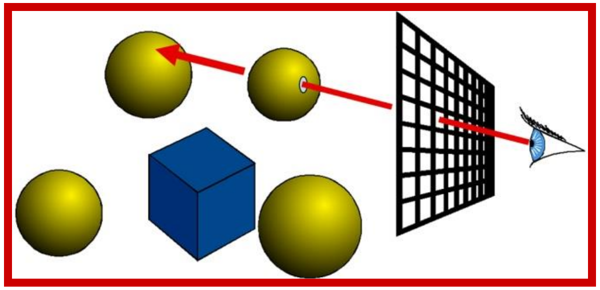
Ray Casting

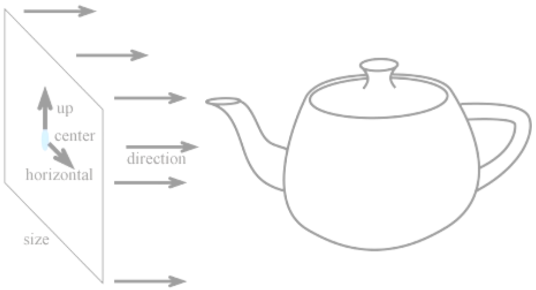
Ray Casting

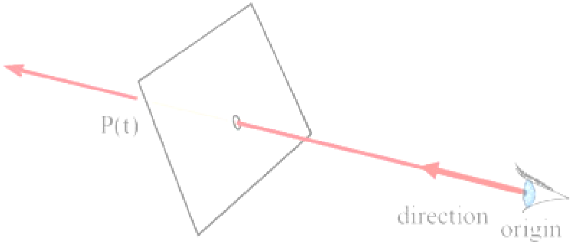
CLASS 1

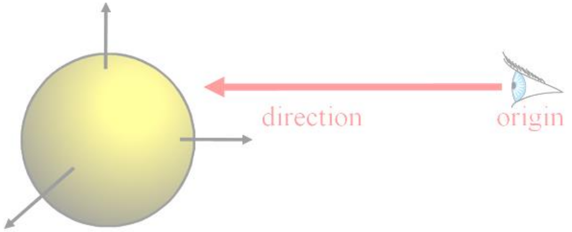
Overview of Today

- Ray Casting Basics
- Camera and Ray Generation
- Ray-Plane Intersection



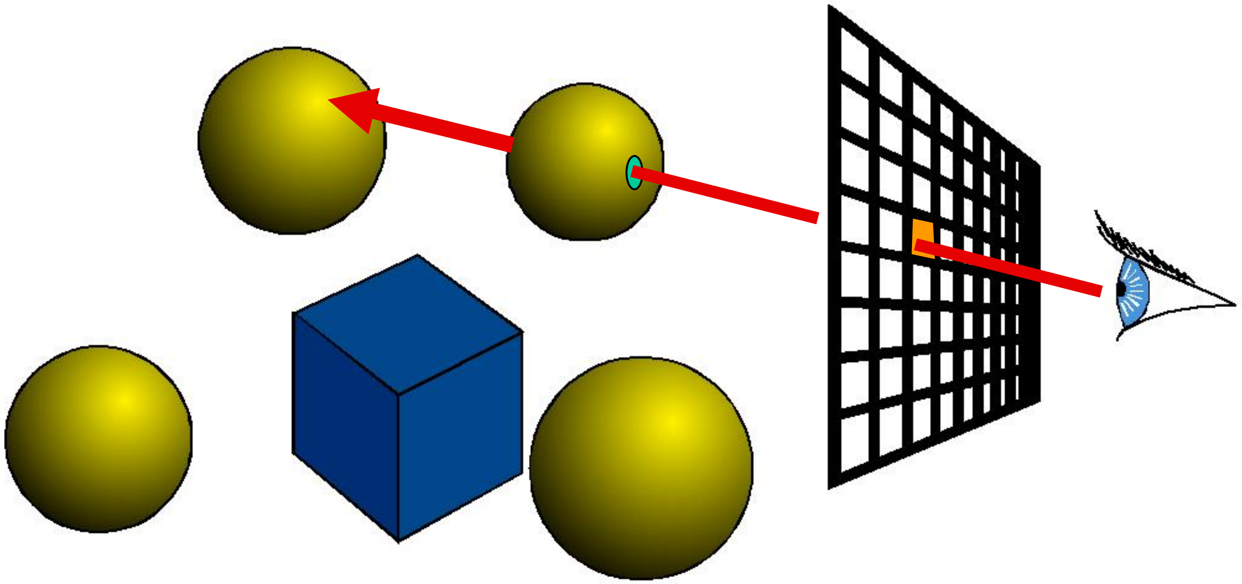






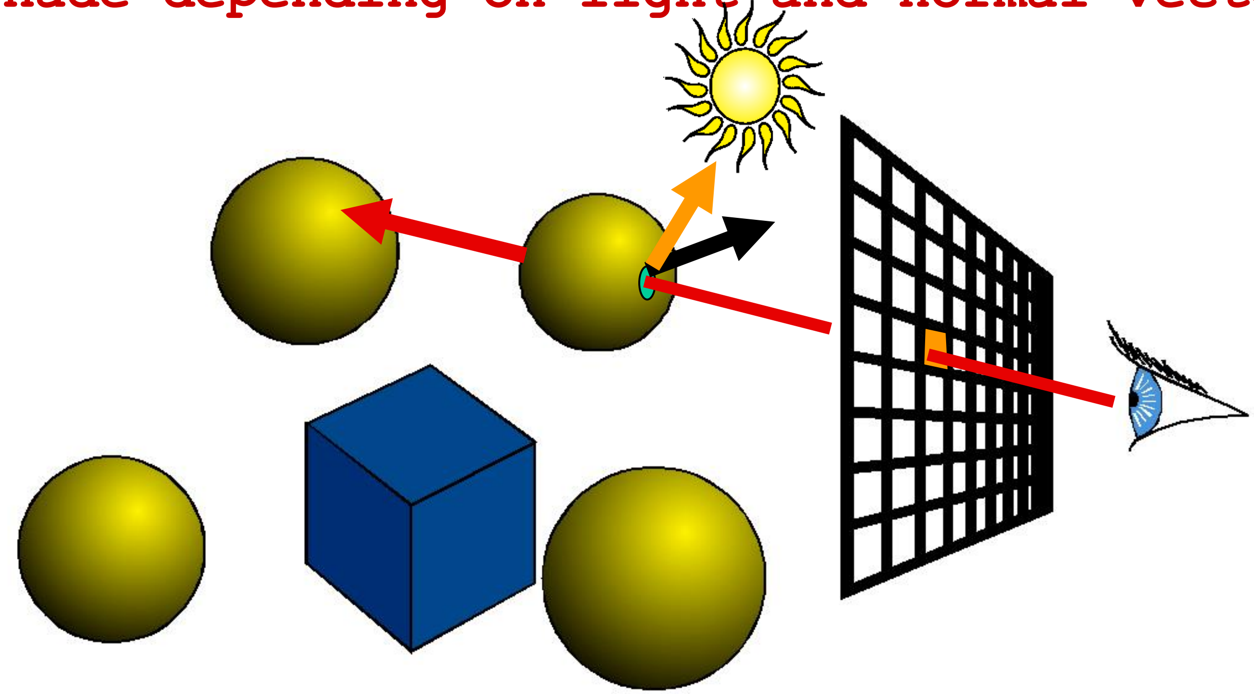
Ray Casting

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



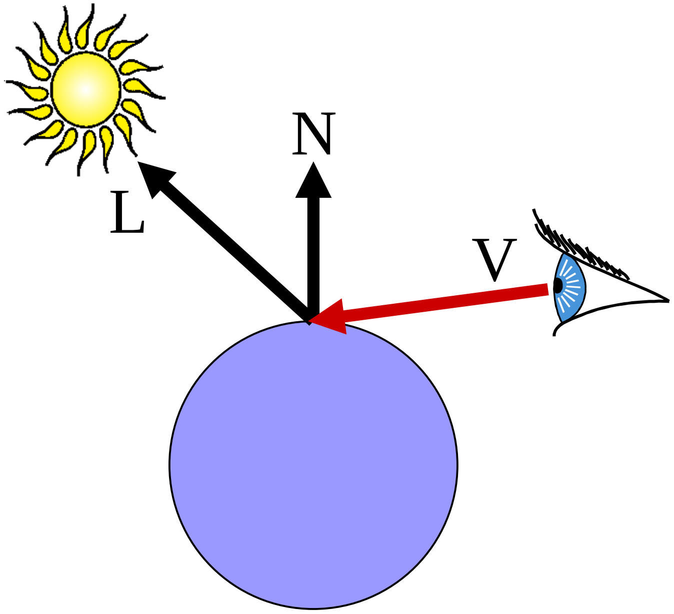
Shading

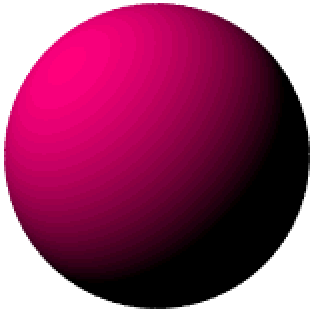
```
For every pixel
   Construct a ray from the eye
   For every object in the scene
        Find intersection with the ray
        Keep if closest
   Shade depending on light and normal vector
```



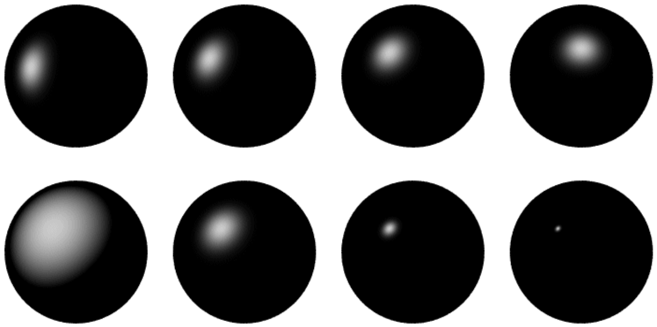
A Note on Shading

- Surface/Scene Characteristics:
  - surface normal
  - direction to light
  - viewpoint
- Material Properties
  - Diffuse (matte)
  - Specular (shiny)
  - …
- Much more soon!





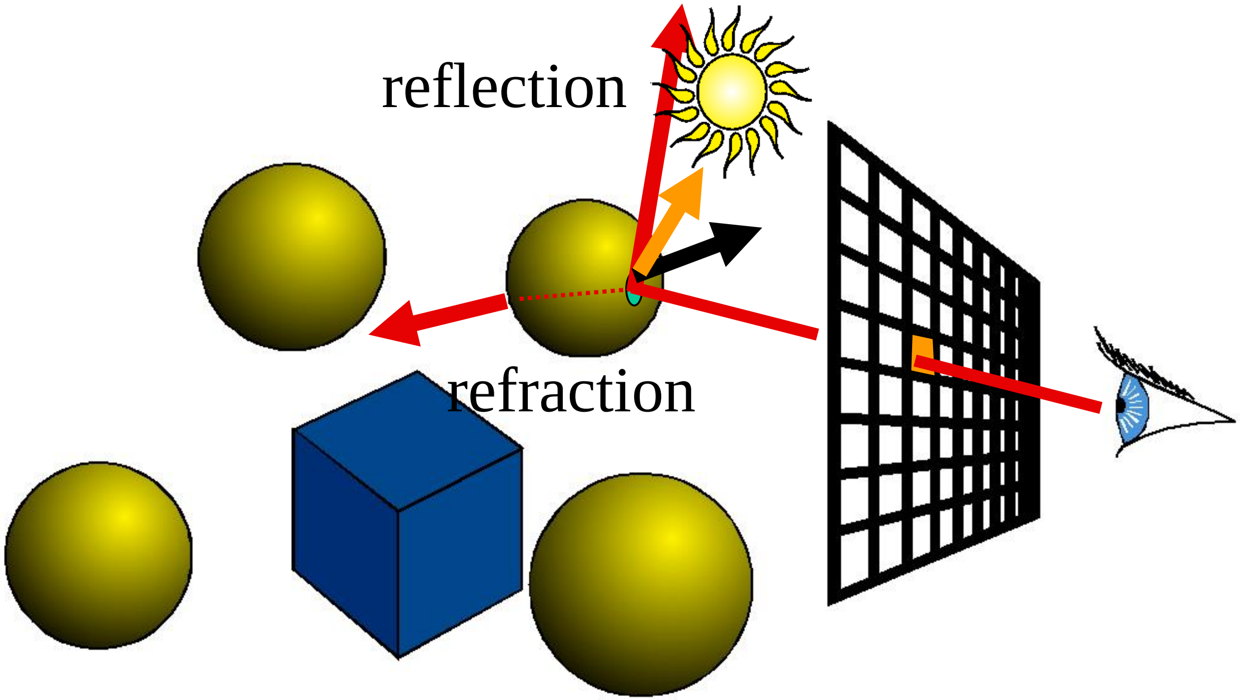
Diffuse sphere



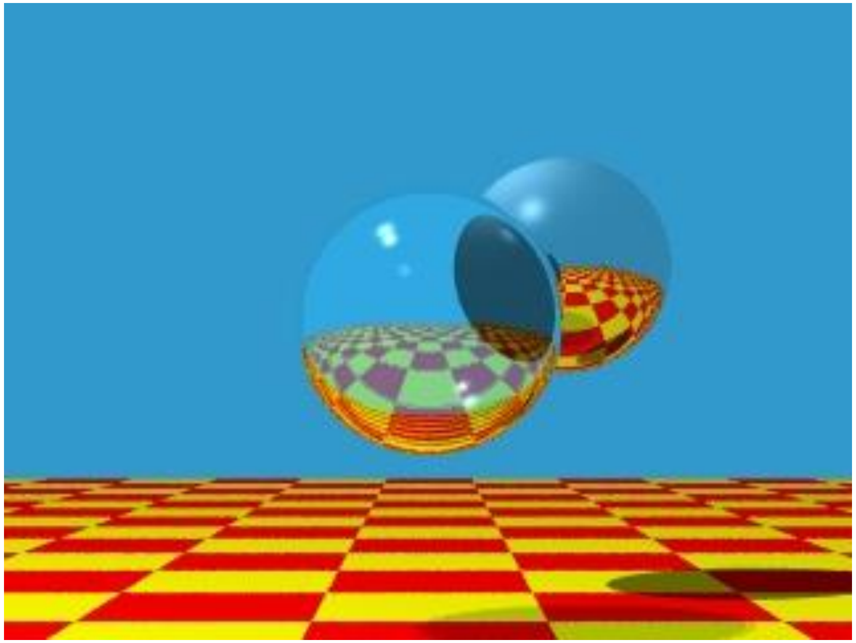
Specular spheres

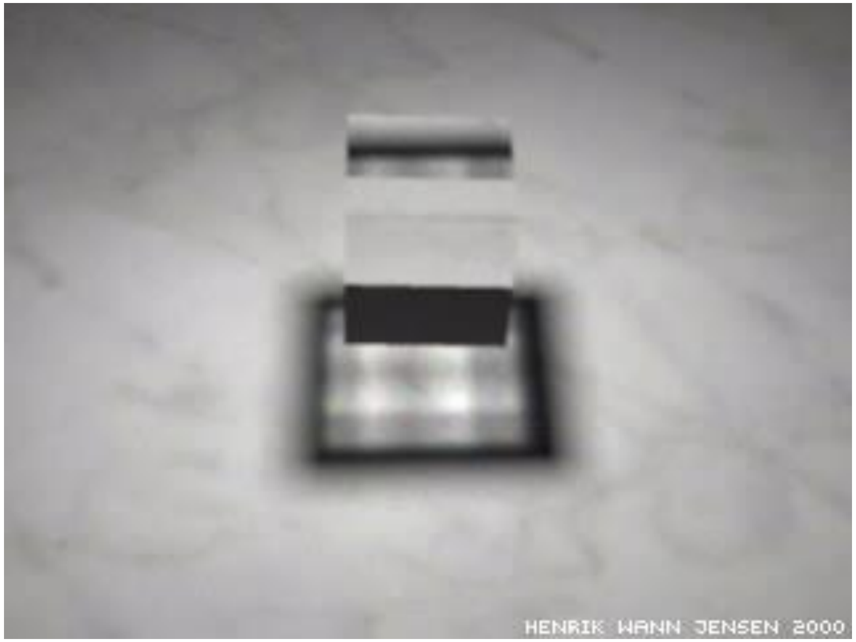
Ray Tracing

- Secondary rays (shadows, reflection, refraction)
- In a couple of weeks



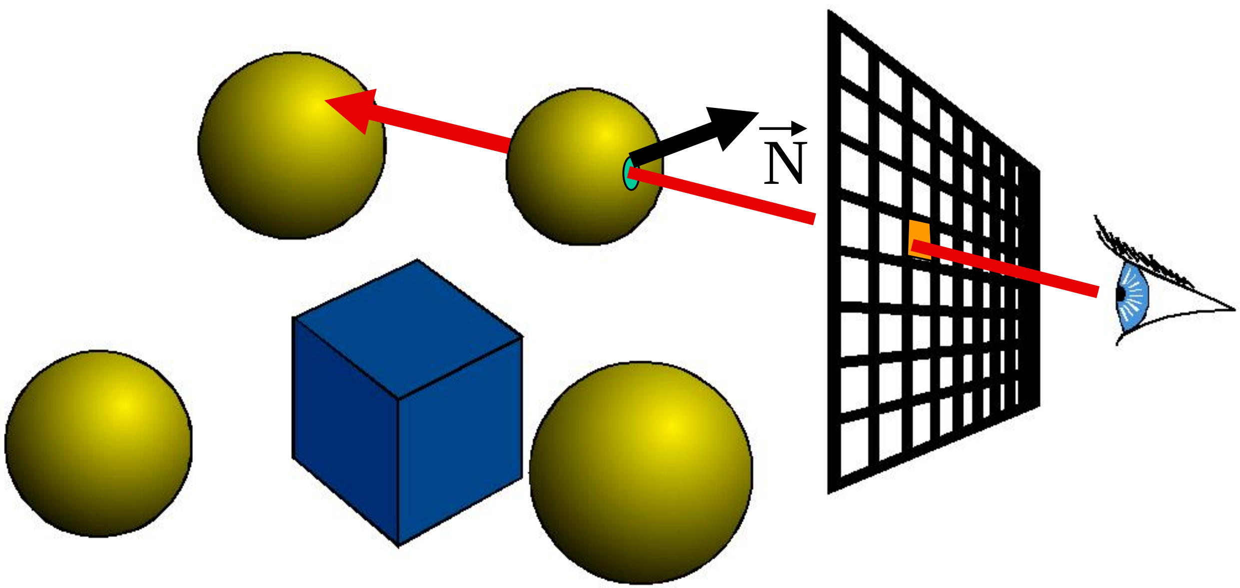
Ray Tracing





Ray Casting

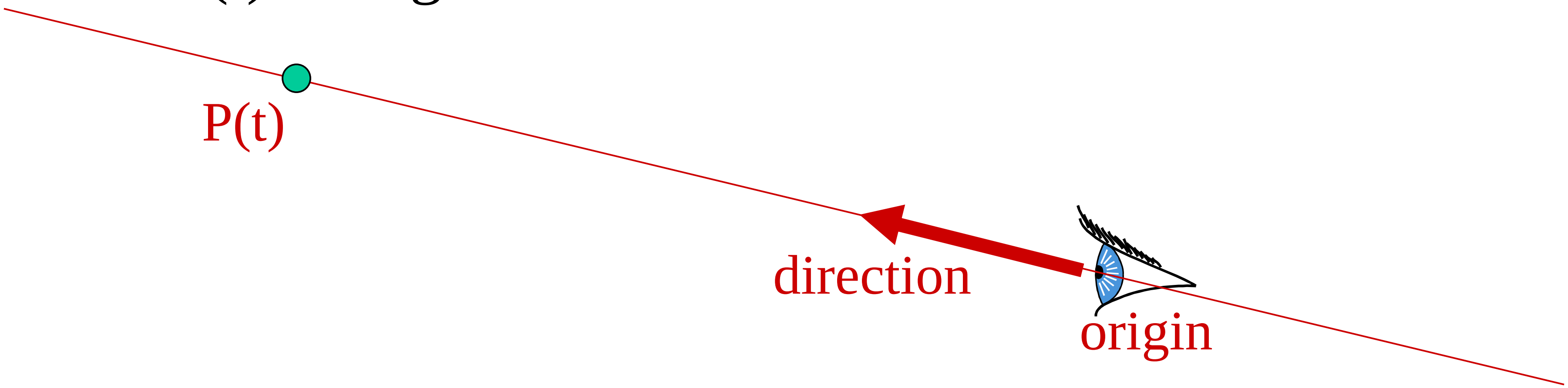
```
For every pixel
   Construct a ray from the eye
   For every object in the scene
        Find intersection with the ray
        Keep if closest
    Shade depending on light and normal vector
```



Finding the intersection and normal is the central part of ray casting

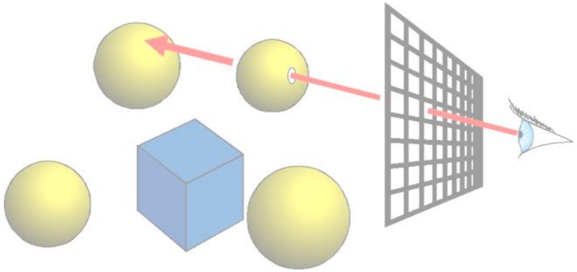
Ray Representation?

- Two vectors:
  - Origin
  - Direction (normalized is better)
- Parametric line
  - P(t) = origin + t * direction

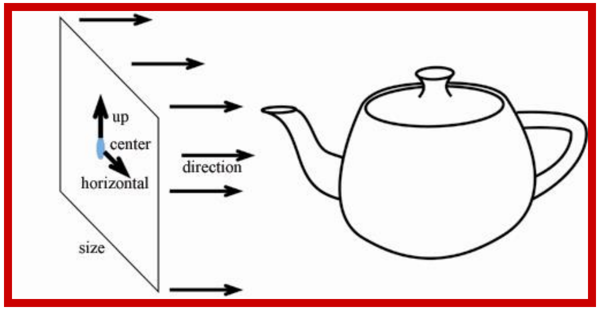


Overview of Today

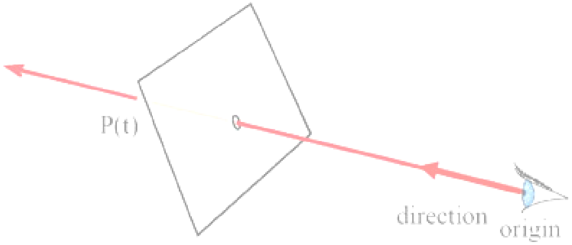
- Ray Casting Basics

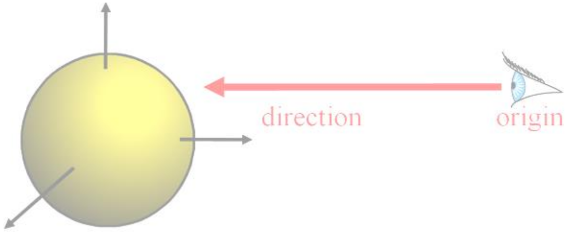


- Camera and Ray Generation



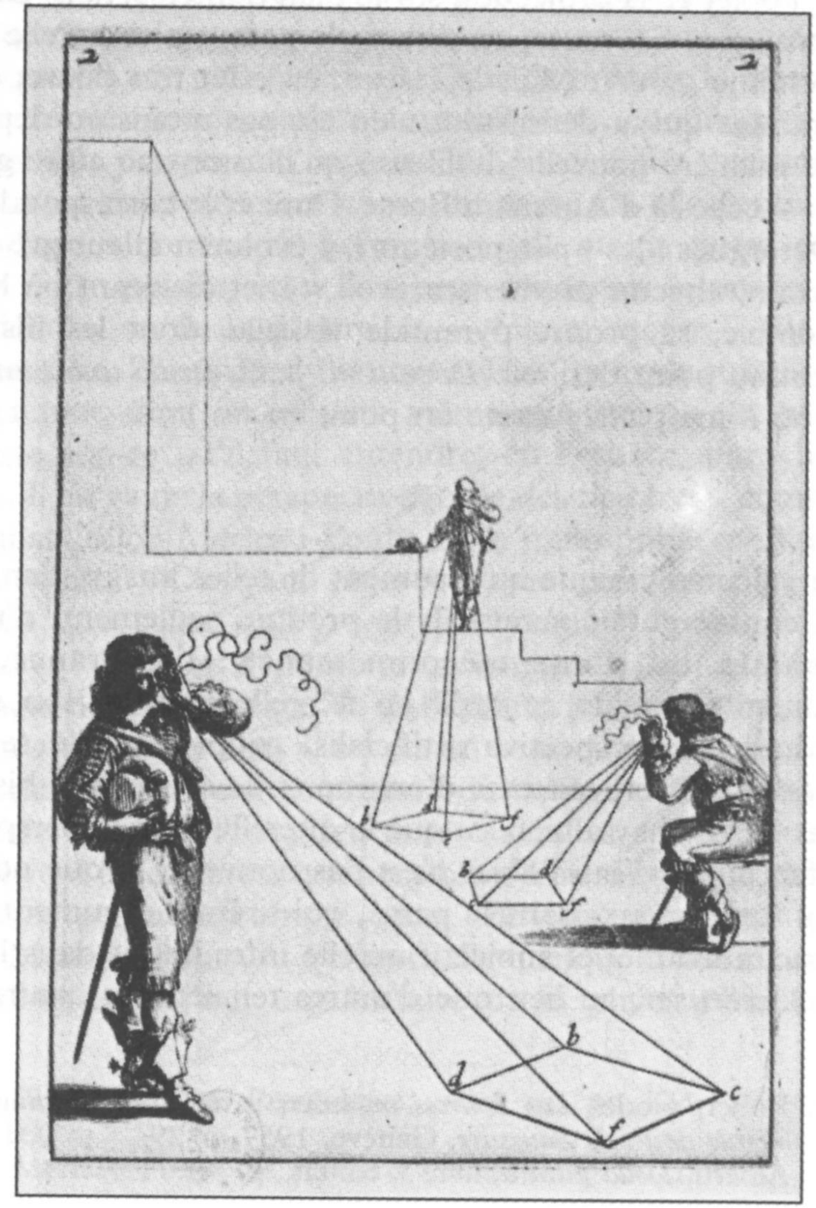
- Ray-Plane Intersection





Cameras

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

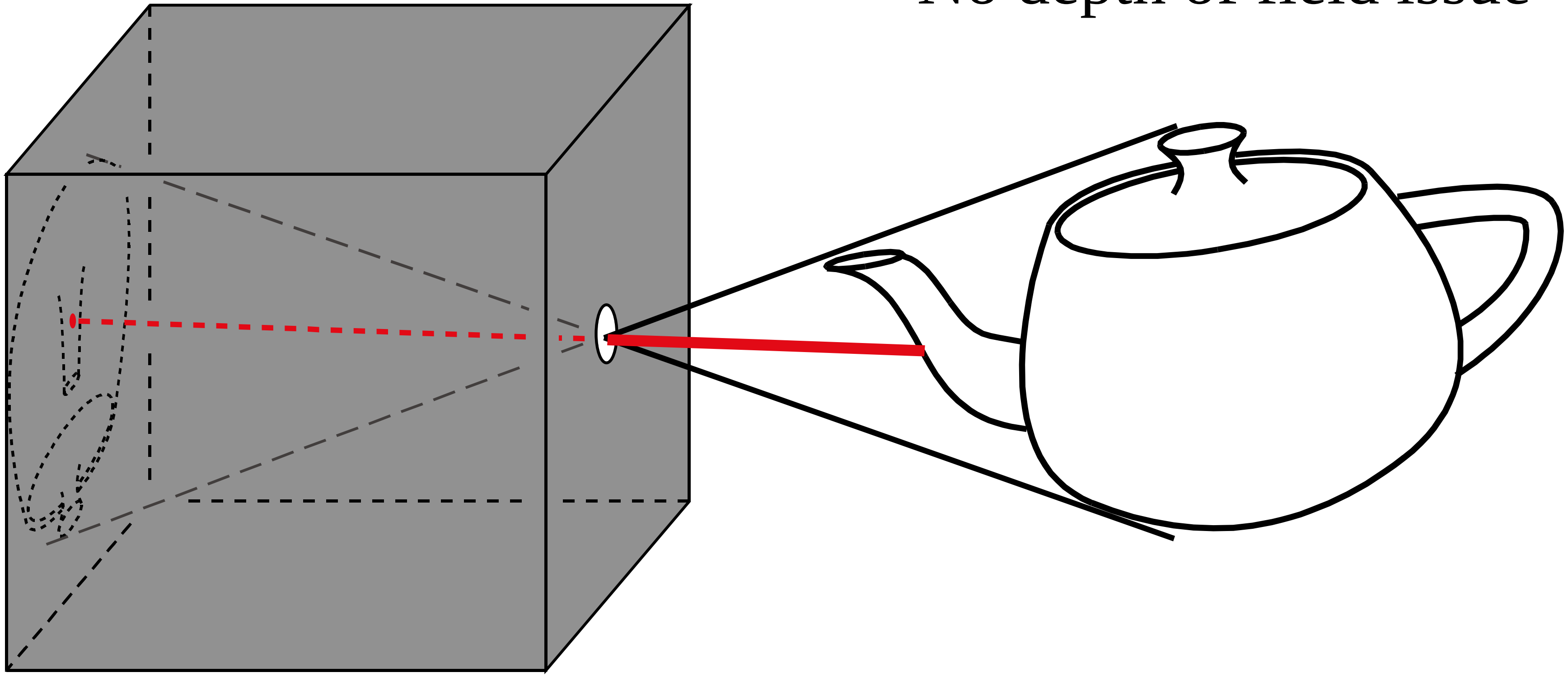


Abraham Bosse, *Les Perspecteurs*. Gravure extraite de la *Manière*

Pinhole Camera

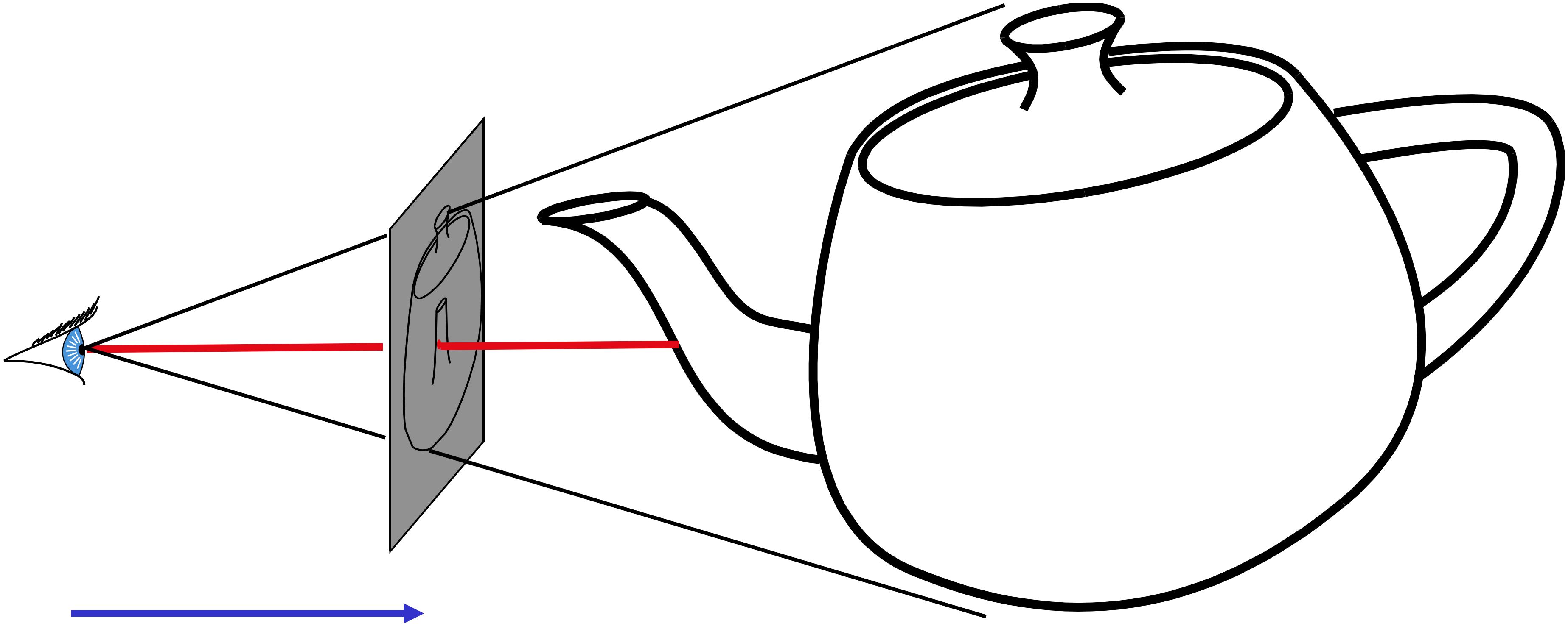
- Box with a tiny hole
- Inverted image
- Similar triangles

- Perfect image if hole infinitely small
- Pure geometric optics
- No depth of field issue



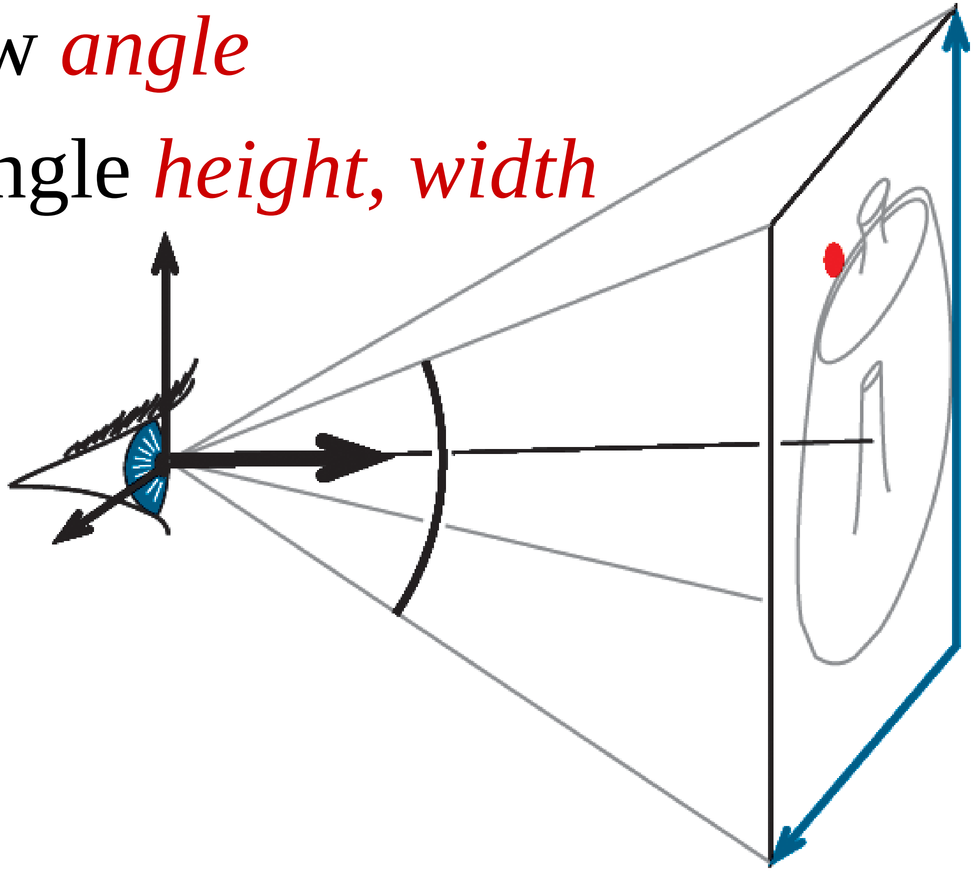
Simplified Pinhole Camera

- Eye-image pyramid (frustum)
- Note that the distance/size of image are arbitrary

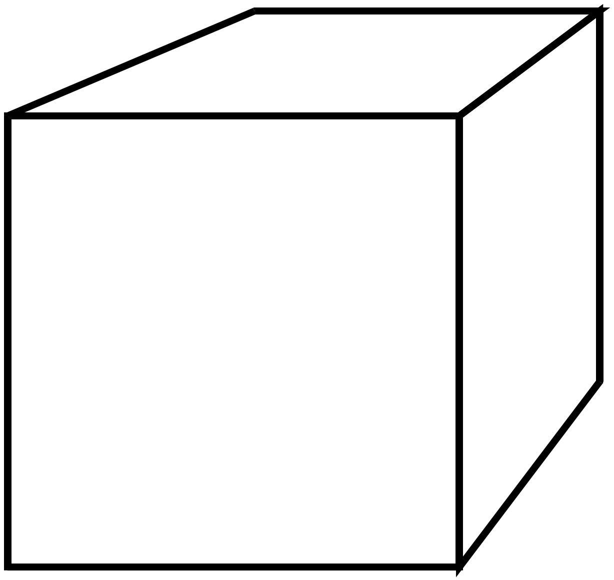


Camera Description?

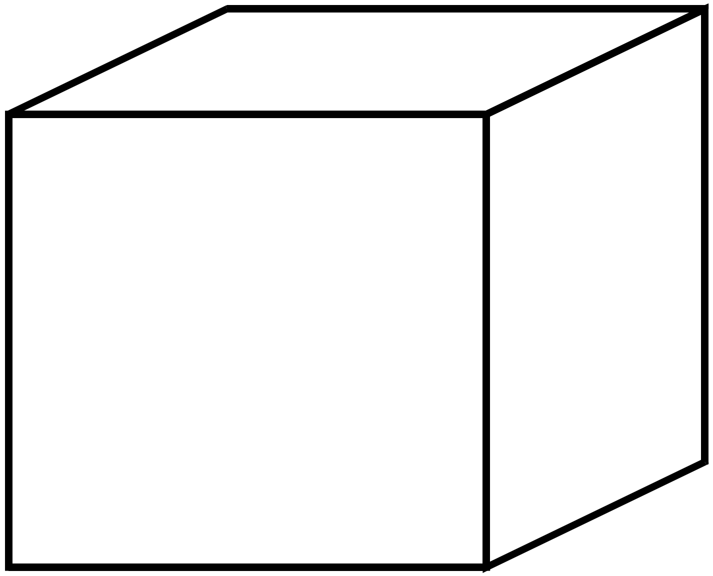
- Eye point *e (center)*
- Orthobasis *u, v, w (horizontal, up, -direction)*
- Field of view *angle*
- Image rectangle *height, width*



Perspective vs. Orthographic



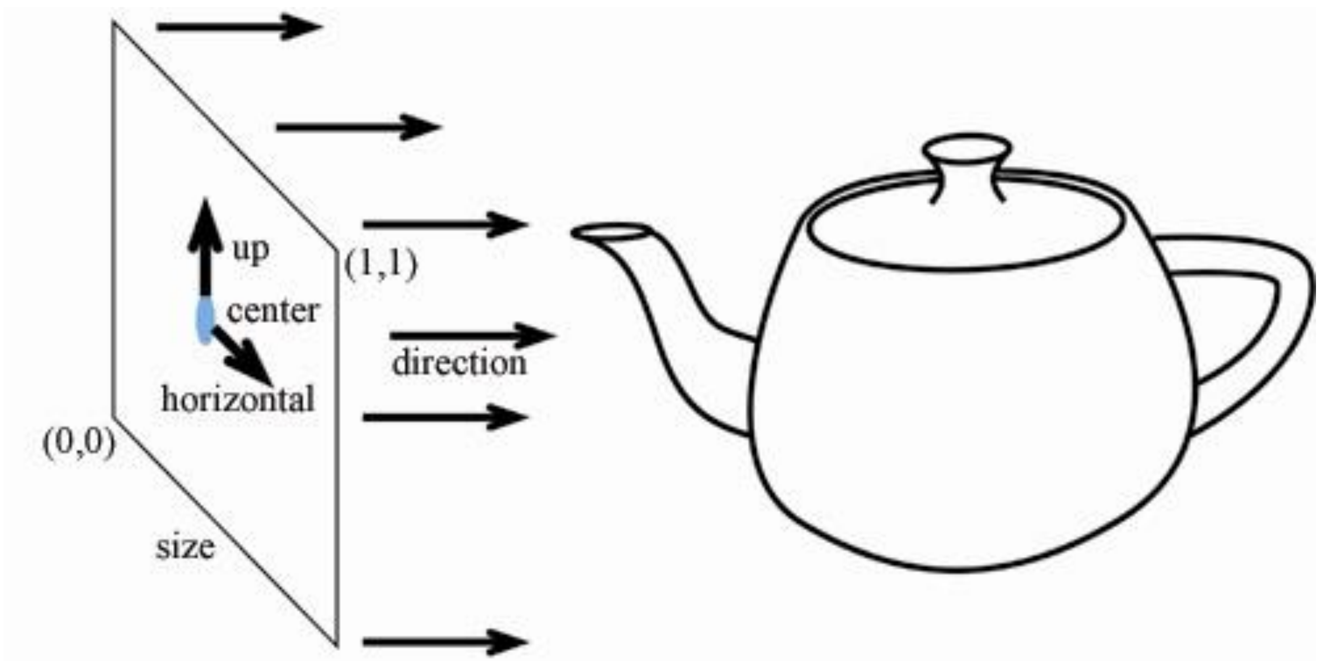
perspective



orthographic

- Parallel projection
- No foreshortening
- No vanishing point

Orthographic Camera



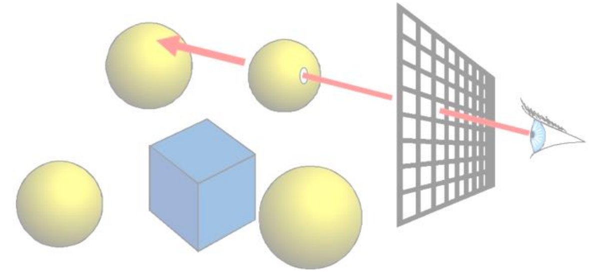
- Ray Generation?
  - Origin = center + (x-0.5)*size*horizontal + (y-0.5)*size*up
  - Direction is constant

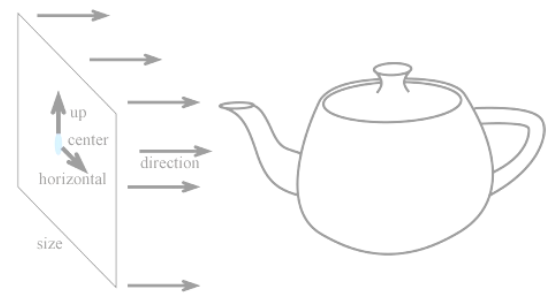
Other Weird Cameras

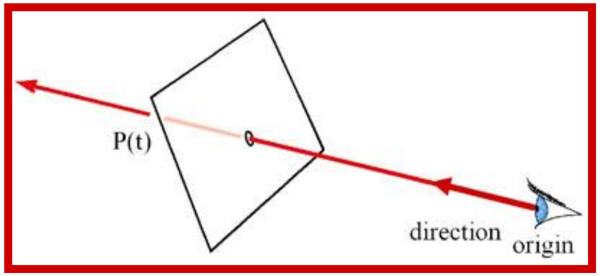
- E.g. fish eye, omnimax, panorama

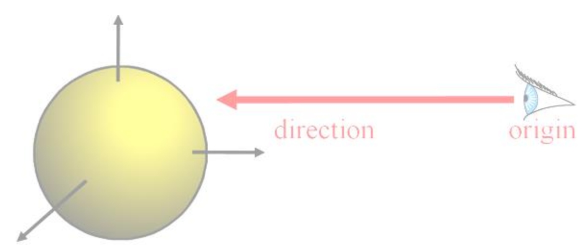
Overview of Today

- Ray Casting Basics
- Camera and Ray Generation
- Ray-Plane Intersection





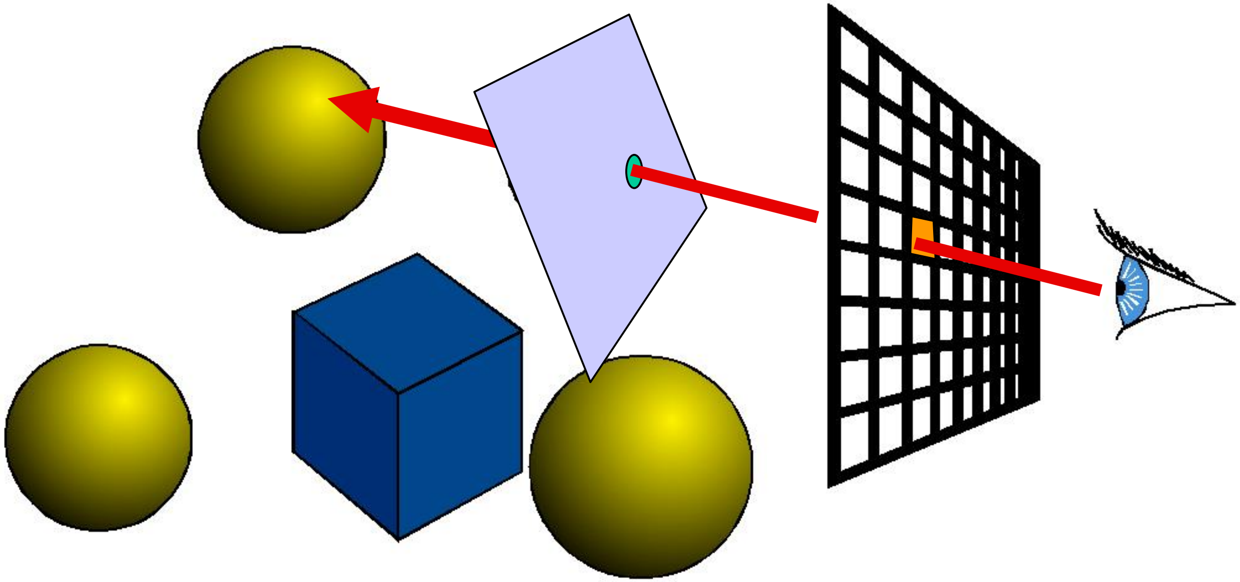




Ray Casting

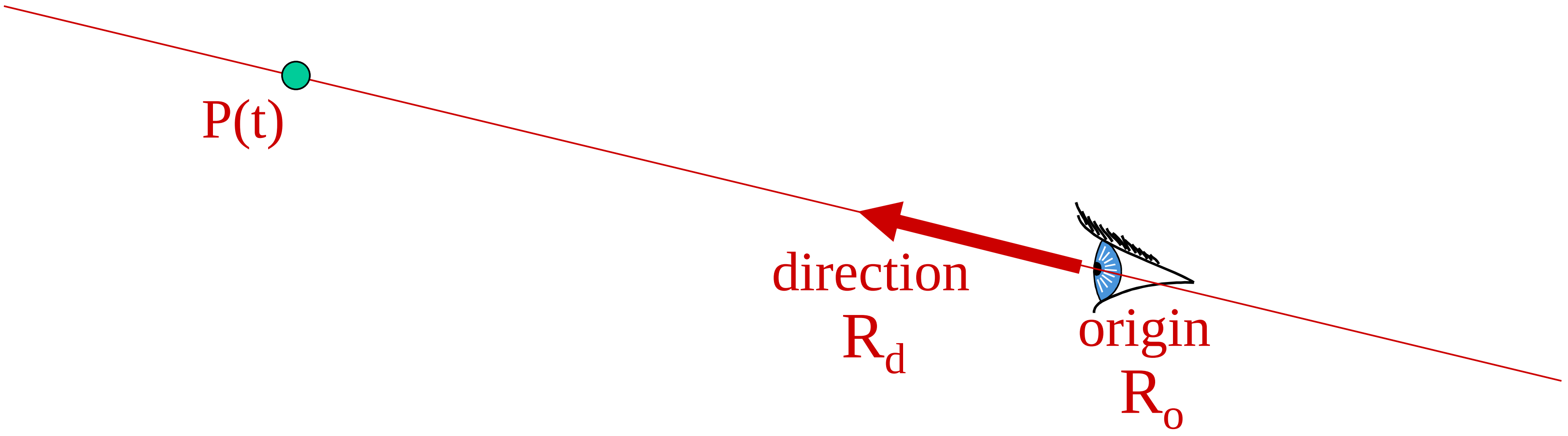
```
For every pixel
  Construct a ray from the eye
  For every object in the scene
      Find intersection with the ray
      Keep if closest
```

First we will study ray-plane intersection



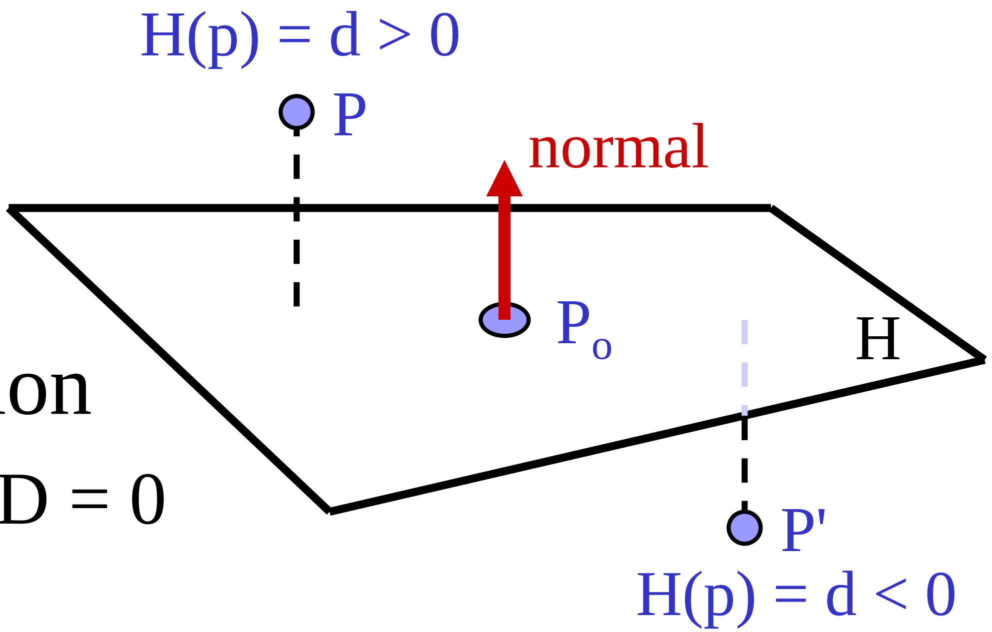
Recall: Ray Representation

- Parametric line
- $P(t) = R_o + t * R_d$
- Explicit representation



3D Plane Representation?

- Plane defined by
  - $P_o = (x,y,z)$
  - $n = (A,B,C)$
- Implicit plane equation
  - $H(P) = Ax+By+Cz+D = 0$
    $= n \cdot P + D = 0$
- Point-Plane distance?
  - If n is normalized,
    distance to plane, $d = H(P)$
  - d is the *signed distance*!



Explicit vs. Implicit?

- Ray equation is explicit $P(t) = R_o + t * R_d$
  - Parametric
  - Generates points
  - Hard to verify that a point is on the ray
- Plane equation is implicit $H(P) = n \cdot P + D = 0$
  - Solution of an equation
  - Does not generate points
  - Verifies that a point is on the plane
- Exercise: Explicit plane and implicit ray

Ray-Plane Intersection

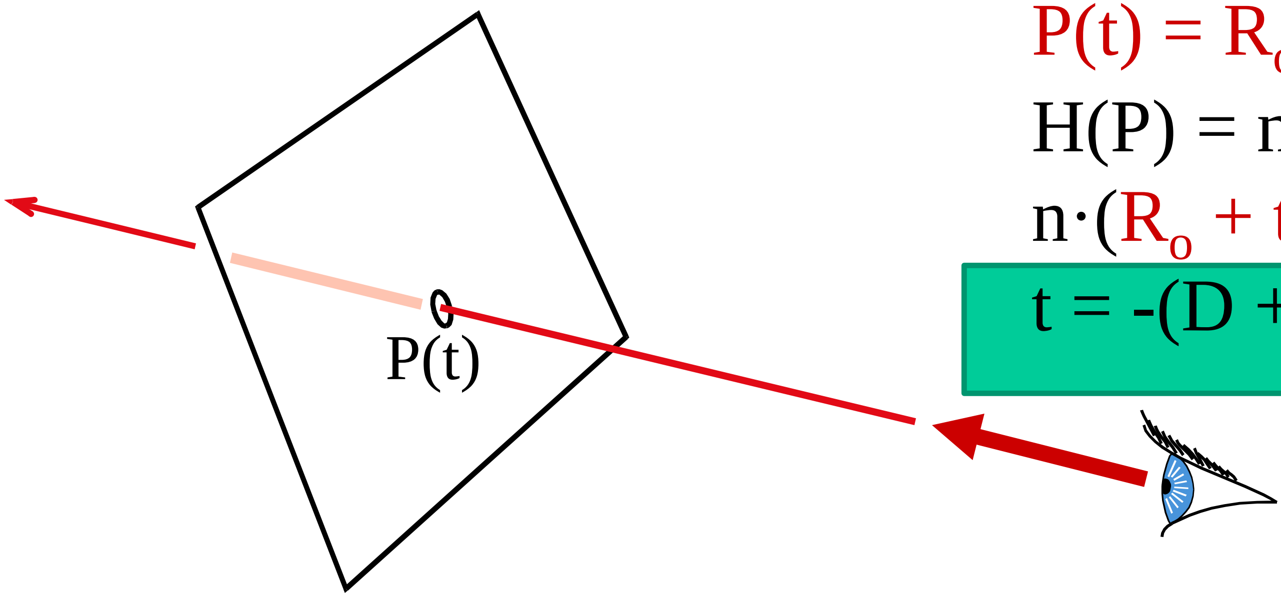
- Intersection means both are satisfied
- So, insert explicit equation of ray into implicit equation of plane & solve for t

$$P(t) = R_o + t * R_d$$

$$H(P) = n \cdot P + D = 0$$

$$n \cdot (R_o + t * R_d) + D = 0$$

$$\boxed{t = -(D + n \cdot R_o) / n \cdot R_d}$$



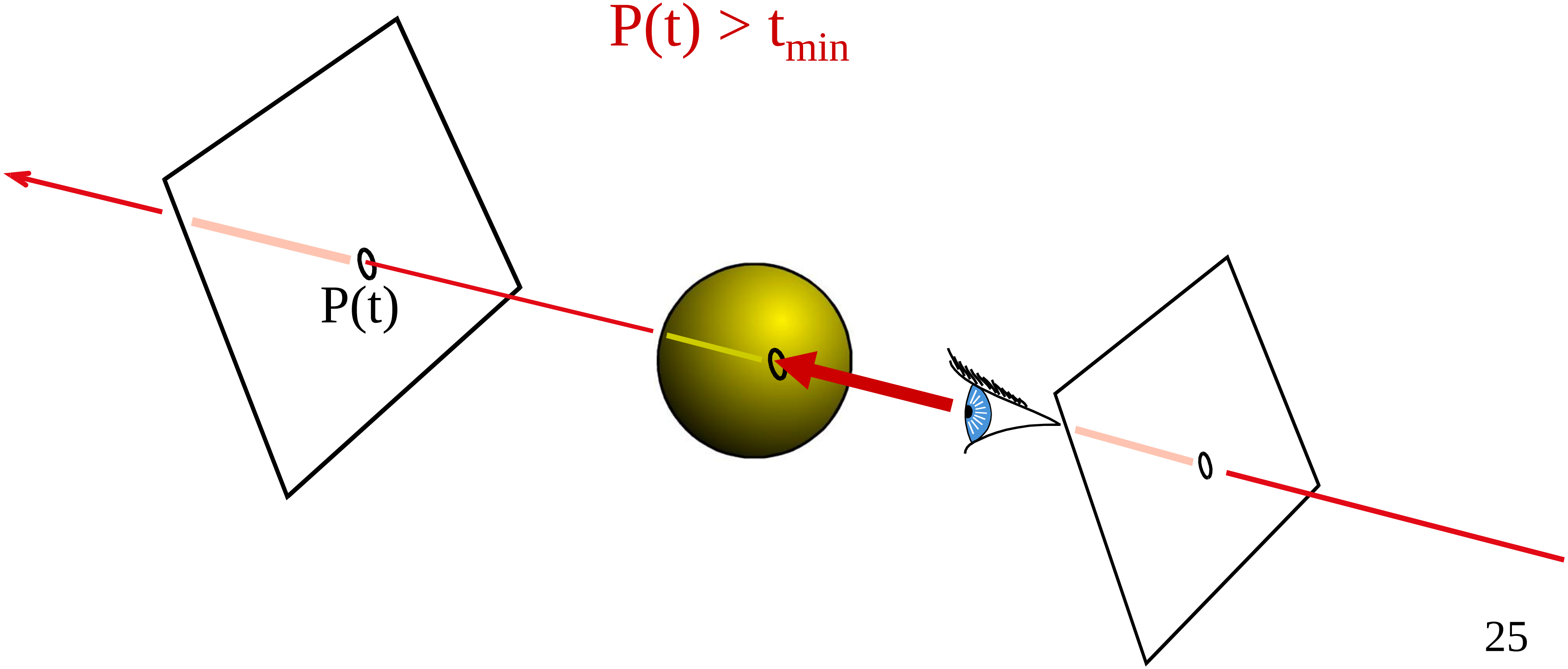
Additional Housekeeping

- Verify that intersection is closer than previous

$$P(t) < t_{current}$$

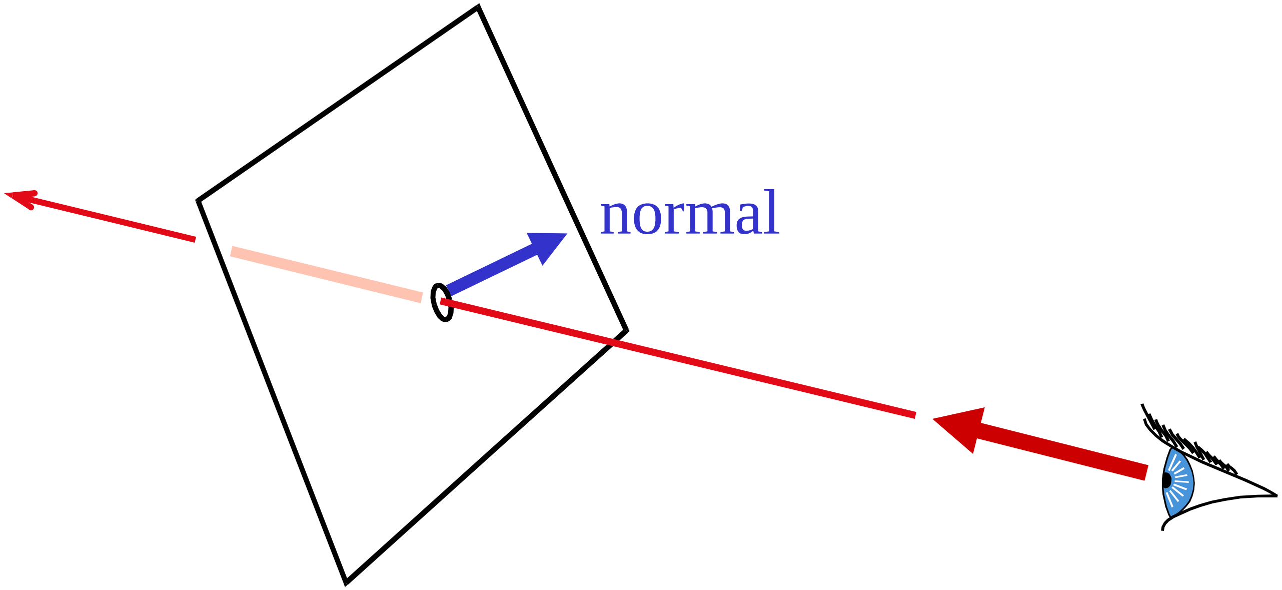
- Verify that it is not out of range (behind eye)

$$P(t) > t_{min}$$



Normal

- For shading
  - diffuse: dot product between light and normal
- Normal is constant



A moment of mathematical beauty

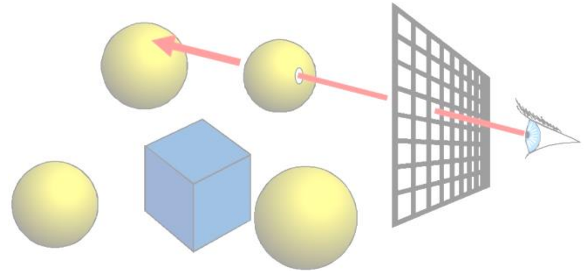
- Duality : points and planes are dual when you use homogeneous coordinates
- Point (x, y, z, 1)
- Plane (A, B, C, D)
- Plane equation ➔ dot product
- You can map planes to points and points to planes in a dual space.
- Lots of cool equivalences
  – e.g. intersection of 3 planes define a point
  – ➔ 3 points define a plane!

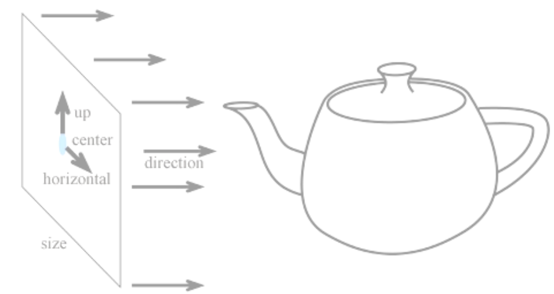
Ray Casting

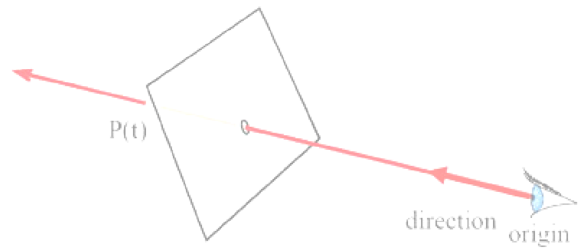
CLASS 2

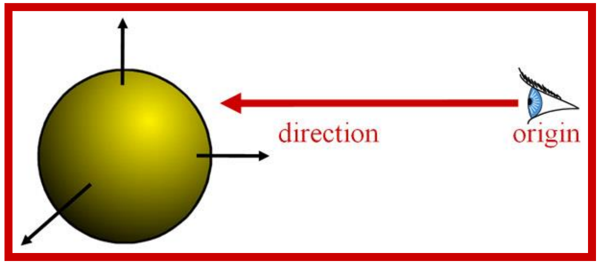
Overview of Today

- Ray-Sphere Intersection
- Ray-Triangle Intersection
- Implementing CSG



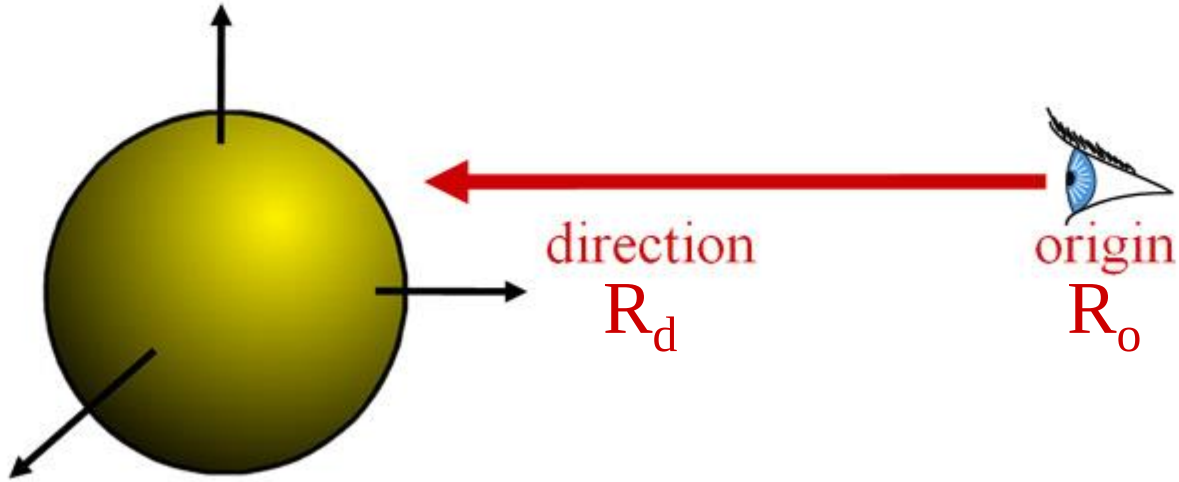






Sphere Representation?

- Implicit sphere equation
  - Assume centered at origin (easy to translate)
  - $H(P) = P \cdot P - r^2 = 0$



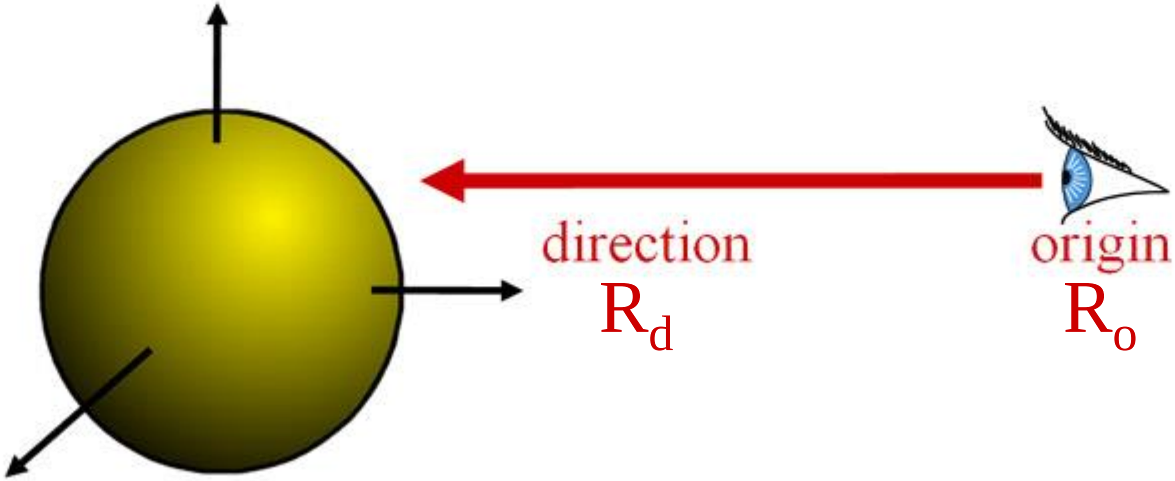
Ray-Sphere Intersection

- Insert explicit equation of ray into implicit equation of sphere & solve for t

$$P(t) = R_o + t^*R_d \qquad H(P) = P \cdot P - r^2 = 0$$

$$(R_o + tR_d) \cdot (R_o + tR_d) - r^2 = 0$$

$$R_d \cdot R_d t^2 + 2R_d \cdot R_o t + R_o \cdot R_o - r^2 = 0$$

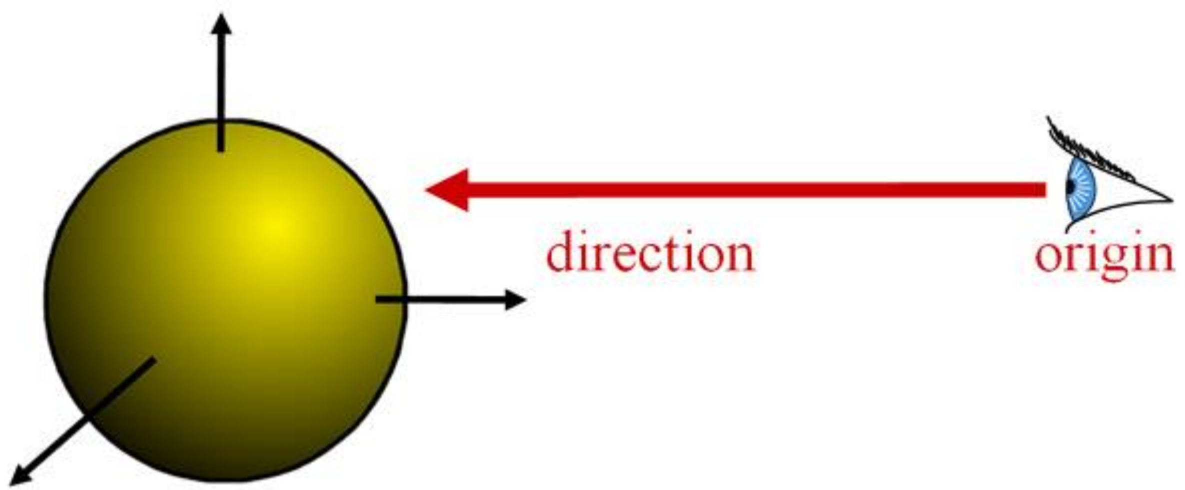


Ray-Sphere Intersection

- Quadratic: $at^2 + bt + c = 0$
  - $a = 1$ (remember, $\|R_d\| = 1$)
  - $b = 2R_d \cdot R_o$
  - $c = R_o \cdot R_o - r^2$
- with discriminant $$d = \sqrt{b^2 - 4ac}$$
- and solutions $$t_{\pm} = \frac{-b \pm d}{2a}$$

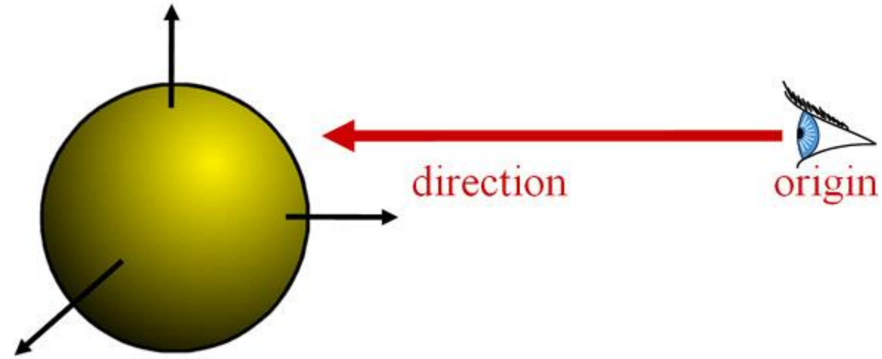
Ray-Sphere Intersection

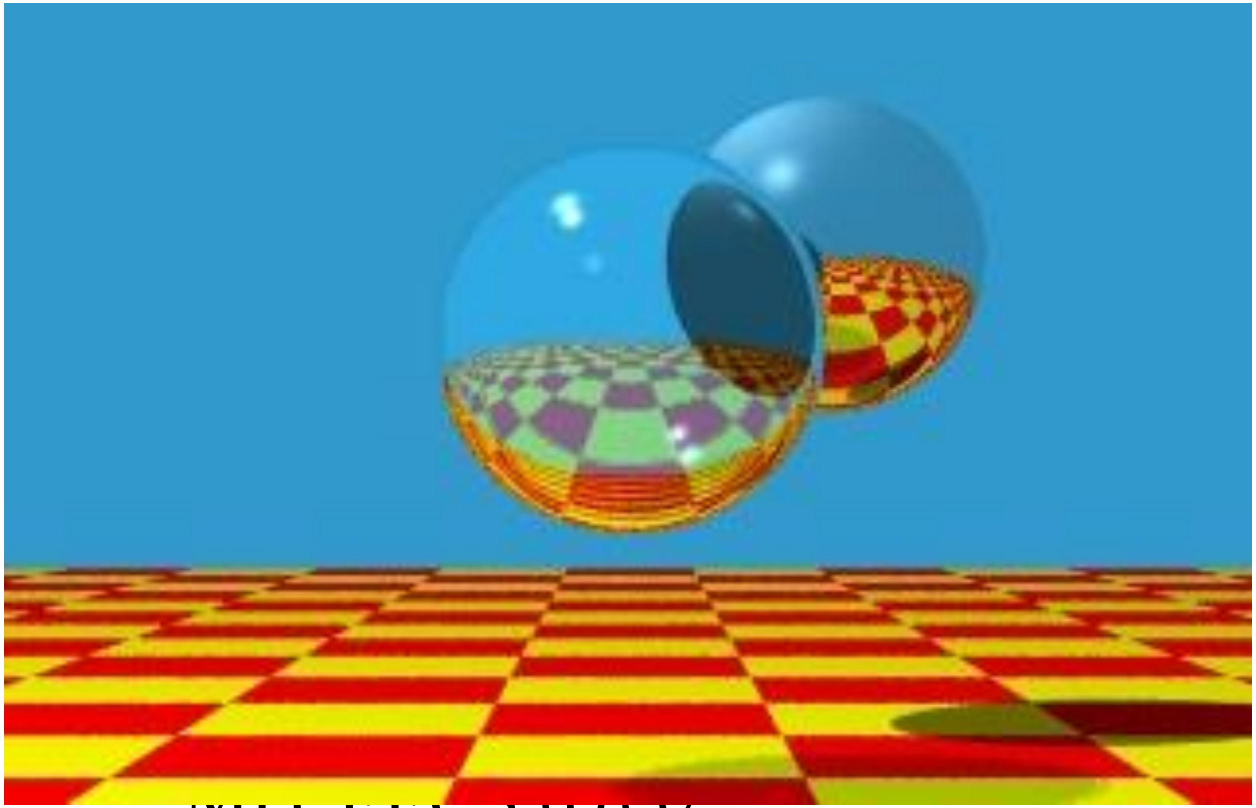
- 3 cases, depending on the sign of $b^2 - 4ac$
- What do these cases correspond to?
- Which root (t+ or t-) should you choose?
  - Closest positive! (usually t-)



Ray-Sphere Intersection

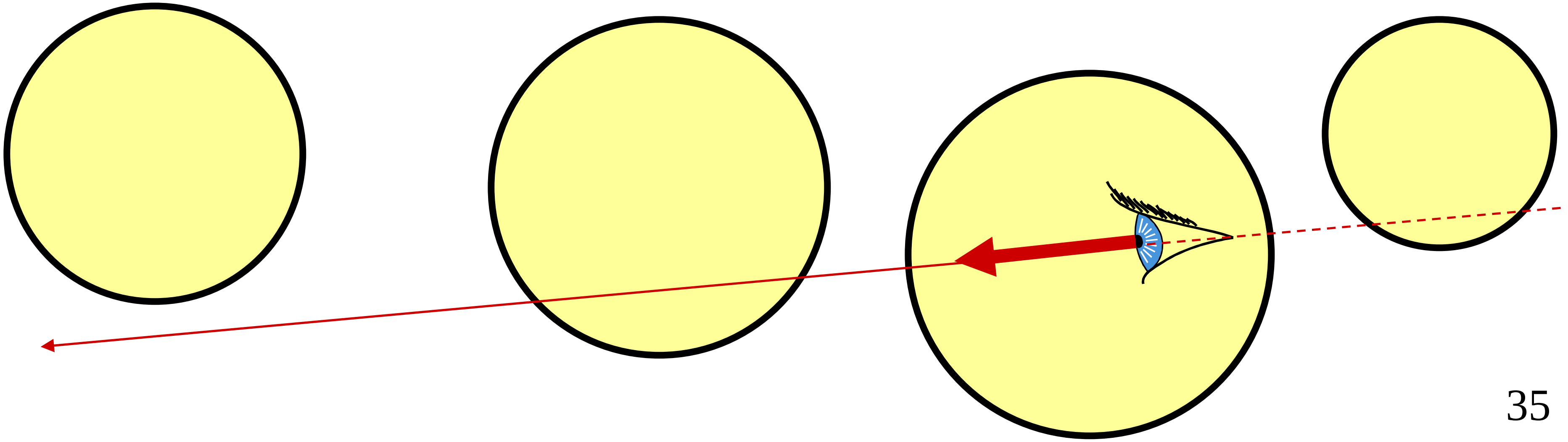
- It's so easy that all ray-tracing images have spheres!





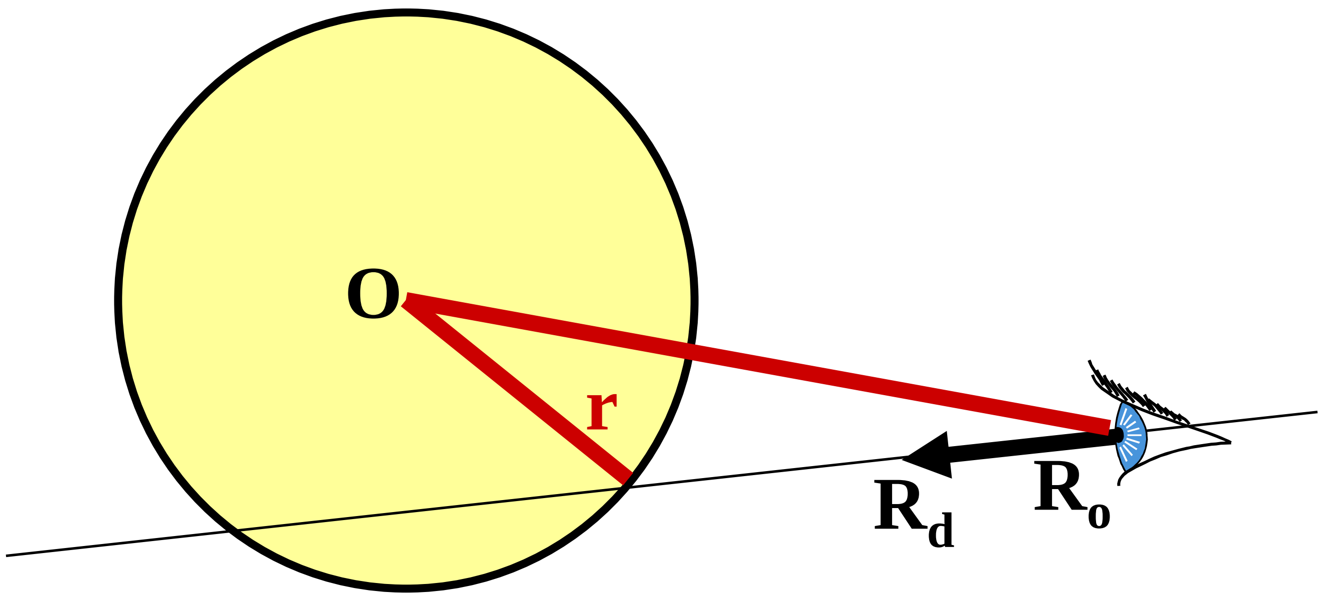
Geometric Ray-Sphere Intersection

- Shortcut / easy reject
- What geometric information is important?
  - Ray origin inside/outside sphere?
  - Closest point to ray from sphere origin?
  - Ray direction: pointing away from sphere?



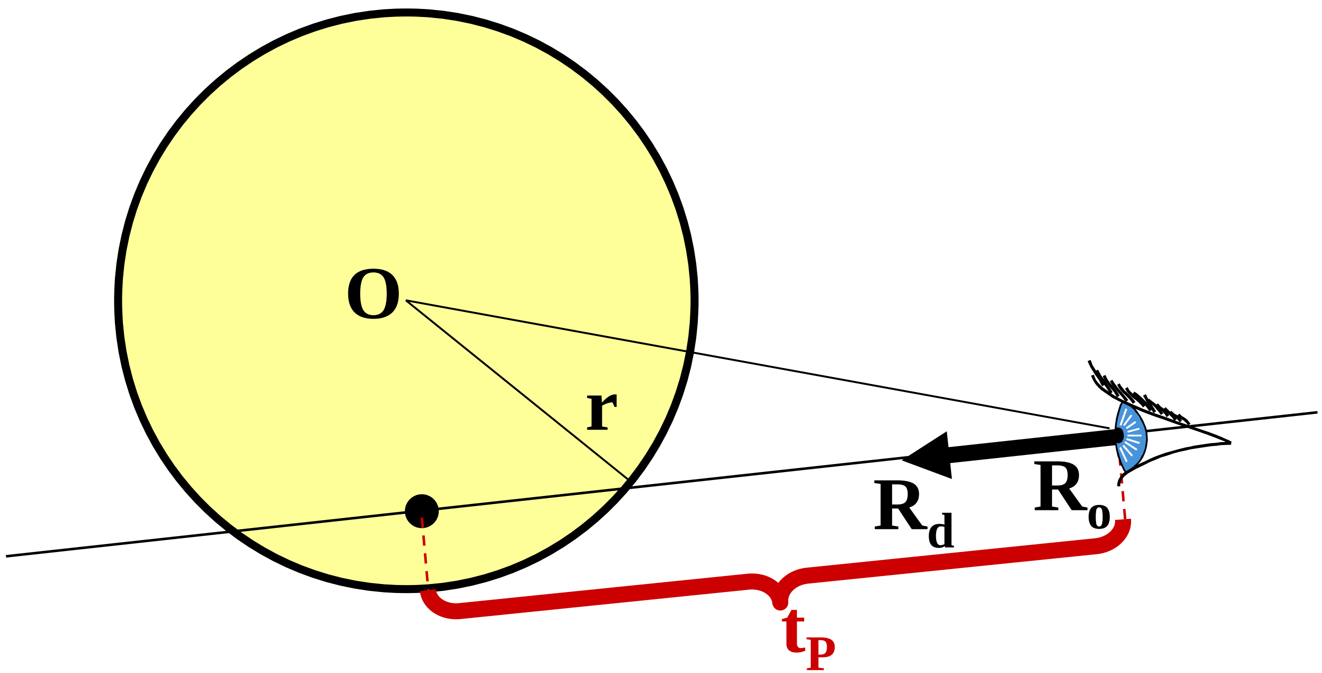
Geometric Ray-Sphere Intersection

- Is ray origin **inside/outside/on** sphere?
  - ($R_o \cdot R_o < r^2$ / $R_o \cdot R_o > r^2$ / $R_o \cdot R_o = r^2$)
  - If origin on sphere, be careful about degeneracies…



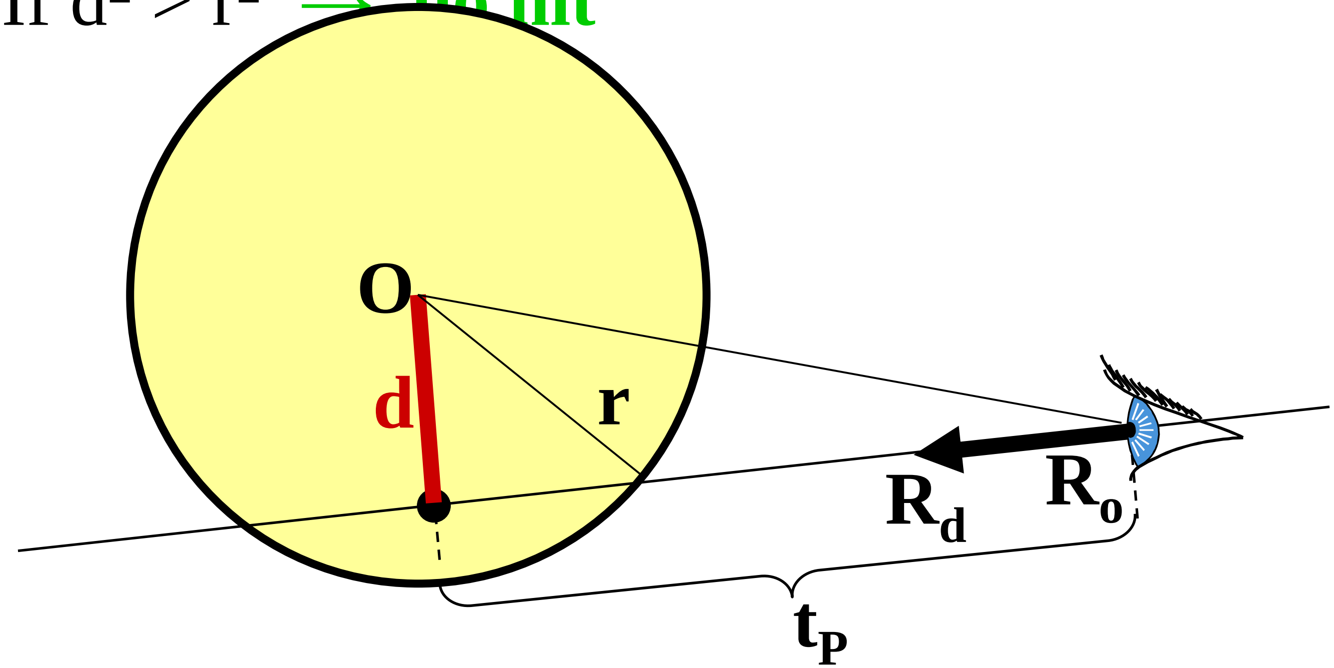
Geometric Ray-Sphere Intersection

- Is ray origin **inside/outside/on** sphere?
- Find closest point to sphere center, $\mathbf{t_P = - R_o \cdot R_d}$
  - If origin outside & $t_P < 0$ → **no hit**



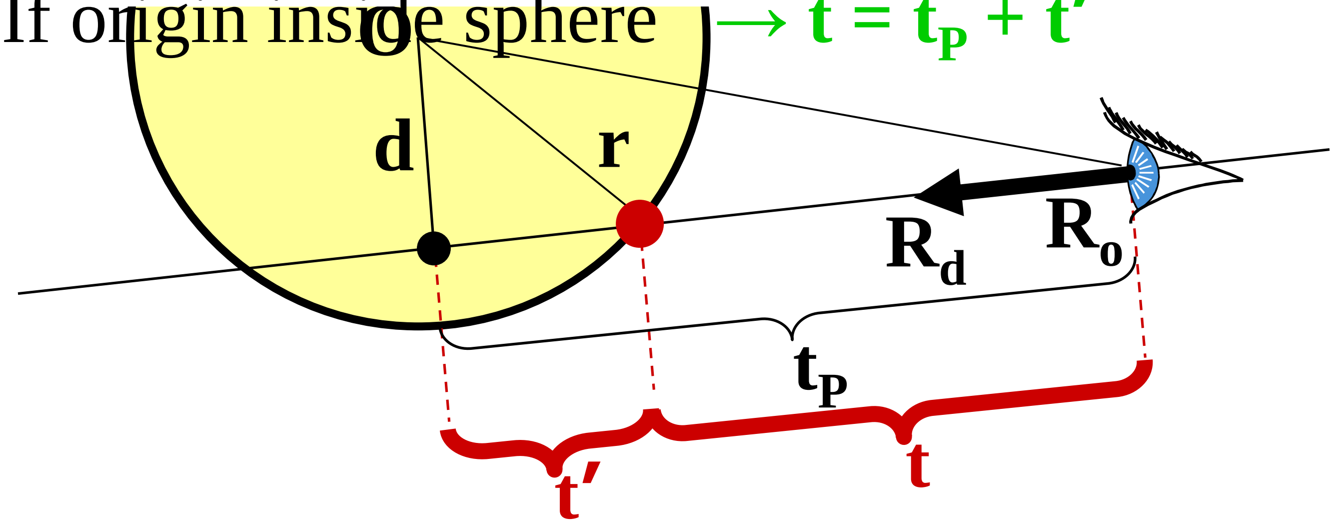
Geometric Ray-Sphere Intersection

- Is ray origin **inside/outside/on** sphere?
- Find closest point to sphere center, $\mathbf{t_P = - R_o \cdot R_d}$.
- Find squared distance, $\mathbf{d^2 = R_o \cdot R_o - t_P{}^2}$
  - If $d^2 > r^2$ $\rightarrow$ **no hit**



Geometric Ray-Sphere Intersection

- Is ray origin **inside/outside/on** sphere?
- Find closest point to sphere center, $\mathbf{t_P = - R_o \cdot R_d}$.
- Find squared distance: $\mathbf{d^2 = R_o \cdot R_o - t_P^2}$
- Find distance (t′) from closest point (t_P) to correct intersection: $\mathbf{t'^2 = r^2 - d^2}$
  - If origin outside sphere → $\mathbf{t = t_P - t'}$
  - If origin inside sphere → $\mathbf{t = t_P + t'}$

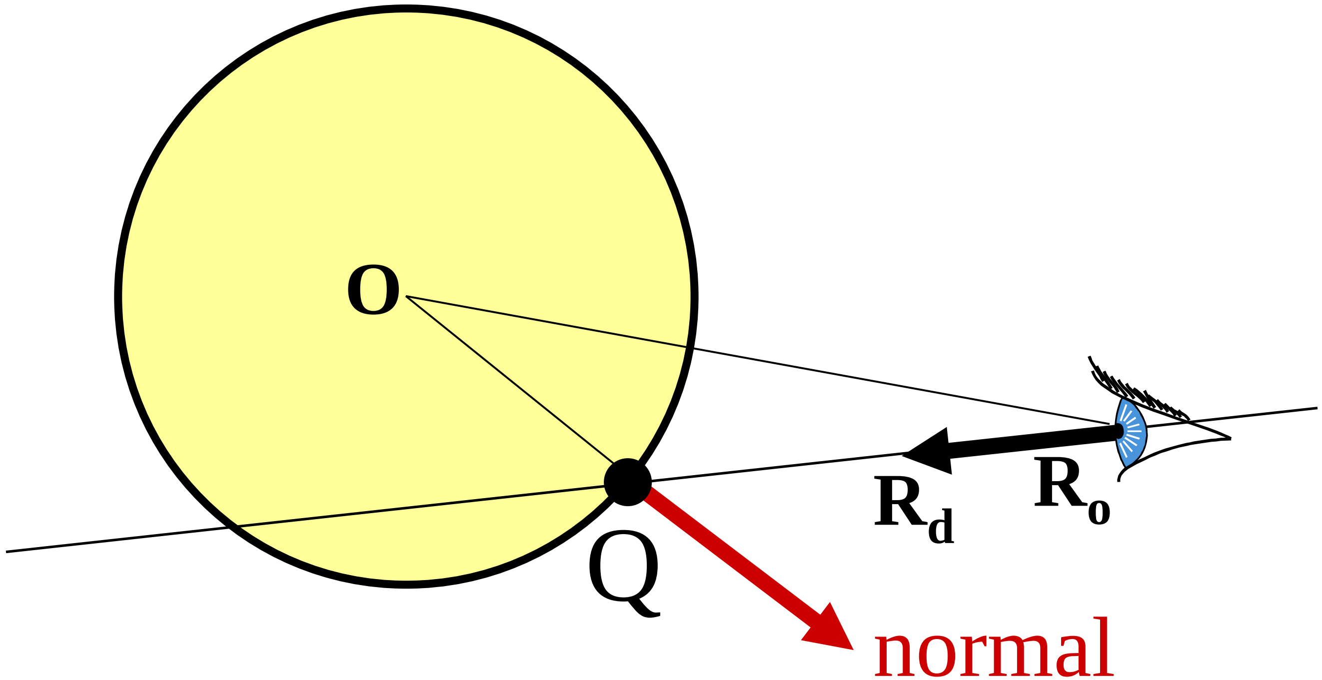


Geometric vs. Algebraic

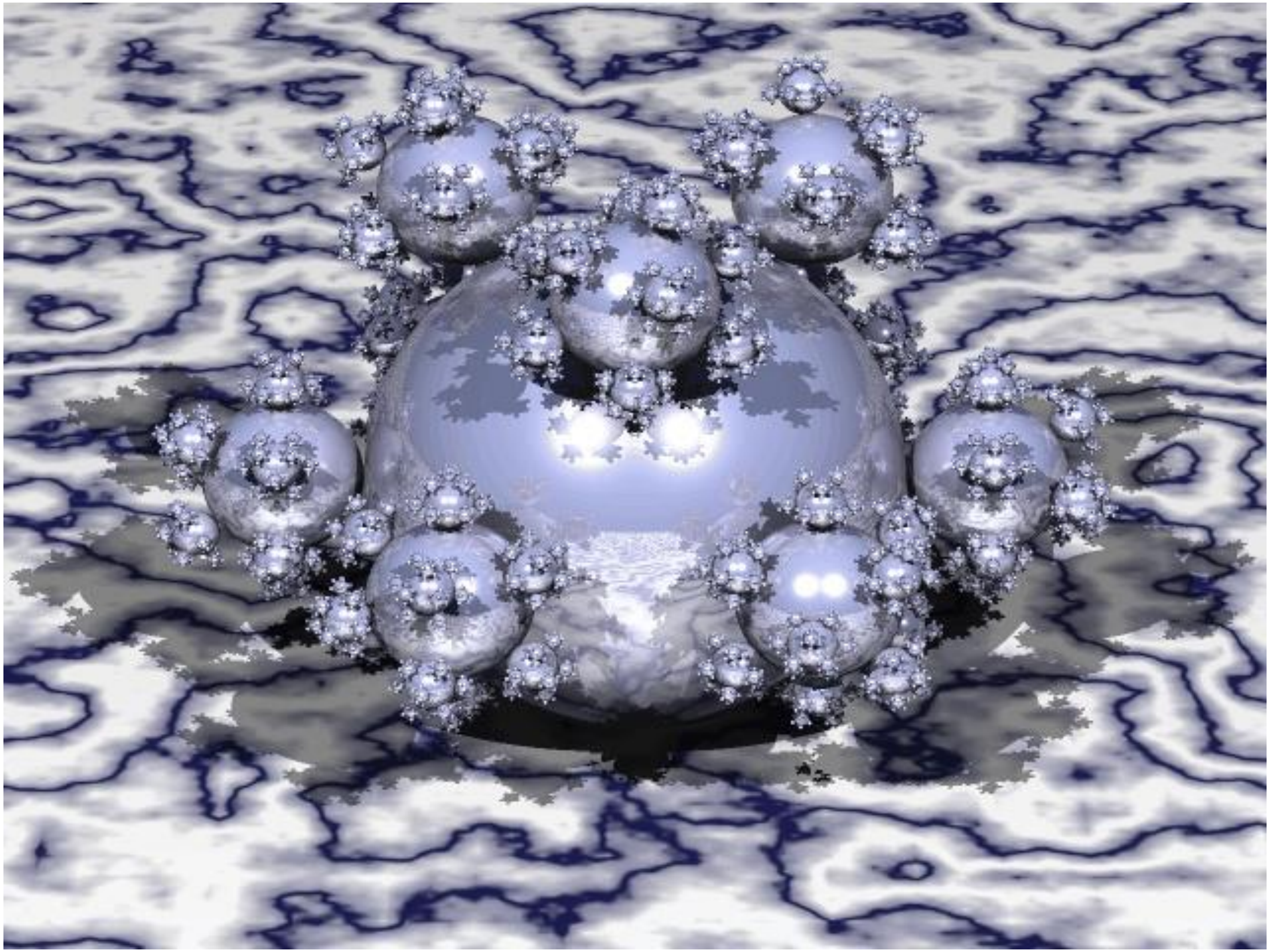
- Algebraic is simple & generic
- Geometric is more efficient
  - Timely tests
  - In particular for rays outside and pointing away

Sphere Normal

- Simply Q/||Q||
  - Q = P(t), intersection point
  - (for spheres centered at origin)

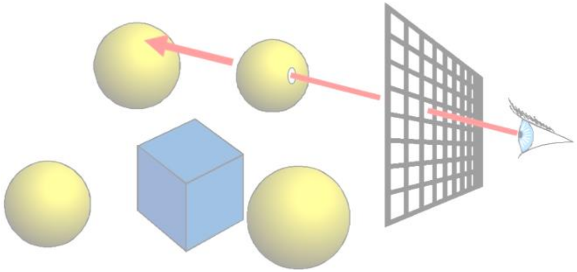


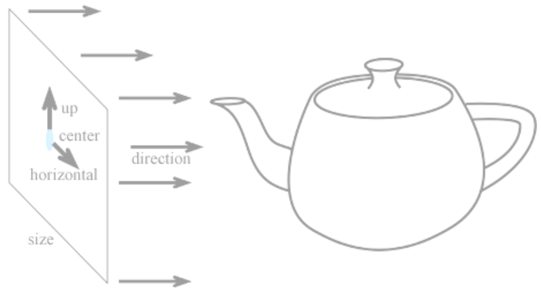
Questions?

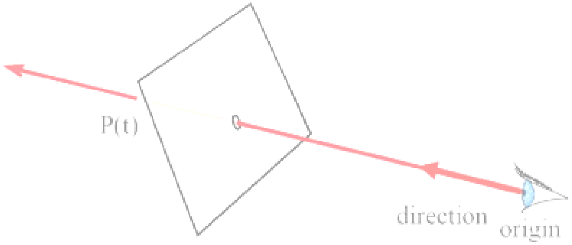


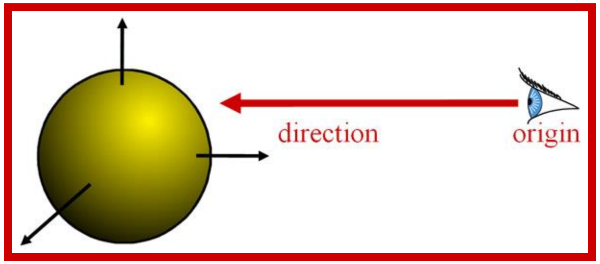
Overview of Today

- Ray-Sphere Intersection
- Ray-Triangle Intersection
- Implementing CSG



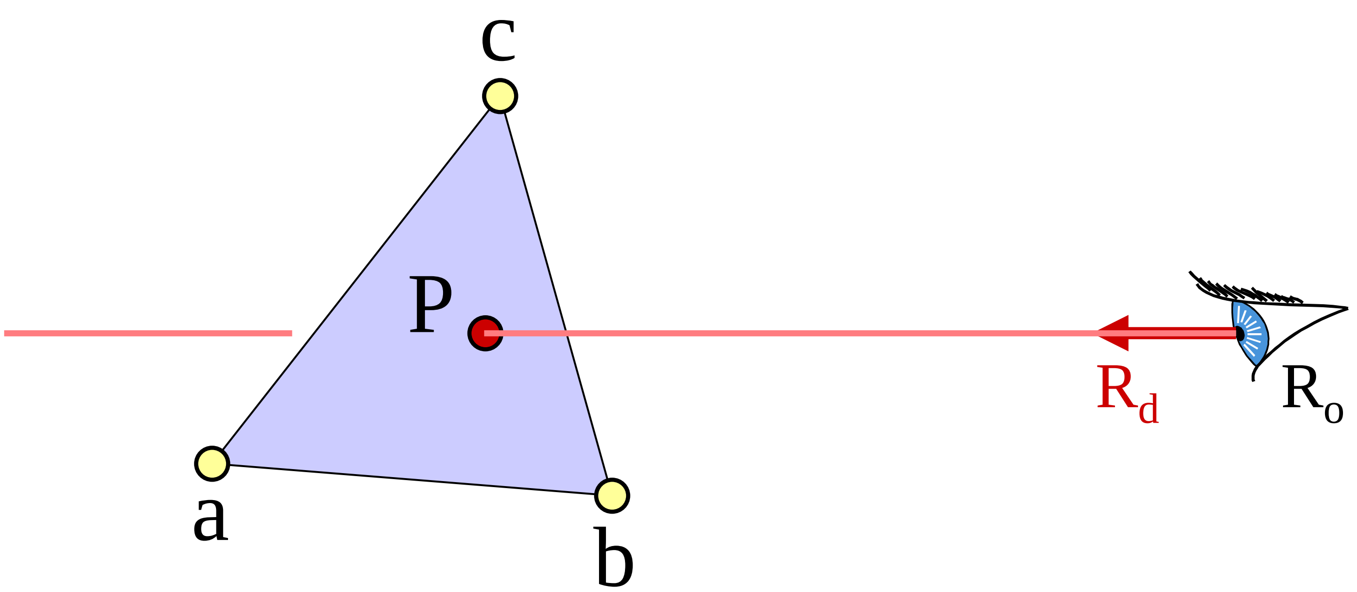






Ray-Triangle Intersection

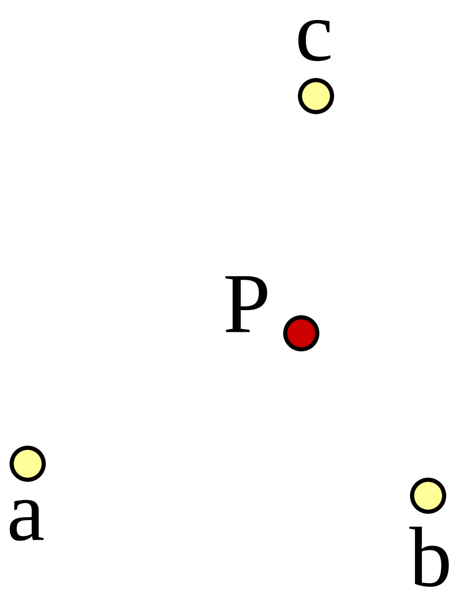
- Use general ray-polygon
- Or try to be smarter
  - Use barycentric coordinates



Barycentric Definition of a Plane

[Möbius, 1827]

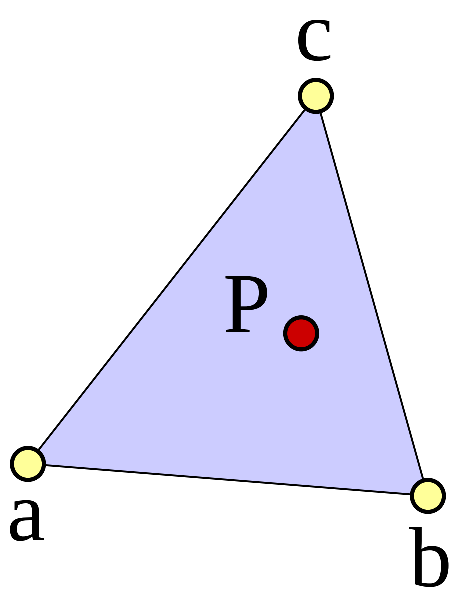
- $P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$
  with $\alpha + \beta + \gamma = 1$
- Is it explicit or implicit?



P is the *barycenter*:
the single point upon which the plane would balance if weights of size α, β, & γ are placed on points a, b, & c.

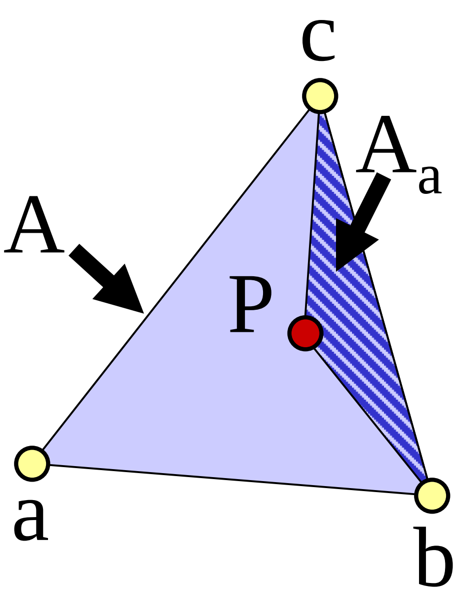
Barycentric Definition of a Triangle

- $P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$
  with $\alpha + \beta + \gamma = 1$
- AND $0 < \alpha < 1$ & $0 < \beta < 1$ & $0 < \gamma < 1$



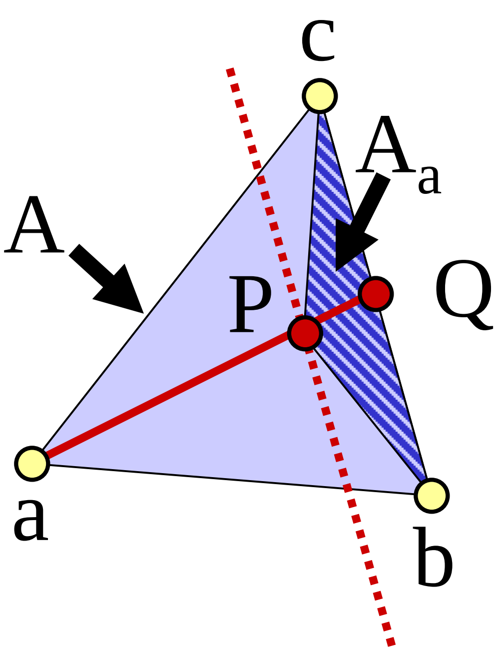
How Do We Compute α, β, γ ?

- Ratio of opposite sub-triangle area to total area
  - $\alpha = A_a/A$ $\beta = A_b/A$ $\gamma = A_c/A$
- Use signed areas for points outside the triangle



Intuition Behind Area Formula

- P is barycenter of a and Q
- A_a is the interpolation coefficient on $\overline{aQ}$
- All points on lines parallel to $\overline{bc}$ have the same α (All such triangles have same height/area)



Simplify

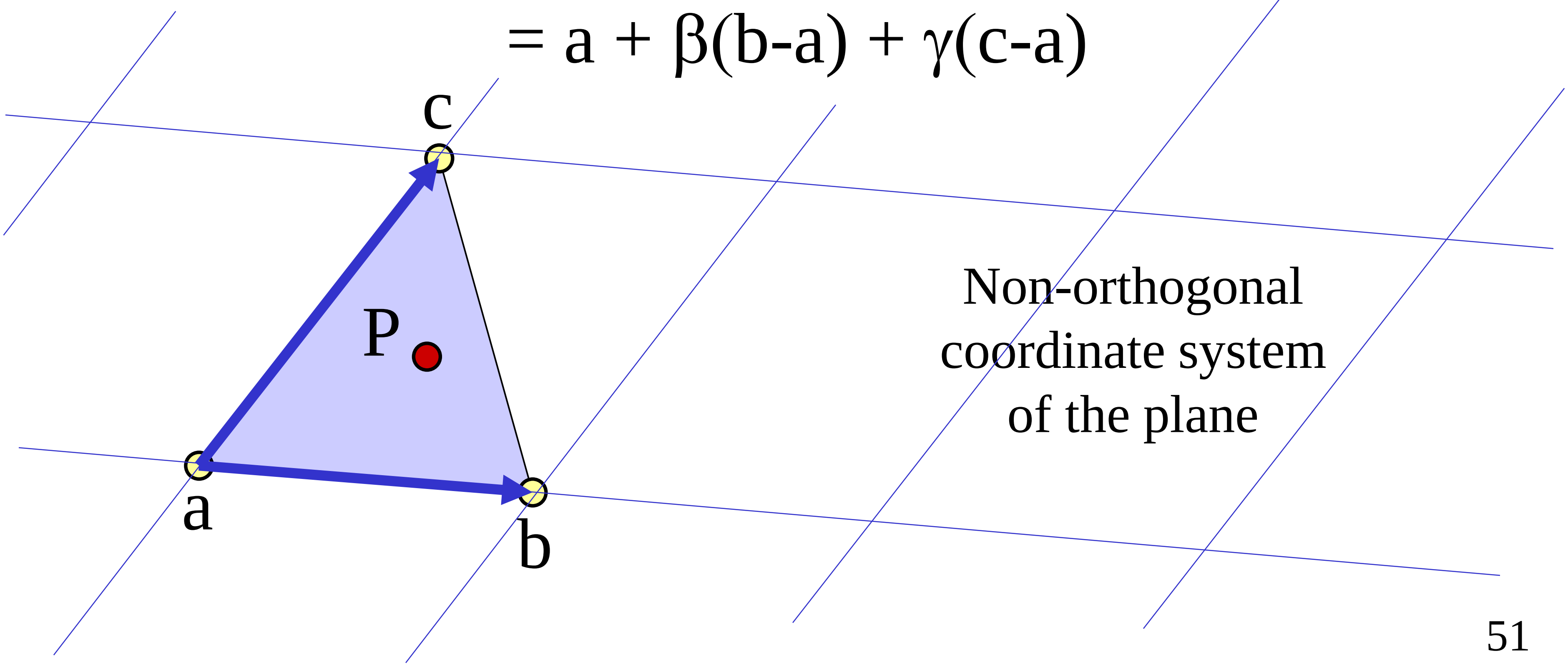
- Since $\alpha + \beta + \gamma = 1$, we can write $\alpha = 1 - \beta - \gamma$

$$P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$$

rewrite

$$P(\beta, \gamma) = (1-\beta-\gamma)a + \beta b + \gamma c$$

$$= a + \beta(b-a) + \gamma(c-a)$$



Intersection with Barycentric Triangle

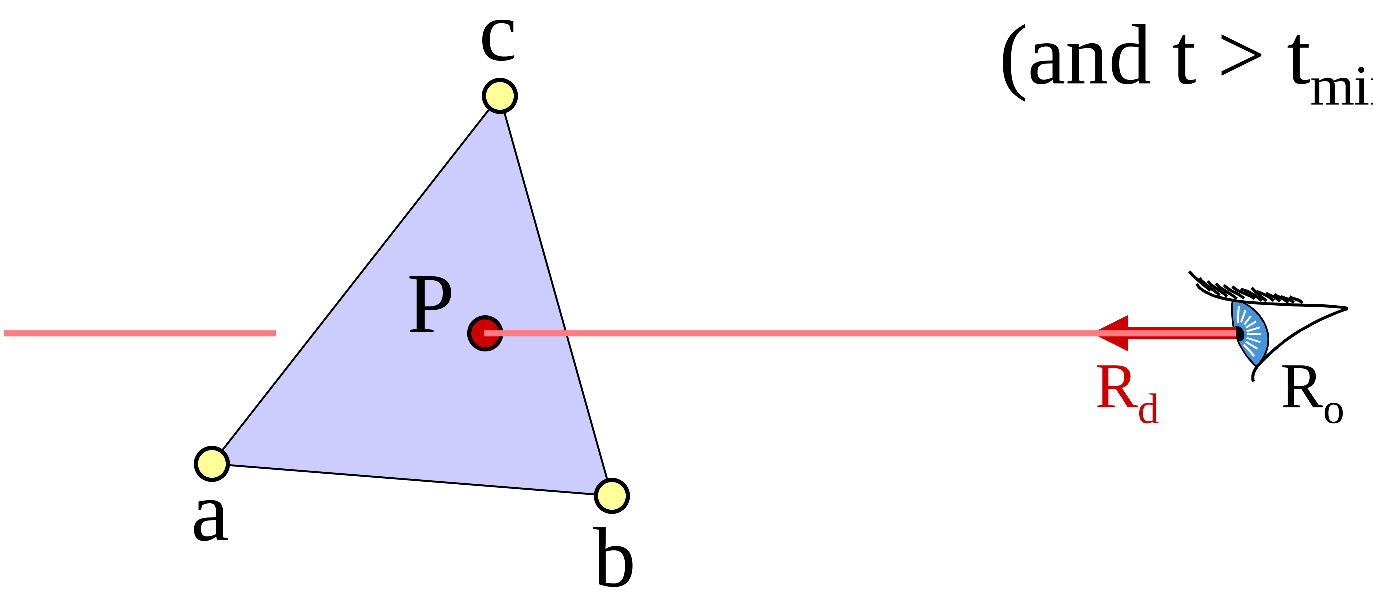
- Set ray equation equal to barycentric equation

$$P(t) = P(\beta, \gamma)$$

$$R_o + t * R_d = a + \beta(b-a) + \gamma(c-a)$$

- Intersection if $\beta + \gamma < 1 \quad \& \quad \beta > 0 \quad \& \quad \gamma > 0$

  (and $t > t_{min}$ …)



Intersection with Barycentric Triangle

- $R_o + t * R_d = a + \beta(b\text{-}a) + \gamma(c\text{-}a)$

$$\left.\begin{aligned} R_{ox} + tR_{dx} &= a_x + \beta(b_x - a_x) + \gamma(c_x - a_x) \\ R_{oy} + tR_{dy} &= a_y + \beta(b_y - a_y) + \gamma(c_y - a_y) \\ R_{oz} + tR_{dz} &= a_z + \beta(b_z - a_z) + \gamma(c_z - a_z) \end{aligned}\right\} \text{3 equations, 3 unknowns}$$

- Regroup & write in matrix form:

$$\begin{bmatrix} a_x - b_x & a_x - c_x & R_{dx} \\ a_y - b_y & a_y - c_y & R_{dy} \\ a_z - b_z & a_z - c_z & R_{dz} \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} a_x - R_{ox} \\ a_y - R_{oy} \\ a_z - R_{oz} \end{bmatrix}$$

Cramer's Rule

- Used to solve for one variable at a time in system of equations

$$\beta = \frac{\begin{vmatrix} a_x - R_{ox} & a_x - c_x & R_{dx} \\ a_y - R_{oy} & a_y - c_y & R_{dy} \\ a_z - R_{oz} & a_z - c_z & R_{dz} \end{vmatrix}}{|A|} \qquad \gamma = \frac{\begin{vmatrix} a_x - b_x & a_x - R_{ox} & R_{dx} \\ a_y - b_y & a_y - R_{oy} & R_{dy} \\ a_z - b_z & a_z - R_{oz} & R_{dz} \end{vmatrix}}{|A|}$$

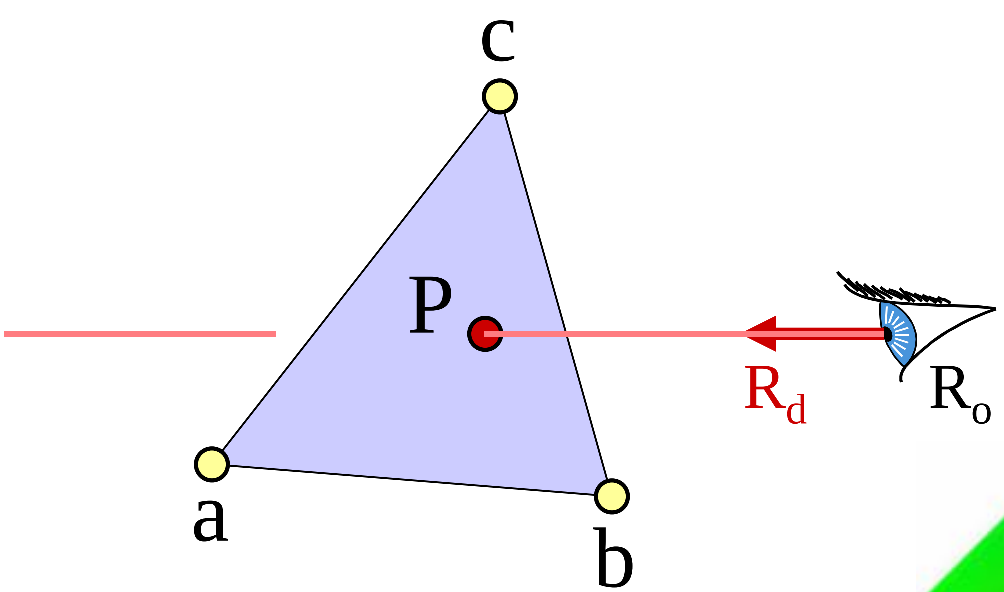
$$t = \frac{\begin{vmatrix} a_x - b_x & a_x - c_x & a_x - R_{ox} \\ a_y - b_y & a_y - c_y & a_y - R_{oy} \\ a_z - b_z & a_z - c_z & a_z - R_{oz} \end{vmatrix}}{|A|}$$

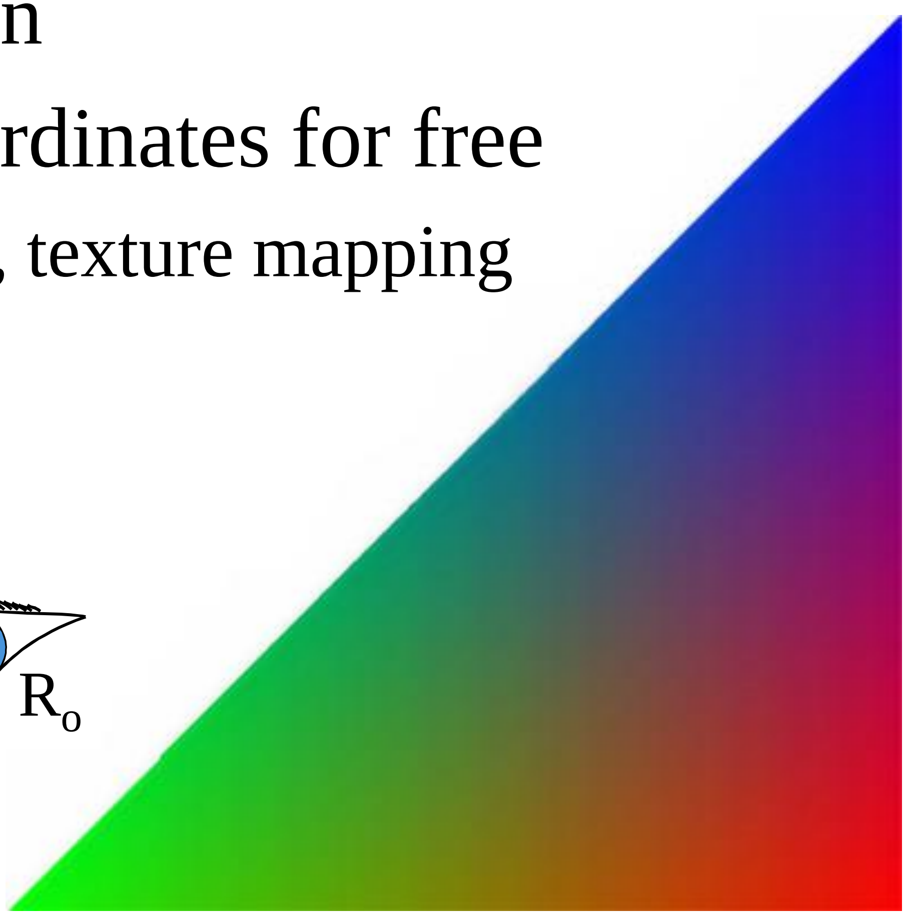
| | denotes the determinant

Can be copied mechanically into code

Advantages of Barycentric Intersection

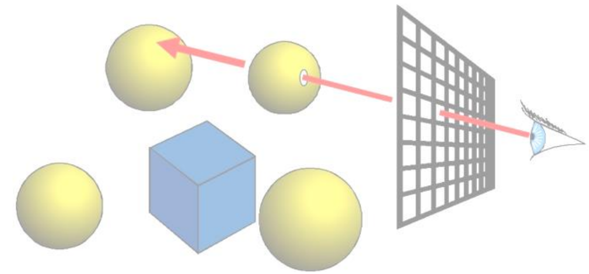
- Efficient
- Stores no plane equation
- Get the barycentric coordinates for free
  - Useful for interpolation, texture mapping

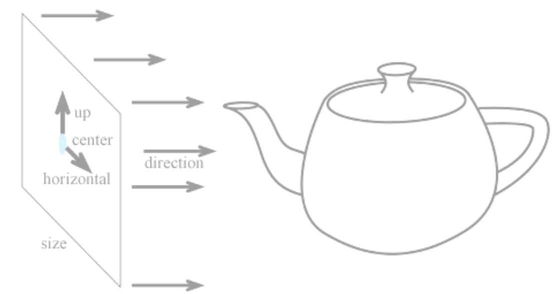


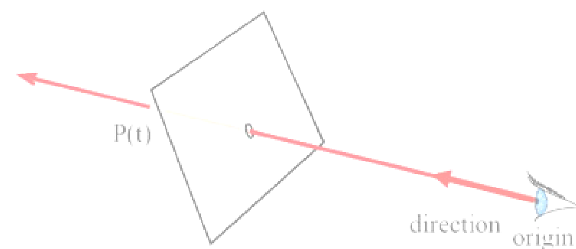


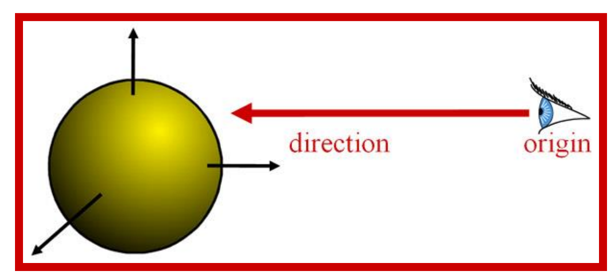
Overview of Today

- Ray-Sphere Intersection
- Ray-Triangle Intersection
- Implementing CSG

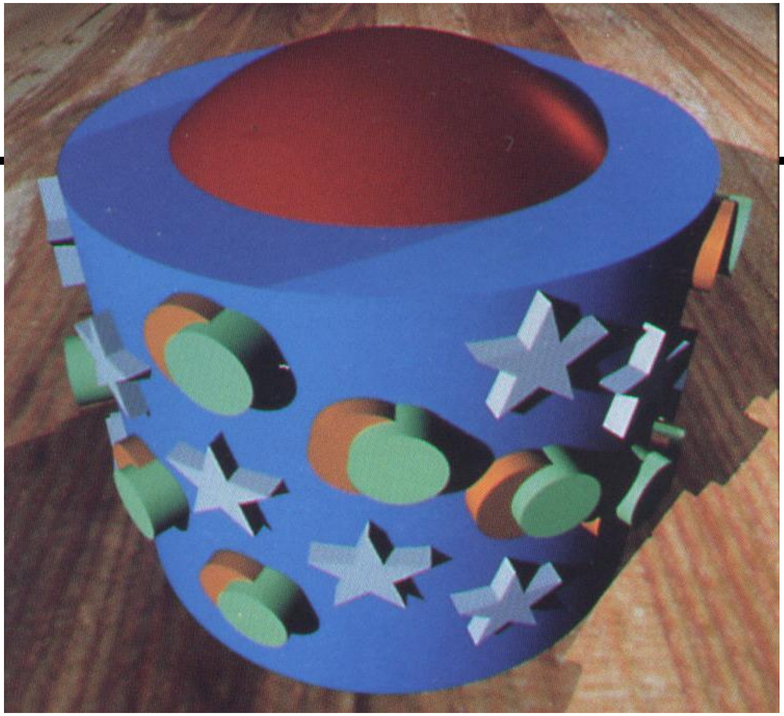


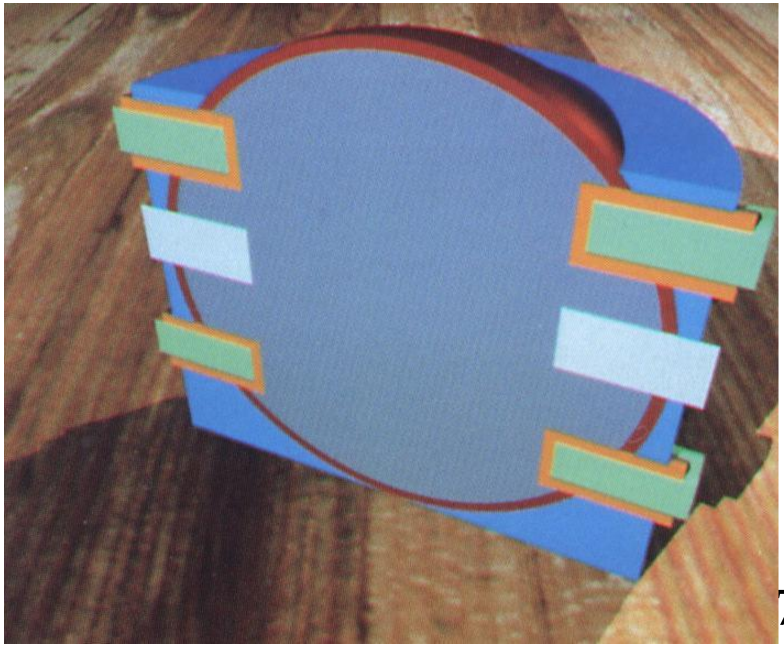




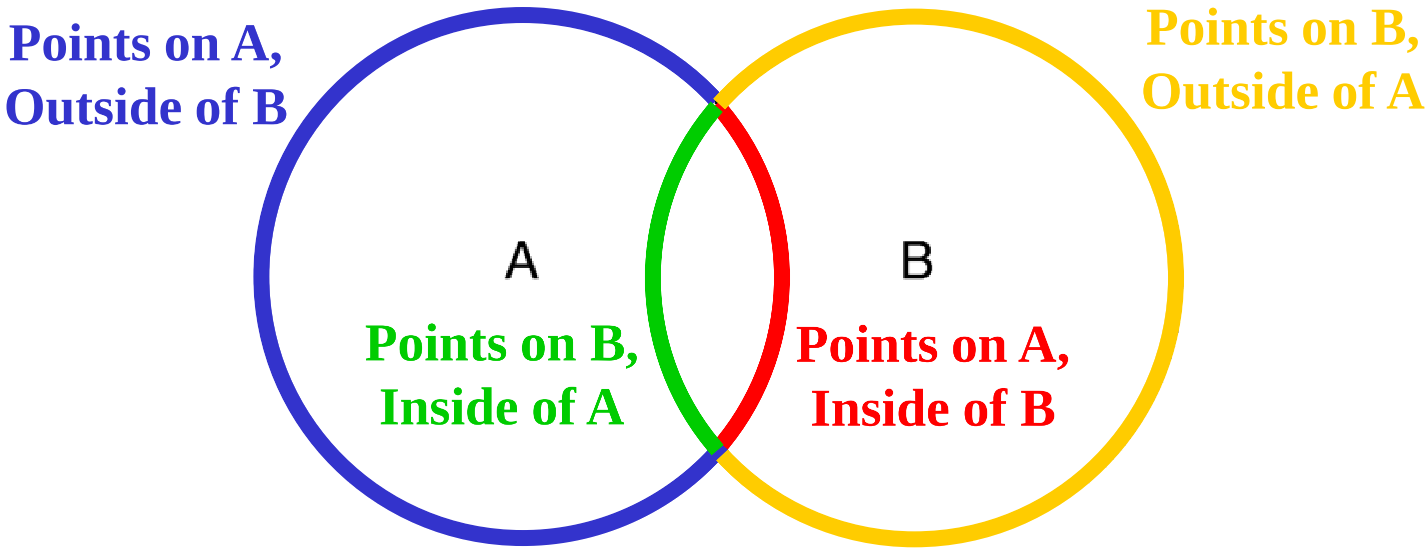


For example:

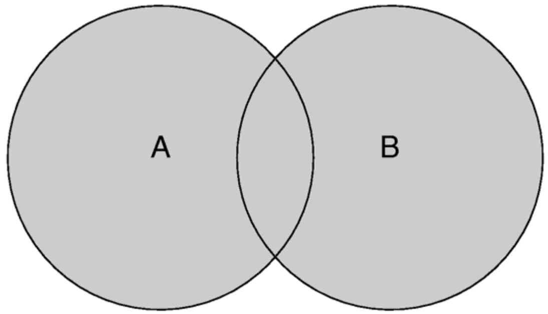




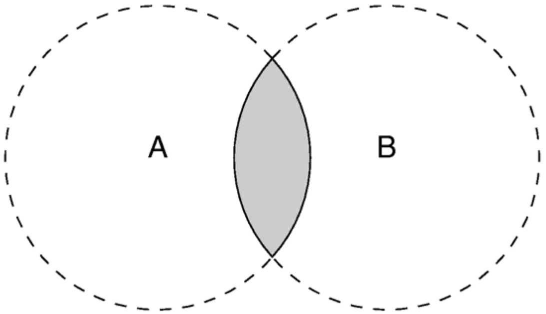
How can we implement CSG?



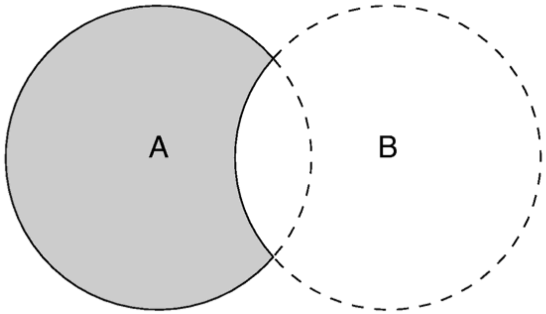
Union



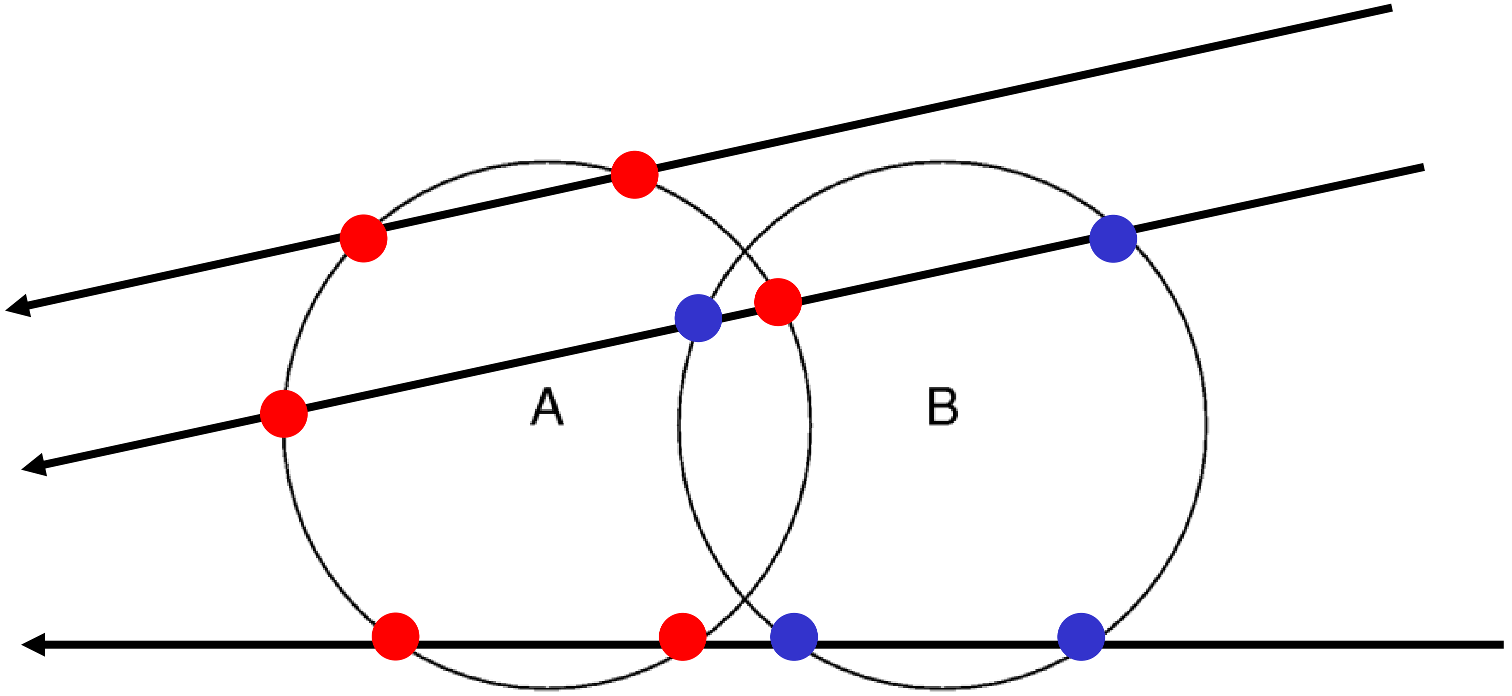
Intersection



Subtraction



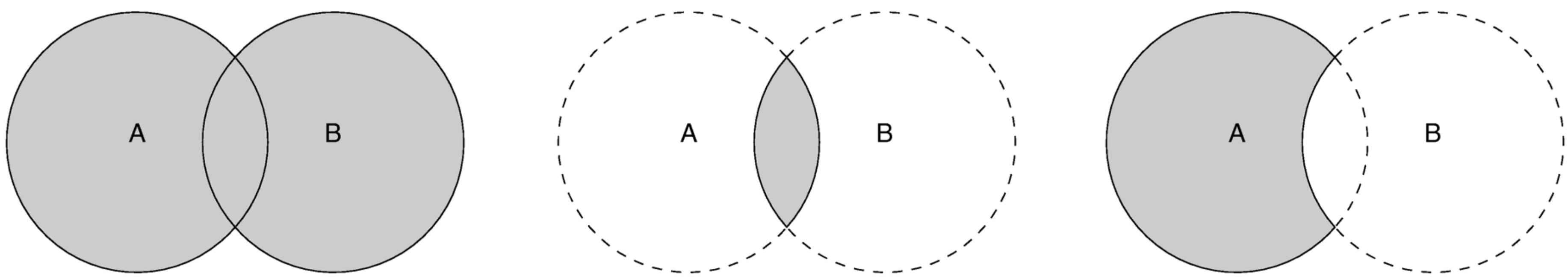
Collect all the intersections



Union

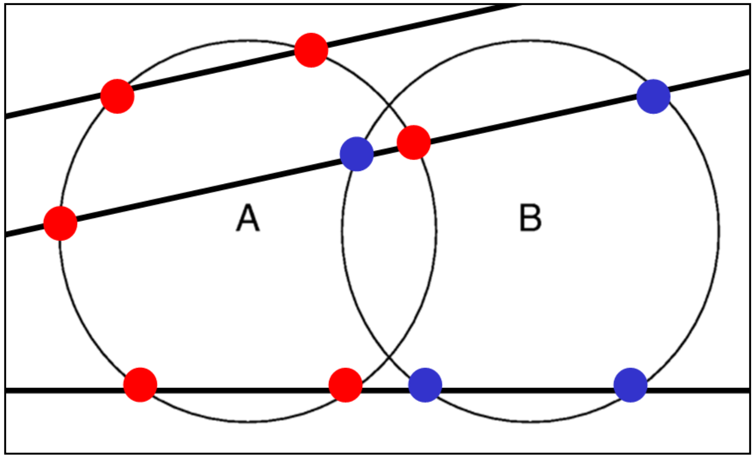
Intersection

Subtraction

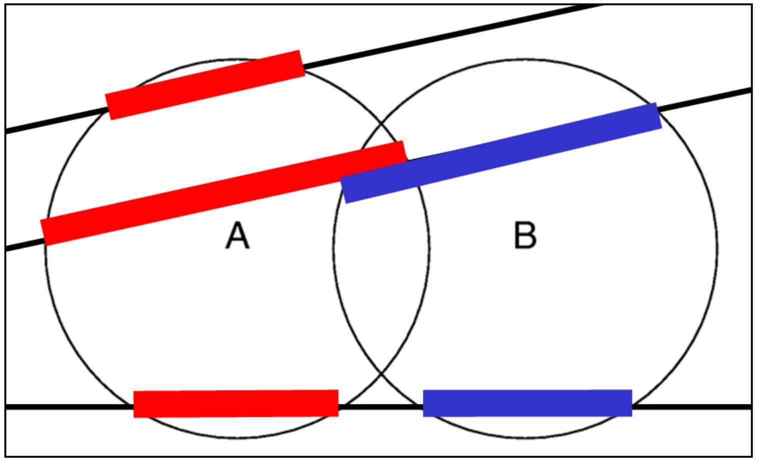


Implementing CSG

1. Test "inside" intersections:
   - Find intersections with A, test if they are inside/outside B
   - Find intersections with B, test if they are inside/outside A

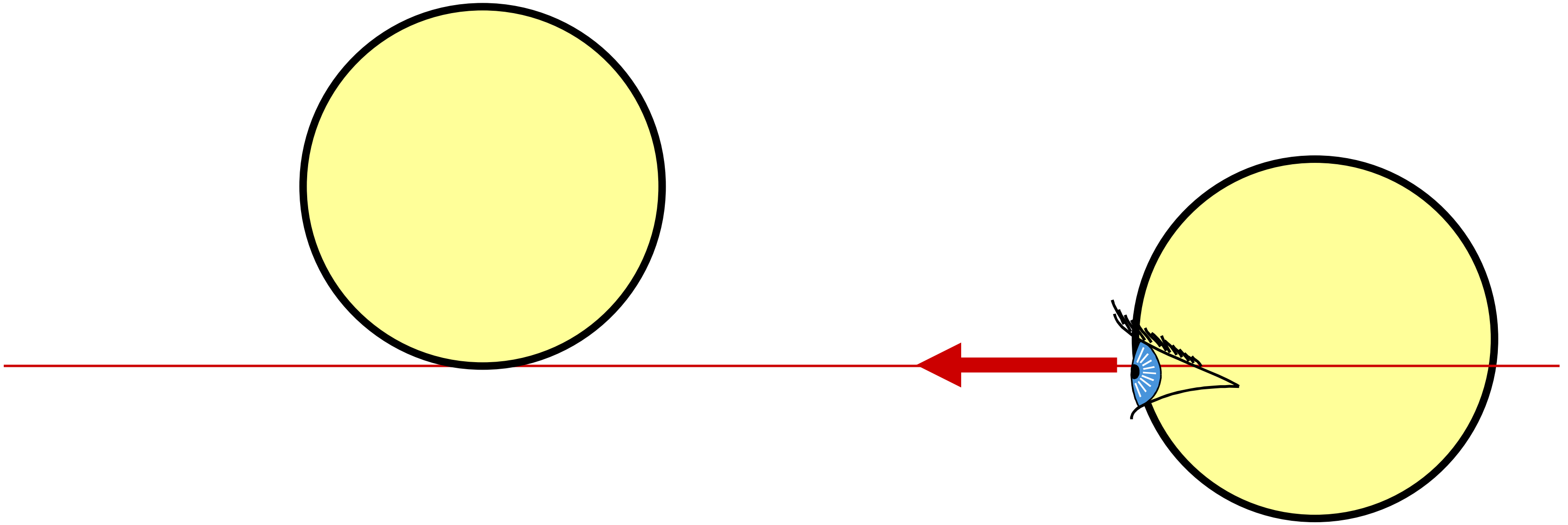


2. Overlapping intervals:
   - Find the intervals of "inside" along the ray for A and B
   - Compute union/intersection/subtraction of the intervals



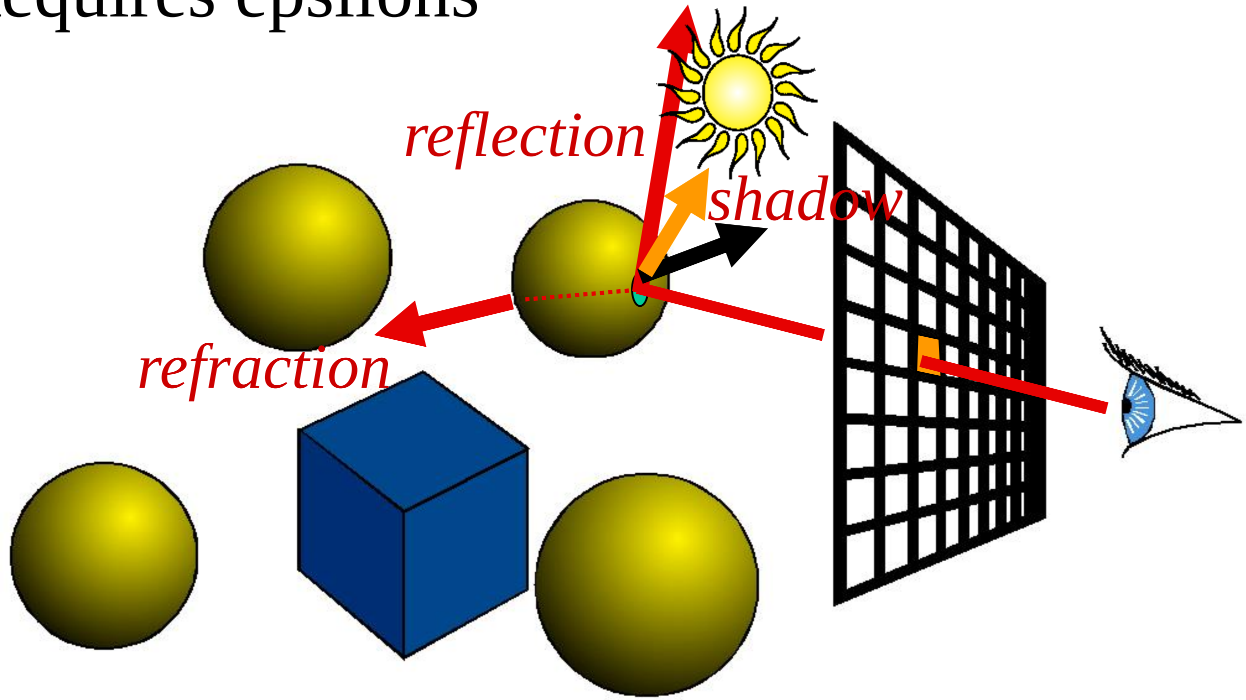
Precision

- What happens when
  - Origin is on an object?
  - Grazing rays?
- Problem with floating-point approximation



The evil ε

- In ray tracing, do NOT report intersection for rays starting at the surface (no false positive)
  - Because secondary rays
  - Requires epsilons



The evil ε: a hint of nightmare

- Edges in triangle meshes
  - Must report intersection (otherwise not watertight)
  - No false negative