

Interactive Computer Graphics: Lecture 9

Ray tracing



© Steve Anger 1993



Direct and Global Illumination

- **Direct illumination**: A surface point receives light directly from all light sources in the scene.
 - Computed by the direct illumination model.
- **Global illumination**: A surface point receives light after the light rays interact with other objects in the scene.
 - Points may be in shadow.
 - Rays may refract through transparent material.
 - Computed by reflection and transmission rays.

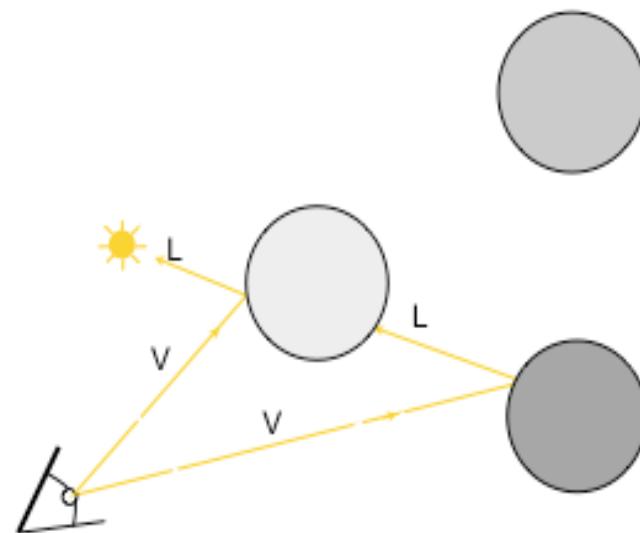
Albrecht Dürer's Ray Casting Machine

- Albrecht Dürer, 16th century



Arthur Appel, 1968

- On calculating the illusion of reality, 1968
- Cast one ray per pixel (ray casting).
 - For each intersection, trace one ray to the light to check for shadows
 - Only a local illumination model
- Developed for pen-plotters



Ray casting

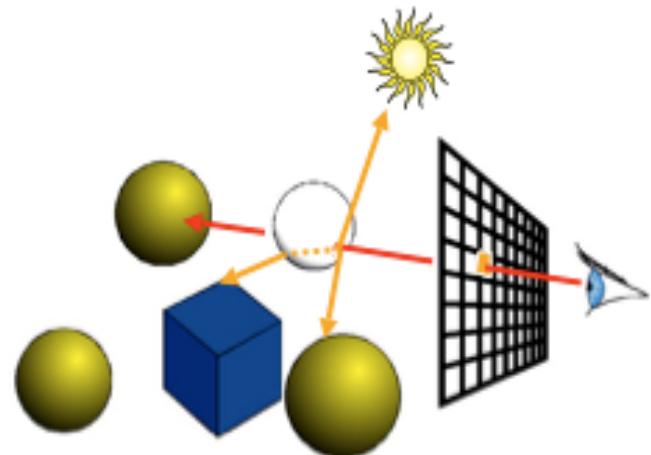
cast ray

Intersect all objects

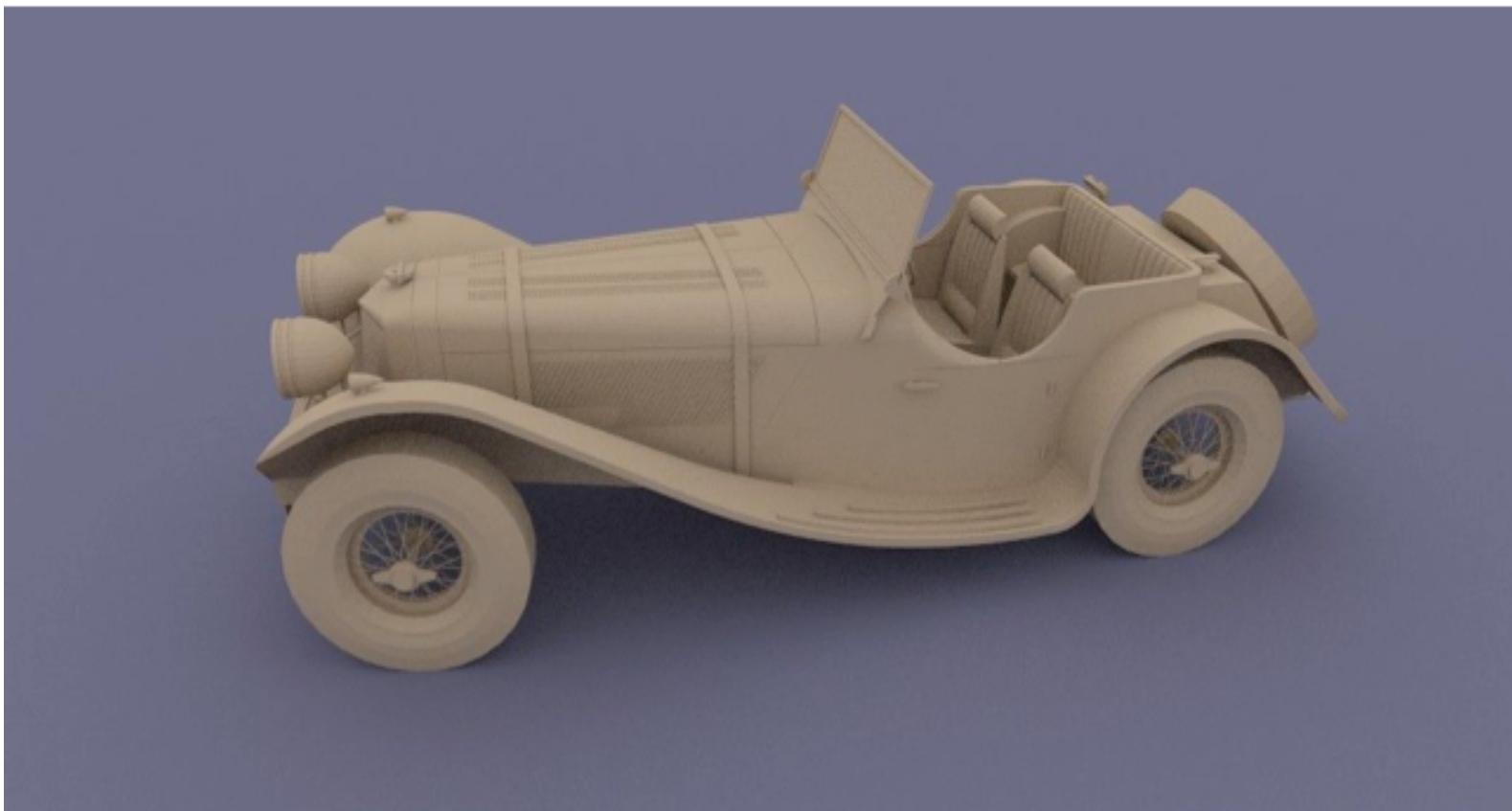
color = ambient term

For every light cast shadow ray

col += local shading term

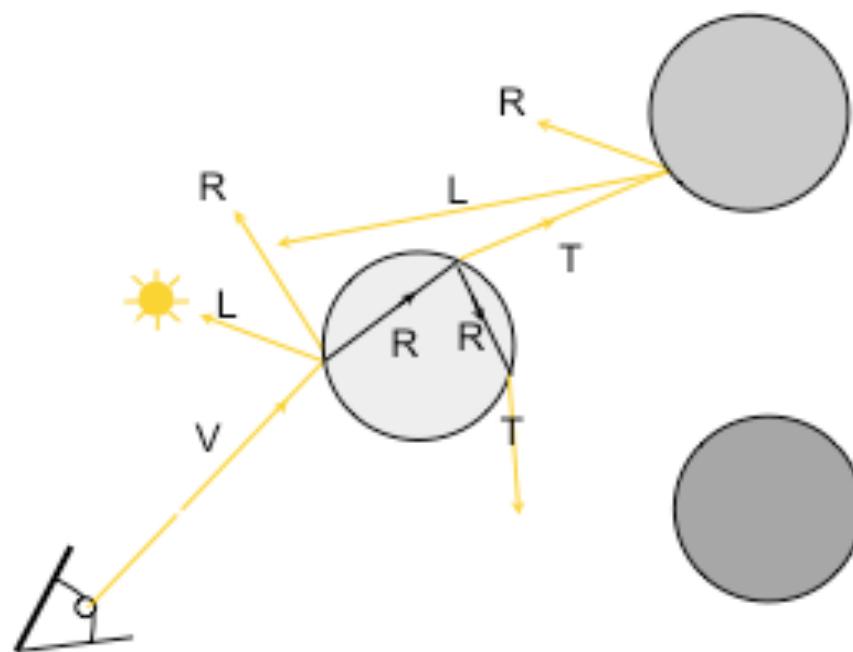


Ray casting

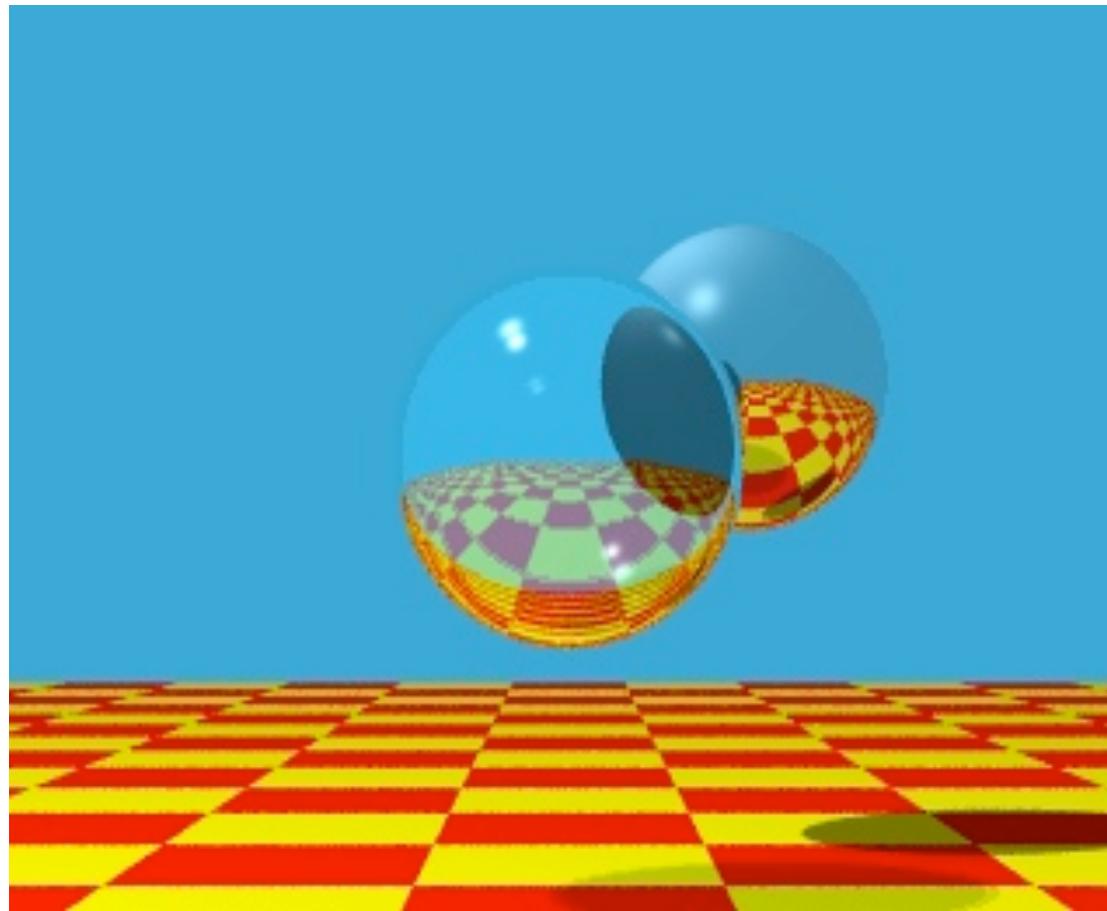


Turner Whitted, 1980

- An Improved Illumination Model for Shaded Display, 1980
- First global illumination model:
 - An object's color is influenced by lights and other objects in the scene
 - Simulates specular reflection and refractive transmission



Turner Whitted, 1980



Recursive ray casting

trace ray

 Intersect all objects

 color = ambient term

 For every light

 cast shadow ray

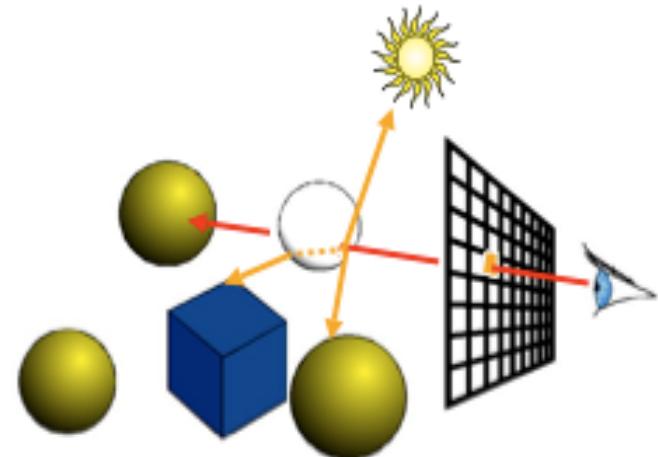
 col += local shading term

 If mirror

 col += k_refl * trace reflected ray

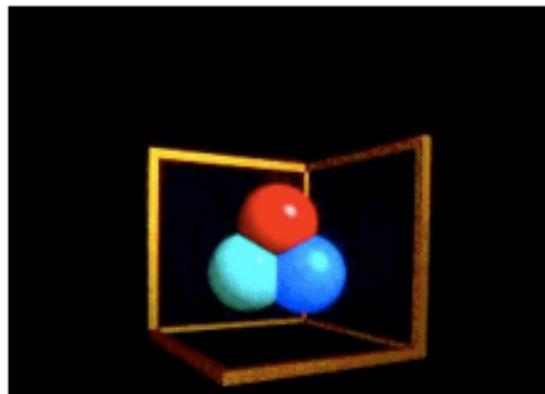
 If transparent

 col += k_trans * trace transmitted ray

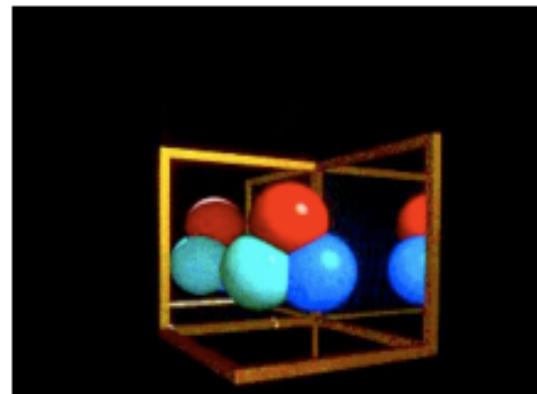


Does it ever end?

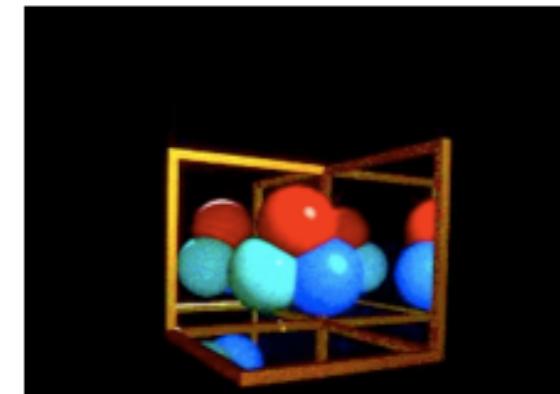
- Stopping criteria:
 - Recursion depth: Stop after a number of bounces
 - Ray contribution: Stop if reflected / transmitted contribution becomes too small



0 recursion



1 recursion



2 recursions

Ray tracing: Primary rays

- For each ray we need to test which objects are intersecting the ray:
 - If the object has an intersection with the ray we calculate the distance between viewpoint and intersection
 - If the ray has more than one intersection, the smallest distance identifies the visible surface.
- Primary rays are rays from the view point to the nearest intersection point
- Local illumination is computed as before:

$$L = k_a + (k_d(\mathbf{n} \cdot \mathbf{l}) + k_s(\mathbf{v} \cdot \mathbf{r})^q)I_s$$

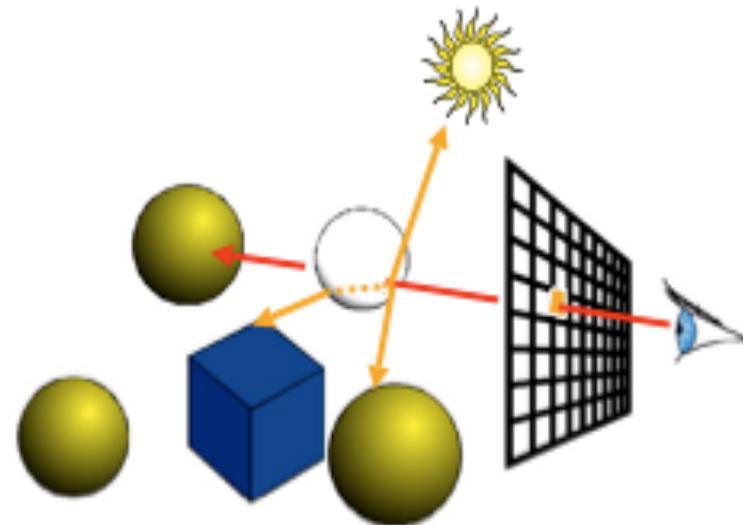
Ray tracing: Secondary rays

- Secondary rays are rays originating at the intersection points
- Secondary rays are caused by
 - rays reflected off the intersection point in the direction of reflection
 - rays transmitted through transparent materials in the direction of refraction
 - shadow rays

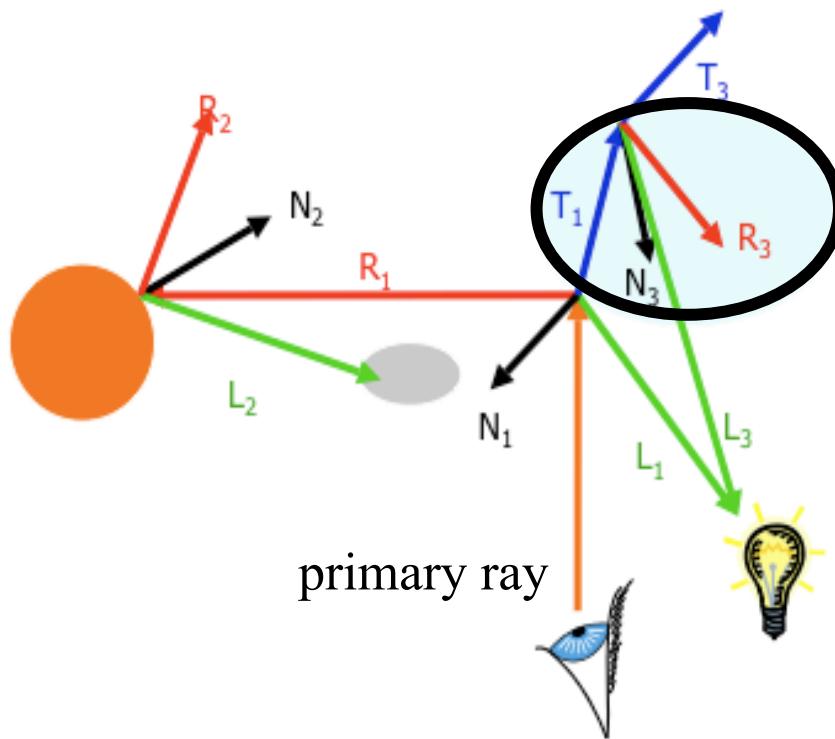
Recursive ray tracing: Putting it all together

- Illumination can be expressed as

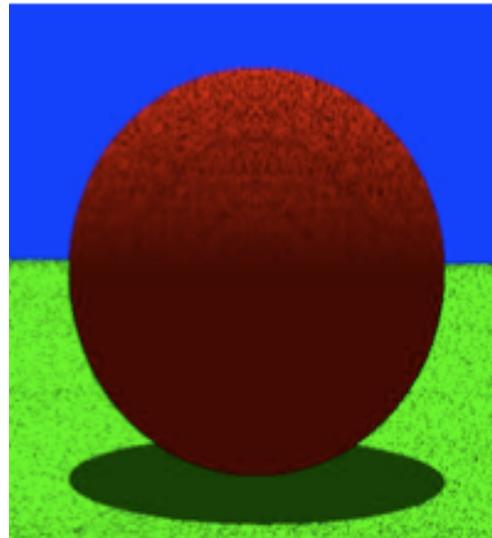
$$L = k_a + (k_d(\mathbf{n} \cdot \mathbf{l}) + k_s(\mathbf{v} \cdot \mathbf{r})^q)I_s + k_{reflected}L_{reflected} + k_{refracted}L_{refracted}$$



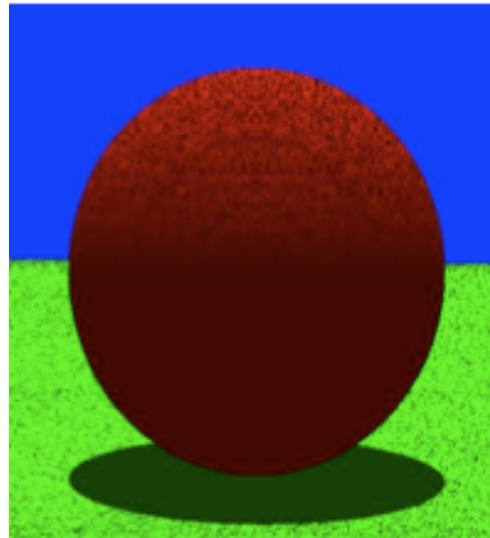
Recursive Ray Tracing: Ray Tree



Precision Problems



Precision Problems



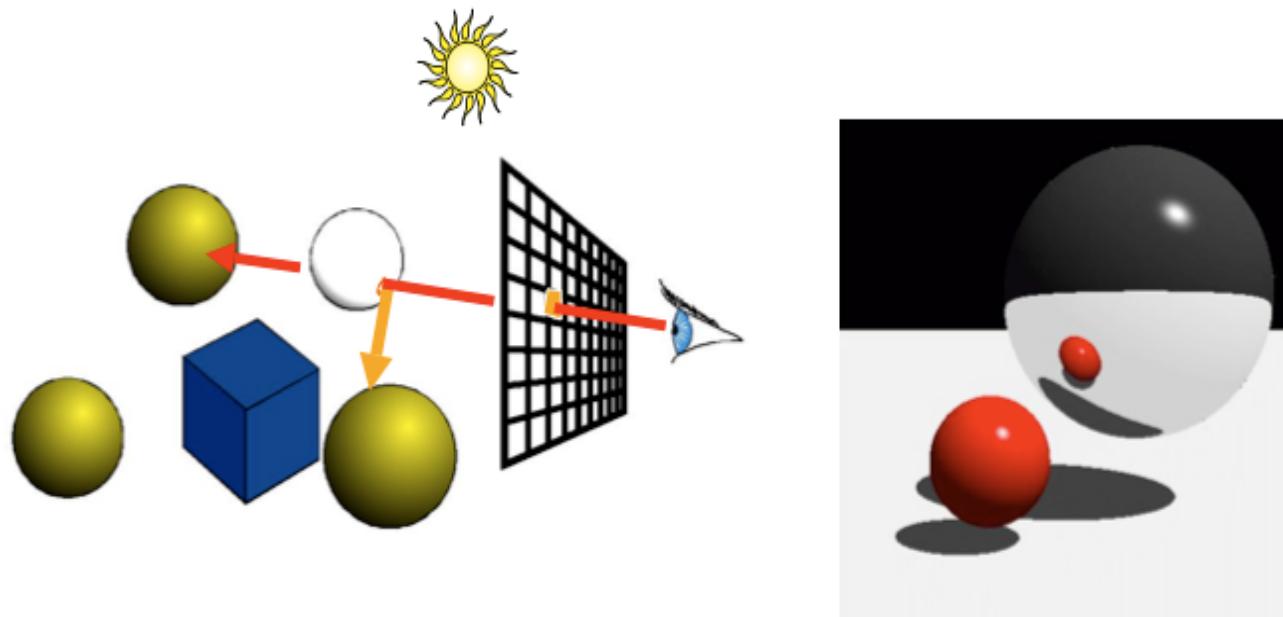
- In ray tracing, the origin of (secondary) rays is often on the surface of objects
 - Theoretically, the intersection point should be on the surface
 - Practically, calculation imprecision creeps in, and the origin of the new ray is slightly beneath the surface
- Result: the surface area is shadowing itself

ε to the rescue ...

- Check if t is within some epsilon tolerance:
 - if $\text{abs}(\mu) < \varepsilon$
 - point is on the surface
 - else
 - point is inside/outside
 - Choose the ε tolerance empirically
- Move the intersection point by epsilon along the surface normal so it is outside of the object
- Check if point is inside/outside surface by checking the sign of the implicit (sphere etc.) equation

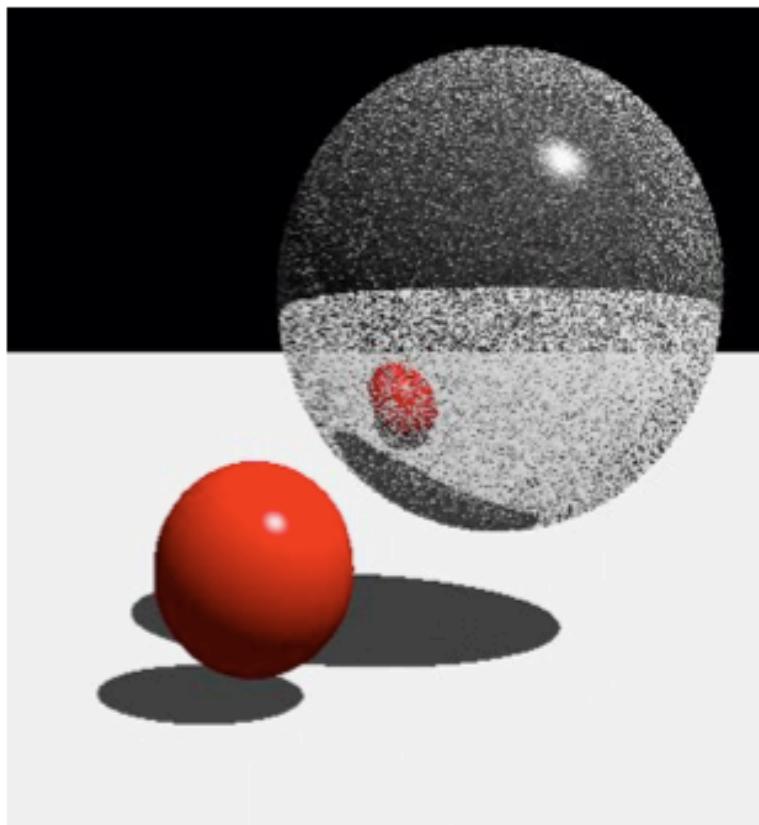
Mirror reflection

- Compute mirror contribution
- Cast ray in direction symmetric wrt. normal
- Multiply by reflection coefficient (color)

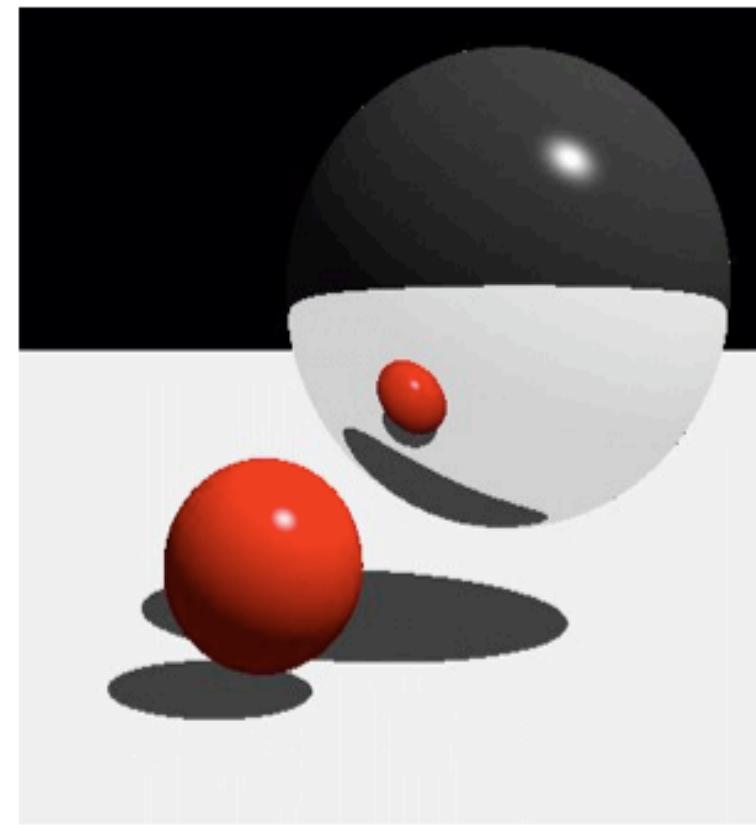


Mirror reflection

- Don't forget to add epsilon to the ray

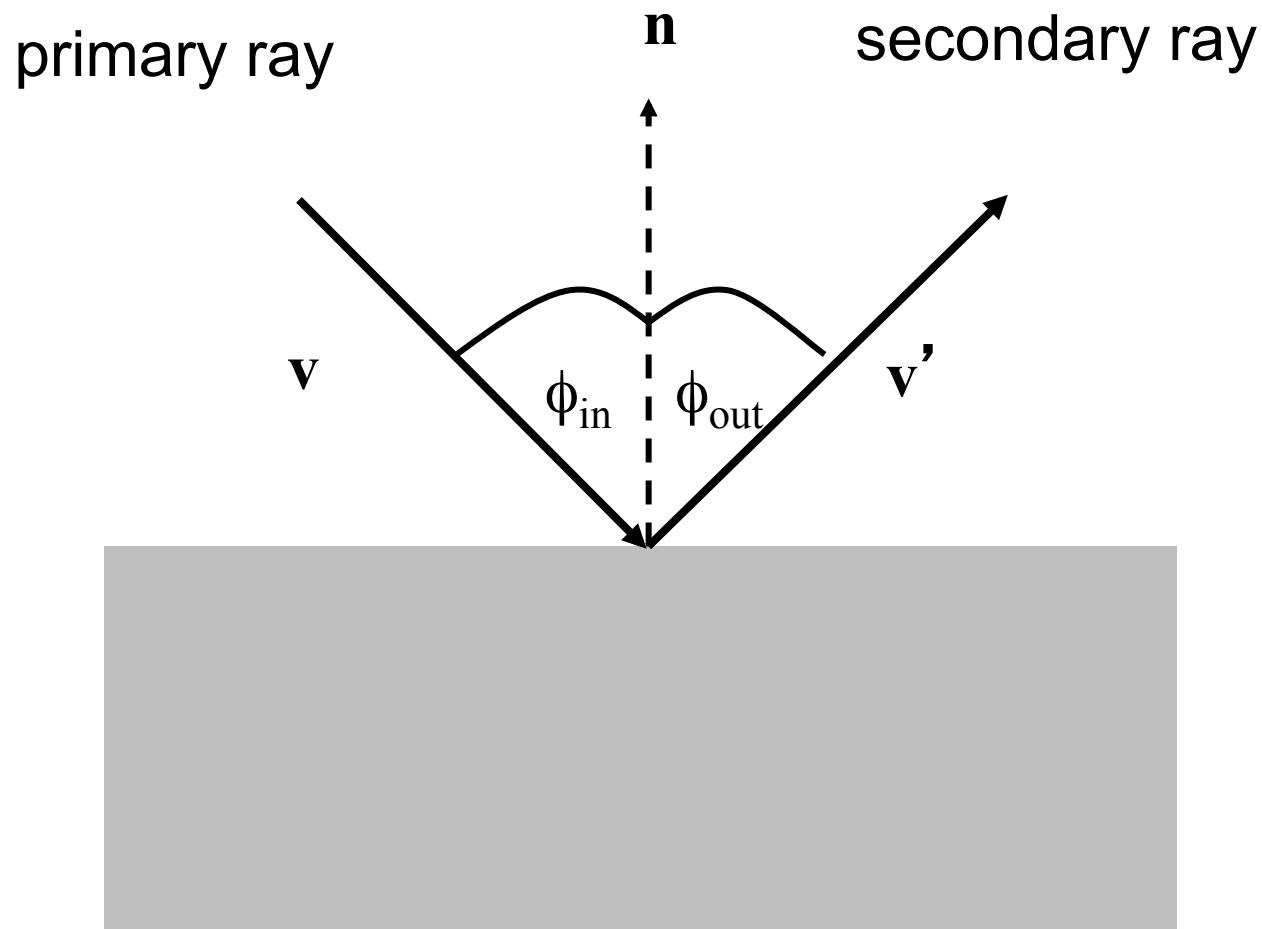


Without epsilon



With epsilon

Mirror reflection



Mirror reflection

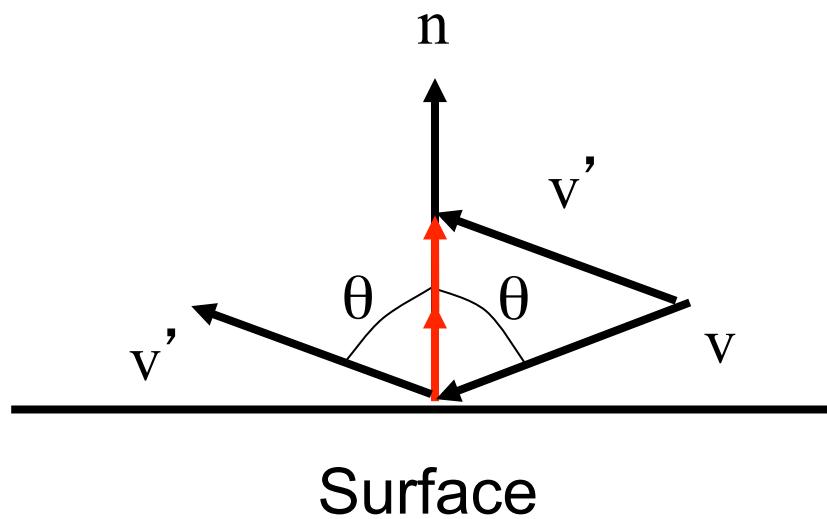
- To calculate illumination as a result of reflections
 - calculate the direction of the secondary ray at the intersection of the primary ray with the object.

given that

- \mathbf{n} is the unit surface normal
- \mathbf{v} is the direction of the primary ray
- \mathbf{v}' is the direction of the secondary ray as a result of reflections

$$\mathbf{v}' = \mathbf{v} - (2\mathbf{v} \cdot \mathbf{n})\mathbf{n}$$

Mirror reflection



$$\mathbf{v}' = (2\mathbf{v} \cdot \mathbf{n})\mathbf{n} - \mathbf{v}$$

Mirror reflection

The \mathbf{v} , \mathbf{v}' and \mathbf{n} are unit vectors and coplanar so:

$$\mathbf{v}' = \alpha \mathbf{v} + \beta \mathbf{n}$$

Taking the dot product with \mathbf{n} yields the eq.:

$$\mathbf{n} \cdot \mathbf{v}' = \alpha \mathbf{v} \cdot \mathbf{n} + \beta = \mathbf{v} \cdot \mathbf{n}$$

Requiring \mathbf{v}' to be a unit vector yields the second eq.:

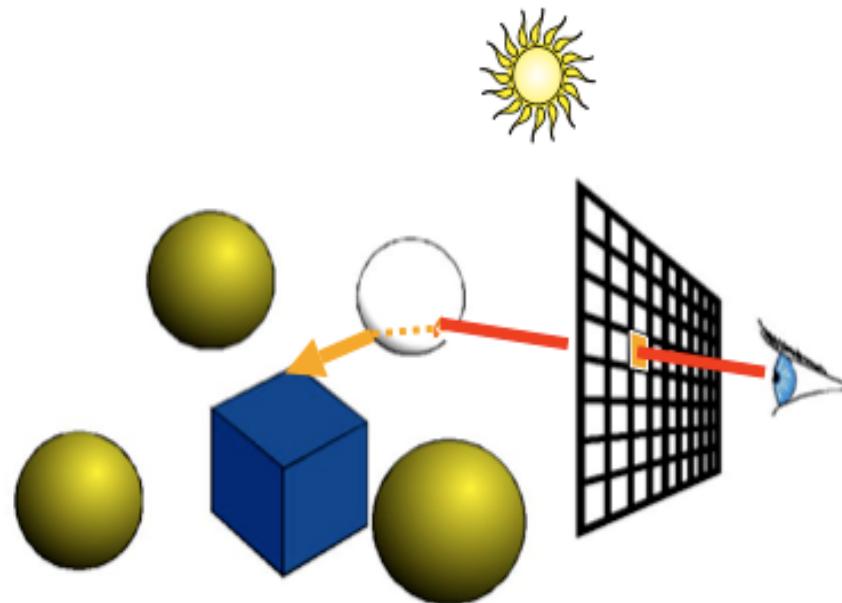
$$1 = \mathbf{v}' \cdot \mathbf{v}' = \alpha^2 + 2 \alpha \beta \mathbf{v} \cdot \mathbf{n} + \beta^2$$

Solving both equations yields:

$$\mathbf{v}' = (2\mathbf{v} \cdot \mathbf{n})\mathbf{n} - \mathbf{v}$$

Transparency

- Compute transmitted contribution
- Cast ray in refracted direction
- Multiply by transparency coefficient

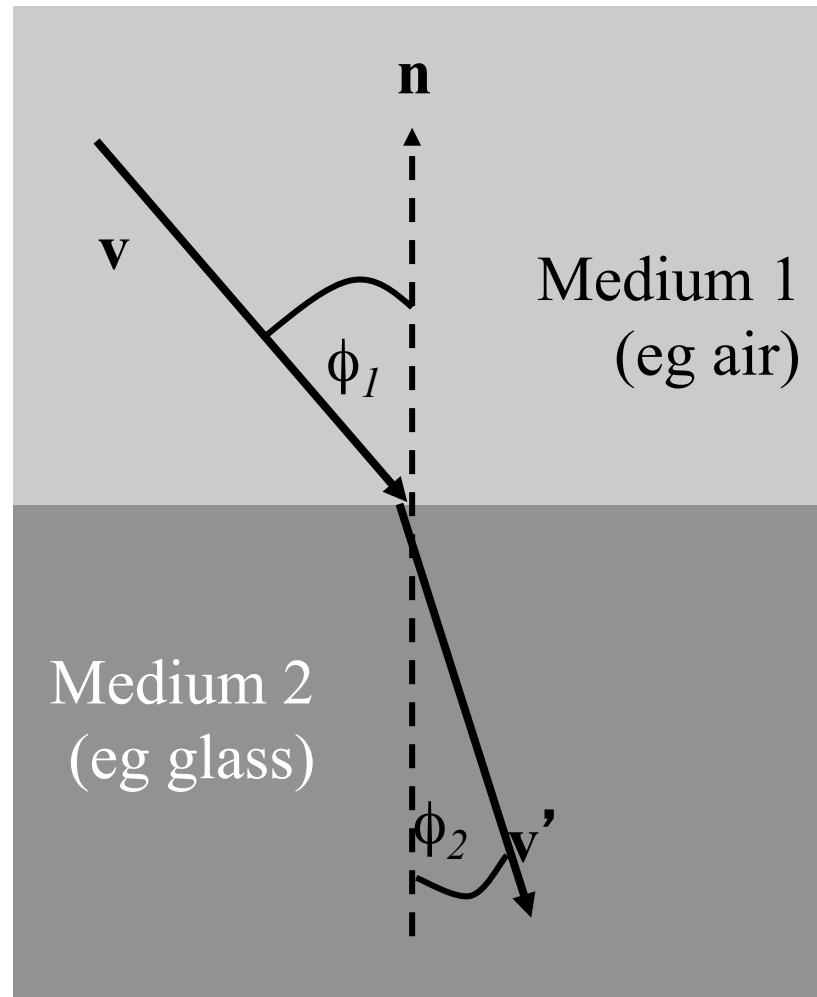


Refraction

- The angle of the refracted ray can be determined by Snell's law:

$$\eta_1 \sin(\phi_1) = \eta_2 \sin(\phi_2)$$

- η_1 is a constant for medium 1
- η_2 is a constant for medium 2
- ϕ_1 is the angle between the incident ray and the surface normal
- ϕ_2 is the angle between the refracted ray and the surface normal



Refraction

- In vector notation Snell's law can be written:

$$k_1(\mathbf{v} \times \mathbf{n}) = k_2(\mathbf{v}' \times \mathbf{n})$$

- The direction of the refracted ray is

$$\mathbf{v}' = \frac{\eta_1}{\eta_2} \left(\sqrt{(\mathbf{n} \cdot \mathbf{v})^2 + \left(\frac{\eta_2}{\eta_1} \right)^2 - 1} - \mathbf{n} \cdot \mathbf{v} \right) \cdot \mathbf{n} + \mathbf{v}$$

Refraction

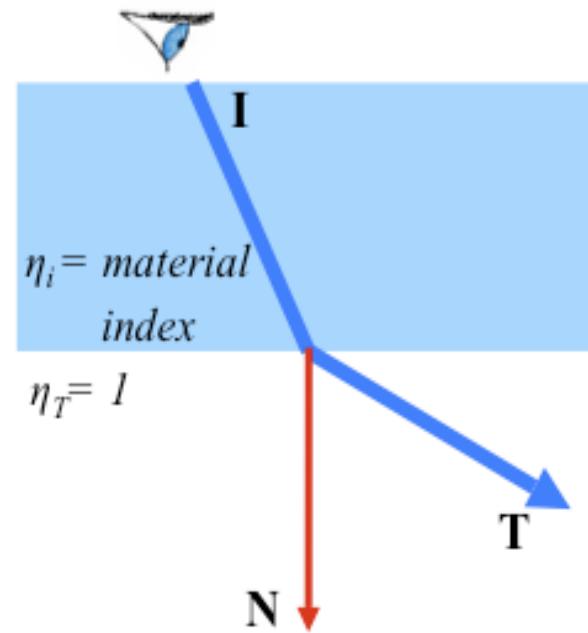
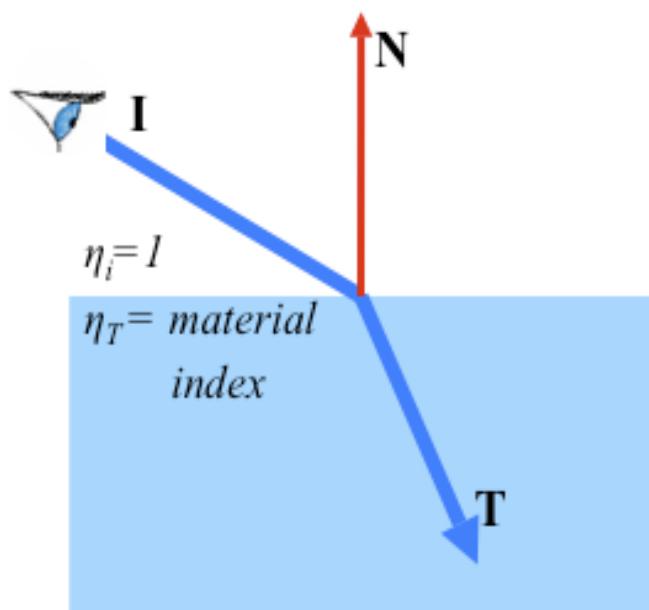
- This equation only has a solution if

$$(\mathbf{n} \cdot \mathbf{v})^2 > 1 - \left(\frac{\eta_2}{\eta_1} \right)^2$$

- This illustrates the physical phenomenon of the limiting angle:
 - if light passes from one medium to another medium whose index of refraction is low, the angle of the refracted ray is greater than the angle of the incident ray
 - if the angle of the incident ray is large, the angle of the refracted ray is larger than 90°
 - ➔ the ray is reflected rather than refracted

Refraction

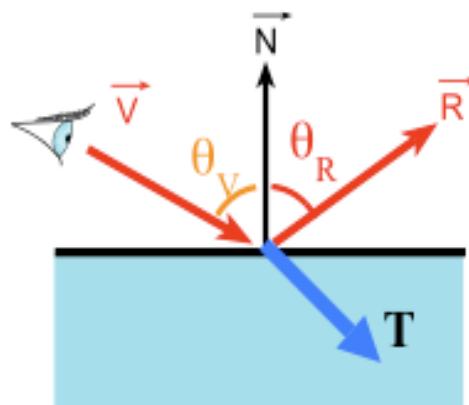
- Make sure you know whether you are entering or leaving the transmissive material



Amount of reflection and refraction

- Traditional (hacky) ray tracing
 - Constant coefficient reflection
 - Component per component multiplication
- Better: Mix reflected and refracted light according to the Fresnel factor.

$$L = k_{fresnel} L_{reflected} + (1 - k_{fresnel}) L_{refracted}$$

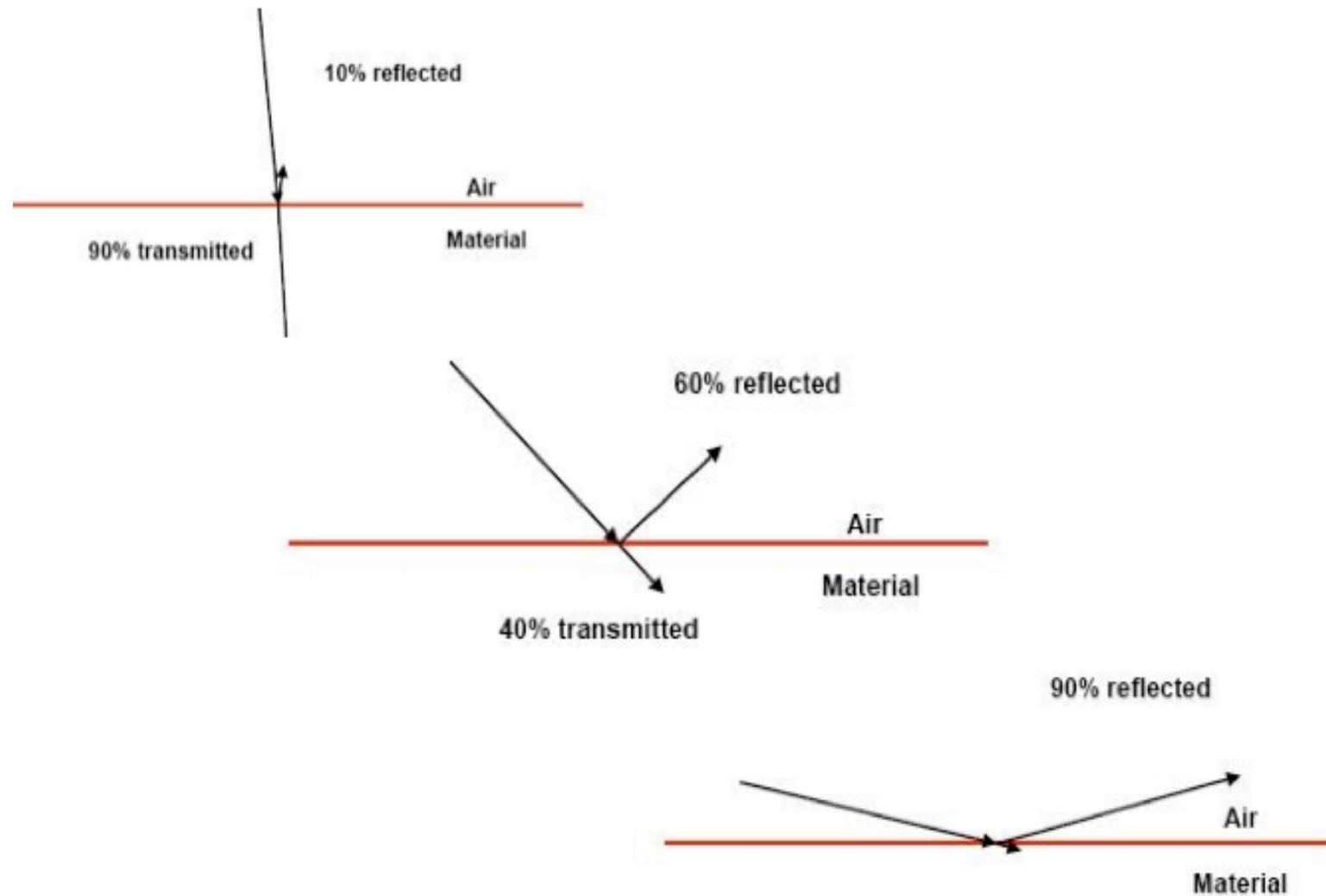


Fresnel factor

- More reflection at grazing angle



Fresnel factor



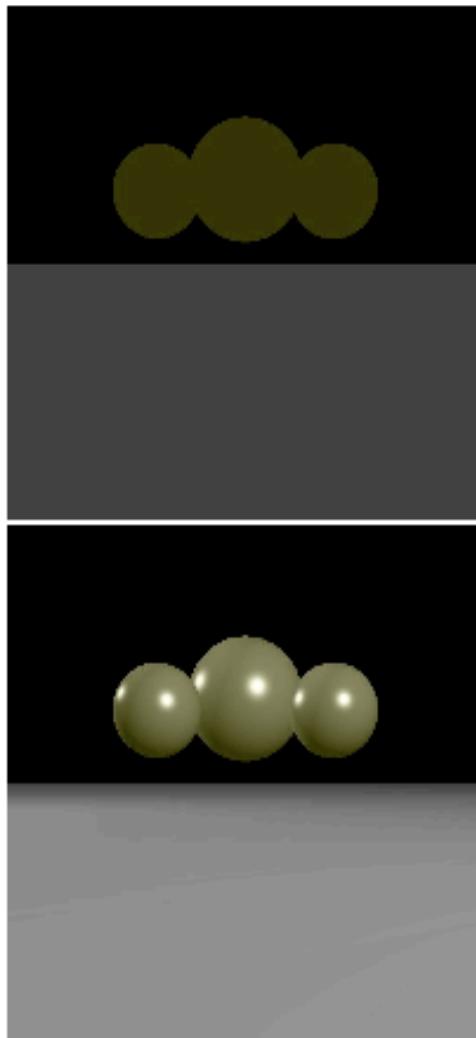
Schlick's Approximation

- Schlick' s approximation

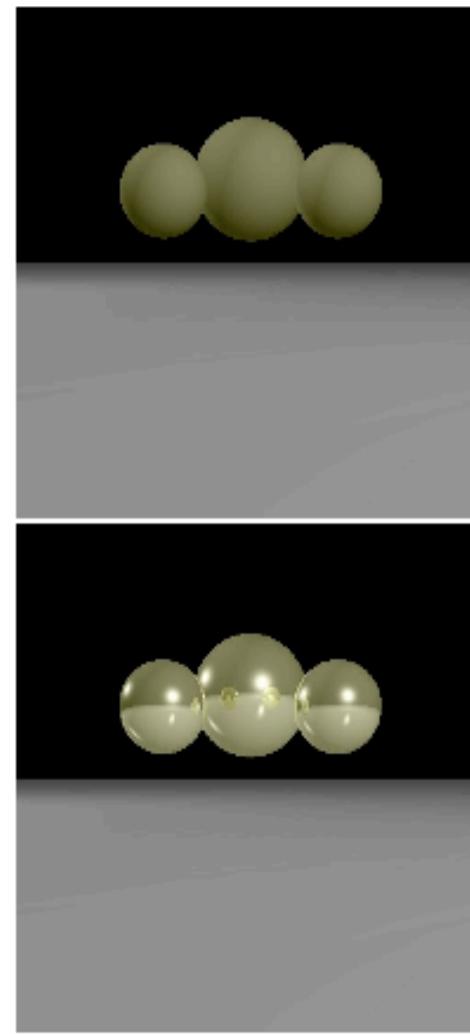
$$k_{fresnel}(\theta) = k_{fresnel}(0) + (1 - k_{fresnel}(0))(1 - (\mathbf{n} \cdot \mathbf{l}))^5$$

- $k_{fresnel}(0)$ = Fresnel factor at zero degrees
- Choose $k_{fresnel}(\theta) = 0.8$, this will look like stainless steel

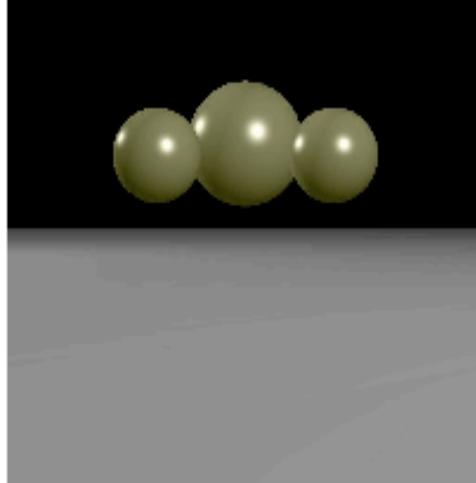
Example



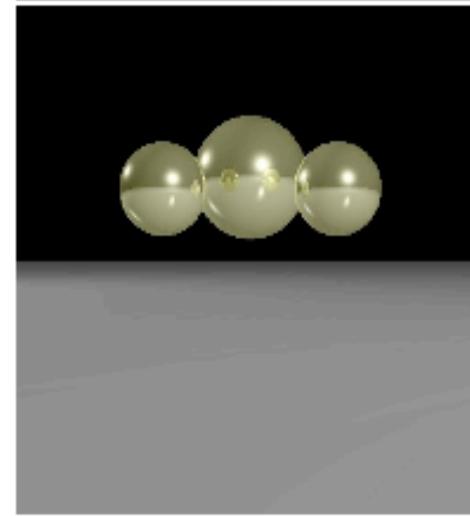
Ambient



+ Diffuse

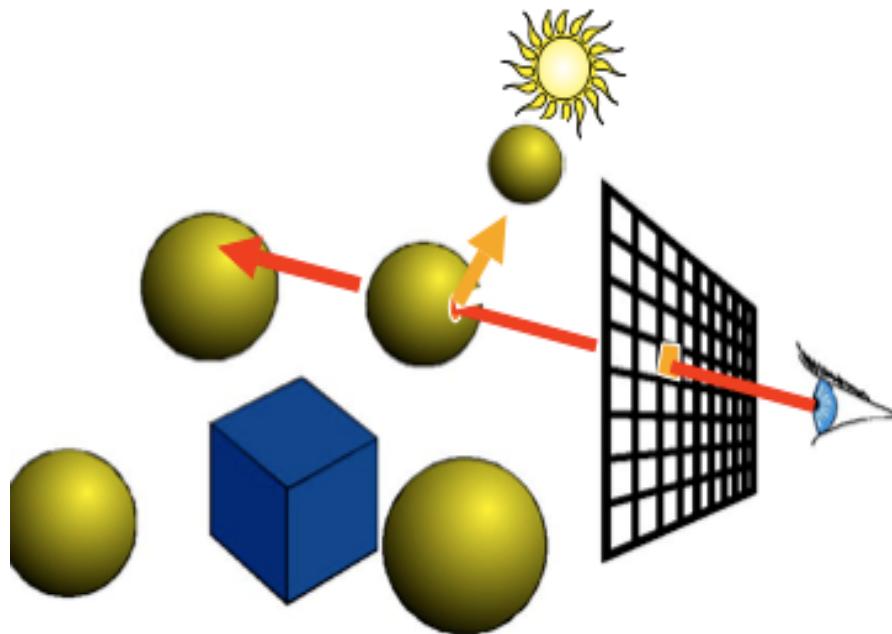


+ Specular

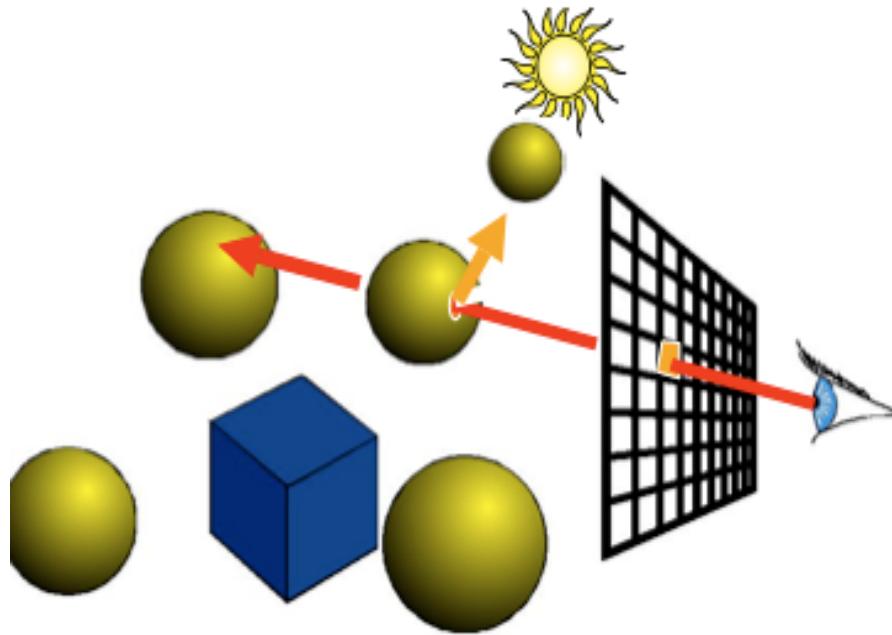


+ $I_{\text{reflected}}$

How do we add shadows?



How do we add shadows?

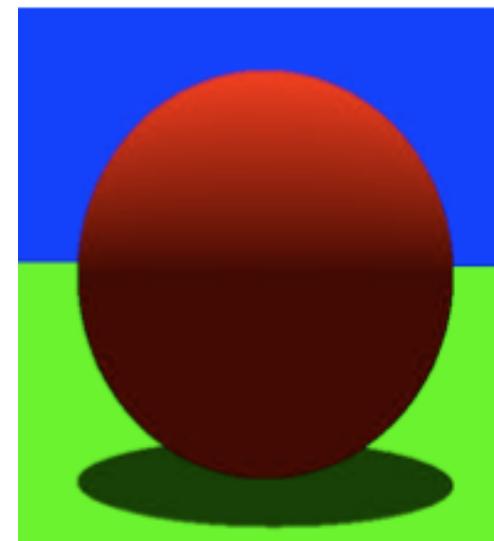
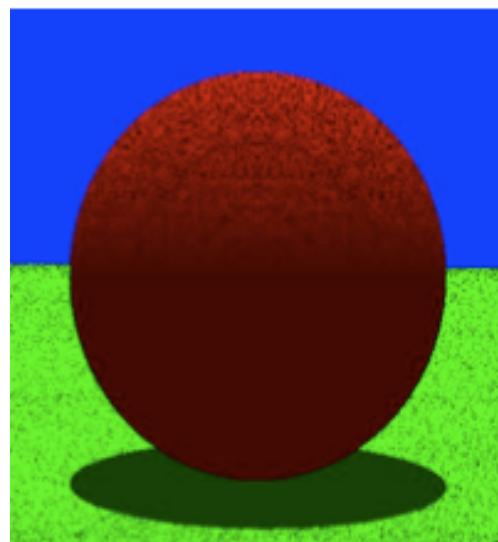


$$L = k_a + s(k_d(\mathbf{n} \cdot \mathbf{l}) + k_s(\mathbf{v} \cdot \mathbf{r})^q)I_s + k_{reflected}L_{reflected} + k_{refracted}L_{refracted}$$

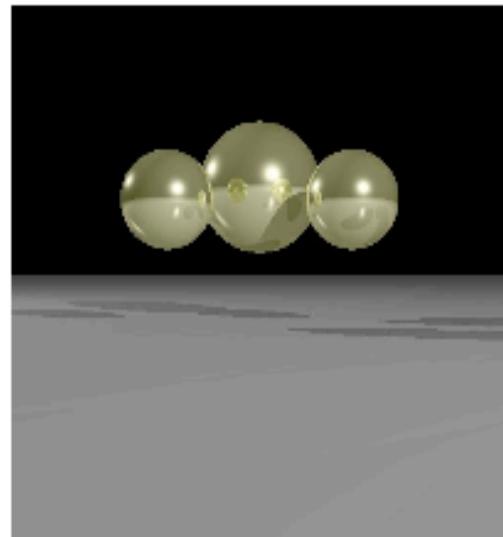
$$s = \begin{cases} 0 & \text{if light source is obscured} \\ 1 & \text{if light source is not obscured} \end{cases}$$

Shadows: Problems?

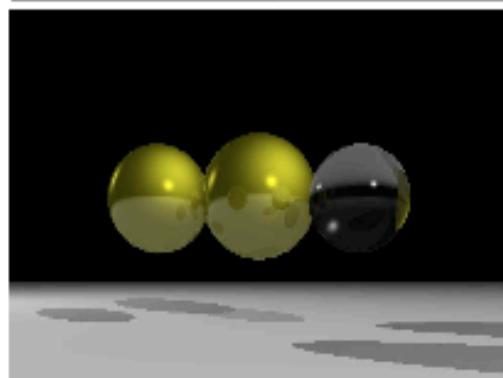
- Make sure to avoid self-shadowing



Example

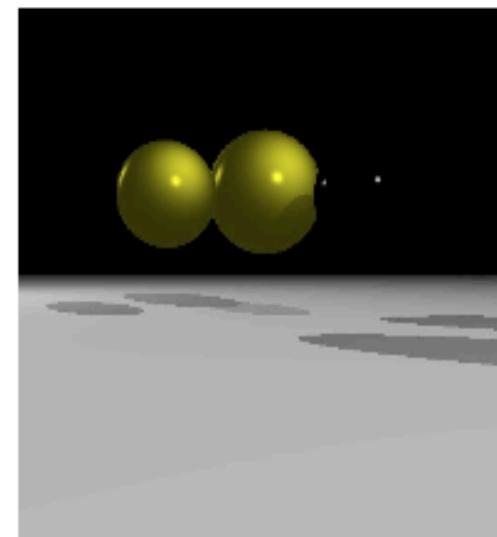


+ Shadows

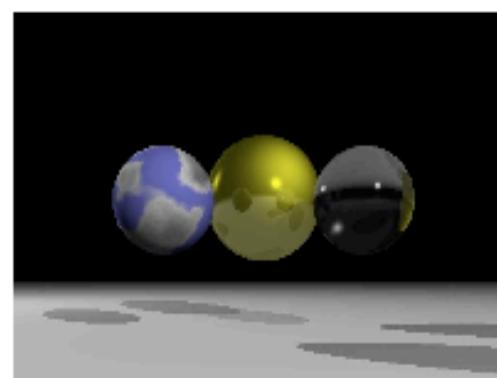


+ $I_{\text{reflected}}$

G



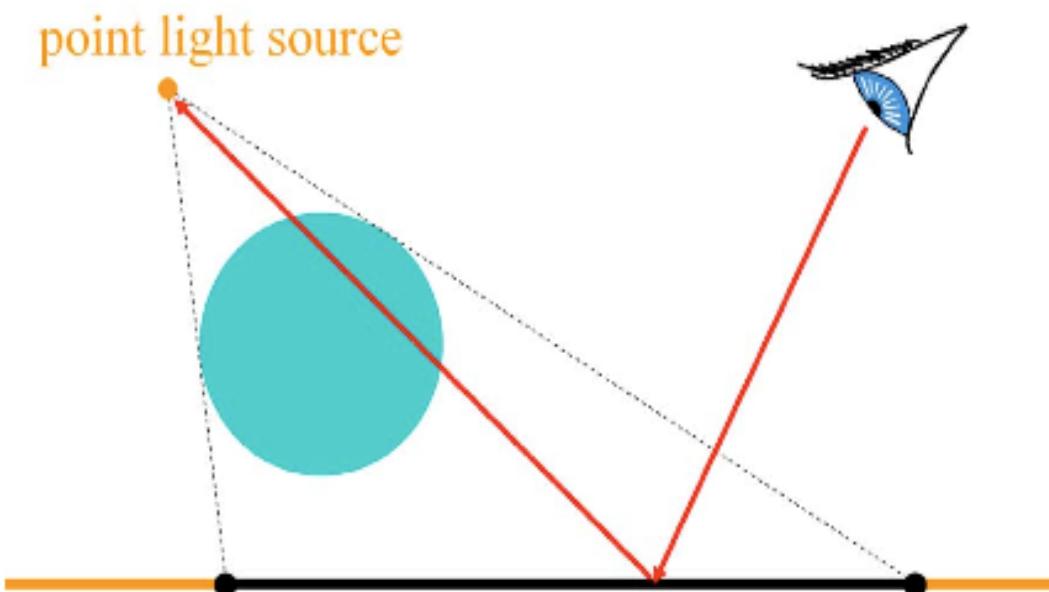
- $I_{\text{reflected}}$
+ transmitted



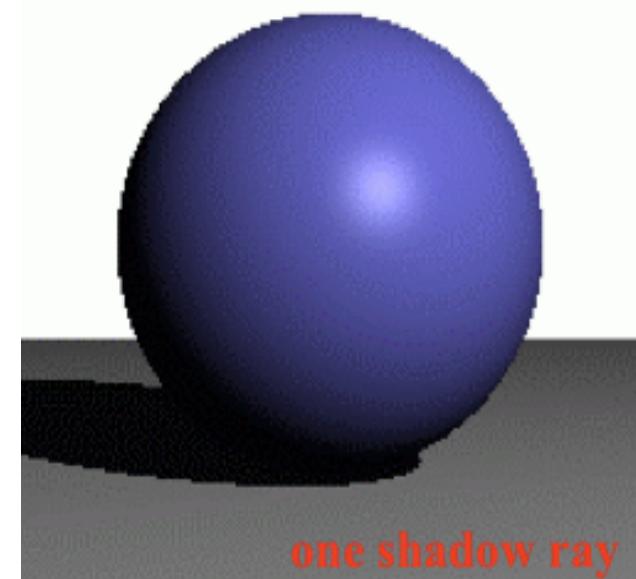
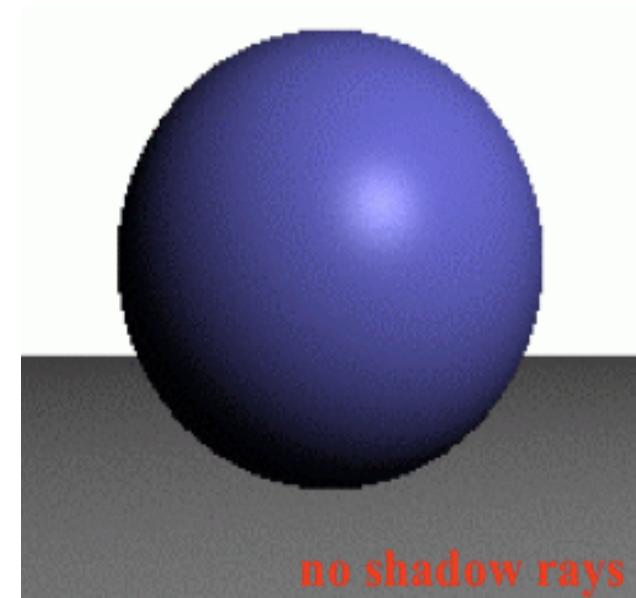
+ textures

Shadows

- One shadow ray per intersection per point light source

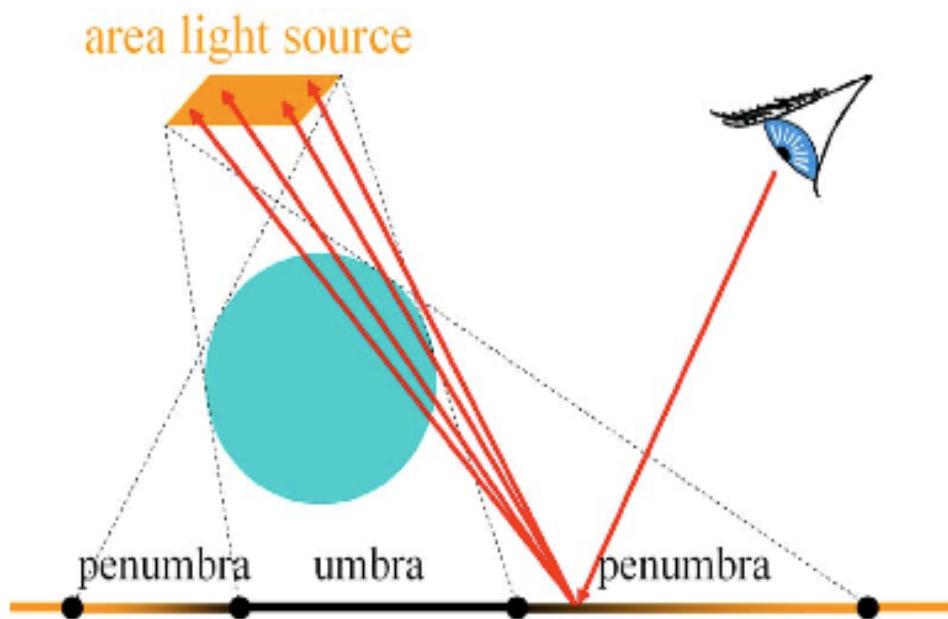


Graphics Lecture 9: Slide 40

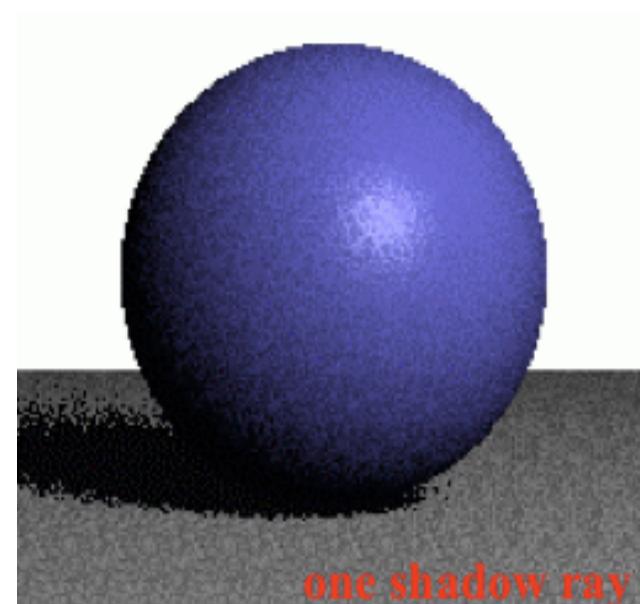


Soft shadows

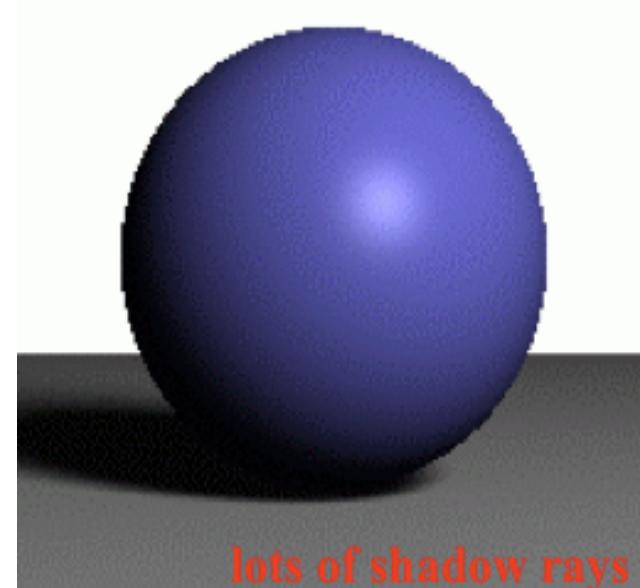
- Multiple shadow rays to sample area light source



Graphics Lecture 9: Slide 41



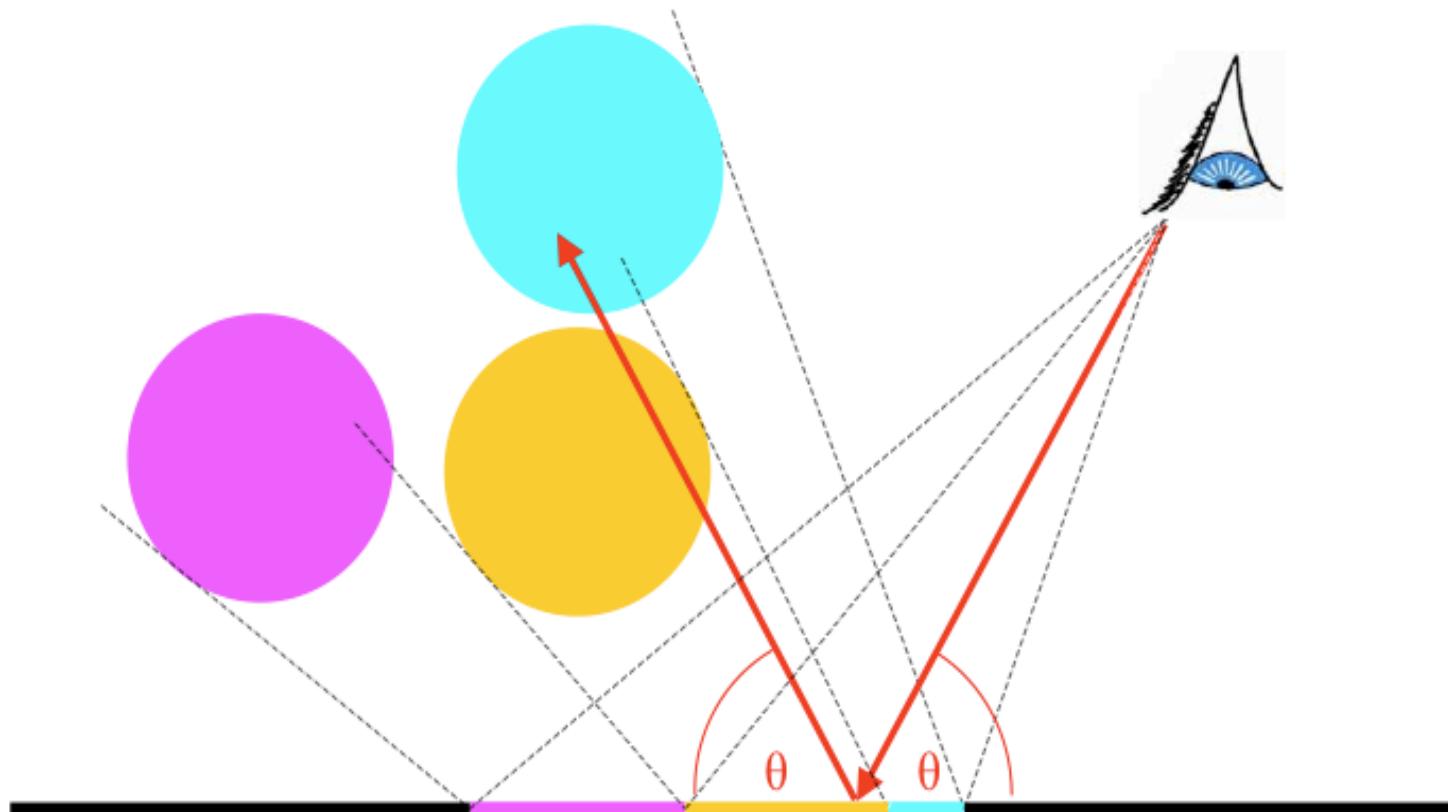
one shadow ray



lots of shadow rays

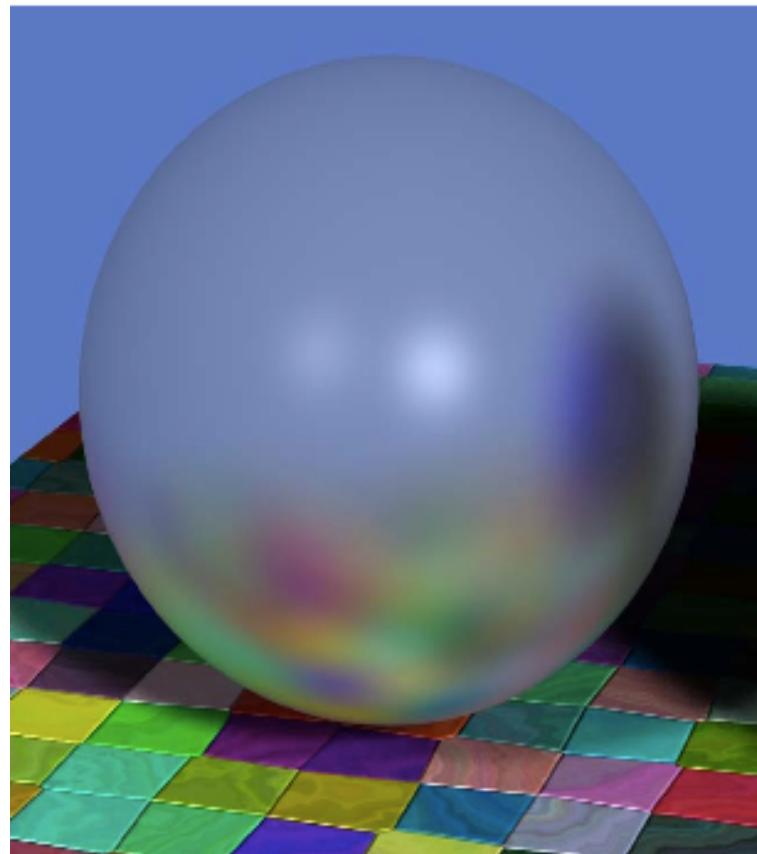
Reflection: Conventional ray tracing

- One reflection per intersection



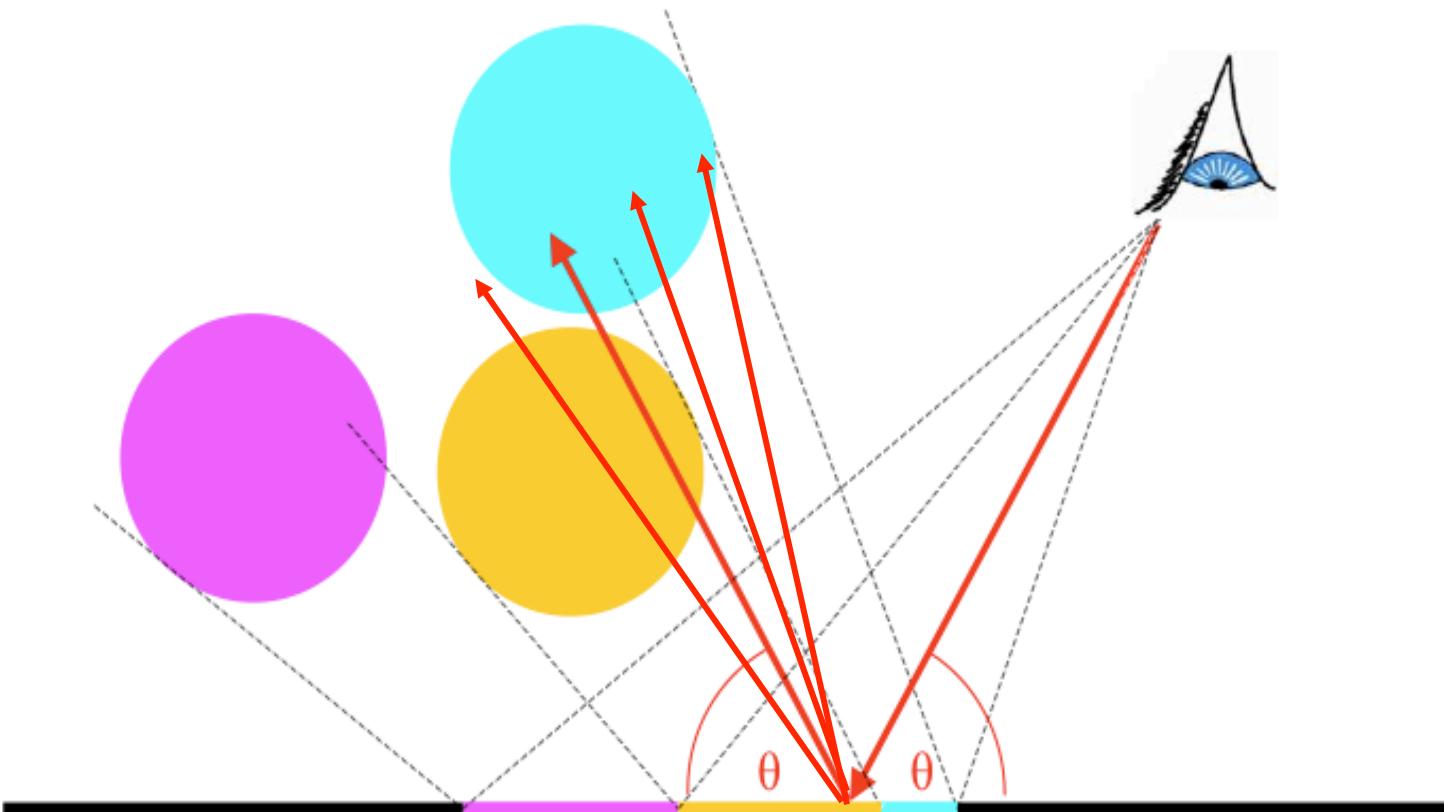
Reflection: Conventional ray tracing

- How can we create effects like this?

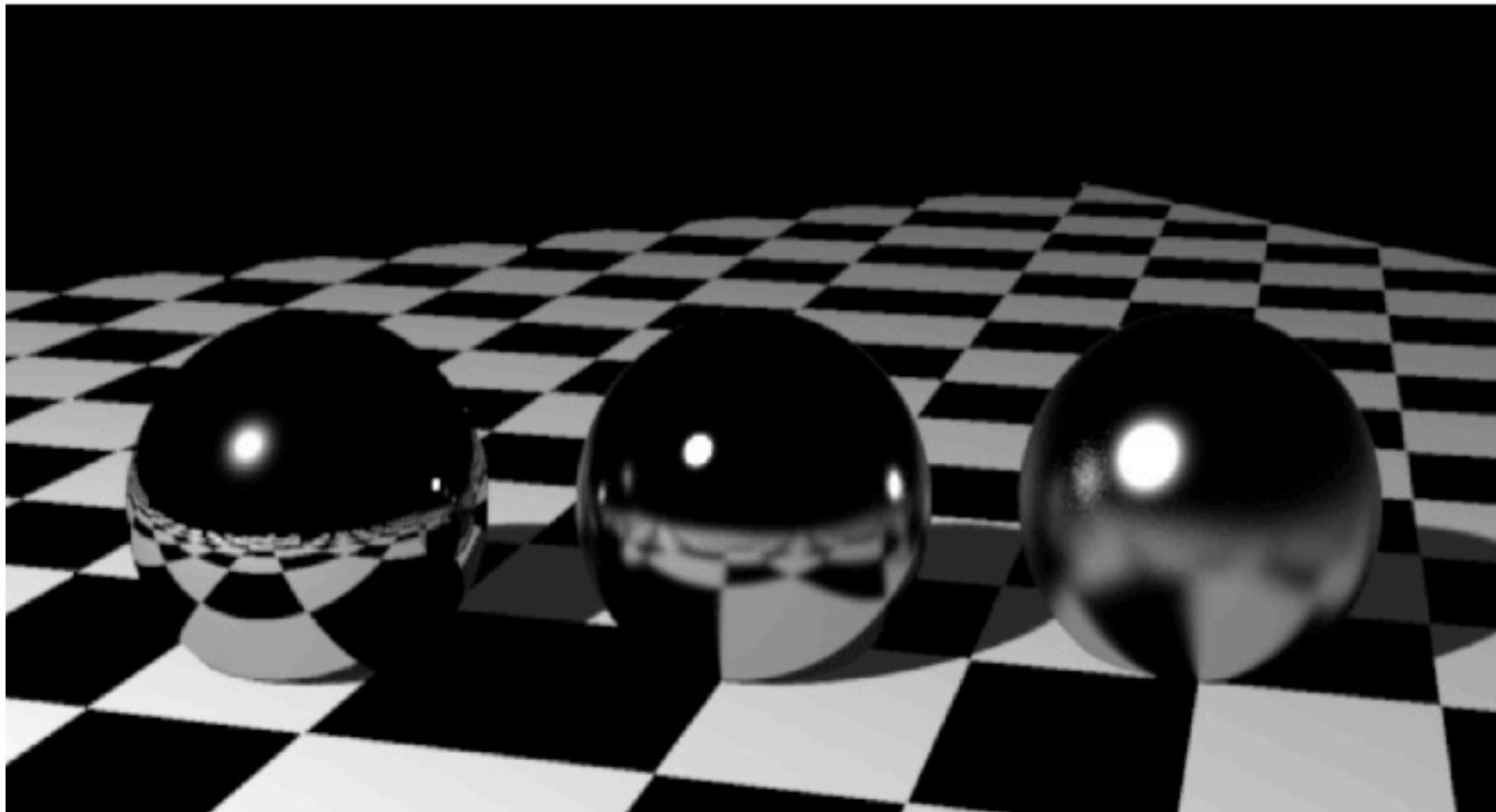


Reflection: Monte Carlo ray tracing

- Random reflection rays around mirror direction

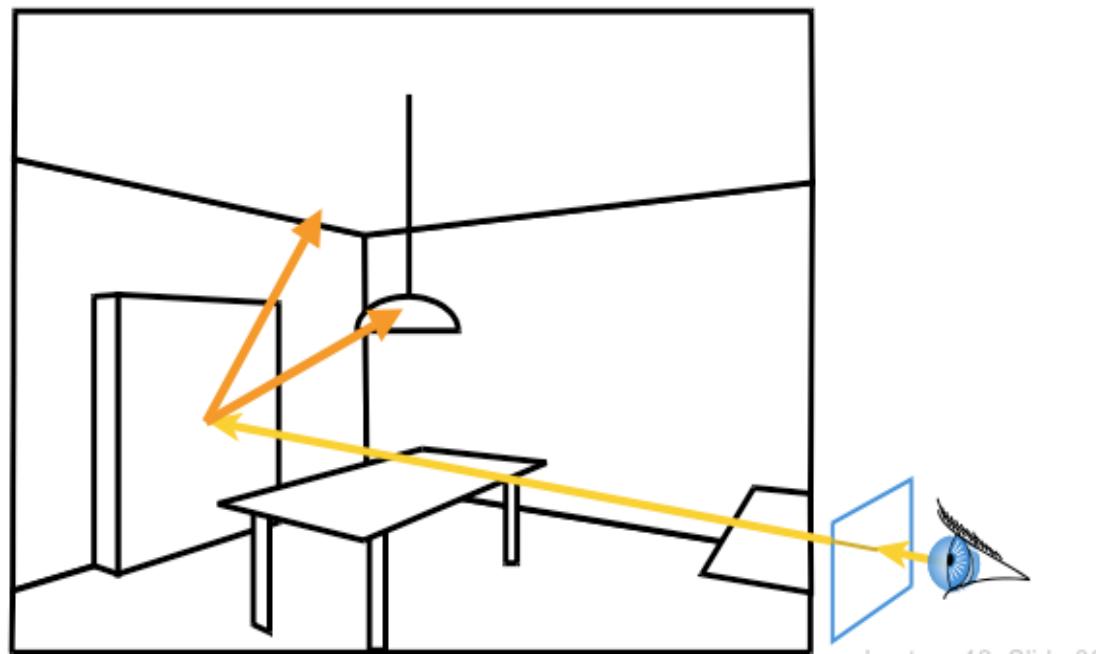


Glossy surfaces



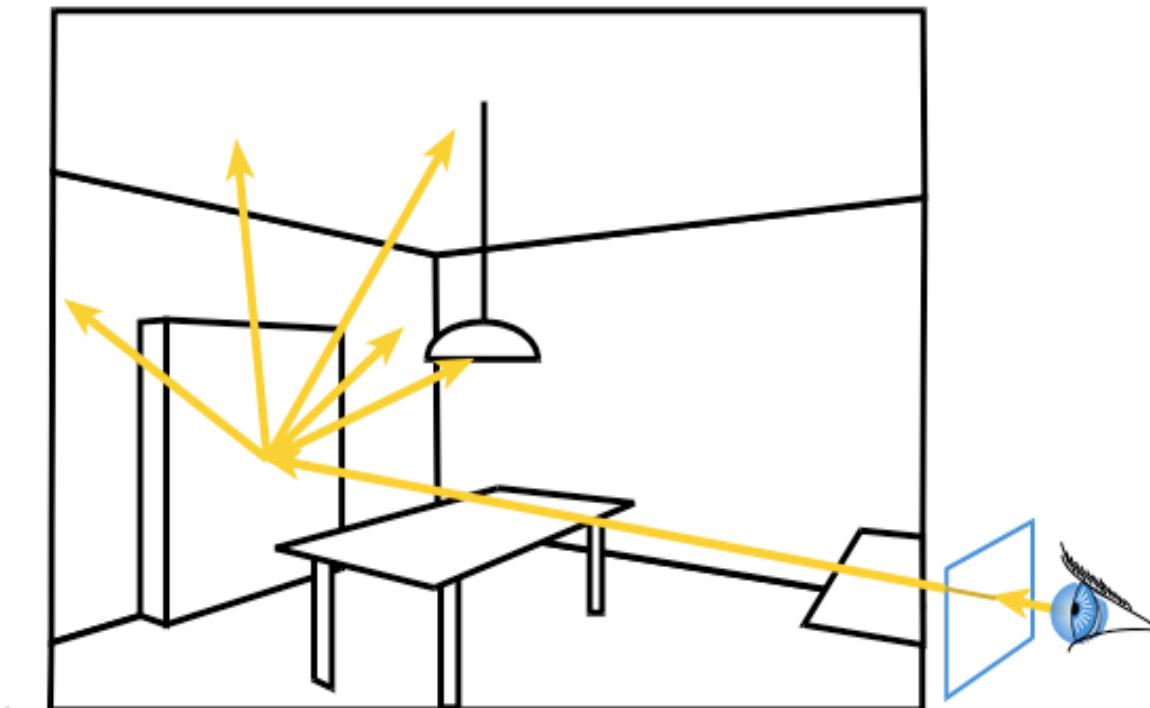
Ray tracing

- Cast a ray from the eye through each pixel
- Trace secondary rays (light, reflection, refraction)



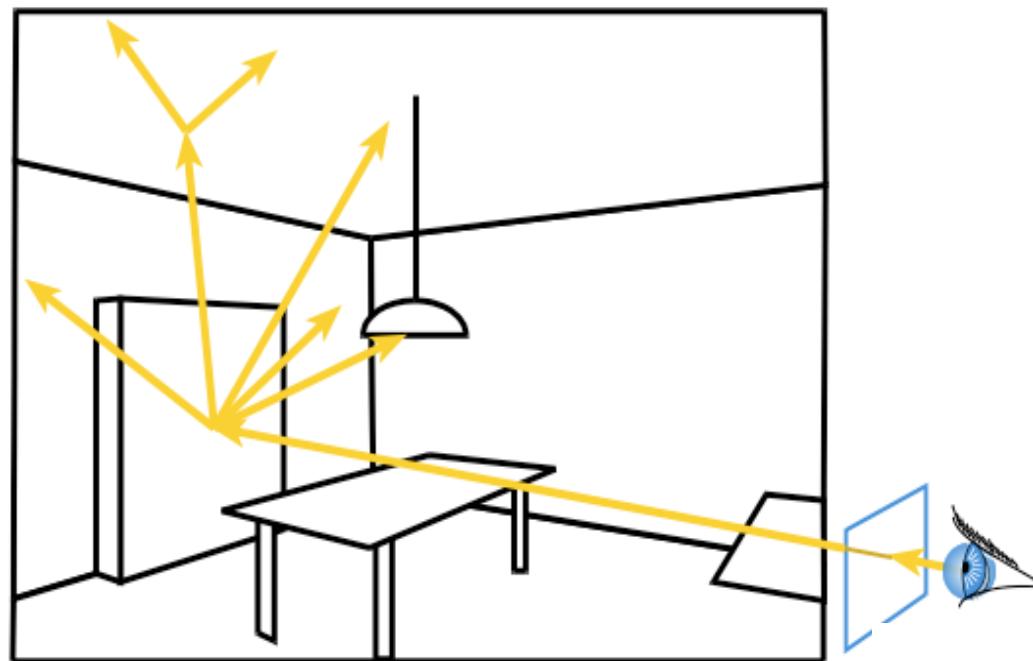
Monte-Carlo Ray Tracing

- Cast a ray from the eye through each pixel
- Cast random rays from the visible point
 - Accumulate radiance contribution



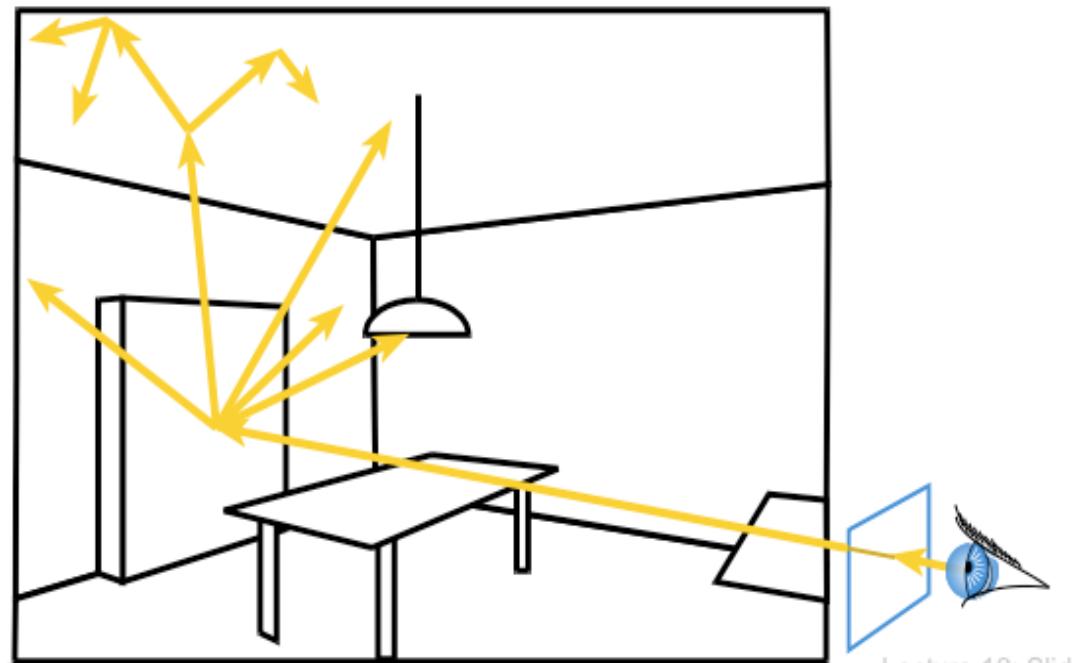
Monte-Carlo Ray Tracing

- Cast a ray from the eye through each pixel
- Cast random rays from the visible point
- Recurse



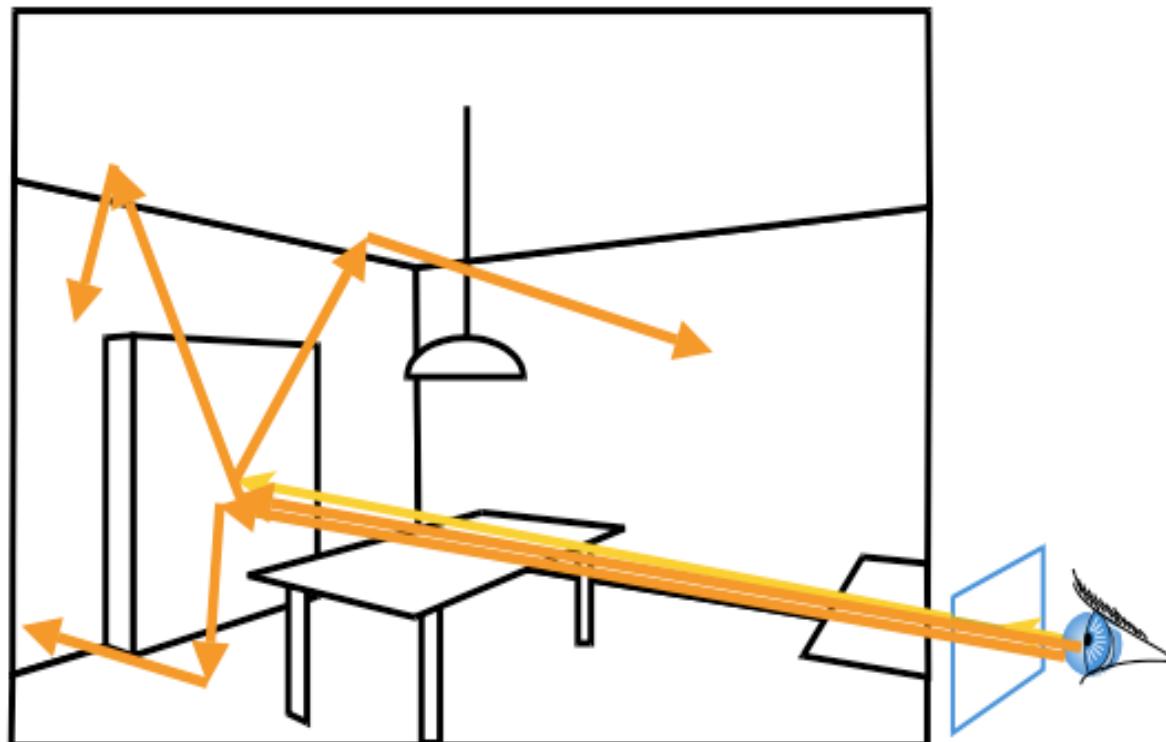
Monte-Carlo Ray Tracing

- Cast a ray from the eye through each pixel
- Cast random rays from the visible point
- Recurse



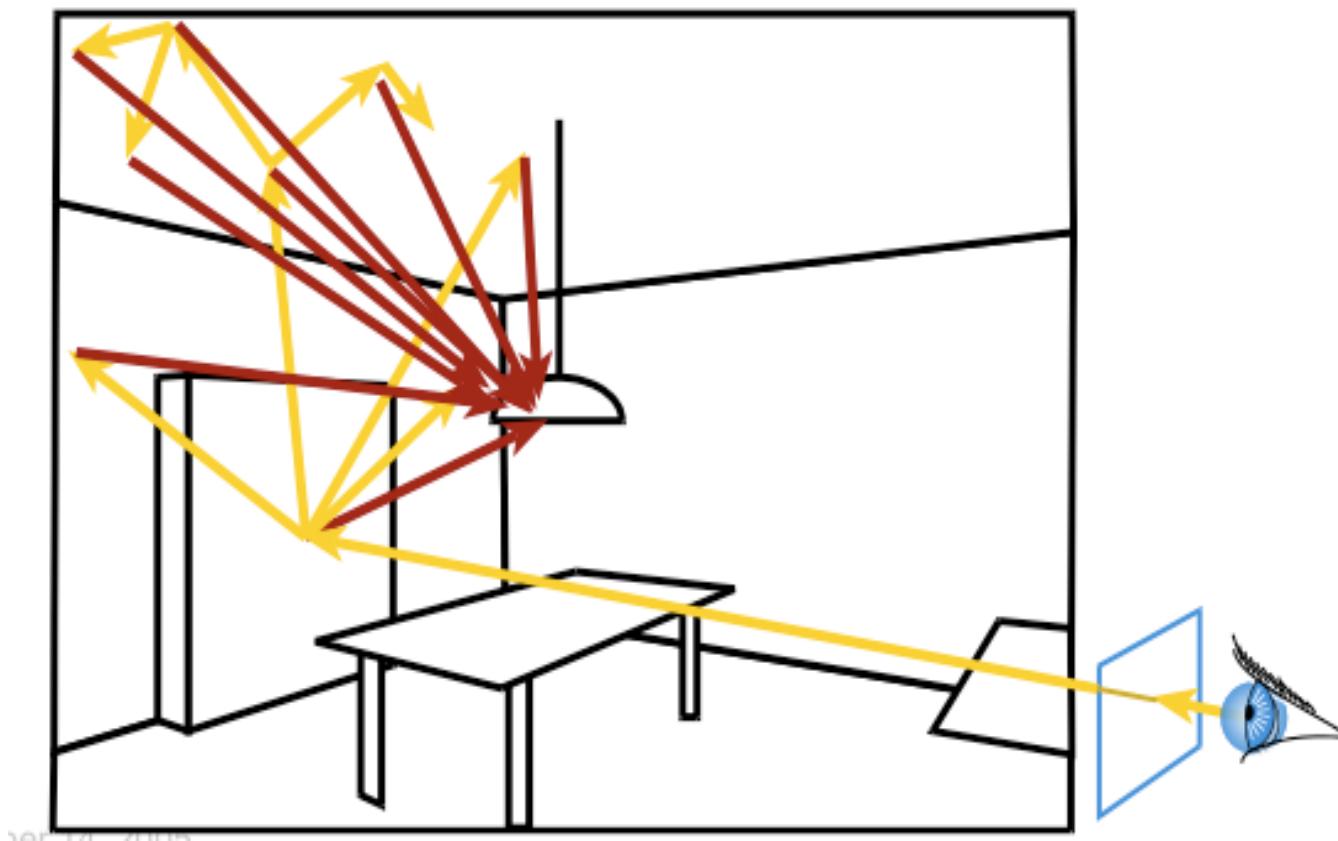
Monte Carlo Path Tracing

- Trace only one secondary ray per recursion
- But send many primary rays per pixel

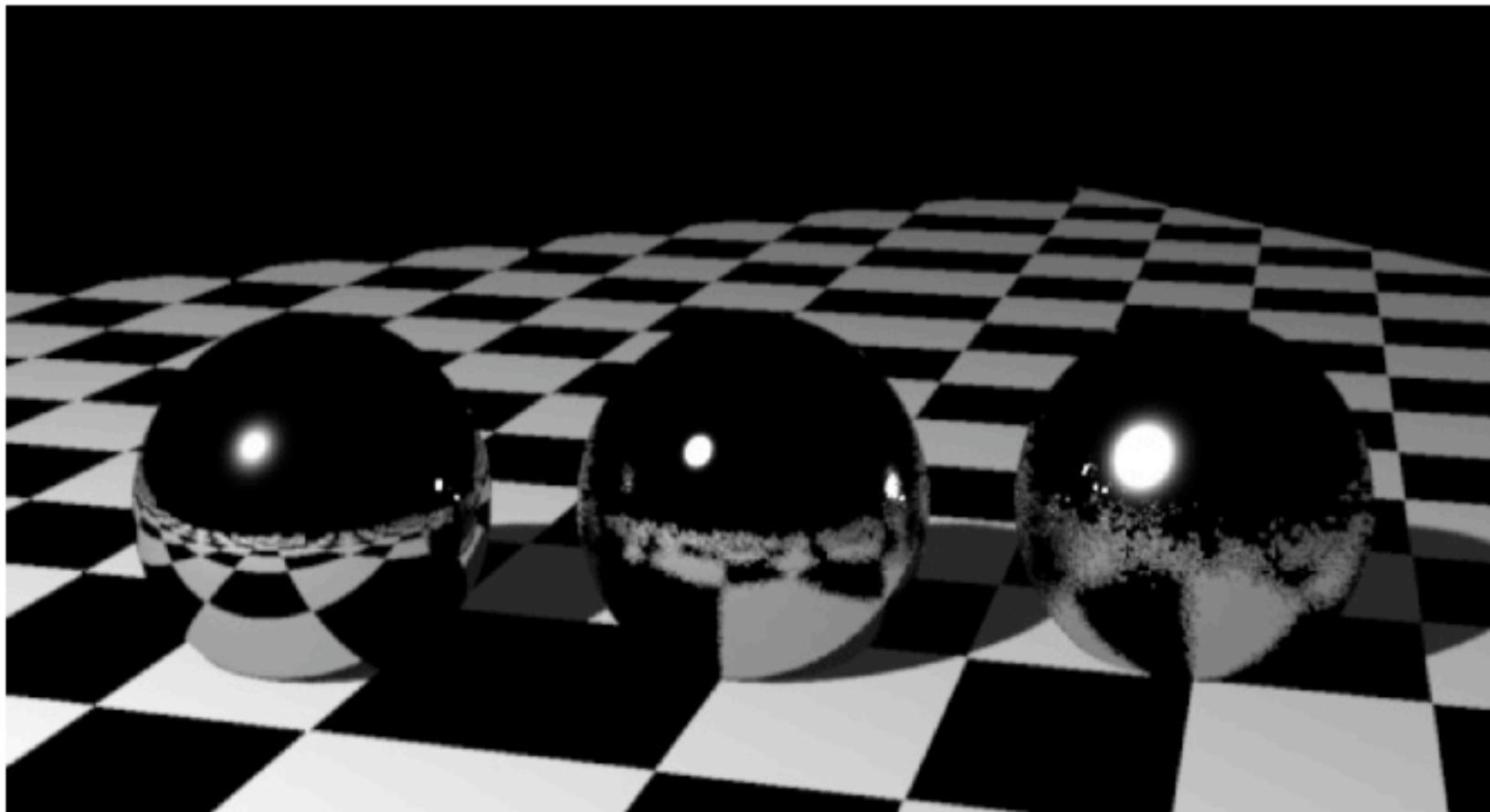


Monte Carlo Ray Tracing

- Send rays to light

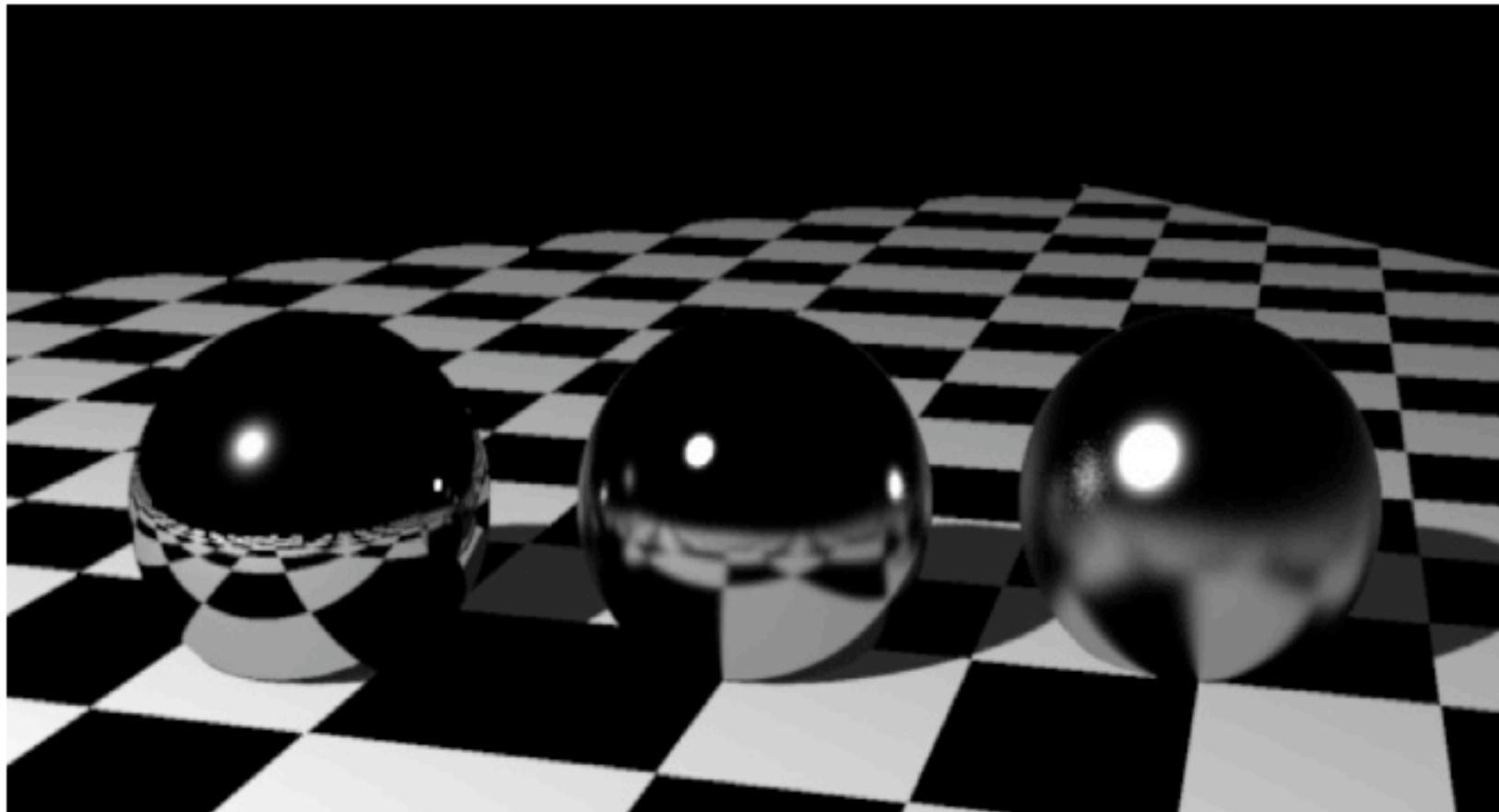


Example



1 sample per ray

Example



256 samples per ray

Some cool pictures



Copyright 2000 Gilles Tran

Some cool pictures



The Office - NBC | 14x01 | Seven Pictures | 100% Day 3 E

Some cool pictures



Some cool pictures



Some cool pictures



Graphics Lecture 9: Slide 58

took 4.5 days to render! (10 years ago)