MIT EECS 6.837 Computer Graphics

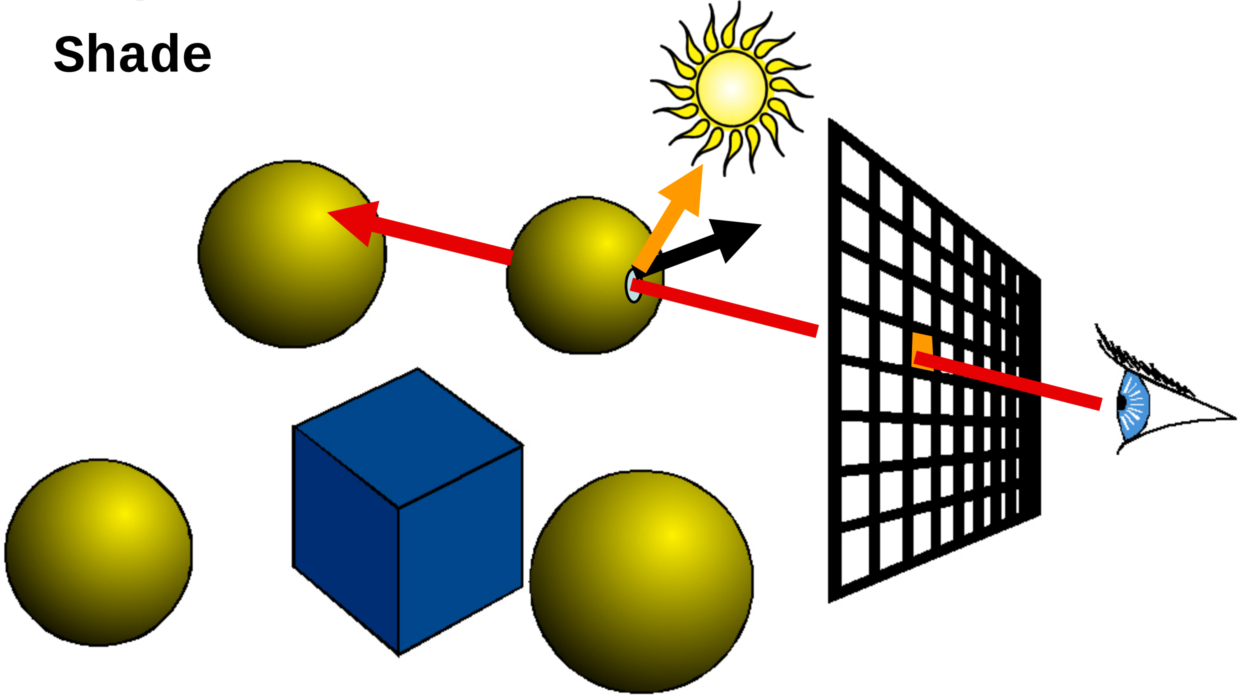
Ray Tracing

Henrik Wann Jensen

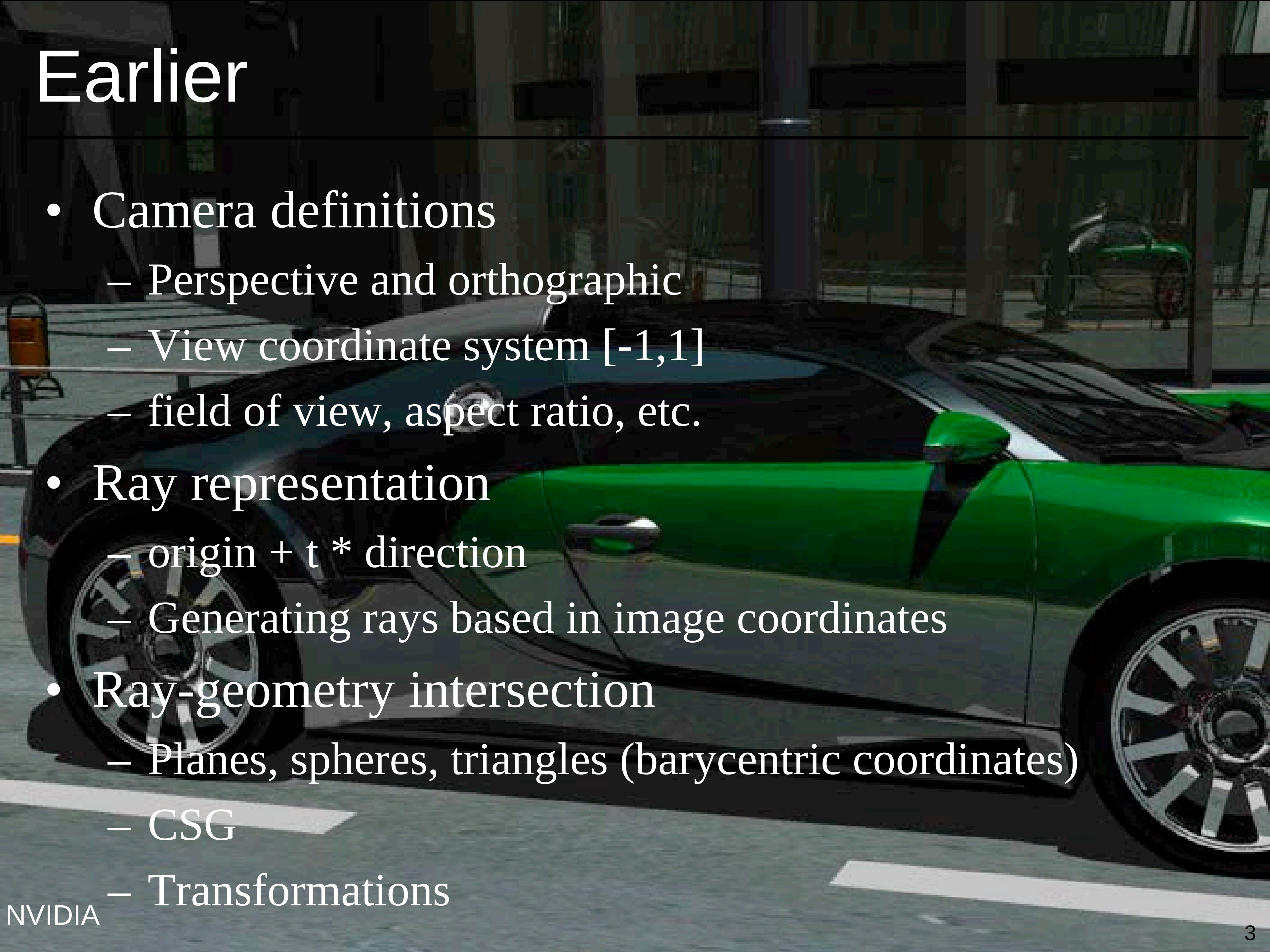
Wojciech Matusik, MIT EECS
Many slides from Jaakko Lehtinen and Fredo Durand

Ray Casting

```
For every pixel
  Construct a ray from the eye
  For every object in the scene
      Find intersection with the ray
      Keep if closest
       Shade
```

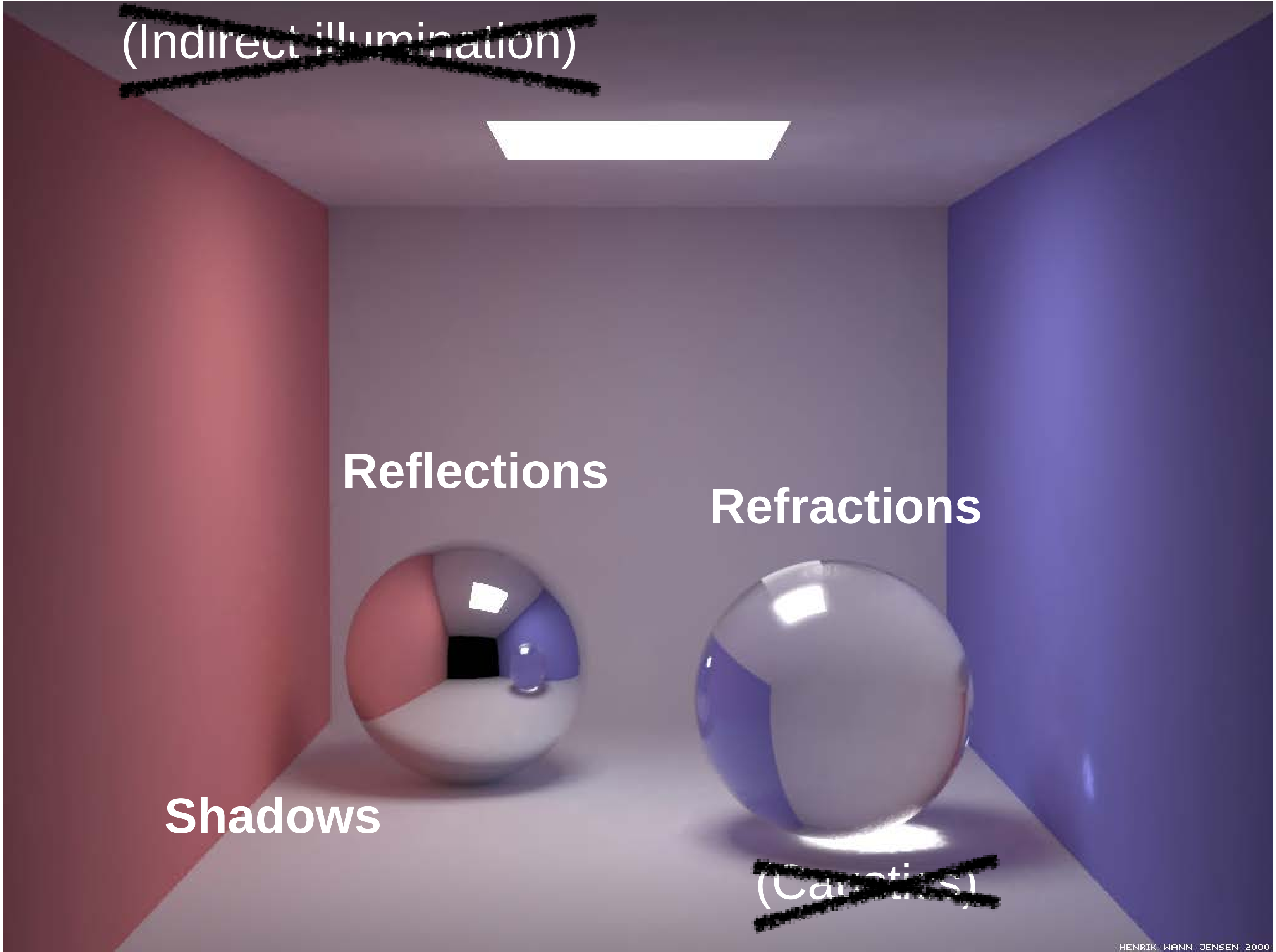


Earlier



- Camera definitions
  - Perspective and orthographic
  - View coordinate system [-1,1]
  - field of view, aspect ratio, etc.
- Ray representation
  - origin + t * direction
  - Generating rays based in image coordinates
- Ray-geometry intersection
  - Planes, spheres, triangles (barycentric coordinates)
  - CSG
  - Transformations

Today – Ray Tracing



~~(Indirect illumination)~~

Reflections

Refractions

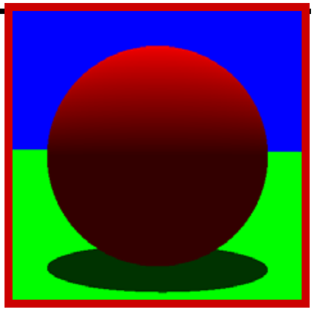
Shadows

~~(Caustics)~~

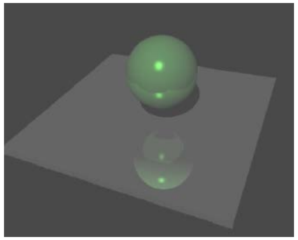
Henrik Wann Jensen

Overview of Today

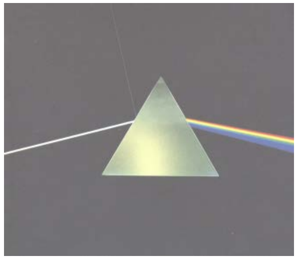
- Shadows



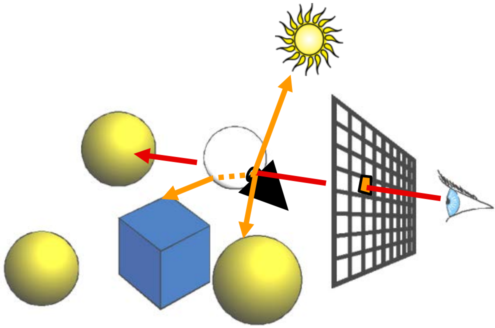
- Reflection



- Refraction

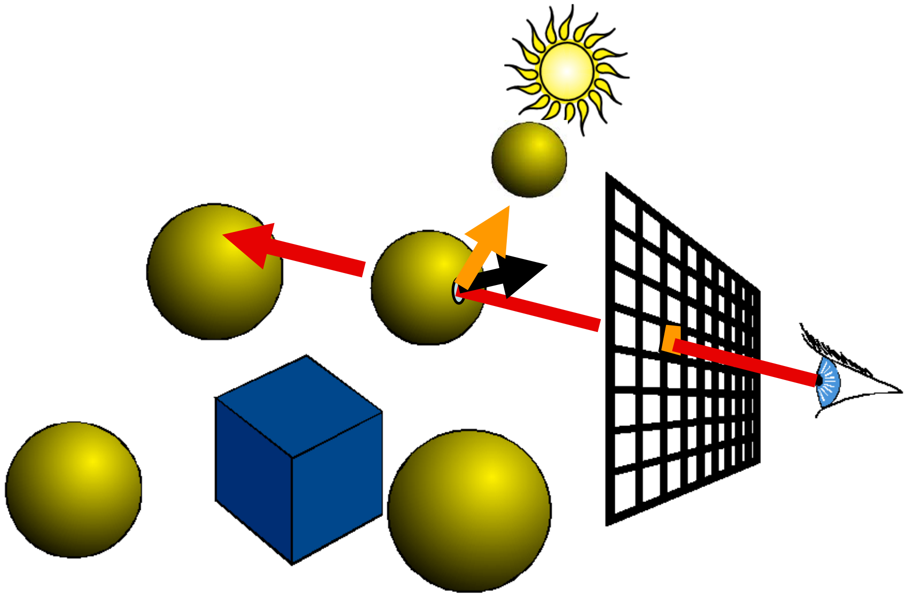


- Recursive Ray Tracing
  - "Hall of mirrors"



How Can We Add Shadows?

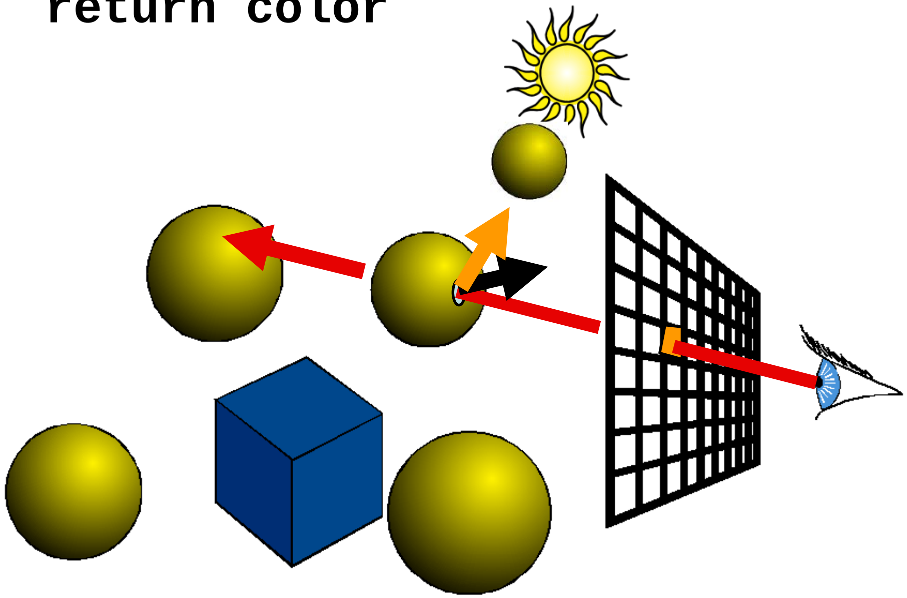
```
For every pixel
    Construct a ray from the eye
    For every object in the scene
          Find intersection with the ray
          Keep if closest
           Shade
```

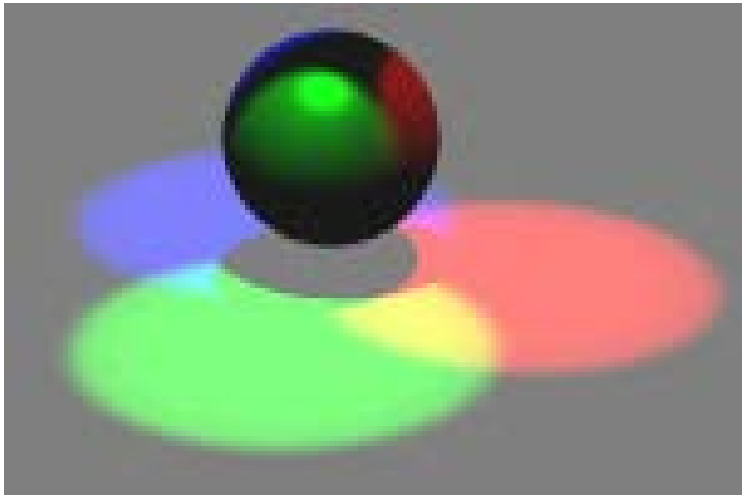


How Can We Add Shadows?

```
color = ambient*hit->getMaterial()->getDiffuseColor()
for every light
   Ray ray2(hitPoint, directionToLight)
   Hit hit2(distanceToLight, NULL, NULL)
   For every object
      object->intersect(ray2, hit2, 0)
   if (hit2->getT() = distanceToLight)
      color += hit->getMaterial()->Shade
                   (ray, hit, directionToLight, lightColor)
return color
```

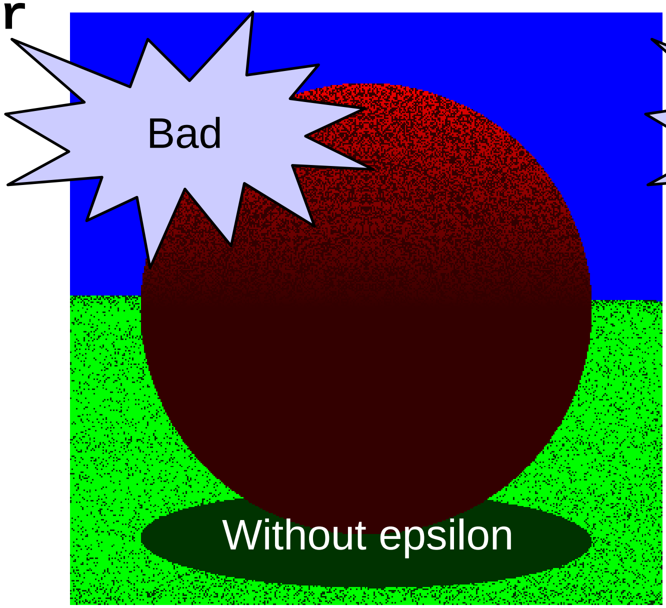
ambient = k_a
diffuseColor = k_d





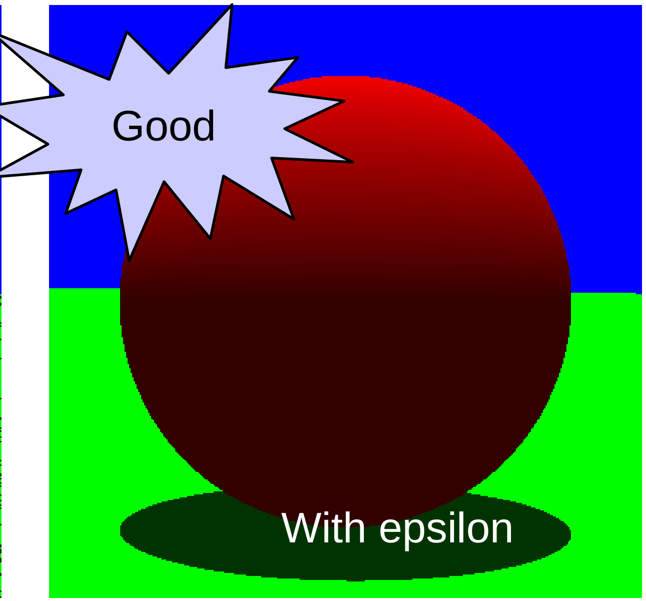
Problem: Self-Shadowing

```
color = ambient*hit->getMaterial()->getDiffuseColor()
for every light
   Ray ray2(hitPoint, directionToLight)
   Hit hit2(distanceToLight, NULL, NULL)
   For every object
      object->intersect(ray2, hit2, epsilon)
   if (hit2->getT() = distanceToLight)
      color += hit->getMaterial()->Shade
                 (ray, hit, directionToLight, lightColor)
return color
```



Bad

Without epsilon

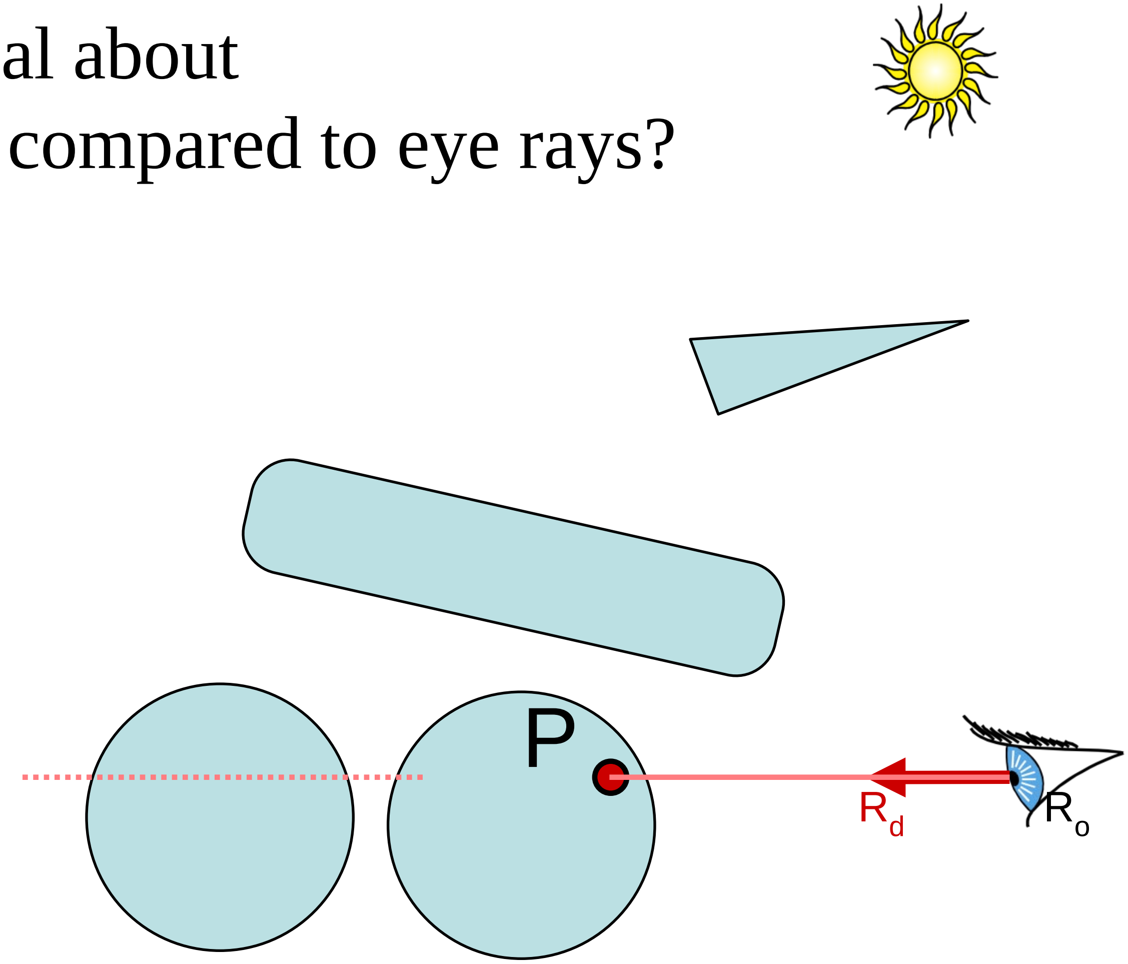


Good

With epsilon

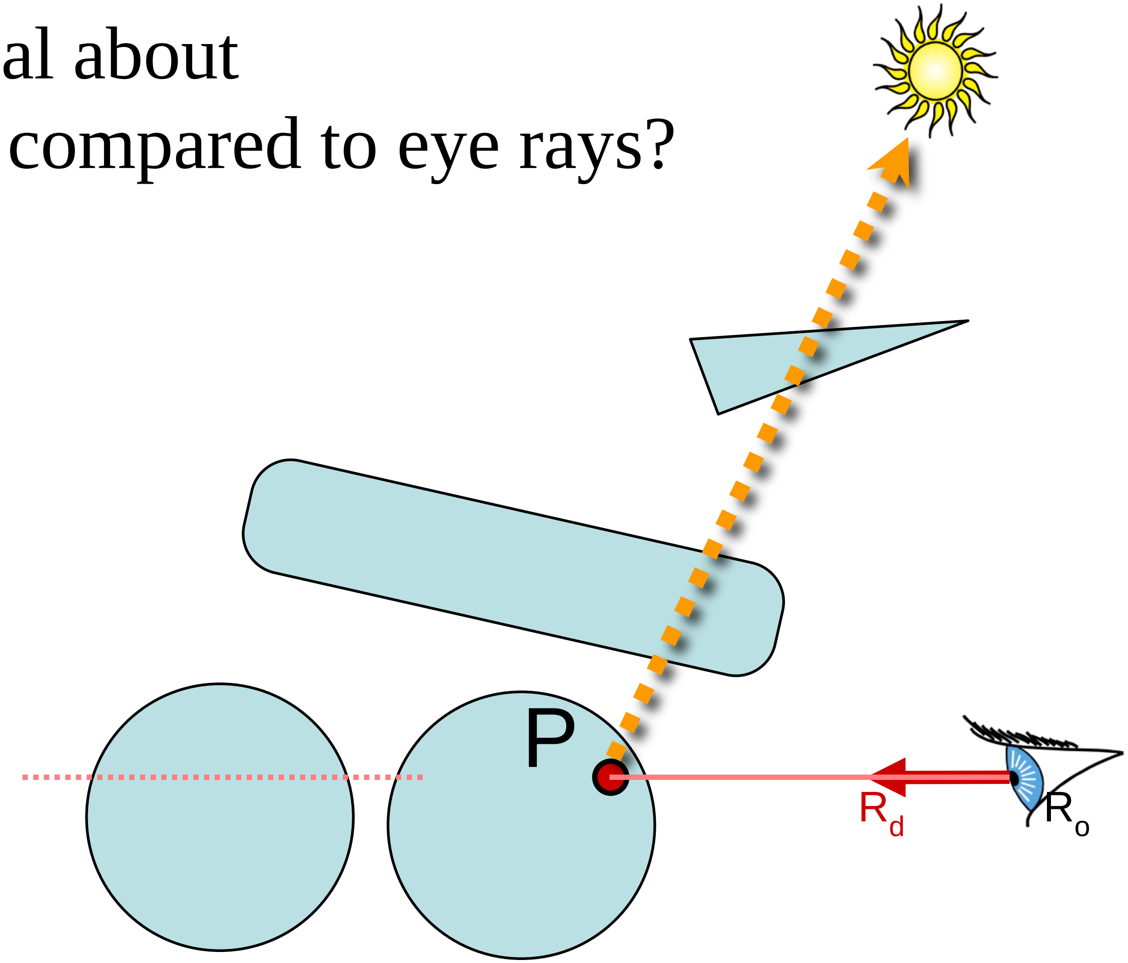
Let’s Think About Shadow Rays

- What’s special about shadow rays compared to eye rays?



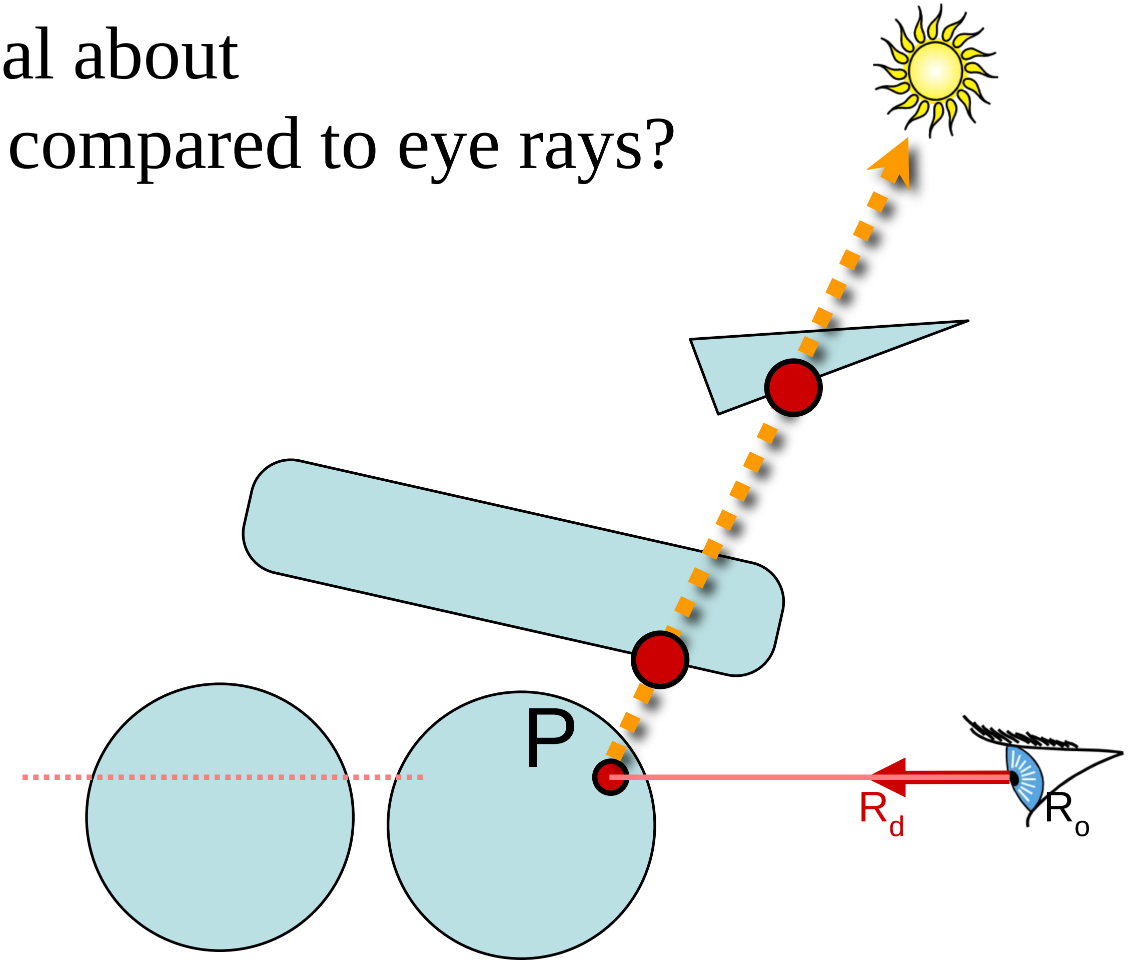
Let’s Think About Shadow Rays

- What’s special about shadow rays compared to eye rays?



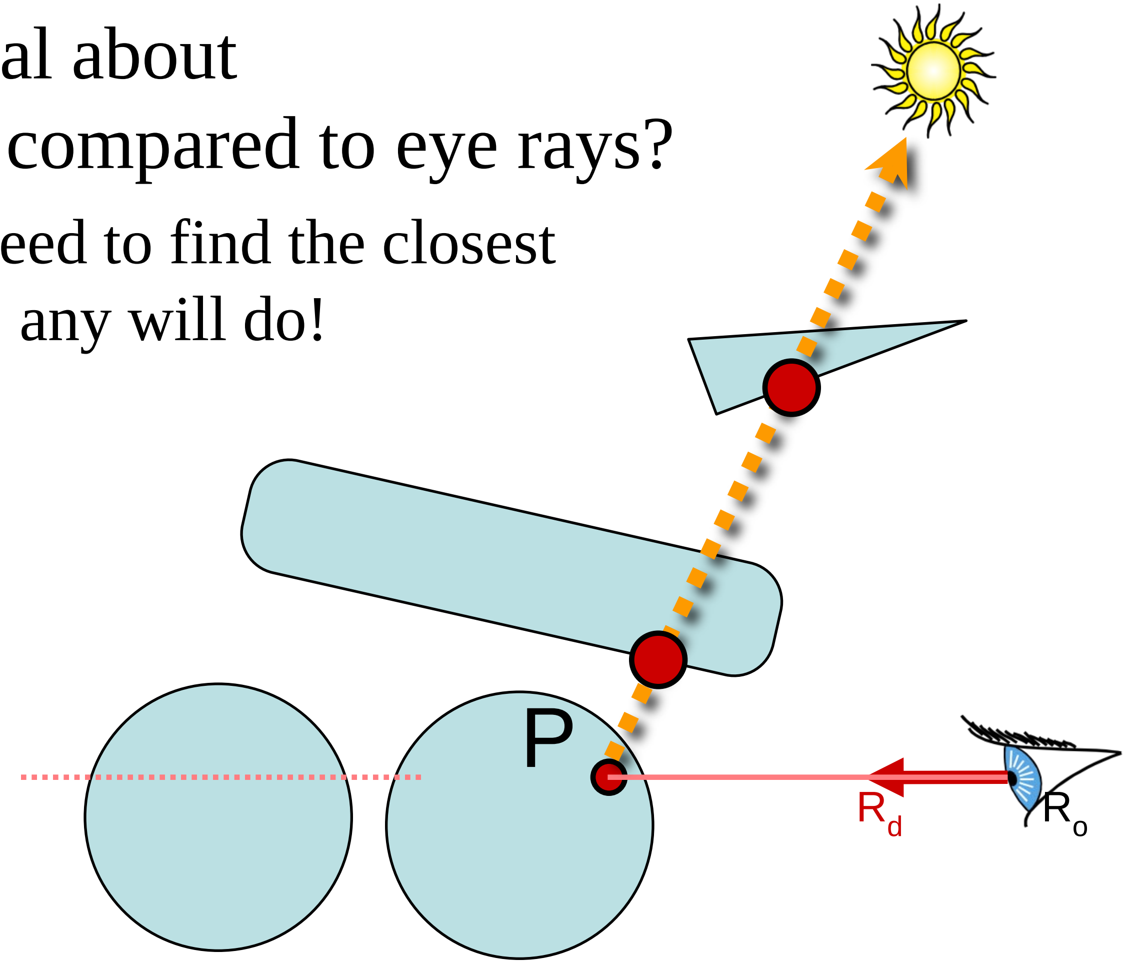
Let’s Think About Shadow Rays

- What’s special about shadow rays compared to eye rays?



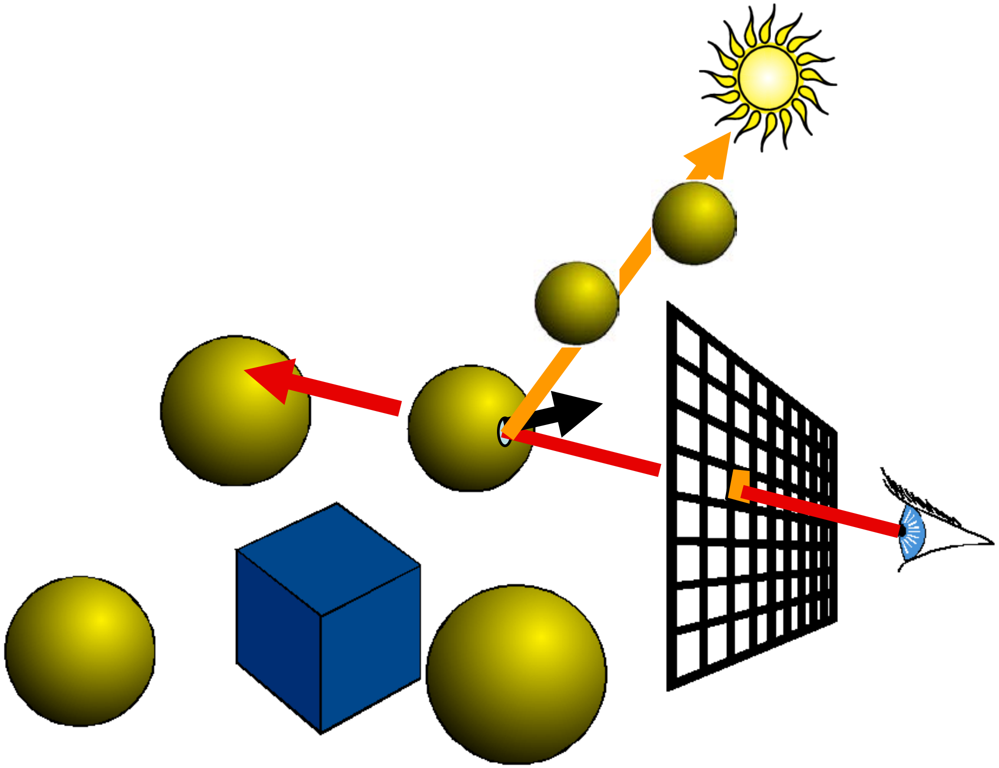
Let’s Think About Shadow Rays

- What’s special about shadow rays compared to eye rays?
  - We do not need to find the closest intersection, any will do!

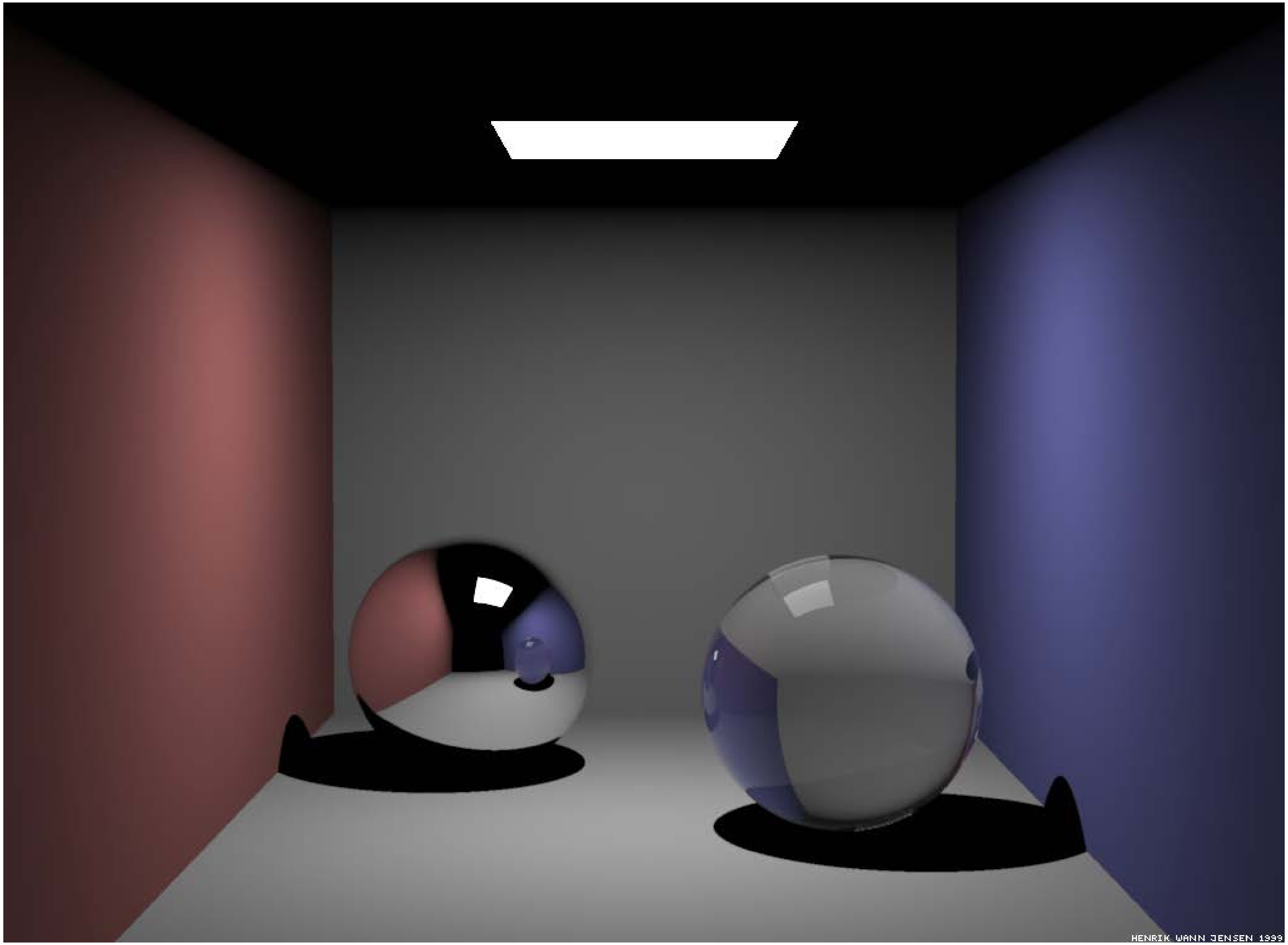


Shadow Optimization

- We only want to know whether there is an intersection, *not* which one is closest
- Special routine `Object3D::intersectShadowRay()`
  - Stops at first intersection



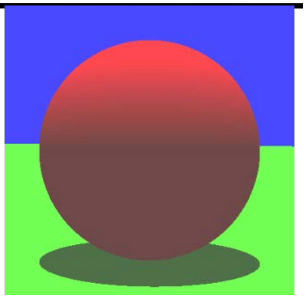
Questions?



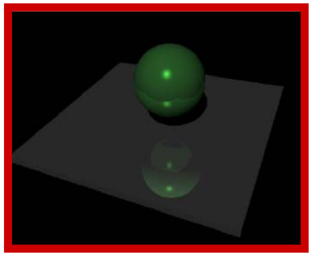
Henrik Wann Jensen

Overview of Today

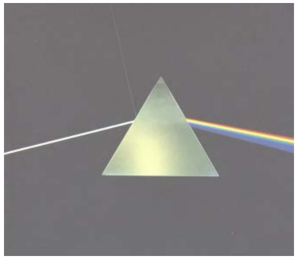
- Shadows



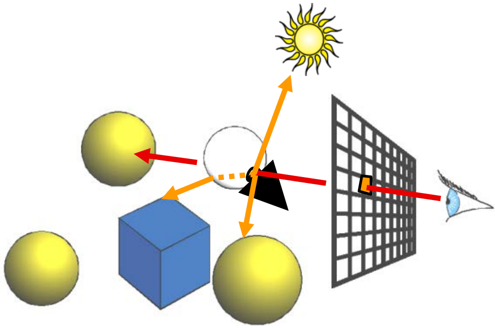
- Reflection



- Refraction

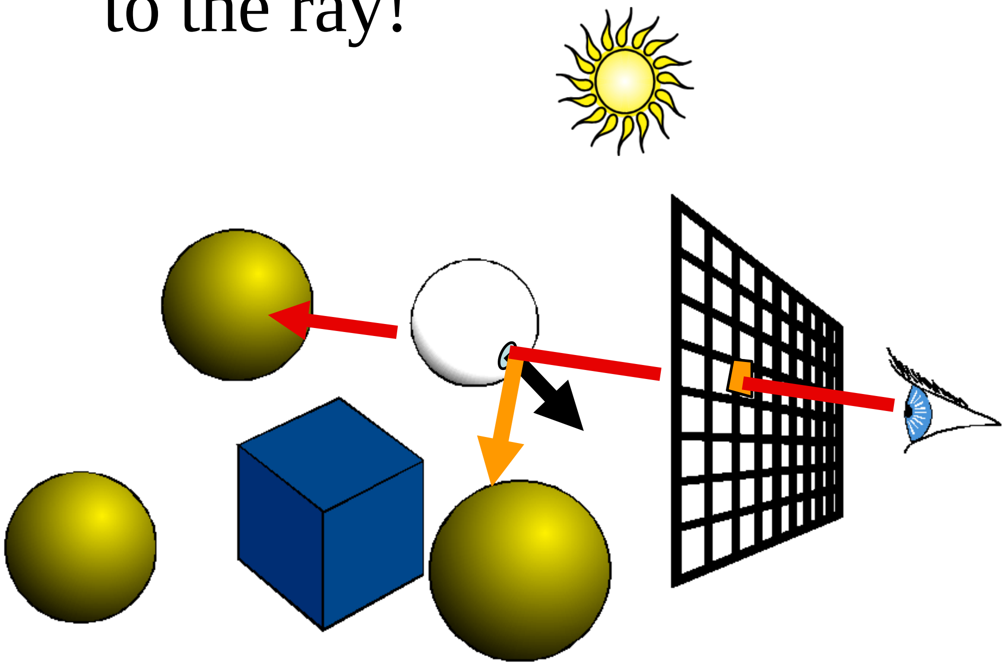


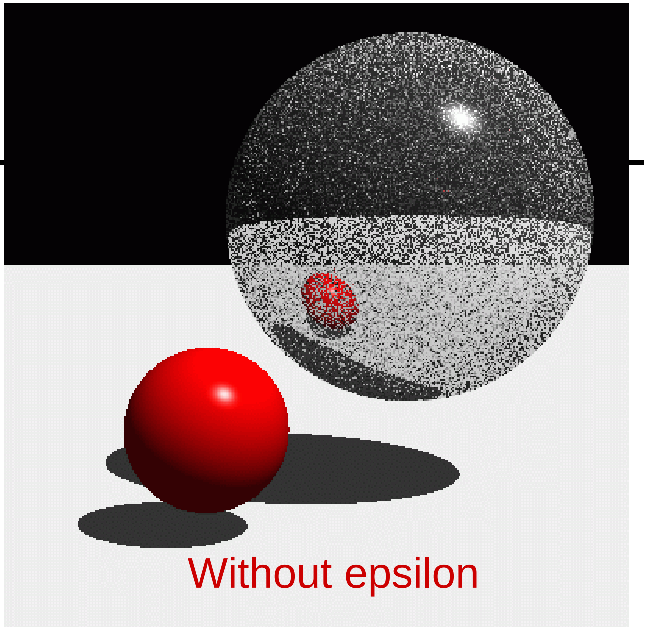
- Recursive Ray Tracing



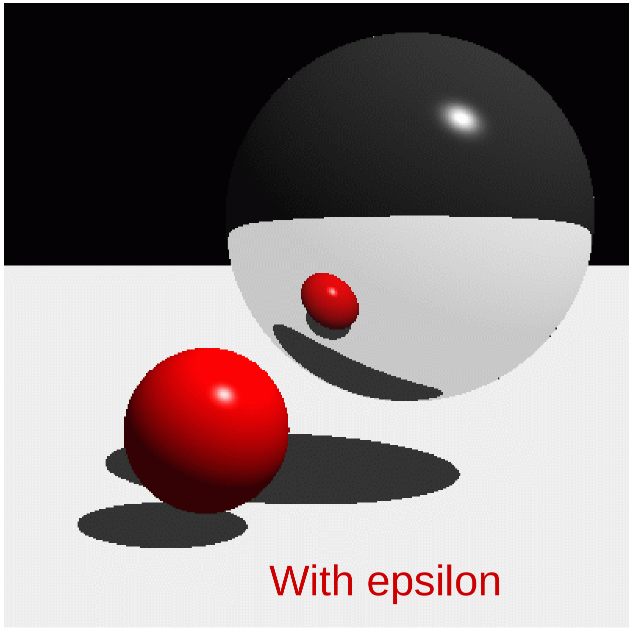
Mirror Reflection

- Cast ray symmetric with respect to the normal
- Multiply by reflection coefficient k_s (color)
- Don’t forget to add epsilon to the ray!





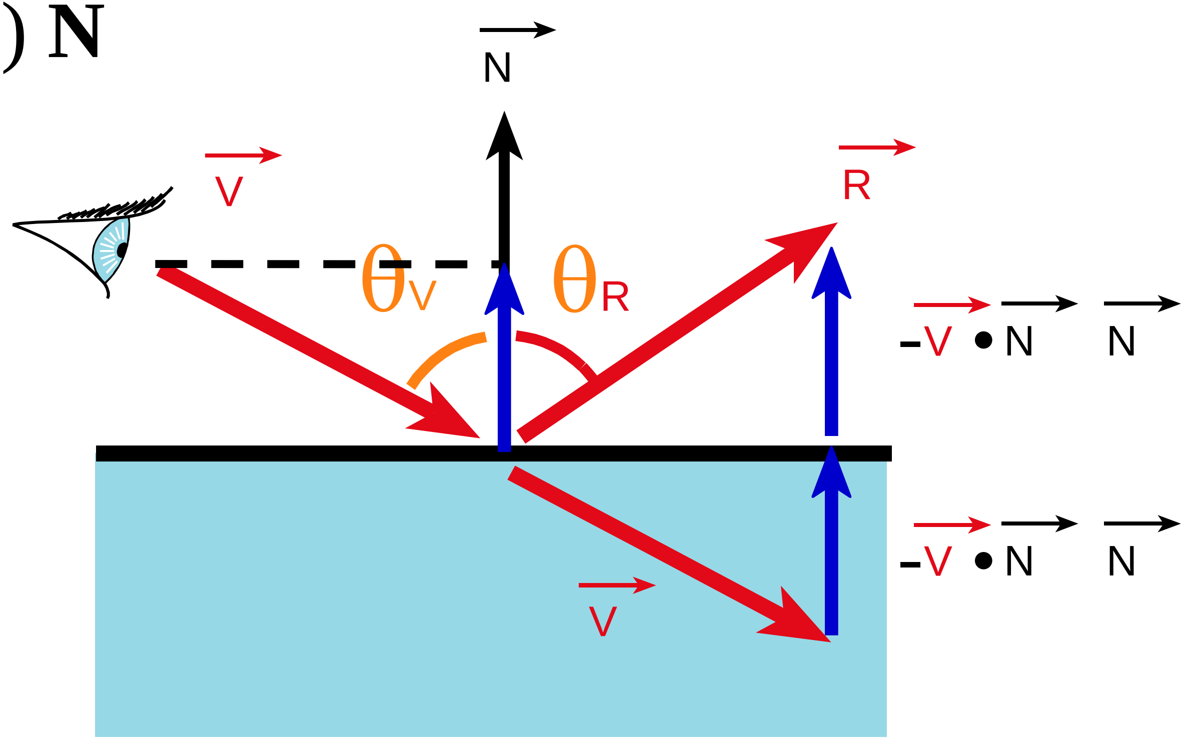
Without epsilon



With epsilon

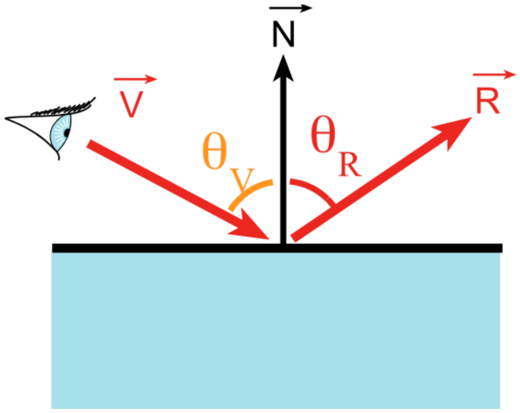
Perfect Mirror Reflection

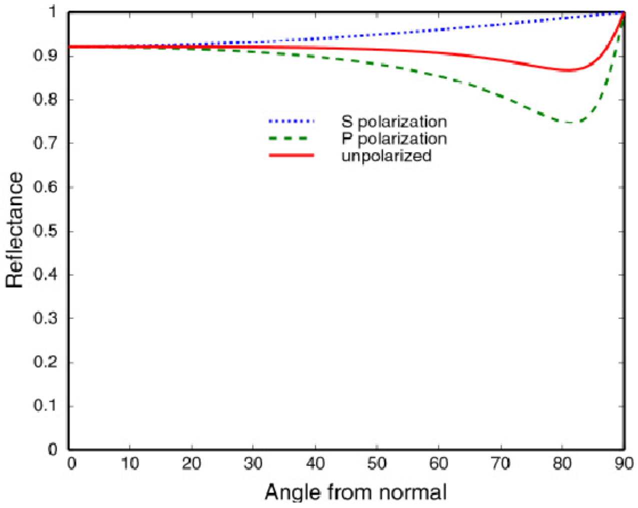
- Reflection angle = view angle
  - Normal component is negated
  - Remember particle collisions?
- **R** = **V** – 2 (**V** · **N**) **N**


$\vec{N}$
$\vec{V}$
$\vec{R}$
θ_V
θ_R
$-\vec{V} \bullet \vec{N}$ $\vec{N}$
$-\vec{V} \bullet \vec{N}$ $\vec{N}$
$\vec{V}$

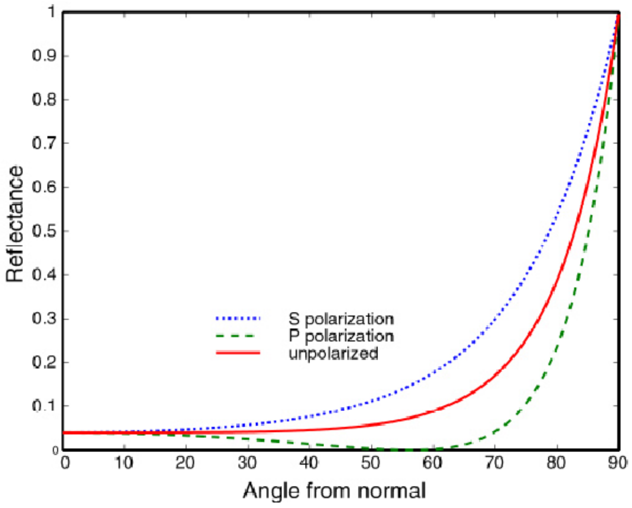
Amount of Reflection

- Traditional ray tracing (hack)
  - Constant k_s
- More realistic (we'll do this later):
  - Fresnel reflection term (more reflection at grazing angle)
  - Schlick's approximation: $R(\theta)=R_0+(1-R_0)(1-\cos\theta)^5$
- Fresnel makes a big difference!





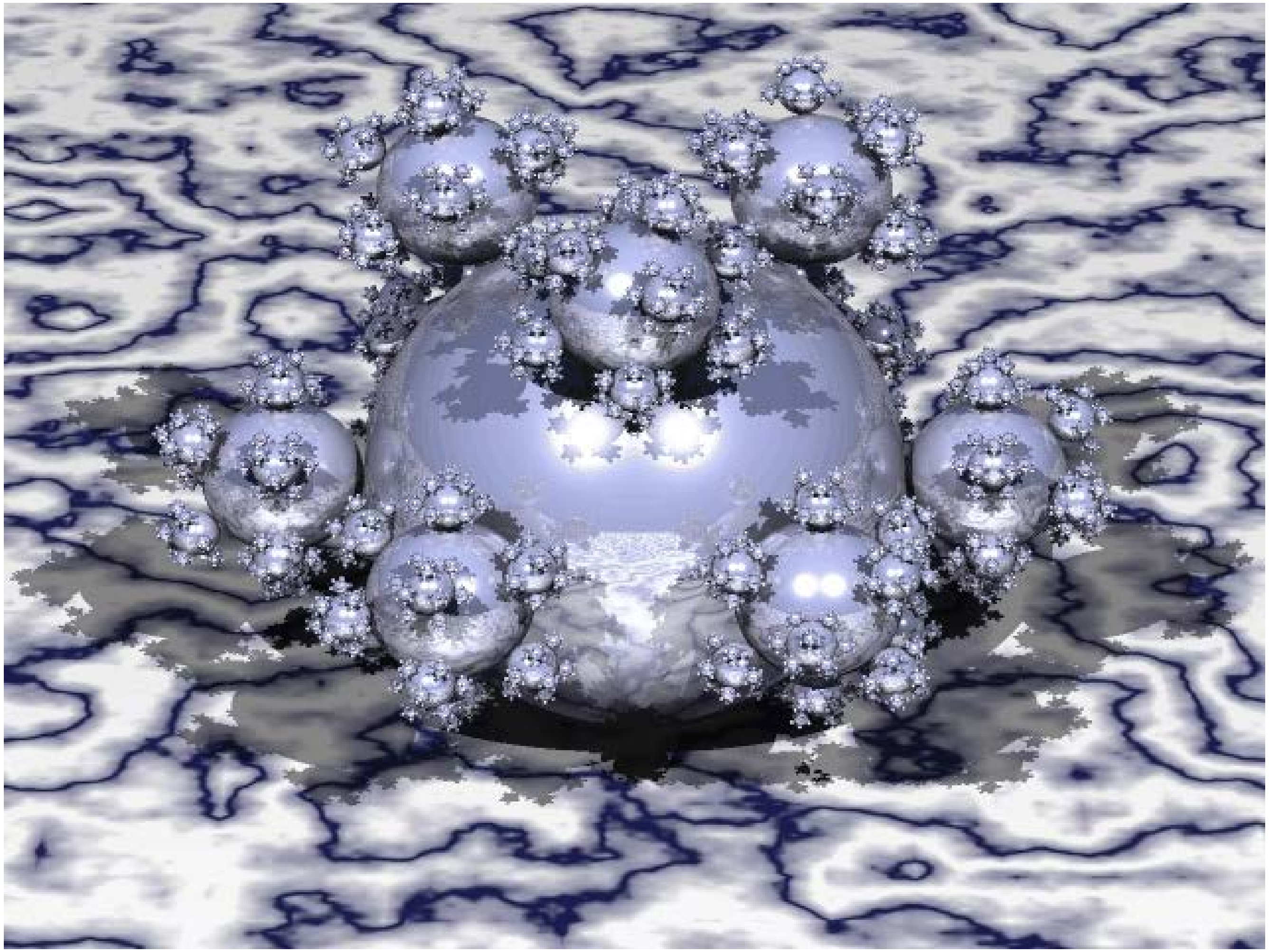
metal



Dielectric (glass)

Questions?

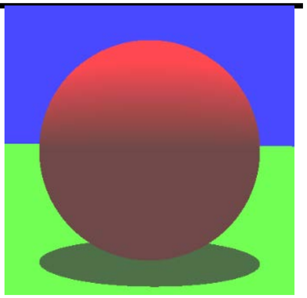
"Sphereflake" fractal



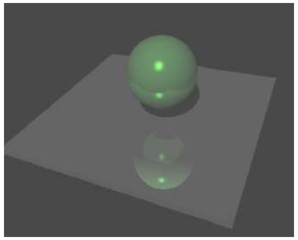
Henrik Wann Jensen

Overview of Today

- Shadows

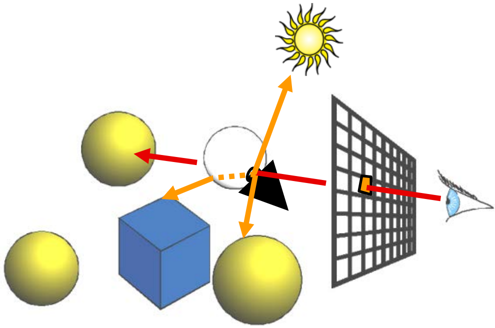


- Reflection



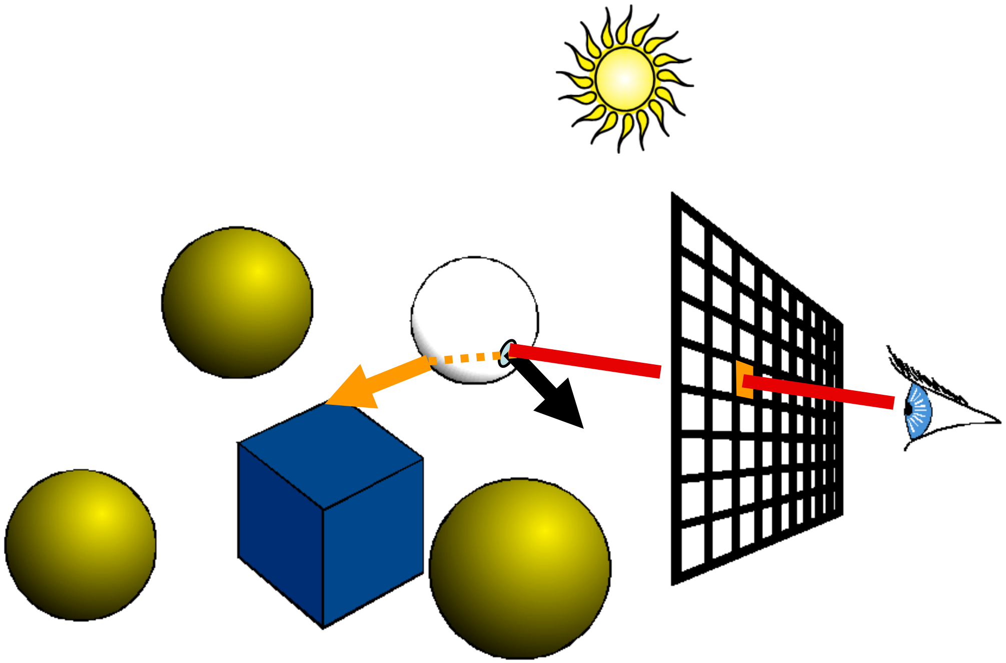
- Refraction

- Recursive Ray Tracing

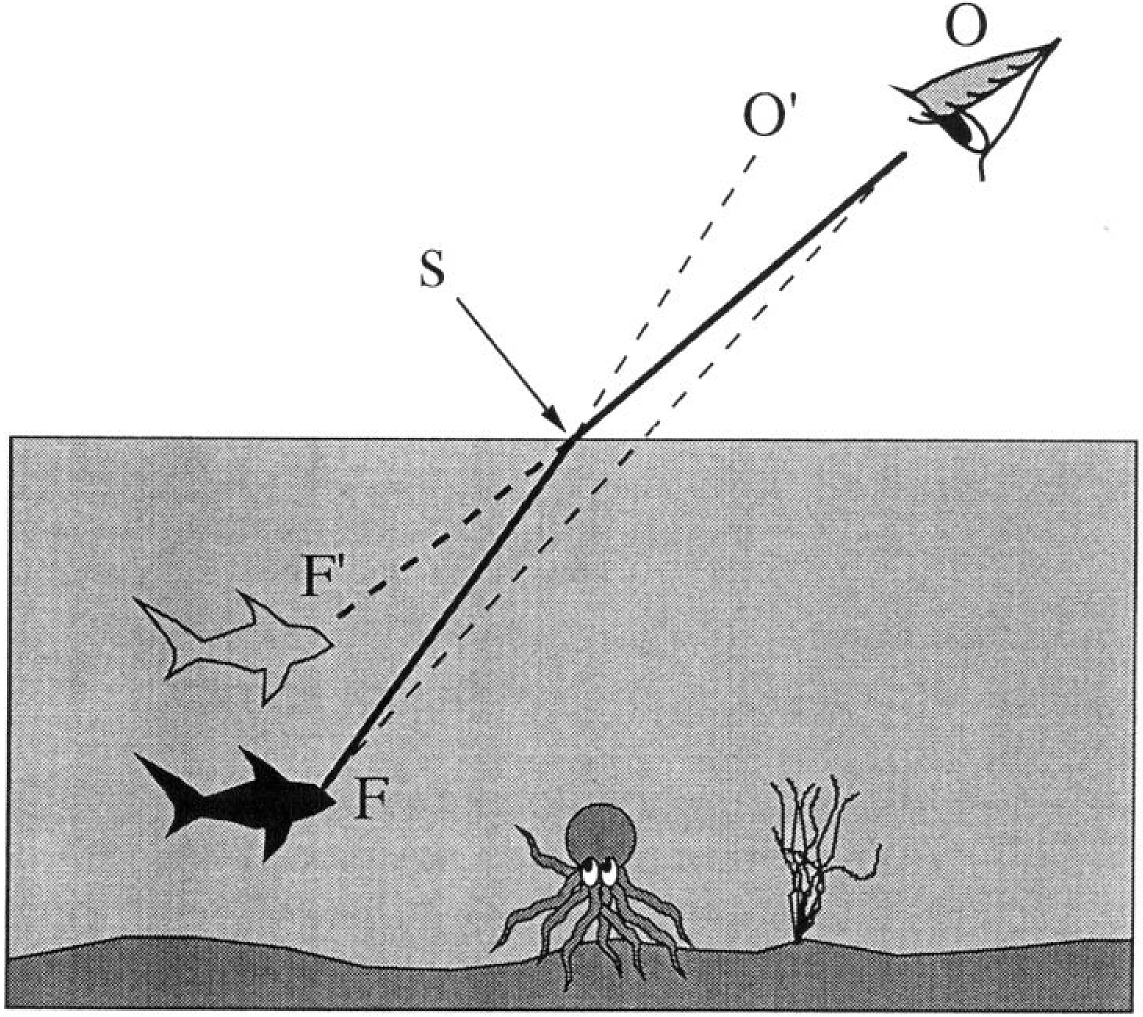


Transparency (Refraction)

- Cast ray in refracted direction
- Multiply by transparency coefficient k_t (color)

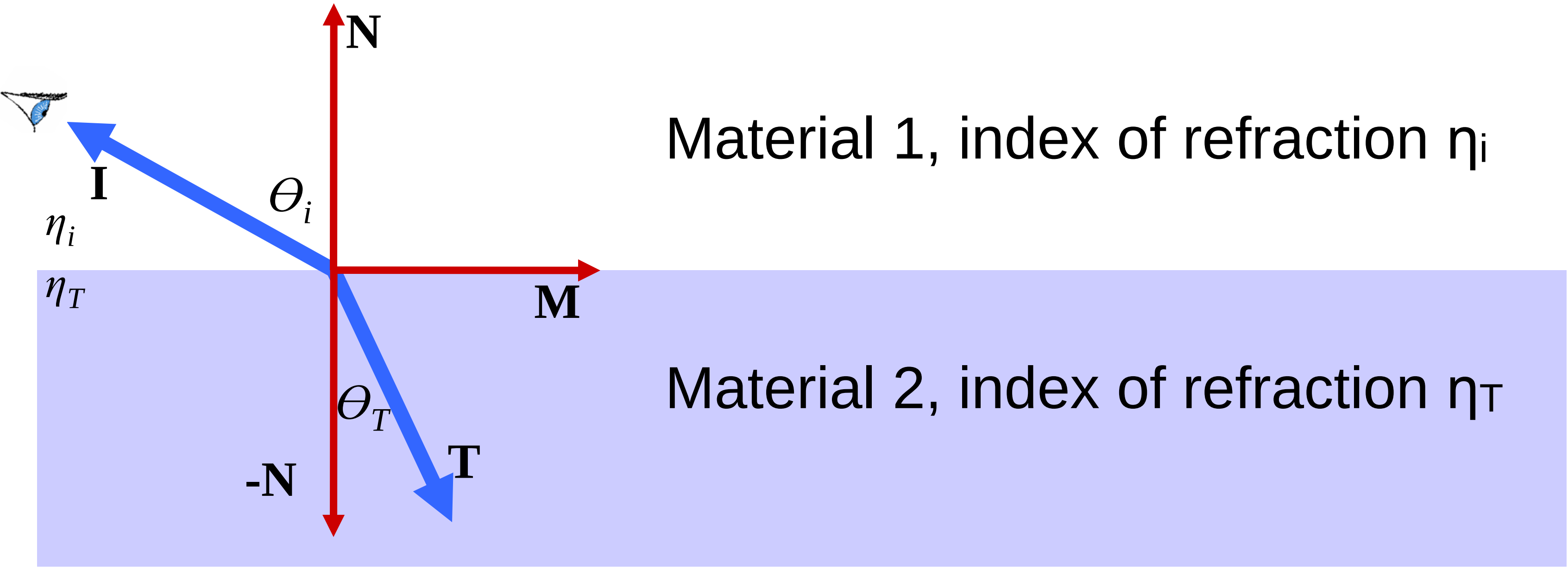


Qualitative Refraction



From “Color and Light in Nature” by Lynch and Livingston

Refraction



Material 1, index of refraction ηi

Material 2, index of refraction ηT

Snell-Descartes Law:

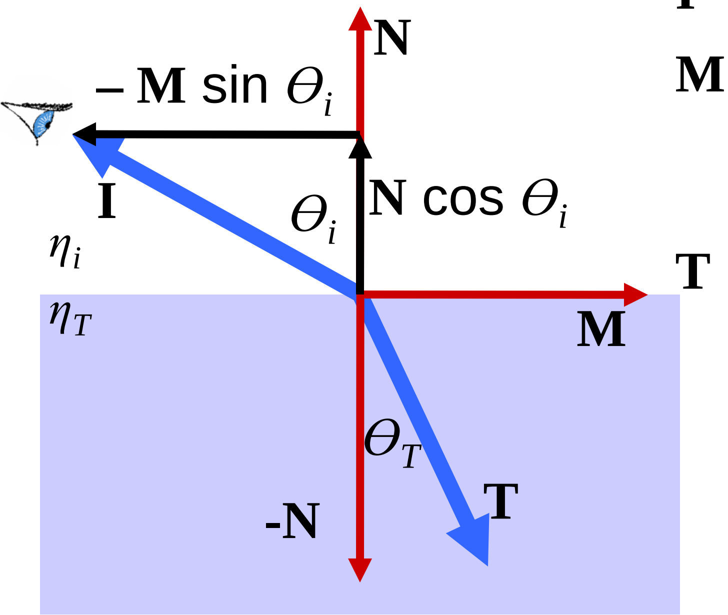
$$n_i \sin\theta_i = n_T \sin\theta_T$$

$$\frac{\sin\theta_T}{\sin\theta_i} = \frac{n_i}{n_T} = n_r$$

Refracted direction **T**?

Relative index of refraction

Refraction



$$\mathbf{I} = \mathbf{N} \cos \Theta_i - \mathbf{M} \sin \Theta_i$$

$$\mathbf{M} = (\mathbf{N} \cos \Theta_i - \mathbf{I}) / \sin \Theta_i$$

$$\begin{aligned}
\mathbf{T} &= - \mathbf{N} \cos \Theta_T + \mathbf{M} \sin \Theta_T \\
&= - \mathbf{N} \cos \Theta_T + (\mathbf{N} \cos \Theta_i - \mathbf{I}) \sin \Theta_T / \sin \Theta_i \\
&= - \mathbf{N} \cos \Theta_T + (\mathbf{N} \cos \Theta_i - \mathbf{I}) \, \eta_r \\
&= [\, \eta_r \cos \Theta_i - \cos \Theta_T \,] \, \mathbf{N} - \eta_r \, \mathbf{I} \\
&= [\, \eta_r \cos \Theta_i - \sqrt{1 - \sin^2 \Theta_T} \,] \, \mathbf{N} - \eta_r \, \mathbf{I} \\
&= [\, \eta_r \cos \Theta_i - \sqrt{1 - \eta_r^2 \sin^2 \Theta_i} \,] \, \mathbf{N} - \eta_r \, \mathbf{I} \\
&= [\, \eta_r \cos \Theta_i - \sqrt{1 - \eta_r^2 (1 - \cos^2 \Theta_i)} \,] \, \mathbf{N} - \eta_r \, \mathbf{I} \\
&= [\, \eta_r (\mathbf{N} \cdot \mathbf{I}) - \sqrt{1 - \eta_r^2 (1 - (\mathbf{N} \cdot \mathbf{I})^2)} \,] \, \mathbf{N} - \eta_r \, \mathbf{I}
\end{aligned}$$

Plug M

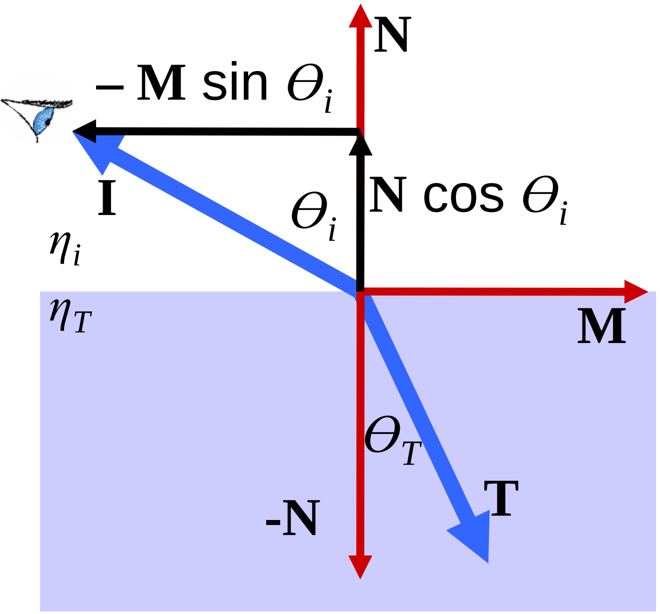
let's get rid of the cos & sin

Snell-Descartes Law:

$$n_i \sin \theta_i = n_T \sin \theta_T$$

$$\frac{\sin \theta_T}{\sin \theta_i} = \frac{n_i}{n_T} = n_r$$

Refraction



$$\mathbf{I} = \mathbf{N} \cos \Theta_i - \mathbf{M} \sin \Theta_i$$

$$\mathbf{M} = (\mathbf{N} \cos \Theta_i - \mathbf{I}) / \sin \Theta_i$$

- **Total internal reflection** when the square root is imaginary (no refraction, just reflection)

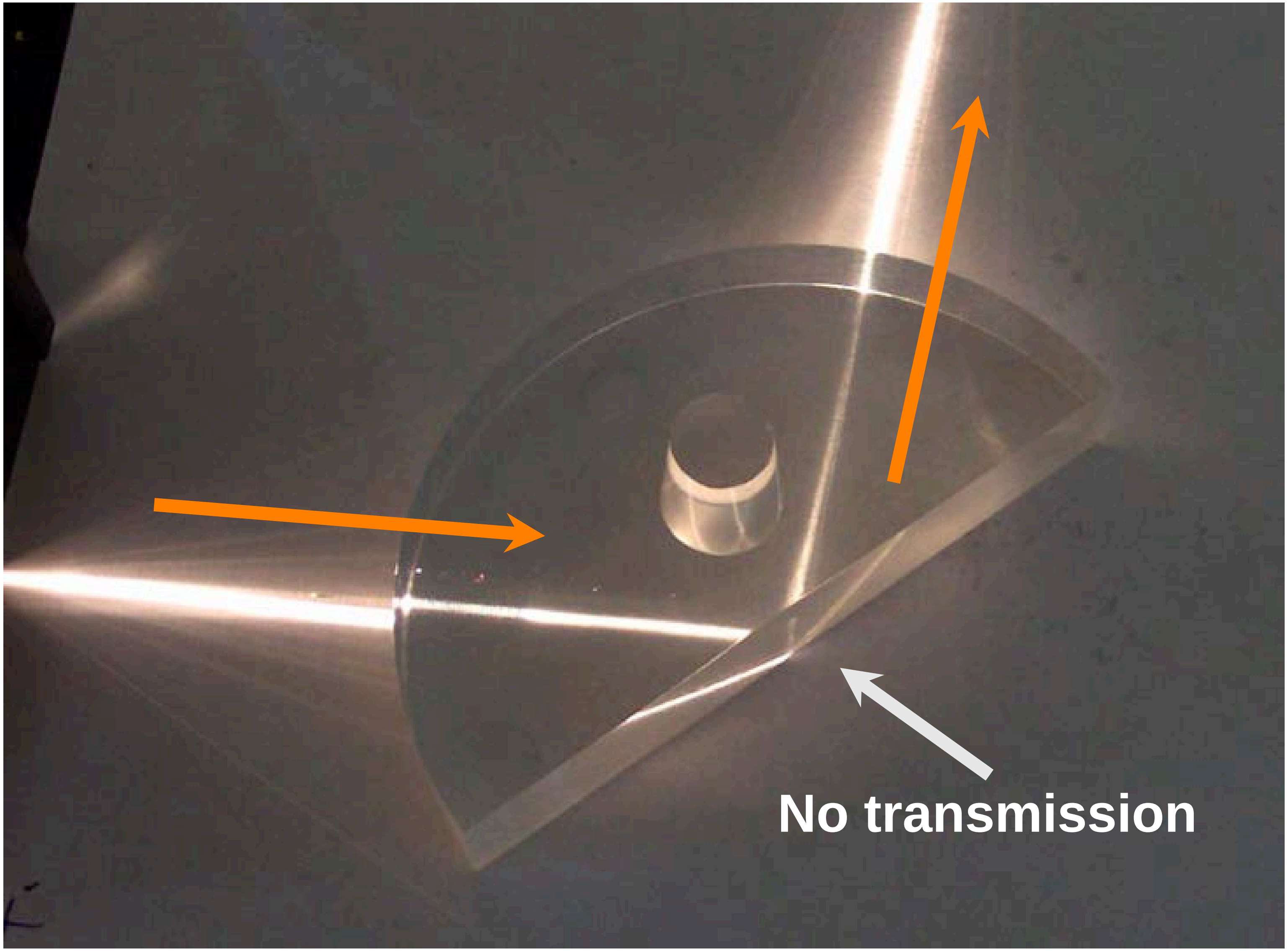
Snell-Descartes Law:

$$n_i \sin \theta_i = n_T \sin \theta_T$$

$$\frac{\sin \theta_T}{\sin \theta_i} = \frac{n_i}{n_T} = n_r$$

$$= [\, \eta_r (\mathbf{N} \cdot \mathbf{I}) - \sqrt{1 - \eta_r{}^2 (1 - (\mathbf{N} \cdot \mathbf{I})^2)}\,]\, \mathbf{N} - \eta_r \mathbf{I}$$

Total Internal Reflection



Wikipedia user Fir0002

Total Internal Reflection

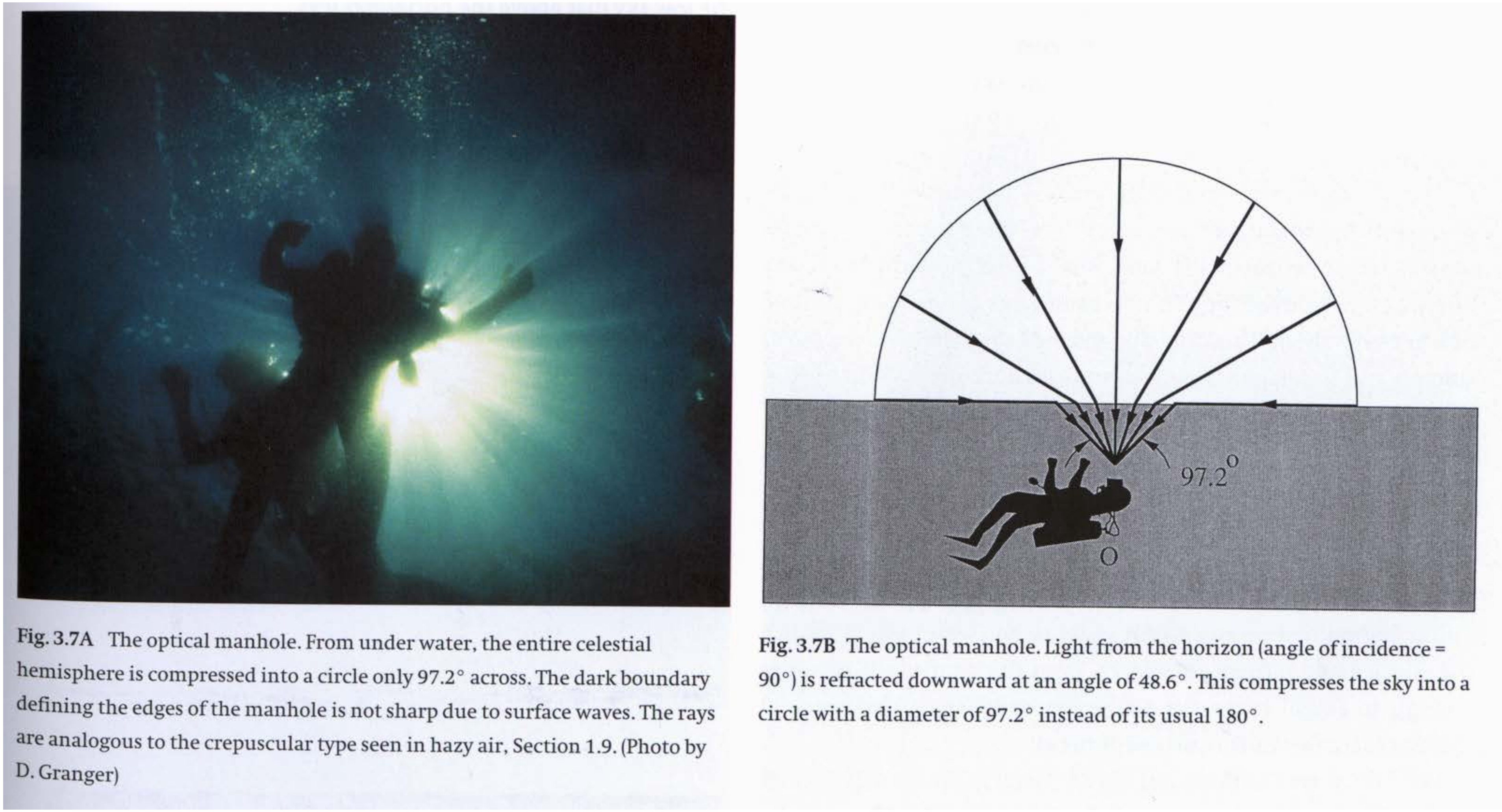


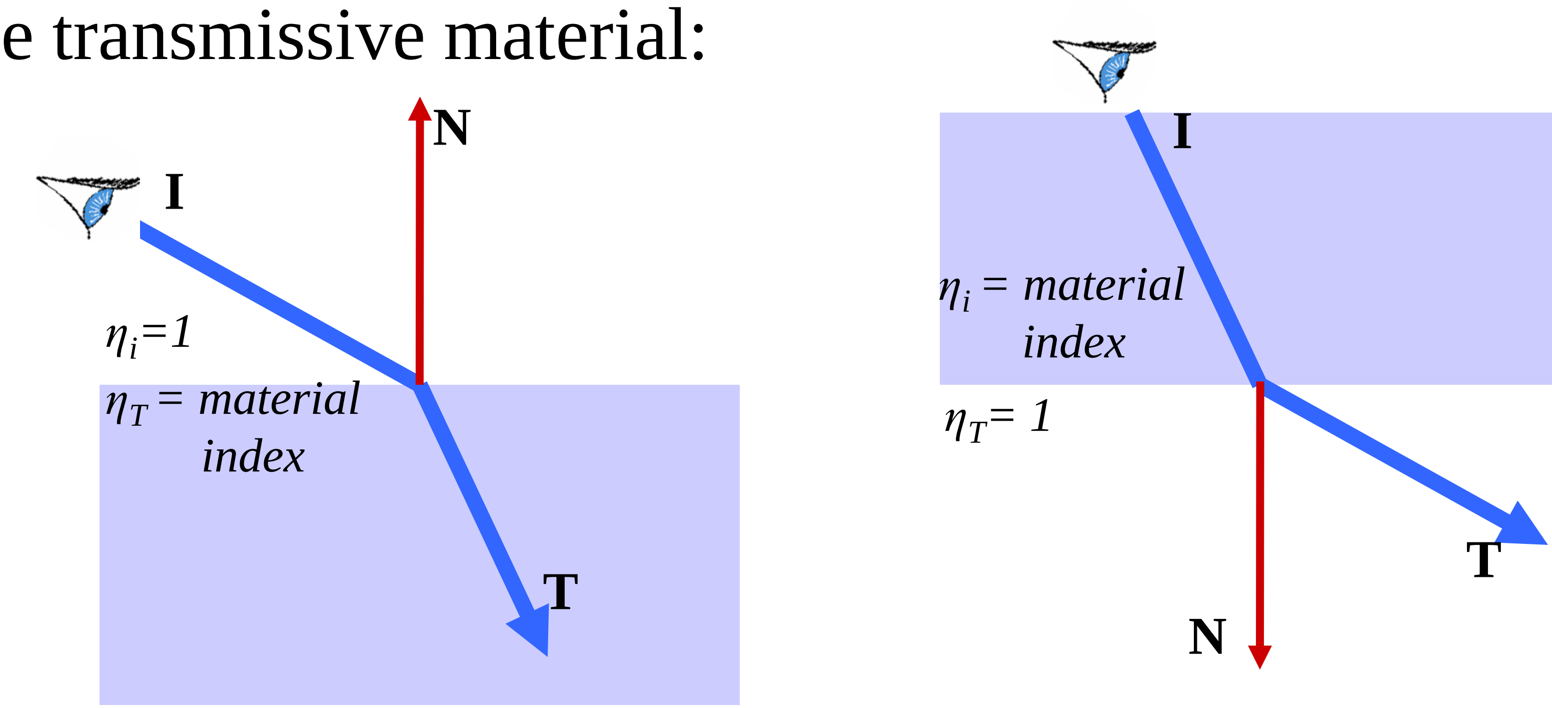
Fig. 3.7A The optical manhole. From under water, the entire celestial hemisphere is compressed into a circle only 97.2° across. The dark boundary defining the edges of the manhole is not sharp due to surface waves. The rays are analogous to the crepuscular type seen in hazy air, Section 1.9. (Photo by D. Granger)

Fig. 3.7B The optical manhole. Light from the horizon (angle of incidence = 90°) is refracted downward at an angle of 48.6°. This compresses the sky into a circle with a diameter of 97.2° instead of its usual 180°.

From “Color and Light in Nature” by Lynch and Livingston

Refraction & Sidedness of Objects

- Make sure you know whether you're entering or leaving the transmissive material:



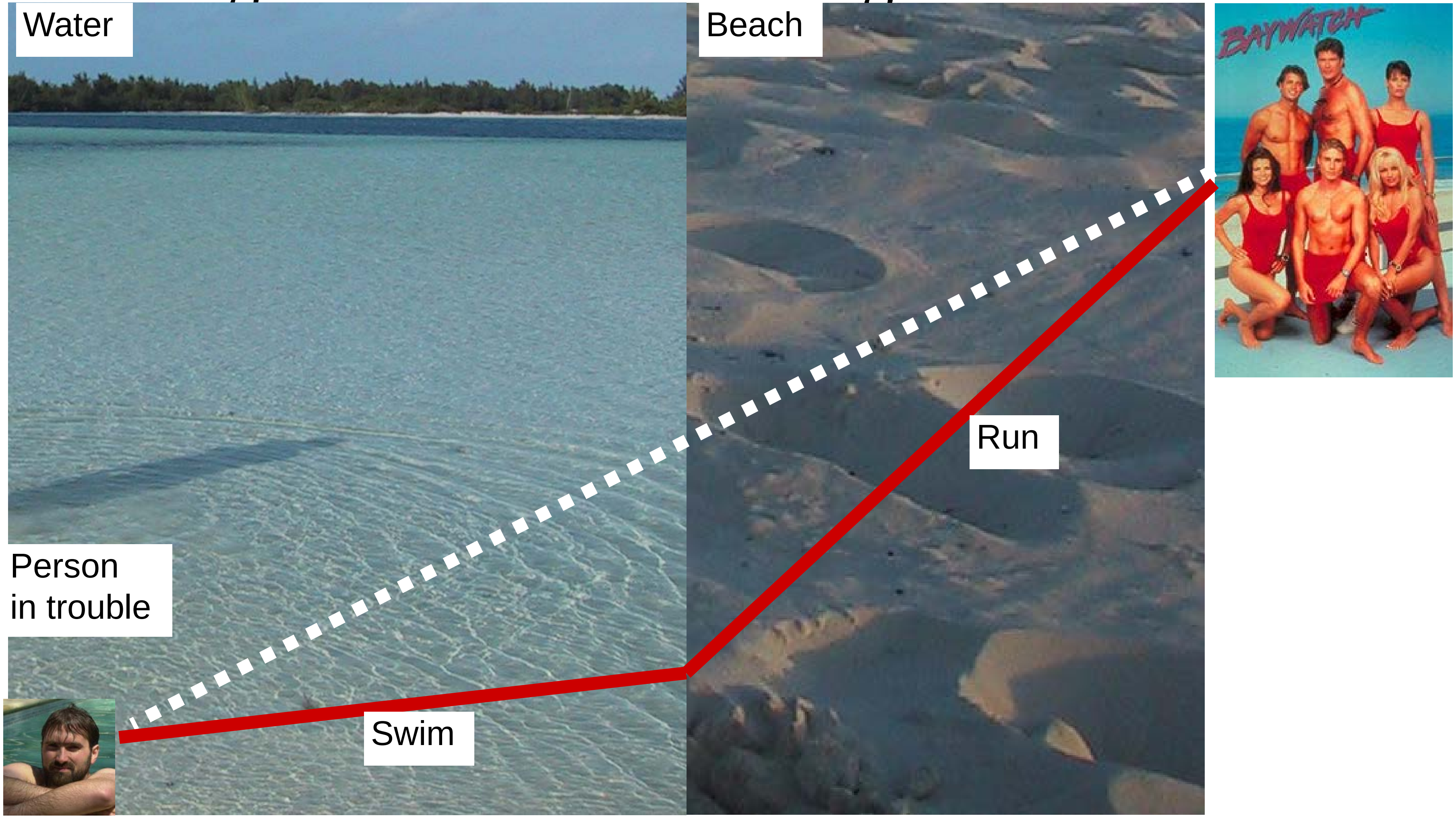
- Note: We won't ask you to trace rays through intersecting transparent objects **:-)**

Cool Refraction Demo

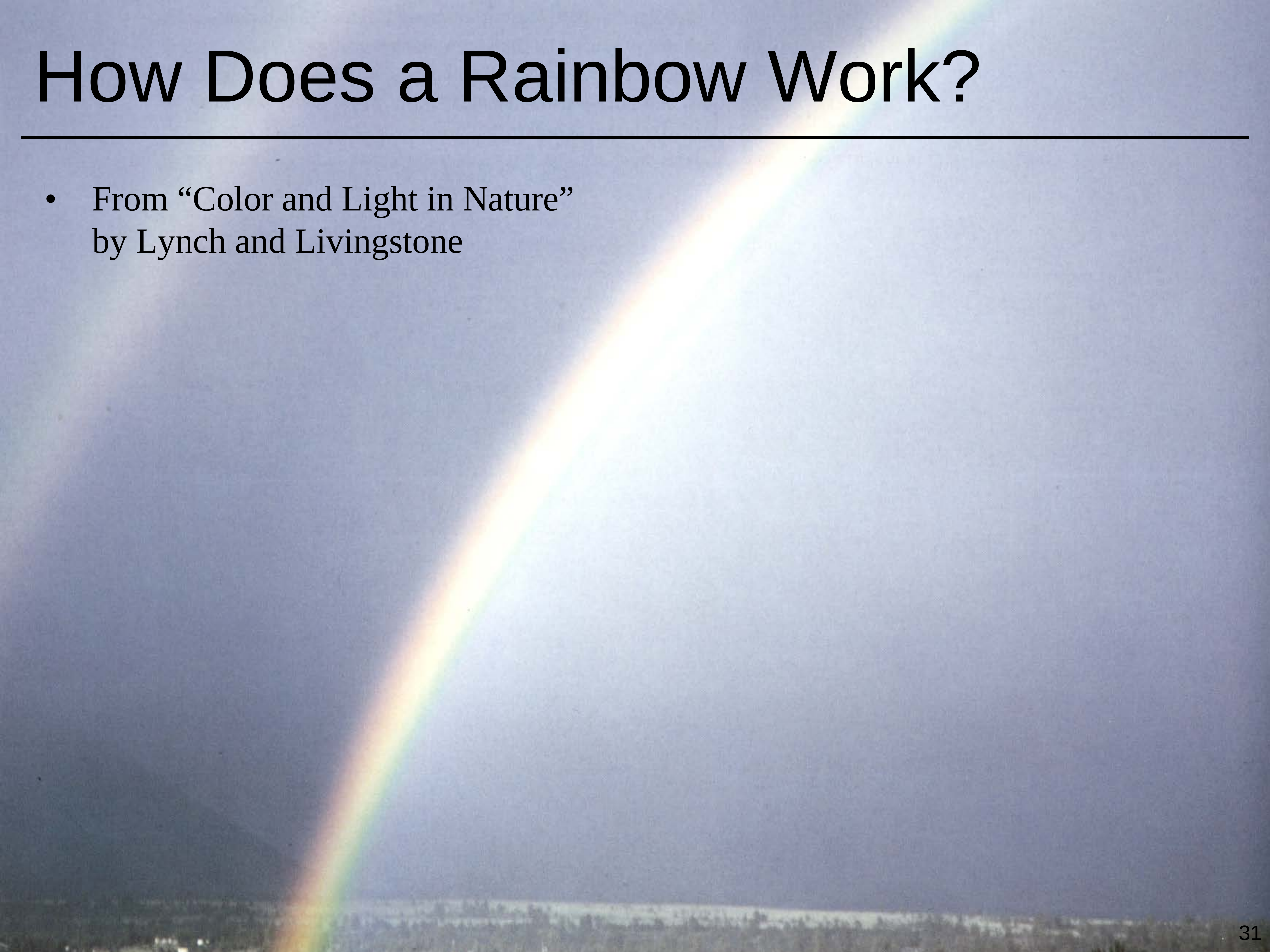
- Enright, D., Marschner, S. and Fedkiw, R., SIGGRAPH 2002

Refraction and the Lifeguard Problem

- Running is faster than swimming



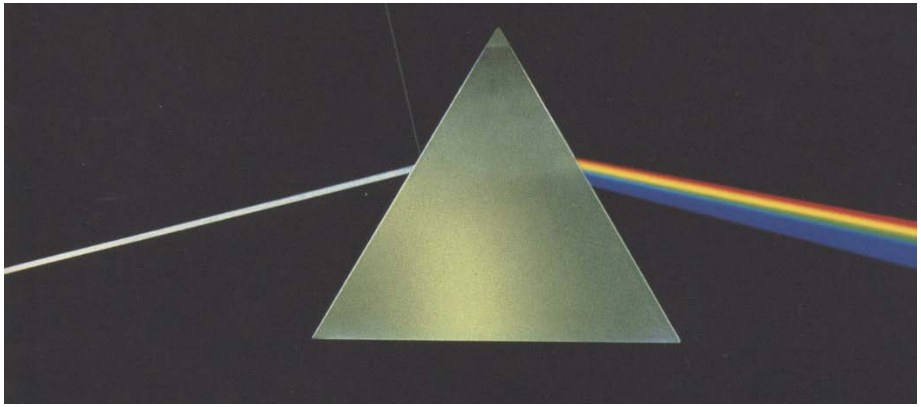
How Does a Rainbow Work?



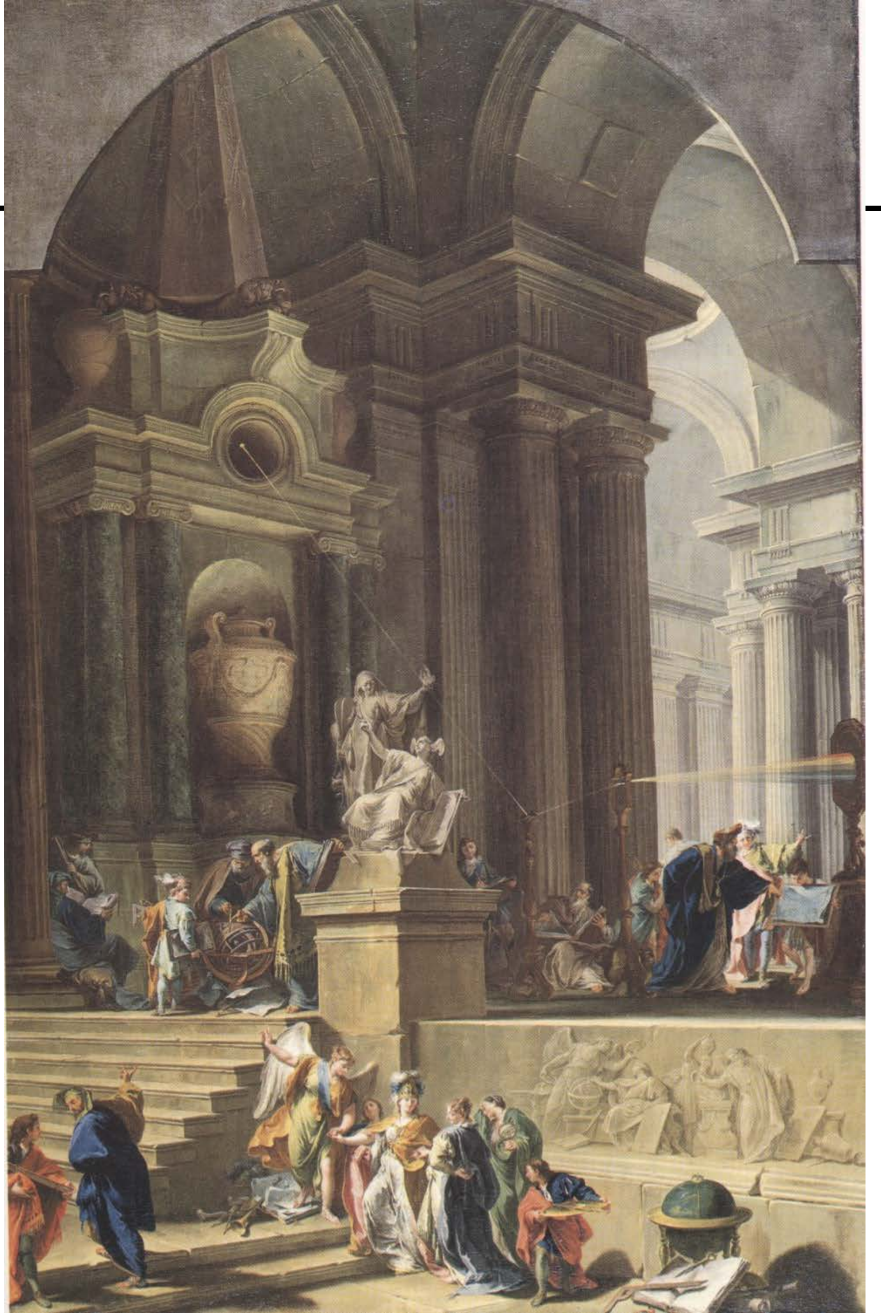
- From “Color and Light in Nature”
  by Lynch and Livingstone

Wavelength

- Refraction is wavelength-dependent (dispersion)
  - Refraction increases as the wavelength of light decreases
  - violet and blue experience more bending than orange and red
- Newton's prism experiment
- **Usually ignored in graphics**



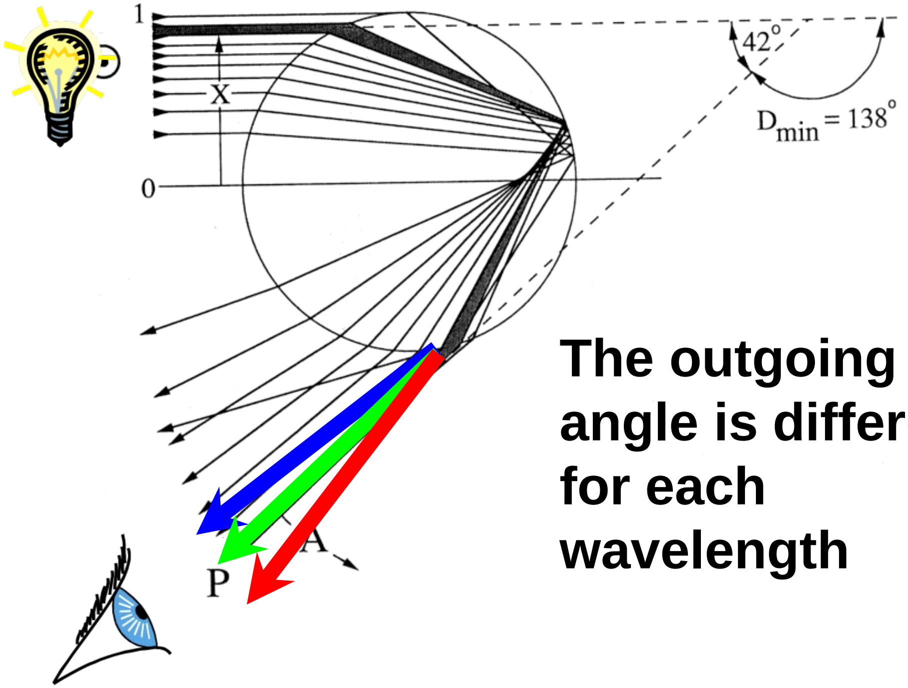
Pink Floyd, *The Dark Side of the Moon*



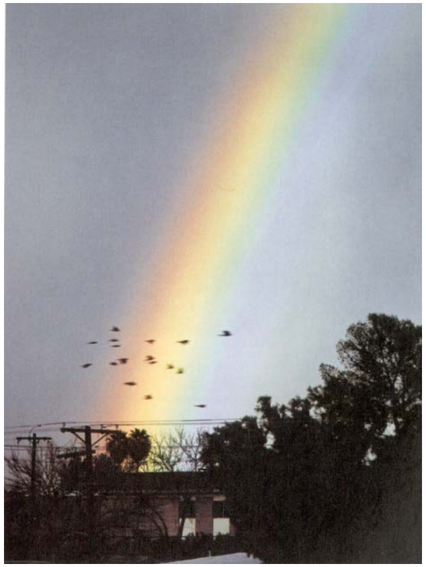
Pittoni, 1725, Allegory to Newton

Rainbow

- Rainbow is caused by refraction + internal reflection + refraction
- Maximum for angle around 42 degrees
- Refraction depends on wavelength (dispersion)



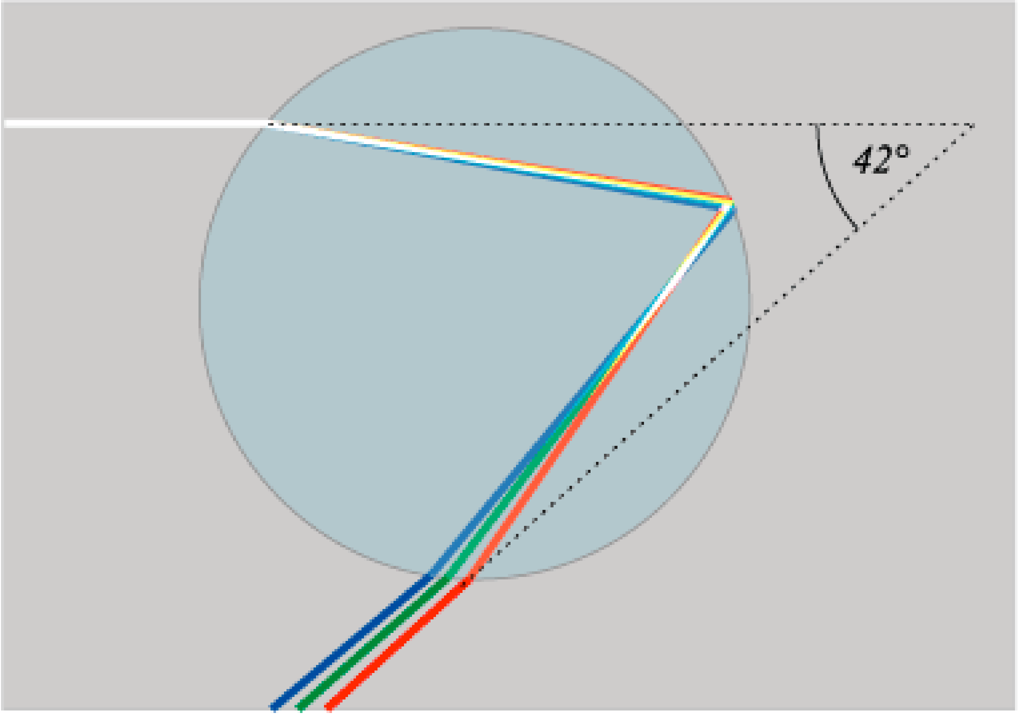
The outgoing angle is different for each wavelength



“Color and Light in Nature” by Lynch and Livingstone

Rainbow

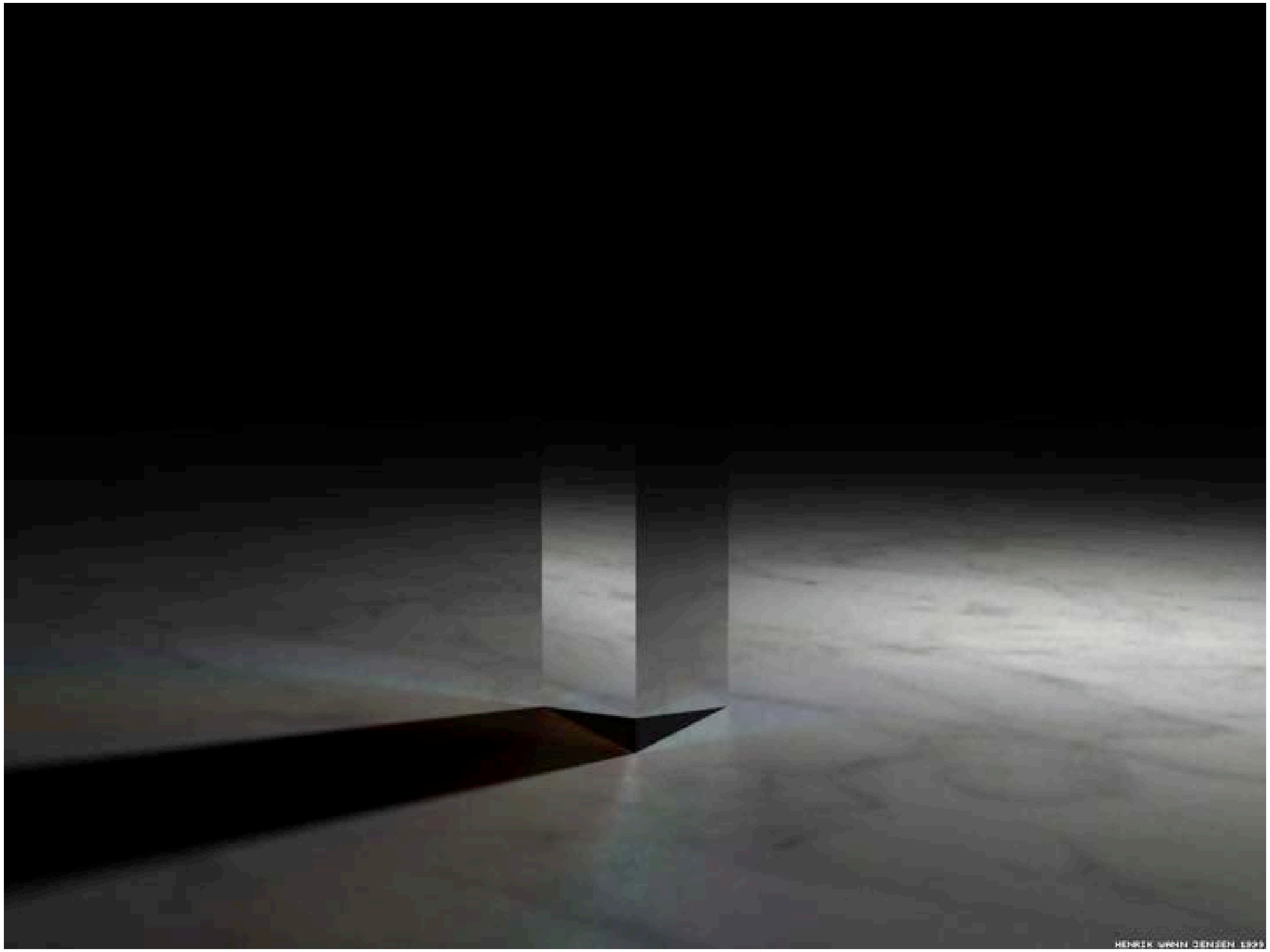
- Rainbow is caused by refraction + internal reflection + refraction
- Maximum for angle around 42 degrees
- Refraction depends on wavelength (dispersion)



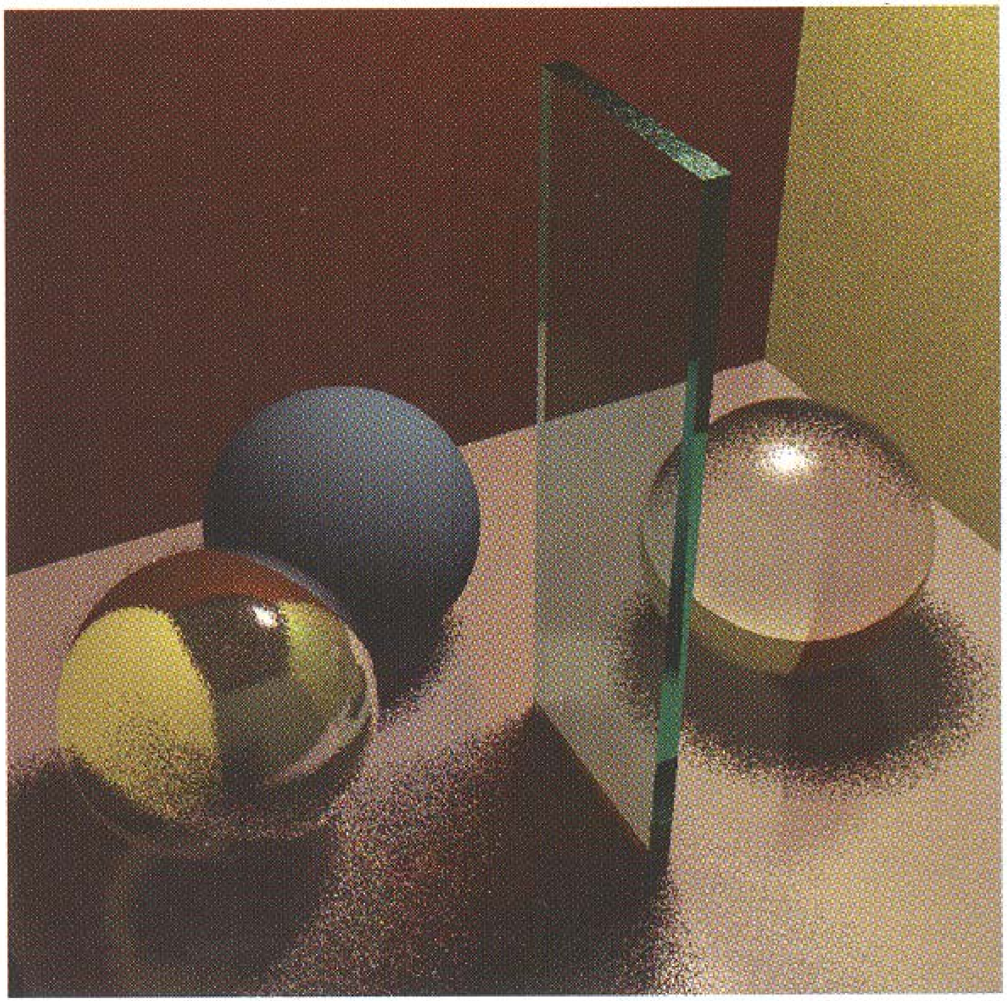
Wikipedia

Dispersion

- Image by Henrik Wann Jensen using Photon Mapping



Questions?



Application: CAD for lenses

- Has revolutionized lens design
  - E.g. zoom lenses are good now

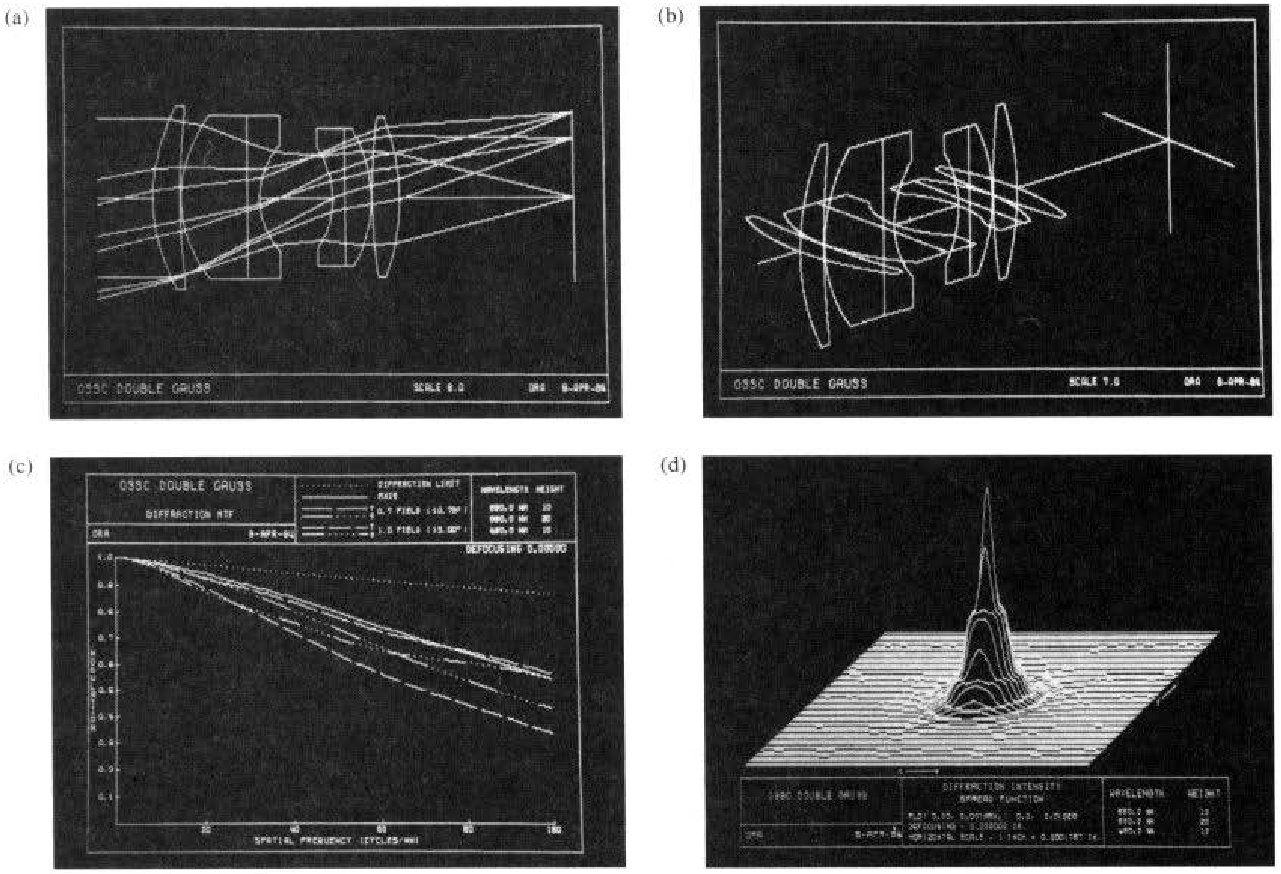


Figure 11.50 An example of the kind of lens design information available via computer techniques. (Photos courtesy Optical Research Associates.)

From Hecht's Optics

Lens design by Ray Tracing

- Used to be done manually, by rooms full of engineers who would trace rays.
- Now software, e.g. Zemax
- More in 6.815/6.865 Computational Photography

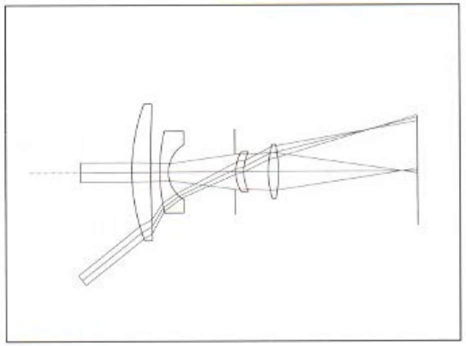


Figure-5

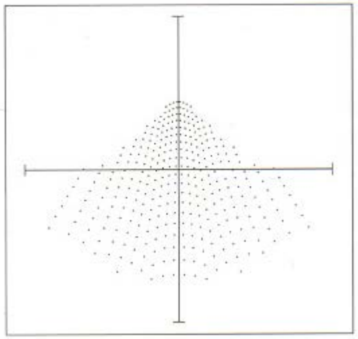


Figure-8

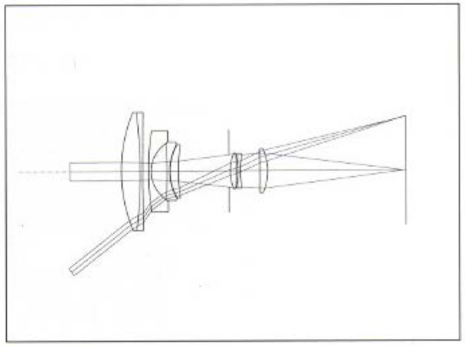


Figure-6

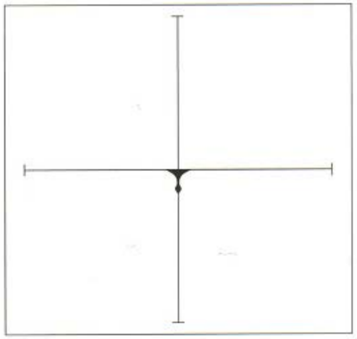


Figure-9

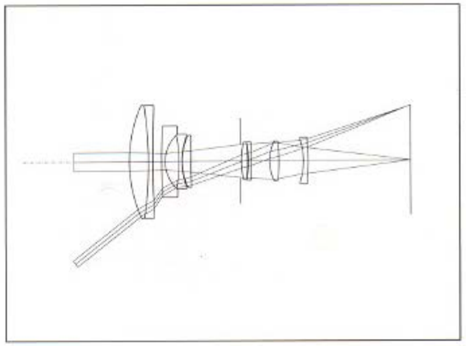


Figure-7

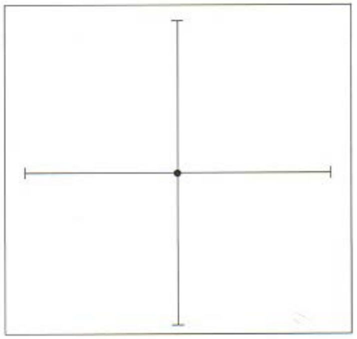
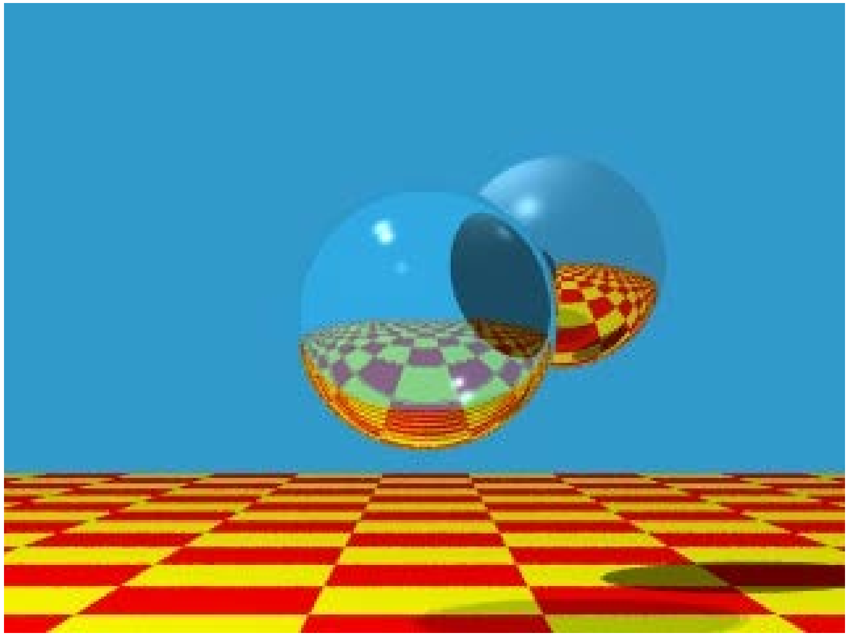


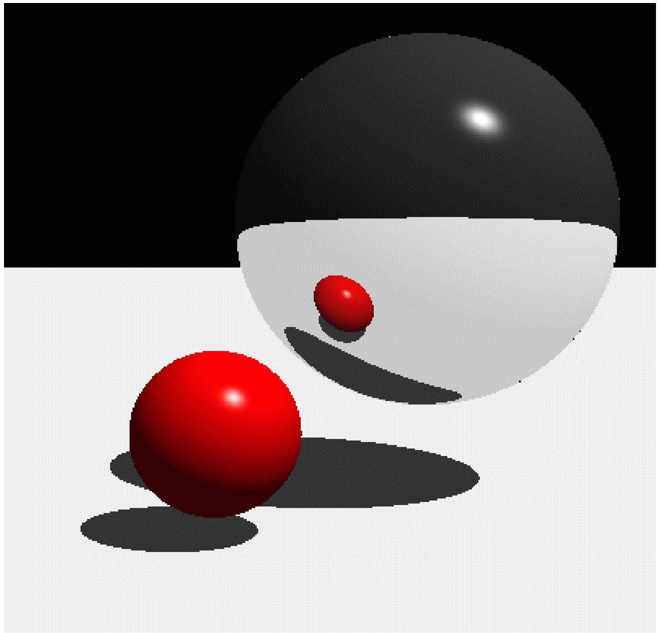
Figure-10

source: canon red book

Let’s Pause for a Moment...

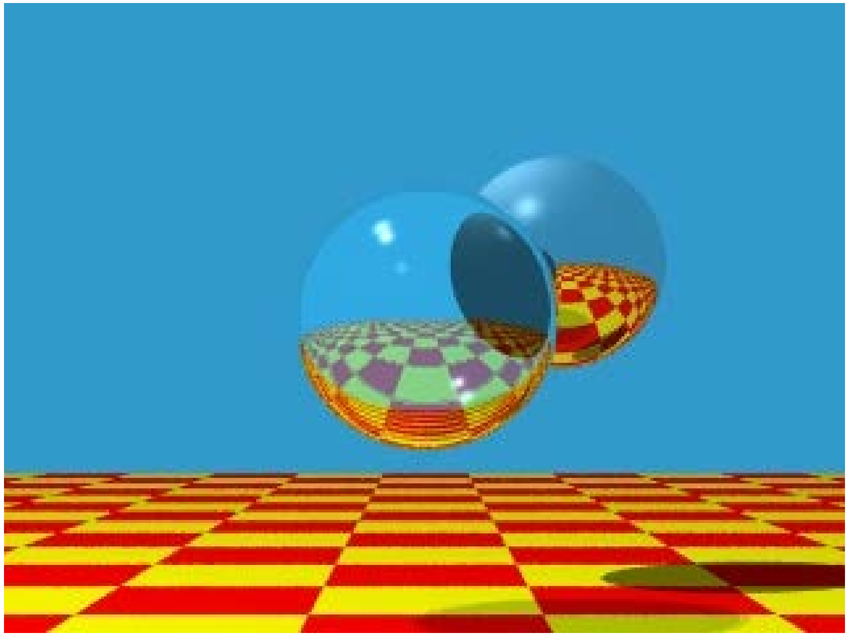
- Do these pictures look real?

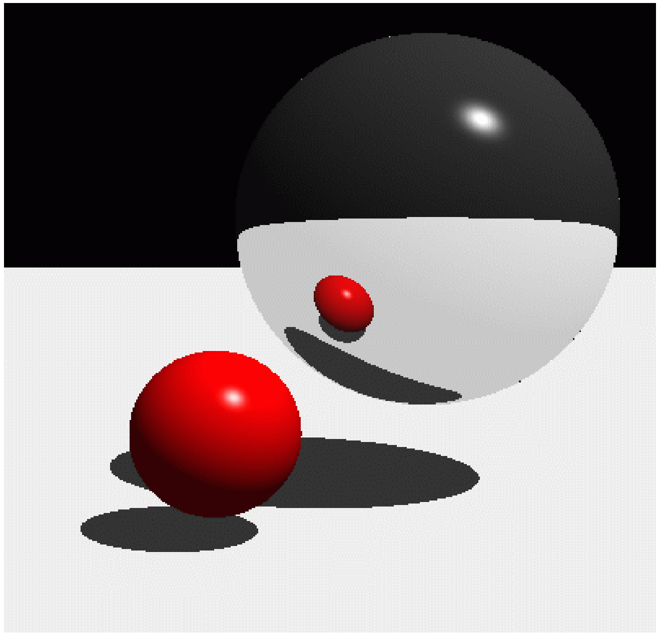




What’s Wrong then?

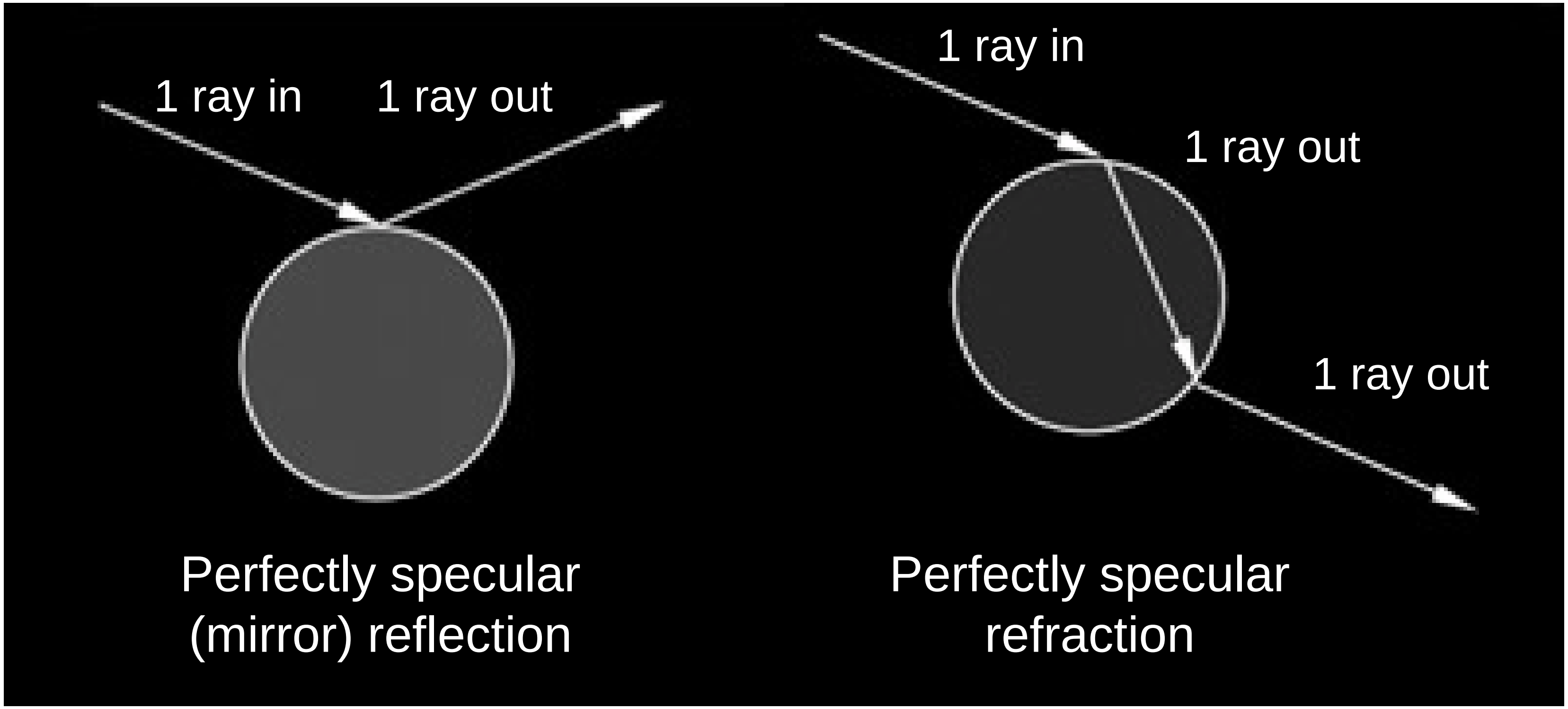
- No surface is a perfect mirror,
  no material interface is perfectly smooth





What’s Wrong then?

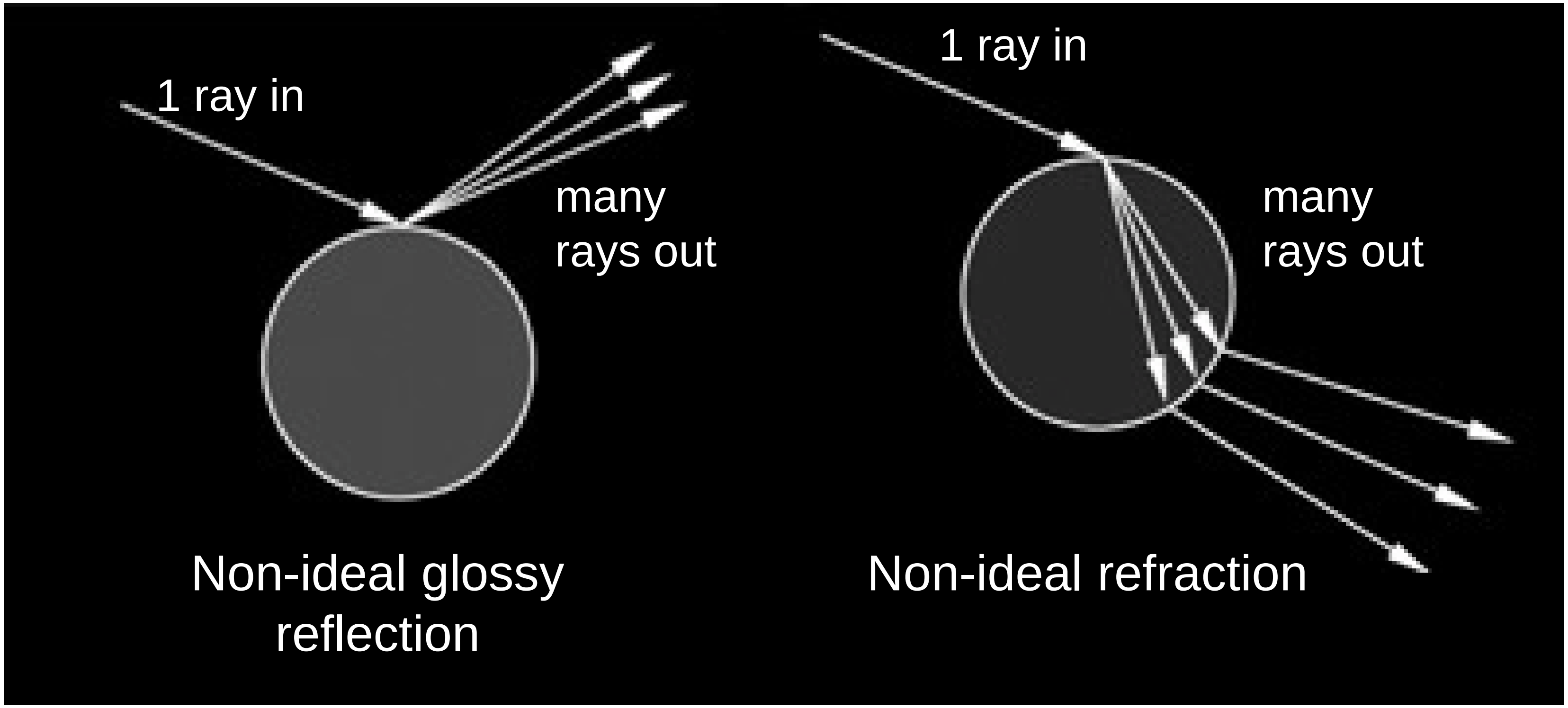
- No surface is a perfect mirror,
no material interface is perfectly smooth


1 ray in
1 ray out
Perfectly specular (mirror) reflection
1 ray in
1 ray out
1 ray out
Perfectly specular refraction

Adapted from blender.org

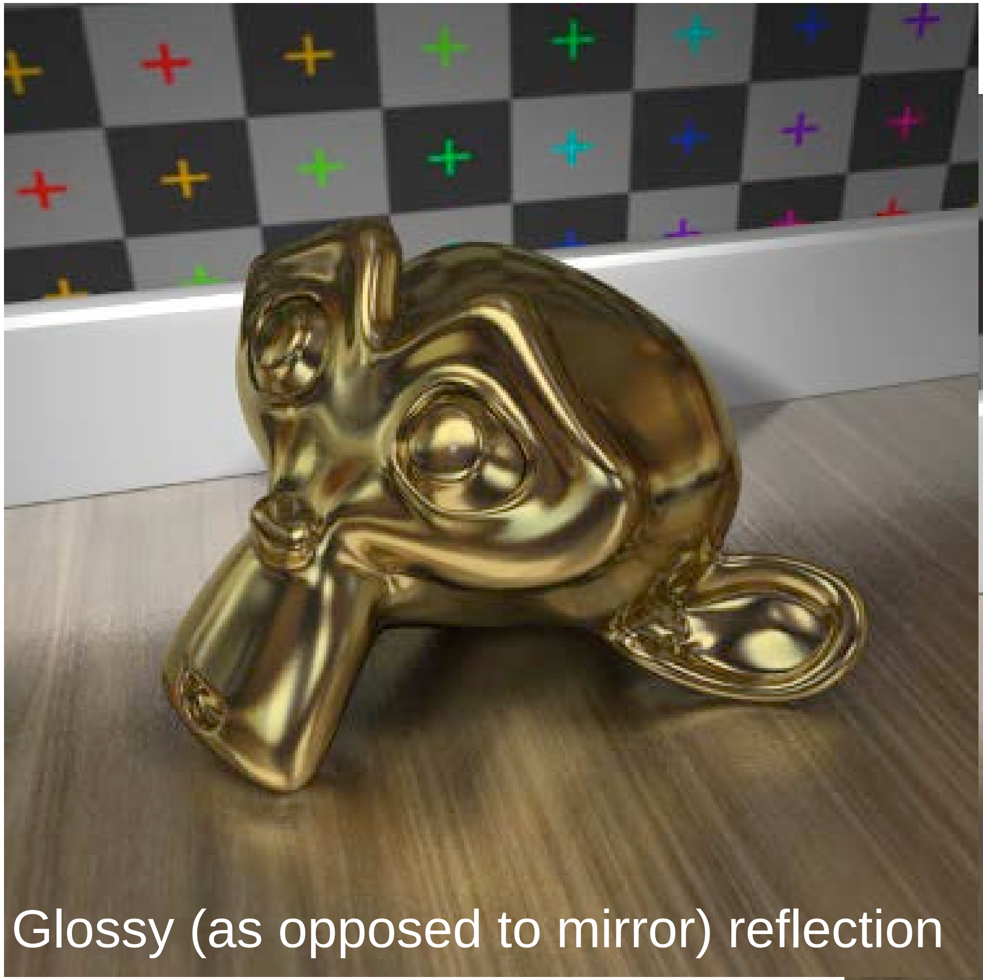
Non-Ideal Reflection/Refraction

- No surface is a perfect mirror,
  no material interface is perfectly smooth

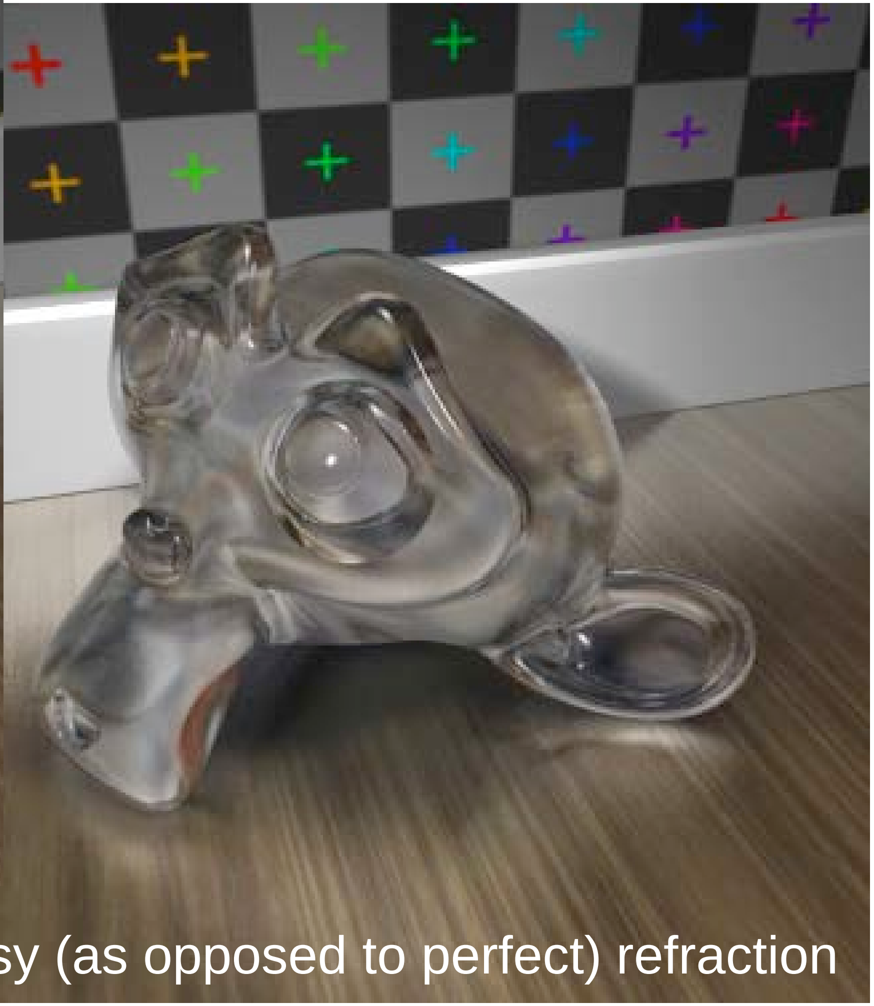


Adapted from blender.org

Non-Ideal Reflection/Refraction



Glossy (as opposed to mirror) reflection

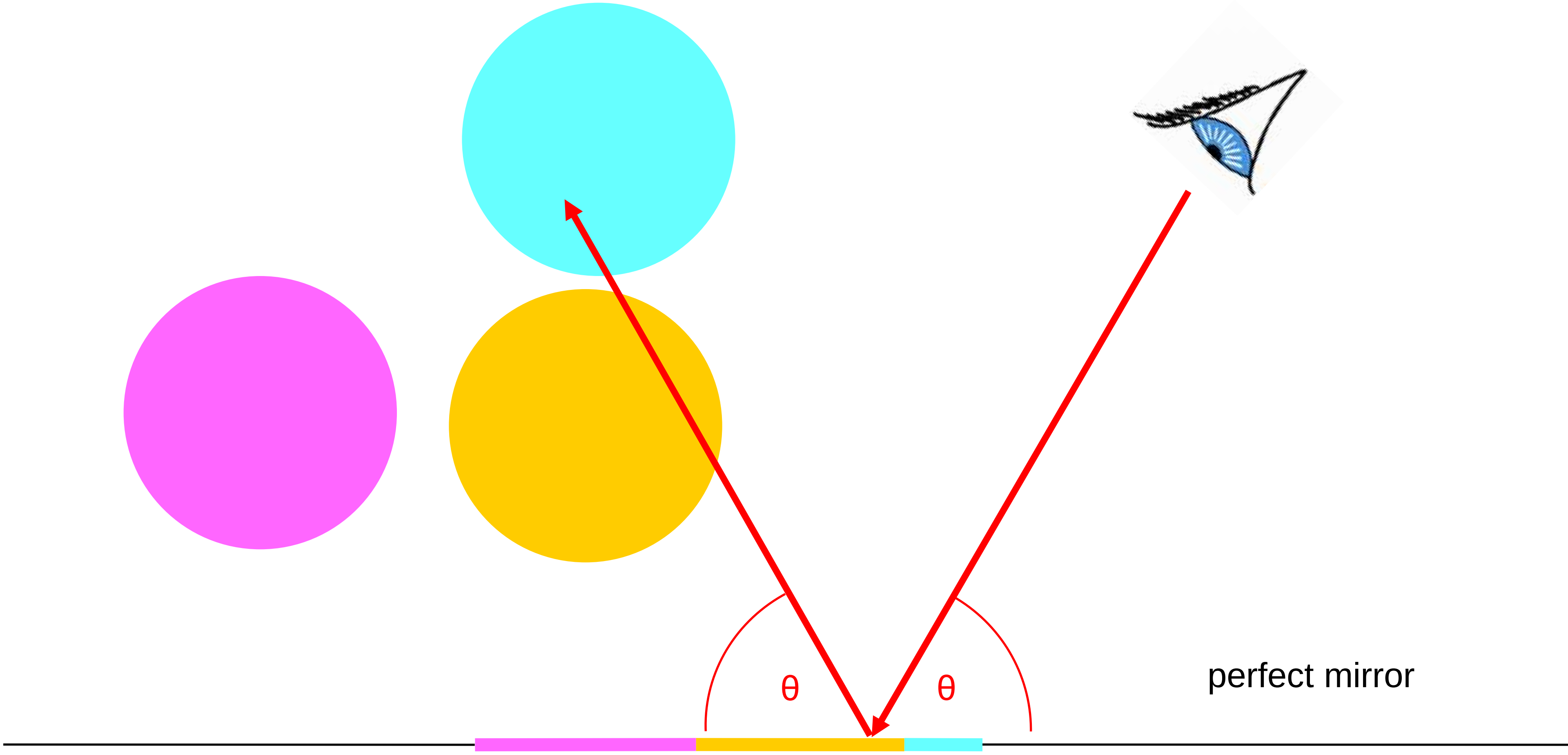


Glossy (as opposed to perfect) refraction

images from blender.org

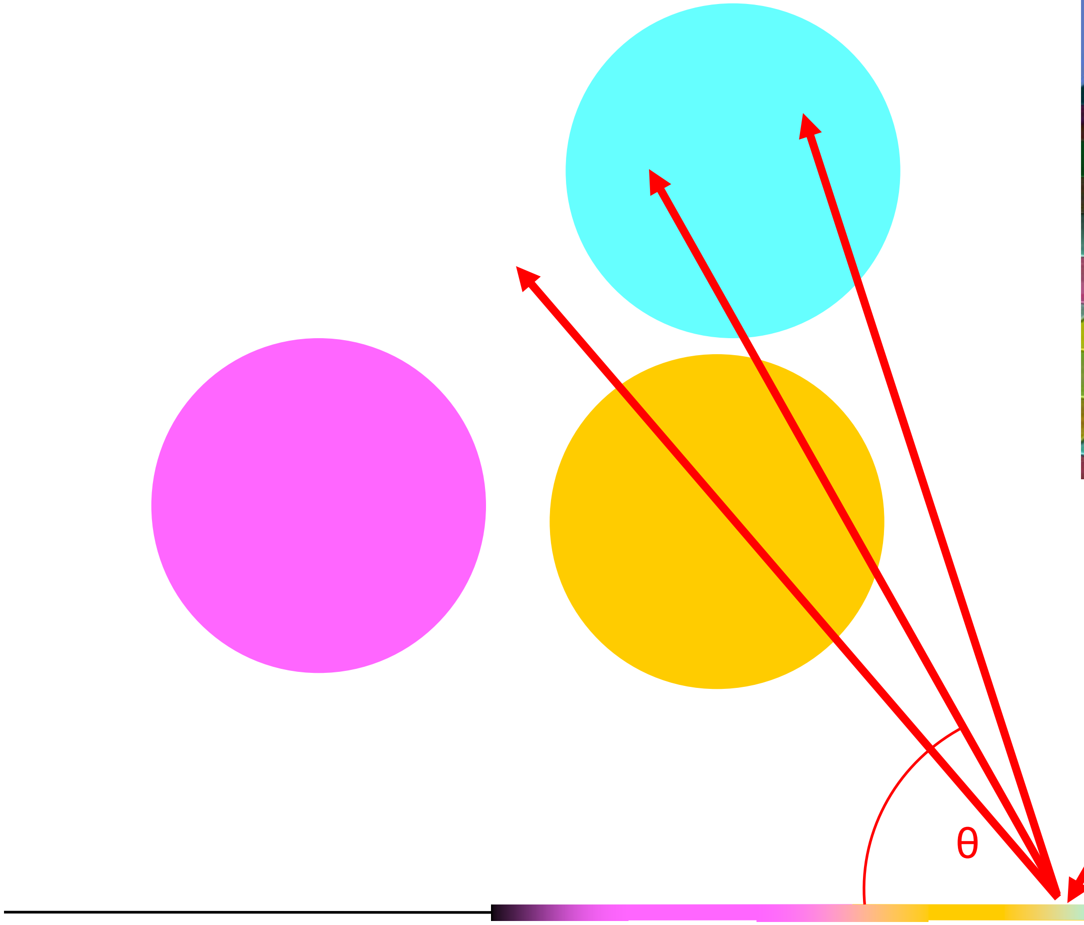
Reflection

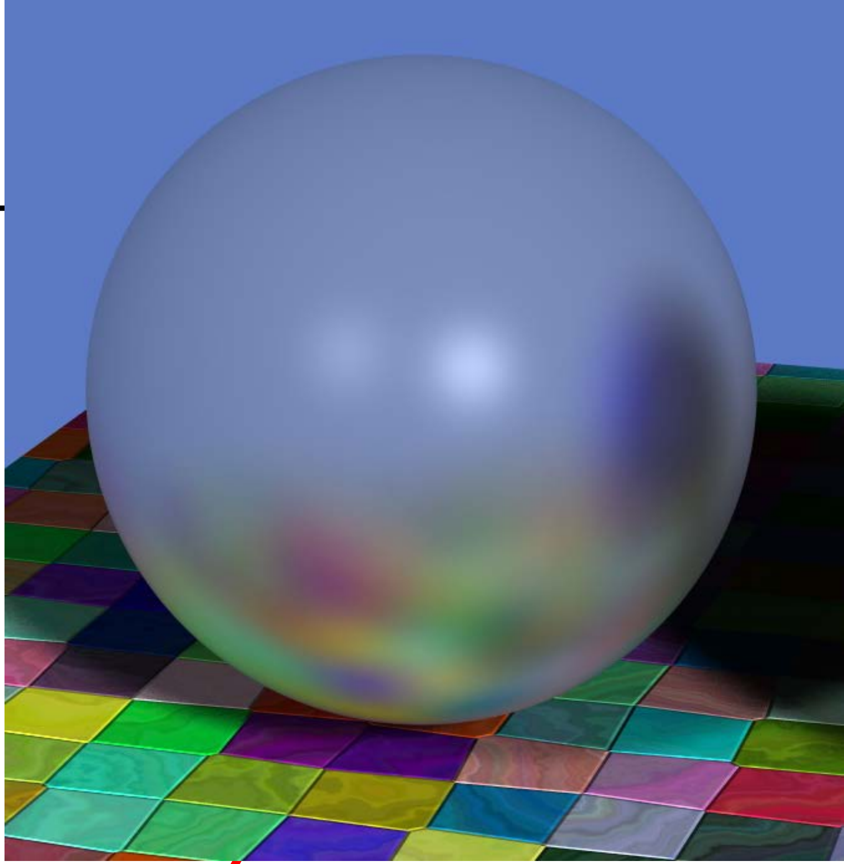
- One reflection ray per intersection



Glossy Reflection

- Multiple reflection rays



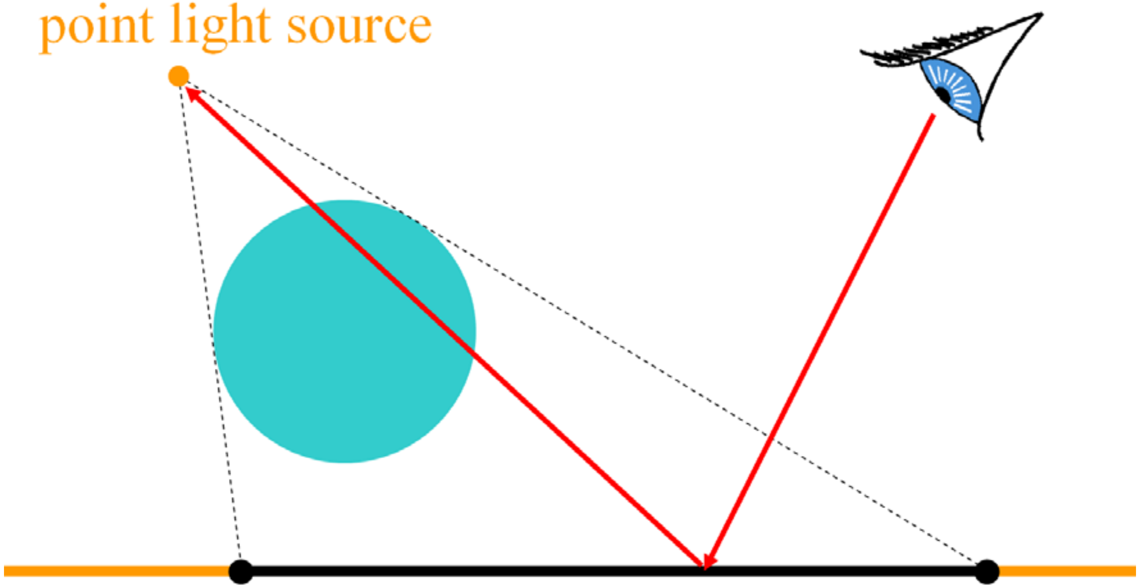


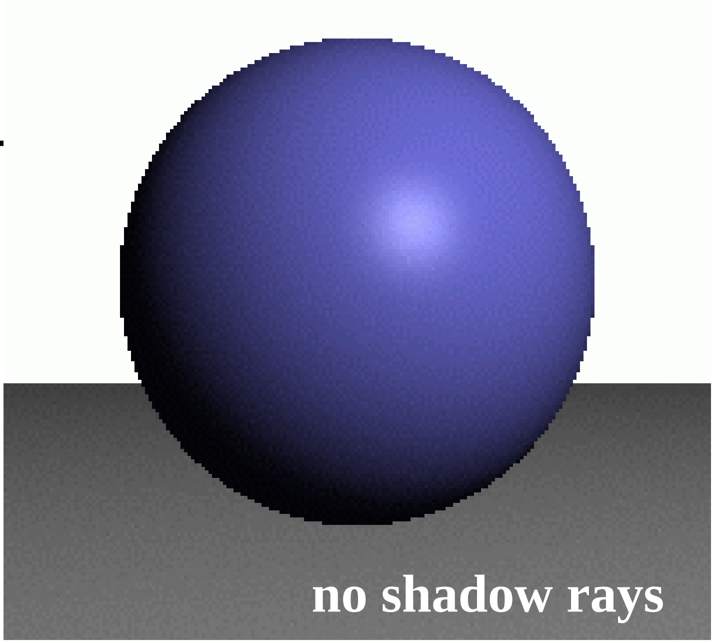
Justin Legakis

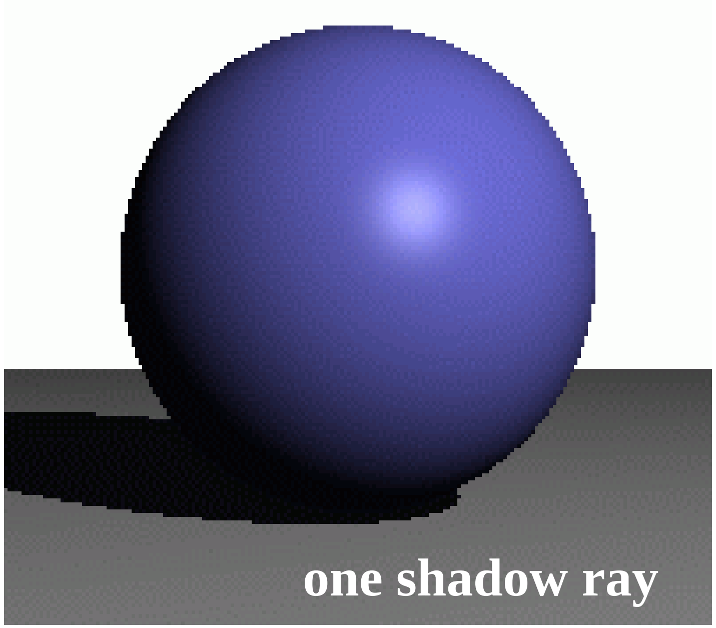
polished surface

Shadows

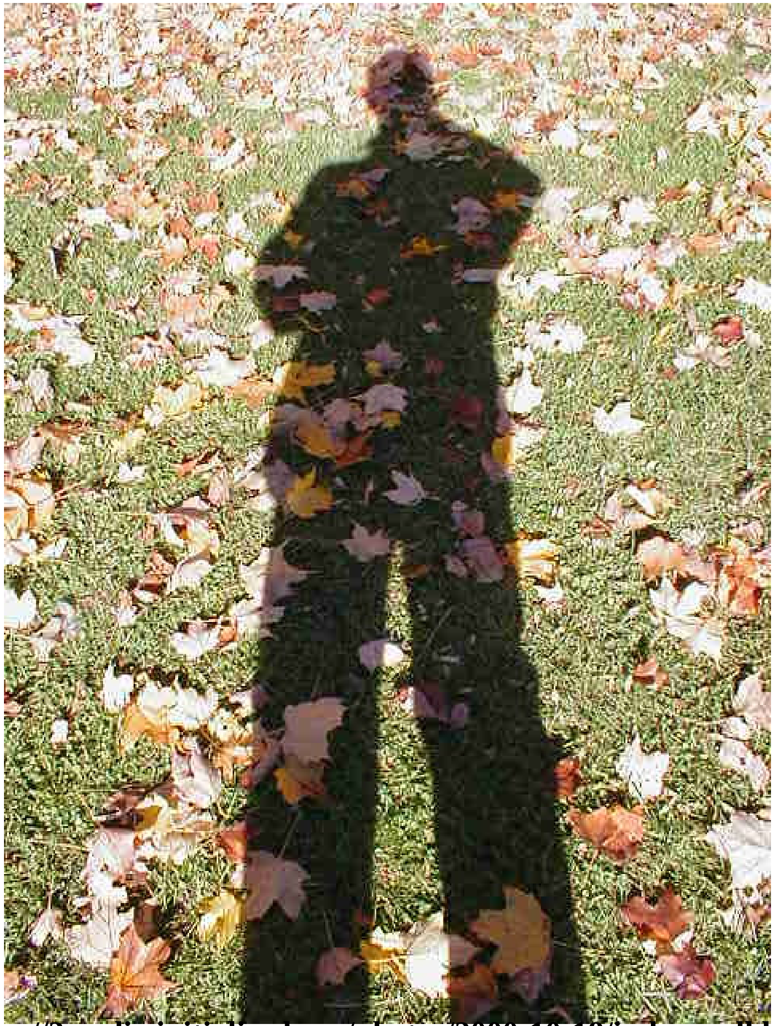
- One shadow ray per intersection per point light source



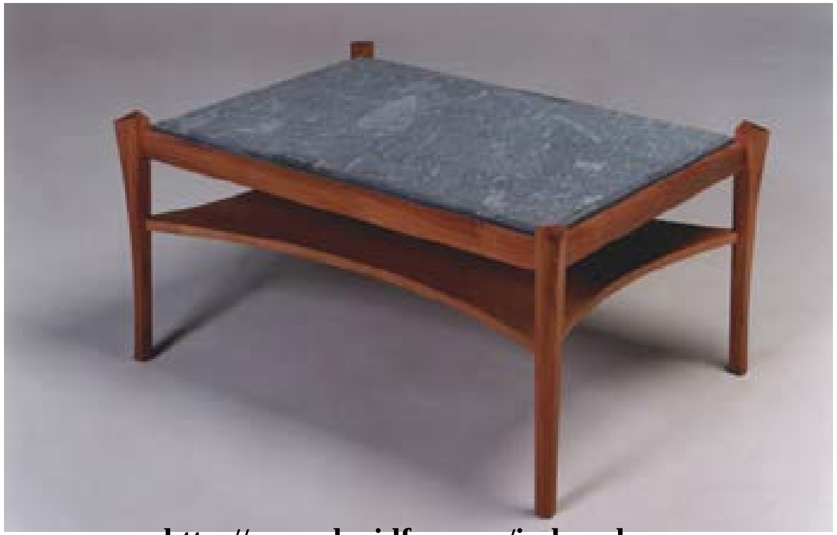




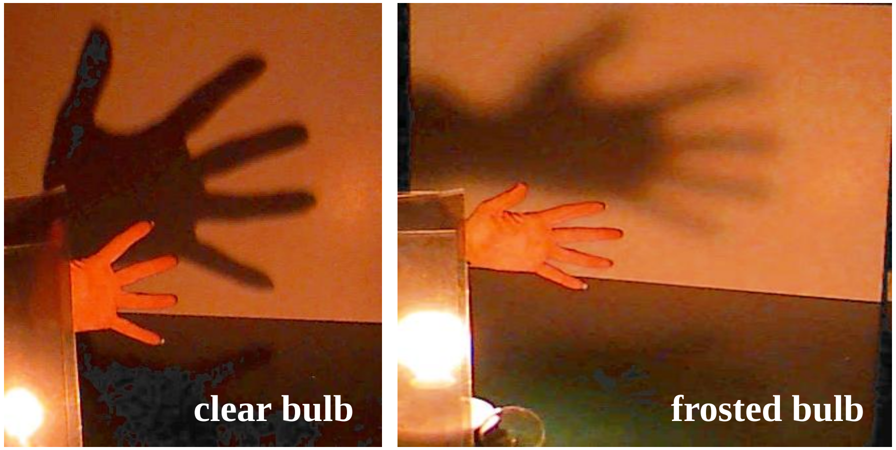
Shadows & Light Sources



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



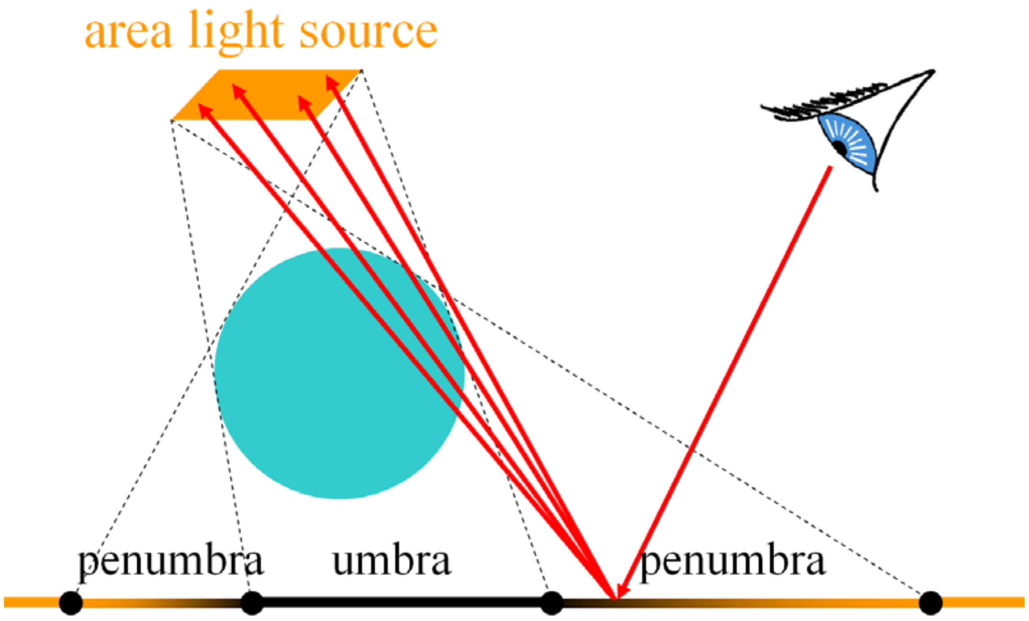
http://www.davidfay.com/index.php

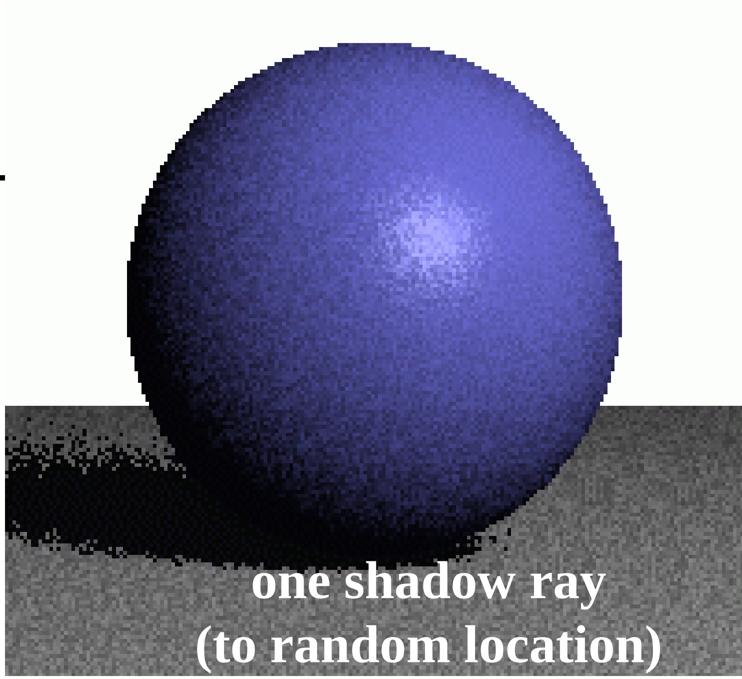


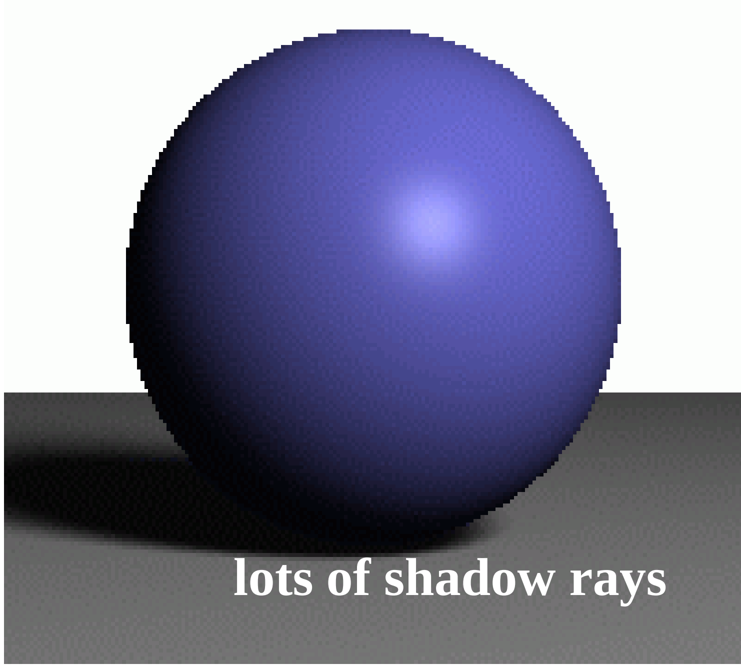
http://www.pa.uky.edu/~sciworks/light/preview/bulb2.htm

Soft Shadows

- Multiple shadow rays to sample area light source

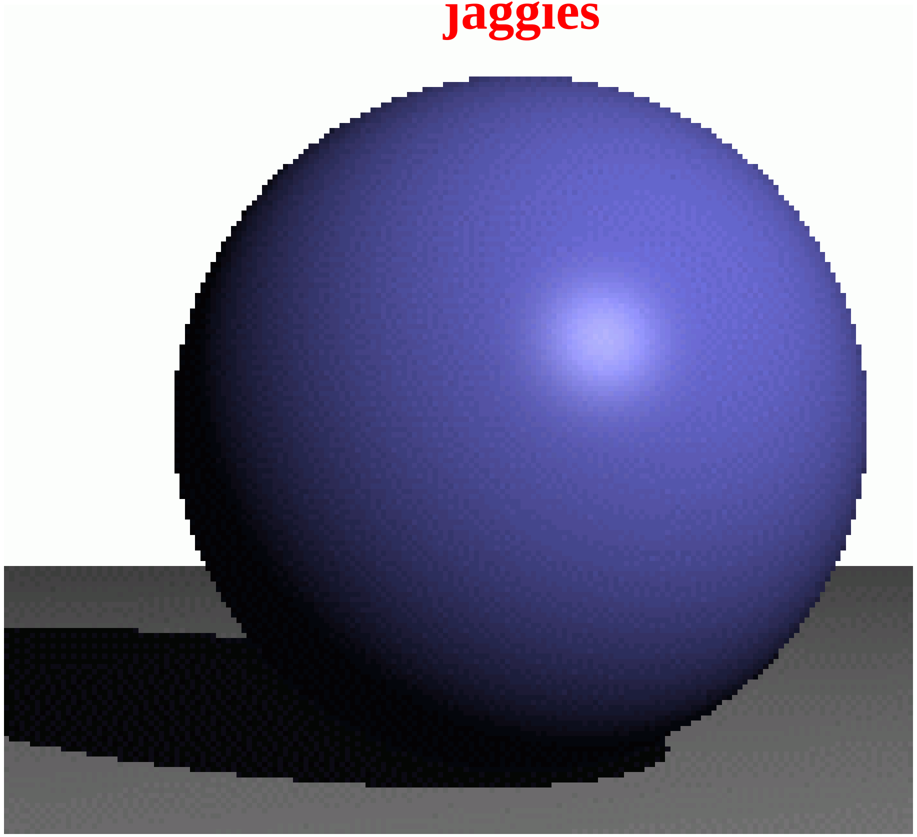



one shadow ray
(to random location)

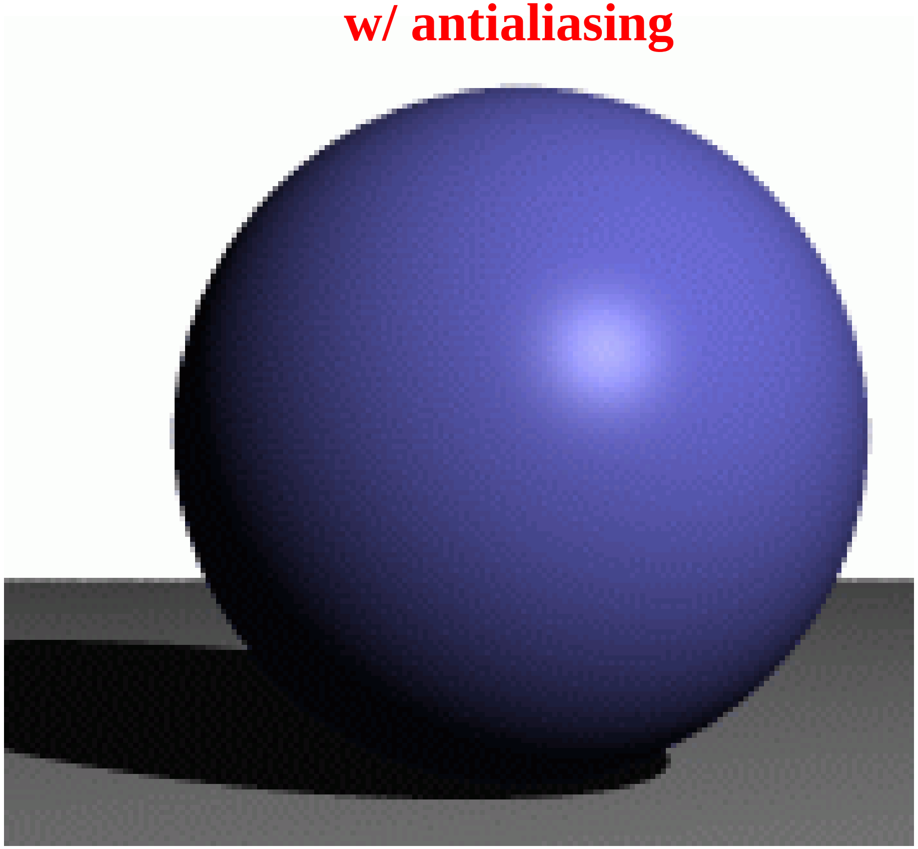

lots of shadow rays

Antialiasing – Supersampling

- Multiple rays per pixel



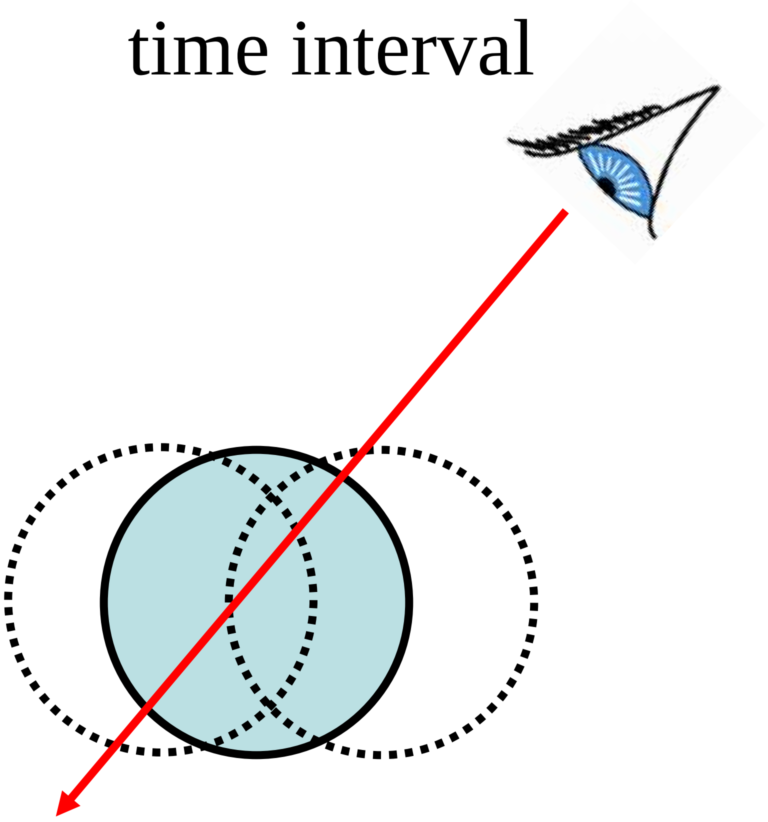
jaggies

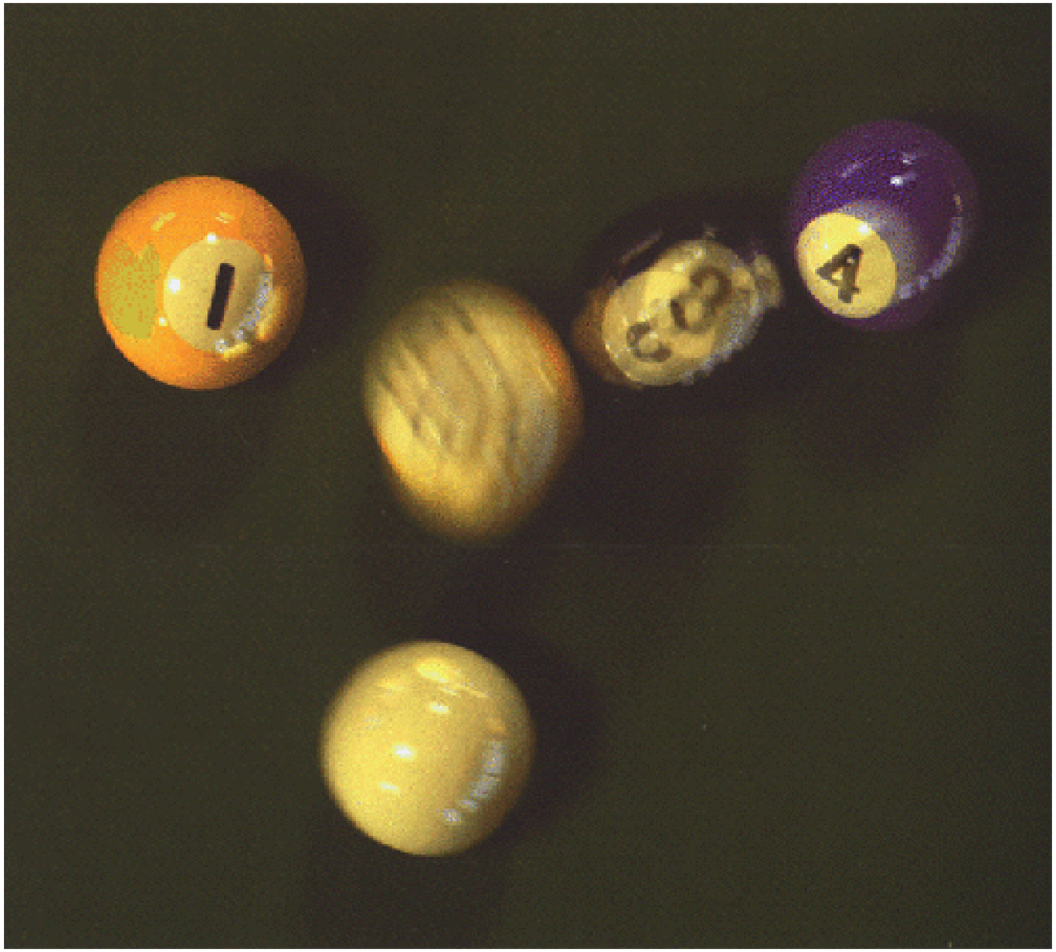


w/ antialiasing

Motion Blur

- Sample objects temporally over time interval

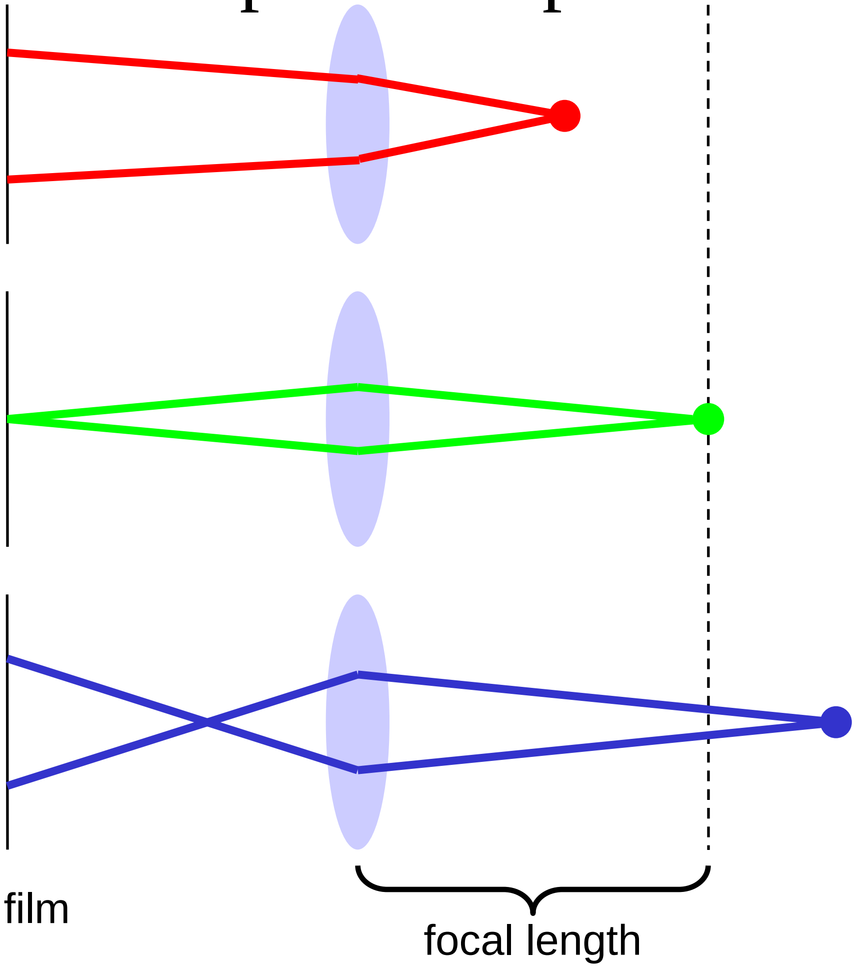


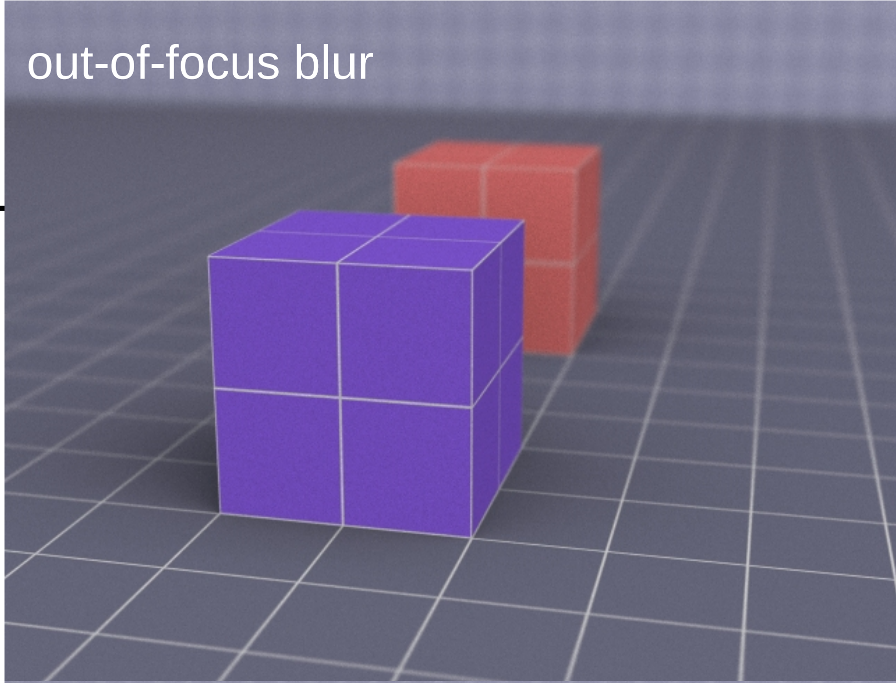


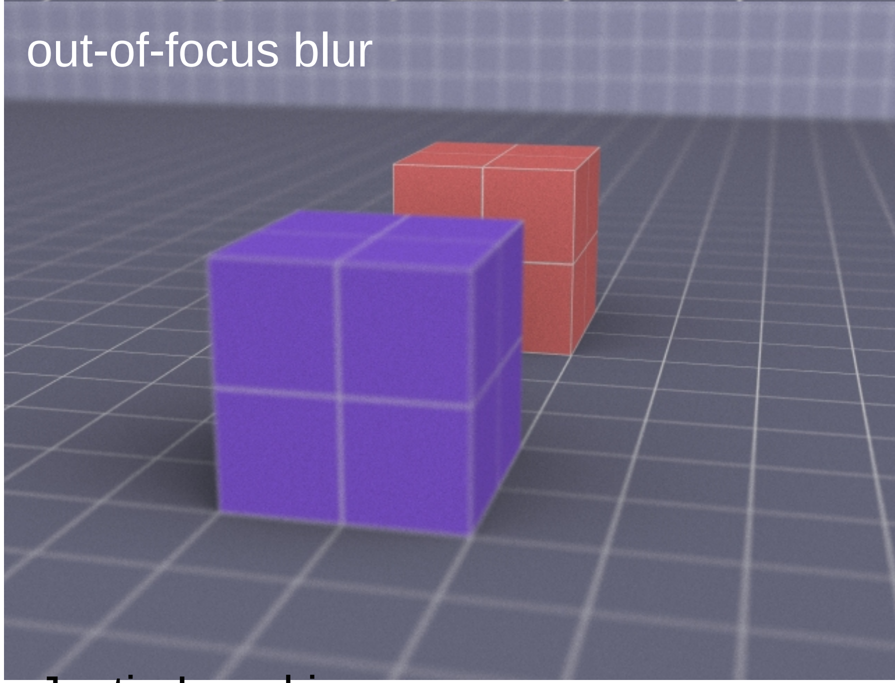
Rob Cook

Depth of Field

- Multiple rays per pixel: sample lens aperture







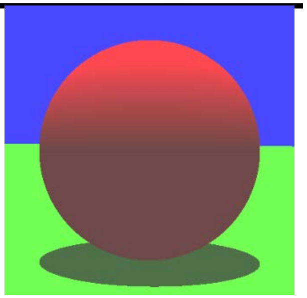
Justin Legakis

Questions?

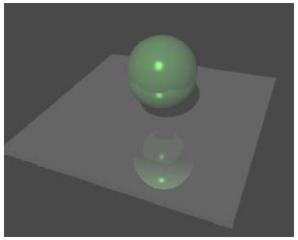
Henrik Wann Jensen

Overview of Today

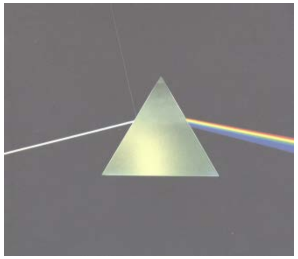
- Shadows



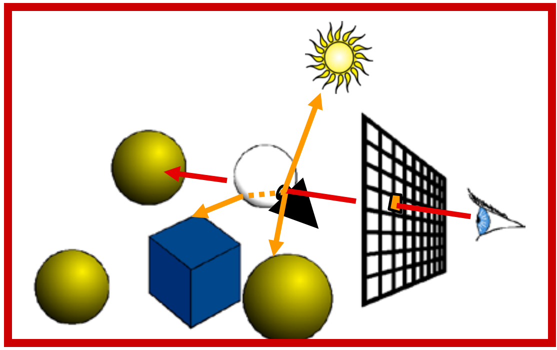
- Reflection



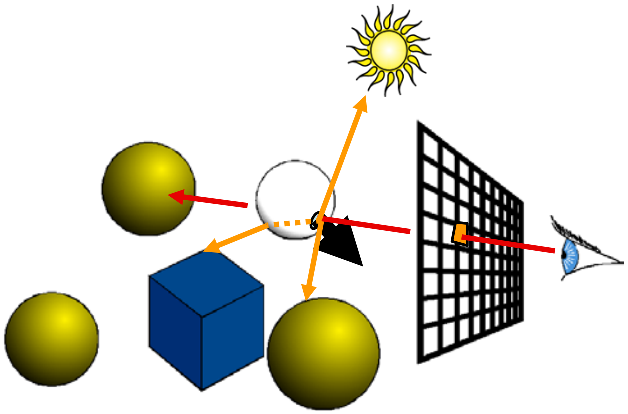
- Refraction



- Recursive Ray Tracing



Recap: Ray Tracing



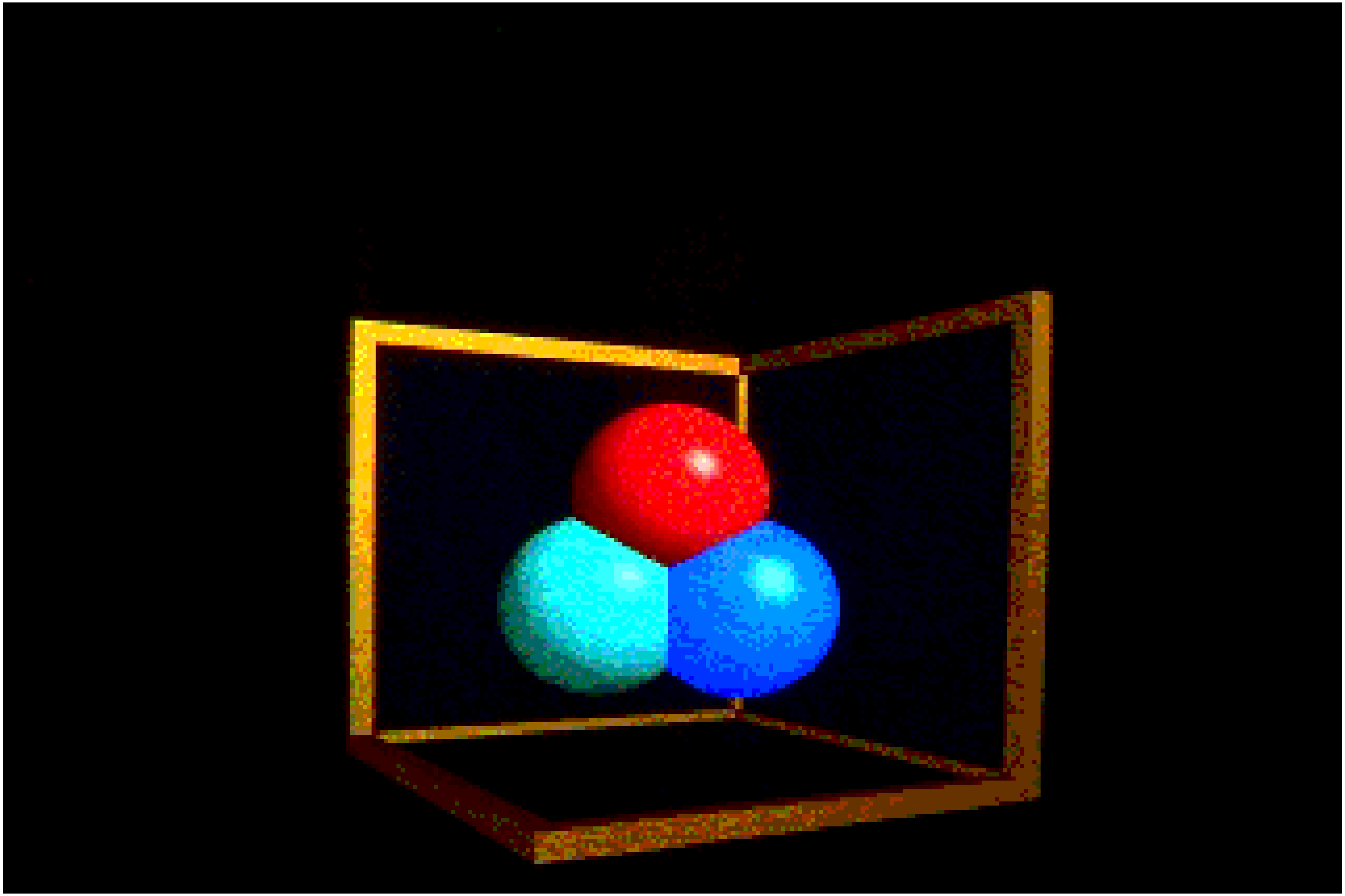
```
trace ray
   Intersect all objects
   color = ambient term
   For every light
      cast shadow ray
      color += local shading term
   If mirror
      color += $\text{color}_{refl}$ *
                trace reflected ray
   If transparent
      color += $\text{color}_{trans}$ *
               trace transmitted ray
```

- ***Does it ever end?***

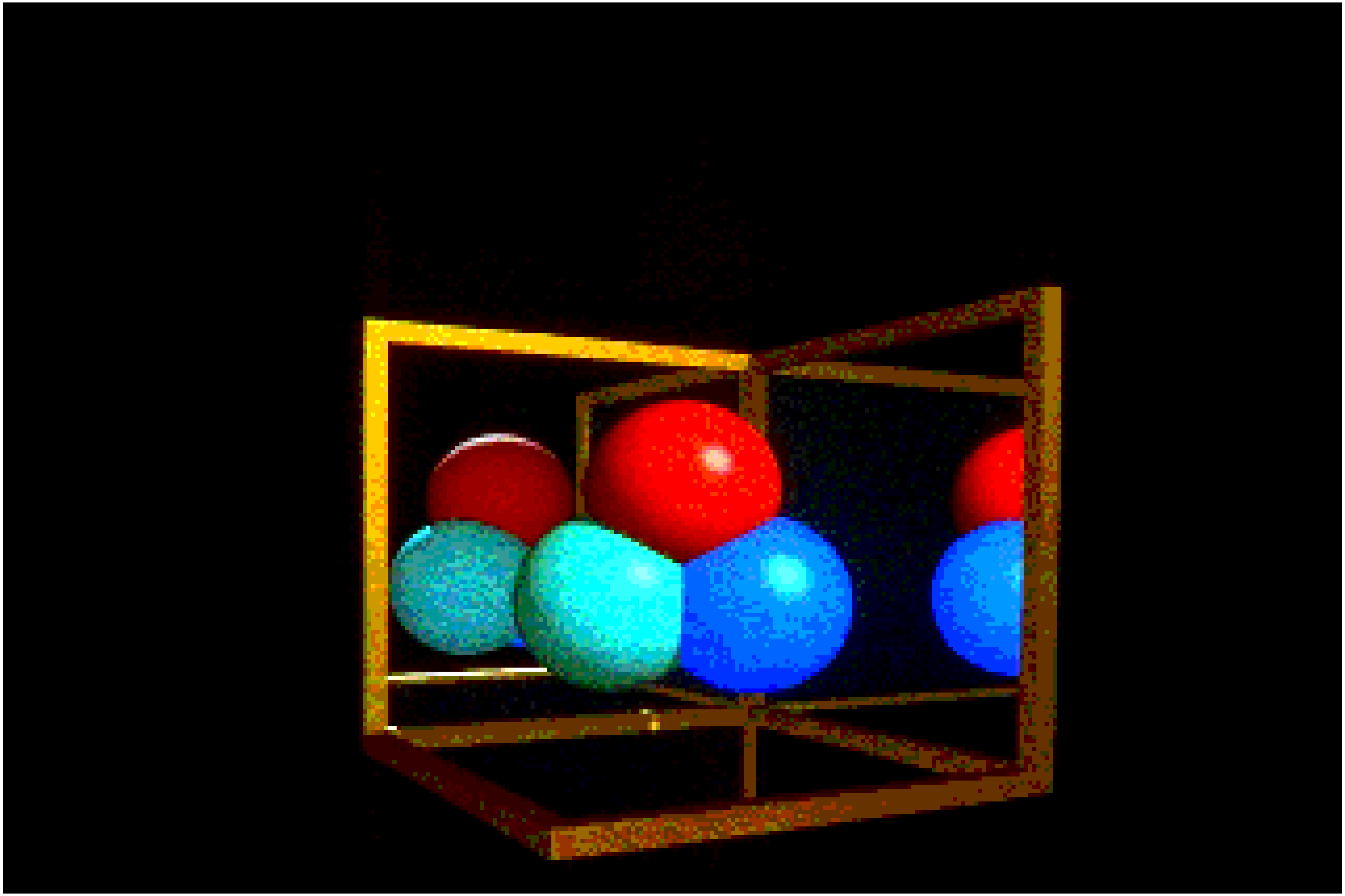
Stopping criteria:

- Recursion depth
  - Stop after a number of bounces
- Ray contribution
  - Stop if reflected / transmitted contribution becomes too small

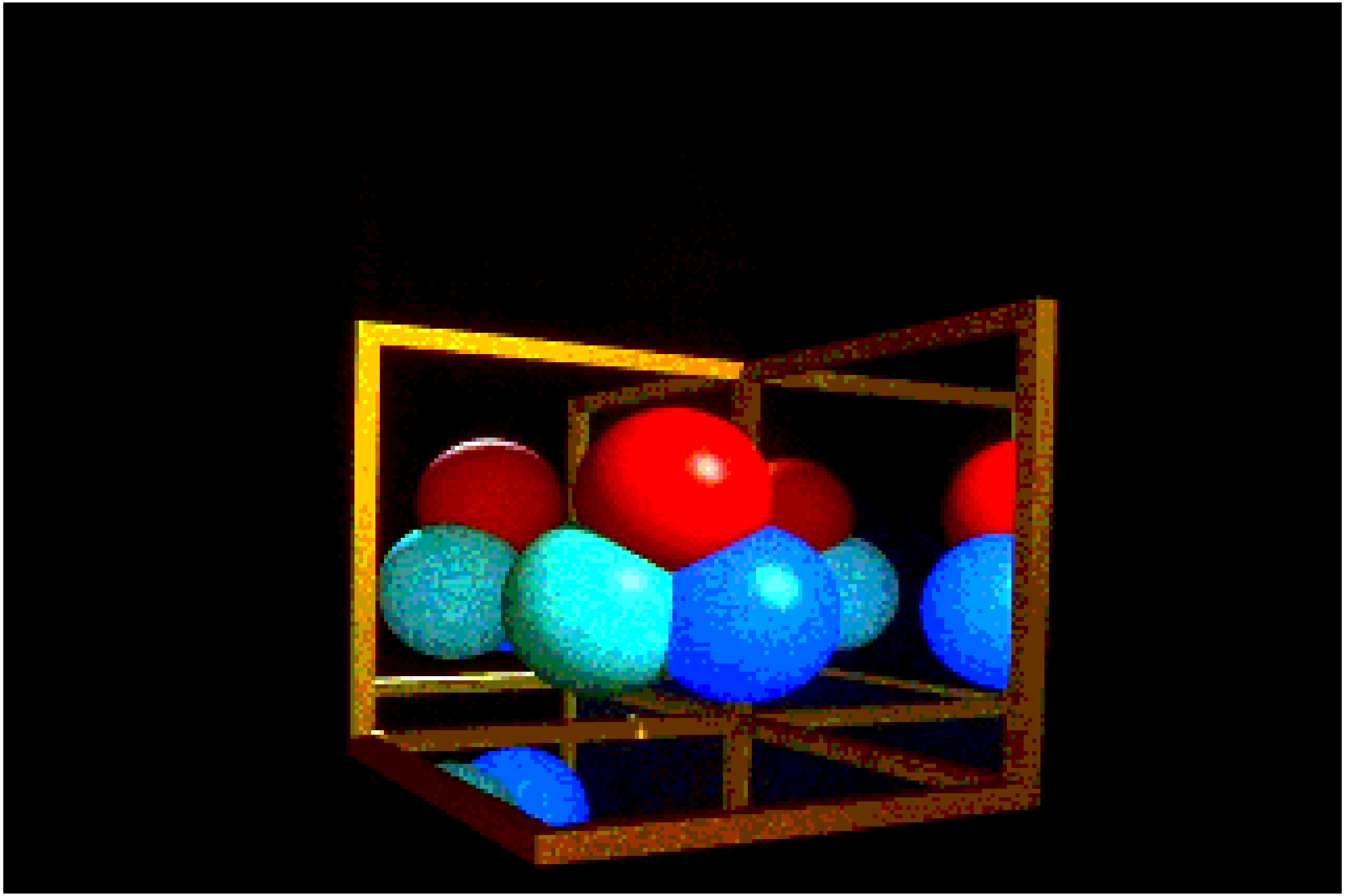
Recursion For Reflection: None



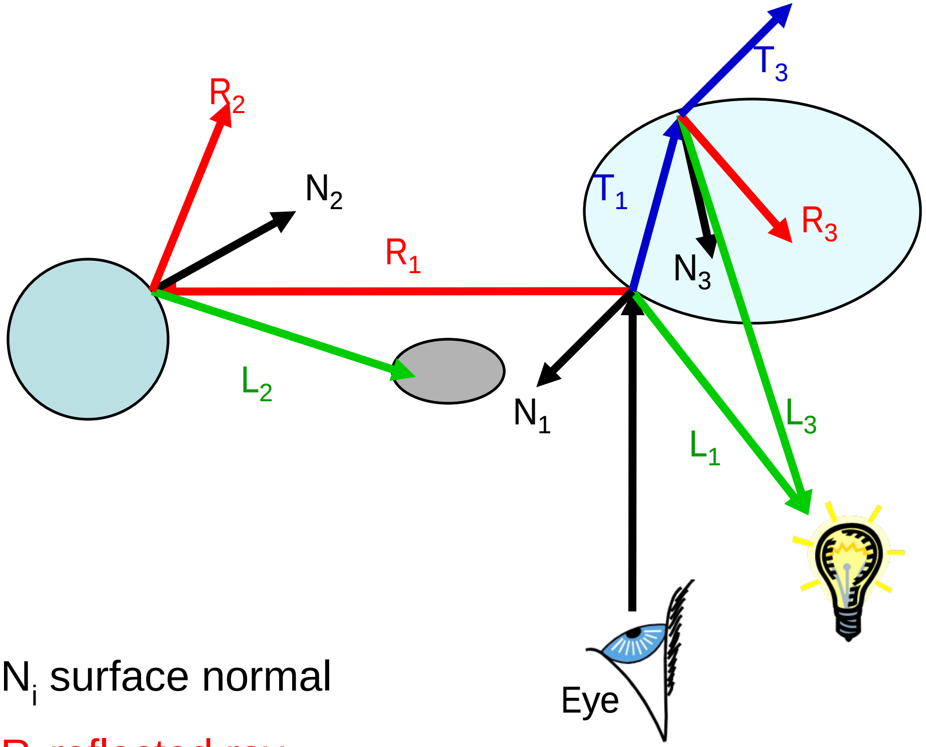
Recursion For Reflection: 1



Recursion For Reflection: 2



The Ray Tree

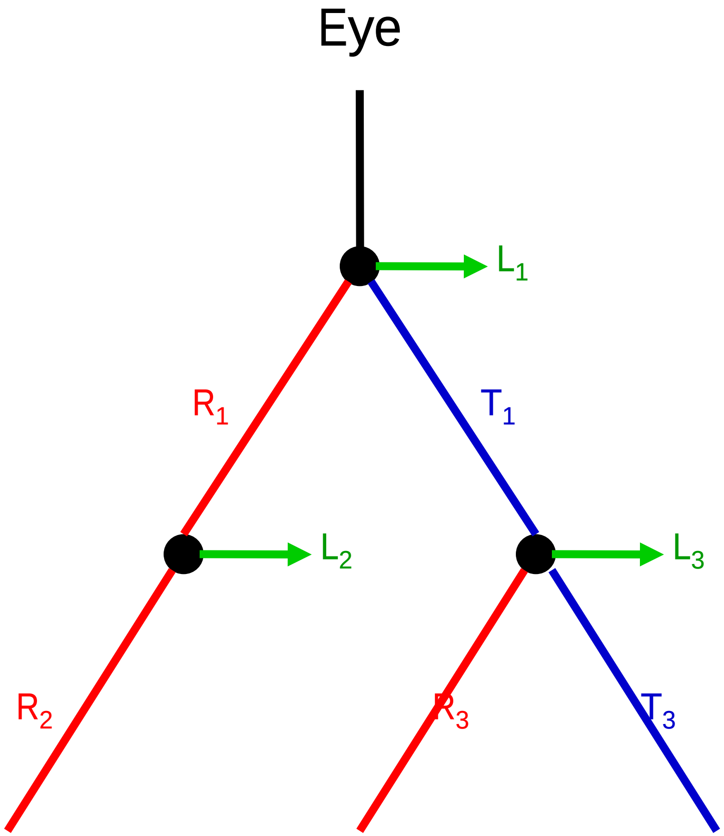


N_i surface normal

R_i reflected ray

L_i shadow ray

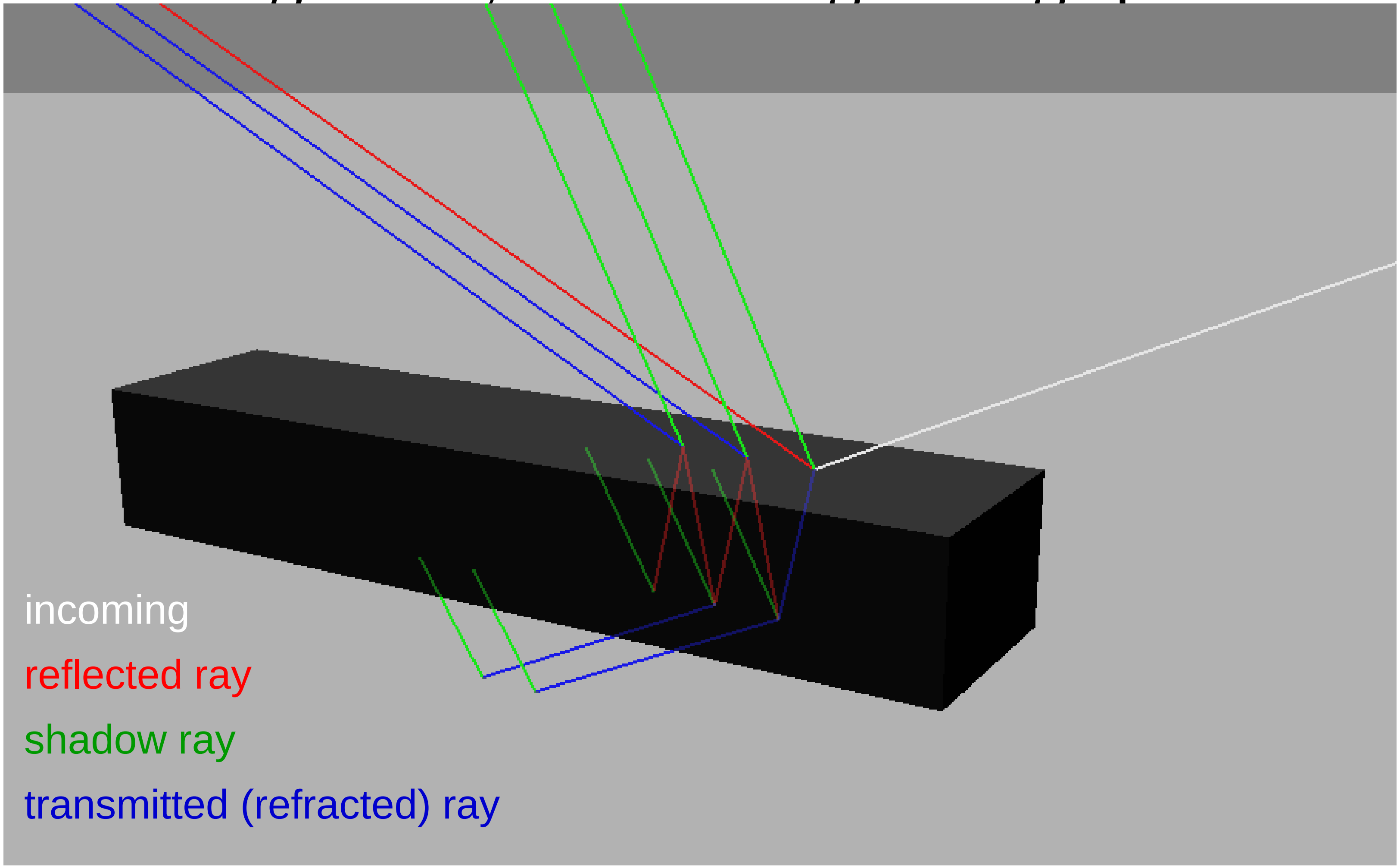
T_i transmitted (refracted) ray



Complexity?

Ray tree

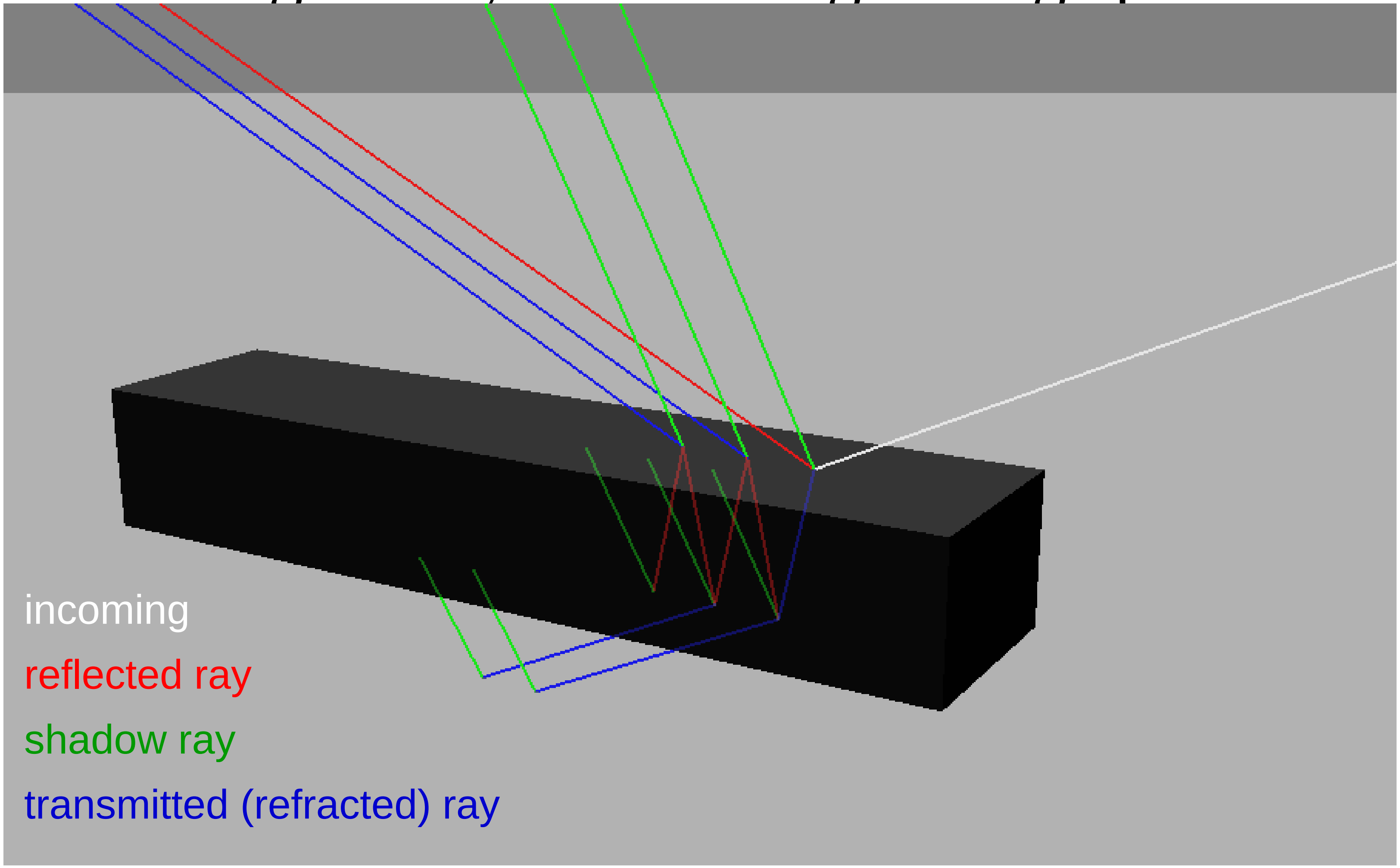
- Visualizing the ray tree for single image pixel

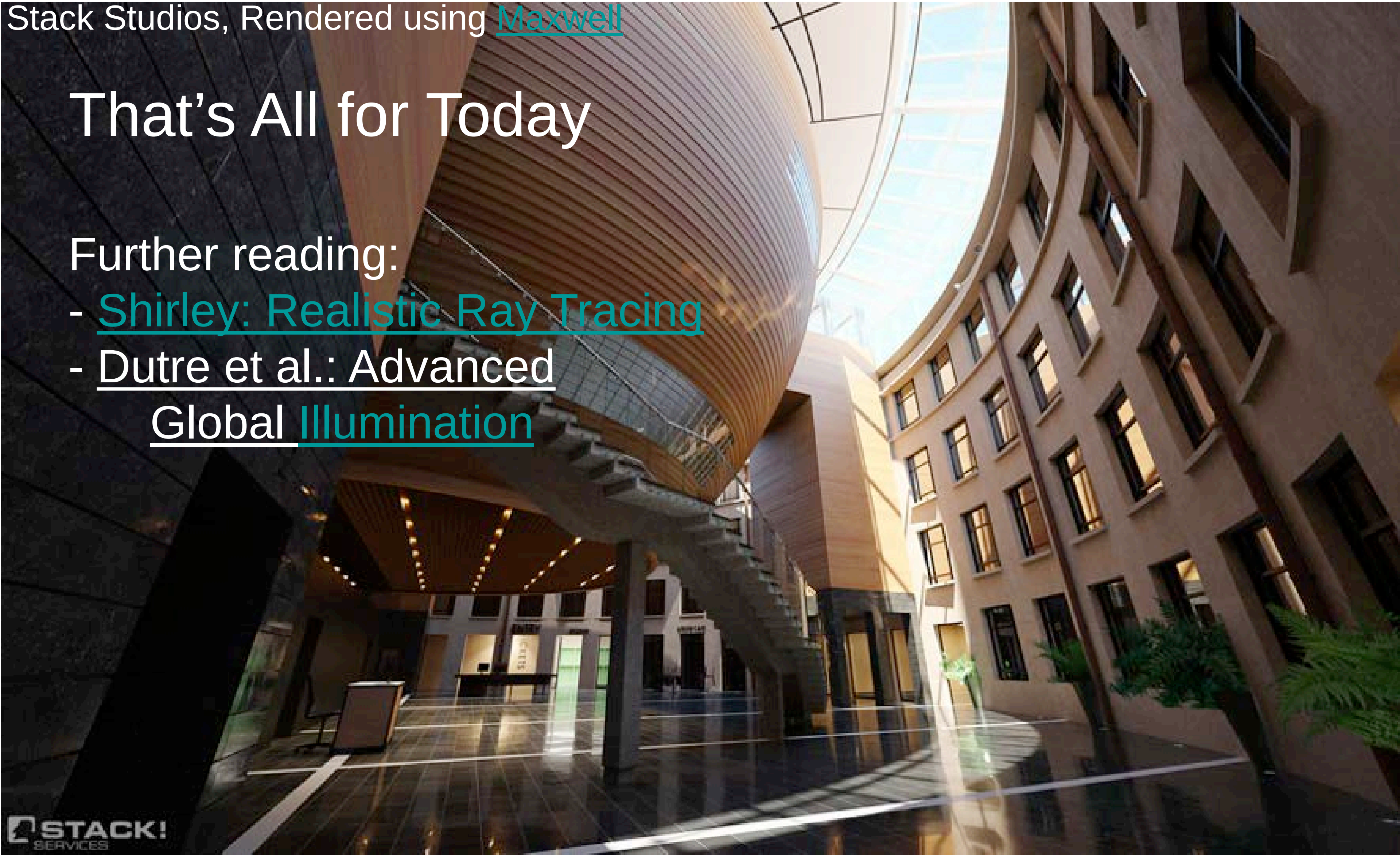


Ray tree

This gets pretty complicated pretty fast!

- Visualizing the ray tree for single image pixel





Stack Studios, Rendered using Maxwell

That’s All for Today

Further reading:

- Shirley: Realistic Ray Tracing
- Dutre et al.: Advanced Global Illumination