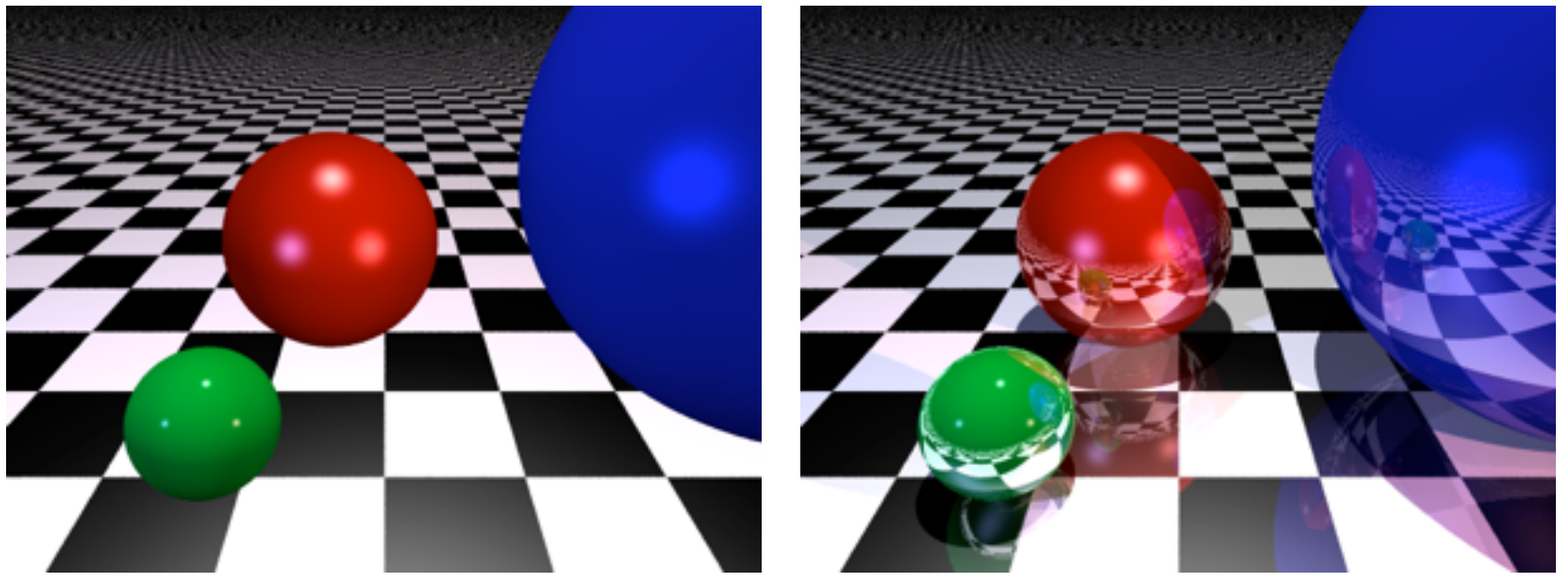
Introduction to Computer Graphics

Ray Intersections



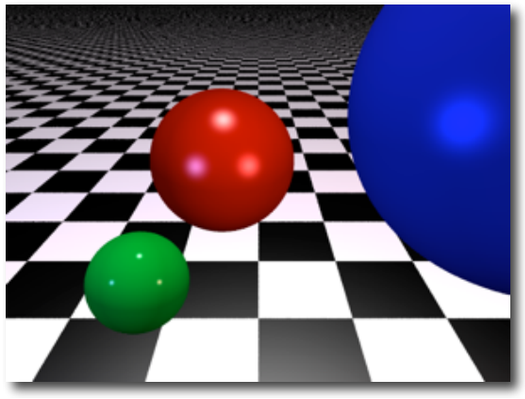
Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing

Universität Bielefeld

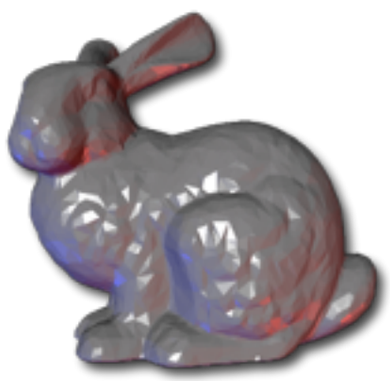
CITEC

Overview

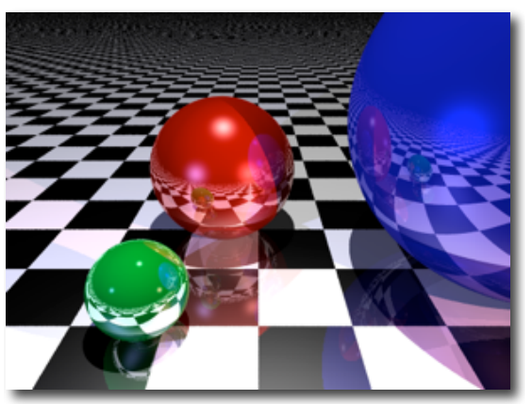


Rendering Pipeline

Geometry



Global Illumination



Image

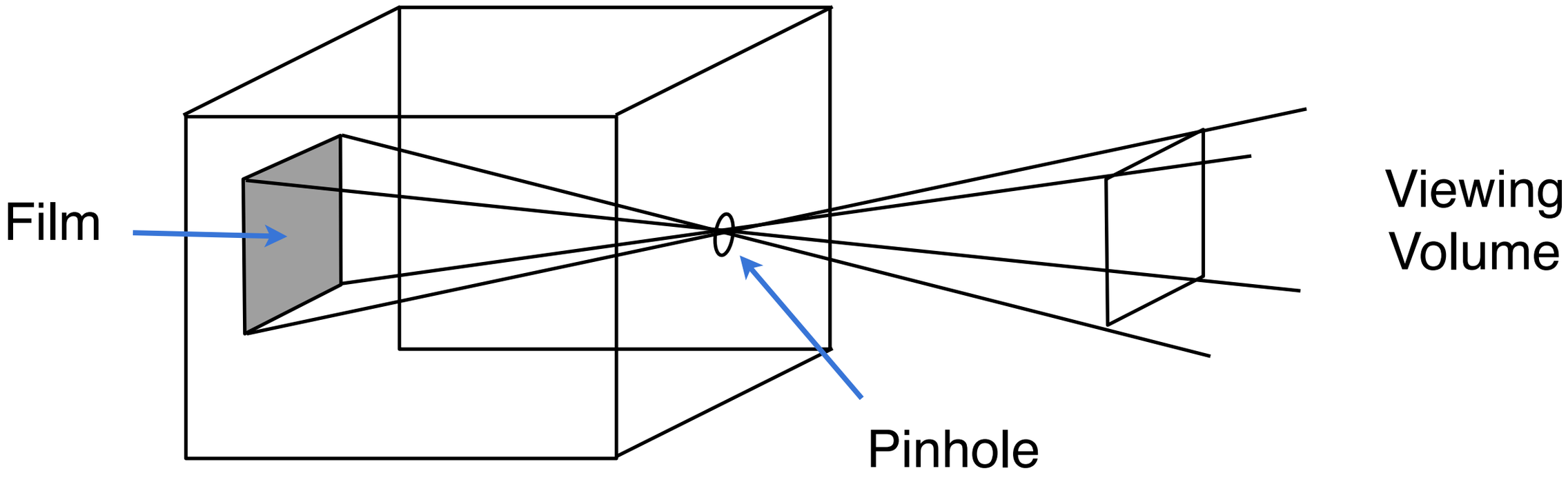
Global Illumination

- Goal: Generate photo-realistic images
- Central concept: Simulate the behavior of light
  - “Physically-Based Rendering”
  - “Light Transport”

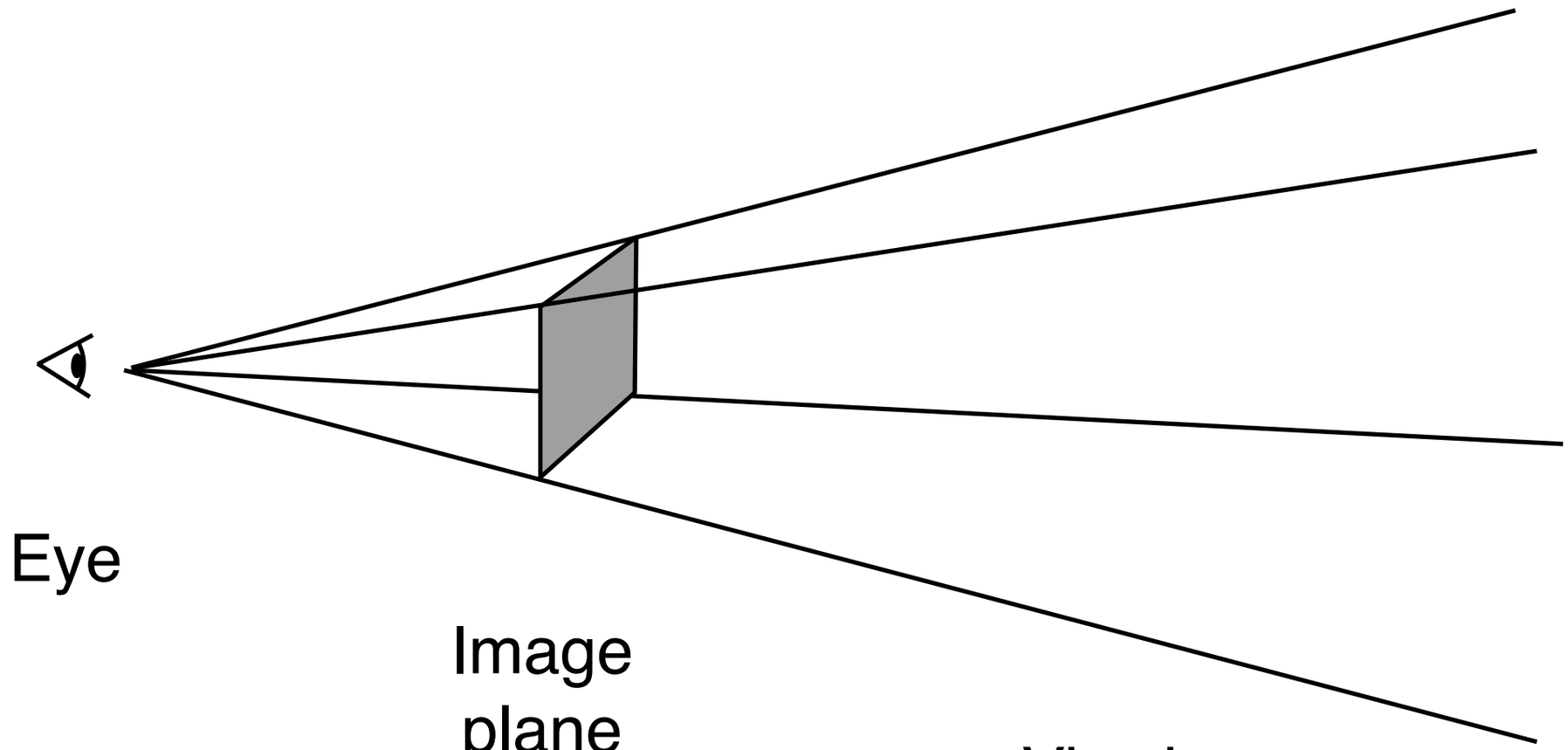
Think hard!

- What is light?
- How is light propagated?
- How does a camera work?
- How does light interact with objects?
- What is a color?

Pinhole Camera

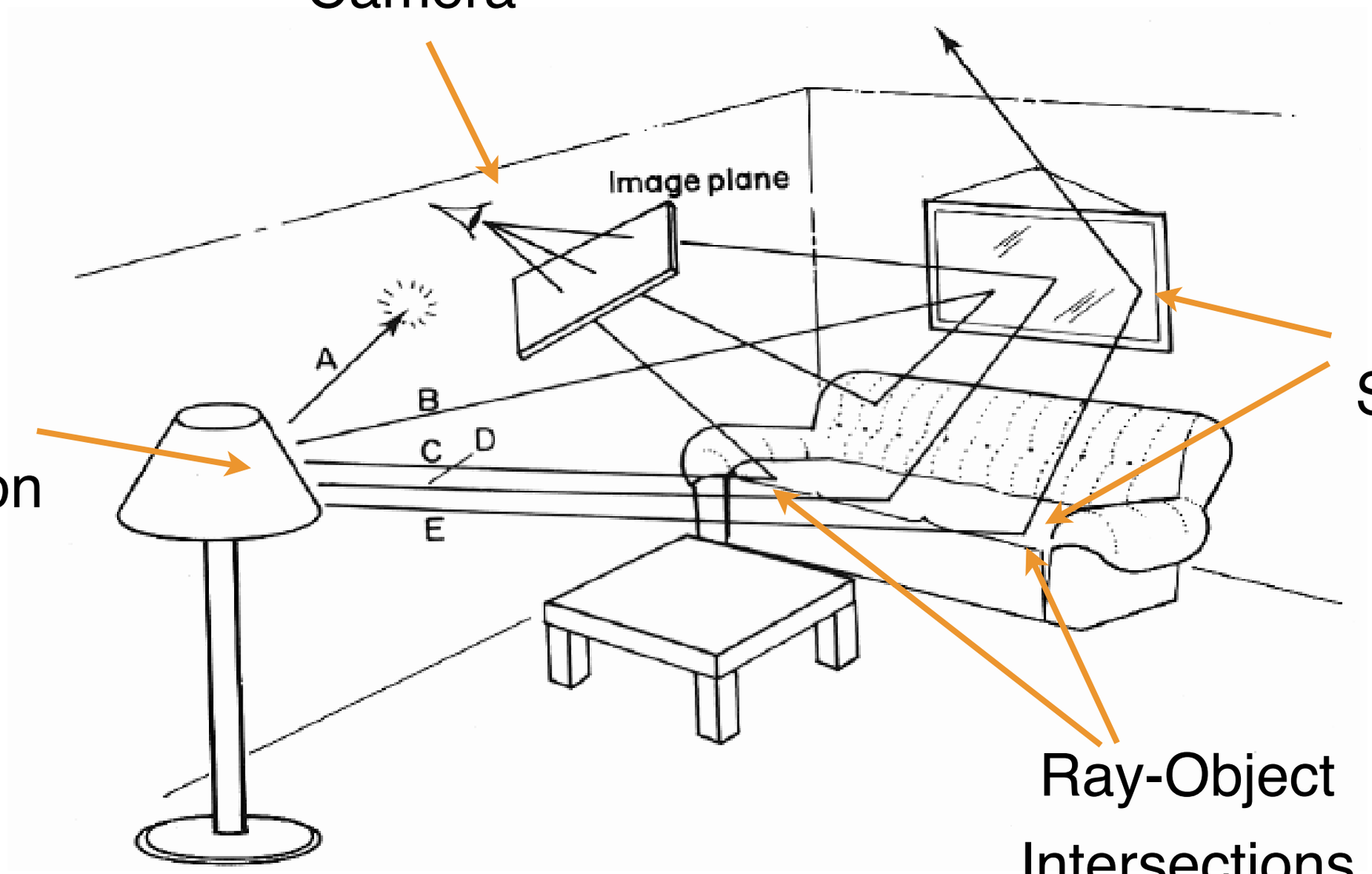


Pinhole Camera



Light Transport

Camera



Light Distribution

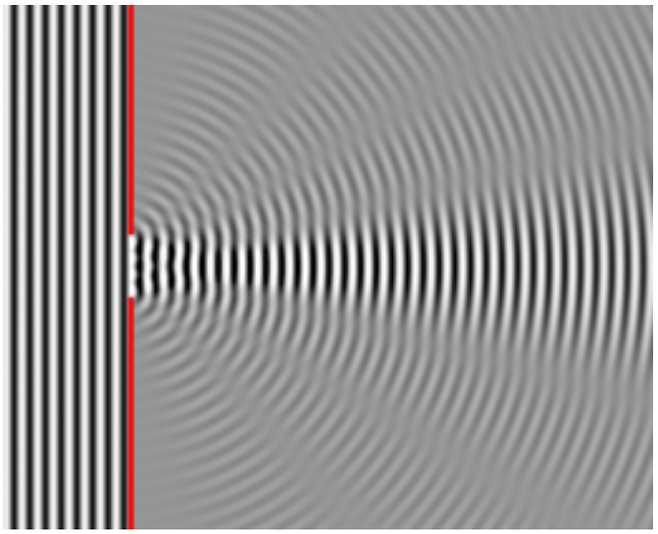
Surface Scattering

Ray-Object Intersections

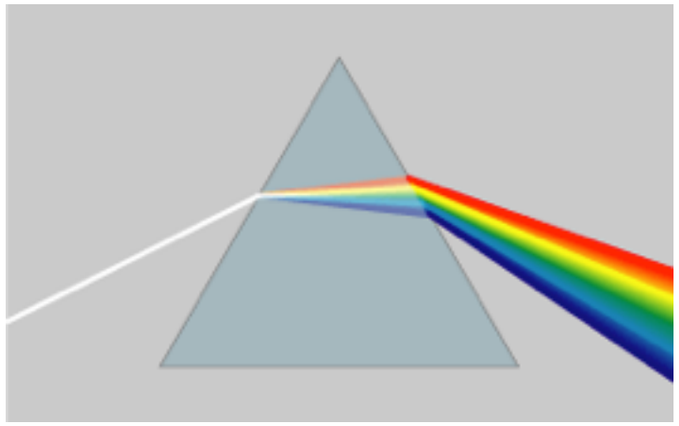
Light Transport - Assumptions

- Geometric optics (ray optics)
  - no diffraction, no interference, no polarization
- Light travels in straight lines through a vacuum
  - no atmospheric scattering, no gravity effects
- Discrete-wavelength color approximation (RGB)
  - no dispersion, no fluorescence
- Superposition
  - no non-linear reflecting materials

Neglected Effects



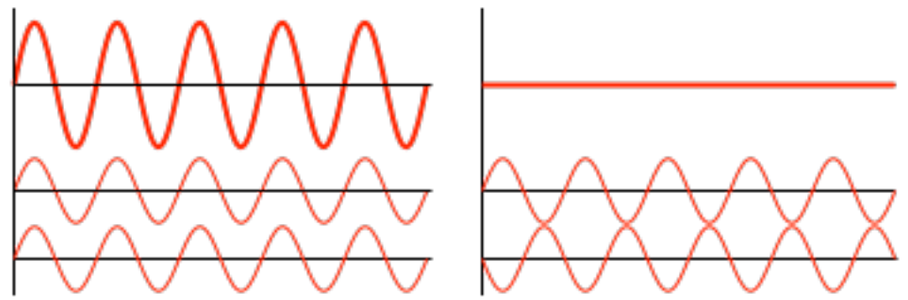
Diffraction



Dispersion

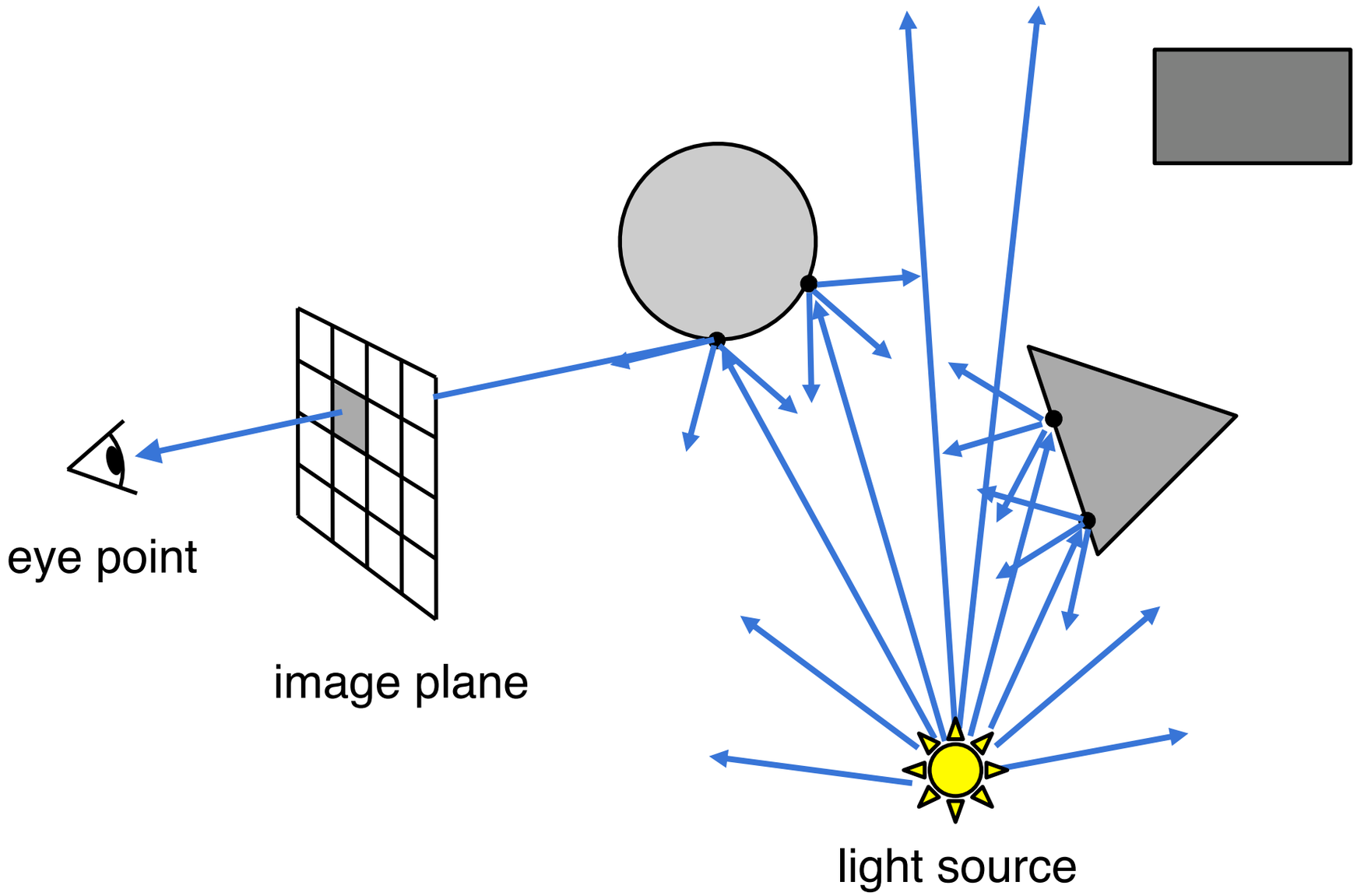
Fluorescence

Polarization

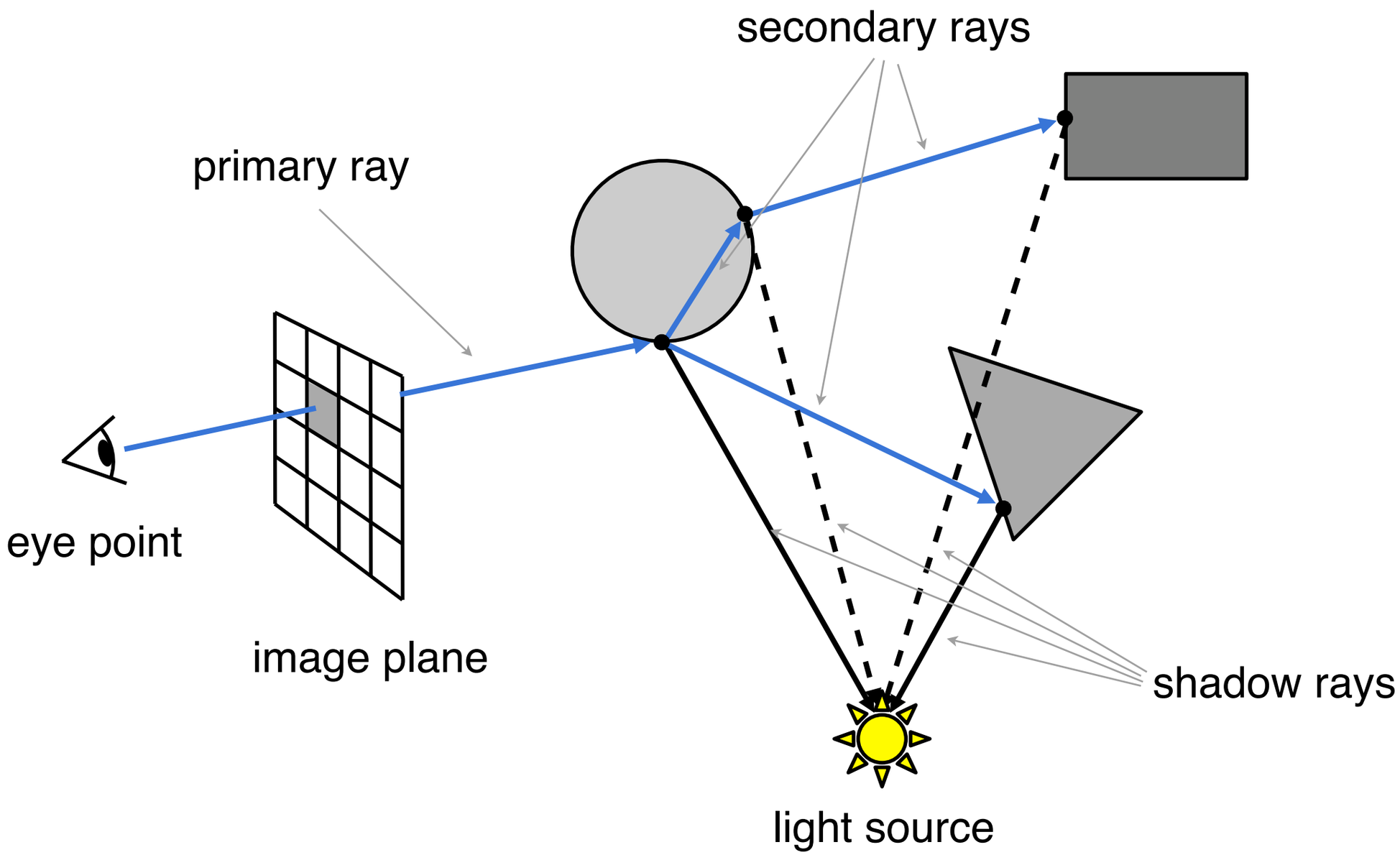


Interference

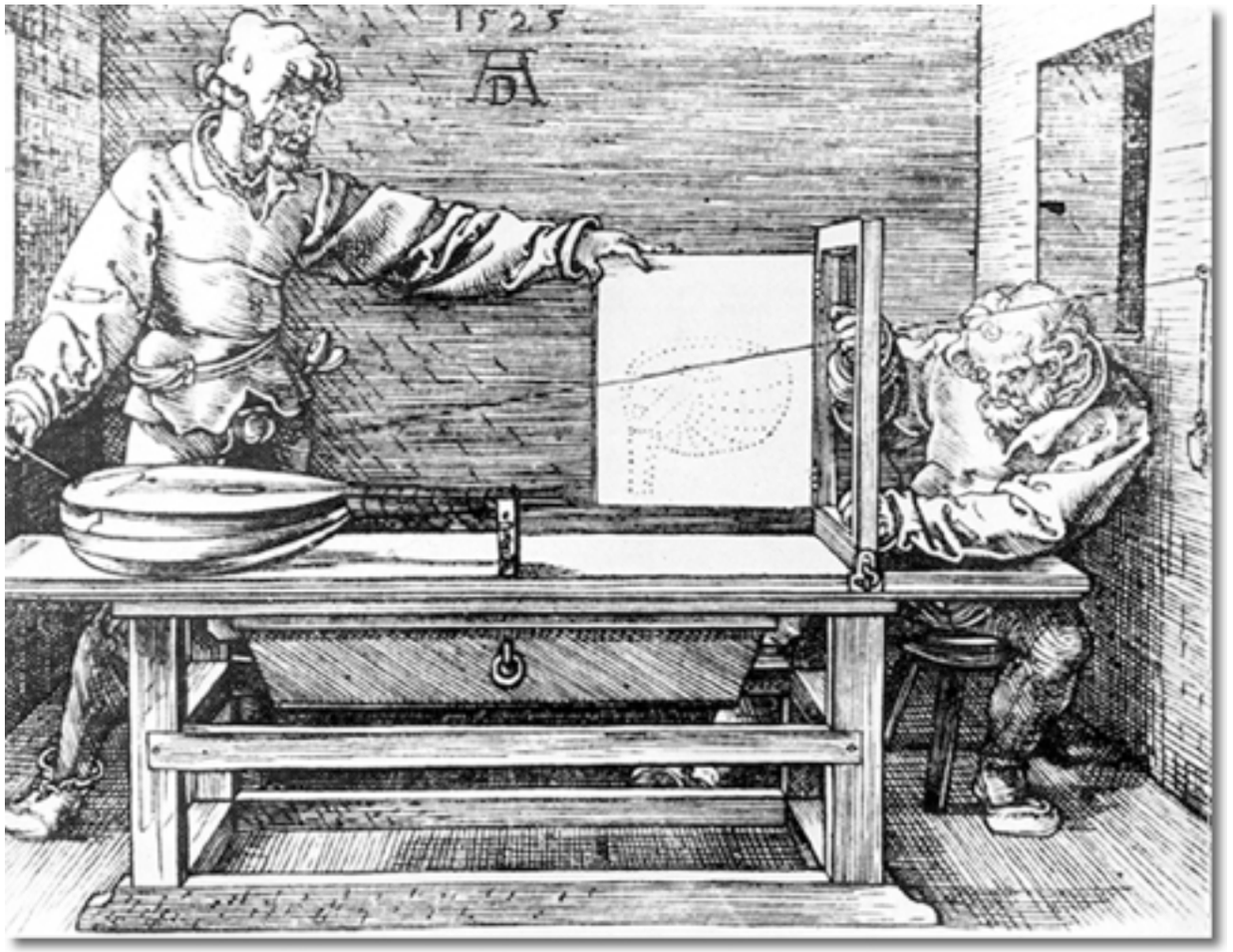
Forward Ray Tracing



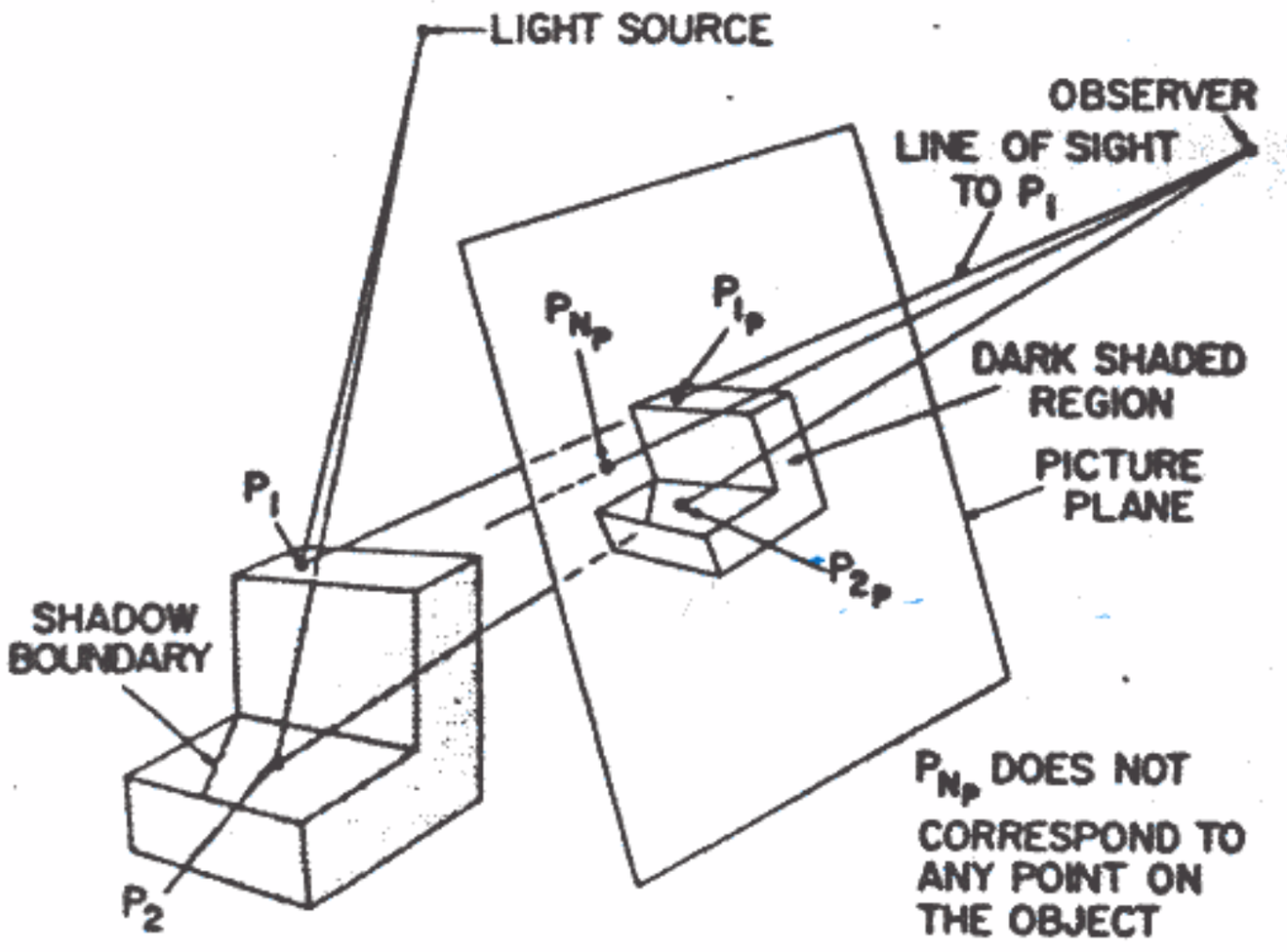
Backward Ray Tracing

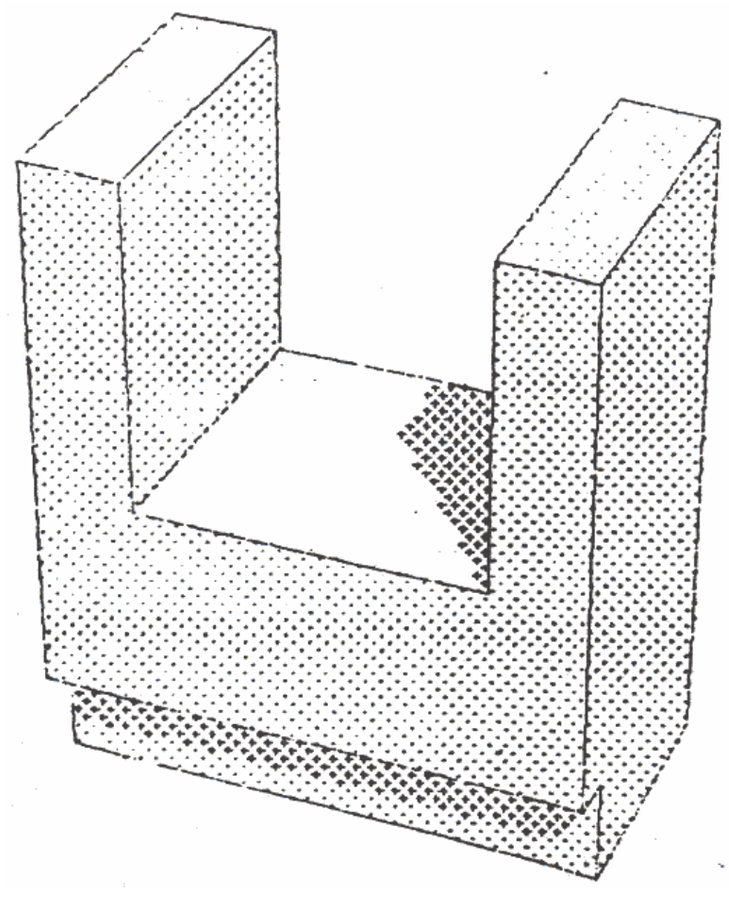


Dürer (1525)

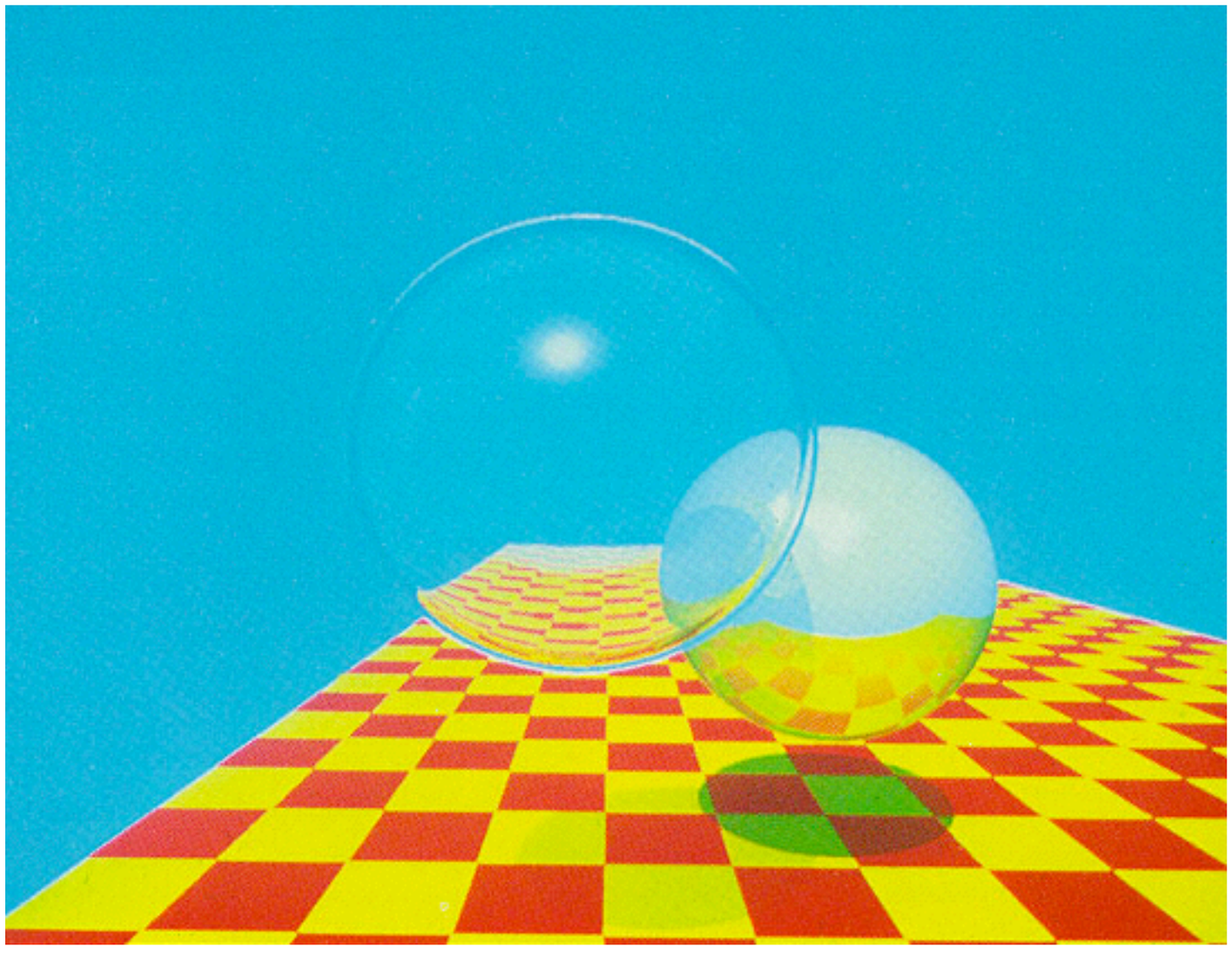


Appel (1968)

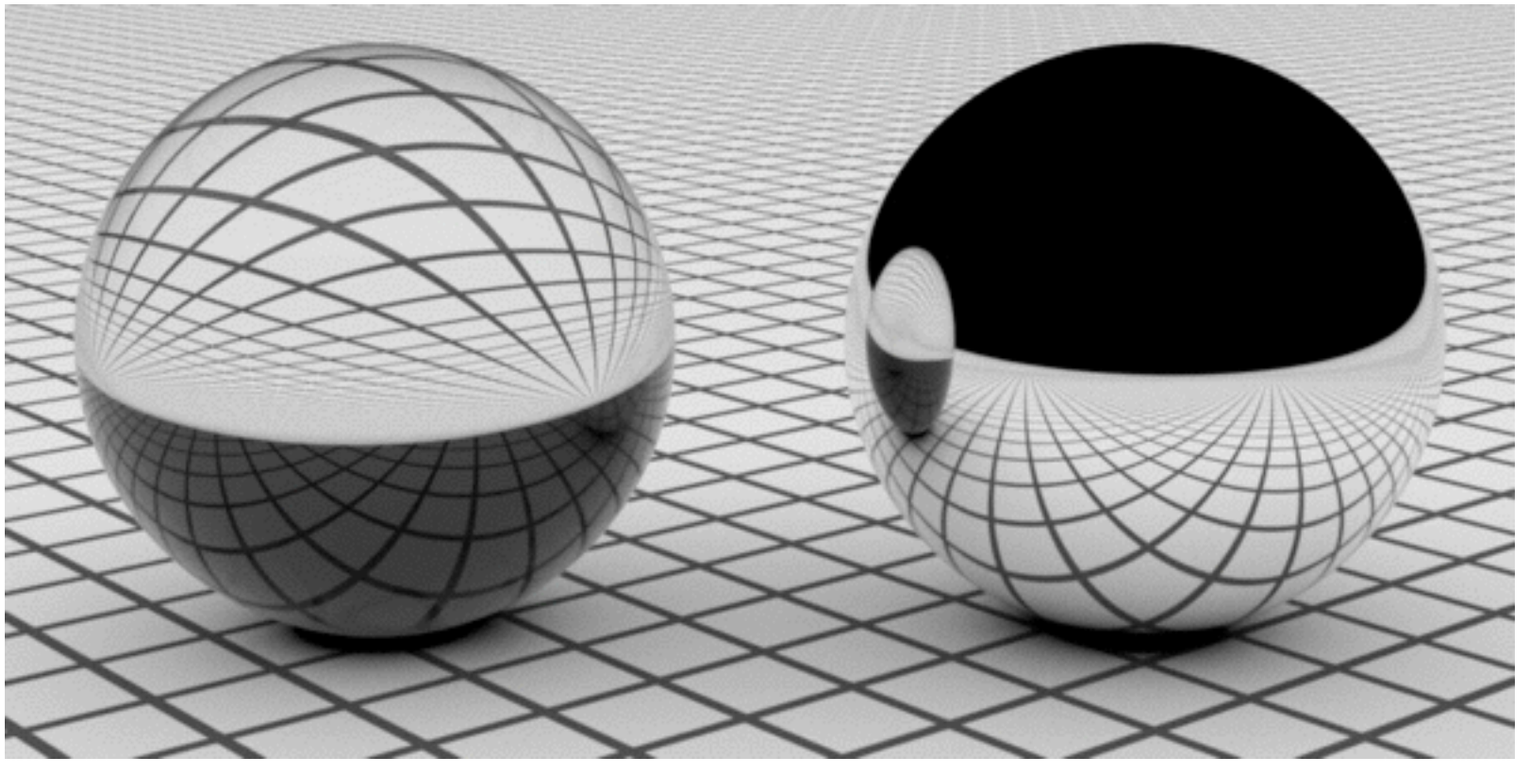

LIGHT SOURCE
OBSERVER
LINE OF SIGHT TO P_1
P_{N_P}
P_{1_P}
DARK SHADED REGION
PICTURE PLANE
P_1
P_{2_P}
SHADOW BOUNDARY
P_2
P_{N_P} DOES NOT CORRESPOND TO ANY POINT ON THE OBJECT



Whitted (1979)



Ray Tracing

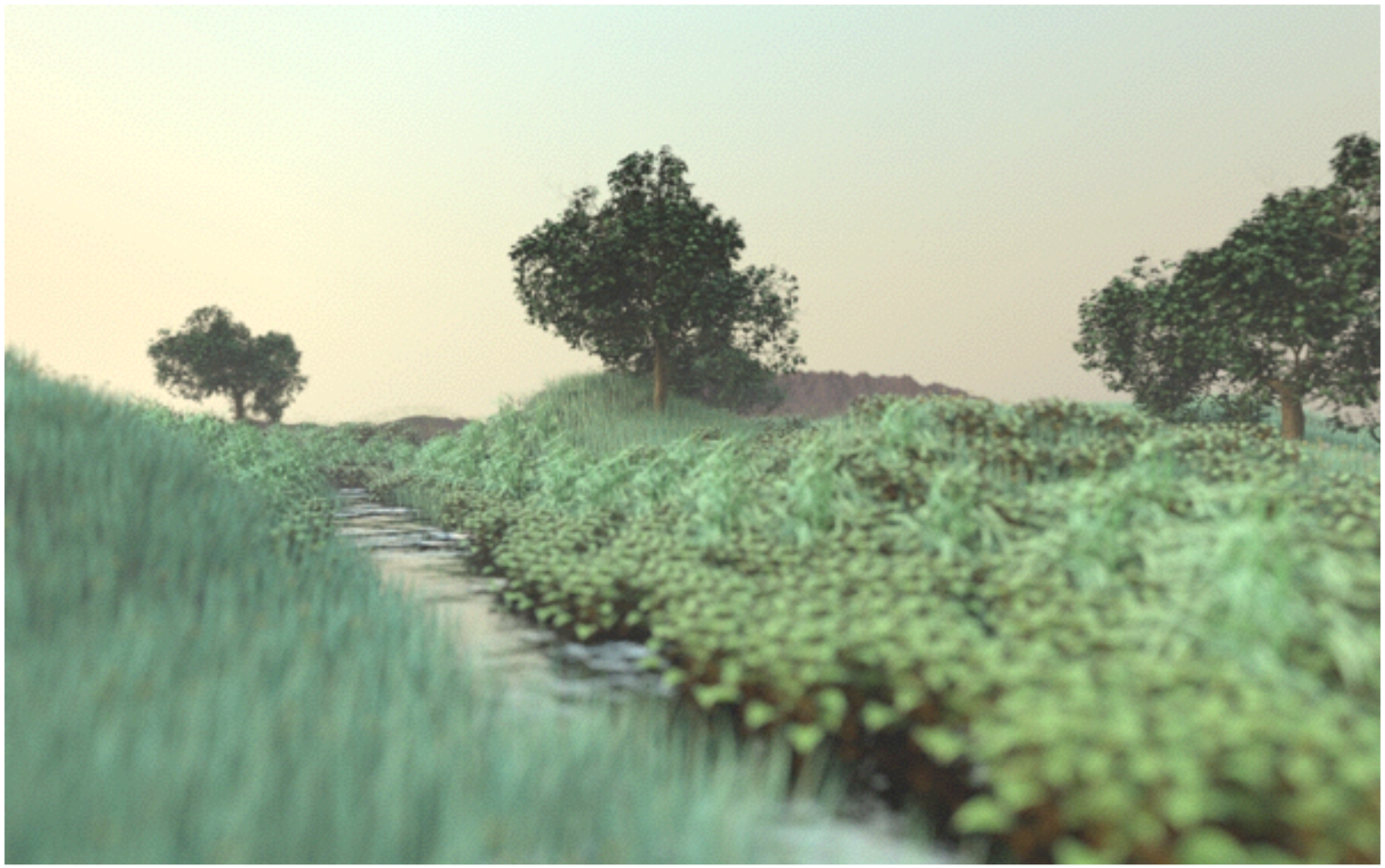


http://www.pbrt.org/gallery.php

Ray Tracing

http://www.pbrt.org/gallery.php

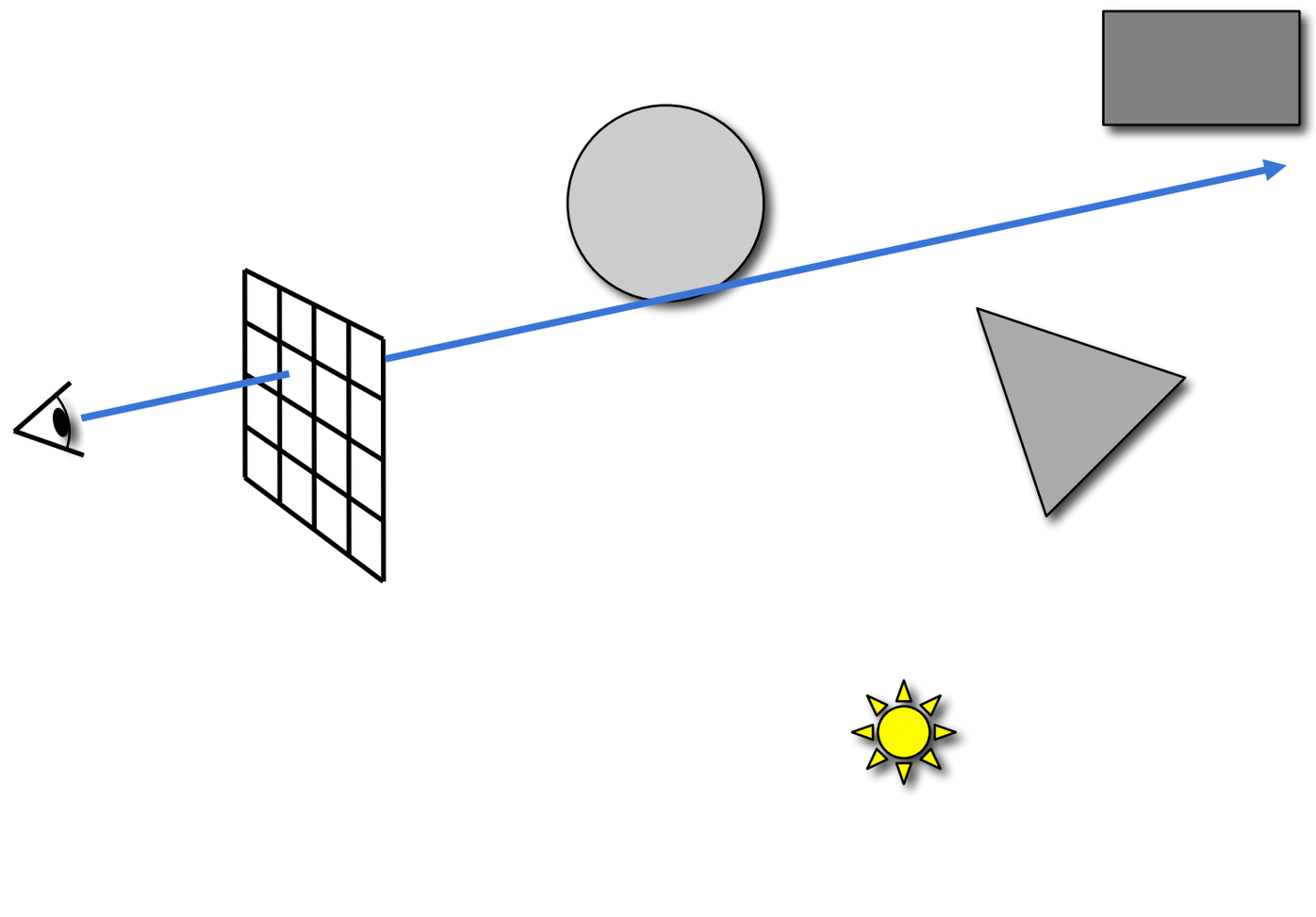
Ray Tracing



http://www.pbrt.org/gallery.php

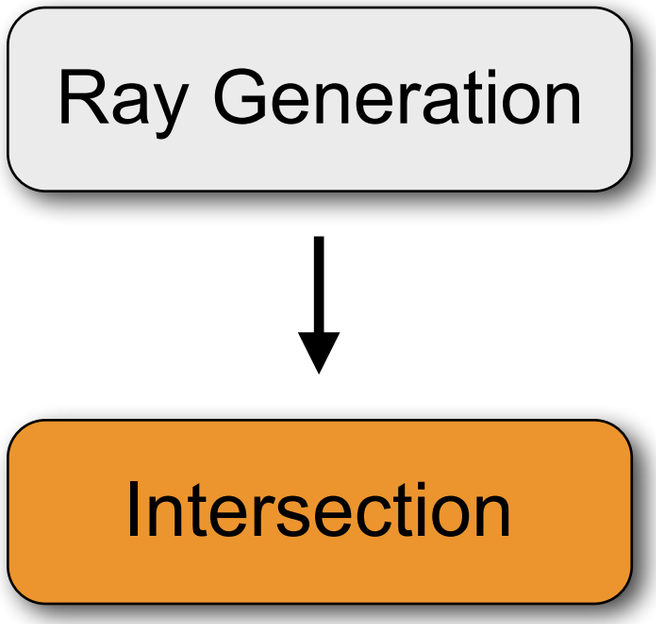
Basic Ray Tracing Pipeline

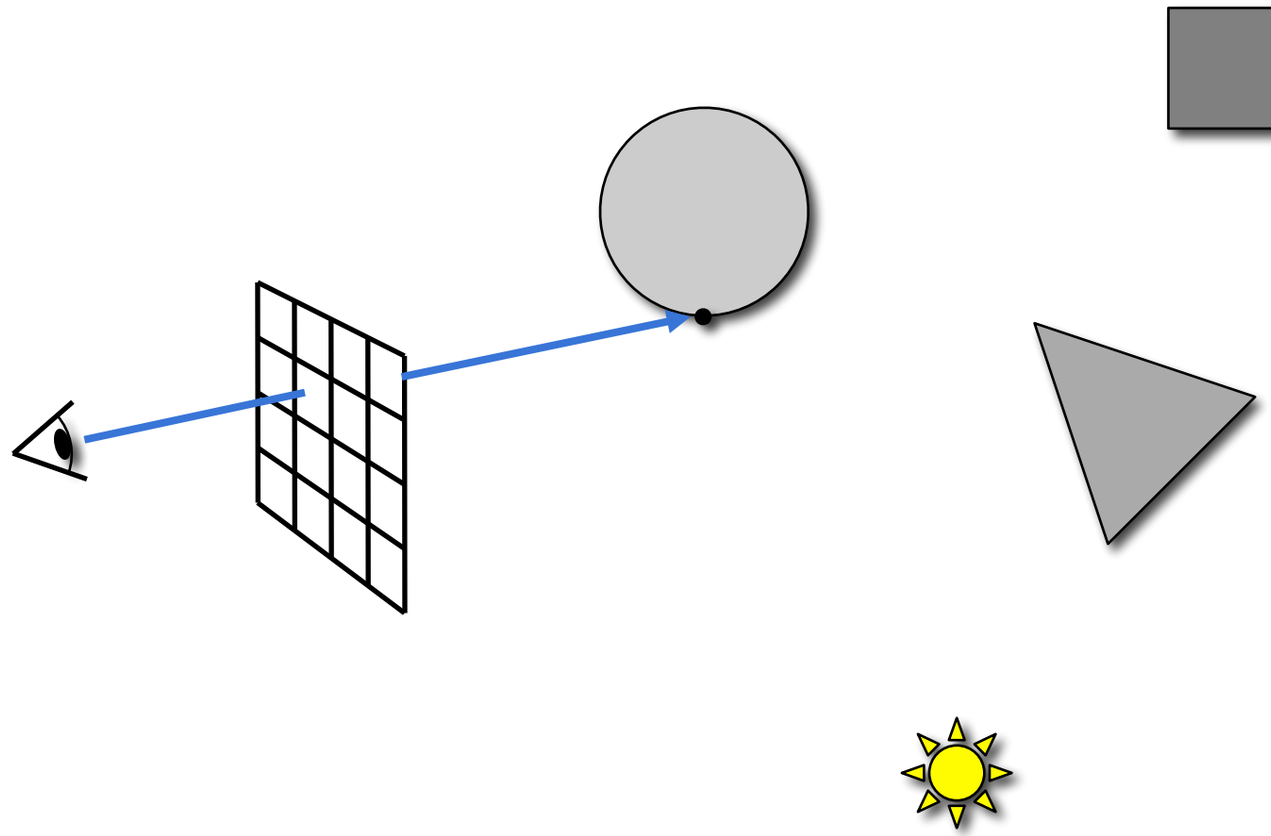
Ray Generation



Basic Ray Tracing Pipeline

Ray Generation



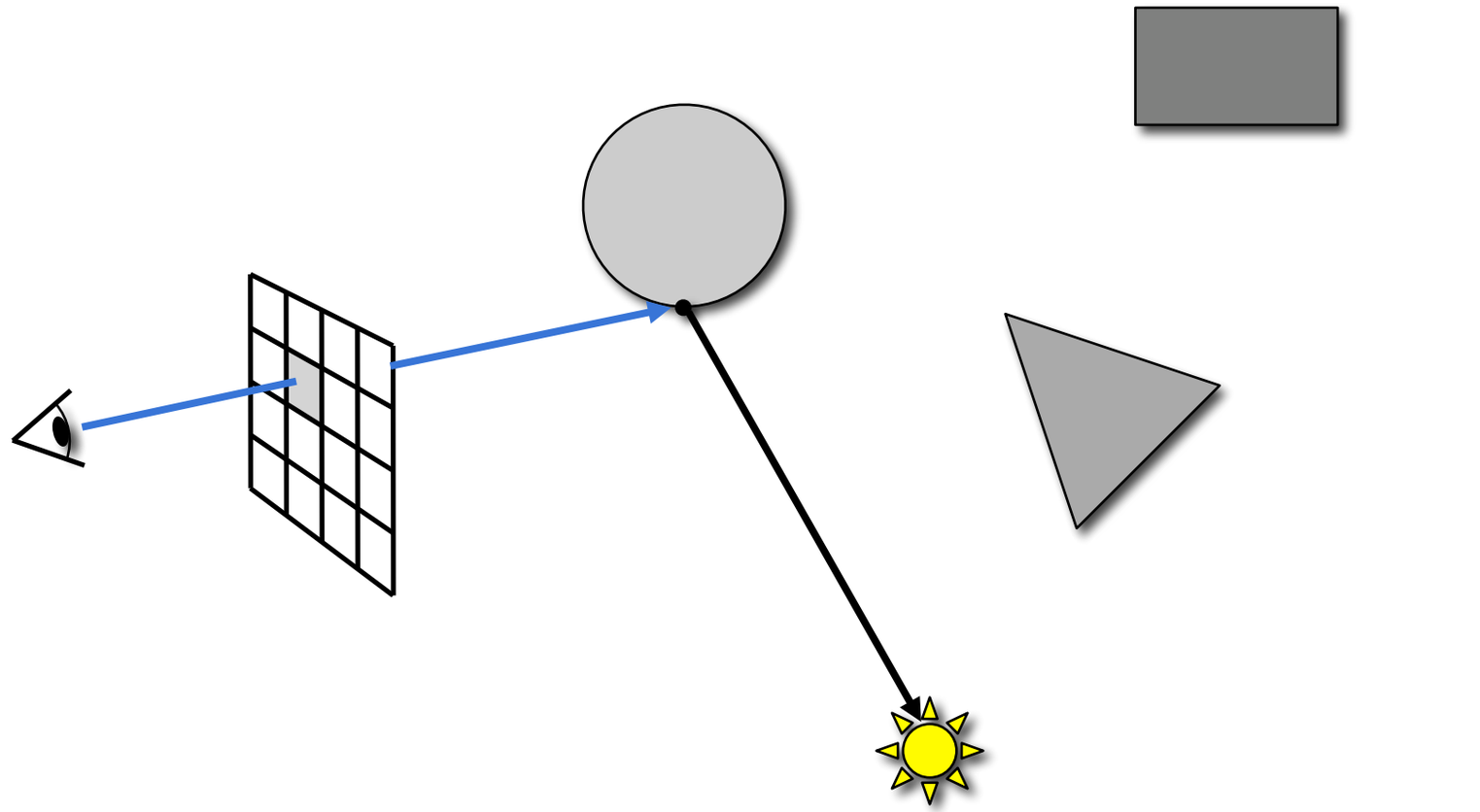


Basic Ray Tracing Pipeline

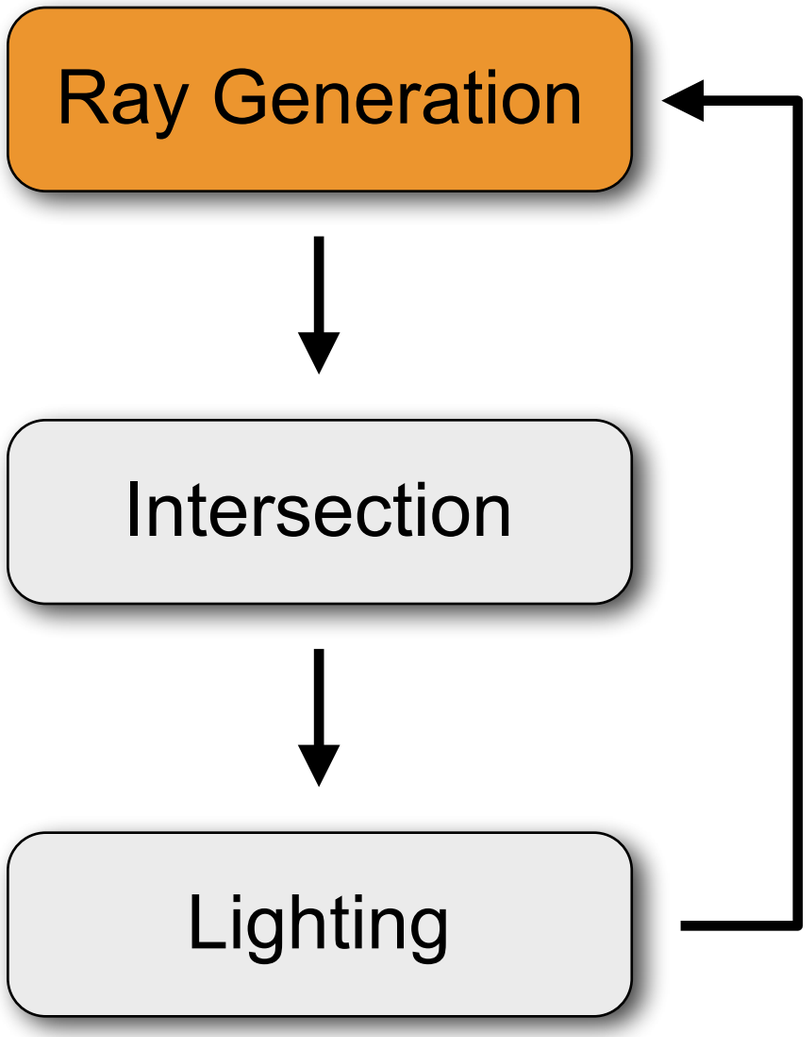
Ray Generation

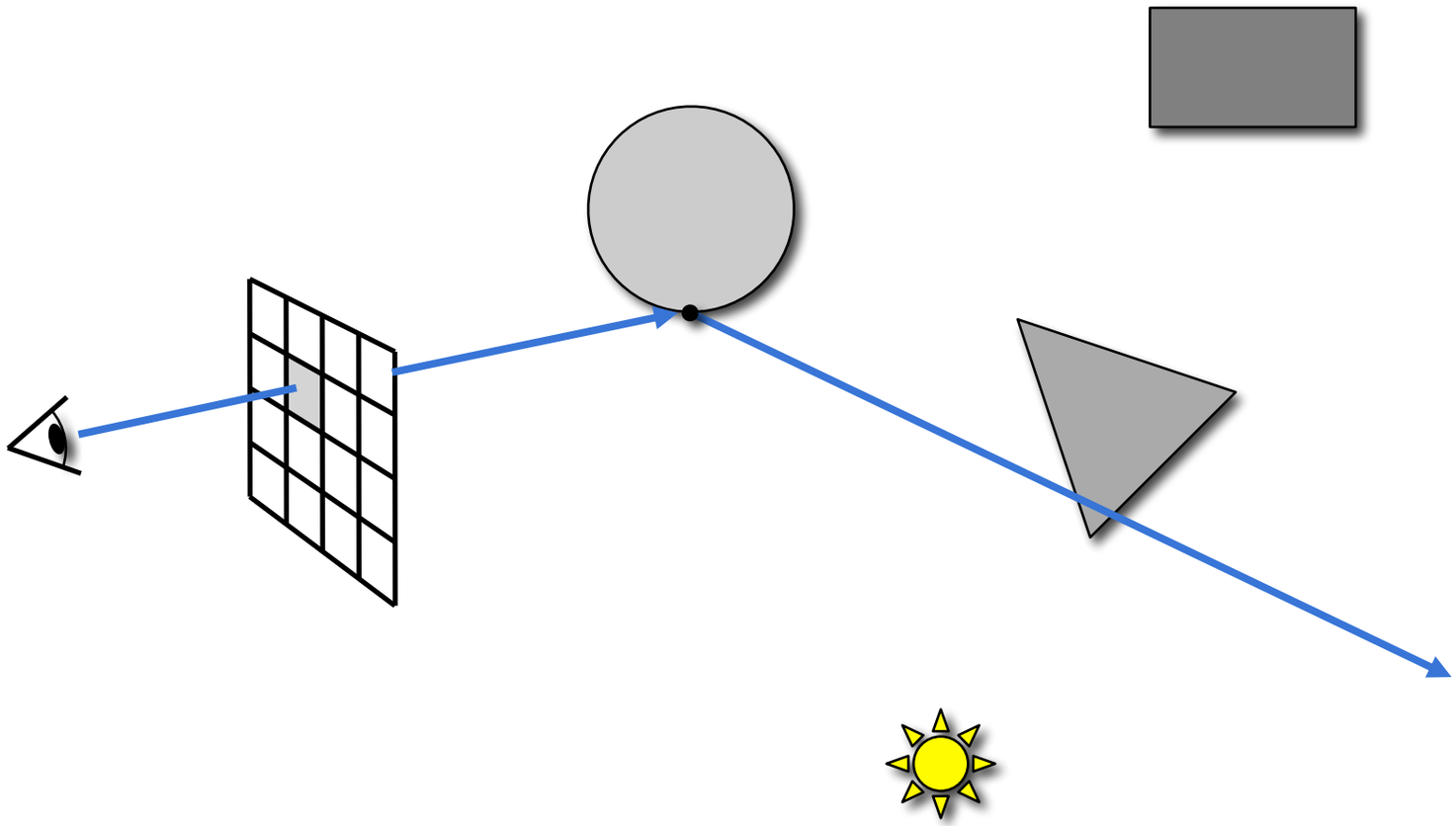
Intersection

Lighting

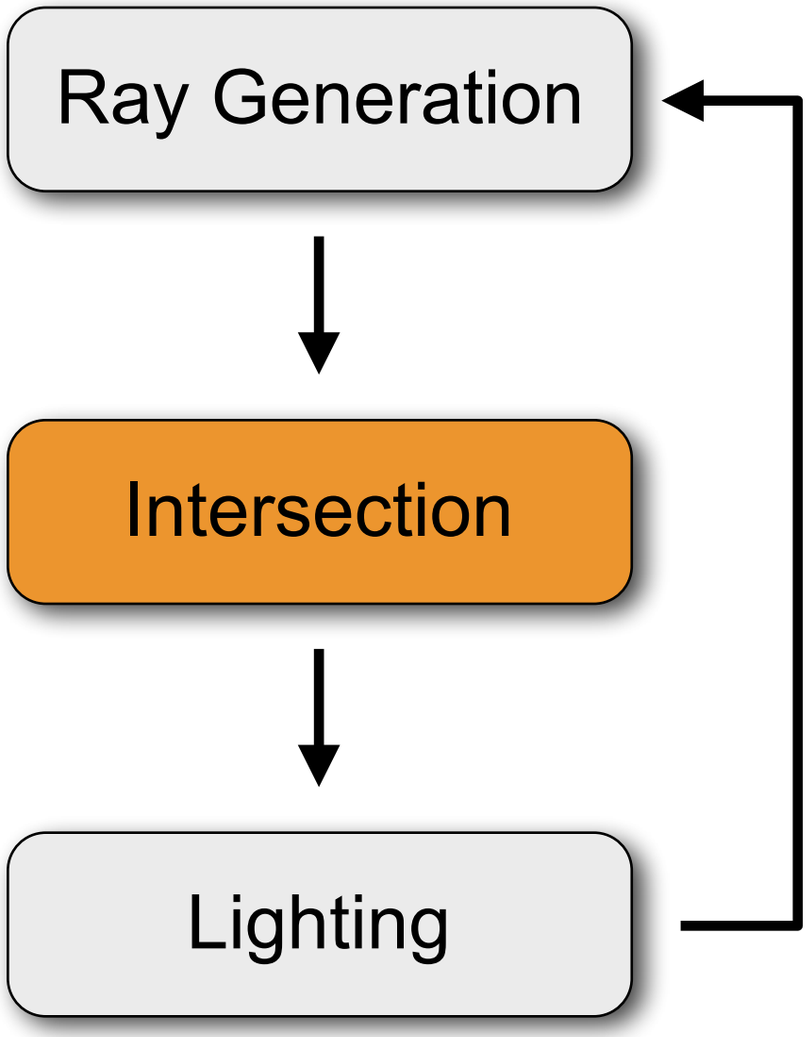


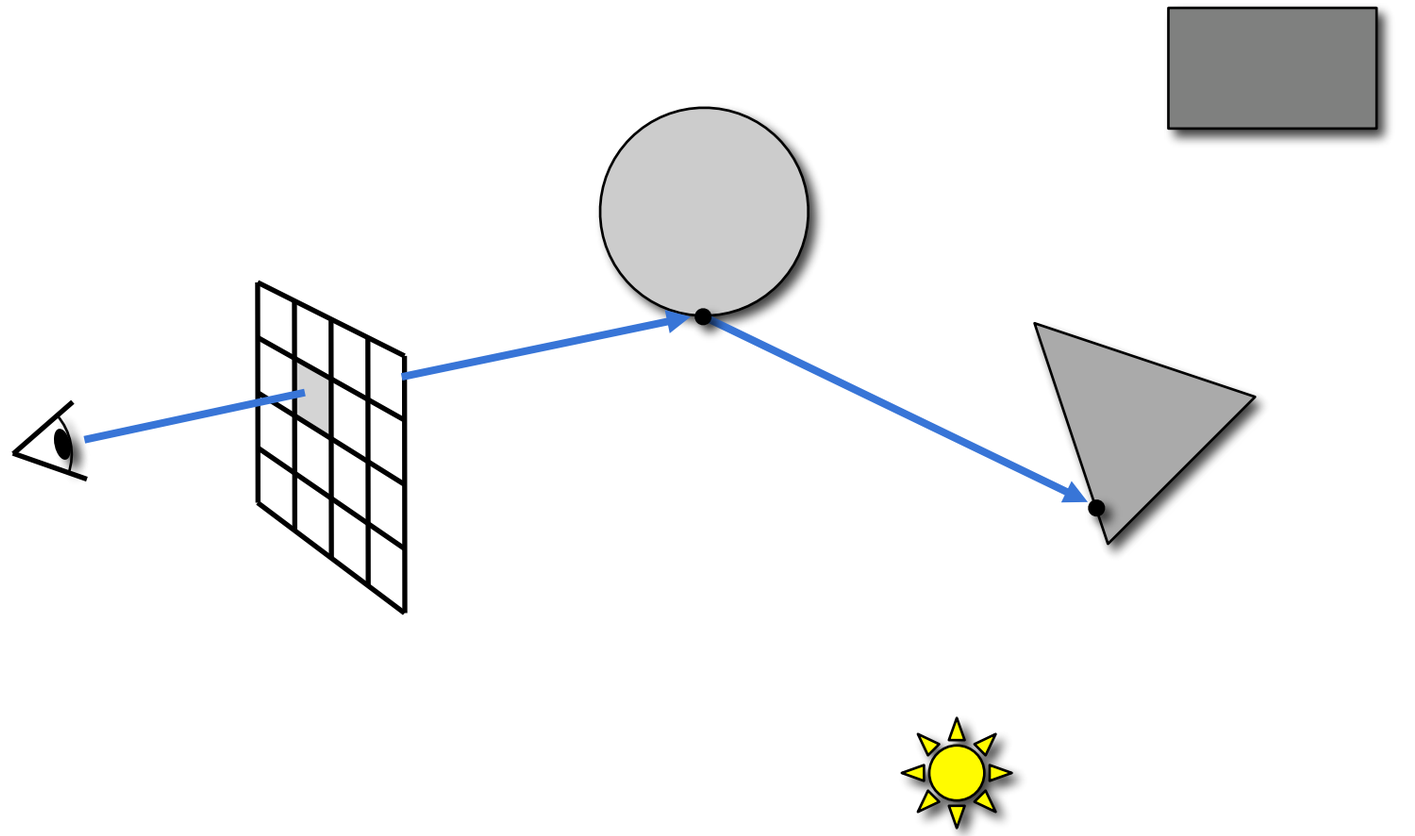
Basic Ray Tracing Pipeline



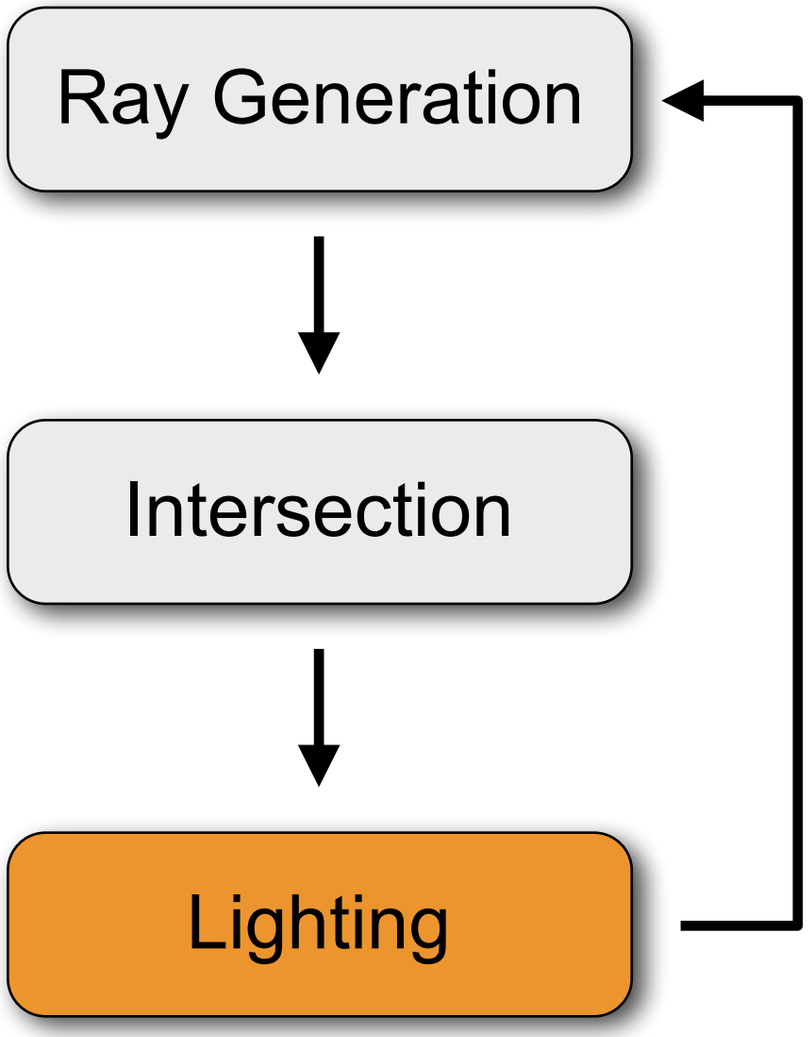


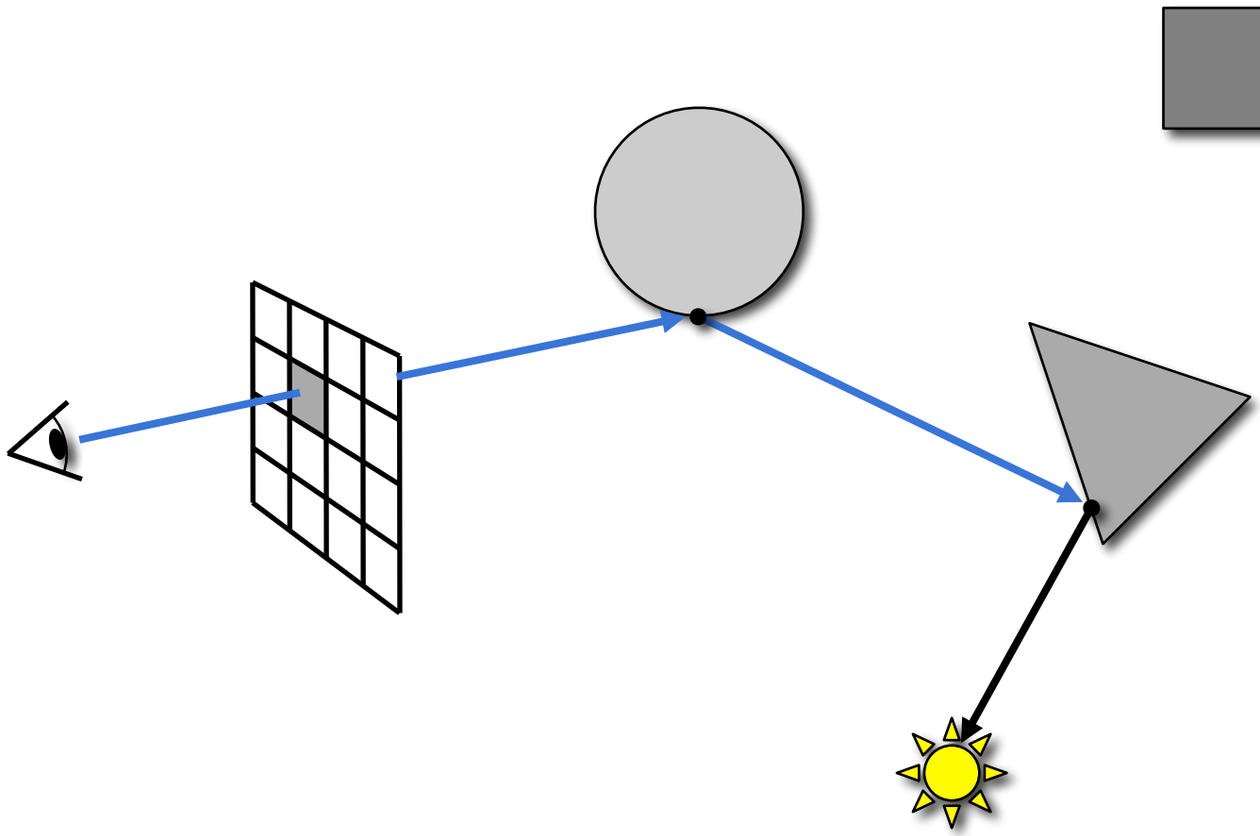
Basic Ray Tracing Pipeline



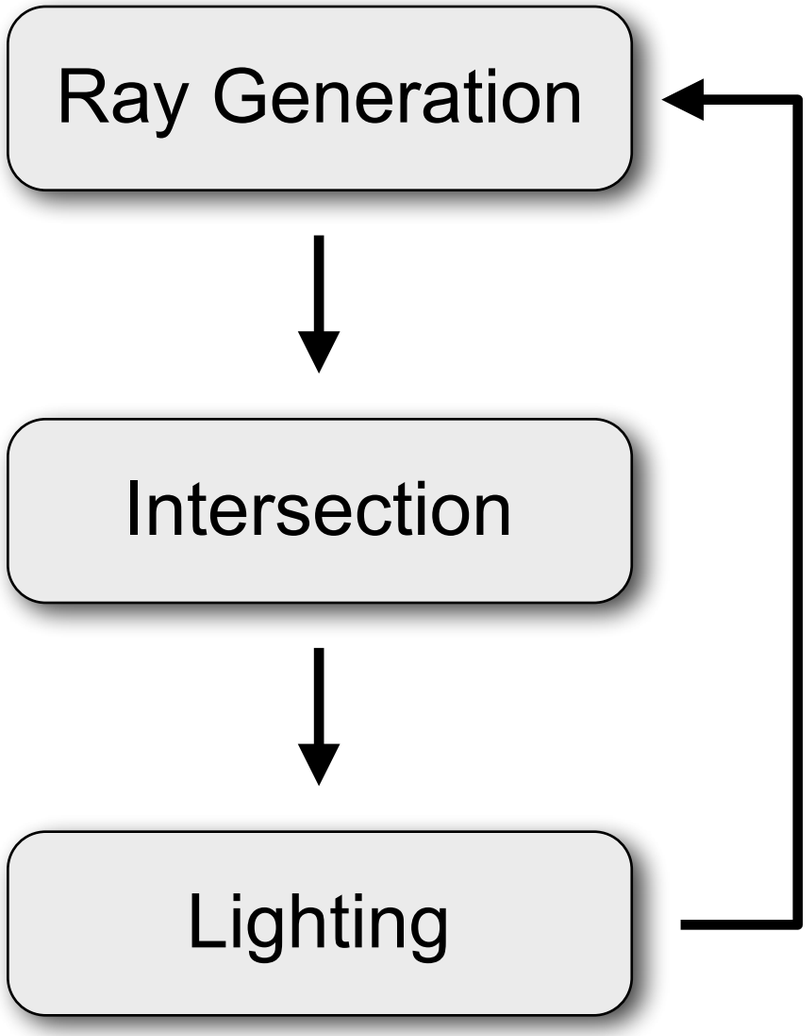


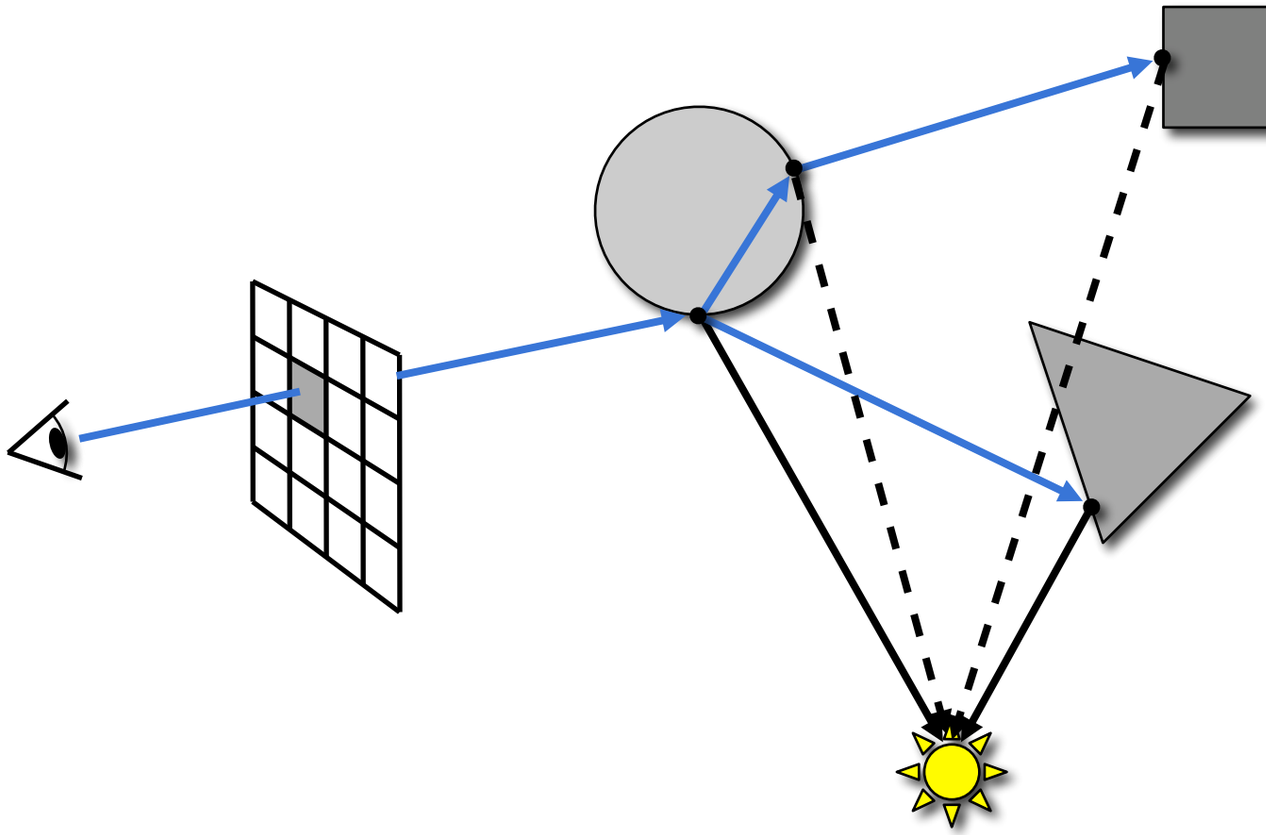
Basic Ray Tracing Pipeline





Basic Ray Tracing Pipeline



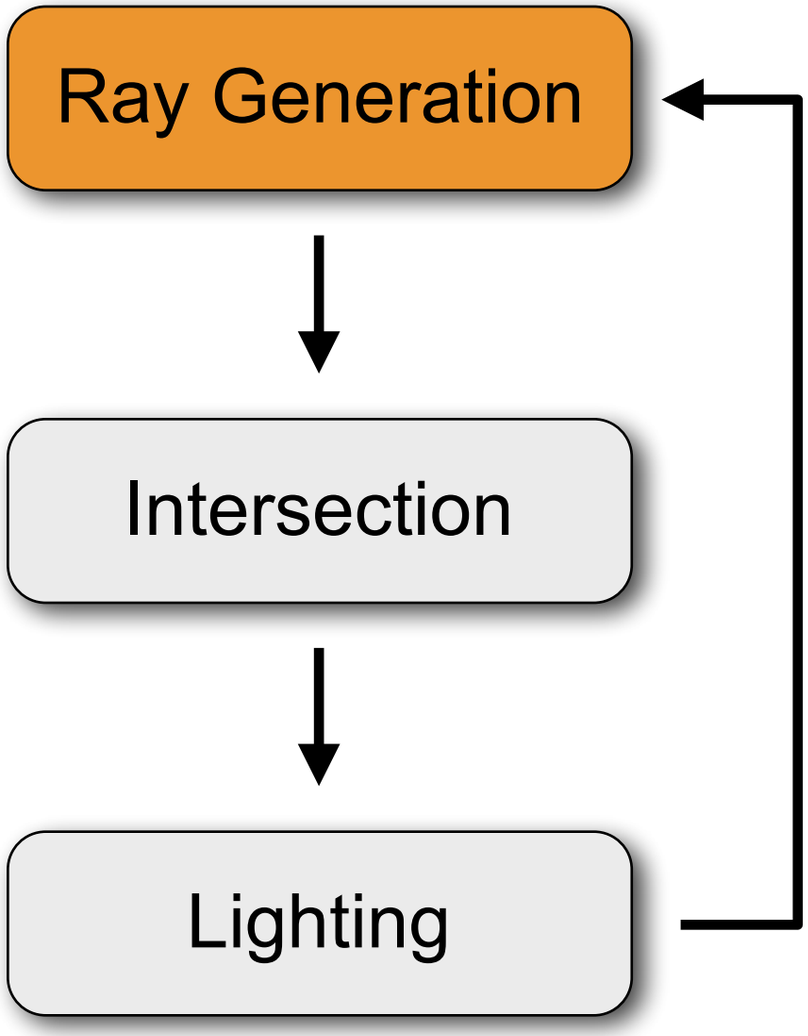


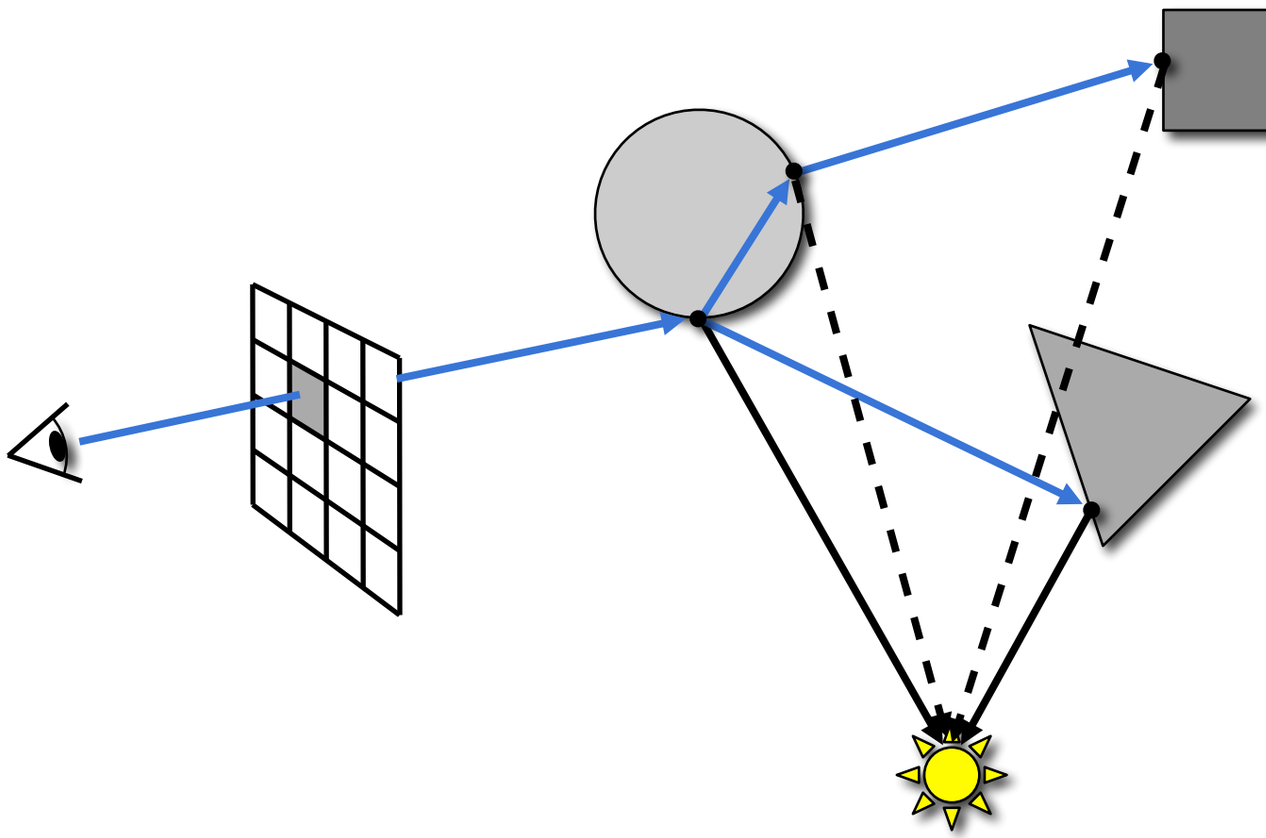
Ray Tracing Pseudo-Code

```
void raytrace()
{
 for (int x=0; x<xresolution; ++x)
 {
    for (int y=0; y<yresolution; ++y)
    {
       ray = generate_primary_ray(x,y);
       color[x,y] = trace(ray);
    }
 }
}
```

recursive!

Ray Generation



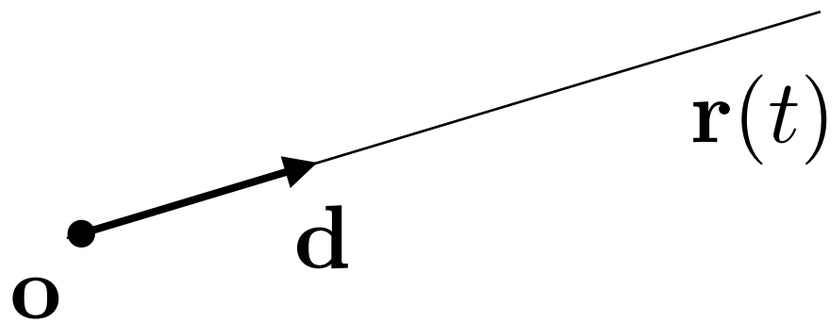


Rays

- Ray equation (explicit form)

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

ray origin direction (normalized)



Vector Space $\mathbf{R}^n$

- Scalars: $\alpha, \beta, \gamma \in \mathbb{R}$
- Points / vectors: $\mathbf{p}, \mathbf{q}, \mathbf{r} \in \mathbb{R}^n$
- Linear combination: $\alpha\,\mathbf{p} + \beta\,\mathbf{q}$

Euclidean Vector Space $\mathbf{R}^n$

- Inner product, scalar product, dot product

$$\langle \mathbf{x}, \mathbf{y} \rangle \;=\; \mathbf{x} \cdot \mathbf{y} \;=\; \mathbf{x}^T \mathbf{y} \;=\; \sum_{i=1}^{n} x_i \, y_i$$

- Induced metric measures length

$$\|\mathbf{x}\| \;=\; \sqrt{\mathbf{x}^T \mathbf{x}} \;=\; \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

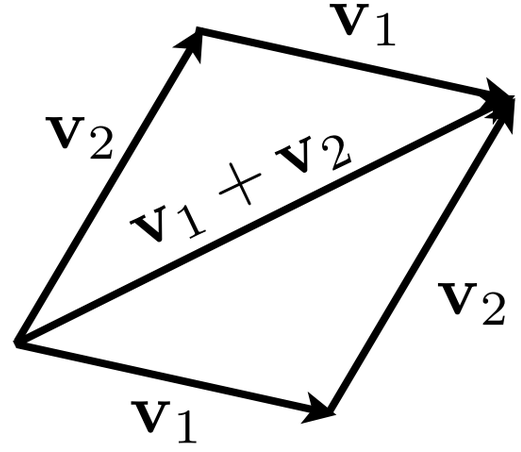
- Measure angle θ between vectors **x** and **y**

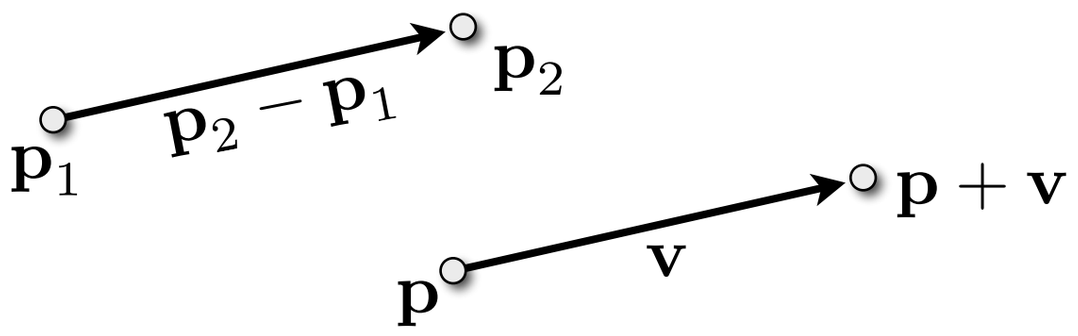
$$\mathbf{x}^T \mathbf{y} \;=\; \|\mathbf{x}\| \cdot \| \mathbf{y}\| \cdot \cos\theta \quad \Rightarrow \quad \theta \;=\; \operatorname{acos}\left(\frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\| \, \|\mathbf{y}\|} \right)$$

Points vs. Vectors

- Subtle distinction
  - points denote positions in $\mathbf{R}^n$
  - vectors denote differences of points
  - vector = point - $(0, ..., 0)^T$

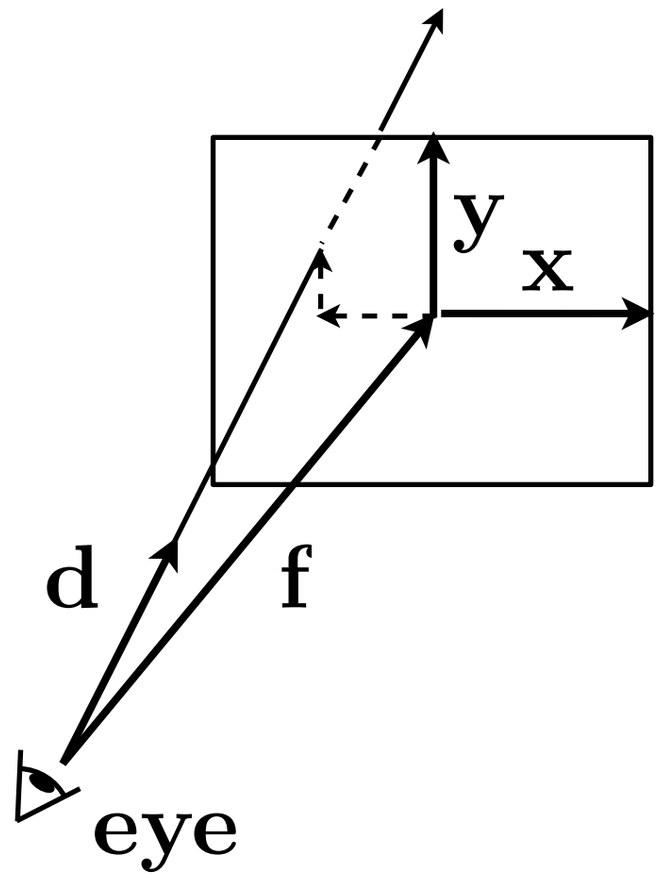
- Meaningful operations
  - vector + vector = vector
  - point - point = vector
  - point + vector = point
  - point + point = ???



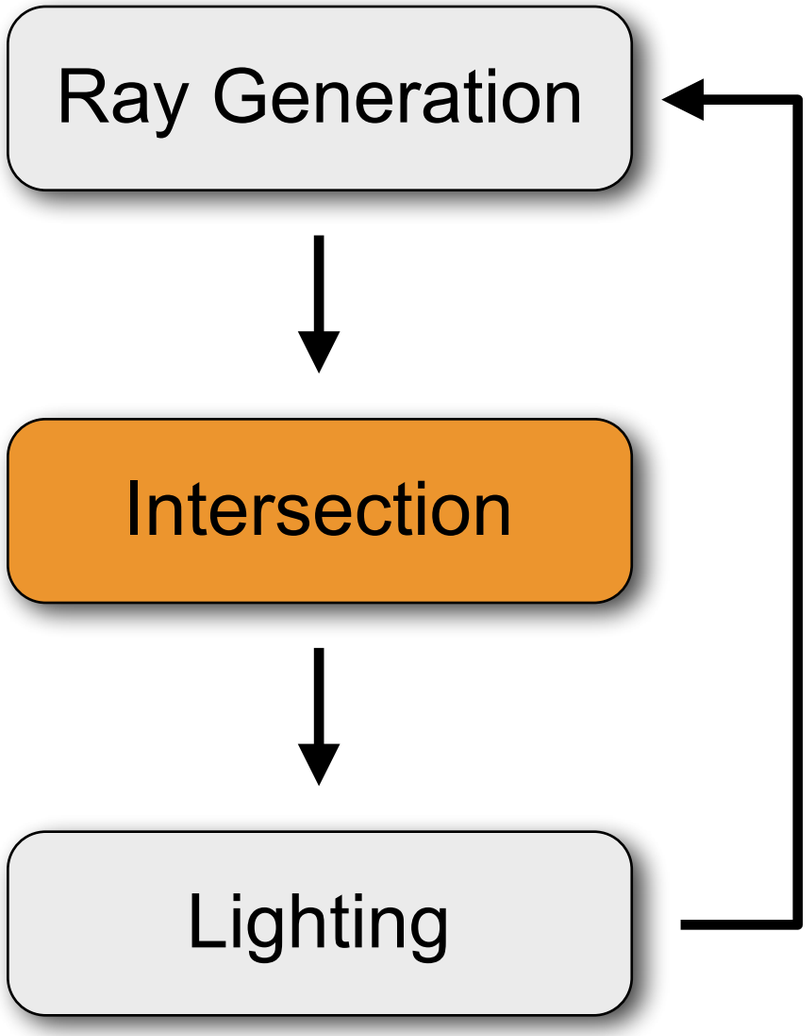


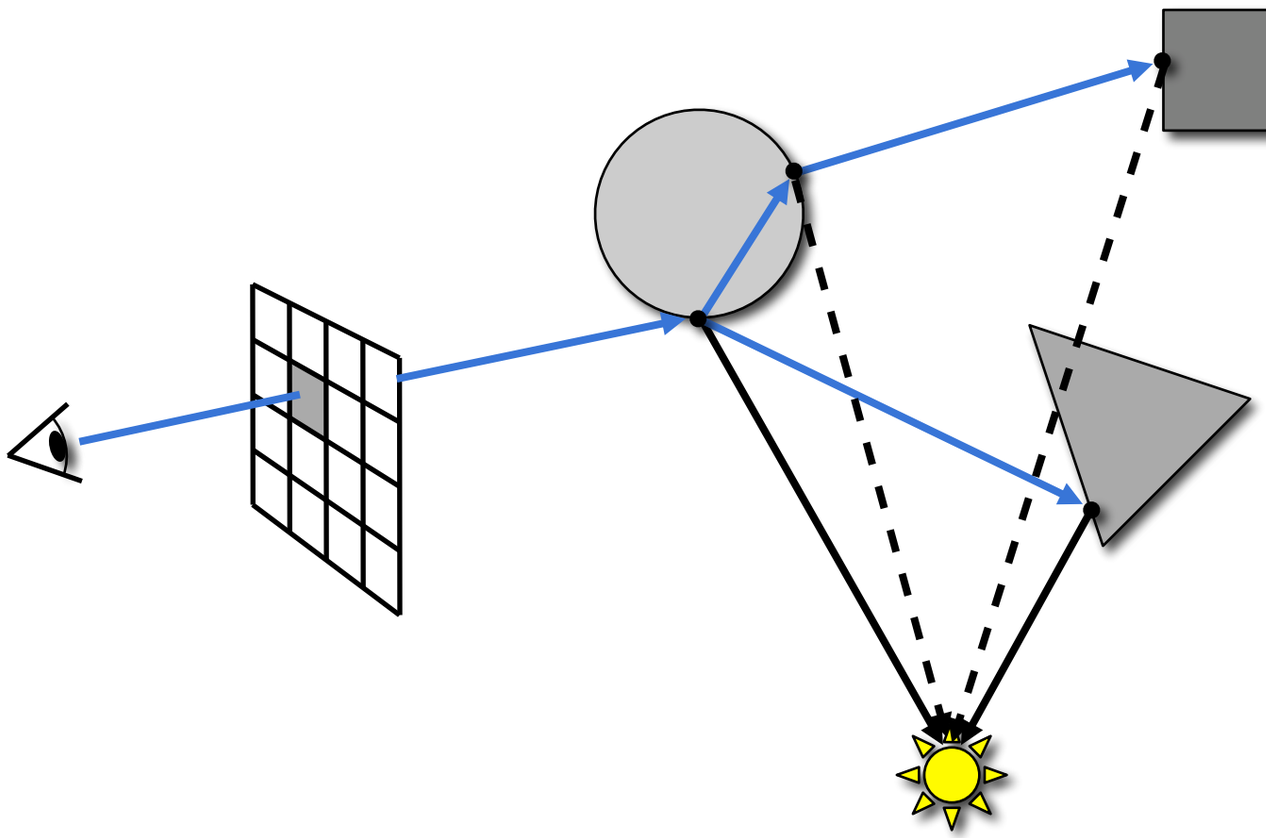
Primary Rays

- Shoot a primary ray through each pixel
- Parameters
  - image resolution
  - eye point (ray origin)
  - origin and axes of image
  - or: opening angle & projective camera transform



Ray-Surface Intersections





Ray-Surface Intersections

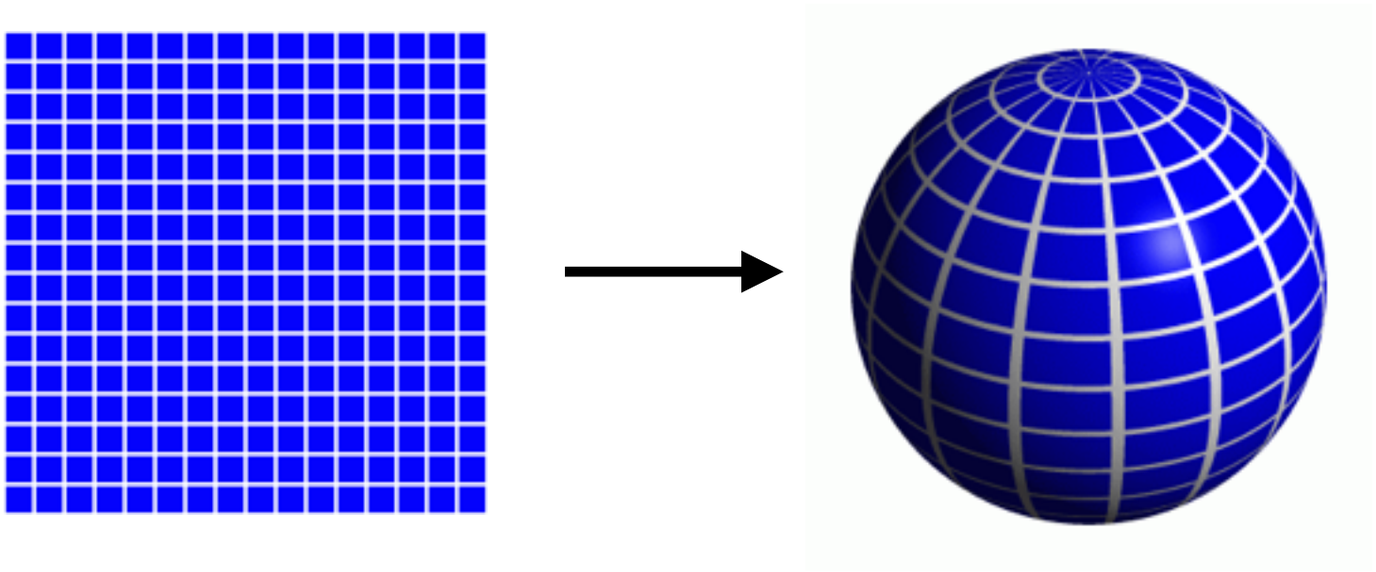
- Surface primitives
  - spheres
  - planes
  - triangles
  - etc.

Think hard!

- How to mathematically represent a sphere?
- How to intersect ray and sphere?

Unit Sphere (explicit)

$$\begin{pmatrix} \phi \\ \theta \end{pmatrix} \mapsto \begin{pmatrix} \cos\phi\cos\theta \\ \sin\phi\cos\theta \\ \sin\theta \end{pmatrix}$$



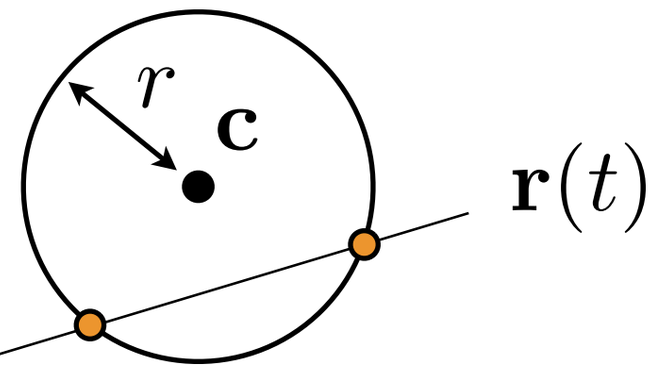
Ray-Sphere Intersection

- Sphere equation (explicit)

$$(\phi, \theta) \mapsto \underbrace{\mathbf{c}}_{\text{center}} + \underbrace{r}_{\text{radius}} \cdot \begin{pmatrix} \cos\phi\cos\theta \\ \sin\phi\cos\theta \\ \sin\theta \end{pmatrix}$$

- Ray point and sphere point have to coincide

$$\mathbf{o} + t\mathbf{d} = \mathbf{c} + r \cdot \begin{pmatrix} \cos\phi\cos\theta \\ \sin\phi\cos\theta \\ \sin\theta \end{pmatrix}$$



Solve for t, φ, θ

complicated nonlinear equation!

Ray-Sphere Intersection

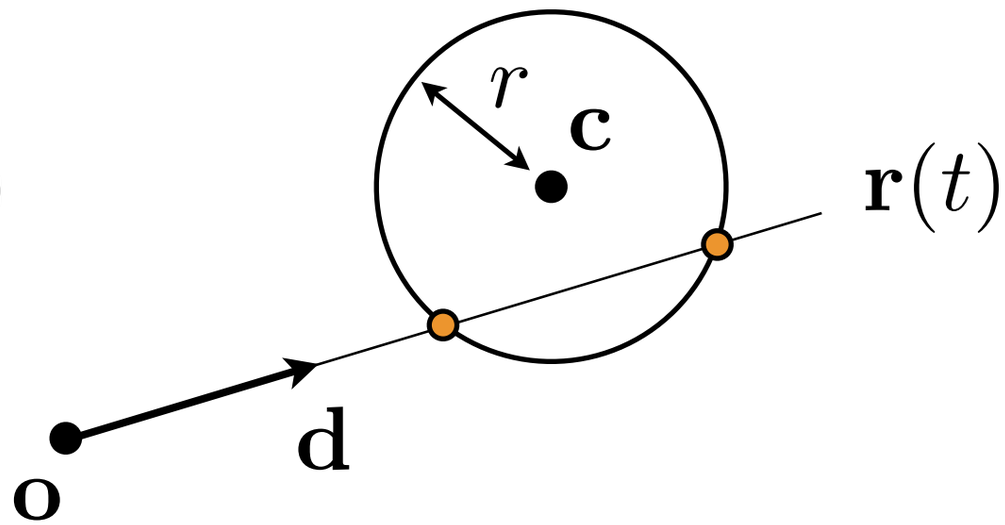
- Sphere equation (implicit)

$$\|\mathbf{x} - \mathbf{c}\|^2 - r^2 = 0$$

point on sphere — center — radius

- Insert explicit ray equation and solve for *t*

$$\|\mathbf{o} + t\mathbf{d} - \mathbf{c}\|^2 - r^2 = 0$$



Ray-Plane Intersection

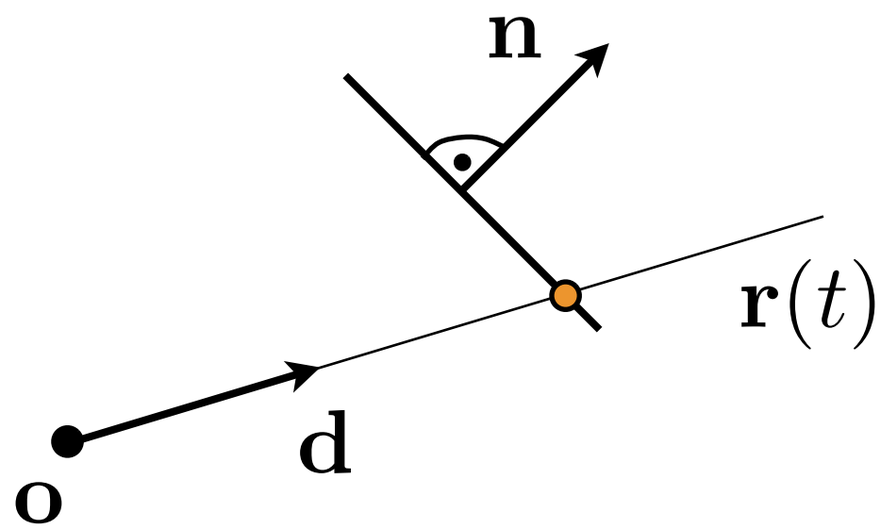
- Plane equation (implicit)

$$\mathbf{x}^T\mathbf{n} - d = 0$$

point on plane — plane normal — distance to origin

- Insert ray equation and solve for *t*

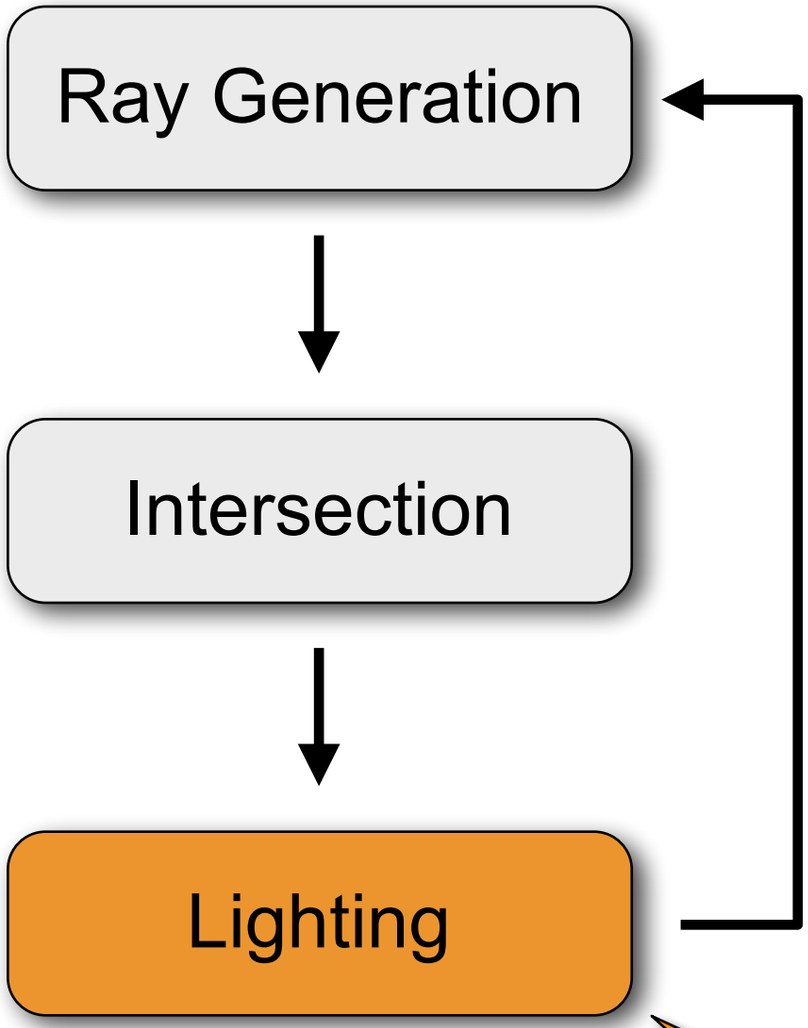
$$(\mathbf{o} + t\mathbf{d})^T\,\mathbf{n} - d = 0$$

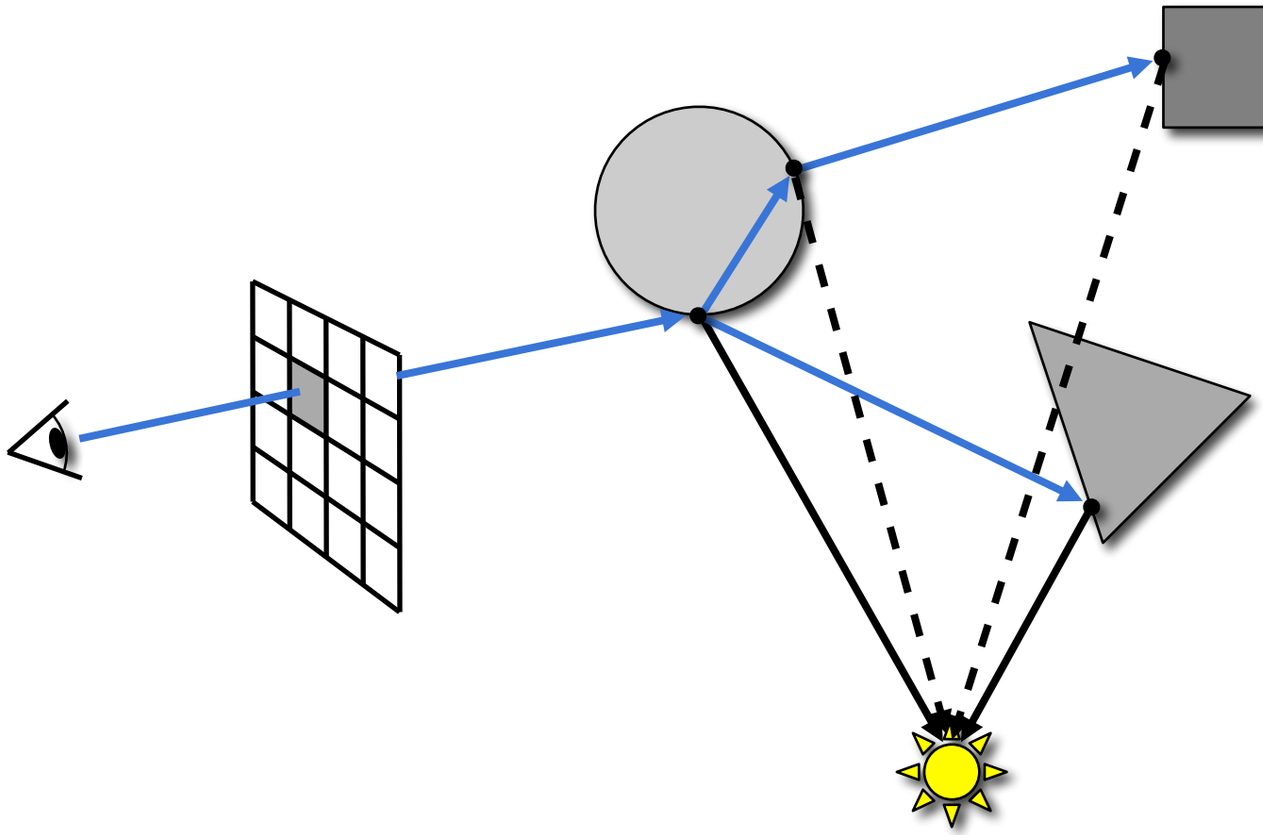


Ray-Surface Intersections

- Other primitives
  - cylinder, cone, paraboloid, hyperboloid
  - torus
  - triangle meshes, polygonal meshes
  - etc.

Lighting

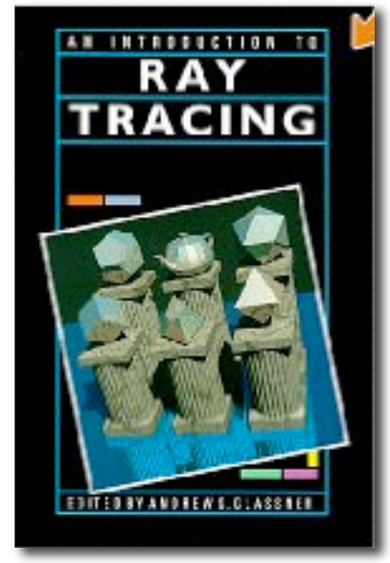




next week...

Literature

- Glassner: ***An Introduction to Ray Tracing***, Academic Press, 1989.
  - Chapters 2 & 4



- Pharr, Humphreys: ***Physically Based Rendering***, Morgan Kaufmann, 2004.
  - Chapters 1-3