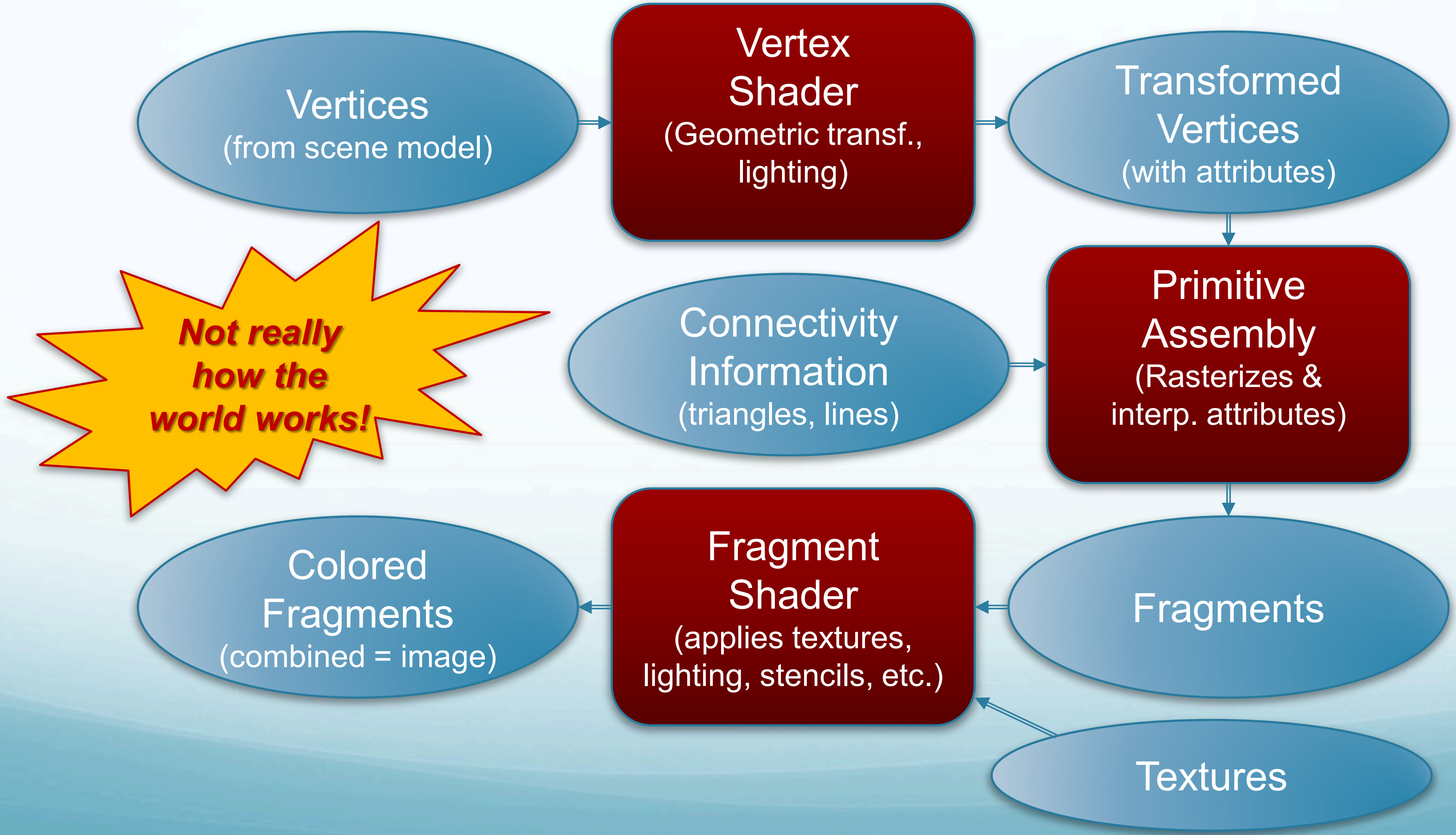
CSC 240 Computer Graphics
Video 18: Ray Tracing Part 1

Nick Howe
Smith College

Some slides & content courtesy Sara Mathieson

Traditional Rendering

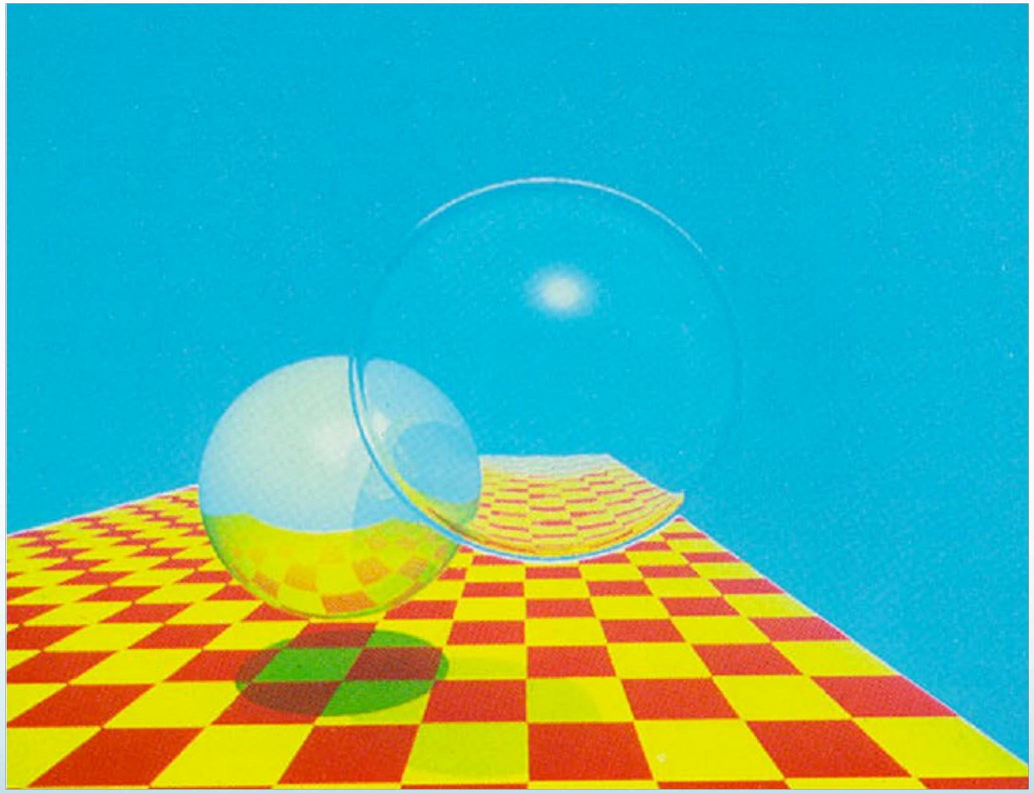
Traditional rendering pipeline draws polygonal surfaces



What if we approached graphics more like a physics simulation?

Ray Tracing

Completely different approach: follow rays of light



Turner Whitted, *An Improved Illumination Model for Shaded Display*, SIGGRAPH 1979

Ray Tracing

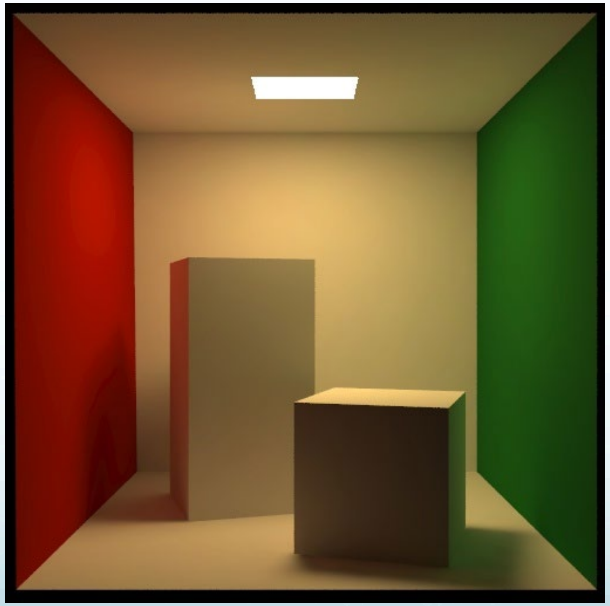
https://youtu.be/k12cf15VvV4

Wikipedia (Gilles Tran)

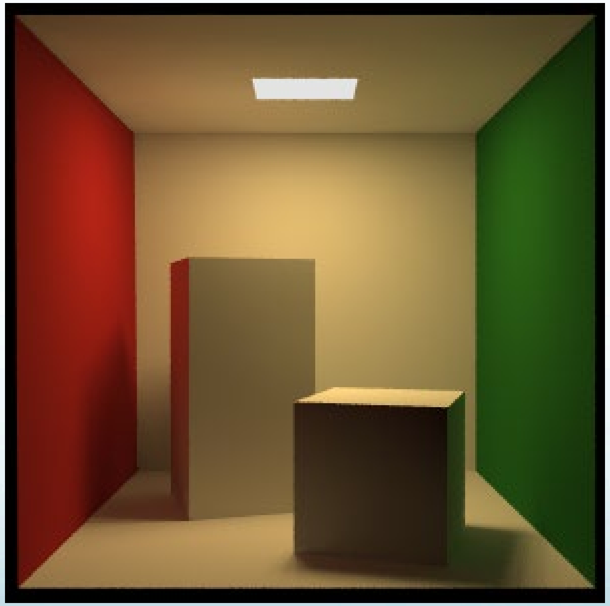
Ray Tracing

The Cornell Box experiment measures verisimilitude

- Render scene with known properties, compare to photo



Real



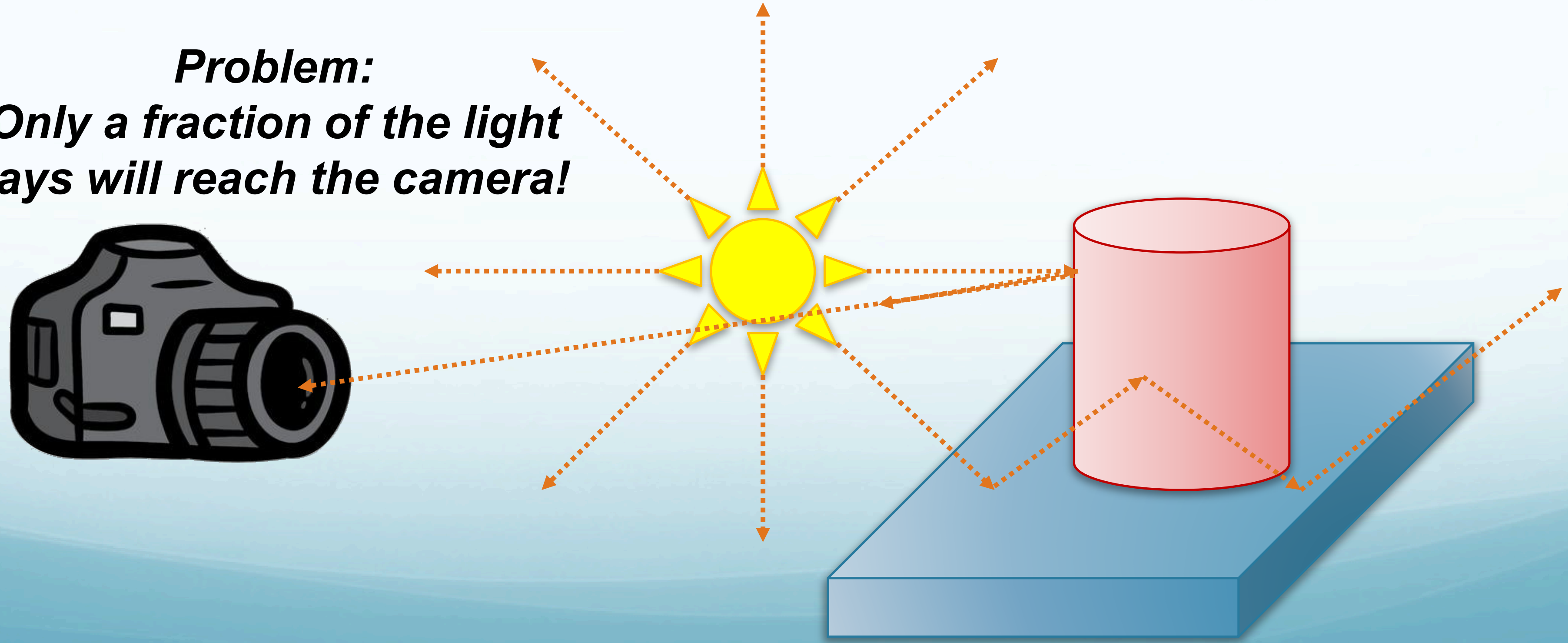
POV-Ray

Ray Tracing

How could we model the physics of image formation?

- Light emitted from source in all directions
- Reflects off objects until reaches eye/camera

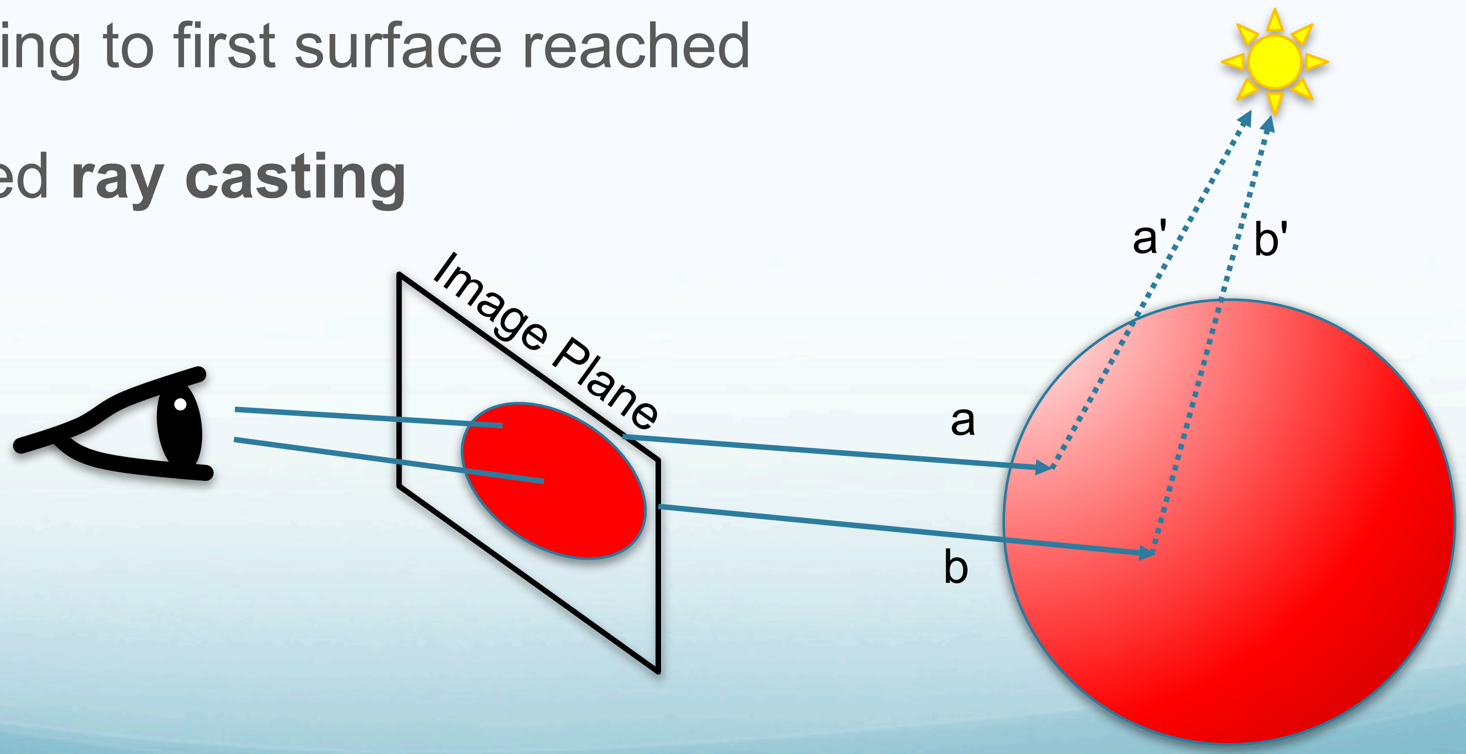
Problem:
Only a fraction of the light rays will reach the camera!



Ray Tracing

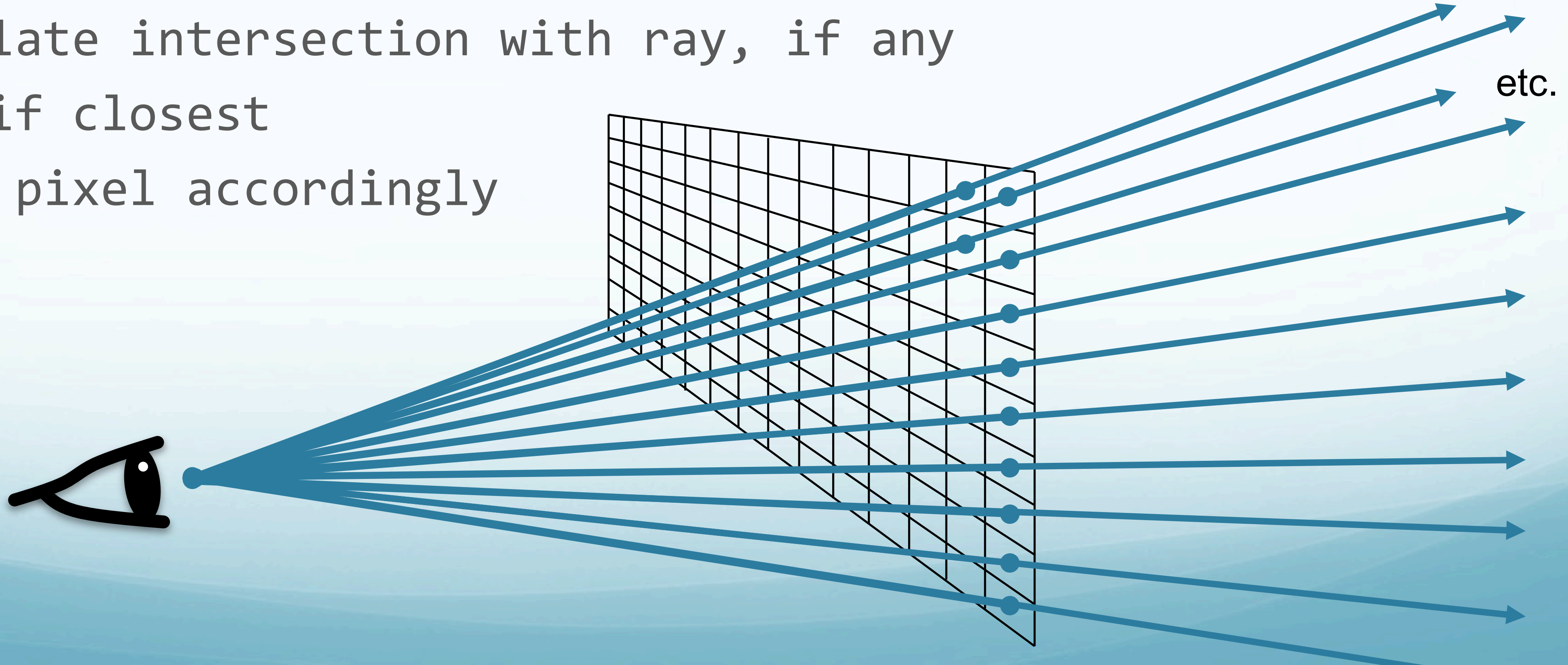
Solution: start from the eye, follow rays backwards

- Loop over pixels: ray from focal point through pixel
- Color according to first surface reached
- Process called **ray casting**



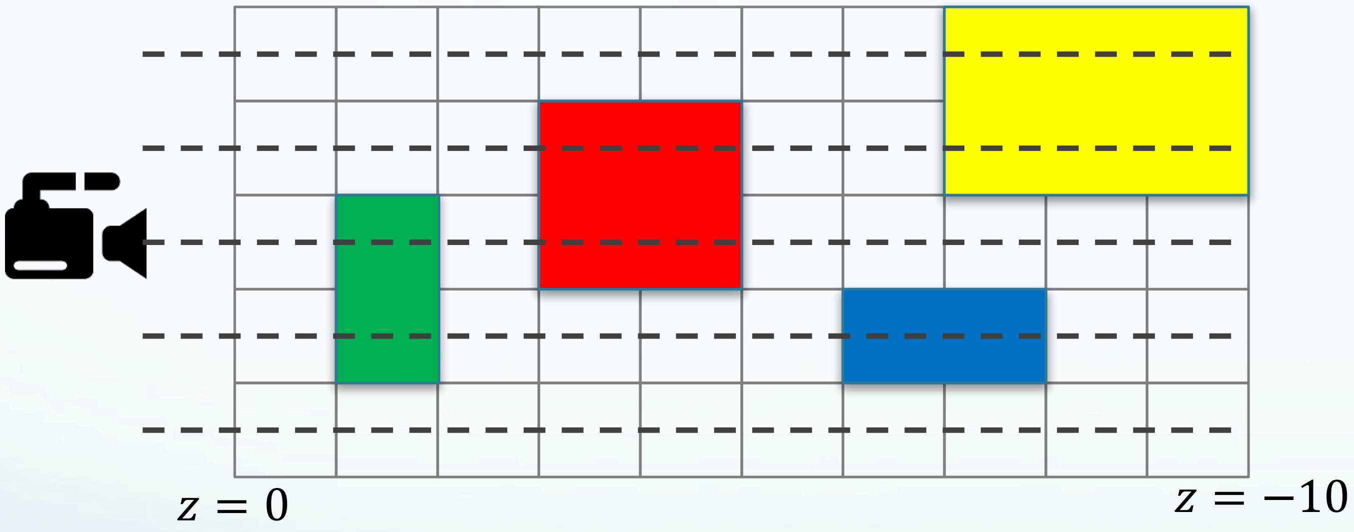
Simple Ray Casting Algorithm

```
Loop over all pixels
  Create ray from eye through pixel center
  Loop over all world objects
    Calculate intersection with ray, if any
    Keep if closest
    Color pixel accordingly
```



Simple Ray Casting Demo

For this demo, use orthographic projection & order red-yellow-green-blue



Render:

Ray #1 intersections:
R: no; Y: -7; G: no; B: no
➔ Color is Yellow

Ray #2 intersections:
R: -3; Y: -7; G: no; B: no
➔ Color is Red

Ray #3 intersections:
R: -3; Y: no; G: -1; B: no
➔ Color is Green

Ray #4 intersections:
R: no; Y: no; G: -1; B: -6
➔ Color is Green

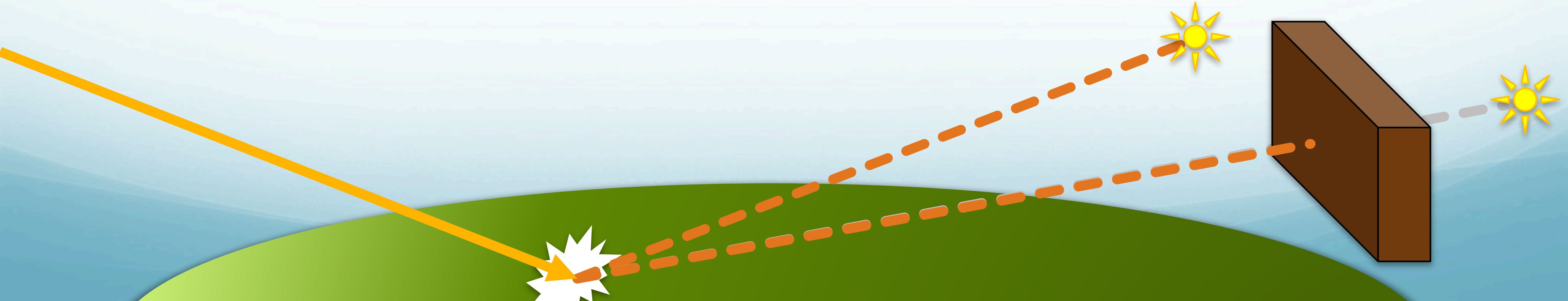
Ray #5 intersections:
R: no; Y: no; G: no; B: no
➔ Color is Black

Color Determination

So we've determined that a light ray hit a surface. What color is the pixel?

- Depends on lighting. As before:
  - Diffuse reflection of specific light sources
  - Specular reflection of specific light sources
  - Ambient light reflection
  - Emission of light from surface

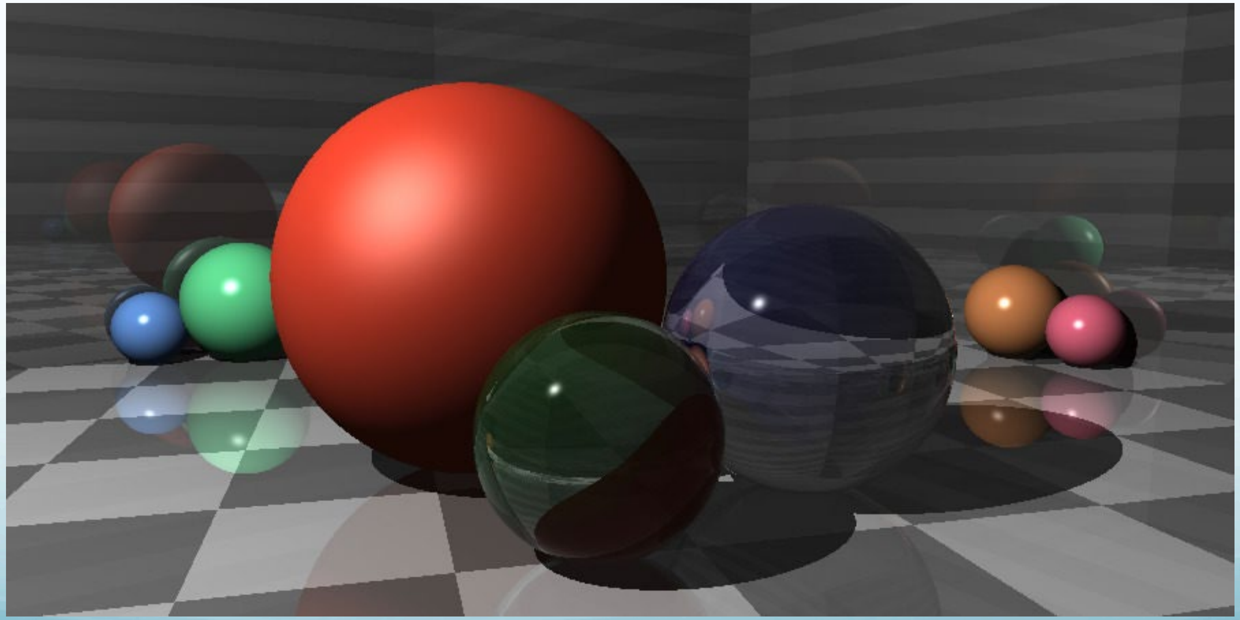
These require computing angle to each light source, and possibly further ray tracing to check for intervening objects (shadows)



Optical Effects

Ray tracing allows computation of advanced optical effects.

- Light can reflect off one object onto another
- Light can refract as it passes through transparent objects

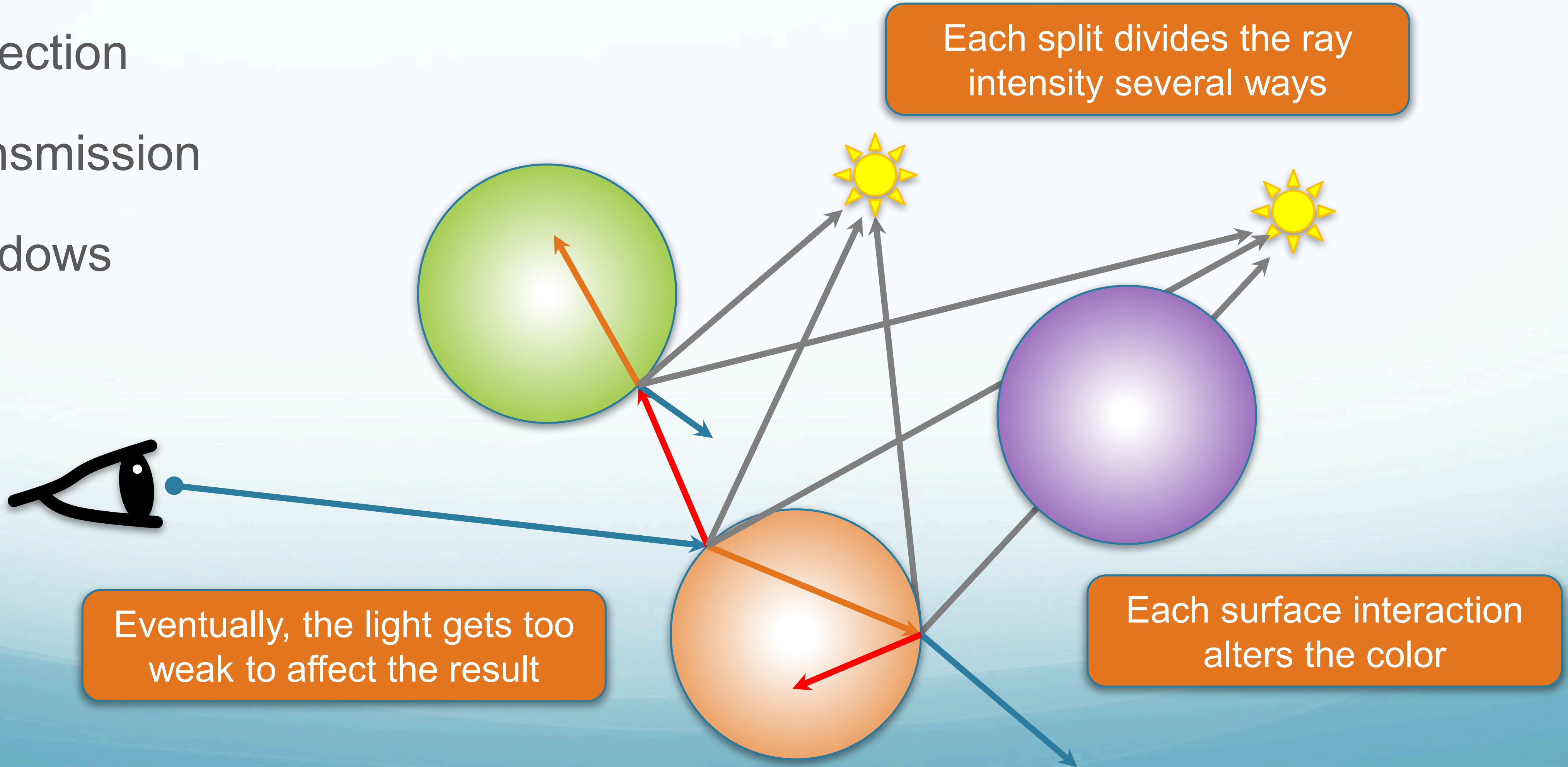


https://forum.raytracerchallenge.com/thread/4/reflection-refraction-scene-description

Recursive Ray Tracing

Each surface interaction splits a ray multiple ways:

- Reflection
- Transmission
- Shadows



Questions

PAUSE NOW & ANSWER

1. **Why do we trace rays from the camera instead of the light source?**

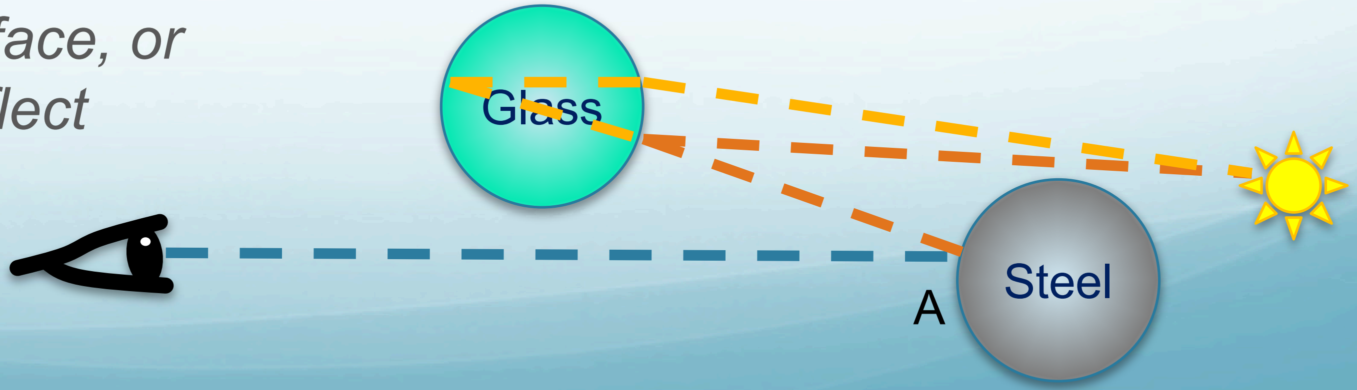
   To avoid wasting work on light that is never seen by the camera.

2. **In the scene below, which of the following might contribute to the color of point A under simple ray tracing? diffuse, specular, emissive, ambient.**

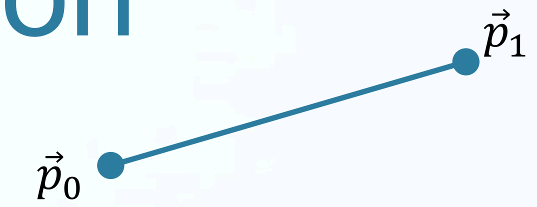
   Since the point is shadowed, only ambient and emissive.

3. **In the scene below, how might light reach point A under recursive ray tracing with a maximum of three levels of secondary rays?**

   It might reflect off the glass surface, or possibly refract through and reflect from the inner surface.



Ray Representation



We can represent light rays parametrically.

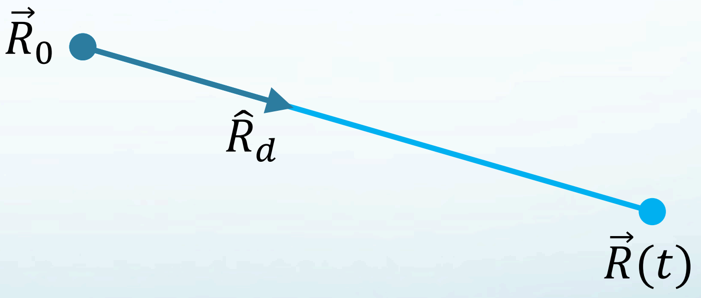
- Recall 1st order Bézier curve: weighted sum of 2 points

$$\vec{p} = (1-t)\vec{p}_0 + t\vec{p}_1$$
$$\vec{p} = \vec{p}_0 + t(\vec{p}_1 - \vec{p}_0)$$

Rearrange and group terms

- Suggests a formulation: origin + scaled direction vector

- $\vec{R}(t) = \vec{R}_0 + t\hat{R}_d$
  - $\vec{R}_0$ is starting point (i.e., camera position)
  - $\hat{R}_d$ is unit vector in travel direction of ray
  - Parameter t is distance traveled along ray
  - Scale by distance from focal point to image plane

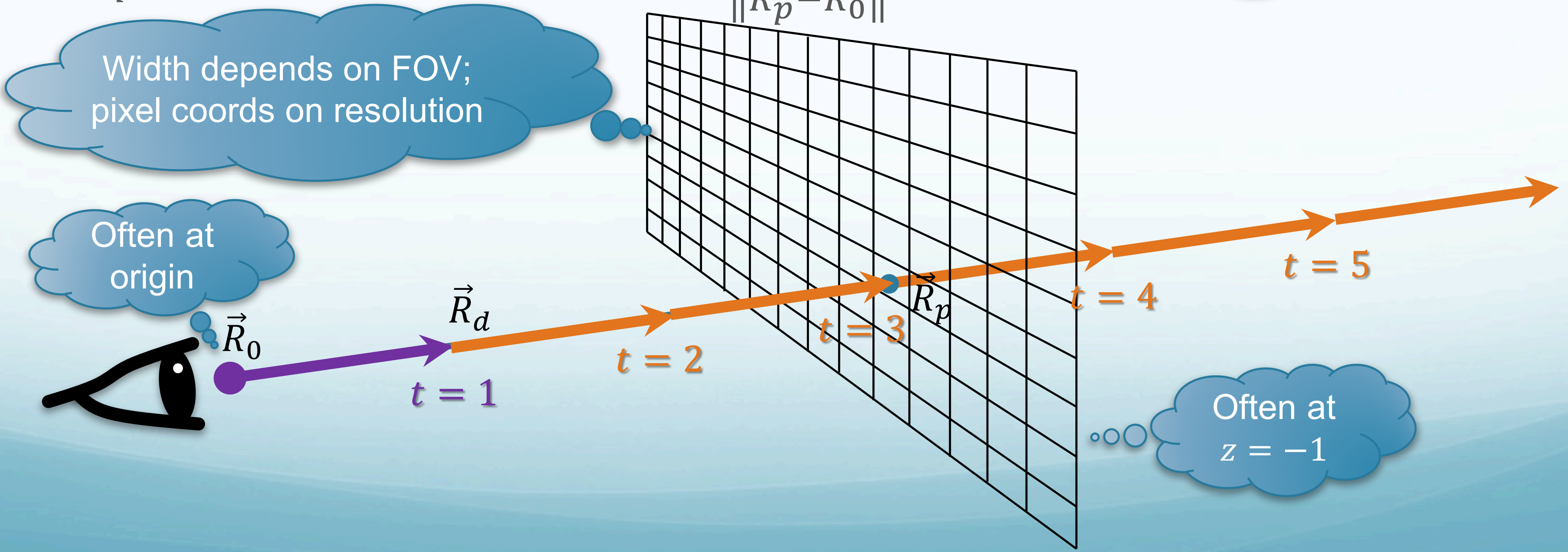


Math of Ray Casting

Ray tracing vector math: $\vec{R}(t) = \vec{R}_0 + t\hat{R}_d$

- Let $\vec{R}_0$ be the focal point, $\vec{R}_p$ the pixel center
- Scale $\vec{R}_p - \vec{R}_0$ to unit length: $\vec{R}_d = \frac{\vec{R}_p - \vec{R}_0}{\|\vec{R}_p - \vec{R}_0\|}$

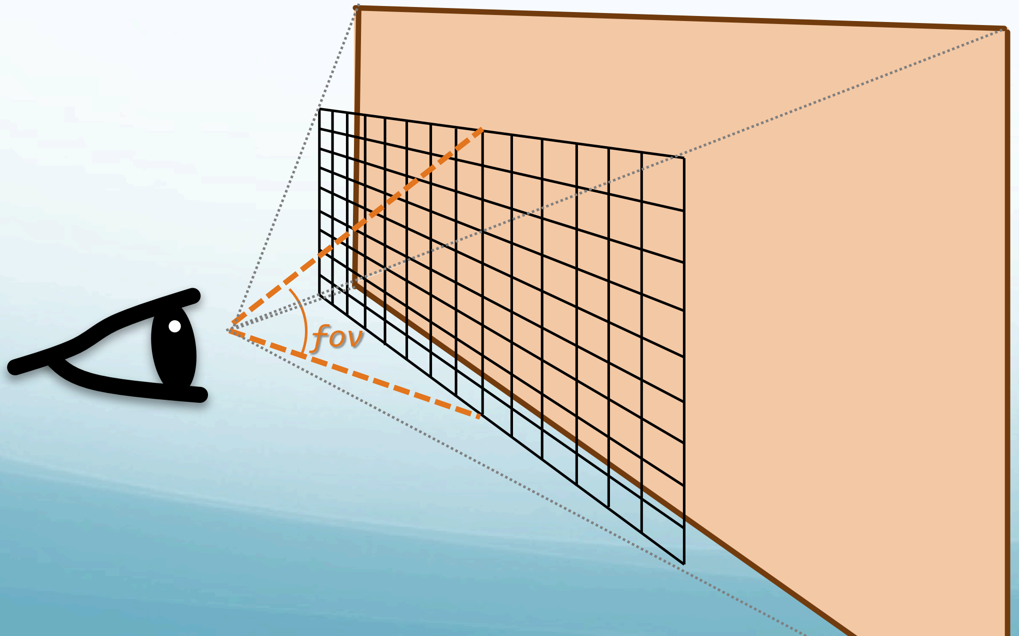
Requires specifics on position of focal point, view plane, # pixels, etc.

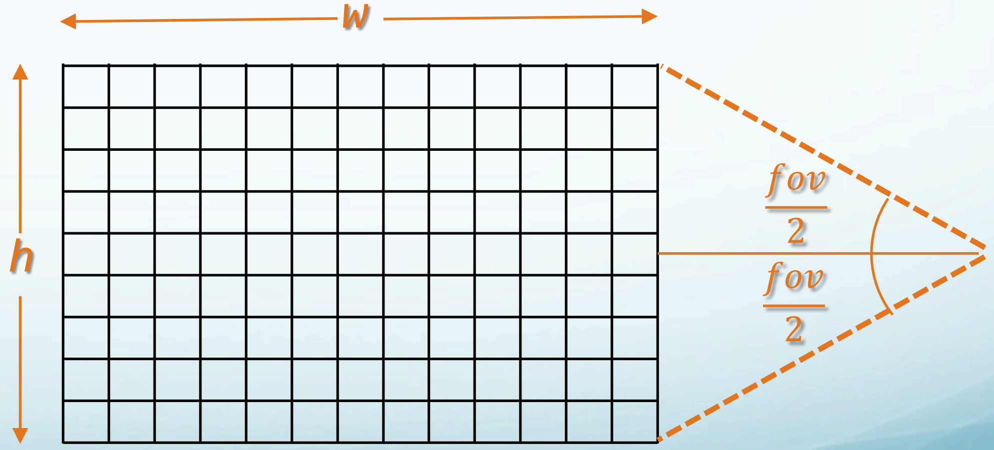


Ray Casting Pixel Math

Given a set of camera parameters, how do we find rays?

- Specify: field of view, aspect ratio, clipping planes
- Assume camera at origin, facing negative z axis



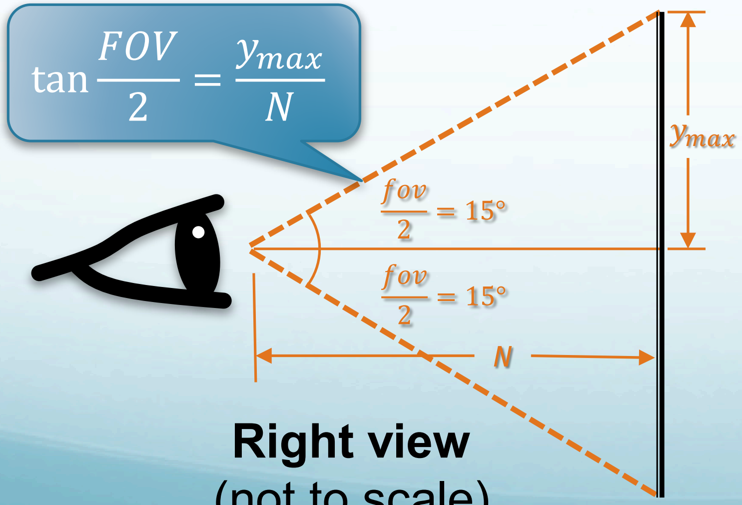


Aspect ratio = $\frac{w}{h}$

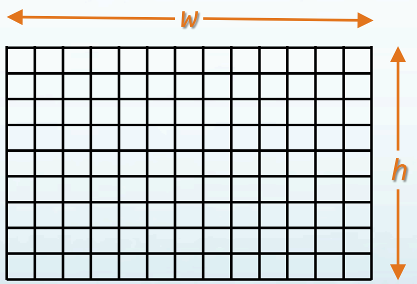
Ray Casting Pixel Math

Example: FOV=30°, AR=2, N=5, F=20, 600×300 pixels, pixel at (150,150).

1. Use FOV & N to compute y_{max} $\quad y_{max} = N \tan \frac{FOV}{2} = 5 \tan 15° = 1.34$
2. Use AR and y_{max} to compute x_{max} $\quad x_{max} = AR \cdot y_{max} = 2 \cdot 1.34 = 2.68$
3. Translate pixel dimensions to viewport dimensions



Right view
(not to scale)



Aspect ratio = $\frac{w}{h}$

$x_{max} = AR \cdot y_{max}$

Pixel (i,j):

$$x = \left(\frac{i}{w}\right) 2x_{max} - x_{max}$$

$$y = \left(\frac{h-j}{h}\right) 2y_{max} - y_{max}$$

Pixel (150,150):→ (-1.34,0,5)

$$x = \frac{150}{600} \cdot 5.36 - 2.68 = -1.34$$

$$y = \frac{150}{300} \cdot 2.68 - 1.34 = 0$$

Ray Casting Pixel Math

Now find the ray equation, given $\vec{R}_0 = (0,0,0)$ and $\vec{R}_p = (-1.34, 0, -5)$

$$\vec{R}_d = \frac{(-1.34,0,-5) - (0,0,0)}{\|(-1.34,0,-5) - (0,0,0)\|} = \frac{(-1.34,0,-5)}{\|(-1.34,0,-5)\|}$$

$$= \frac{(-1.34,0,-5)}{\sqrt{(-1.34)^2+0^2+(-5)^2}}$$

$$= \frac{(-1.34,0,-5)}{\sqrt{26.5}} = \left(\frac{-1.34}{5.18}, \frac{0}{5.18}, \frac{-5}{5.18}\right)$$

$$= (-0.26, 0, -0.97)$$

$$\vec{R}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} -0.26 \\ 0 \\ -0.97 \end{bmatrix}$$

Summary of Key Formulas

- Extent of viewport image plane: $y_{max} = N \tan\frac{FOV}{2}$ and $x_{max} = AR \cdot y_{max}$
- Pixel coordinates to world coordinates $\vec{R}_p = (x, y, z)$:

$$x = \left(\frac{i}{w}\right) 2x_{max} - x_{max} \qquad y = \left(\frac{h-j}{h}\right) 2y_{max} - y_{max} \qquad z = -N$$

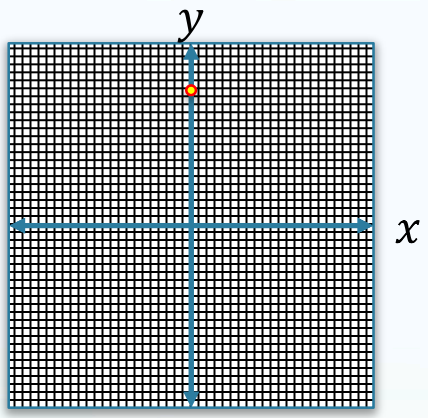
- Ray direction unit vector from focal point $\vec{R}_0$ and pixel coordinates $\vec{R}_p$:

$$\vec{R}_d = \frac{\vec{R}_p - \vec{R}_0}{\left\|\vec{R}_p - \vec{R}_0\right\|}$$

- Final parametric ray equation: $\vec{R}(t) = \vec{R}_0 + t\hat{R}_d$

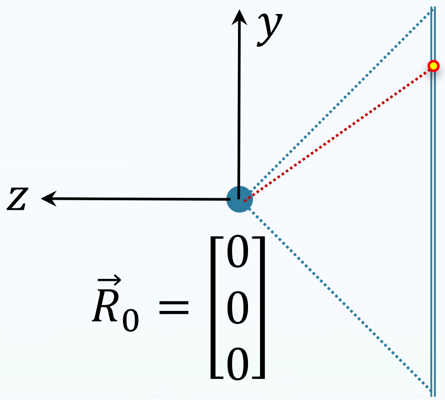
Ray Casting, Example 2

Camera at origin, view plane at $z = -1$, 90º FOV,
80x80 pixels resolution, pixel (40,10)



Front view

Center pixel is (40,40)
Edges are at ± 1 world
(40,10) is at $x = 0, y = 0.75$



$$\vec{R}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Right view

$$\boxed{\vec{R}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} 0 \\ 0.6 \\ -0.8 \end{bmatrix}}$$

$$\vec{R}_p = \begin{bmatrix} 0 \\ 0.75 \\ -1 \end{bmatrix}$$

$$\vec{R}_p - \vec{R}_0 = \begin{bmatrix} 0 \\ 0.75 \\ -1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.75 \\ -1 \end{bmatrix}$$

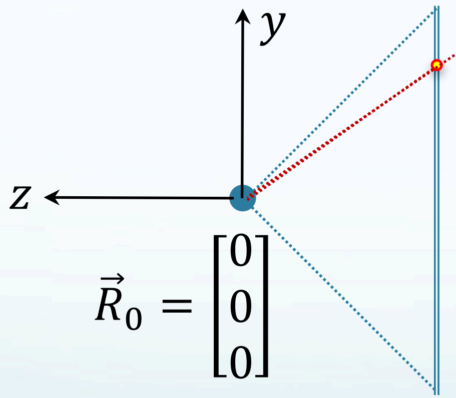
$$\left\| \vec{R}_p - \vec{R}_0 \right\| = \sqrt{0^2 + 0.75^2 + (-1)^2}$$
$$= \sqrt{0.5625 + 1}$$
$$= 1.25$$

$$\vec{R}_d = \begin{bmatrix} 0 \\ 0.75 \\ -1 \end{bmatrix} / 1.25 = \begin{bmatrix} 0 \\ 0.6 \\ -0.8 \end{bmatrix}$$

Ray Casting, Example 2

Next step is to **intersect** ray with objects in scene

$$\vec{R}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} 0 \\ 0.6 \\ -0.8 \end{bmatrix} = \begin{bmatrix} ? \\ ? \\ -3 \end{bmatrix}$$



$$\vec{R}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Right view

Intersection point

Simple case:

$$-0.8t = -3$$

$$t = \frac{-3}{-0.8} = 3.75$$

$$\vec{R}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + 3.75 \begin{bmatrix} 0 \\ 0.6 \\ -0.8 \end{bmatrix} = \begin{bmatrix} 0 \\ 2.25 \\ -3 \end{bmatrix}$$

More details on these computations coming next week!

e.g., plane at $z = -3$

Questions

PAUSE NOW & ANSWER

1. Suppose that you want to cast a ray from (1,2,3) towards (1,6,3). What is the parametric equation of the ray?

Unit vector is (0,1,0). Equation is: $\vec{R}(t) = (1,2,3) + t(0,1,0)$

2. At what value of t would the ray hit (1,6,3)?

Since the ray travels 1 unit distance for each unit of t, it will take until $t = 4$.

3. Suppose that a canvas has FOV = 90, AR = 1, N = 1, and is 100x100 pixels. What are the coordinates of the center of pixel (10,90)?

$$x_{max} = y_{max} = 1 \tan 45°$$

$$x = \left(\frac{i}{w}\right) 2x_{max} - x_{max} = \left(\frac{10}{100}\right) 2 - 1 = -0.8$$

$$y = \left(\frac{j}{h}\right) 2y_{max} - y_{max} = \left(\frac{90}{100}\right) 2 - 1 = 0.8$$

(-0.8,0.8,-1)

Review

After watching this video, you should be able to…

- Define ray tracing and describe how it differs from ordinary rendering
- Give pseudocode & computations for a simple raycasting algorithm
- Understand recursive ray tracing and how to follow light rays in a scene
- Compute the formula for the ray that travels between two given endpoints
- Compute the intersection between a ray and a plane

Next time: more intersections

Music: https://www.bensound.com