#7: Photorealism

The High End

Elements of the Final Image

- Geometry
- Lights
- Materiality
- View points & focal points
- **Cameras & optics**
- **Rendering: color & light**

Synthetic Camera Issues

"the familiar"

Position: view/eye point

Orientation: focal/target point

Projection: lens (telephoto -> fish-eye)

VRay "EV"

- Aperture: f-stop
- Film speed: ISO
- Shutter speed: how long the shutter is open

Synthetic Camera

"beyond our scope"

Depth of field: what's in focus

Lens Flare: in-lens reflections

Chromatic Aberration: baby rainbows

Motion Blur: shutter speed, motion

Rendering: Color & Light

"it's about light & surface"

Perfectly Diffuse Reflections? No!

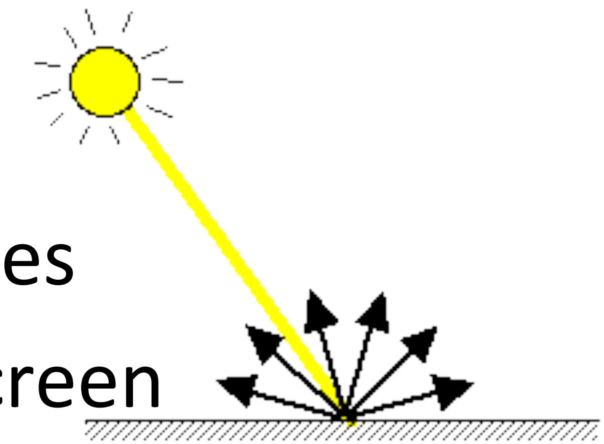
Perfectly Specular Reflections? No!

So … ??

Light Bounces in complex ways!

Review: Classical Rendering

"Top-down" (Phong, *et al.*)



- Assumes perfectly diffuse surfaces
- Works from geometry data to screen
- Aside from shadows, ignores other geometry
- Does not render indirect illumination, color bleed, ambient occlusion, caustics, or most specular effects (refraction and reflection).
- Solution is view-dependent

Wait! What do we mean by …

- Indirect illumination
- Color bleed
- Specular effects: Reflection & Refraction
- Caustics
- Ambient occlusion
- View-independence

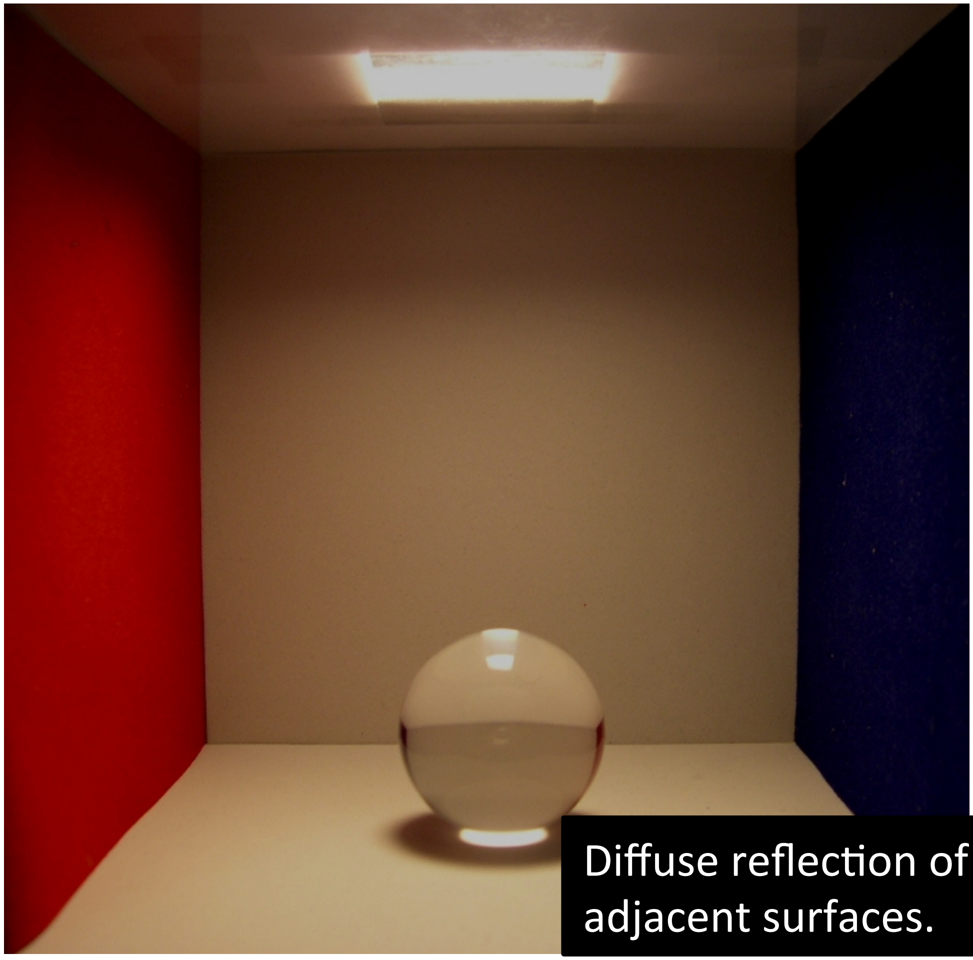
Indirect illumination

Color Bleed

What color is this wall?

What color is this wall?

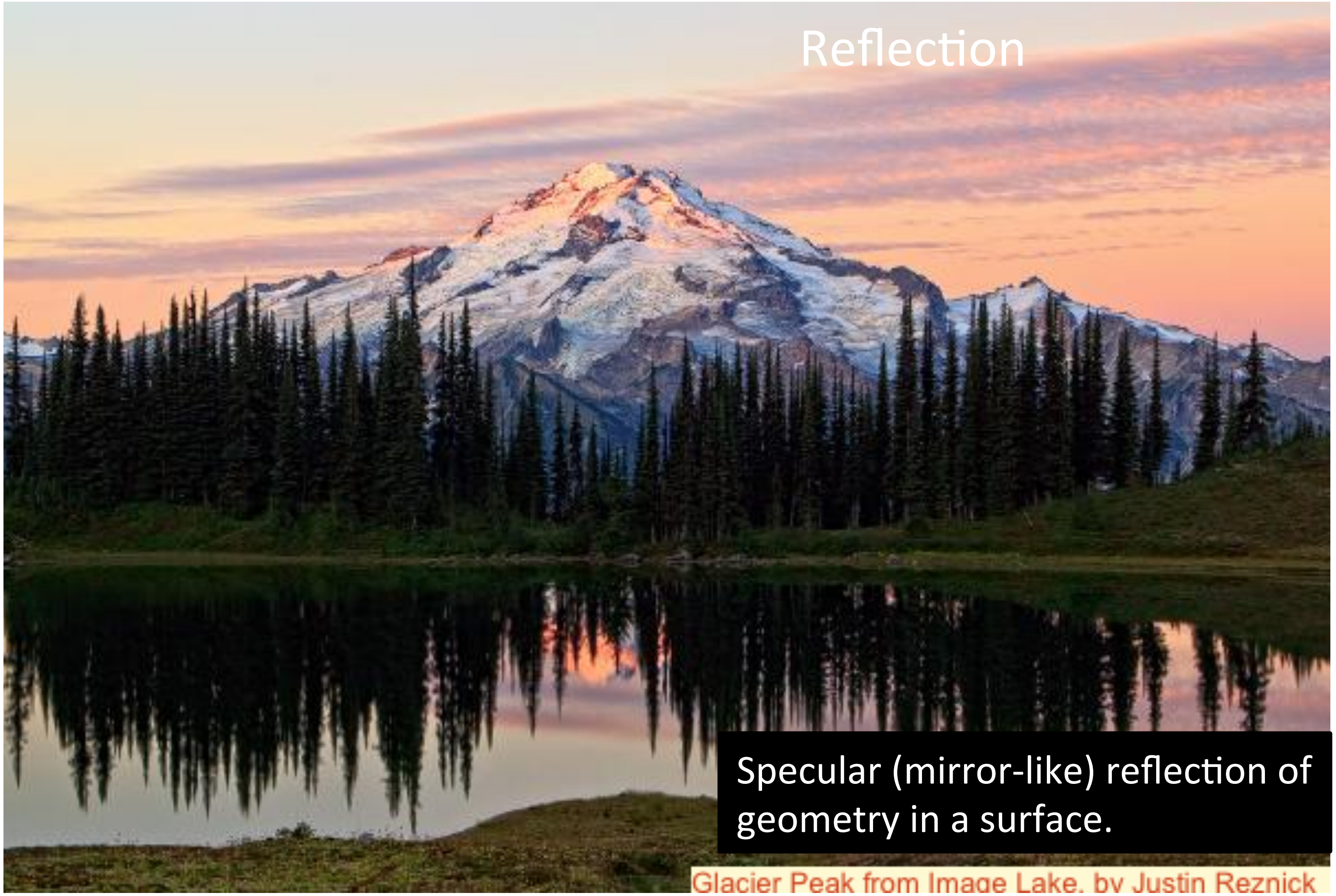
Color Bleed



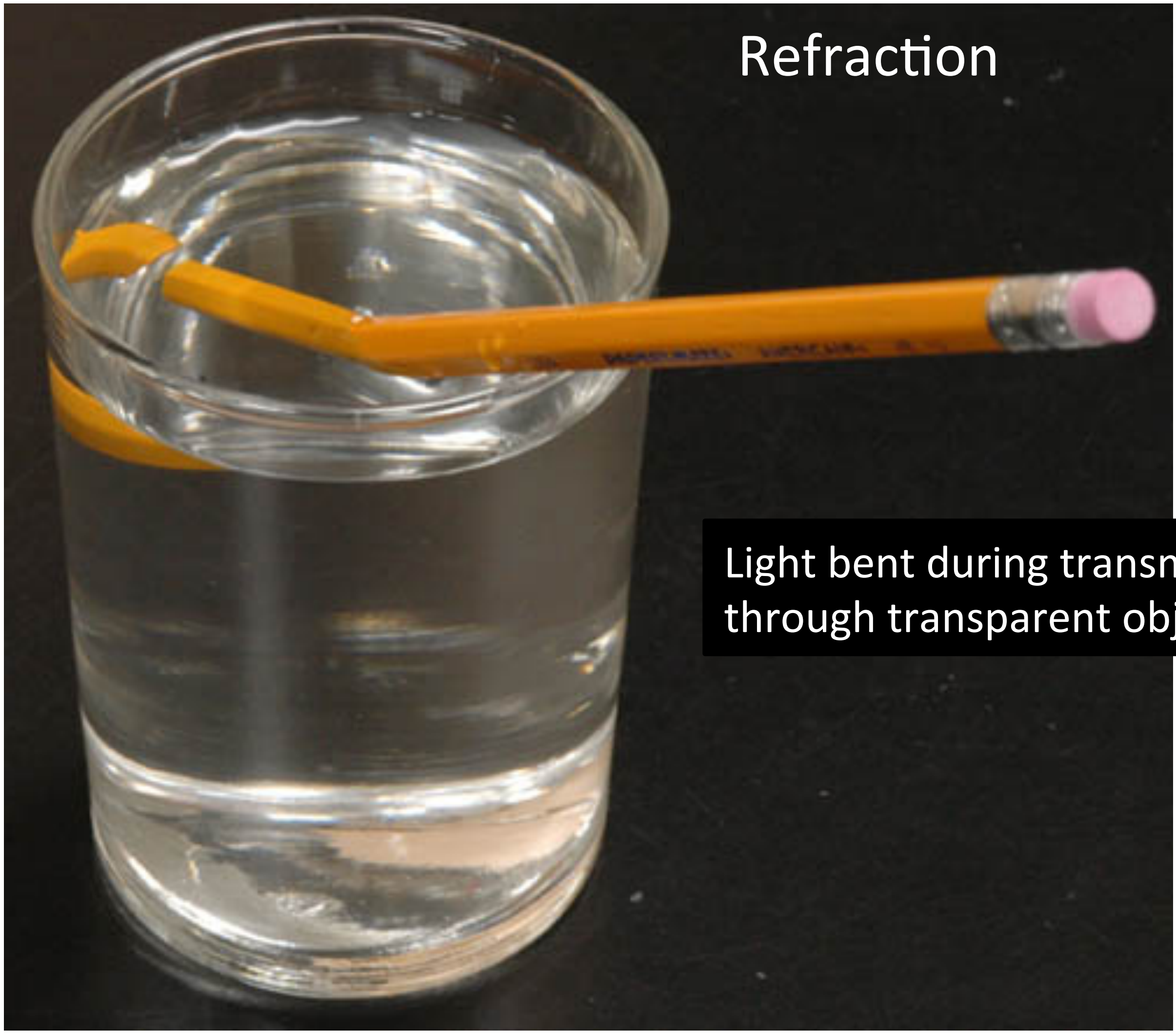
Diffuse reflection of color from adjacent surfaces.

The result of diffuse inter-reflection.

Specular Effects



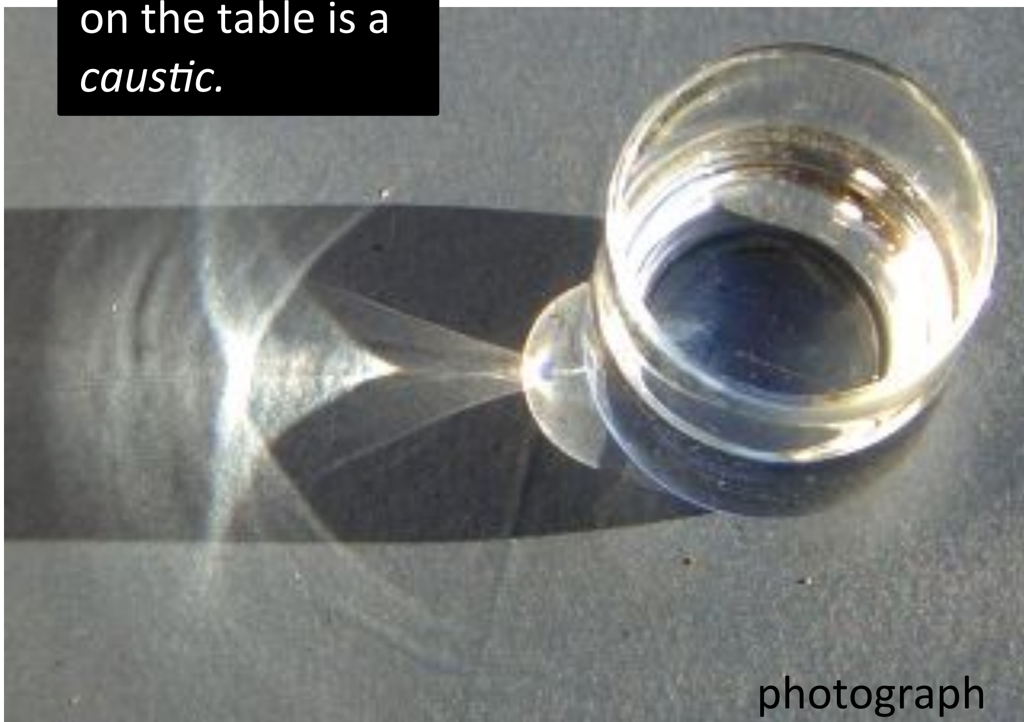
Specular Effects



Light bent during transmission through transparent objects.

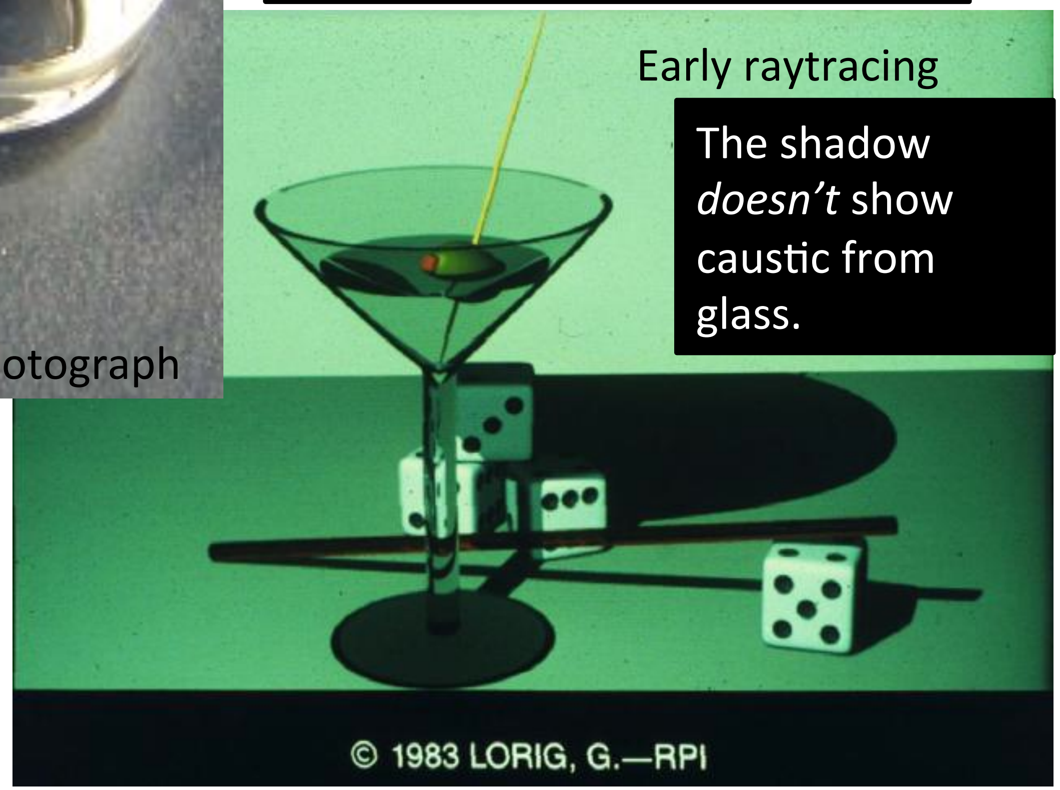
Caustics

The light pattern on the table is a *caustic.*



Refracted light

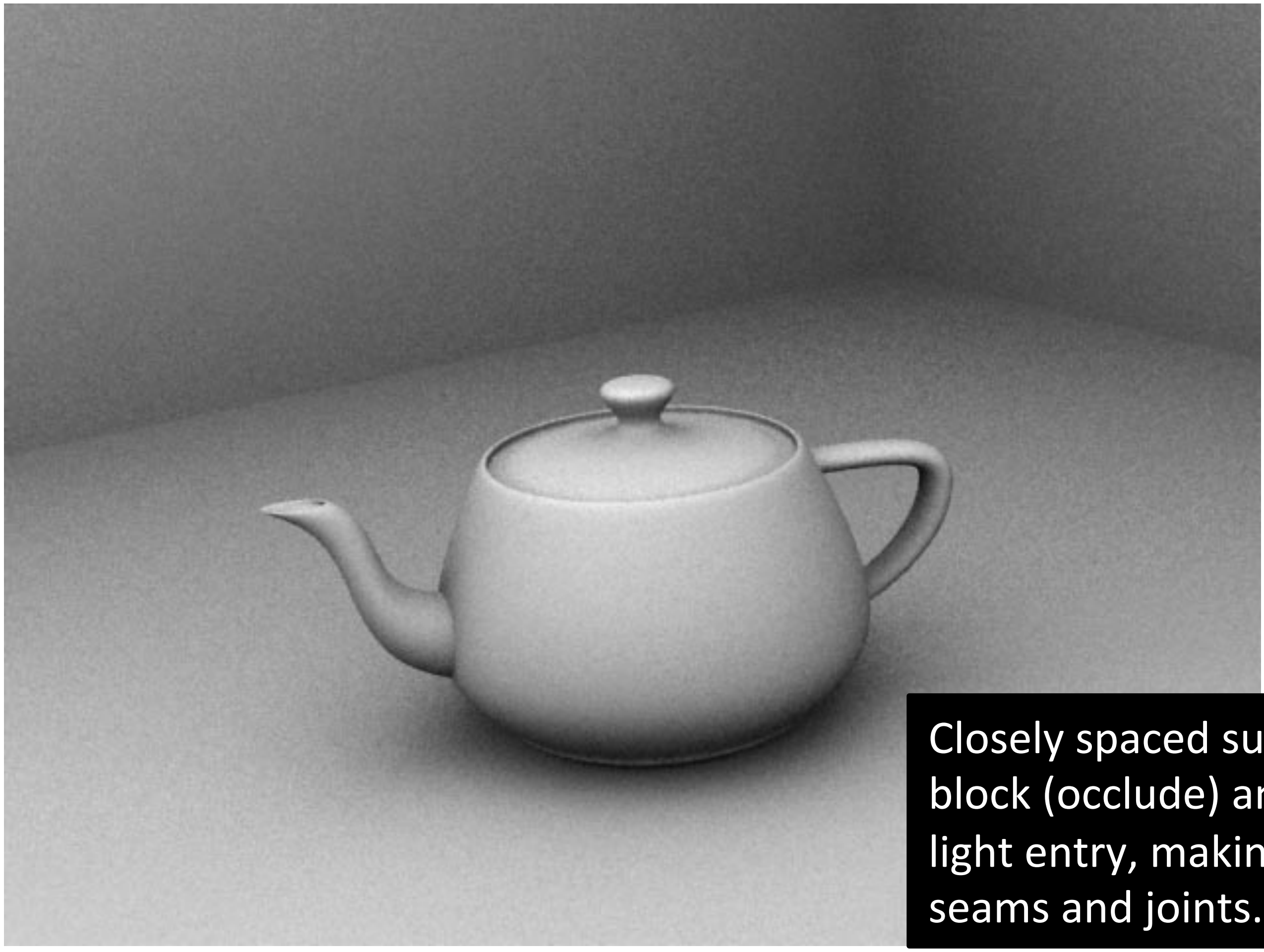
Diffuse reflection of light AFTER specular refraction.



The shadow *doesn't* show caustic from glass.

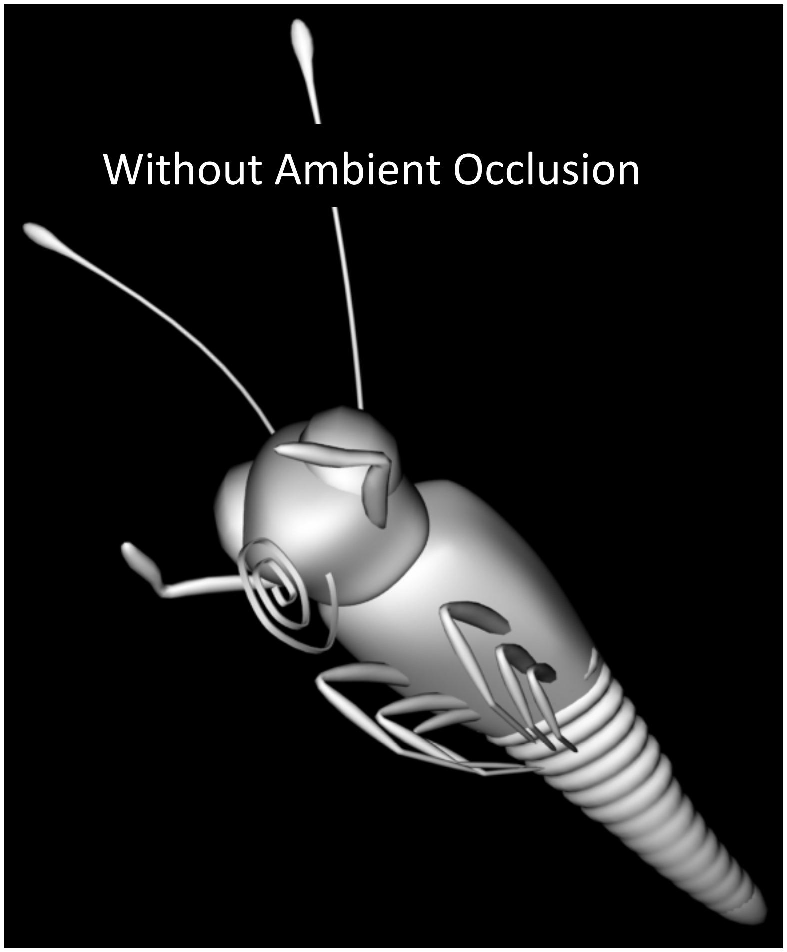
Diffuse Surface

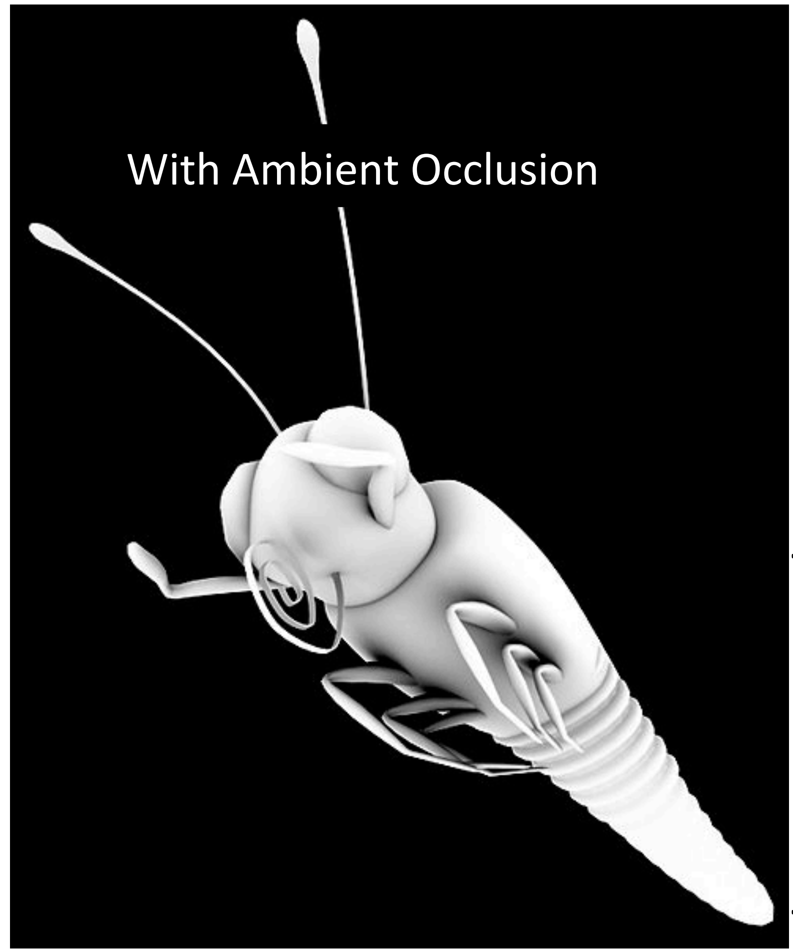
Ambient Occlusion



Closely spaced surfaces block (occlude) ambient light entry, making dark seams and joints.

Ambient Occlusion





Chaos-group graphic

What is "View-dependence"

- Light distribution in the real world does not depend on the camera position (no flash!).
- A rendering always has geometrical dependencies on viewpoint/etc.
- Given the cost of computing light distribution, it would be nice if a single lighting solution could be shared by a series of renderings, as in an animation.

Now: Improving Classical Rendering

- Anti-alias by super-sampling @ 3:1 or 4:1
- Anti-alias textures too
- Use extra “fill” lights (no shadows)

Let’s try a different approach

Light reflects in “infinite” ways, but there are a finite number of screen pixels and model surfaces.

This suggest two strategies:

-Use “finite element” logic on surfaces

-Follow a finite number of individual rays (frontwards or back)

Real Light: Two Possibilities

1. Compute *energy exchange* from geometry (radiosity, radiance)

2. Sample the scene by *tracing* a finite number of individual sight and/or light rays (V-Ray)

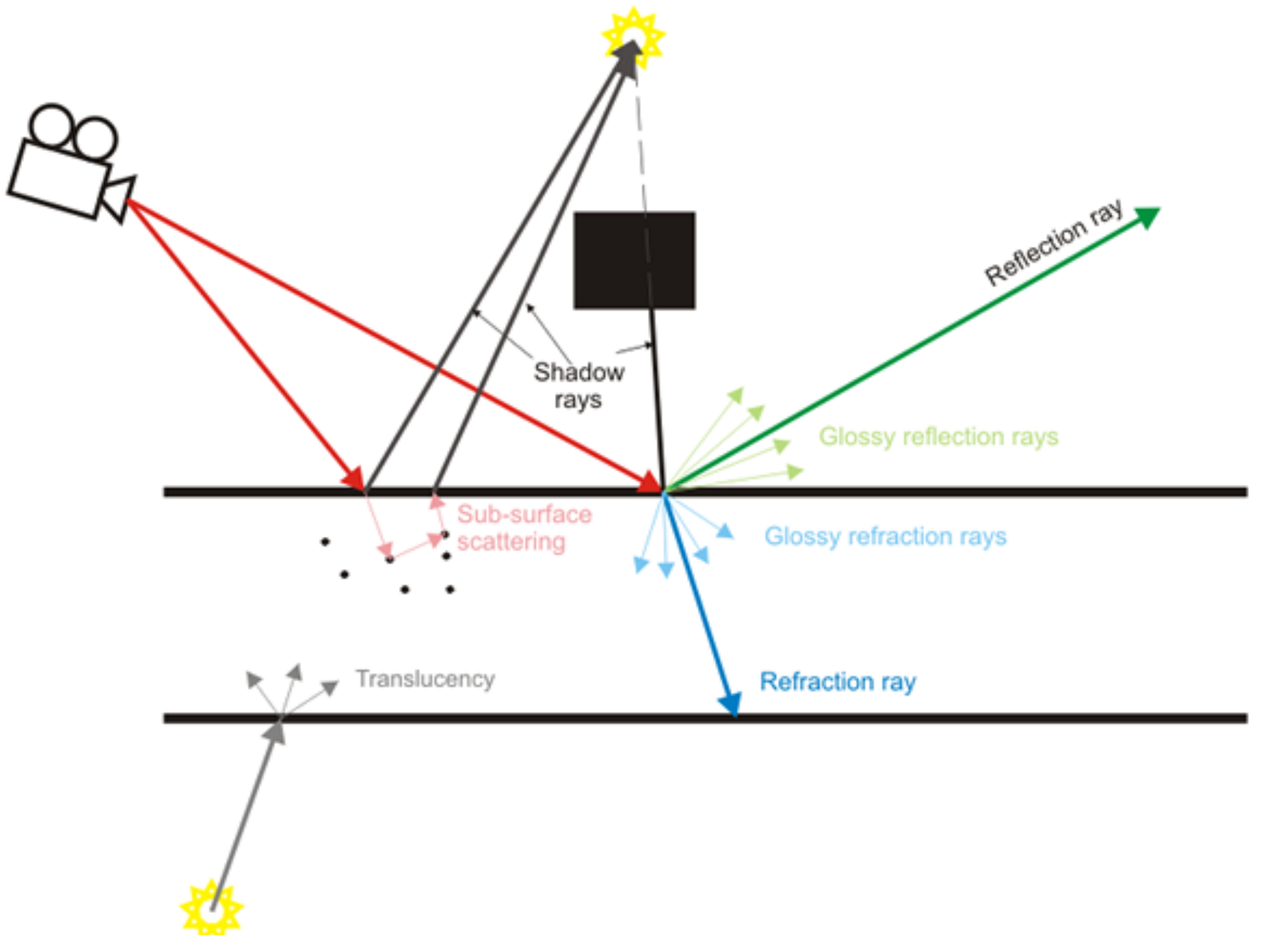
First Approximation: Ray Tracing
“individual rays”

Almost Real Light

IF:

- light travels in straight lines
- all reflections are specular

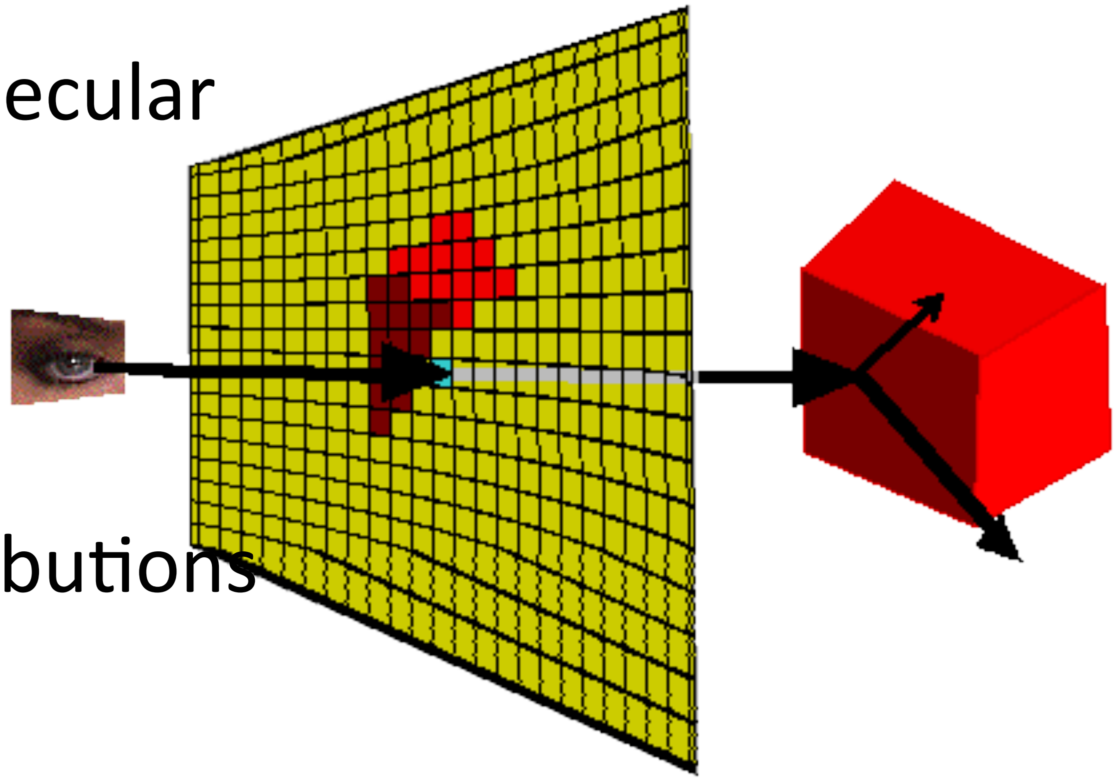
Ray-Surface Interactions



Chaos-group graphic

Light travels in straight lines

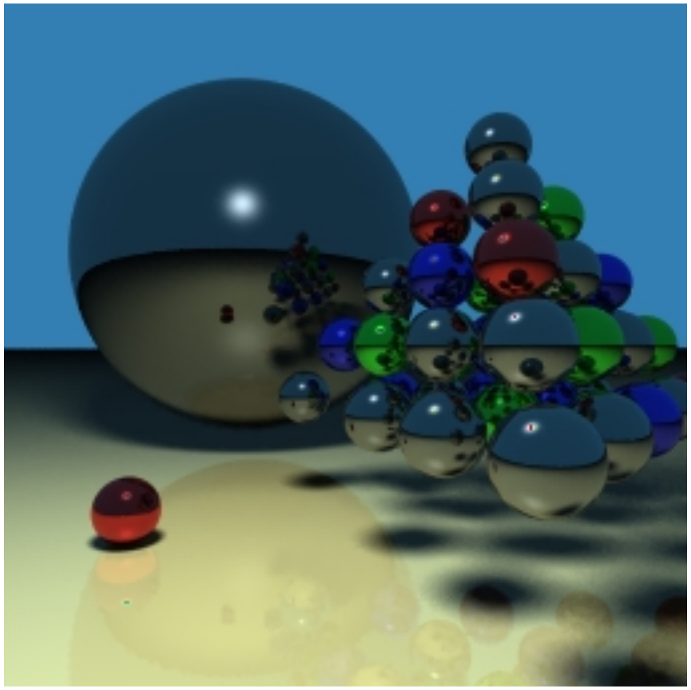
- To reach the eye, it came from the pixel
- To reach the pixel it came “from” the model
- Each bounce is specular
- Work backwards
- Each pixel is separate
- Add up the contributions of each bounce.



Ray Tracing “score card”

- Reflection ✔
- Refraction ✔
- Indirect Illumination ✖
- Color Bleed ✖
- Caustics ✖
- Ambient Occlusion ✖

- View-dependent solution



Getting a Good Ray-tracing

- Anti-alias by super-sampling @ 3:1 or 4:1
- Anti-alias textures too
- Set "maximum-bounces" higher
  (note, bounces = 0/1 -> "hidden surfaces")

The FEM "energy exchange" model

Correct in diffuse environments

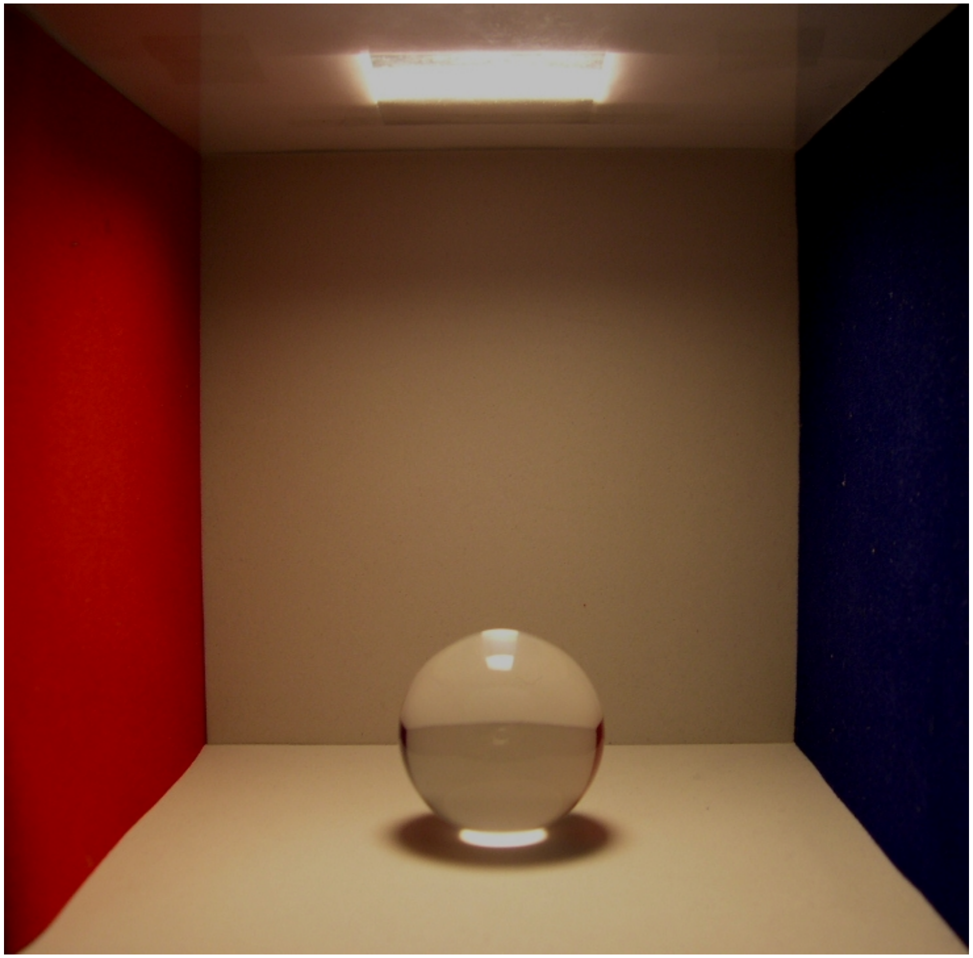
Given:

light is a form of energy

all reflections are diffuse

Developed at Cornell University

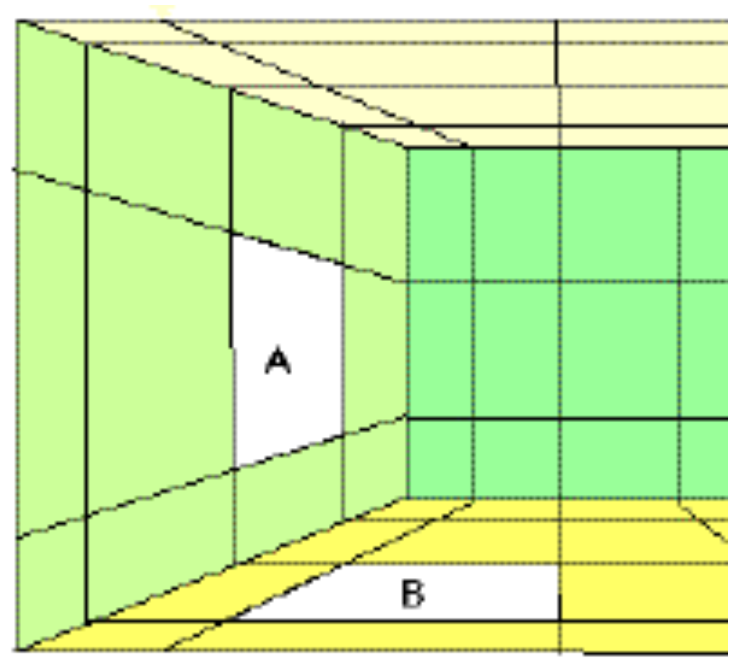
Radiosity energy exchange



Developed at Cornell University

Radiosity: energy exchange

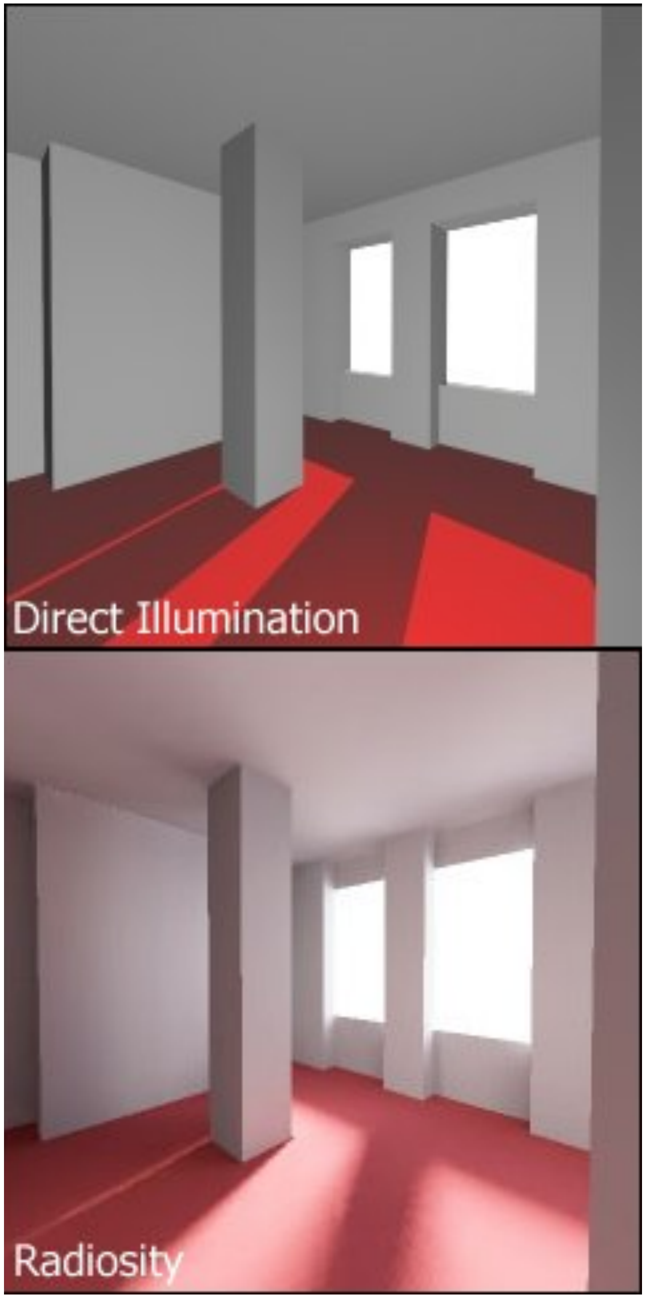
- Divide surfaces
- Compute 'angle factors'
- Distribute direct illumination
- Compute bounce redistribution.
- Subdivide high-contrast edges.
- Smooth resulting image



Radiosity "score card"

- Reflection ✖
- Refraction ≈
- Indirect Illumination ✔
- Color Bleed ✔
- Caustics ✖
- Ambient Occlusion ✖

\+ Solution is view-independent



Getting Good Radiosity Result

- Anti-alias by super-sampling @ 3:1 or 4:1
- Anti-alias textures too
- Set “% energy distribution” to high number (e.g. 99%)
- Combine with Ray-Tracing for reflection & refraction effects

Getting Both Specular and Diffuse Effects in One Rendering?

Path-Tracing: Enhanced ray-tracing

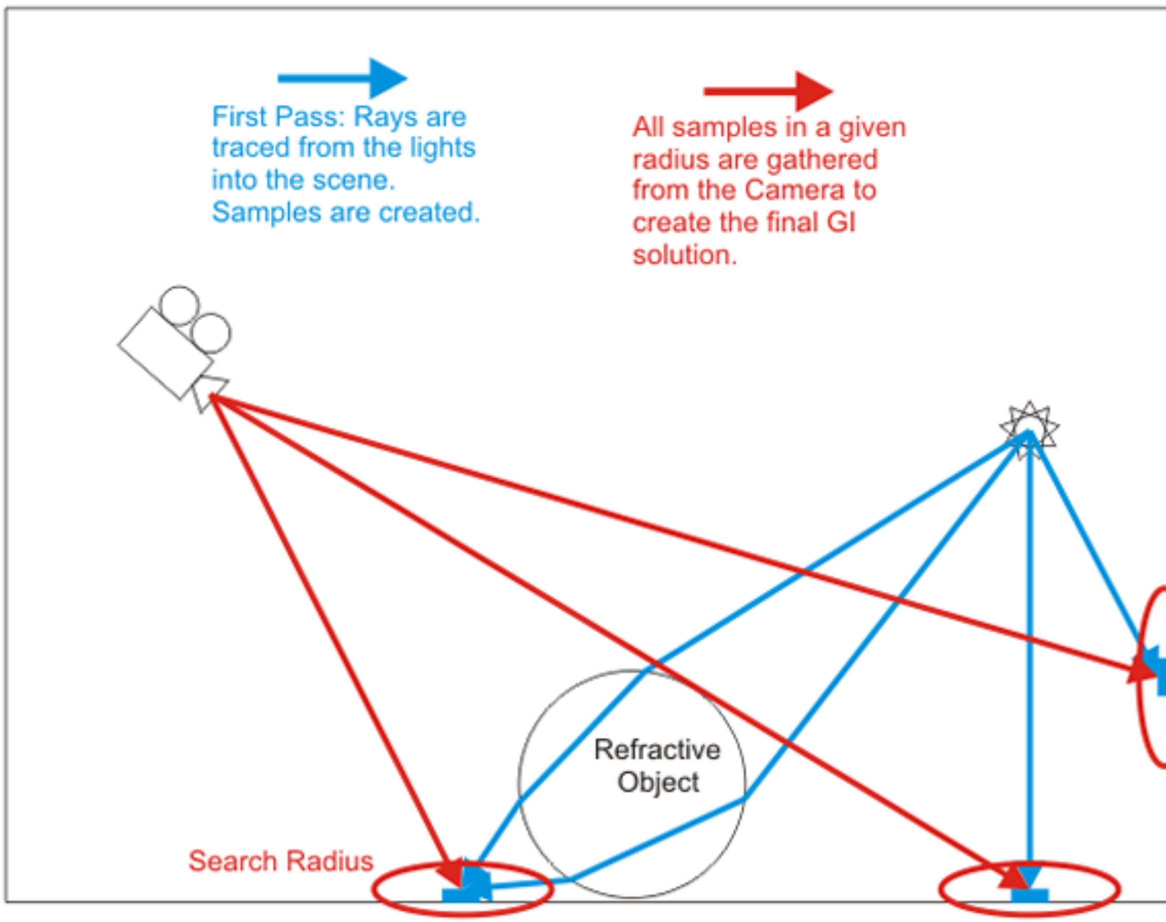
Work the problem from both ends:

1. Trace light from sources to surfaces
2. Back-trace light from eye into scene.

-AND-

3. Compute diffuse reflections by using statistical ("sampling") approach

Path Tracing: Two Passes



Chaos-group graphic

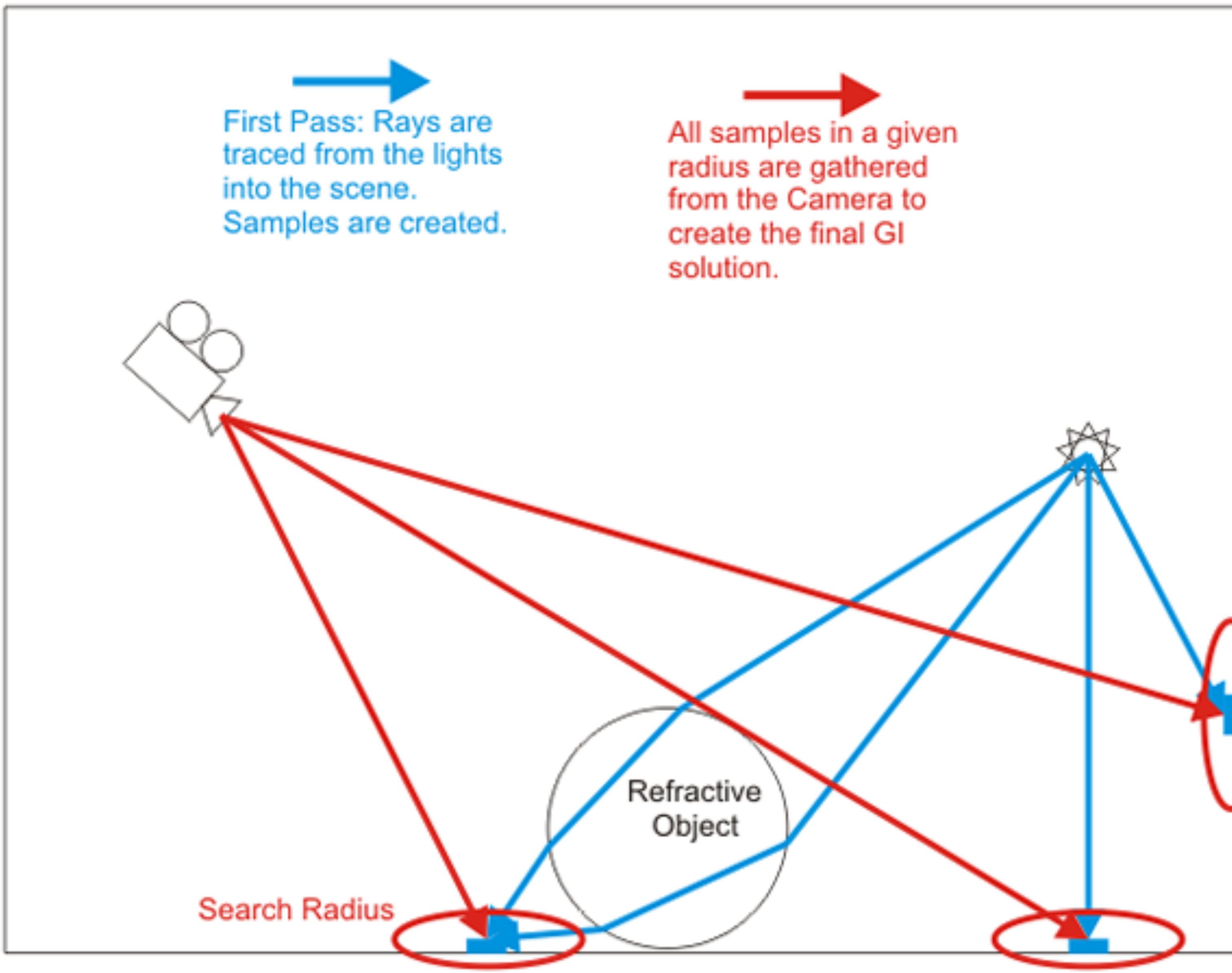
1. Trace Forward from the Lights

- Send "photons" (light rays) into the scene (distribute accurately)
- Whenever a photon strikes a surface, record where & in what direction in a "photon map"
- Use surface qualities (BRDF) to compute probable bounce direction.
- Bounce
- Repeat

2. Trace Backwards from the Eye

- "Back-trace" ray into scene
- At surfaces, search around in photon map.
- Compute effect of photons.
- If surface is specular, follow classical bounce

Review…


First Pass: Rays are traced from the lights into the scene. Samples are created.
All samples in a given radius are gathered from the Camera to create the final GI solution.
Refractive Object
Search Radius

Improving Path Tracing

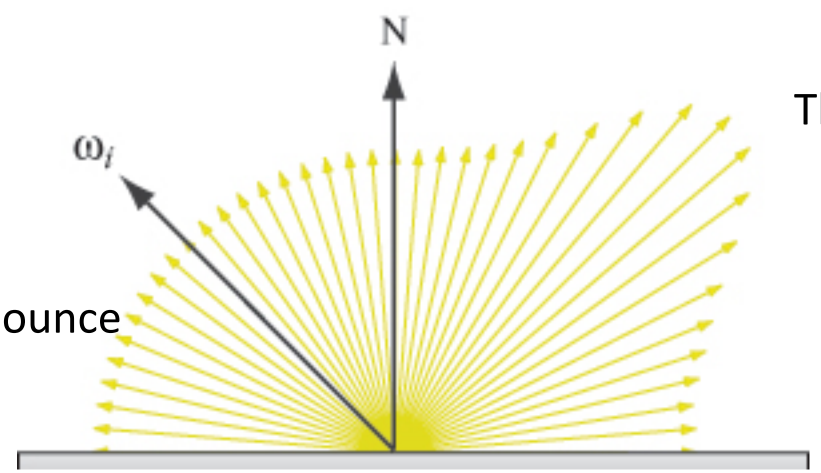
- Anti-alias by super-sampling @ 3:1 or 4:1
- Anti-alias textures too
- Increase number of photons in phase 1
- Increase number of bounces in phase 2
- Decrease search radius to sharpen image
- Increase image samples to improve detail

… umm .. BRDF?

Real surfaces bounce light in complex ways that depend on the wavelength of the light, the angle of incidence, etc. The function that describes how much light goes where is called …

"BRDF"

Bidirectional **R**eflectance **D**istribution **F**unction
(where reflected light really goes)



The specular-ish bounce

The diffuse-ish bounce

#7: Photorealism

Not The End