Off-line rendering and global illumination

Performance demanding methods that give better lighting

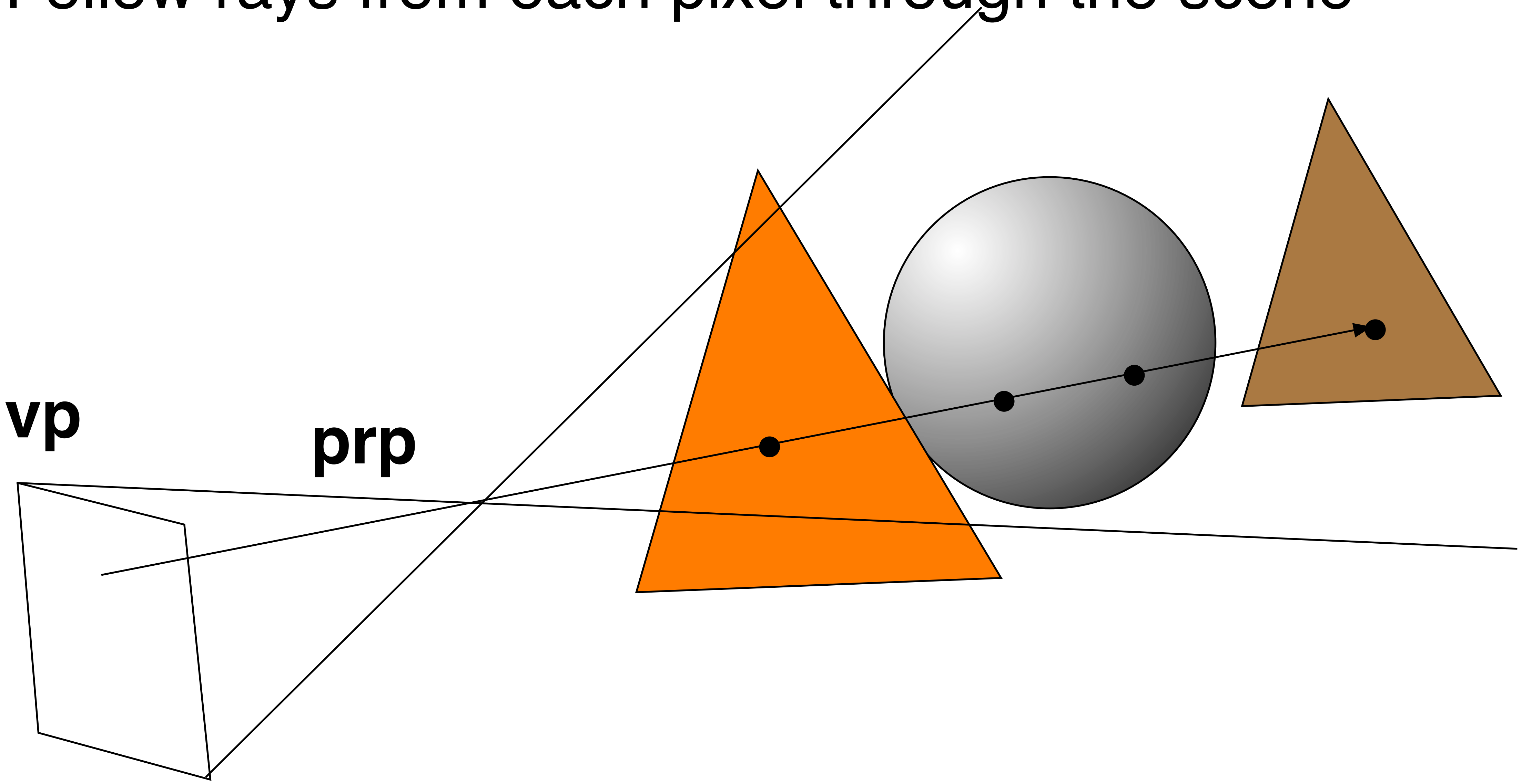
Ray-casting

Radiosity

Photon mapping

Ray-casting

Follow rays from each pixel through the scene



Full 3D raycasting

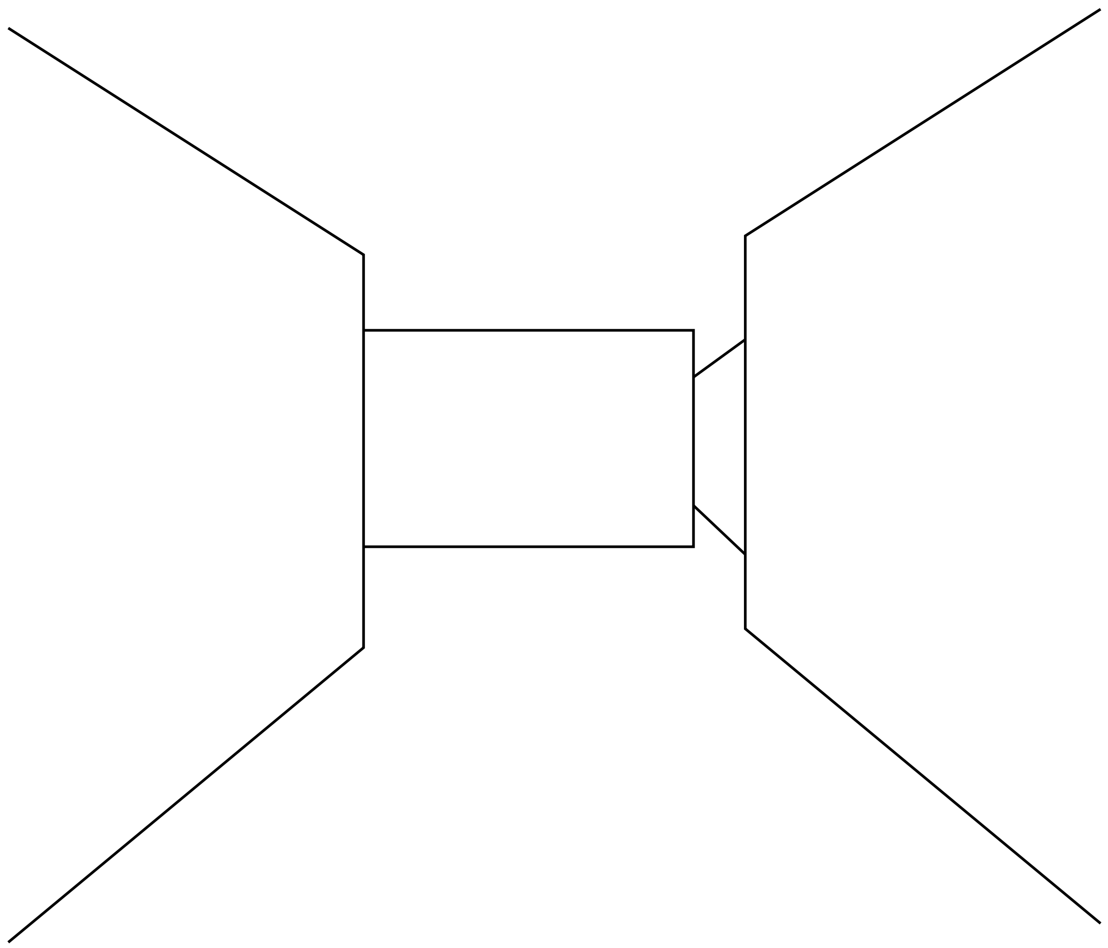
for every pixel (x,y) in the image

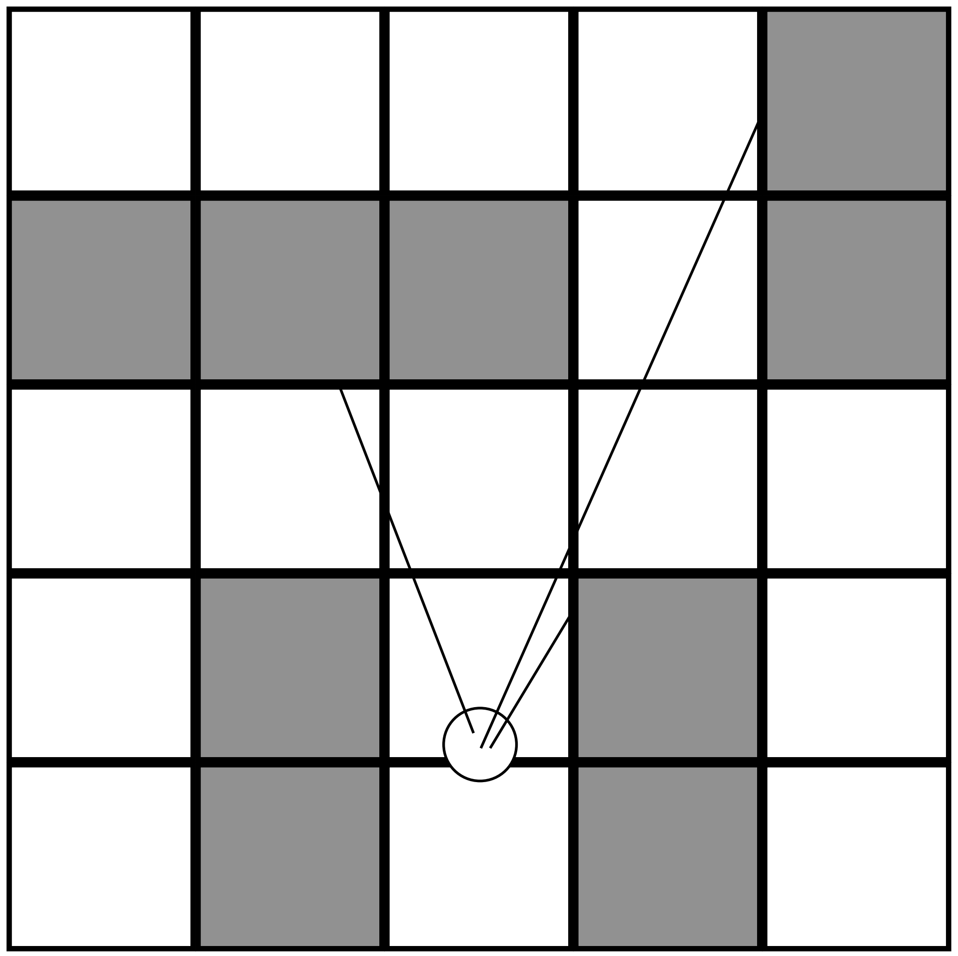
calculate a ray from the pixel through the camera (cop) and through the scene

calculate intersecions with all objects in the scene

the pixel value is calculated from the closest intersection found

Raycasting in 2D grid ("Ray-marching")





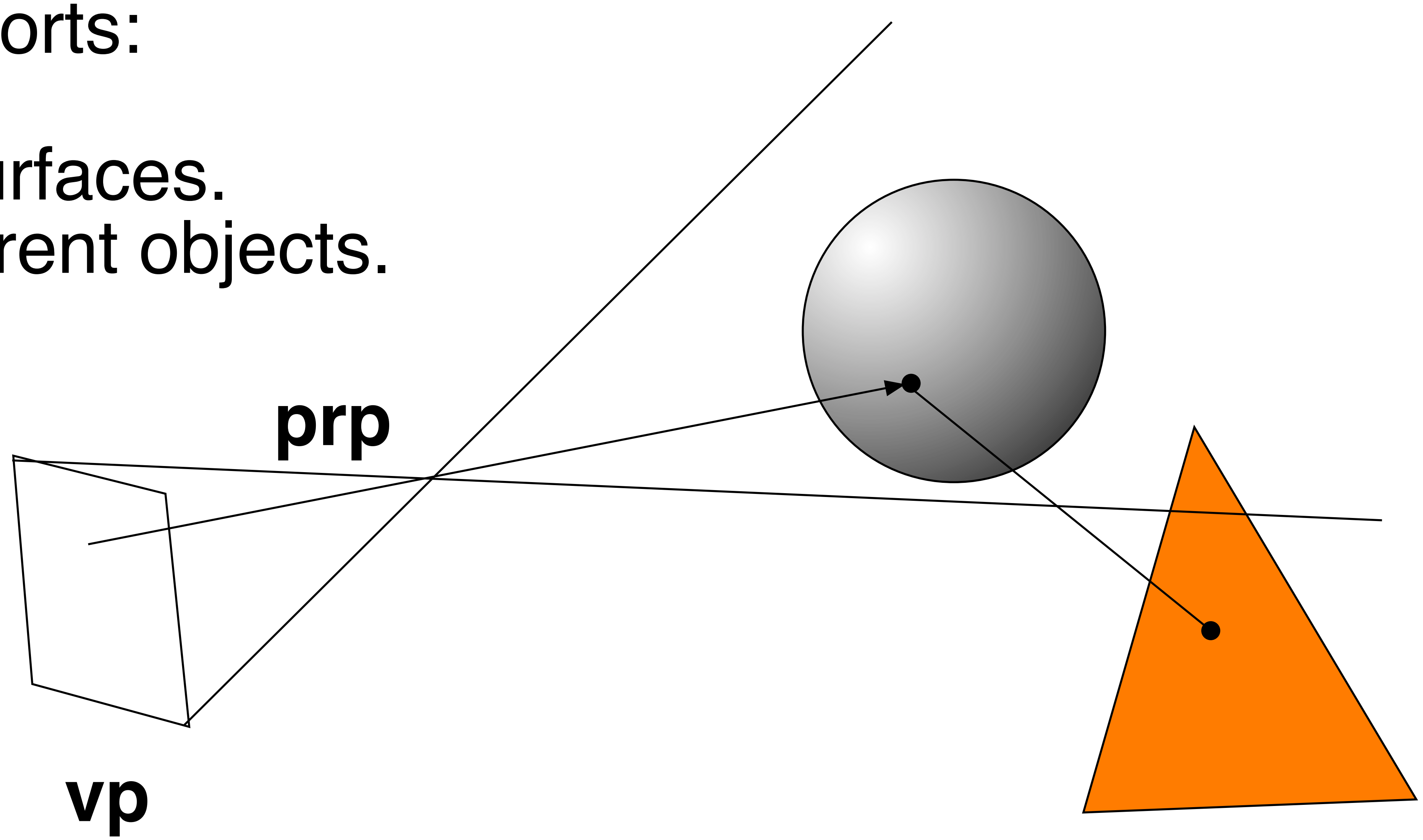
Ray-tracing

The classic method for rendering realistic images of shiny and transparent objects.

Ray-tracing

From some surfaces, follow rays to the next surface.
This supports:

- Shiny surfaces.
- Transparent objects.



At the intersection

Three things can happen when a ray intersects an object

1) Non-mirroring reflection

2) Reflection

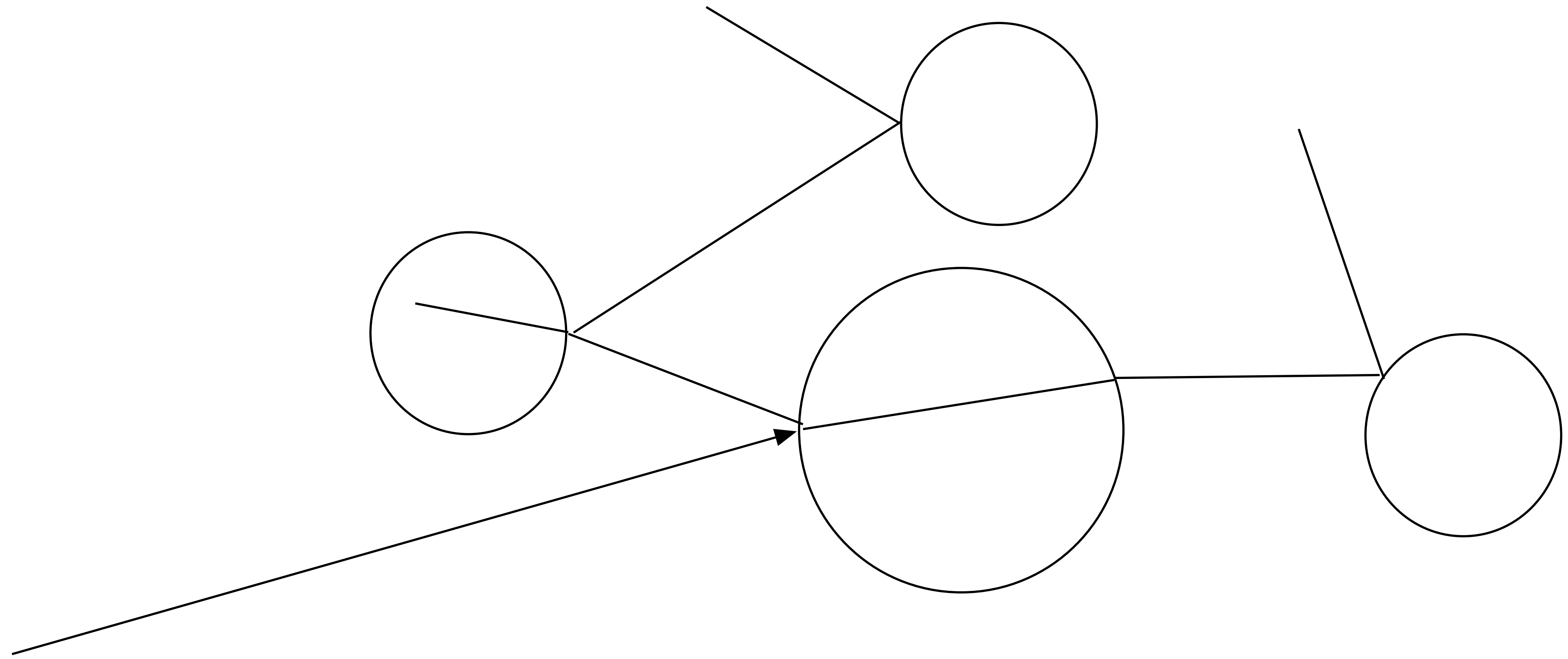
3) Refraction

Non-mirroring reflection

Apply the three-component light model

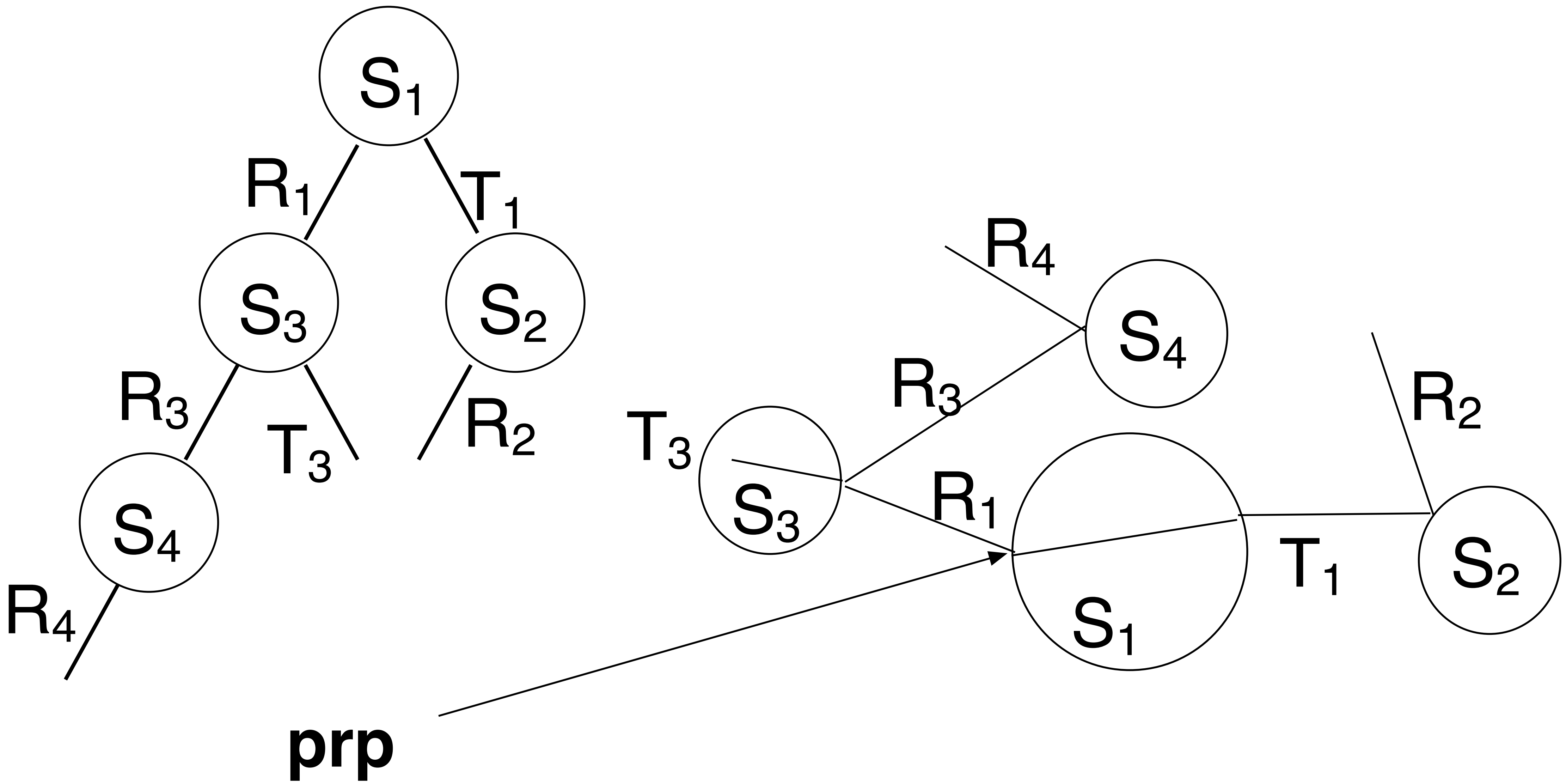
Ambient light
Diffuse reflection
Specular reflection

Reflection and refraction

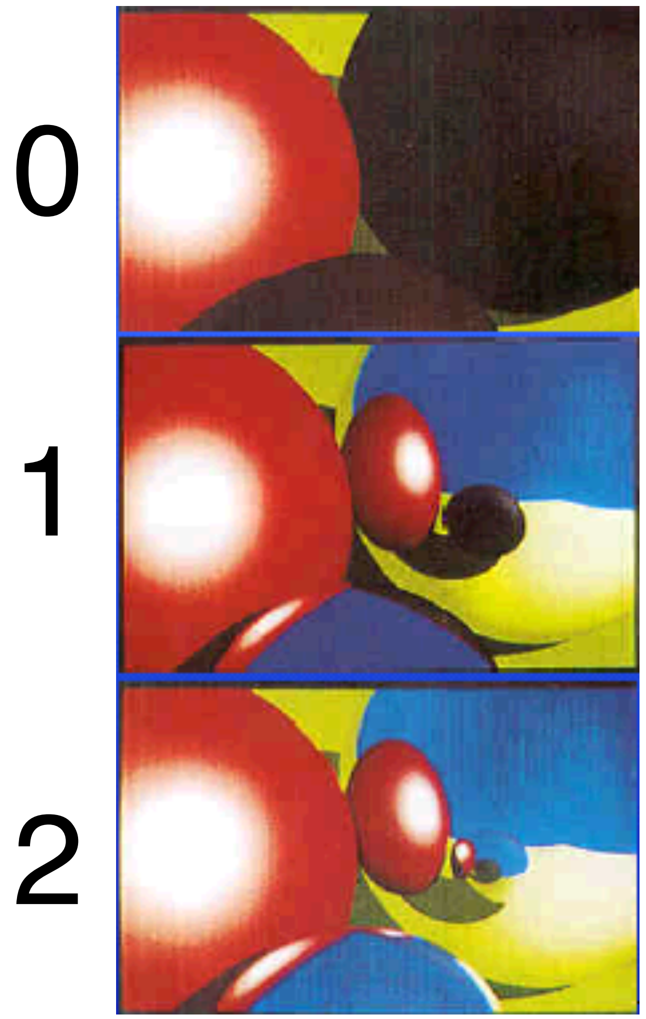


Recursive process!

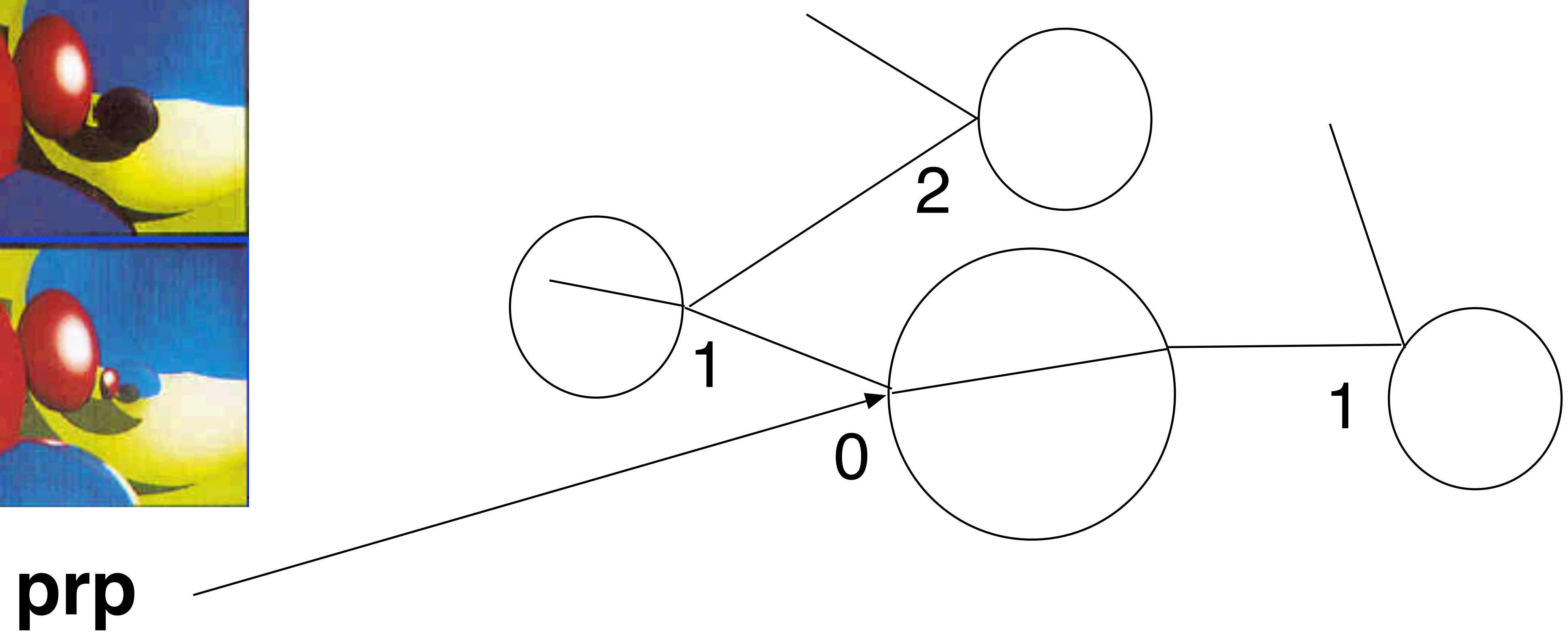
The ray-tracing tree



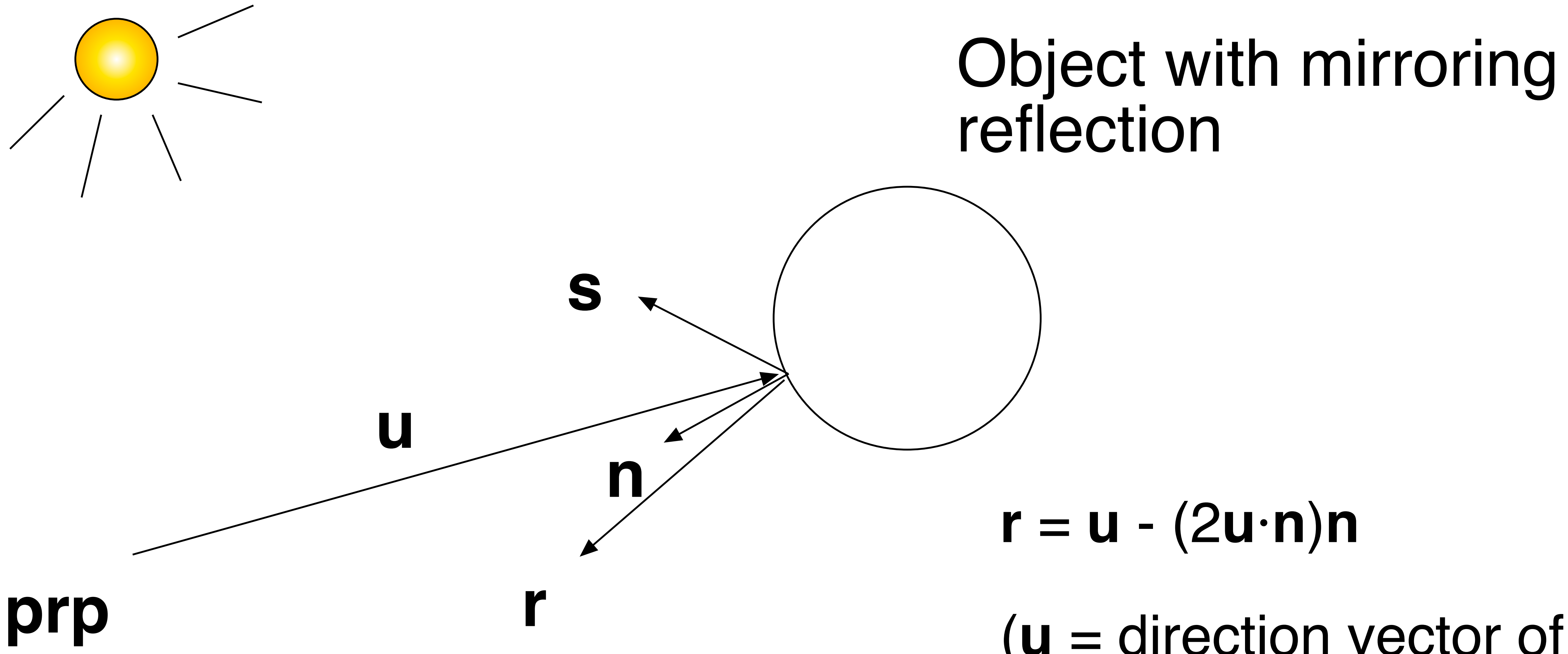
The maximum depth



How deep can the tree get?
How many reflections and refractions are allowed?



Reflections



Look for more contributions here

r = **u** - (2**u**·**n**)**n**

(**u** = direction vector of the incoming ray)

Refractions

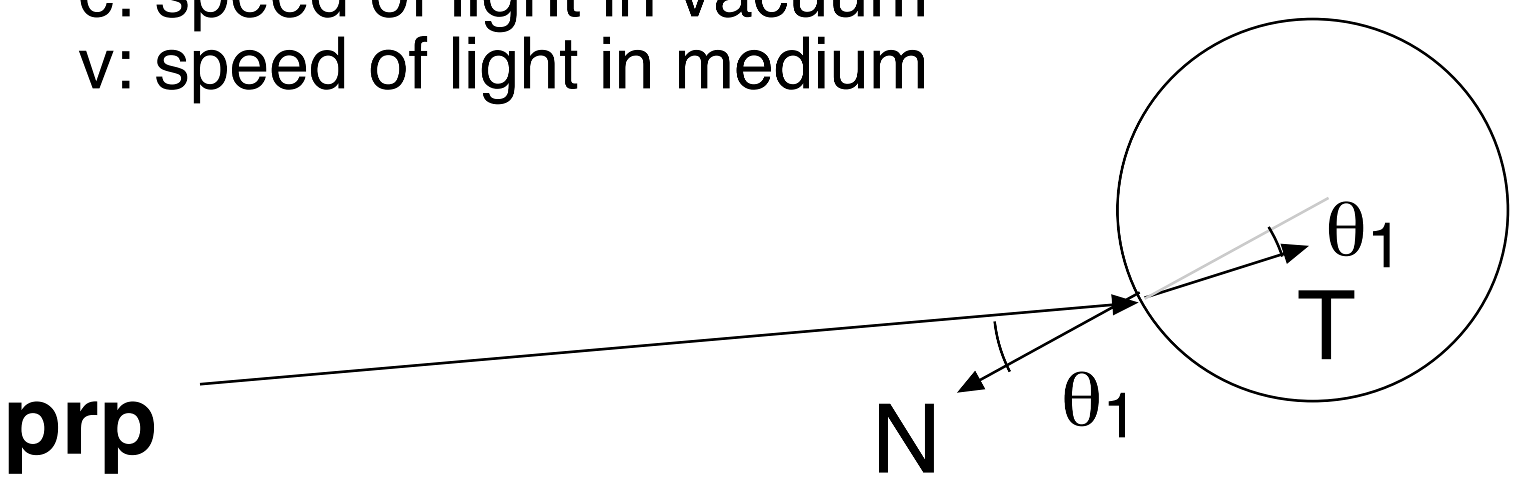
Snell's law:

$\eta_1 \sin\theta_1 = \eta_2 \sin\theta_2$

Refraction index:
$\eta = c / v$
c: speed of light in vacuum
v: speed of light in medium

Transparent object

Look for more contributions here



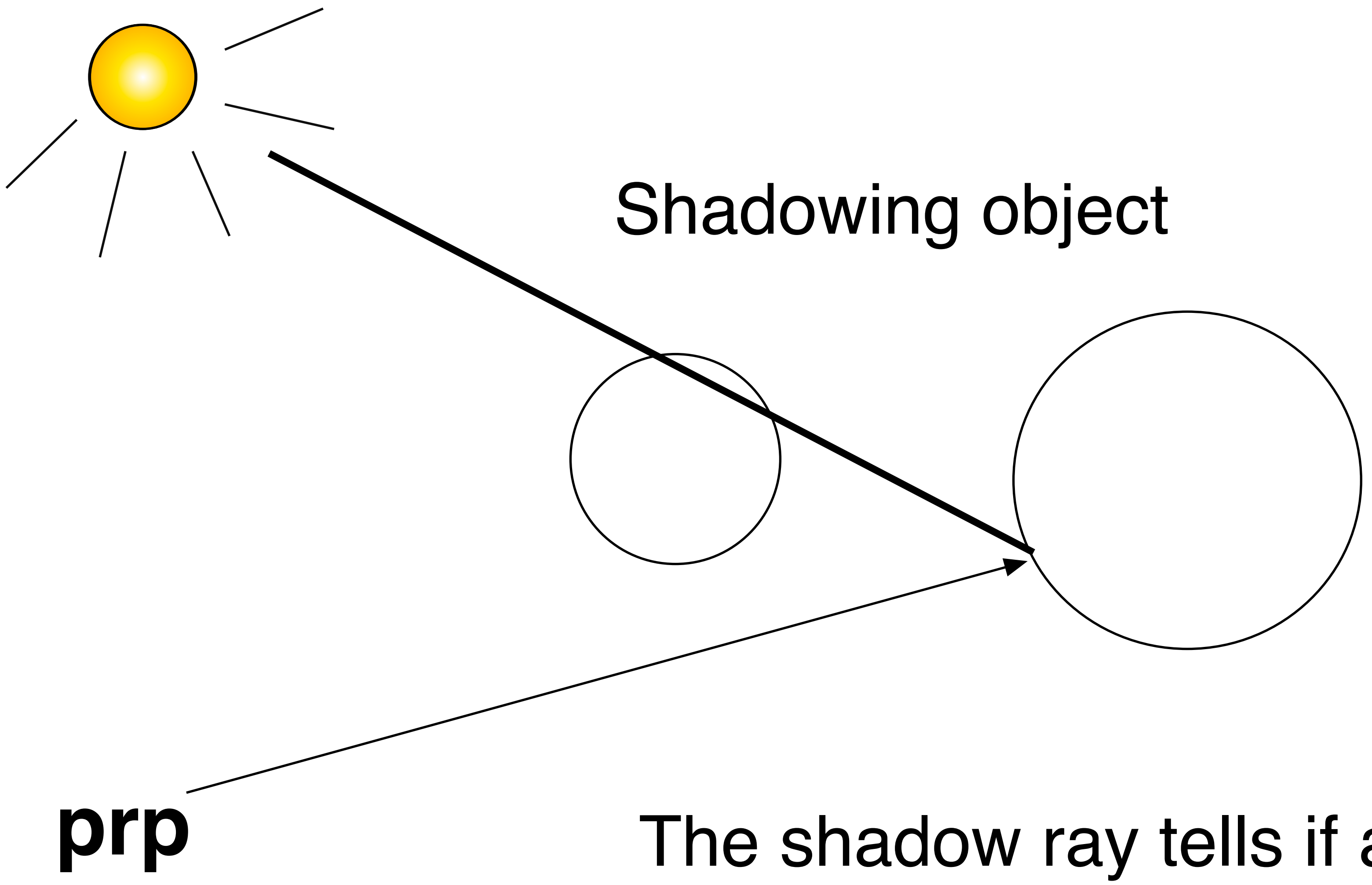
Outgoing angle is given by incoming angles and the density of each material.

Summing up

The total intensity is the sum of

- **ambient light**
- **diffuse reflections from each light source**
- **specular reflections from each light source**
- **mirroring reflections**
- **refractions**

The shadow ray



Object that can have illumination (shading) by diffuse or non-perfect specular reflections

The shadow ray tells if a light source can contribute to illumination

Ray-surface intersections

Ray equation:

$\mathbf{p} = \mathbf{p_0} + \mu\mathbf{u}, \mu > 0$

Combine with the equation of the surface.

Easiest surface: Sphere!

$x^2 + y^2 + z^2 = r^2$

Not quite as easy as for ray casting, since p_0 can now be any point.

```
function RayTrace(p0, u, depth)

if depth > max then return BLACK

μ := FindIntersection(p0, u) // Returns more data, see below

if μ <= 0 then return BACKGROUND_COLOR

Ilocal :=0
IR := 0
IT := 0

if ka ≠ 0 and kd ≠ 0 and ks ≠ 0 then
 Ilocal := ka*Ia + ∑ (diffuse shading + specular shading)
// Sum is for all visible light sources

if kR ≠ 0 then
 R := CalculateReflection(u, N)
 IR := RayTrace(p0 + μ*u, R, depth+1)

if kT ≠ 0 then
 T := CalculateRefraction(u, N, h1, h2)
 IT := RayTrace(p0 + μ*u, T, depth+1)

return Ilocal + IR + IT
```