CS475/CS675
Computer Graphics

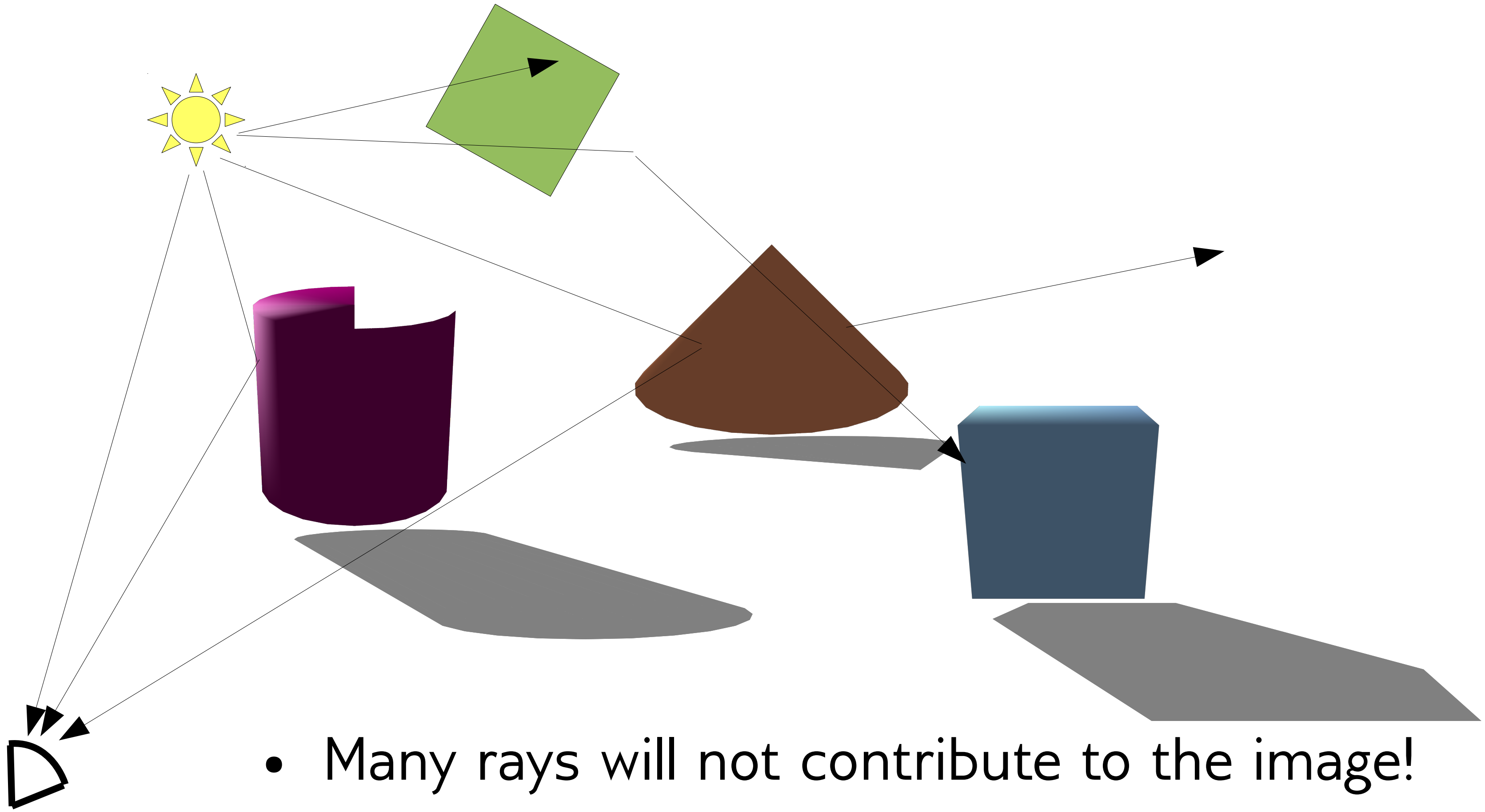
Ray Tracing 1

Rendering

- Drawing images on the computer screen.
- We have seen one rendering method already.
- Issues:
  - Visibility
    - What parts of a scene are visible?
      - Clipping
      - Culling (Backface and Occlusion)
  - Illumination
    - Reflection, Refraction, Shadows, etc.

Ray Tracing

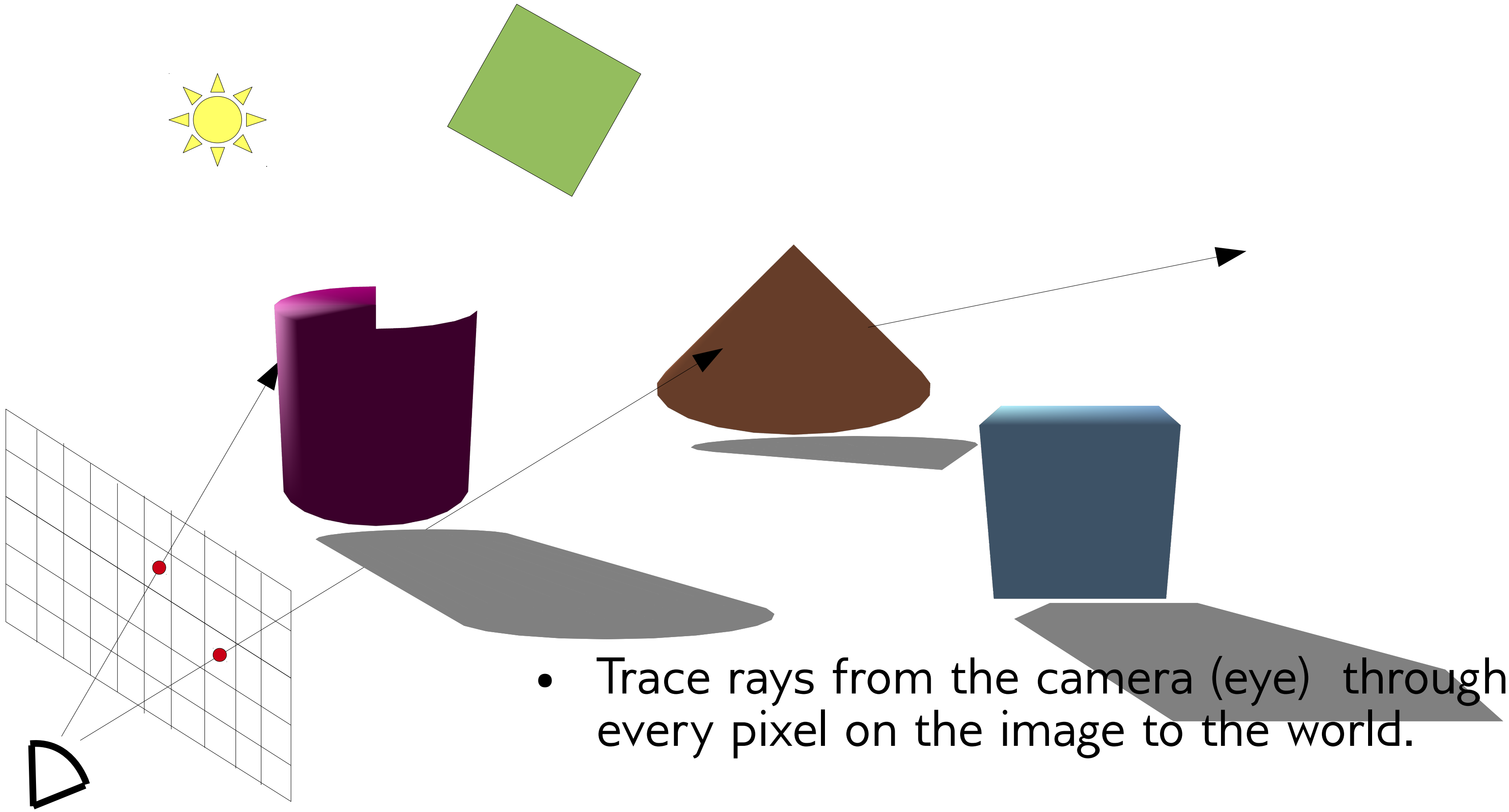
- Tracing the rays of light to model the interaction of light with objects.



- Many rays will not contribute to the image!

Ray Tracing

- Inverse or Backward Ray Tracing



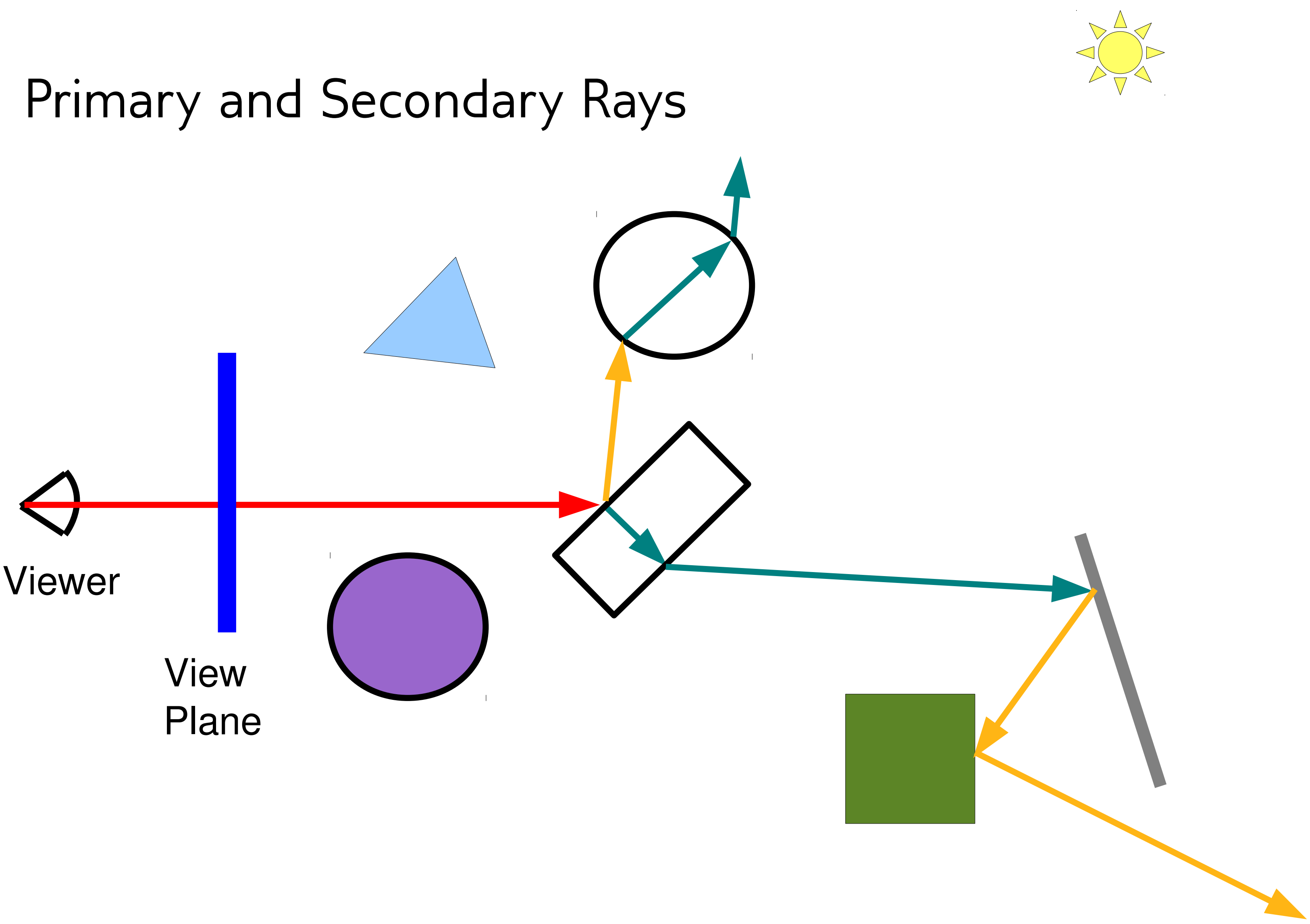
- Trace rays from the camera (eye) through every pixel on the image to the world.

Ray Tracing

- Basic Idea
  - For every pixel in the image
    - Shoot a ray
    - Find closest intersection with object
    - Find normal at the point of intersection
    - Compute illumination at point of intersection
    - Assign pixel color

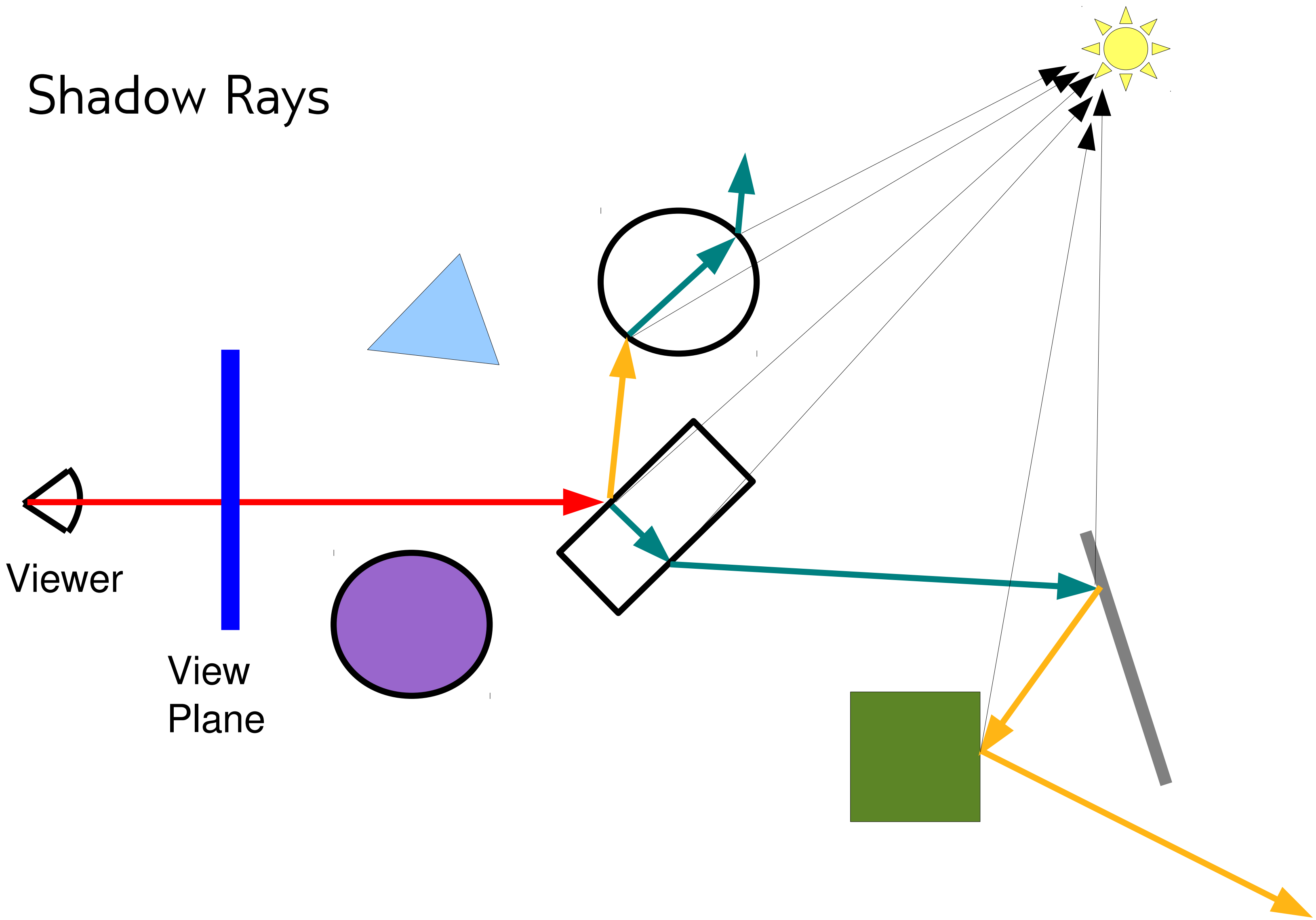
Ray Tracing

- Primary and Secondary Rays



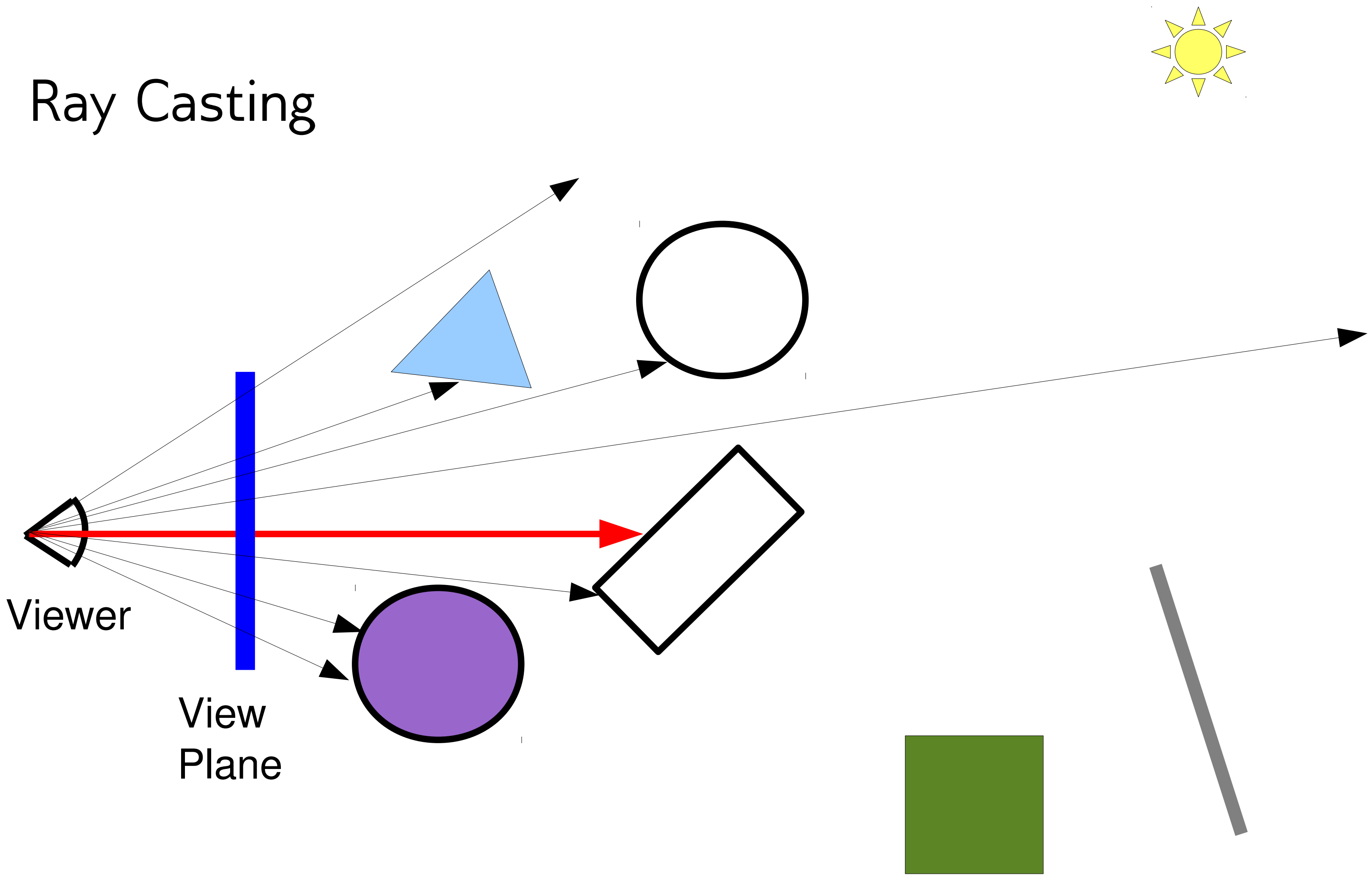
Ray Tracing

- Shadow Rays



Ray Tracing

- Ray Casting

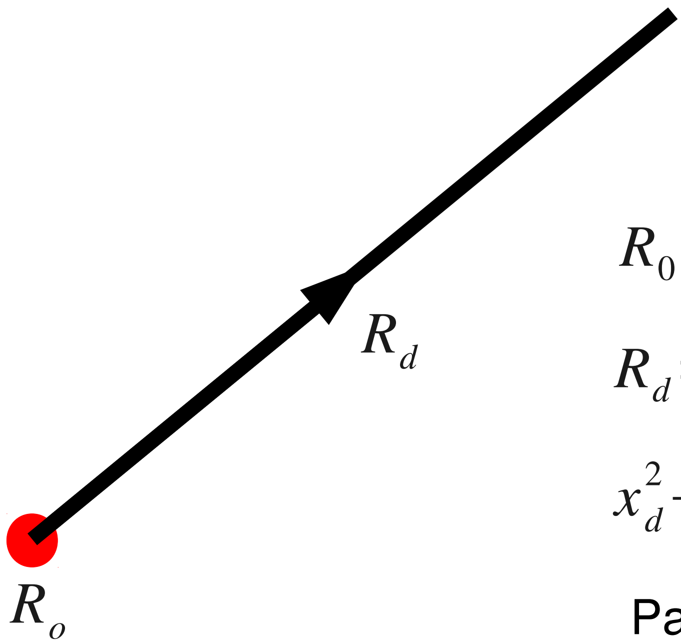


Ray Tracing

- Visibility
  - Ray – Object Intersections
- Illumination
  - Pixel Colour determination (shading)

Ray Tracing

- Ray Representation



$$R_0 = \begin{bmatrix} x_o & y_o & z_o \end{bmatrix}^T$$ Ray Origin

$$R_d = \begin{bmatrix} x_d & y_d & z_d \end{bmatrix}^T$$ Ray Direction

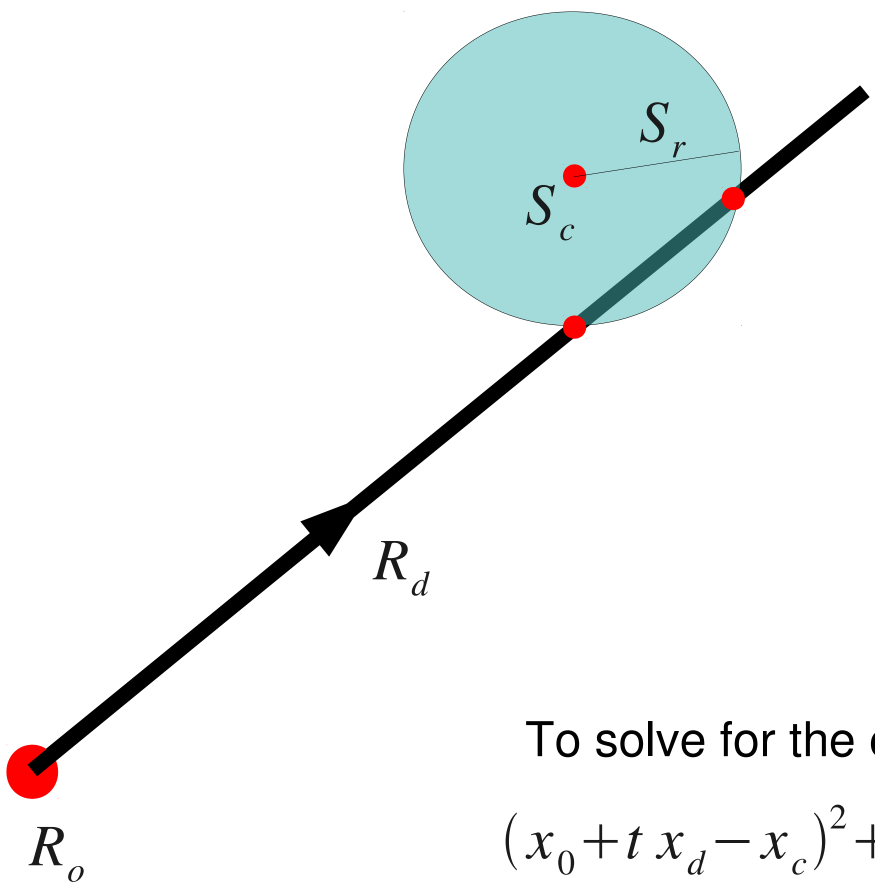
$$x_d^2 + y_d^2 + z_d^2 = 1$$

Parametric form:

$$R(t) = R_0 + tR_d, \qquad t > 0$$

Ray Tracing

- Ray - Sphere Intersection



$S_c = \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T$ Center

S_r Radius

$\begin{bmatrix} x_s & y_s & z_s \end{bmatrix}^T$ Surface Point

Implicit form of the sphere:

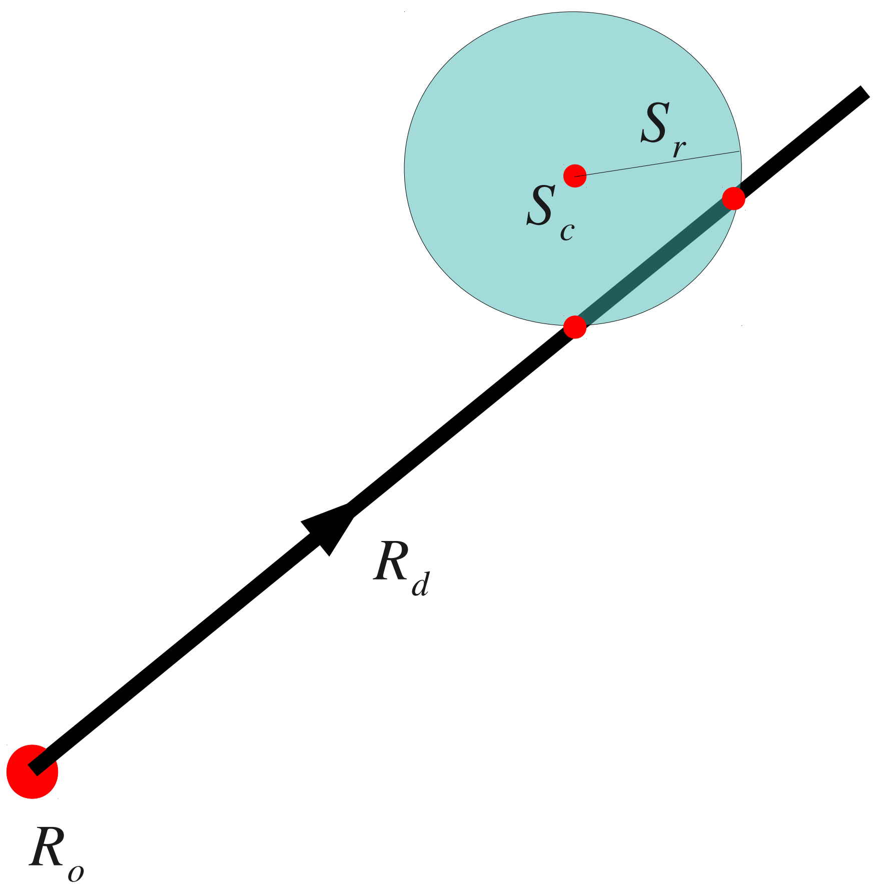
$$(x_s - x_c)^2 + (y_s - y_c)^2 + (z_s - z_c)^2 = S_r^2$$

To solve for the equation we substitute:

$$(x_0 + t\, x_d - x_c)^2 + (y_0 + t\, y_d - y_c)^2 + (z_0 + t\, z_d - z_c)^2 = S_r^2$$

Ray Tracing

- Ray - Sphere Intersection



We get a quadratic in t

$$At^2 + Bt + C = 0$$

where

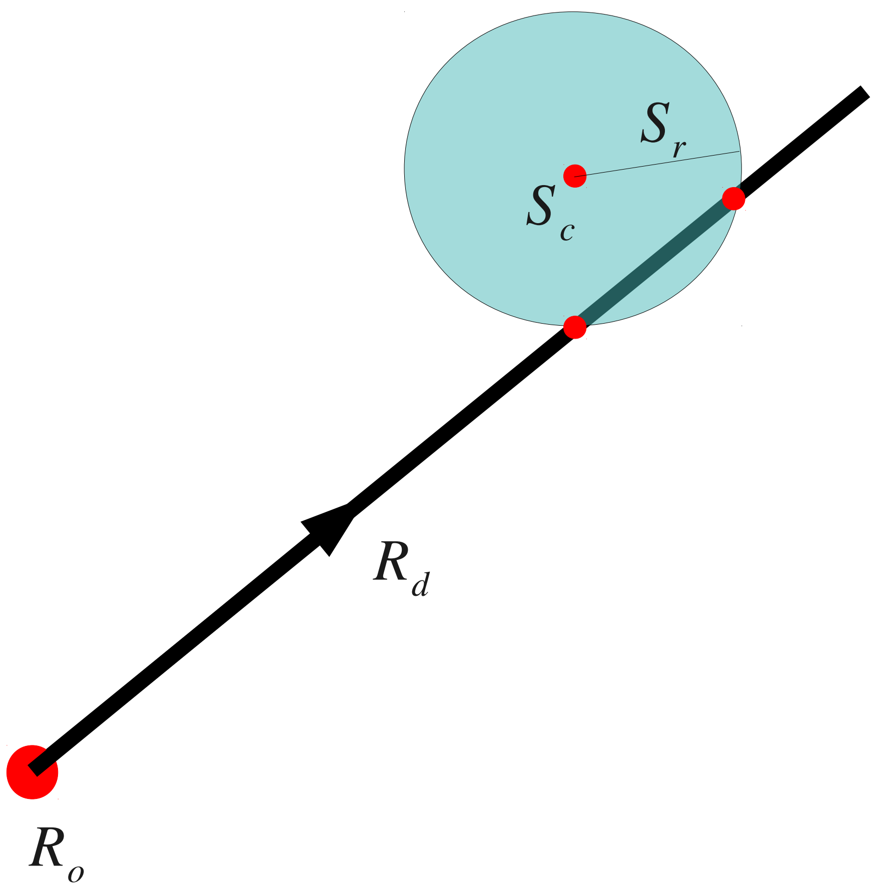
$$A = x_d^2 + y_d^2 + z_d^2 = 1$$

$$B = 2\mathrm{x}_d(x_o - x_c) + y_d(y_o - y_c) + z_d(z_o - z_c)$$

$$C = (x_o - x_c)^2 + (y_o - y_c)^2 + (z_o - z_c)^2 - S_r^2$$

Ray Tracing

- Ray - Sphere Intersection



Solving for t

$$t_0 = \frac{-B - \sqrt{B^2 - 4AC}}{2A}$$

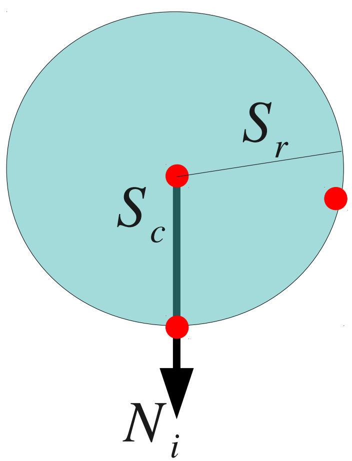
$$t_1 = \frac{-B + \sqrt{B^2 - 4AC}}{2A}$$

Smallest positive value among these two intersections is the closest intersection point.

$$(x_i, y_i, z_i) = (x_0 + t\, x_d,\, y_0 + t\, y_d,\, z_0 + t\, z_d)$$

Ray Tracing

- Ray - Sphere Intersection



The normal at the point of intersection is given by:

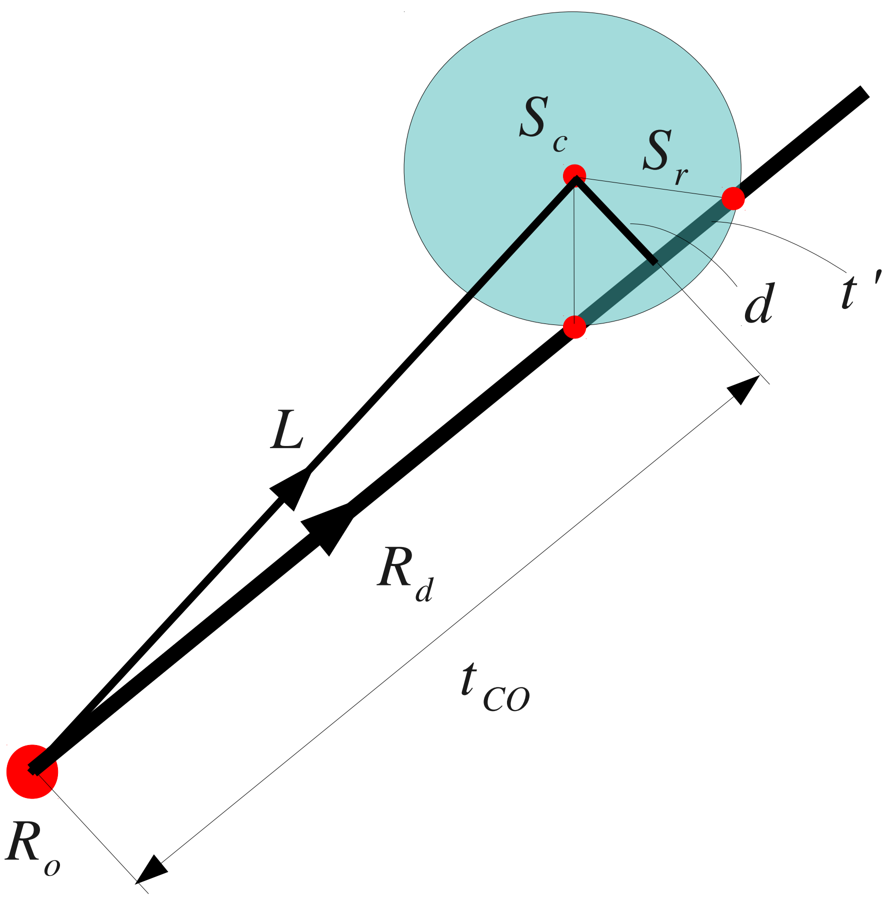
$$N_i = \left(\frac{(x_i - x_c)}{S_r}, \frac{(y_i - y_c)}{S_r}, \frac{(z_i - z_c)}{S_r}\right)$$

The steps are:

·Calculate A, B, C
·Compute the discriminant
·Calculate minimum t.
·Compute intersection point.
·Compute the normal.

Ray Tracing

- Ray - Sphere Intersection



Geometrically:

$$L = S_c - R_o$$

$$t_{CO} = L^T R_d$$

If $t_{CO} < 0$ then no intersection.

$$d^2 = L L^T - t_{CO}^2$$

If $d > S_r$ then no intersection.

$$t' = \sqrt{S_r^2 - d^2}$$

Then the two intersections are given by $t_0 = t_{CO} - t'$ and $t_1 = t_{CO} + t'$

Ray Tracing

- Ray - Plane Intersection

Ray $R(t) = R_0 + tR_d, \qquad t > 0$

Plane $P: Ax + By + Cz + D = 0$

$$A^2 + B^2 + C^2 = 1$$

Plane Normal $P_n = (A, B, C)$

D: Distance from the origin

Ray Tracing

- Ray - Plane Intersection

Substituing

$$A(x_o+tx_d)+B(y_o+ty_d)+C(z_o+tz_d)+D=0$$

Solving:

$$t=\frac{-(Ax_o+By_o+Cz_o+D)}{Ax_d+By_d+Cz_d}=\frac{-(P_n.R_o+D)}{P_n.R_d}$$

If:

$$V_d=Ax_d+By_d+Cz_d=P_n.R_d$$

Now, if $V_d=0$ then ray is parallel to the plane (i.e., no intersection).

Now, if $V_d>0$ then normal is pointing away from the ray. Can be used for backface culling.

Ray Tracing

- Ray - Plane Intersection

Substituing

$$A(x_o+tx_d)+B(y_o+ty_d)+C(z_o+tz_d)+D=0$$

Solving:

$$t=\frac{-(Ax_o+By_o+Cz_o+D)}{Ax_d+By_d+Cz_d}=\frac{-(P_n.R_o+D)}{P_n.R_d}$$

If:

$$V_o=-(Ax_o+By_o+Cz_o+D)=-(P_n.R_o+D) \text{ then } t=\frac{V_o}{V_d}$$

If $t<0$ then plane is behind ray's origin, else compute intersection.

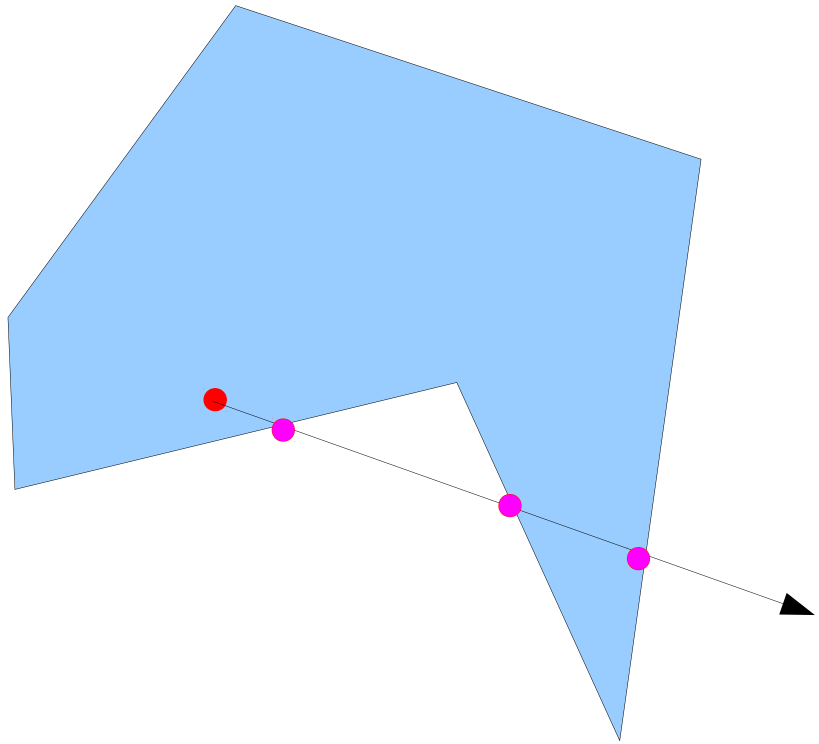
$$(x_i, y_i, z_i)=(x_0+t\,x_d, y_0+t\,y_d, z_0+t\,z_d)$$

$$N_i=P_n$$

Ray Tracing

- Ray - Polygon Intersection

Do a Ray-Plane intersection and then check for containment in the polygon.



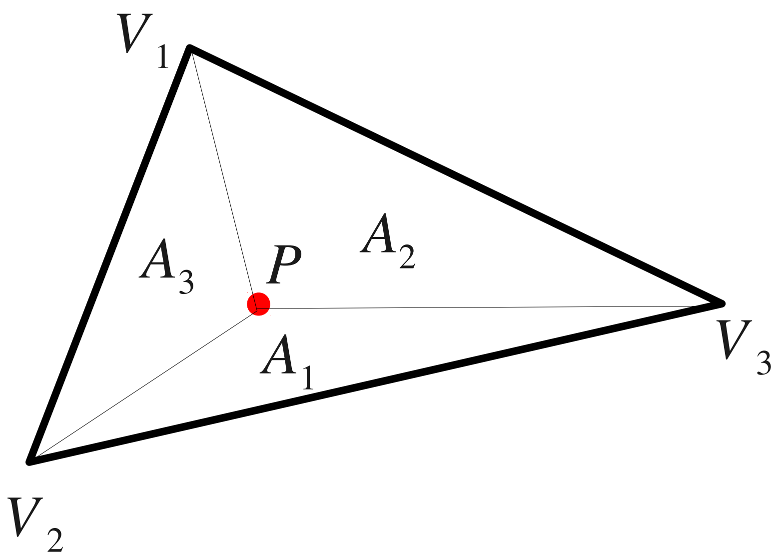
Shoot a ray in any direction from the intersection point in the plane of the polygon.

If number of intersections with the polygon boundary are odd then the point is contained in the polygon

Ray Tracing

- Ray - Triangle Intersection

Do a Ray-Plane intersection and then check for containment in the triangle.



The point of intersection can be written in Barycentric Coordinates as:

$$P = u V_1 + v V_2 + w V_3$$

where

$$u = \frac{A_1}{A}, v = \frac{A_2}{A}, w = \frac{A_3}{A}$$

and

$$A = A_1 + A_2 + A_3$$

The point lies inside the triangle if $u + v + w = 1$
and $u \geq 0, v \geq 0, w \geq 0$

Ray Tracing

- Ray – Quadric Intersection
  - Cylinders, Cones, Sphere, Ellipsoids, Paraboloids

Implicit form of a general quadric is given by:

$$F(x, y, z) = Ax^2 + 2\mathrm{Bxy} + 2\mathrm{Cxz} + 2\mathrm{Dx} + Ey^2 + 2\mathrm{Fyz} + 2\mathrm{Gy} + Hz^2 + 2\mathrm{Iz} + J = 0$$

Ray: $R(t) = R_0 + tR_d, \qquad t > 0$

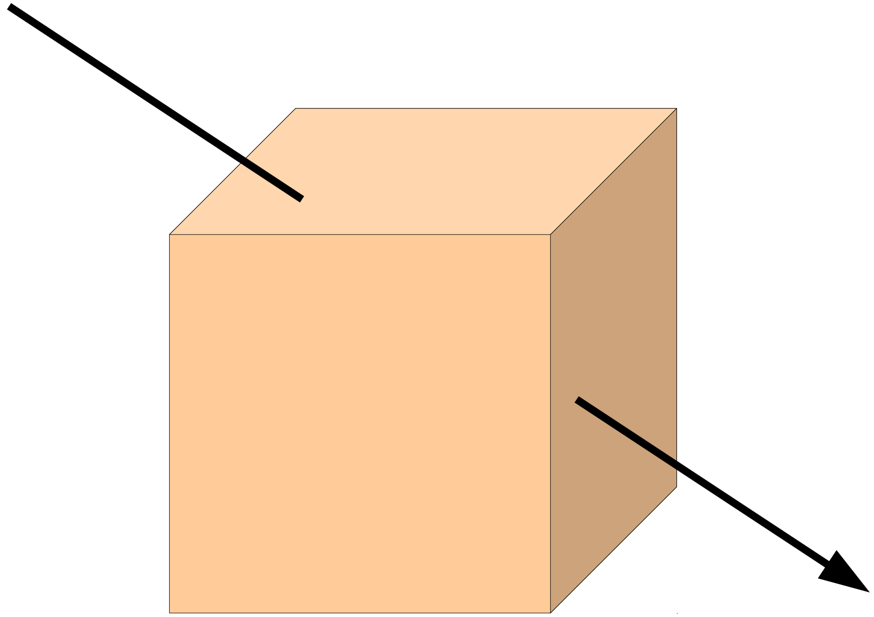
Substitute and solve the quadratic in t

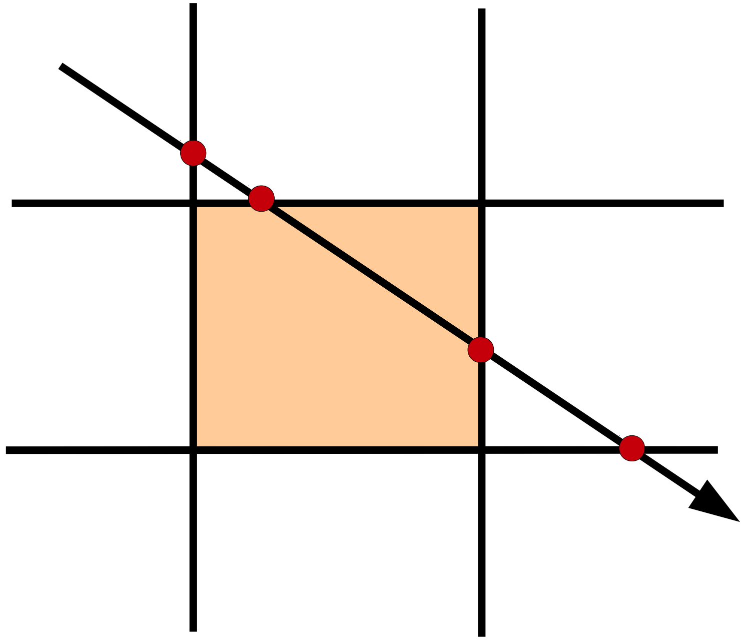
Normal at the point of intersection:

$$N_i = \left(\frac{\partial F}{\partial x_i}, \frac{\partial F}{\partial y_i}, \frac{\partial F}{\partial z_i}\right) \qquad \Rightarrow \qquad \begin{aligned} N_{ix} &= 2(Ax_i + By_i + CZ_i + D) \\ N_{iy} &= 2(Bx_i + Ey_i + FZ_i + G) \\ N_{iz} &= 2(Cx_i + Fy_i + HZ_i + I) \end{aligned}$$

Ray Tracing

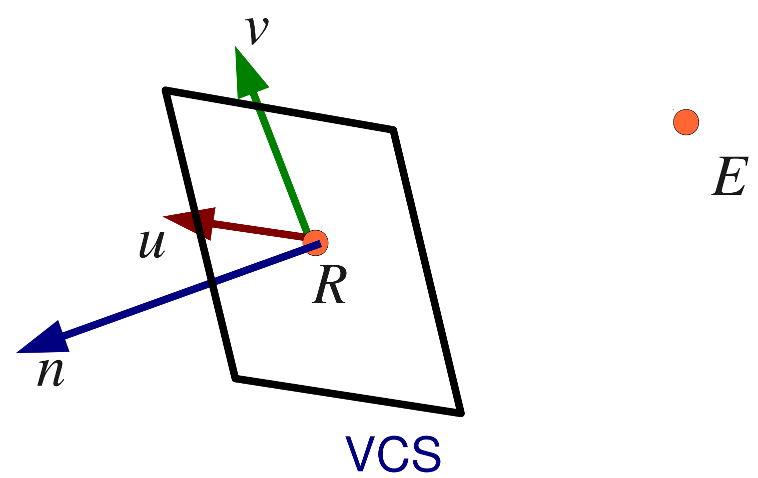
- Ray – Box Intersection
  - User Cyrus Beck/Liang Barsky in 3D

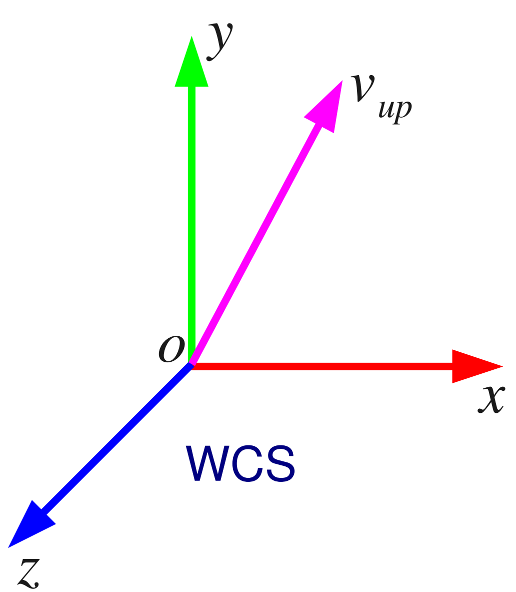




Ray Tracing

- Setting up
  - Viewing





- In WCS
  - Position of VRP, R
  - Normal to View Plane, n
  - Up Vector, v_{up}
- In VCS
  - Extent of window
  - Position of Eye, E

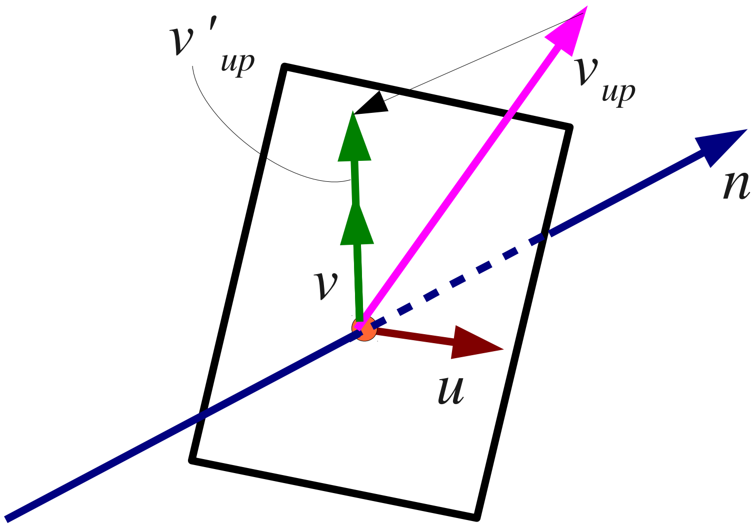
Ray Tracing

- Setting up
  - Viewing

If $v'_{up} = v_{up} - (v_{up}.n)n$

then we define

$$v = \frac{v'_{up}}{\|v'_{up}\|} \quad \text{and} \quad u = v \times n$$



Ray Tracing

- Setting up
  - Viewing

If a point P has coordinates

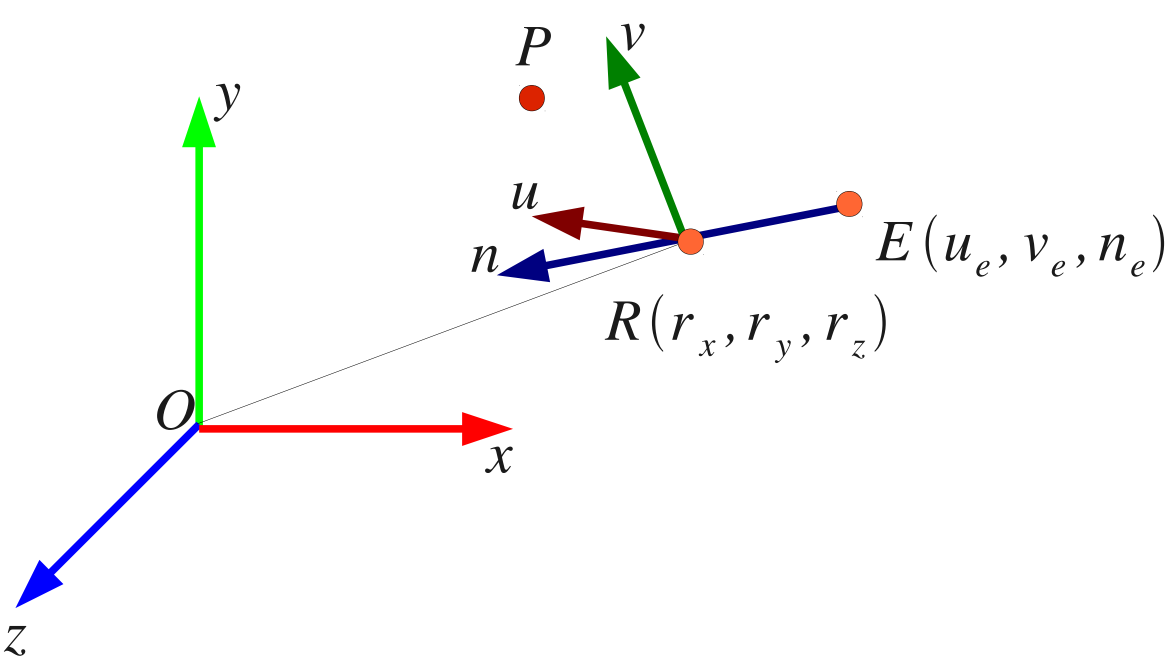
(x, y, z) in WCS

(a, b, c) in VCS

then -

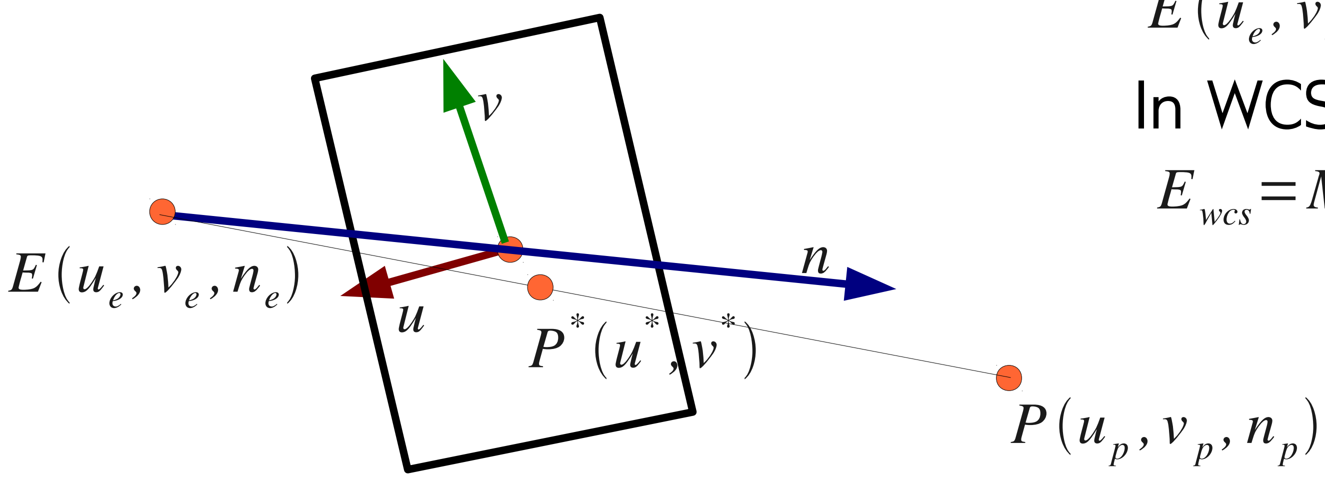
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = M \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} + R$$

$$\text{where}, M = \begin{bmatrix} u_x & v_x & n_x \\ u_y & v_y & n_y \\ u_z & v_z & n_z \end{bmatrix}$$



Ray Tracing

- Setting up
  - Viewing



- Define VRP, $R(r_x, r_y, r_z)$
- Compute M from u, v, n
- Define eye in VCS

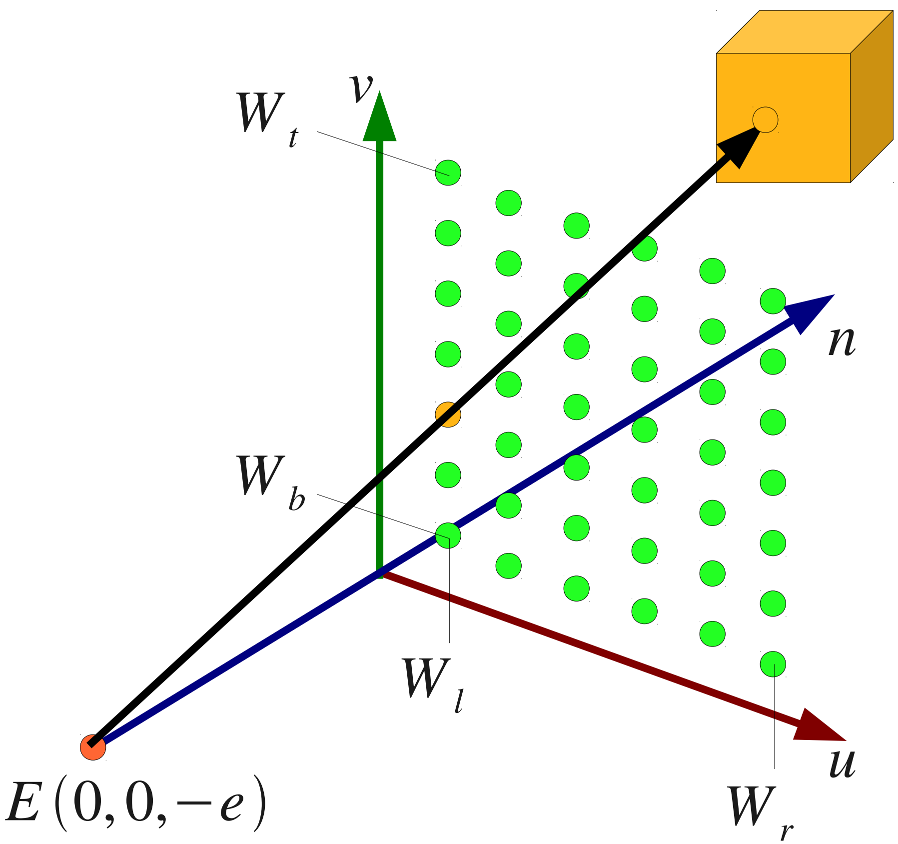
$$E(u_e, v_e, n_e) = (0, 0, -e)$$

In WCS, the eye becomes

$$E_{wcs} = M.E + R$$

Ray Tracing

- Setting up
  - Window



- Define window size

$$W_l, W_r, W_t, W_b$$

$$u_i^* = W_l + i\Delta u \qquad v_i^* = W_t - j\Delta v$$ for pixel (i, j)

where

$$\Delta u = (W_r - W_t)/\mathrm{MAXCOLS}$$
$$\Delta v = (W_t - W_b)/\mathrm{MAXROWS}$$

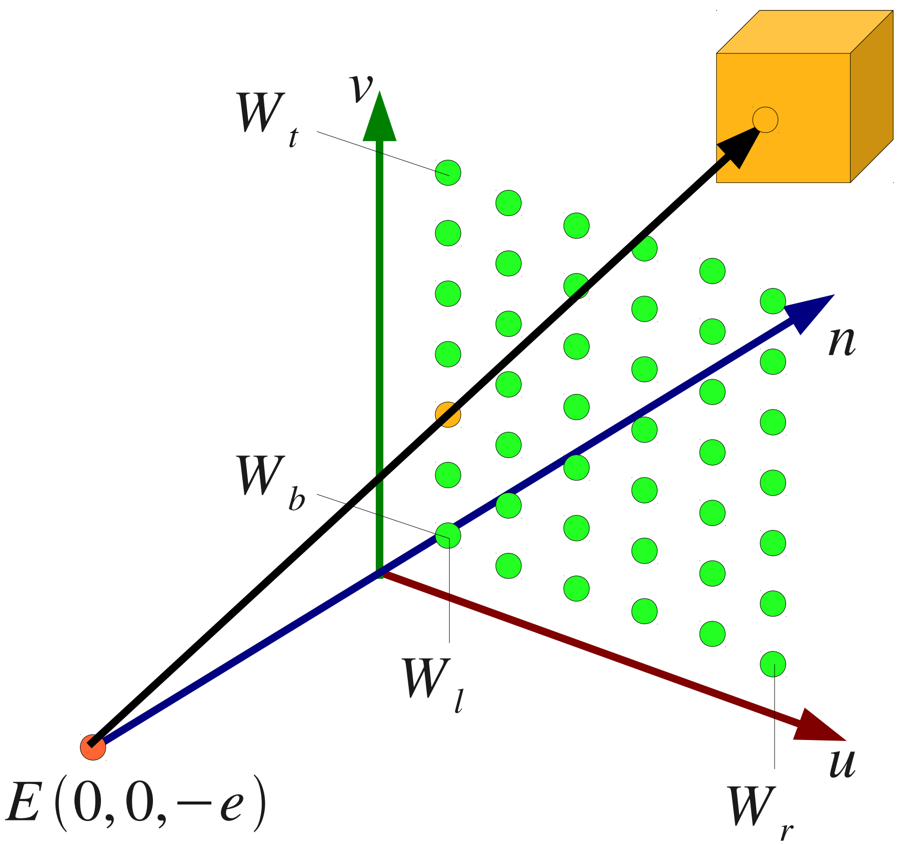
Thus coordinates of pixel (i, j) are

$$P_{ij}^* = (u_i^*, v_j^*, 0)$$

- MAXROWS and MAXCOLS can be used to control the resolution of the image.

Ray Tracing

- Setting up
  - Ray



- $E_{wcs} = M.E + R = M.\begin{bmatrix}0 & 0 & -n_e\end{bmatrix}^T + R$
- Direction of the ray from the eye through pixel at (i,j) in WCS is given by

$$dir_{ij} = M.P_{ij}^* + R - E_{wcs} = M.(P_{ij}^* - E)$$
$$= M.\begin{bmatrix}u_i^* & v_j^* & n_e\end{bmatrix}^T$$

- Ray is through pixel (i, j) given by

$$R_{ij}(t) = E_{wcs} + dir_{ij}t = R_o + R_d t$$

Ray Tracing

- Basic Idea
  - For every pixel in the image
    - Shoot a ray
    - Find closest intersection with object
    - Find normal at the point of intersection
    - Compute illumination at point of intersection
    - Assign pixel color

Ray Tracing

- Transforming objects

  Ray: $s + c\,t$

  Objects: Sphere, Cone, Cylinder, Box

  We assume the objects to be normalized so that the ray-object intersection is easier:

  for e.g., Sphere: $x^2 + y^2 + z^2 = 1$

Ray Tracing

- Transforming objects

  Ray: $s + c\,t$

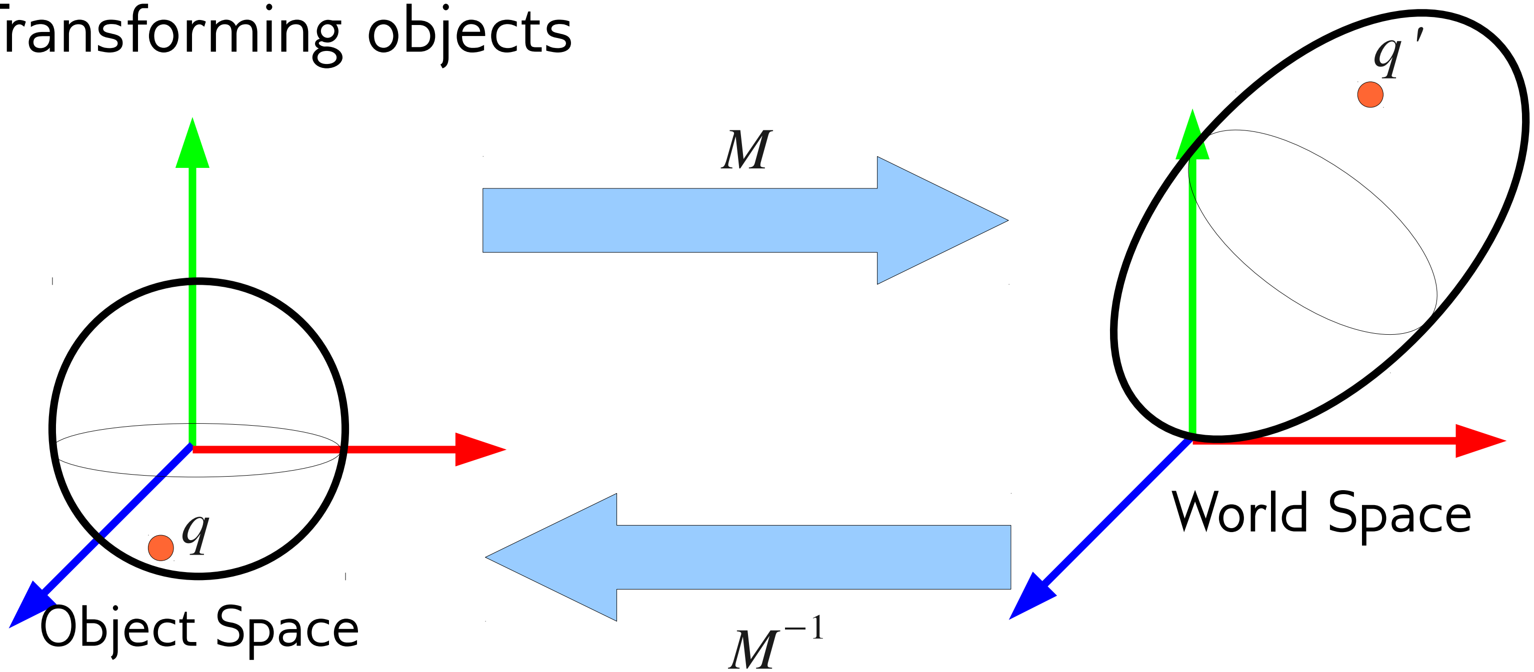
  Objects: Sphere, Cone, Cylinder, Box

  Then we replicate the normalized objects and transform them to create variety in the scene.

  Can ray-object intersections be done on the transformed objects?

Ray Tracing

- Transforming objects

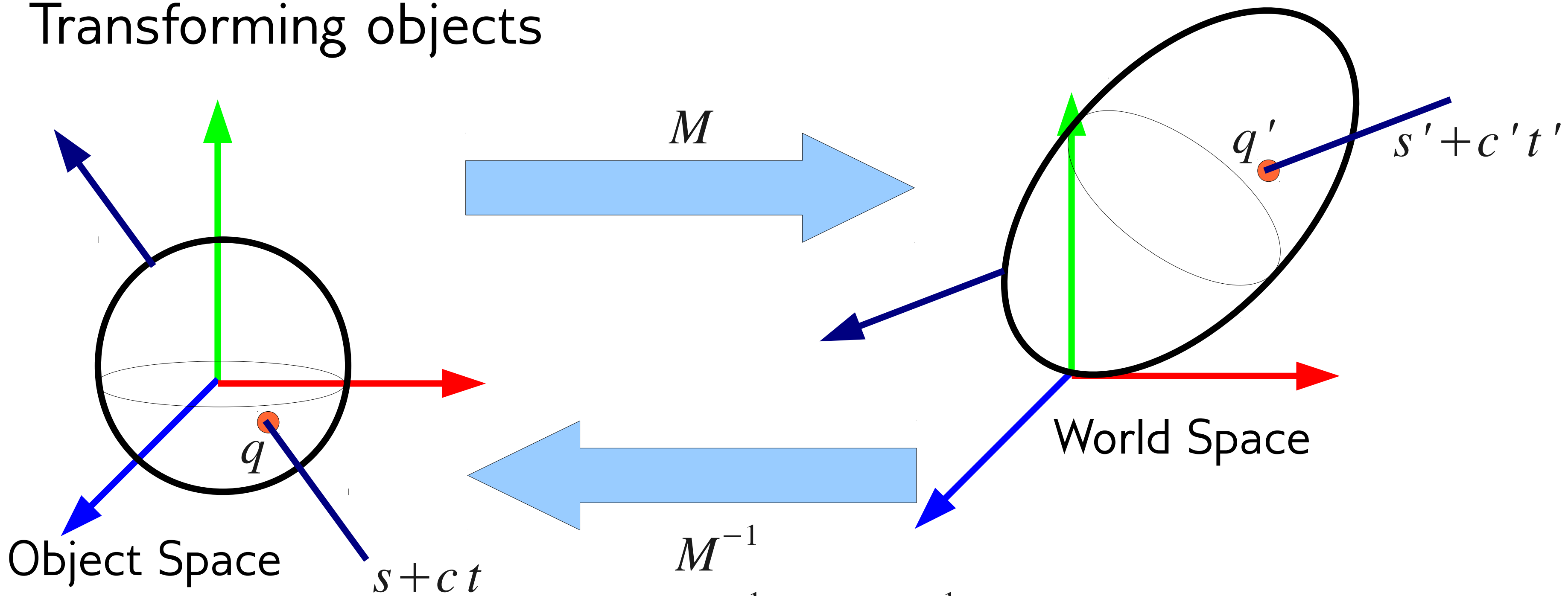


A normalized sphere is transformed under an affine transformation, i.e.,

$$q' = M.q \Leftrightarrow q = M^{-1}.q'$$

Ray Tracing

- Transforming objects

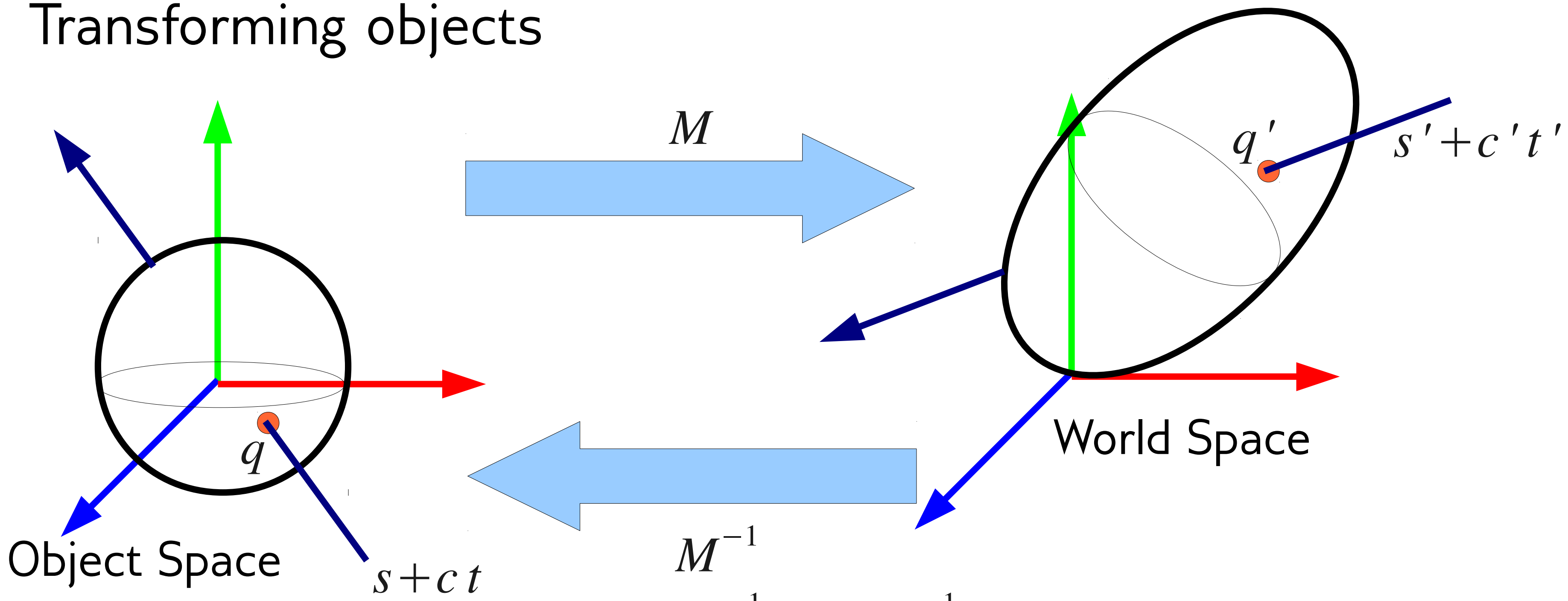


$$s+ct=M^{-1}.s'+M^{-1}.c't'$$

$$\begin{bmatrix} s_x \\ s_y \\ s_z \\ 1 \end{bmatrix} = \begin{bmatrix} a & d & g & l \\ b & e & h & m \\ c & f & i & n \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s'_x \\ s'_y \\ s'_z \\ 1 \end{bmatrix} = \begin{bmatrix} as'_x+ds'_y+gs'_z+l \\ bs'_x+es'_y+hs'_z+m \\ cs'_x+fs'_y+is'_z+n \\ 1 \end{bmatrix}$$

Ray Tracing

- Transforming objects

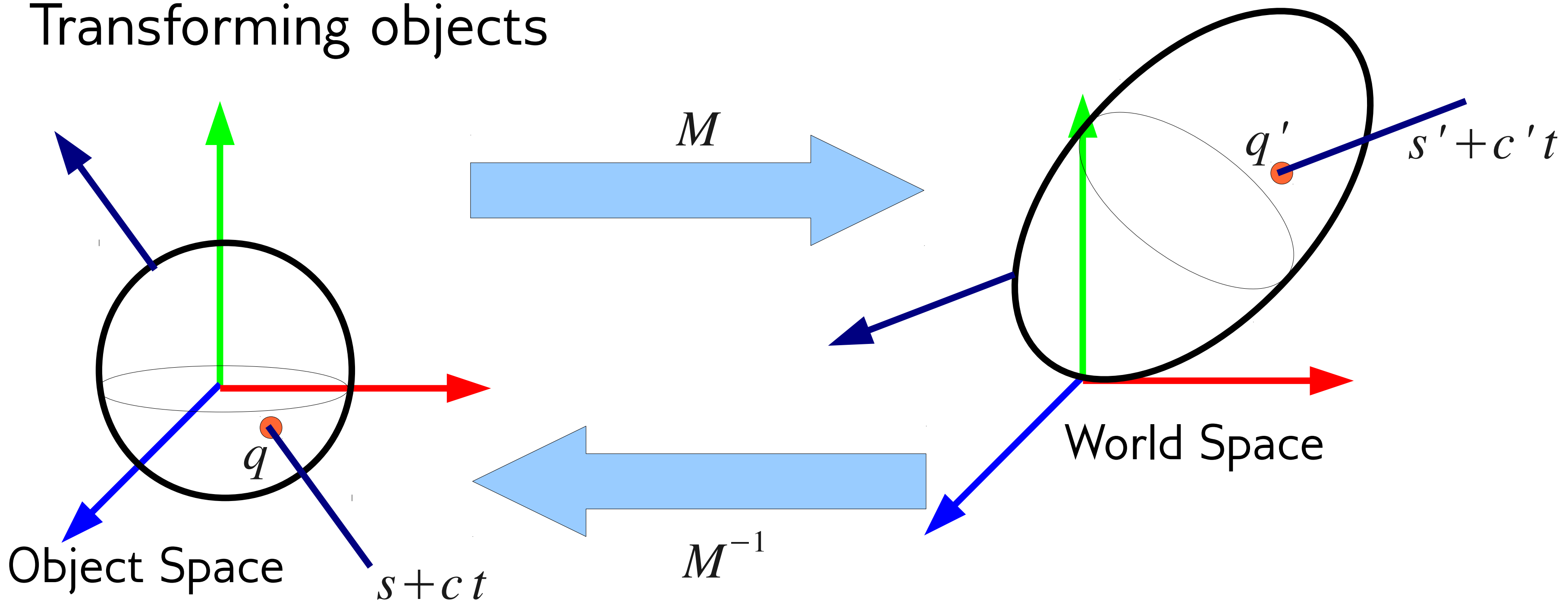


$$s+ct=M^{-1}.s'+M^{-1}.c't'$$

$$\begin{bmatrix} c_x \\ c_y \\ c_z \\ 0 \end{bmatrix} = \begin{bmatrix} a & d & g & l \\ b & e & h & m \\ c & f & i & n \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c'_x \\ c'_y \\ c'_z \\ 0 \end{bmatrix} = \begin{bmatrix} ac'_x+dc'_y+gc'_z+l \\ bc'_x+ec'_y+hc'_z+m \\ cc'_x+fc'_y+ic'_z+n \\ 0 \end{bmatrix}$$

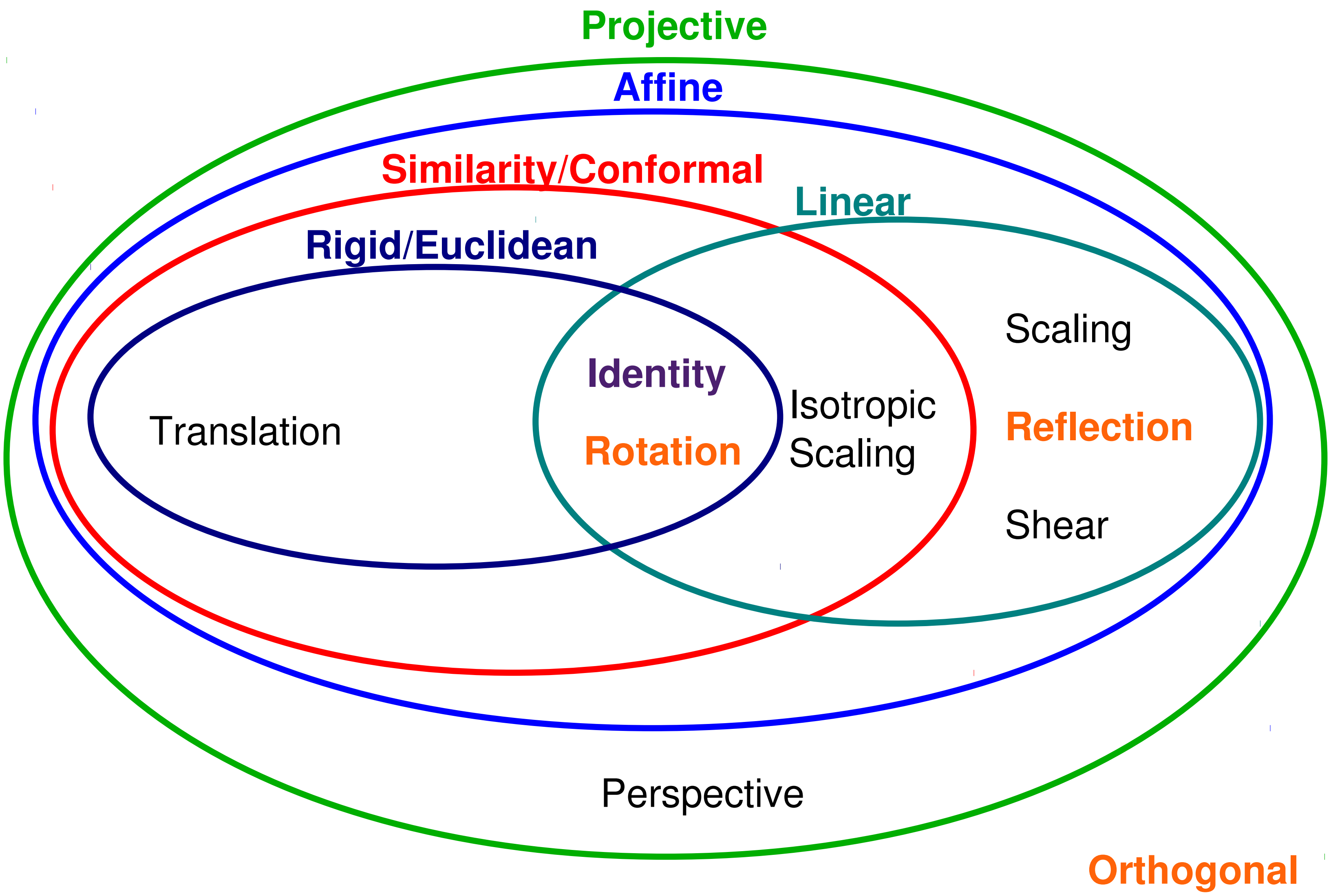
Ray Tracing

- Transforming objects



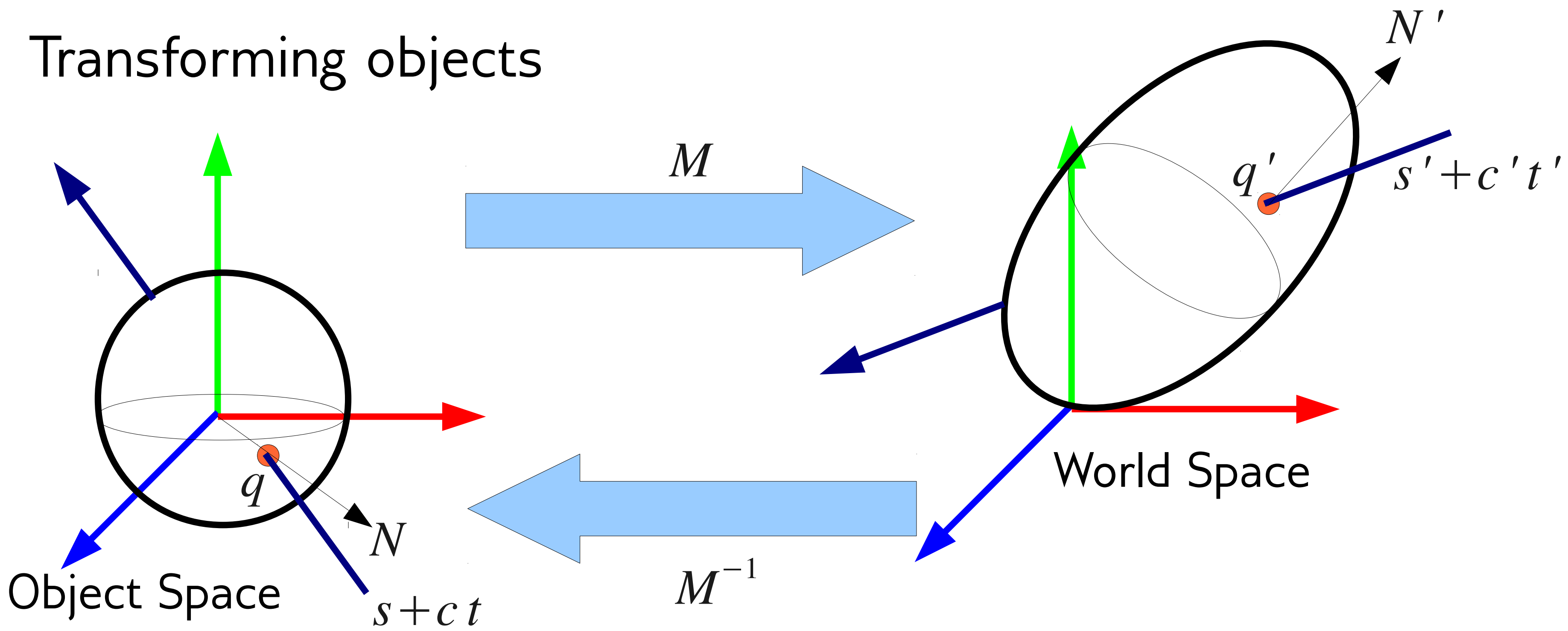
If M includes a scaling then c is not a unit vector after the transformation. If c is renormalized then scale the t value accordingly.

A side note on transformations



Ray Tracing

- Transforming objects



We also need the normal at the point of intersection. How does the normal transform when the object undergoes an affine transformation?