Lecture #17

Ray Tracing – Secondary Effects

Computer Graphics

Winter Term 2016/17

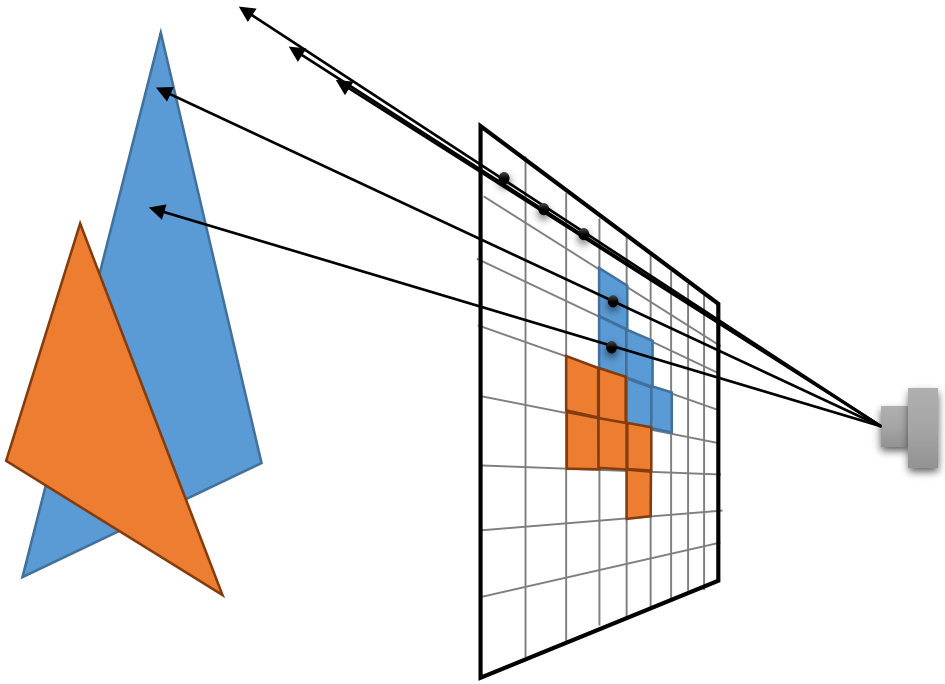
Marc Stamminger / Roberto Grosso

Ray Casting

- Ray Casting ⊂ Ray Tracing

```
for each pixel p
    cast a ray through pixel p
        shade p
```

= “find scene point visible in p”



eye rays only

Ray Tracing

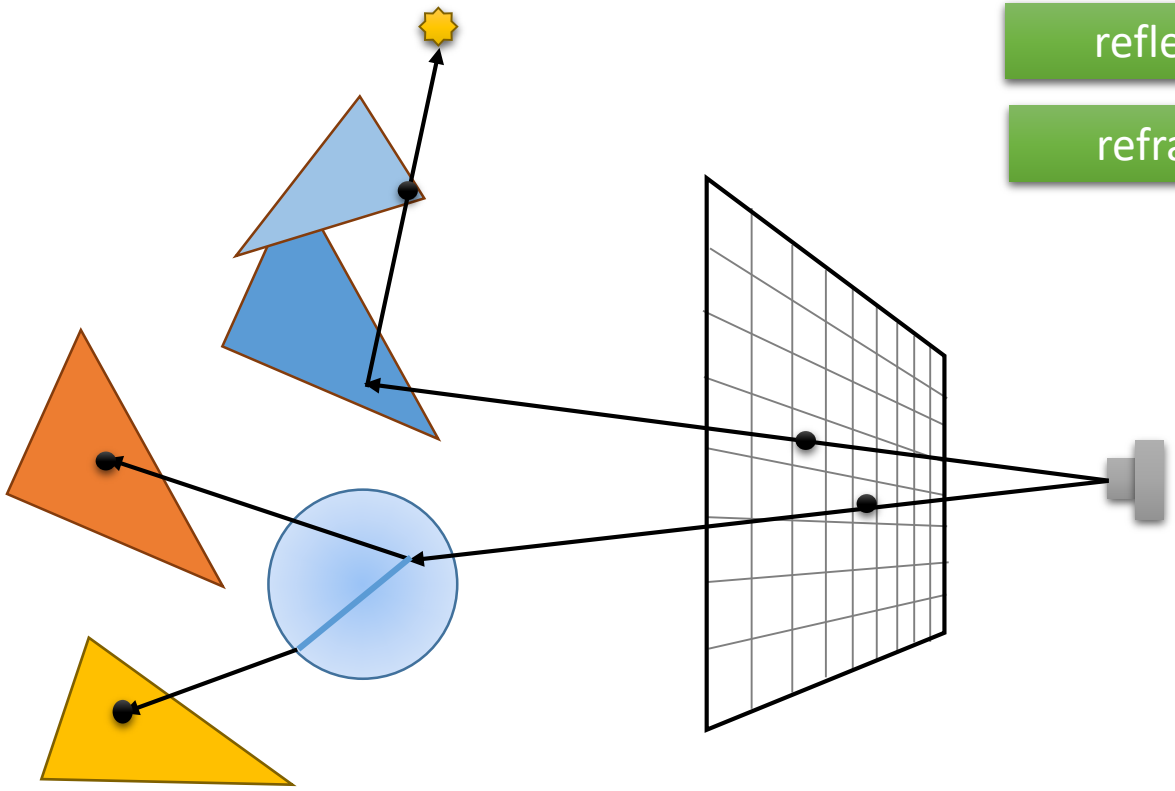
- Ray Casting → Ray Tracing

eye rays

shadow rays

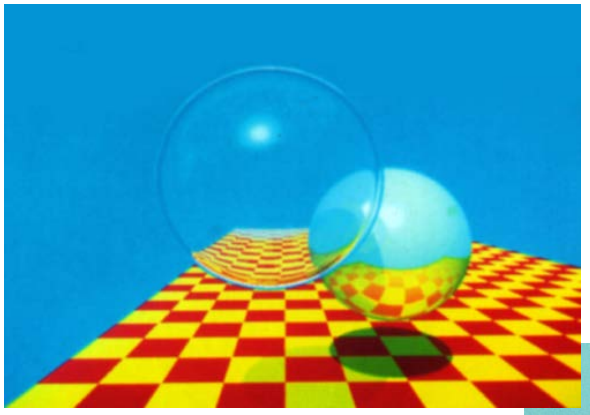
reflection rays

refraction rays

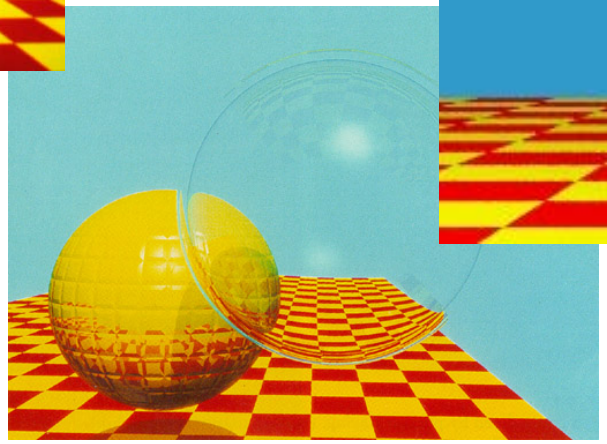


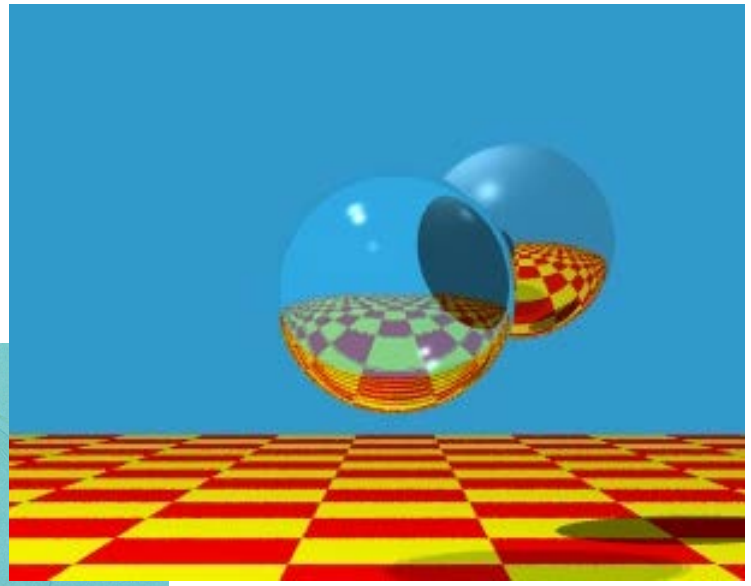
Introduction

- 1968: Ray Casting: Arthur Appel
- 1979: Recursive ray tracing: Turner Whitted



reflection
and
refraction





Images by Turner Whitted

Introduction

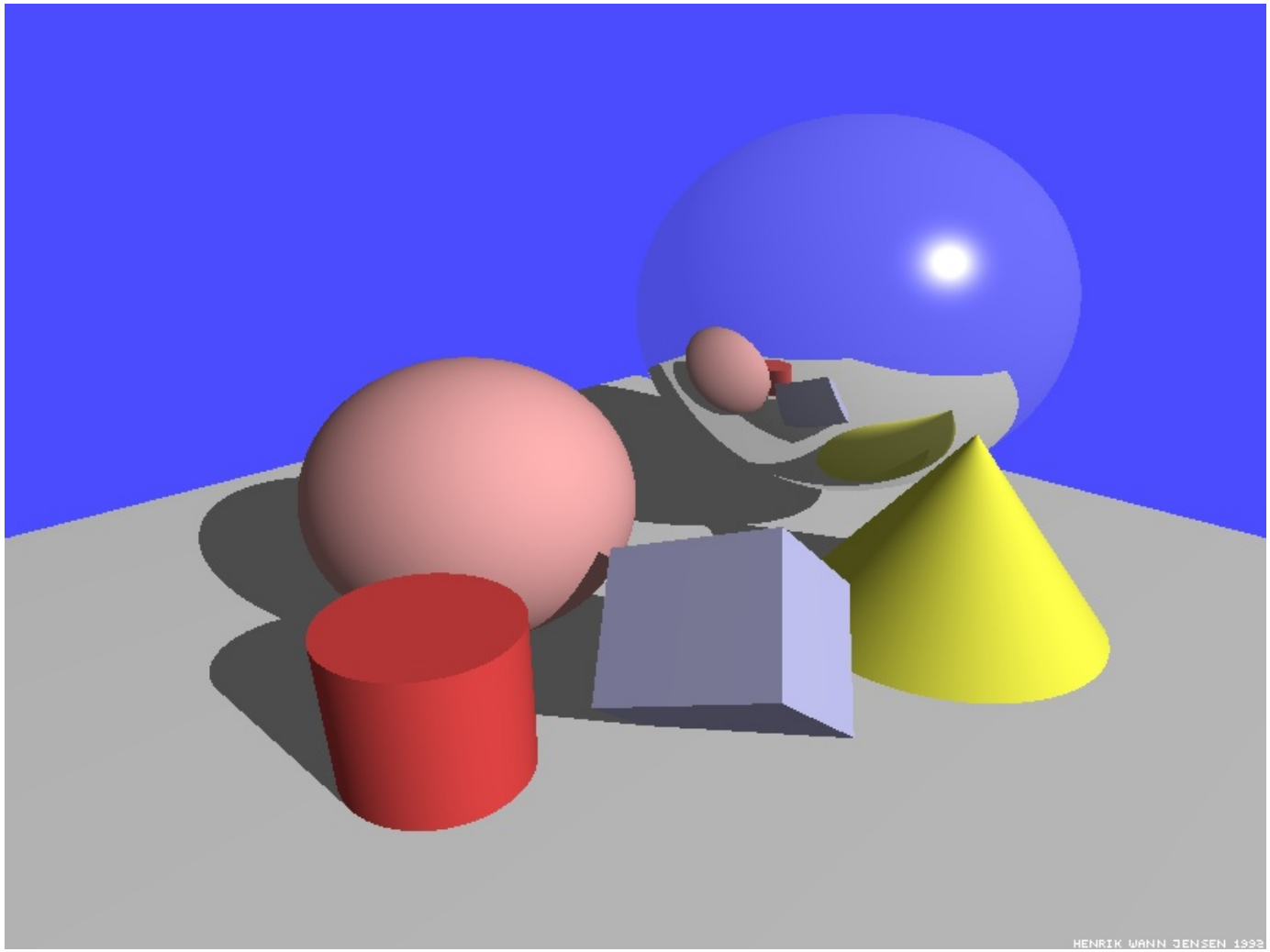
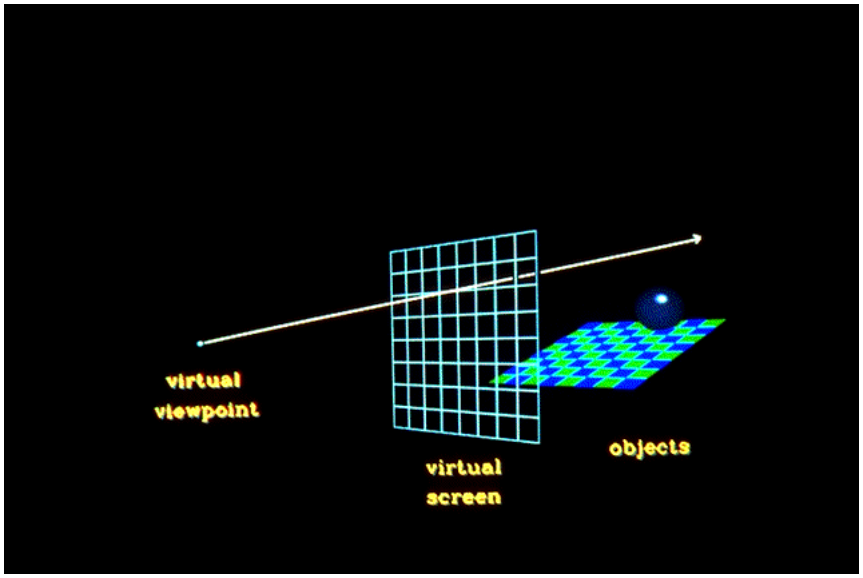


Image by Henrik Wann Jensen. He writes: One of my first ray tracing images (1990-1991). Rendered first time on an Amiga in HAM mode (the good old days).

Ray Tracing

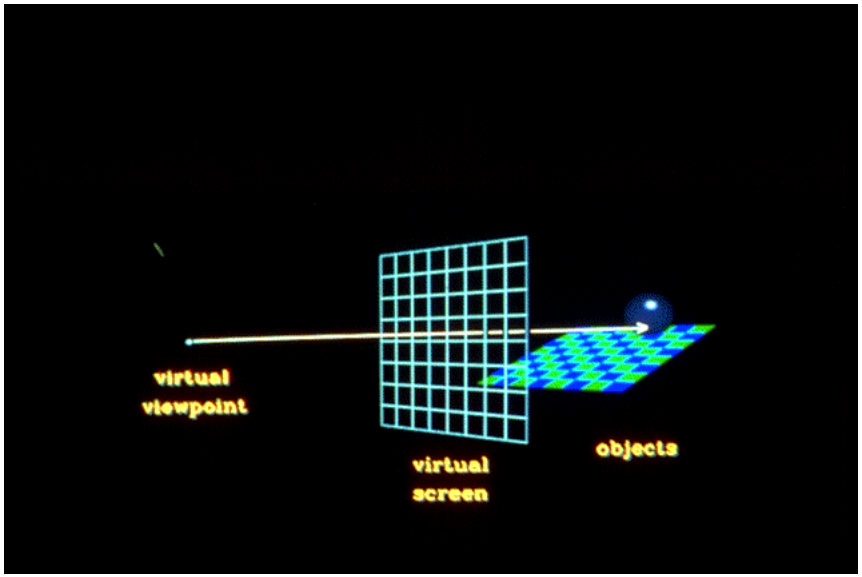
- Primary ray from viewpoint into the scene through pixel image (virtual screen).
- If all objects in the scene are missed, i.e. no intersection, set color to background.



Source Michael Sweeny, SIGGRAPH, 1991

Ray Tracing

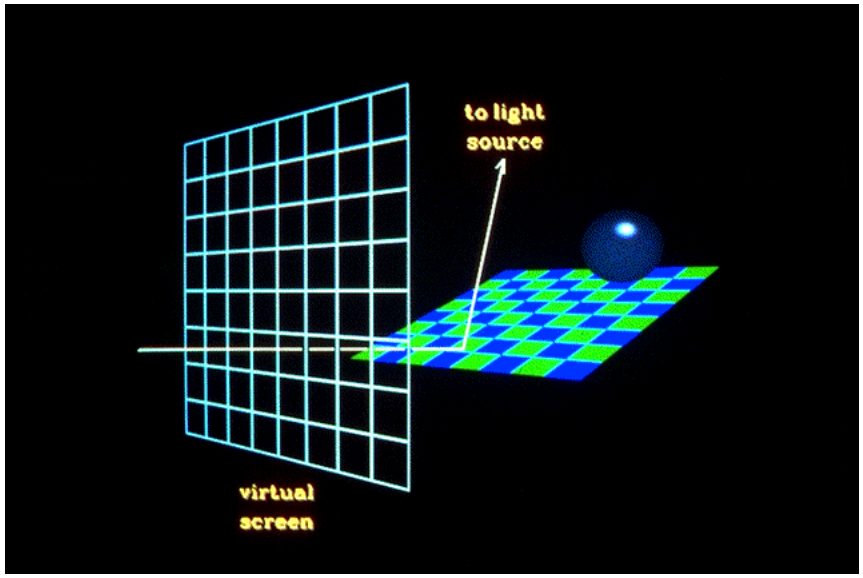
- Some rays will hit an object. Sent secondary rays to estimate pixel color.



Source Michael Sweeny, SIGGRAPH, 1991

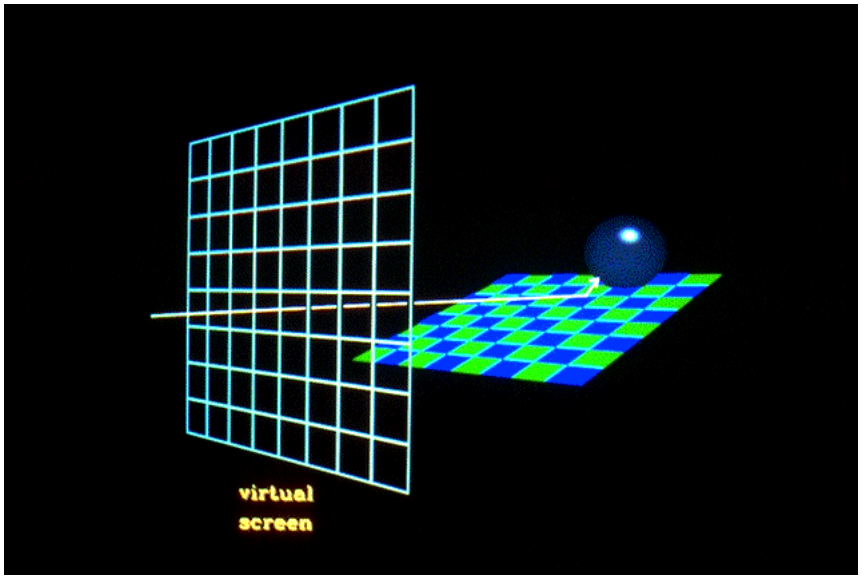
Ray Tracing

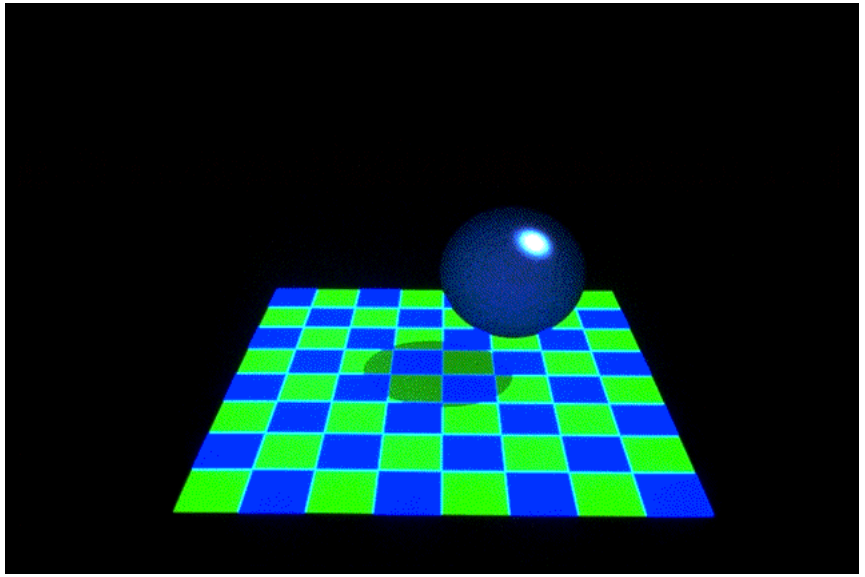
- Shadow ray
  - determine first hit
  - Shot shadow ray - secondary ray - towards light sources.
  - If shadow ray hits another object before it hits a light source, then intersection point is in shadow.
  - In case of Phong illumination only apply the ambient term.



Source Michael Sweeny, SIGGRAPH, 1991

Ray Tracing

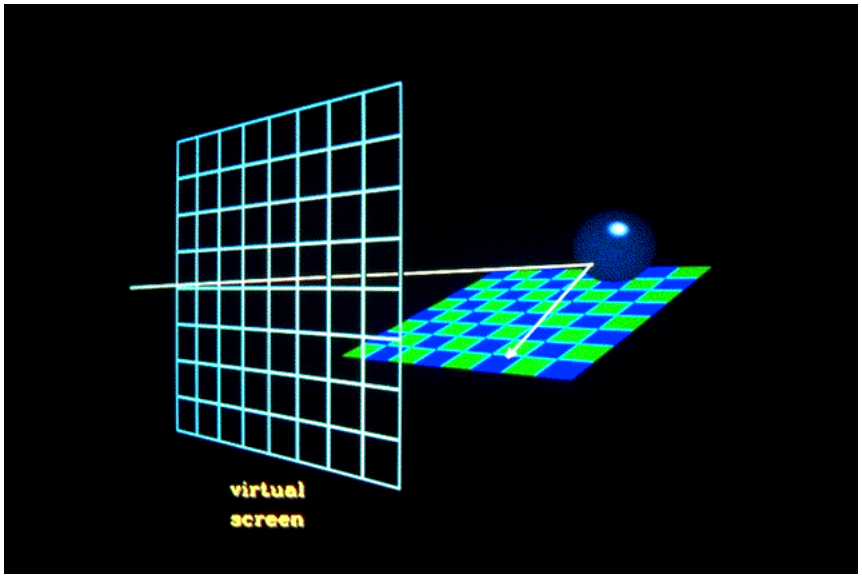




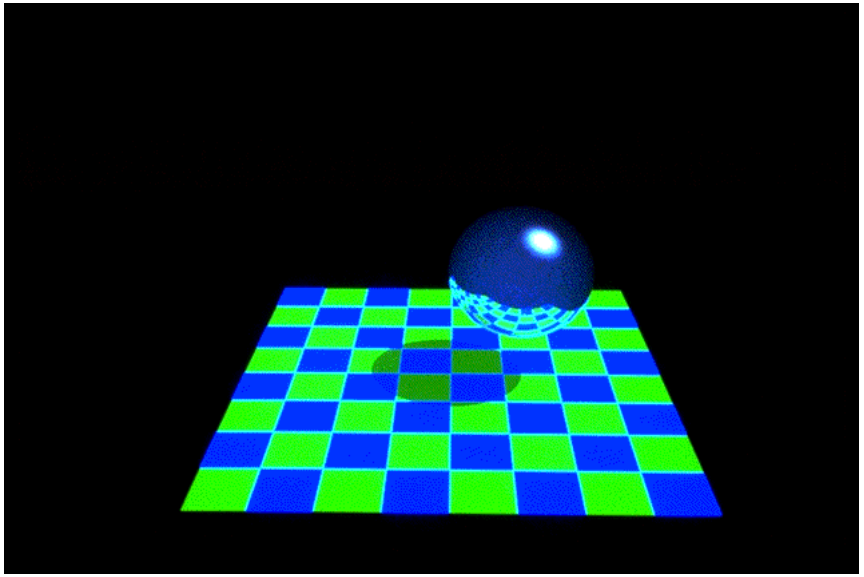
Source Michael Sweeny, SIGGRAPH, 1991

Ray Tracing

- Reflected ray
  - When ray hits an object a reflected ray is generated and tested against all scene objects. Angle of incidence equals angle of reflection.
  - If reflected ray hits an object, light intensity is estimated by evaluating a local illumination model. Result is carried back to the previous intersection.



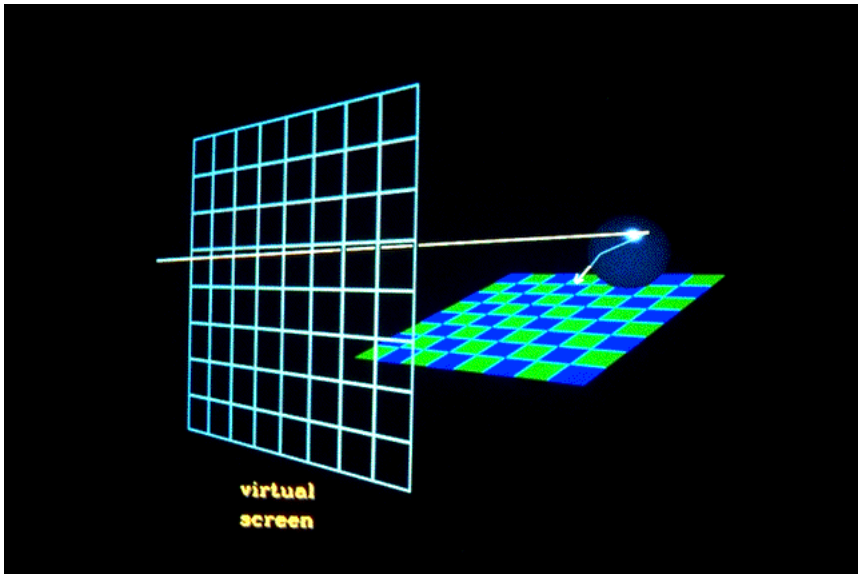
Source Michael Sweeny, SIGGRAPH, 1991



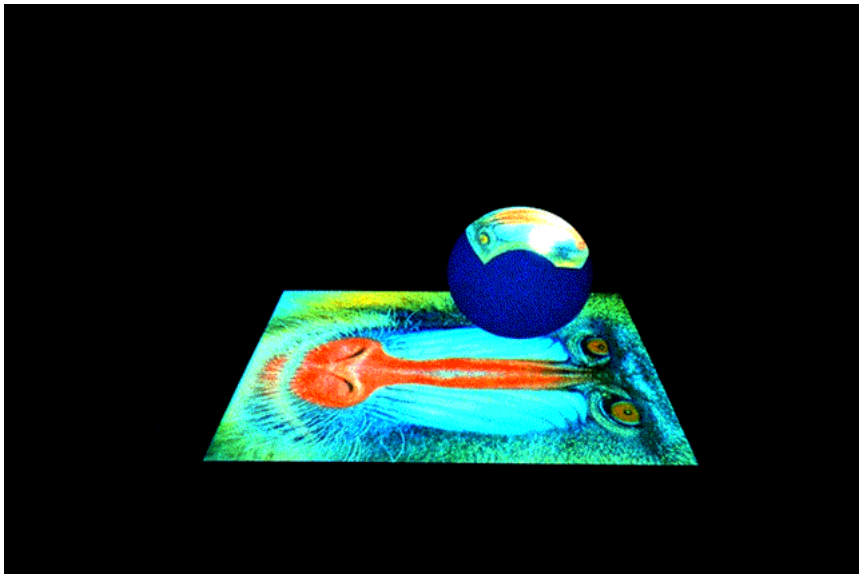
Source Michael Sweeny, SIGGRAPH, 1991

Ray Tracing

- Transmitted Ray
  - If intersected object is transparent, a transmitted ray is generated and tested against all scene objects.
  - If transmitted ray hits an object, proceed as in the case of reflected rays.



Source Michael Sweeny, SIGGRAPH, 1991



Source Michael Sweeny, SIGGRAPH, 1991

Basic Ray Tracing

- Basic Ray Tracing
  - Light hitting the surface point is composed
    - direct illumination by light source (! shadow rays)
    - incoming light from reflected ray
    - incoming light from transmitted/refracted ray
- Ray tracing is recursive!

Basic Ray Tracing

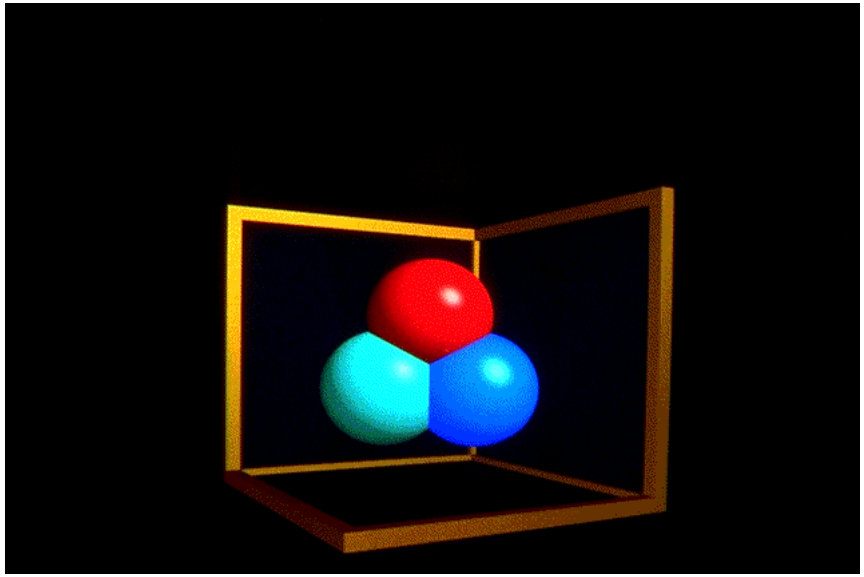
```
for each pixel p
    compute eye ray
    c = raytrace(ray,0)
    set pixel p to color c

raytrace(ray,depth)
    hit = intersect(scene,ray)
    c = black;
    if (hit.shader.isReflective() and depth < maxdepth)
        reflray = compute reflection ray
        c += raytrace(reflray,depth+1) * reflcolor
    if (hit.shader.isRefractive() and depth < maxdepth)
        refrray = compute refraction ray
        c += raytrace(refrray,depth+1) * refrcolor
    shadowRay = compute shadow ray
    if (intersect(scene,shadowRay)
         c += hit.shader.ambientColor
    else
         c += hit.shader.computePhongColor();
    return c;
```

Ray Tracing

- Reflected rays can generate other reflected rays that can generate other reflected rays, etc. (Recursion)

→ Layers of reflection

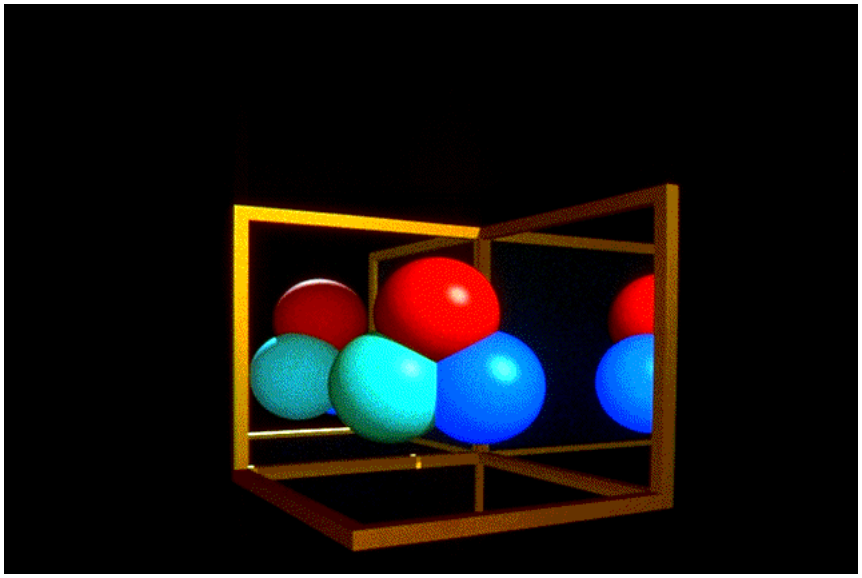
  - Scene traced with no reflection.



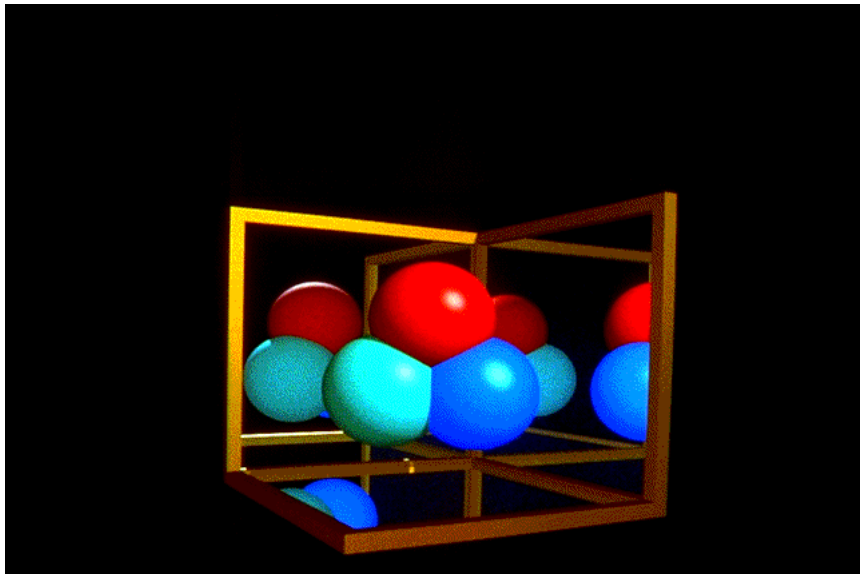
no reflection, maxdepth = 0

Source Image by Michael Sweeny, SIGGRAPH, 1991

Ray Tracing



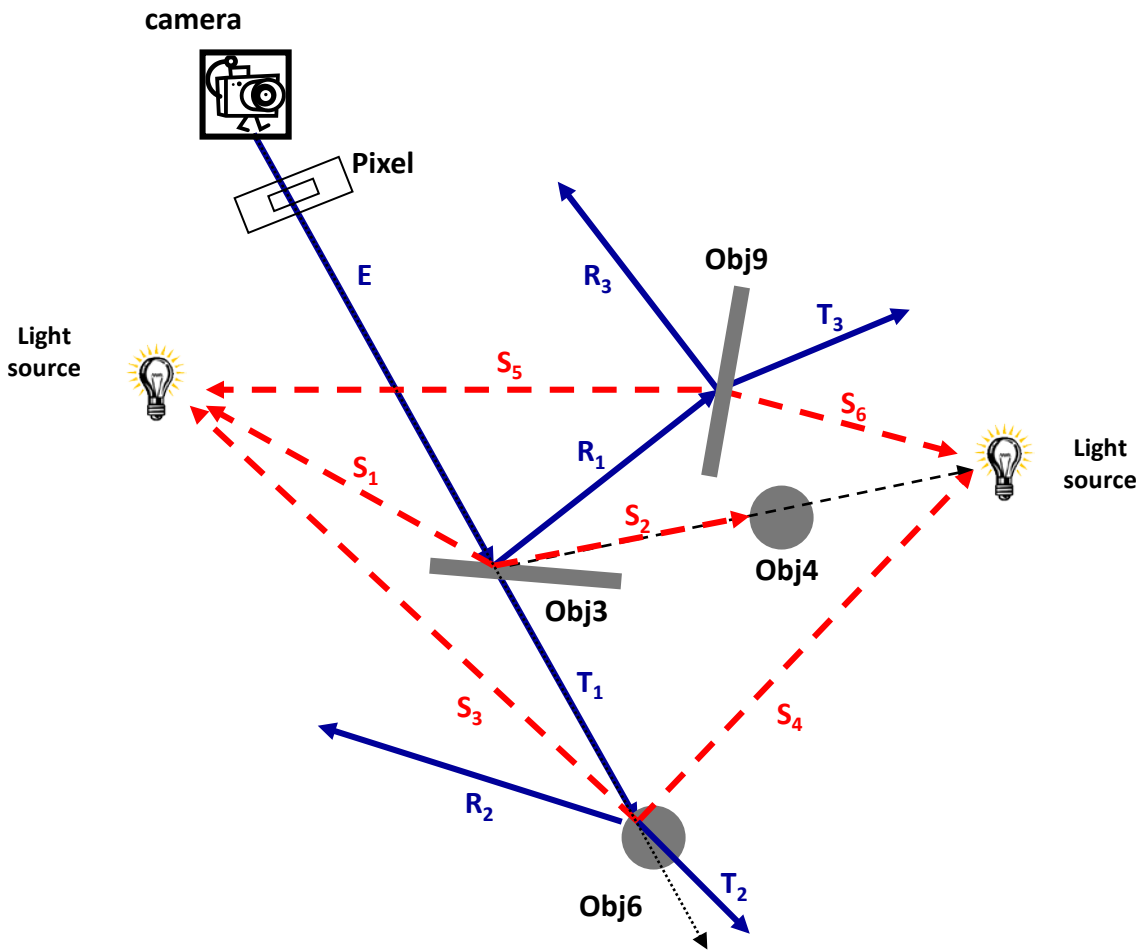
a single reflection, maxdepth =1



double reflection, maxdepth =2

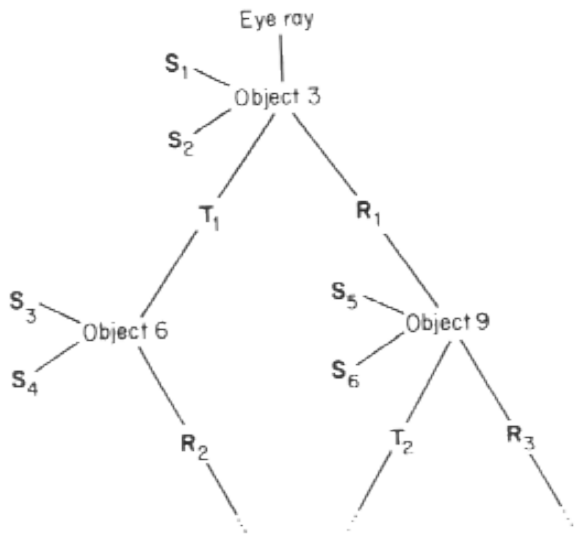
Source Michael Sweeny, SIGGRAPH, 1991

Basic Ray Tracing

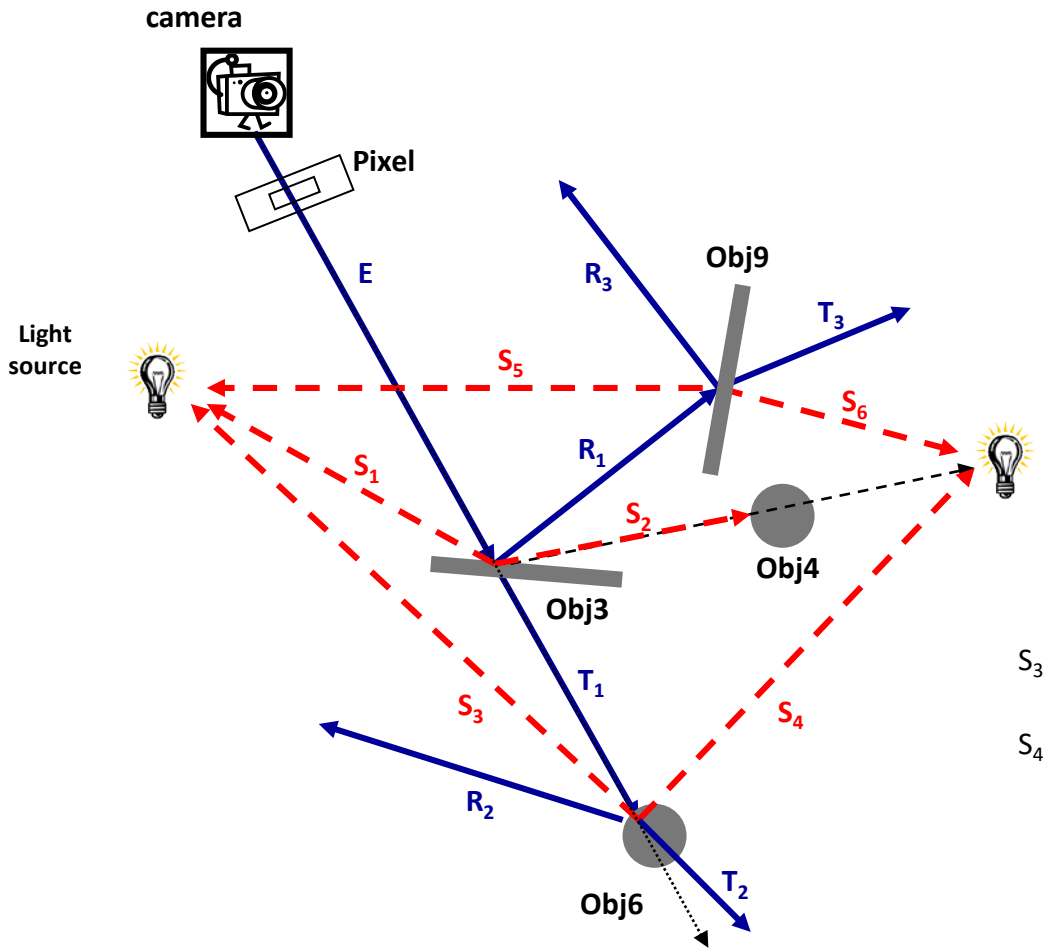


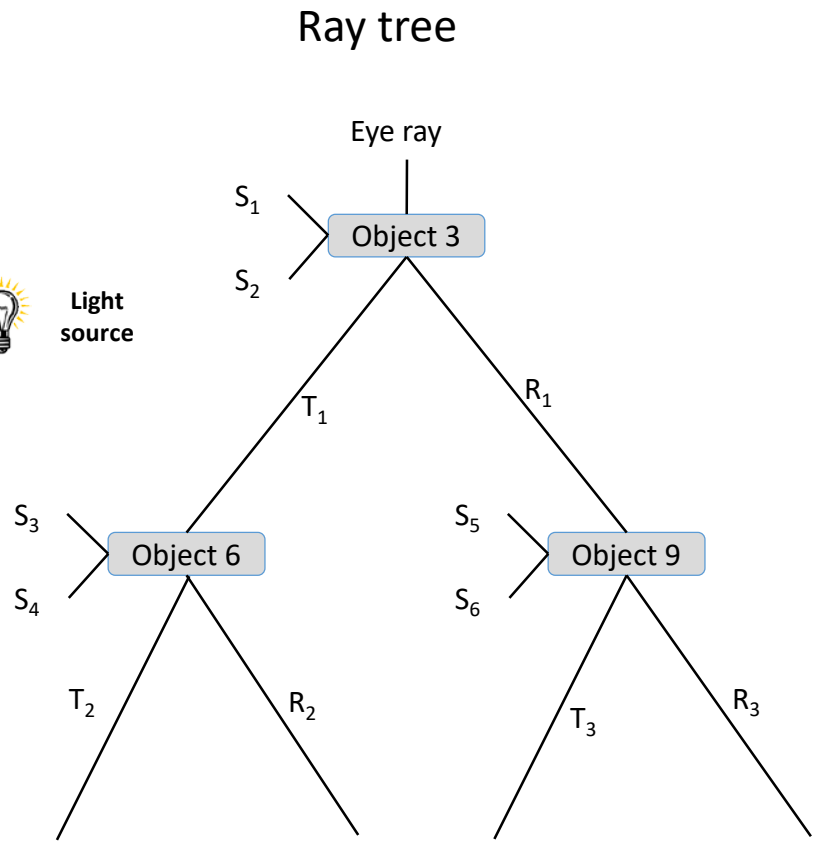
Basic Ray Tracing

- Organization of rays in a Ray Tree
- Recursive construction
- Recursive illumination calculation per node



Basic Ray Tracing



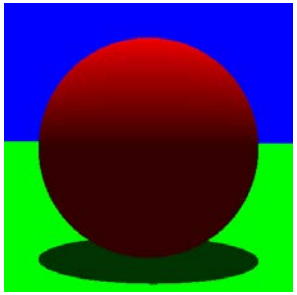


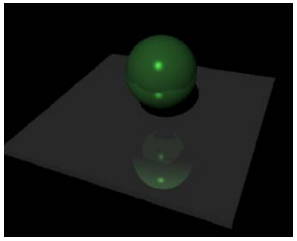
Basic Ray Tracing

- Performance issues
  - Millions of rays are needed for high quality images
  - two major technical components of a ray tracing software package must be optimized:
    - Ray-object intersections (in particular ray-triangle)
    - Data structures to reduce the number of intersection tests, e.g. bounding volumes
  - Both issues are considered later with some detail

Basic Ray Tracing

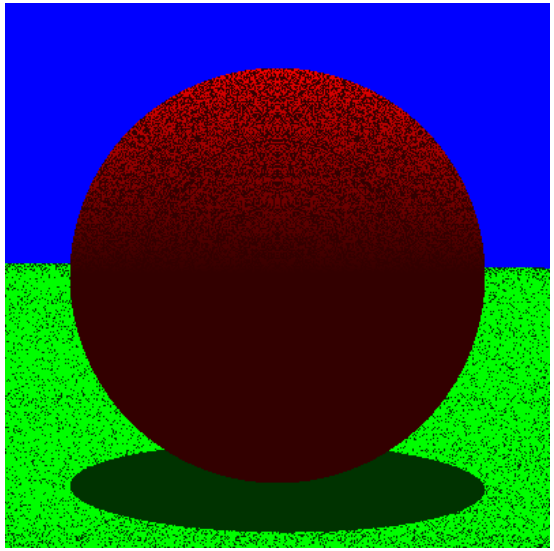
- Discussion:
  - Shadows
  - Reflection
  - Transmission / Refraction



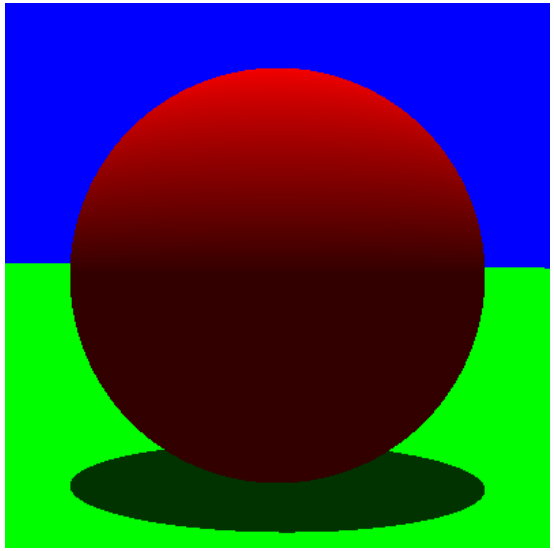


Shadows

- Shadow rays:
  - only intersection between surface point and light source are of interest
  - only search for intersections with $t \in [t_{min}, t_{max}]$
  - $t_{min} = 0$? → better not
- Self-Shadowing
  - Intersection of shadow ray with object itself.
  - Floating point precision problem.
  - Solution: add an epsilon



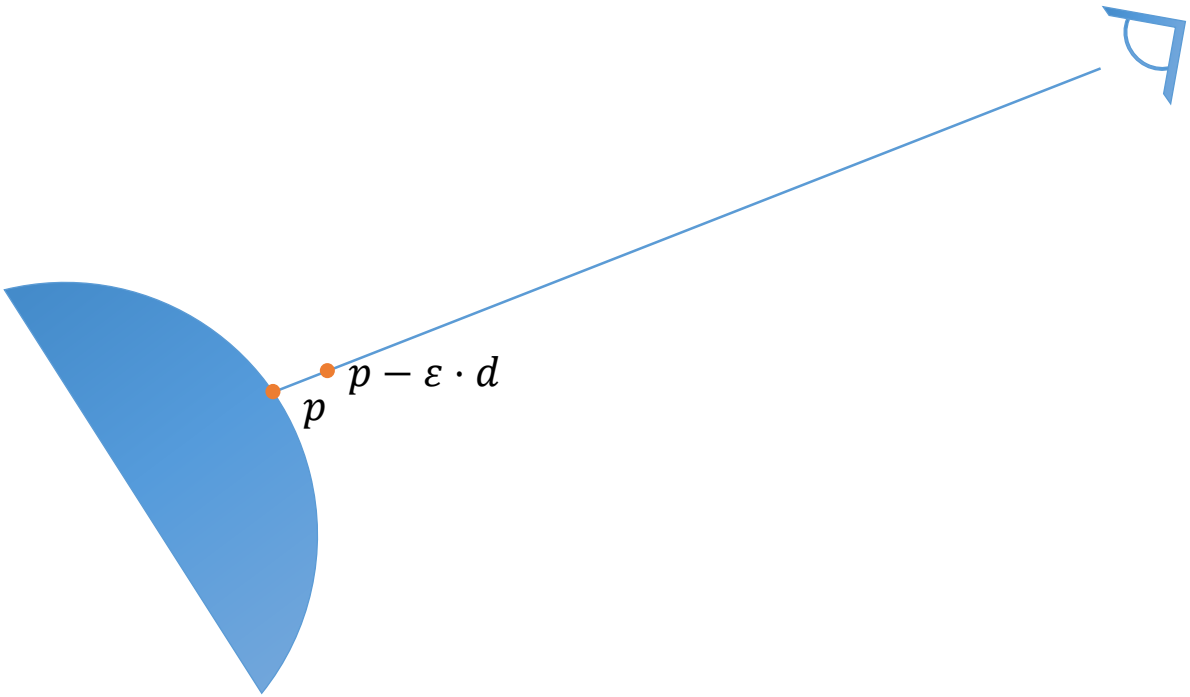
Self-shadowing



Shadow ray with epsilon

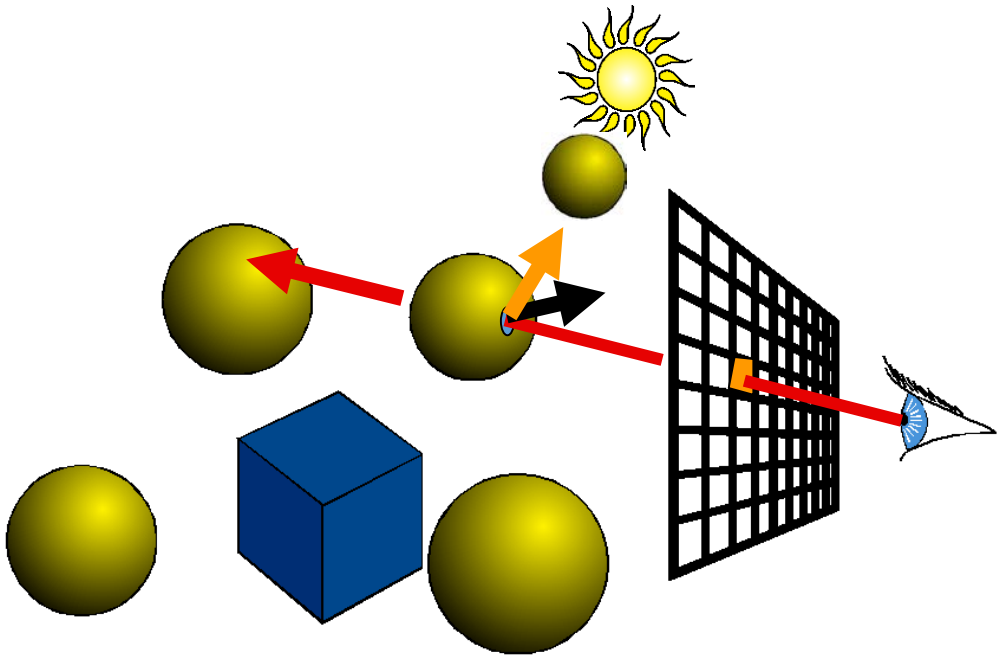
Shadows

- Avoiding self-shadowing
  - Intersection point $p = e + t \cdot d$
  - Use instead $p - \varepsilon \cdot d$
- or
  - use $t_{min} = \epsilon$



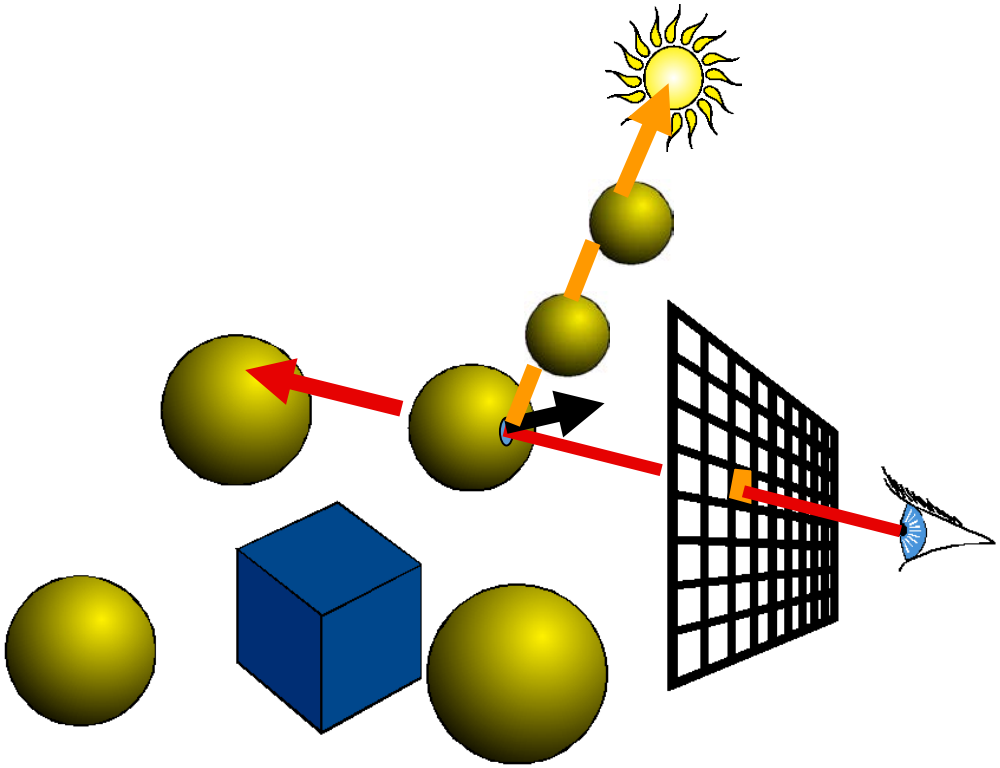
Shadow optimization

- Shadow rays are special
- How can we accelerate our code?



Shadow optimization

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



Shadows

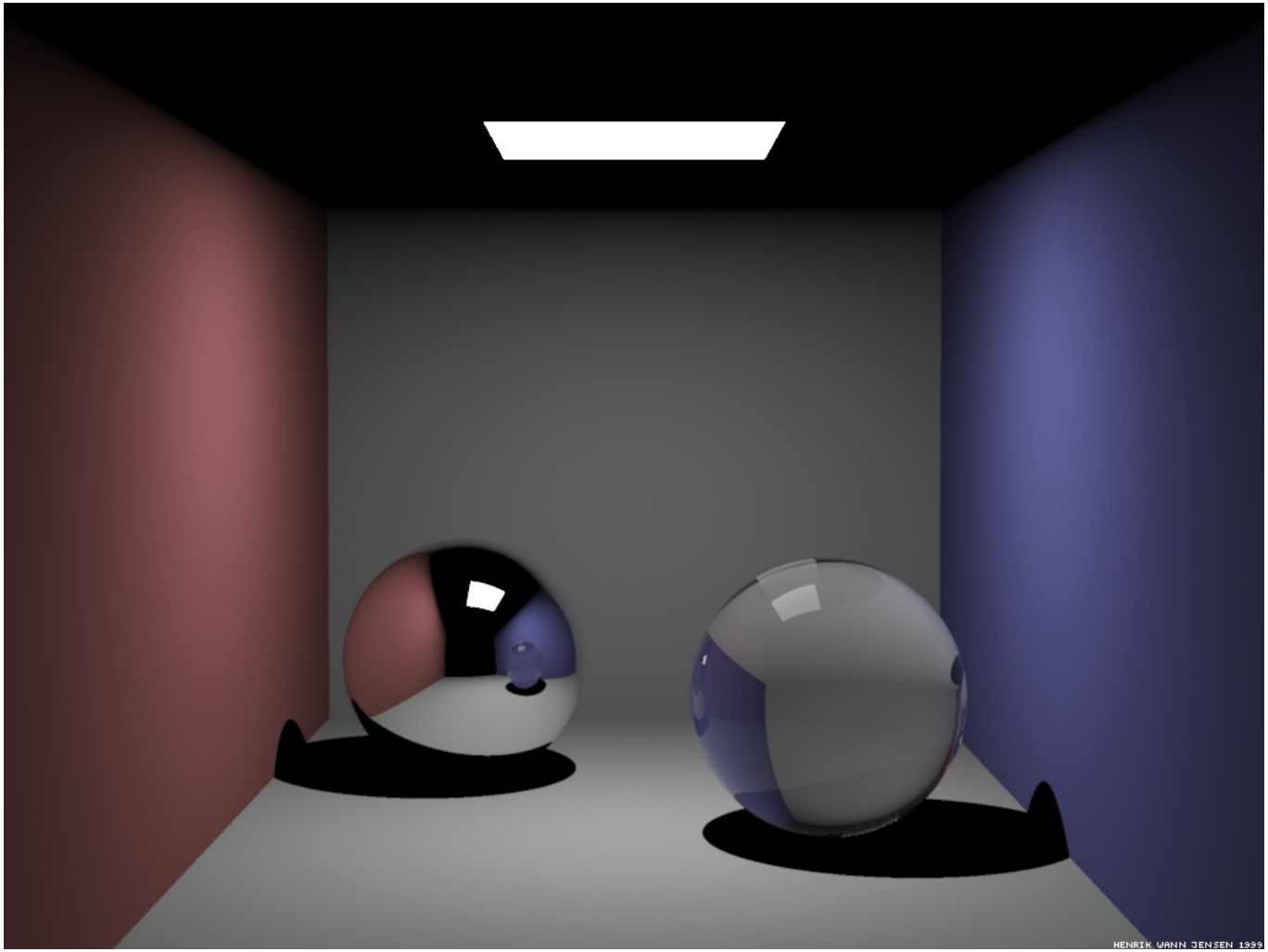
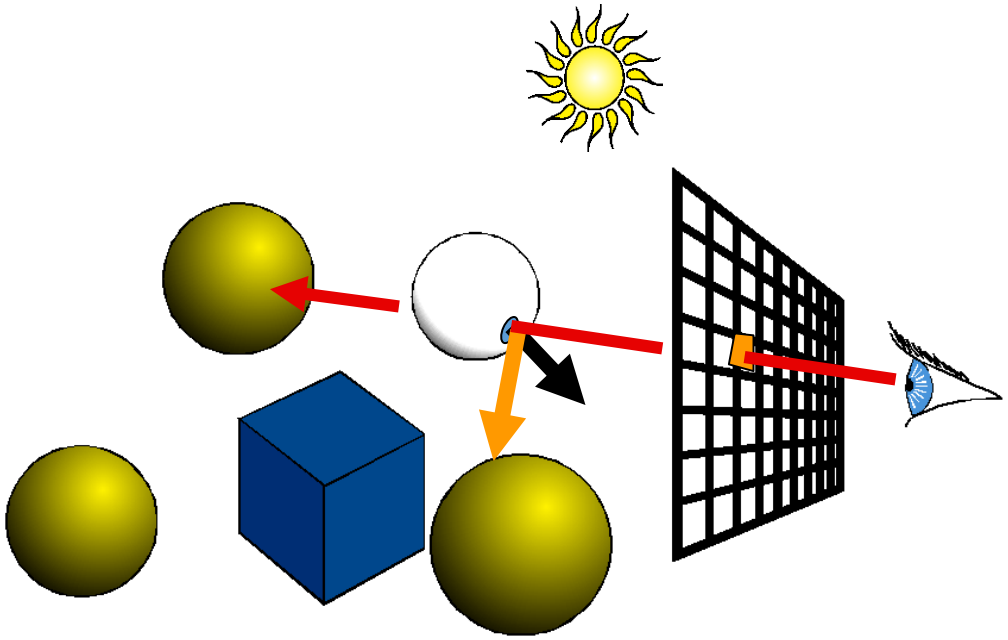
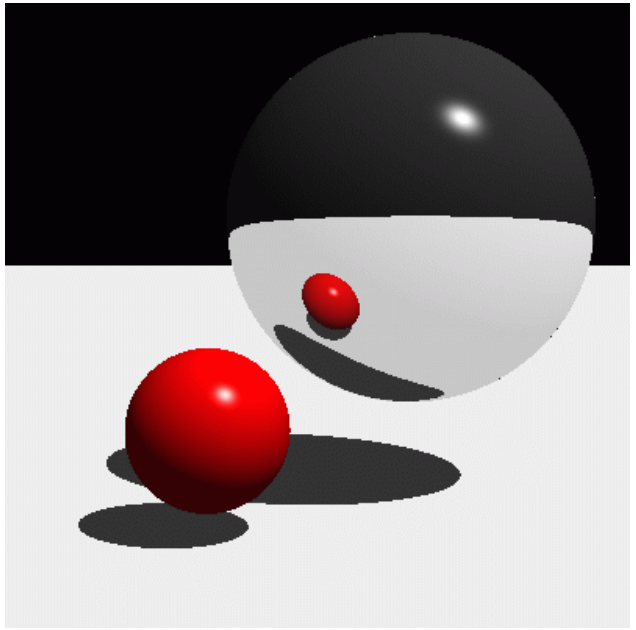


Image by Henrik Wann Jensen

Reflection

- Reflection
  - Compute mirror contribution



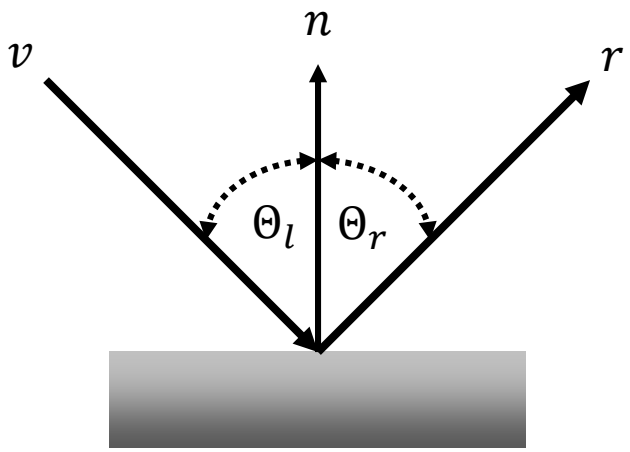


Reflection

- Reflection
  - Compute mirror contribution
  - For every hit point x cast reflection ray in direction symmetric w.r.t. normal, angle of incidence equals angle of reflection.
  - Multiply by reflection coefficient (color)
    - Pixel color = lighting at x + reflection coeff. × light of reflected ray

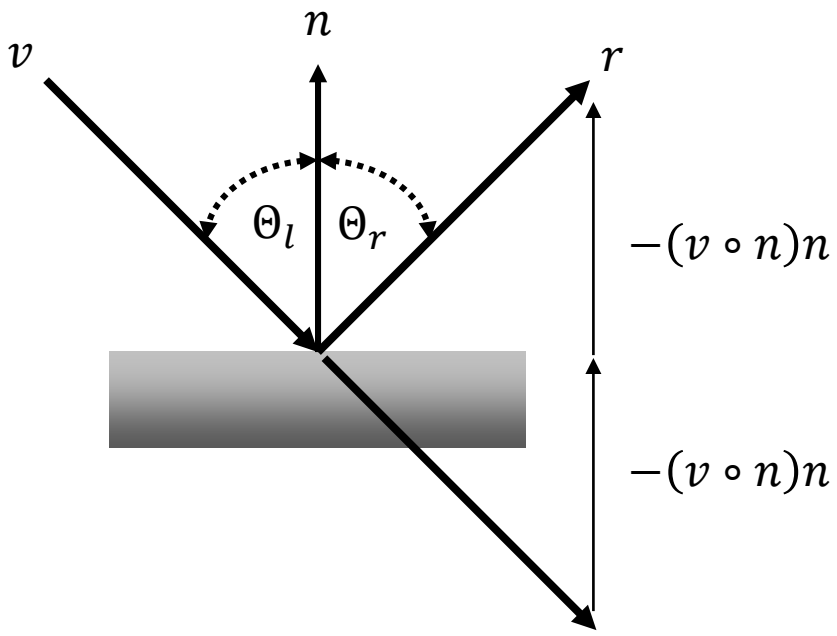
Reflection

- Reflection angle = view angle



Reflection

- Reflection angle = view angle



$$r = v - 2(v \circ n)n$$

Reflection

- Traditional ray tracing
  - Constant coefficient reflection color
  - compute light intensity for each color channel independently
- High quality rendering requires more realistic models
  - non-linear approximations of the reflectance functions
  - e.g. Fresnel reflection/refraction equations

Reflection

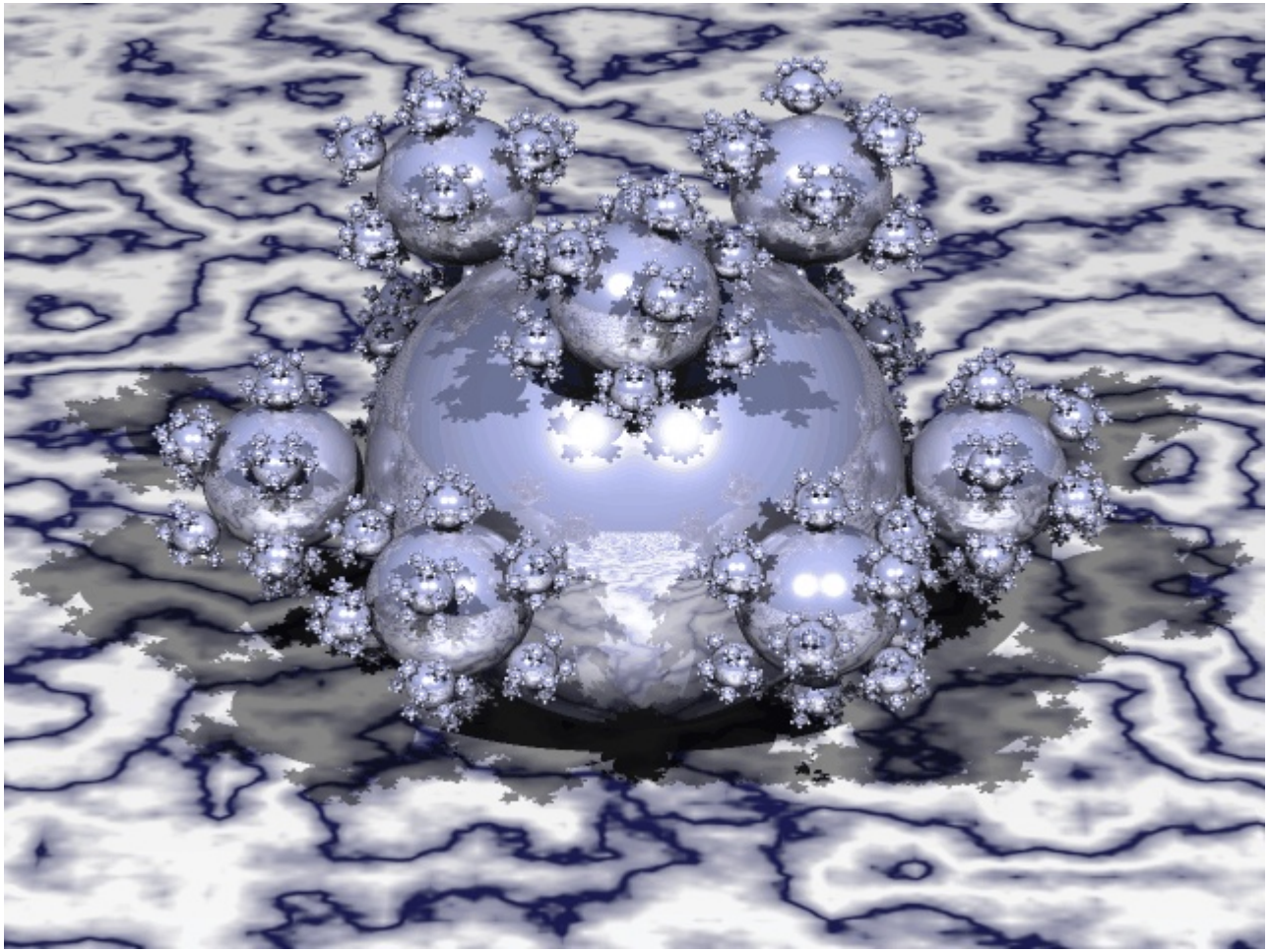
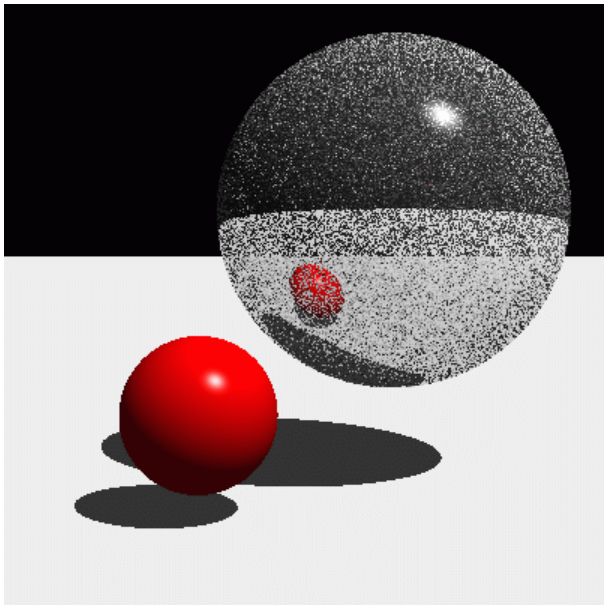


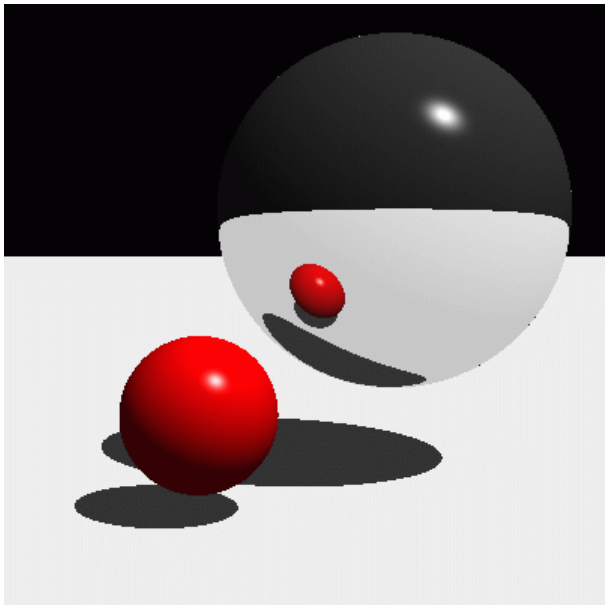
Image by Henrik Wann Jensen, 1992.
It is a procedural (fractal object). Here approximated using 500.000 spheres.

Reflection

- Don’t forget to add epsilon to the ray



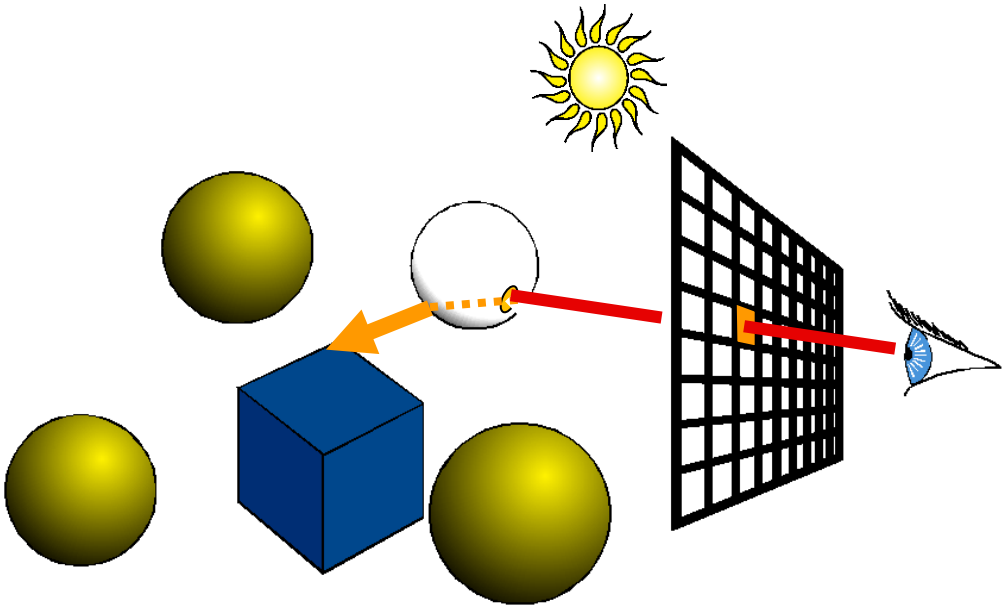
without epsilon



with epsilon

Refraction

- Transparency
  - Compute transmitted contribution

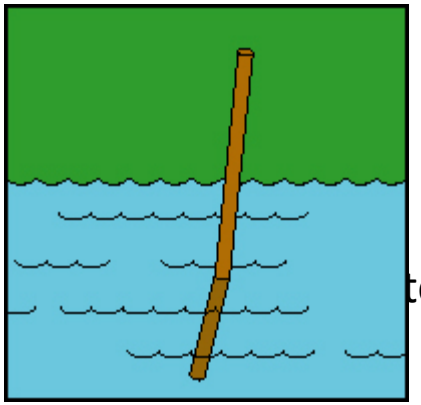


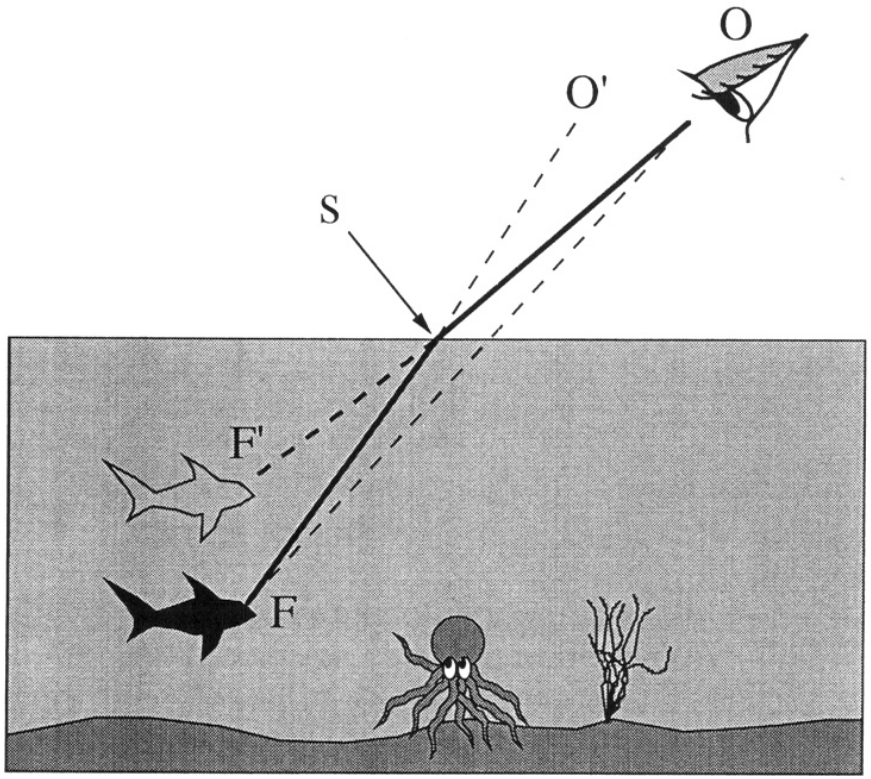
Refraction

- Transparency
  - For every hit point x, cast a ray in direction of refraction
  - Pixel color = lighting at x
    + reflection coeff. × light of reflected ray
    + transparency coeff. × light of refracted ray

Refraction

- Refraction
  - Half submerged straight stick in water
  - Light bends at the point ters the water



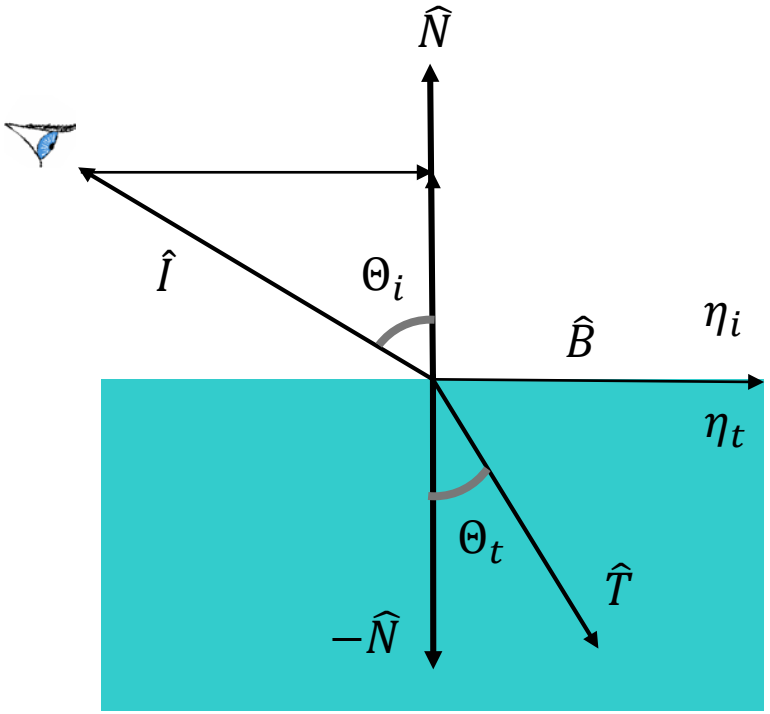


Refraction

- Refraction
  - light passing from one transparent medium to another changes speed and bends
  - Effect depends on
    - refractive index of mediums
    - Angle between light ray and normal

Refraction

- Snell-Descartes Law
  - two media with different refraction indices η_i and η_t
  - $\frac{\sin \Theta_i}{\sin \Theta_t} = \frac{\eta_t}{\eta_i} = \eta_r$



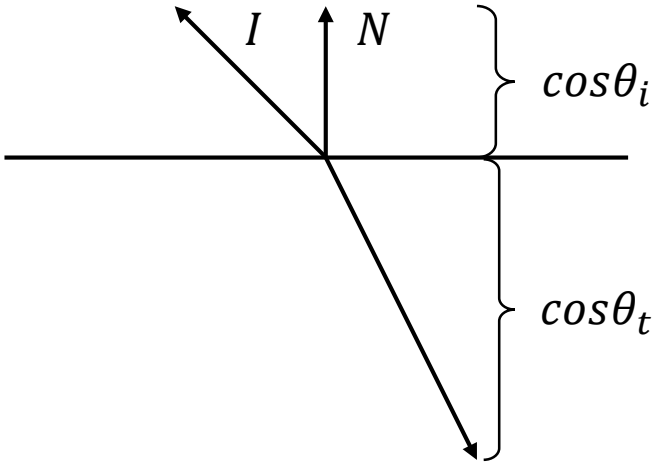
Refraction

- Remark on Snell-Descartes law
  - simplification: use cosines

$$\frac{\sin\Theta_i}{\sin\Theta_t} = \frac{\eta_t}{\eta_i} \quad \rightarrow \quad \eta_i \sin\Theta_i = \eta_t \sin\Theta_t$$

$$\cos^2\Theta_t = 1 - \frac{\eta_i^2(1-\cos^2\Theta_i)}{\eta_t^2}$$

  - The vectors B and T lies in the plane spanned by $\{N, I\}$.



Refraction

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)

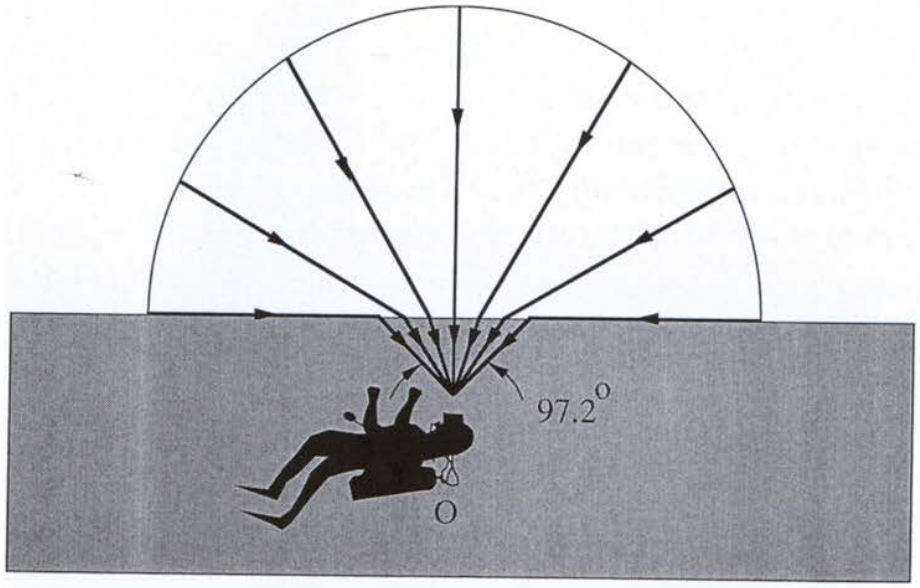
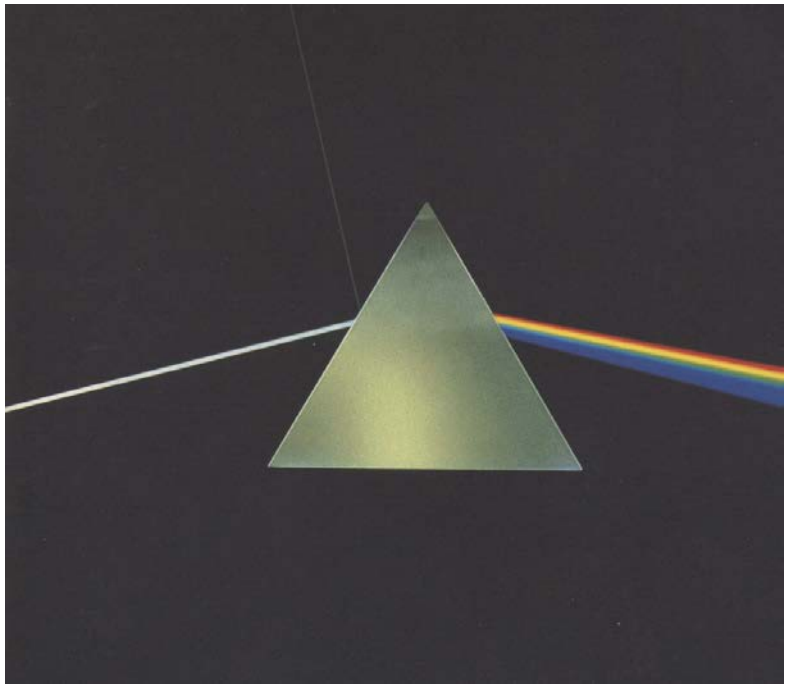


Fig. 3.7B The optical manhole. Light from the horizon (angle of incidence = 90°) is refracted downward at an angle of 48.6°. This compresses the sky into a circle with a diameter of 97.2° instead of its usual 180°.

From "Color and Light in Nature" by Lynch and Livingstone

Refraction

- Refraction is wavelength-dependent
  - Newton's experiment
  - Usually ignored in graphics



Reflection + Refraction

- Fresnel equations
  - light moves from one medium with refractive index n_1 to a medium with refractive index n_2.
  - part of the energy is **reflected** and part is **transmitted**.
  - the fraction of the power reflected is given by the reflectance R.
  - the fraction of the power transmitted is given by the transmittance T.

Reflection + Refraction

- Fresnel equations
  - The constants R and T depend on the polarization of light.
  - The angles of the incident and refracted rays with the normal of the interface are given by the Snell-Descartes law.

Reflection + Refraction

- s-Polarization: electric field perpendicular to plane

$$R_s = \frac{\sin^2(\Theta_t - \Theta_i)}{\sin^2(\Theta_t + \Theta_i)} = \left(\frac{n_1 \cos\Theta_i - n_2 \cos\Theta_t}{n_1 \cos\Theta_i + n_2 \cos\Theta_t}\right)^2$$

- p-polarization: electric field parallel to plane

$$R_s = \frac{\tan^2(\Theta_t - \Theta_i)}{\tan^2(\Theta_t + \Theta_i)} = \left(\frac{n_1 \cos\Theta_t - n_2 \cos\Theta_i}{n_1 \cos\Theta_t + n_2 \cos\Theta_i}\right)^2$$

- un-polarized light

$$R = \frac{R_s + R_p}{2}$$

Reflection + Refraction

- Transmission coefficients

$$T_s = 1 - R, \qquad T_p = 1 - R_p, \qquad T = 1 - R$$

- If light is normal incident

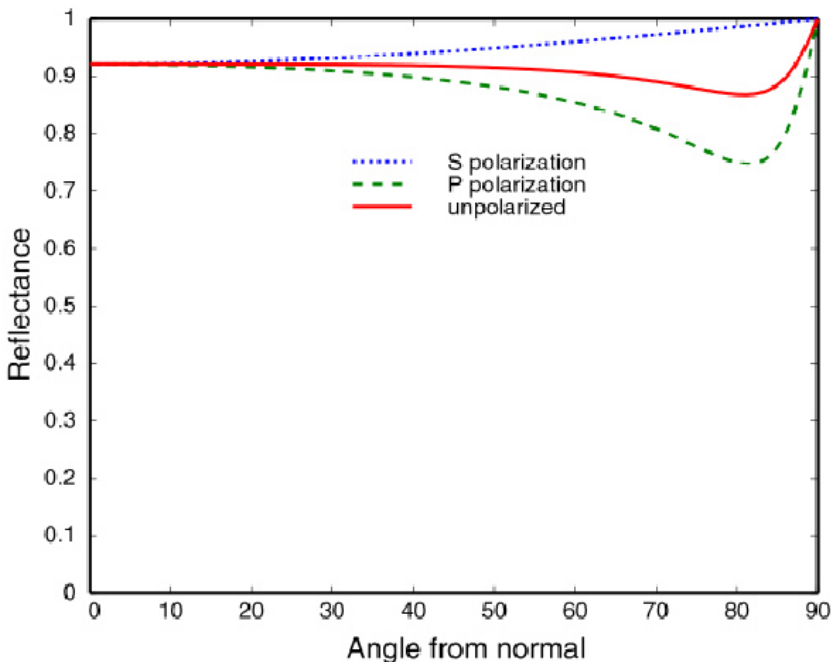
$$R_0 = R_s = R_p = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2$$

$$T_0 = T_s = T_p = 1 - R = \frac{4n_1 n_2}{(n_1 + n_2)^2}$$

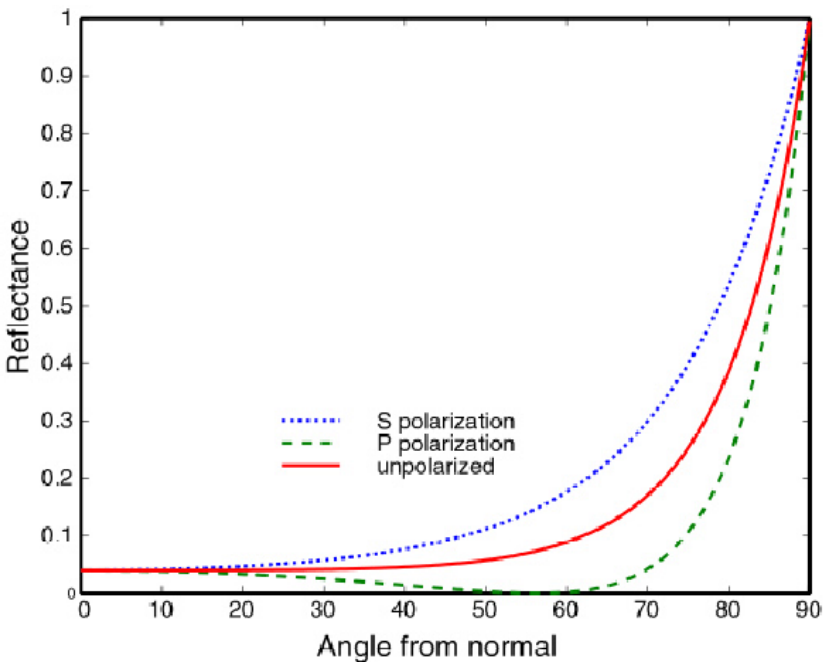
- Schlick approximation (Schlick, 1994)

$$R(\Theta_i) = R_0 + (1 - R_0)(1 - \cos\Theta_i)^5$$

Reflection + Refraction



metal



Dielectric (glass)

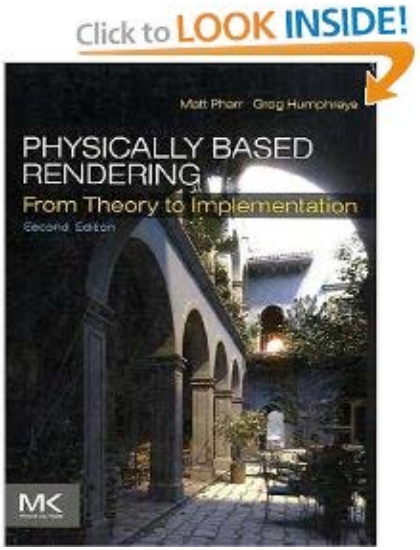
Reflection

- Increasing specular reflection for increasing grazing angles

Non-Linear Approximation of Reflectance Functions
E.P.F. Lafortune et al., SIGGRAPH 1997

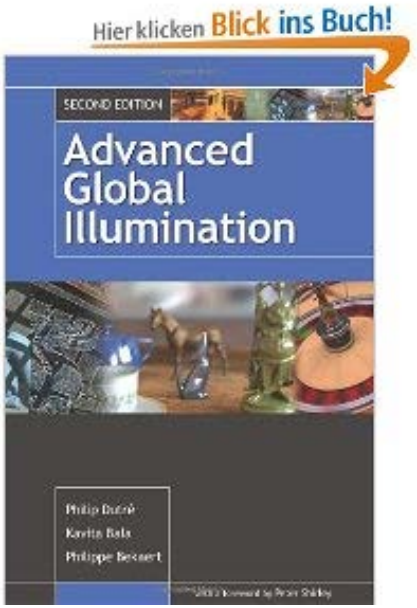
Additional information

- Book:
  - PHYSICALLY BASED RENDERING
  - MATT PHARR • GREG HUMPHREYS
  - Morgan Kaufmann
  - http://www.pbrt.org/



Additional Information

- Book:
  - Advanced Global Illumination
  - Philip Dutre, Kavita Bala, Philippe Bekaert



Additional information

- Literature:
  - Glassner, Andrew S.: An Introduction to Ray-Tracing. Morgan Kaufmann San Diego: Academic, 1989
- Free software: POVRay, Blender, …

  website: www.povray.org, www.blender.org

Images POV-Ray

»Pebble" © Jonathan Hunt (2008)

"Still with Bolts" © Jaime Vives Piqueres (2002)