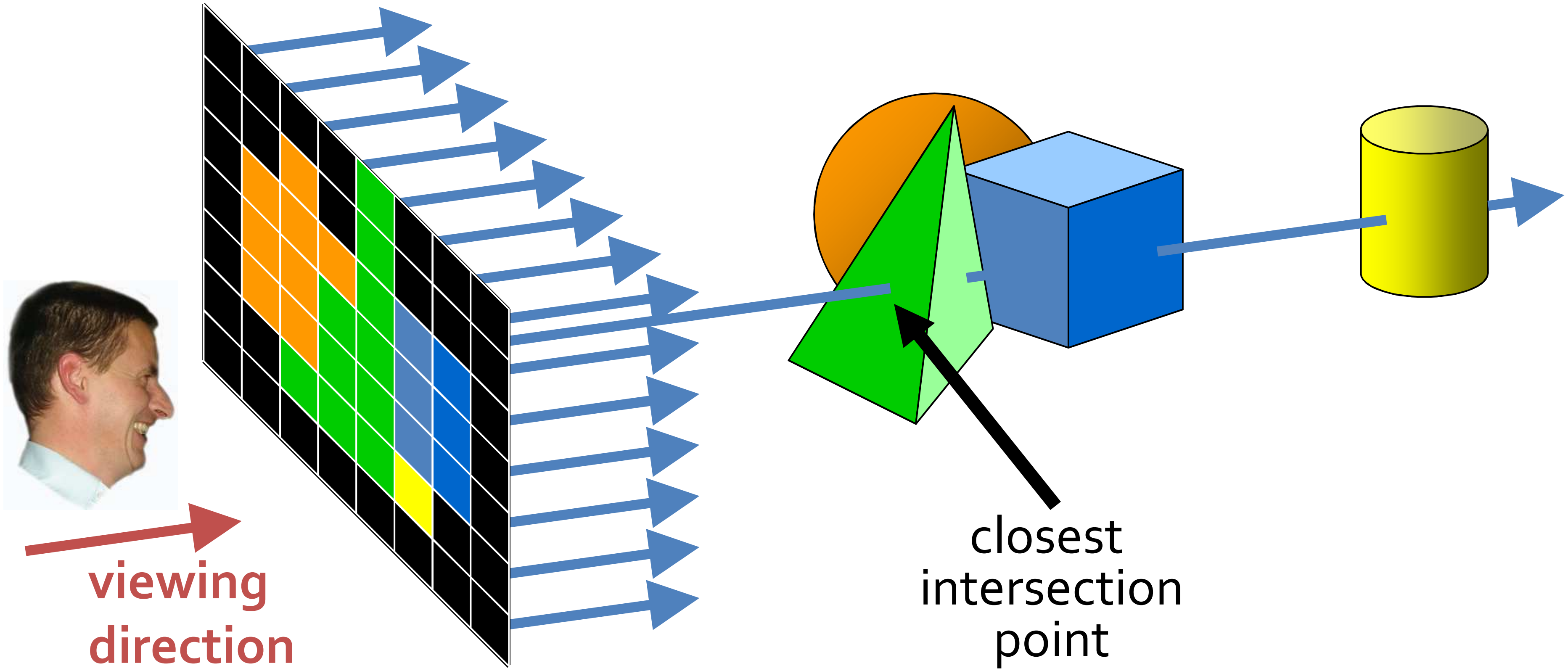
Ray-Casting

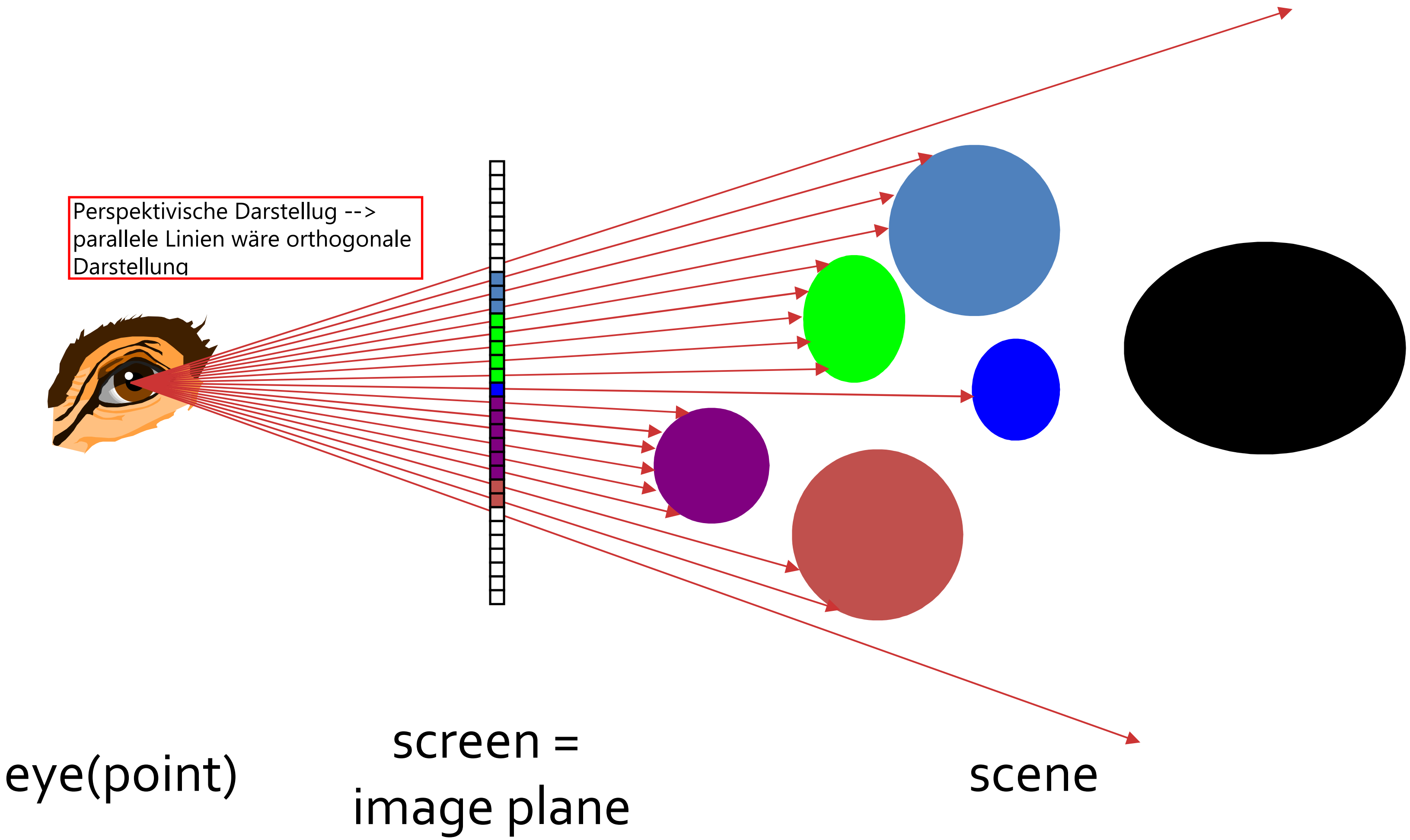
Ray-Casting Method

- line-of-sight of each pixel is intersected with all surfaces
- take closest intersected surface



Generating Rays

- Trace a ray for each pixel in the image plane



Ray Parametric Form

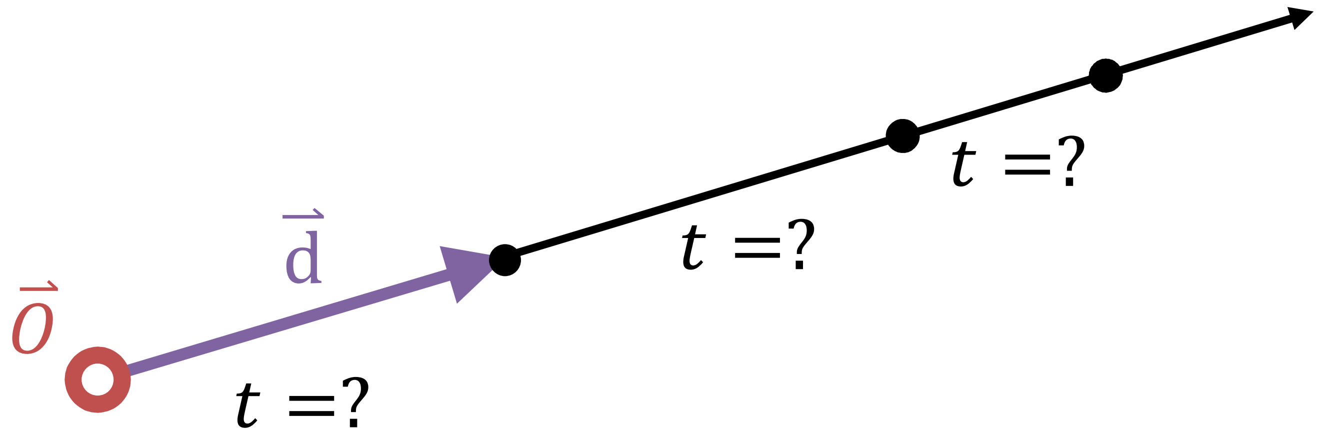
- Ray expressed as function of a single parameter t

$$\vec{\mathrm{P}} = \vec{\mathrm{O}} + t\vec{\mathrm{d}}$$

o = Origin

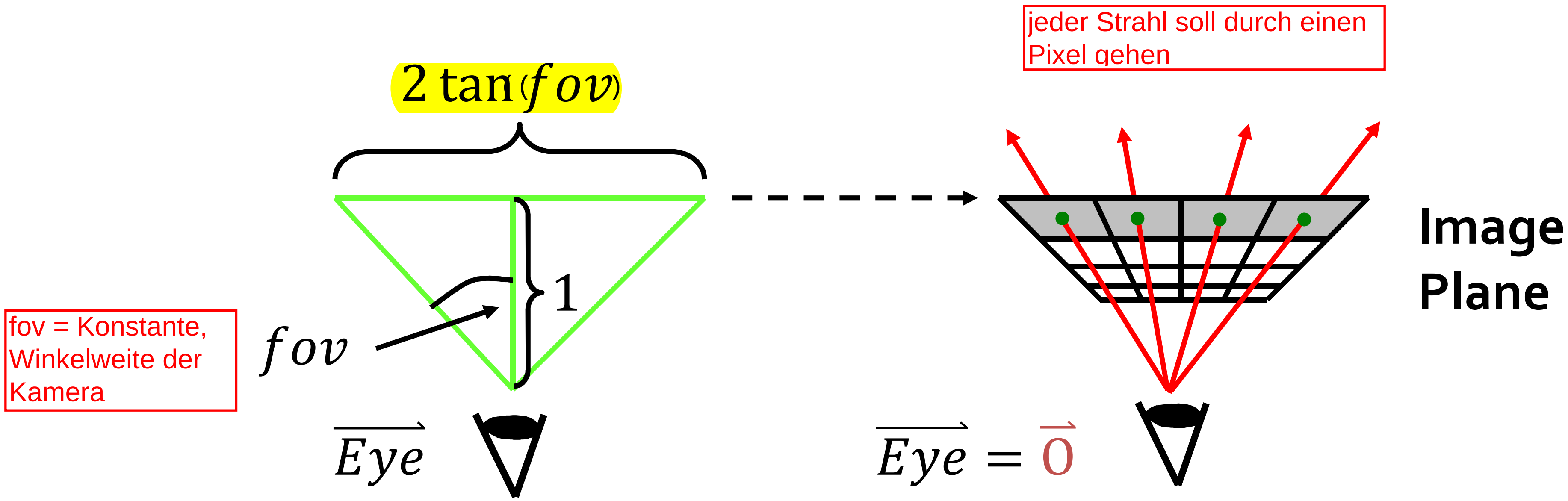
$$= \begin{pmatrix} \mathrm{O}_x \\ \mathrm{O}_y \\ \mathrm{O}_z \end{pmatrix} + t \begin{pmatrix} \mathrm{d}_x \\ \mathrm{d}_y \\ \mathrm{d}_z \end{pmatrix}$$

d = Richtung des Strahls
t = Skalar



Generating Rays

- Trace a ray for each pixel in the image plane



(Looking down from the top)

Generating Rays

- Trace a ray for each pixel in the image plane

(Looking from the side)

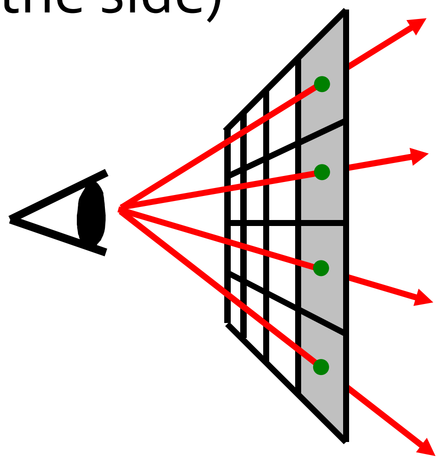
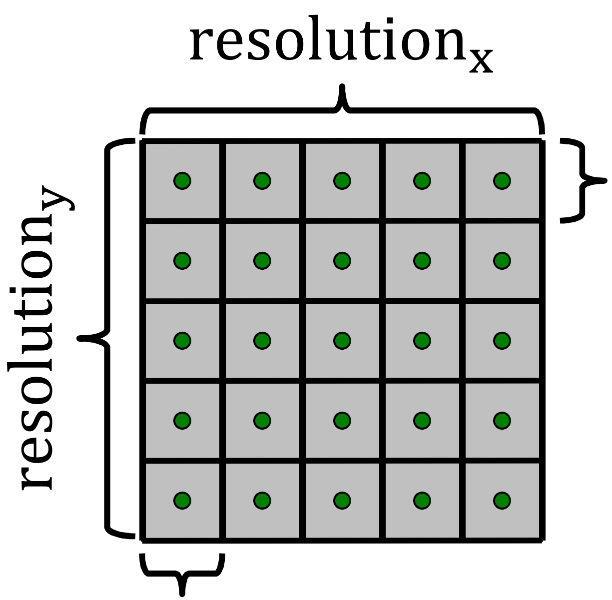


Image Plane



aspect ratio
(kann auch bei dx stehen, je nachdem wie er berechnet wurde

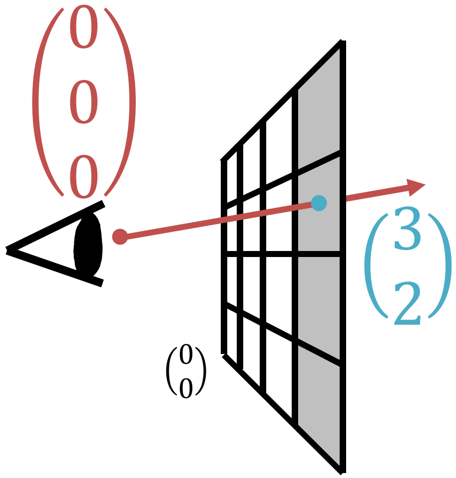
$$d_y = \frac{2\tan fov}{\text{resolution}_y}\,aspect$$

$$d_x = \frac{2\tan fov}{\text{resolution}_x}$$

Generating Rays

dx, dy vorher berechnet

- Trace a ray for each pixel in the image plane
- For a pixel $\begin{pmatrix} x \\ y \end{pmatrix}$ we get $\vec{\mathrm{P}} = \vec{\mathrm{O}} + t\vec{\mathrm{d}} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + t \begin{pmatrix} x d_x \\ y d_y \\ 1 \end{pmatrix}$



Generating Rays

- Trace a ray for each pixel in the image plane

```
renderImage(){
  dx = 2*tan(fov)/resolution.x * aspect;
  dy = 2*tan(fov)/resolution.y;
  for each pixel x, y in the image
    ray.O = (0, 0, 0);
    ray.d = ( x * dx, y * dy, 1);
    ray.normalize();
    image[i][j] = intersect(ray);
  }
```

origin

direction

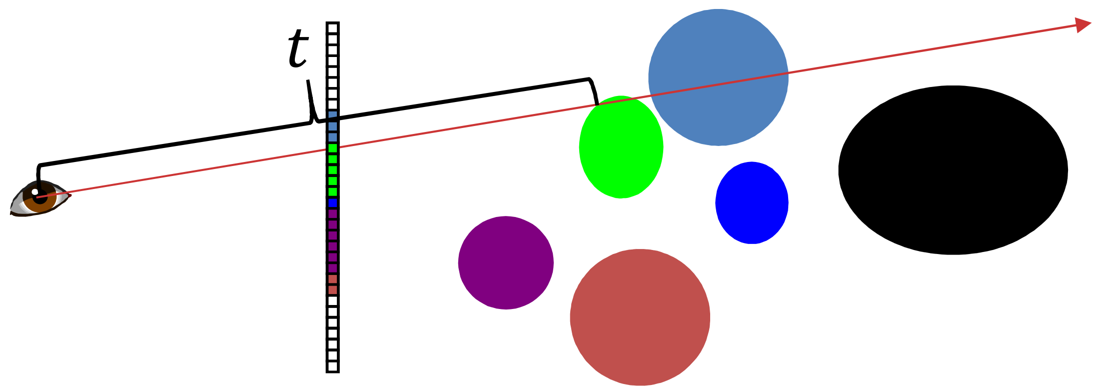
Vektor auf Länge 1 (muss nicht, ist aber evtl später leichter beim rechnen

Ray-Object Intersections

```
intersect(Ray r) {
  foreach object in the scene
      find minimum t > 0 such that r.O+t*r.d hits
    object
    if ( object hit )
    return object
    else return background object
 }
```

für jedes Obkjekt in der Szene

wenn mehrere Objekte getroffen werden, das kleinste t finden, also den vordersten Schnittpunkt herausfinden



Ray-Object Intersections

- Aim: Find the parameter value, t_i, at which the ray first meets object i Ziel, finde t für das näheste Objekt
- Write the surface of the object implicitly: $f(\mathbf{x})=0$
  - Unit sphere at the origin is $\mathbf{x}\bullet\mathbf{x}-1=0$ Kugel im Ursprung mit Größe 1
  - Plane with normal **n** passing through origin is: $\mathbf{n}\bullet\mathbf{x}=0$ Ebene durch den Ursprung mit Normale n
- Put the ray equation in for **x** Wie wird der Schnitt gefunden?
  - Result is an equation of the form $f(t)=0$ where we want t
  - Now it's just root finding

p = o +t*d --> ray(t) = o + t*d
Bsp Kugel:
x*x-1=0 --> ray(t)*ray(t)-1=0
Bsp Plane:
n*x=0 --> n*ray(t)=0

(o+td)*(o+td)-r²=0
r, o, d sind bekannt
Bsp.: r=1, o=(0,0,0), d=(0,0,1)
(t*(0,0,1))*(t*(0,0,1)) = 1
(0,0,t)*(0,0,t)=1
t²=1 --> t=+-1 --> t=1 (wir wollen das kleinste positive t, Blickrichtung positiv)

Ray Object Intersection

- Equation of a ray $r(t) = \mathbf{S} + \mathbf{c}t$
  - **"S"** is the starting point and **"c"** is the direction of the ray
- Given a surface in implicit form *F(x,y,z)*
  - *plane:* $F(x, y, z) = ax + by + cz + d = \mathbf{n} \cdot \mathbf{x} + d$
  - *sphere:* $F(x, y, z) = x^2 + y^2 + z^2 - 1$
  - *cylinder:* $F(x, y, z) = x^2 + y^2 - 1 \qquad 0 < z < 1$
- All points on the surface satisfy *F(x,y,z)=0*

Der Schnittpunkt wird an der Oberfläche(Surface) gesucht, für den Schnittpunkt

- Thus for ray *r(t)* to intersect the surface $F(r(t)) = 0$
- "t" can be got by solving $F(\mathbf{S} + \mathbf{c}t_{hit}) = 0$

Ray Object Intersection

- Ray polygon intersection
  - Plug the ray equation into the implicit representation of the surface
  - Solve for "*t*"
  - Substitute for "*t*" to find point of intersection
  - Check if the point of intersection falls within the polygon

Ray Object Intersection

Beispiel

- Ray sphere intersection $|\mathbf{p}-\mathbf{p}_c|^2 = r^2 \quad \mathbf{p} = (x,y,z), \mathbf{p}_c = (a,b,c)$
  - Implicit form of sphere given center *(a,b,c)* and radius *r*
- Intersection with *r(t)* gives $|\mathbf{S}+\mathbf{c}t-\mathbf{p}_c|^2 = r^2$
- By the identity $|\mathbf{a}+\mathbf{b}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 + 2(\mathbf{a}\cdot\mathbf{b})$
  - Intersection equation is quadratic in "*t*"

    $$|\mathbf{S}+\mathbf{c}t-\mathbf{p}_c|^2 - r^2 = t^2|c|^2 + 2t\mathbf{c}\cdot(\mathbf{S}-\mathbf{p}_c) + \left(|\mathbf{S}-\mathbf{p}_c|^2 - r^2\right)$$
- Solving for "t" $t = -\mathbf{c}\cdot(\mathbf{S}-\mathbf{p}_c) \pm \sqrt{(\mathbf{c}\cdot(\mathbf{S}-\mathbf{p}_c))^2 - |c|^2\left(|\mathbf{S}-\mathbf{p}_c|^2 - r^2\right)}$
  - Real solutions, indicate one or two intersections
  - Negative solutions are behind the eye
  - If discriminant is negative, the ray missed the sphere

Ray-Casting Method

- based on geometric optics, tracing paths of light rays
- backward tracing of light rays
- suitable for complex, curved surfaces
- special case of ray-tracing algorithms
- efficient ray-surface intersection techniques necessary
  - intersection point
  - normal vector