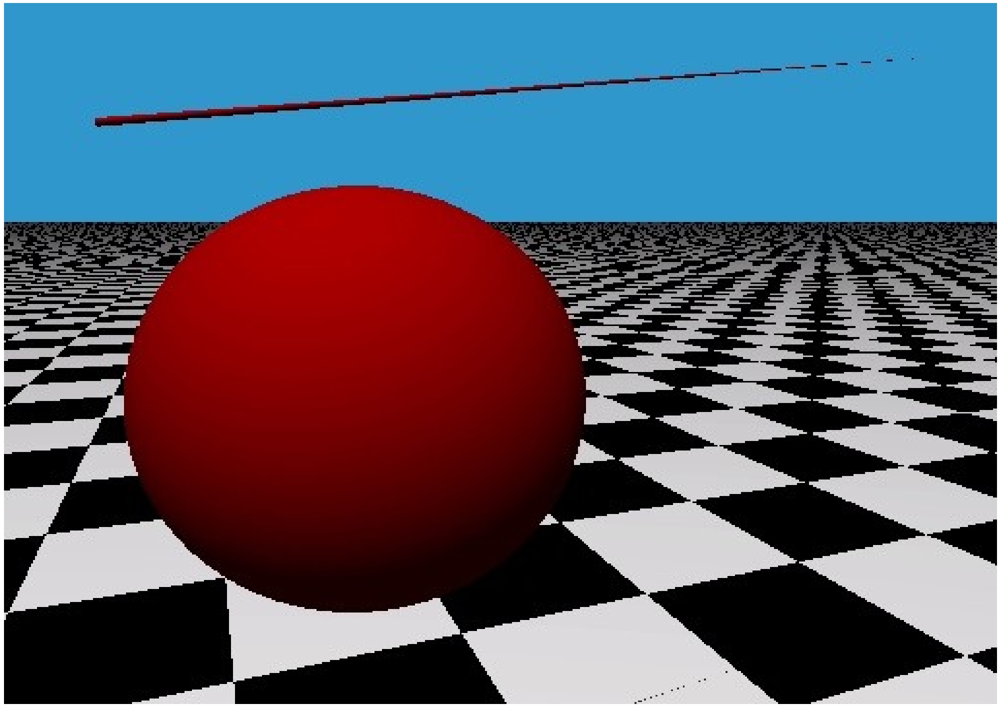
Strategy: random sampling

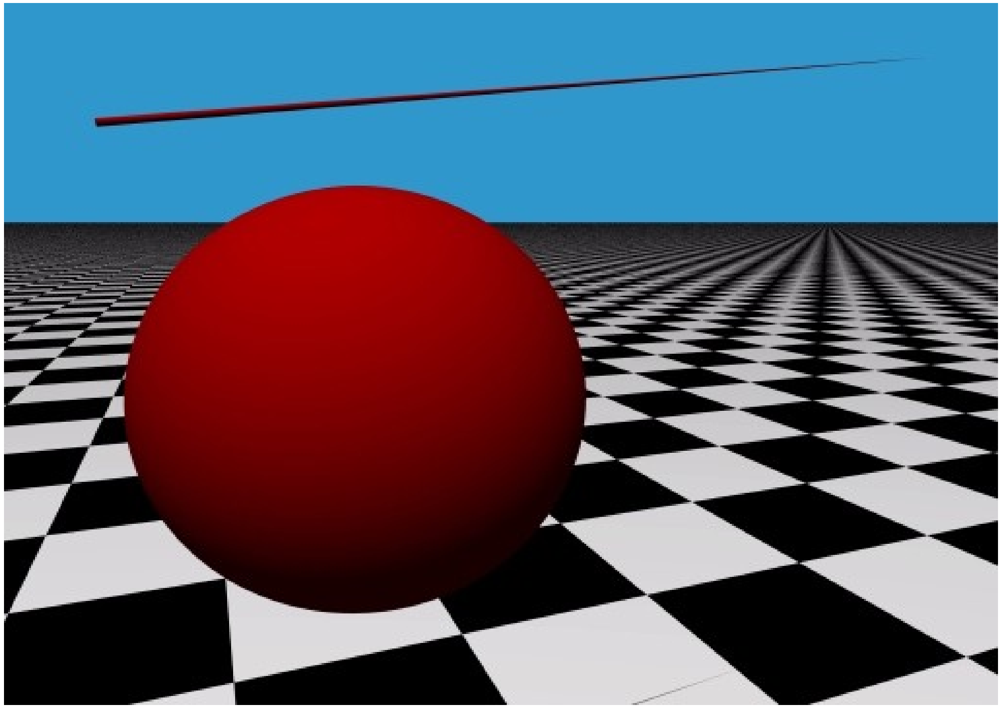
Can address all problems listed on previous slide



anti-aliasing

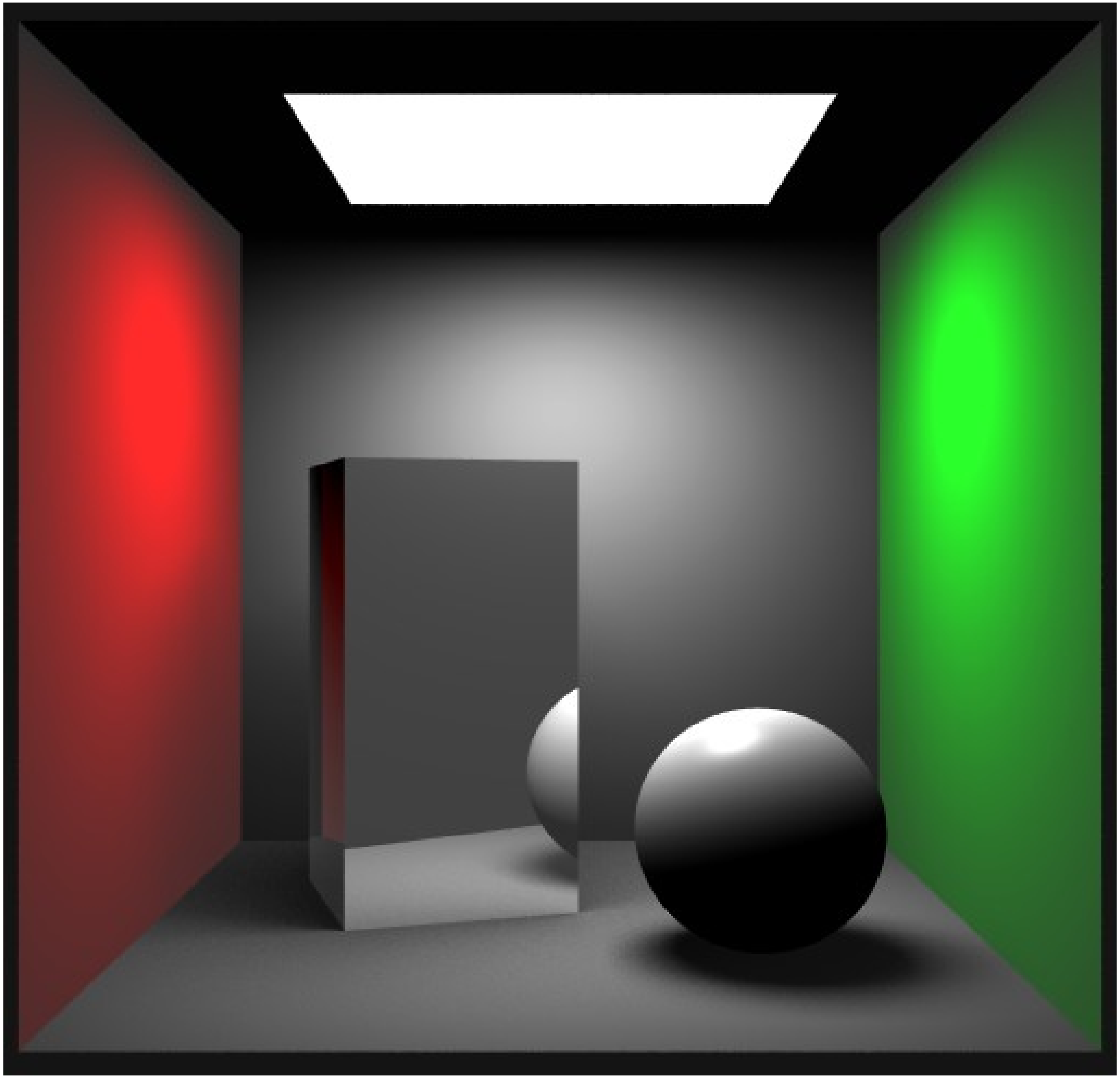
- use many rays per pixel
  - regular sampling
  - random sampling





soft shadows

- one approach: use many lights
  - approximate an area light with dozens of point lights
  - problem: overlapping hard shadows
- alternate approach:
  - sample the area light randomly w/ rays
  - random sampling discussed in text



http://www.cs.wpi.edu/~emmanuel/courses/cs563/write_ups/zackw/realistic_raytracing.html

more effects

- glossy reflection
  - follow multiple reflection rays, jittered randomly
- motion blur
  - multiple rays, jittered in time
- depth of field
  - multiple rays, jittered around eye, through focal plane


© 2003
Peter Hertel
http://hertel.no/peter


[INCUBUS|FINAL]

Wrap up

- Shirley (our textbook) has details on computing random samples effectively
- Project 5 (ray tracing) will go out Monday (after Thanksgiving)