# Ray Casting

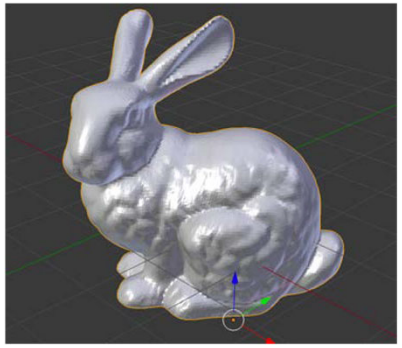
Some Slides/Images adapted from Marschner and Shirley

# Topics

- Rasterization vs. Ray Casting
- The Ray Casting Algorithm
- Introduction to Rays
- The Camera
- Ray-Object Intersection
  - Ray-Plane Intersection
  - Ray-Sphere Intersection
  - Ray-Triangle Intersection

# Photography

Input:



Objects

Lights


Camera

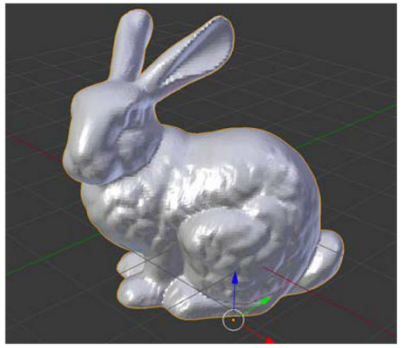
---

Output:

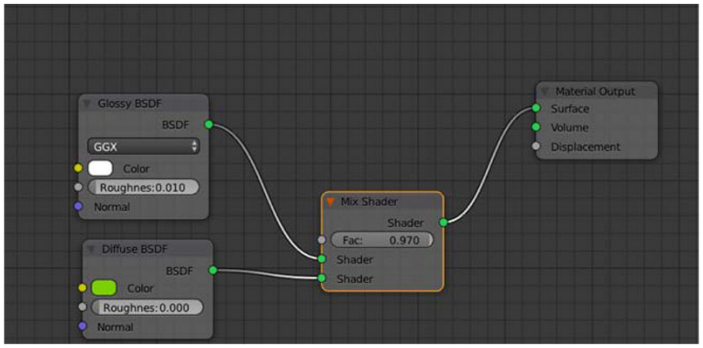
$I(x, y)$

# Rendering

Input:



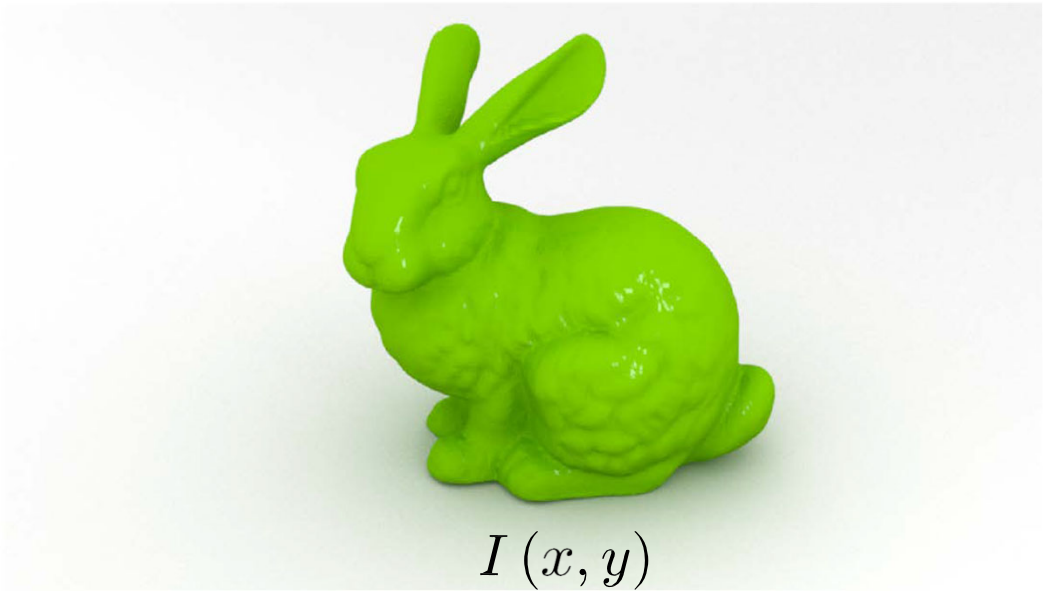
Objects



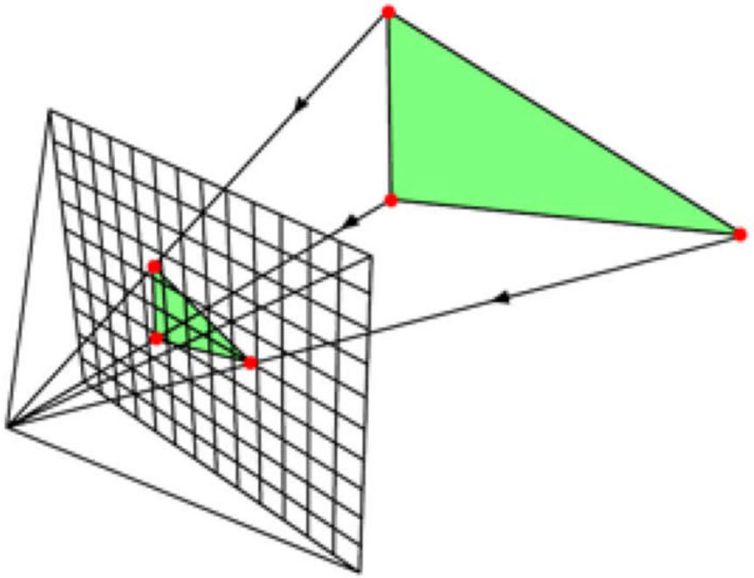

Materials

---

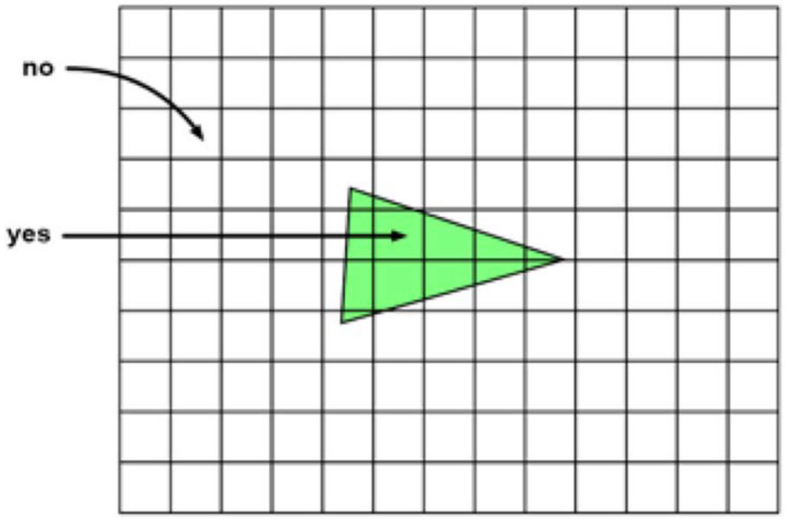
Output:



# Rasterization




1. Project Vertices to Image Plane




2. Turn on pixels inside triangle

https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation

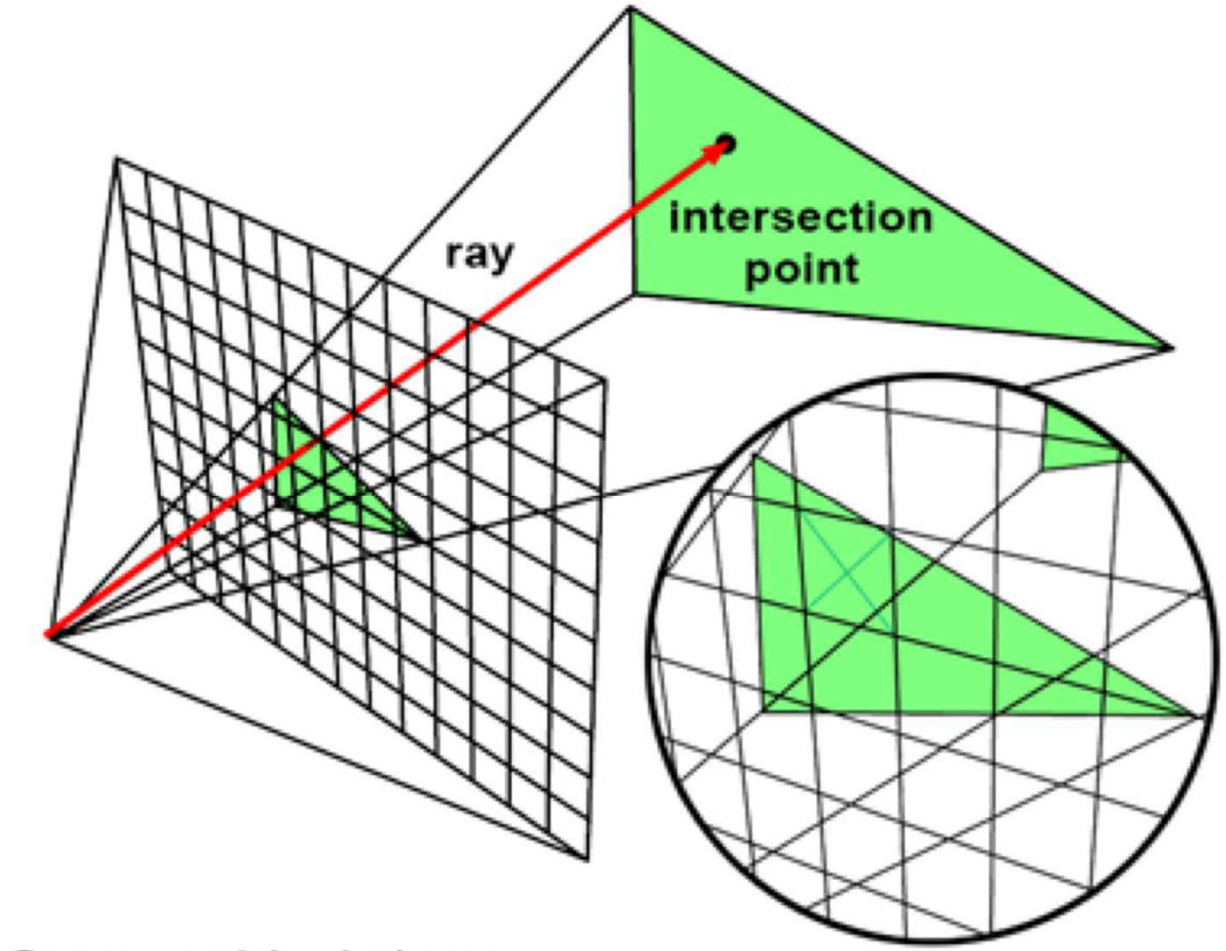

# Fortnite | Epic Games

# Rasterization

```
for each scene object {
    for each image pixel{
        if (object affects pixel) {
            do something
        }
    }
}
```


operations can be done quickly on the GPU!

# Ray Casting




https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation

# Ray Casting

```
for each image pixel {
    generate a ray
    for each scene object {
        if (ray intersects object) {
            set pixel color
        }
    }
}
```

# Ray Casting vs. Rasterization

**Ray Casting**

```
for each image pixel {
     for each scene object {
          ...
     }
}
```

**Rasterization**

```
for each scene object {
     for each image pixel {
          ...
     }
}
```

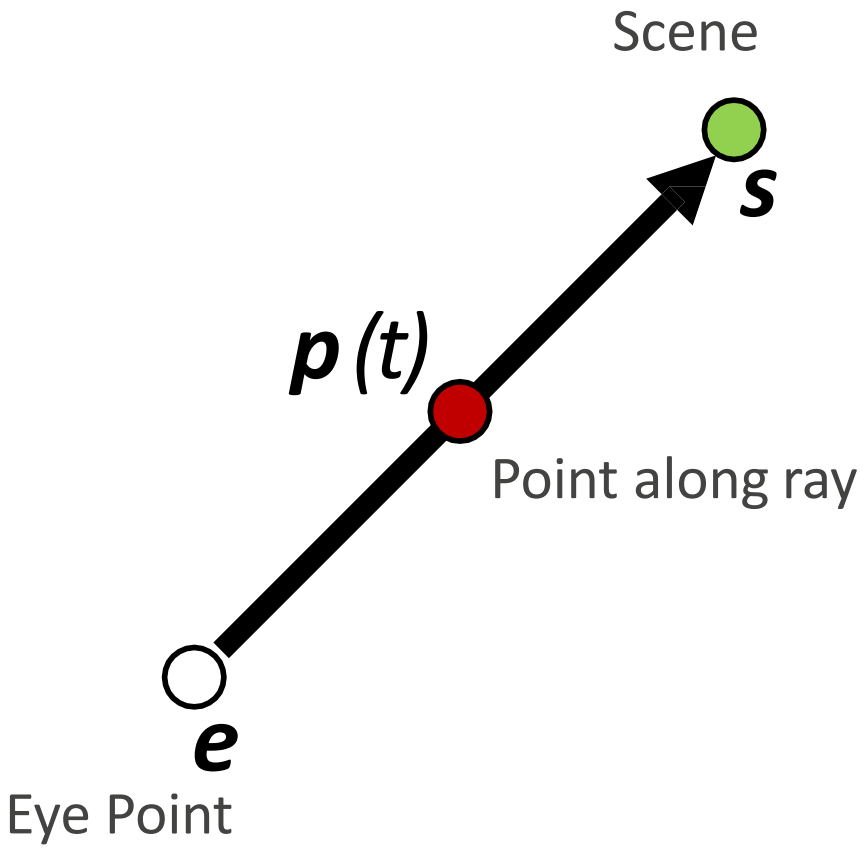
## Basic Components of Ray Casting

- Ray
- Camera
- Intersection Tests

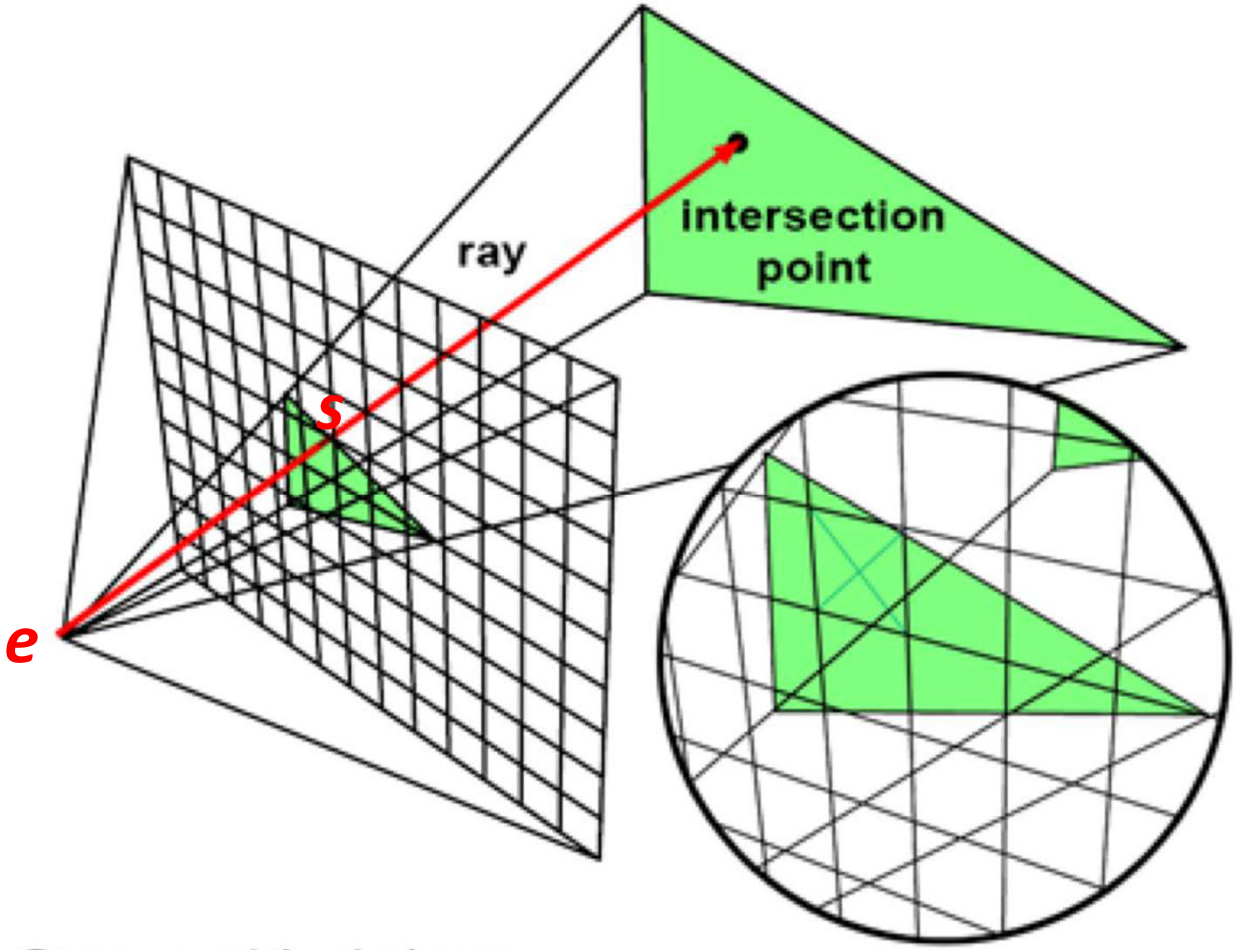
# Ray

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(0) = \mathbf{e}$$
$$\mathbf{p}(1) = \mathbf{s}$$

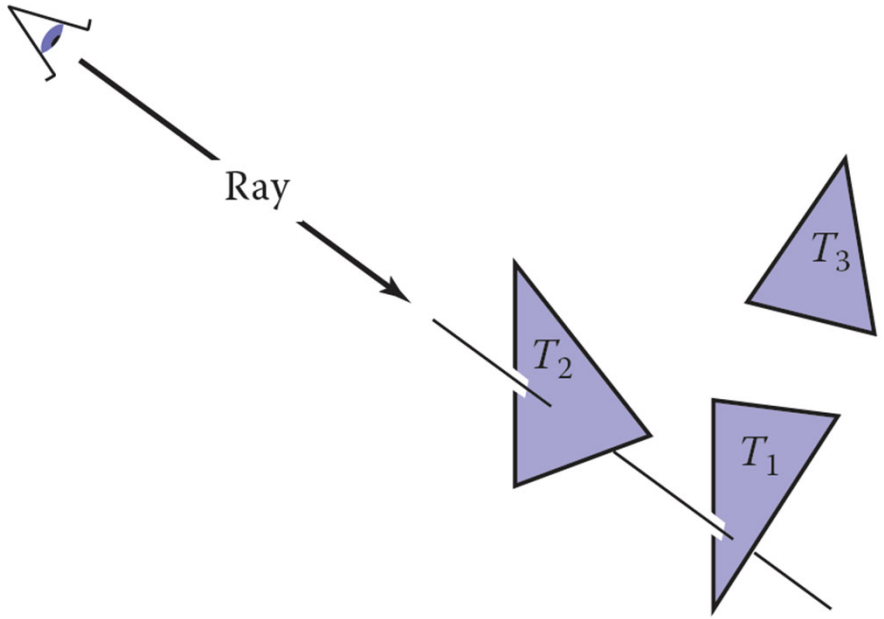


# Ray Casting




https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation

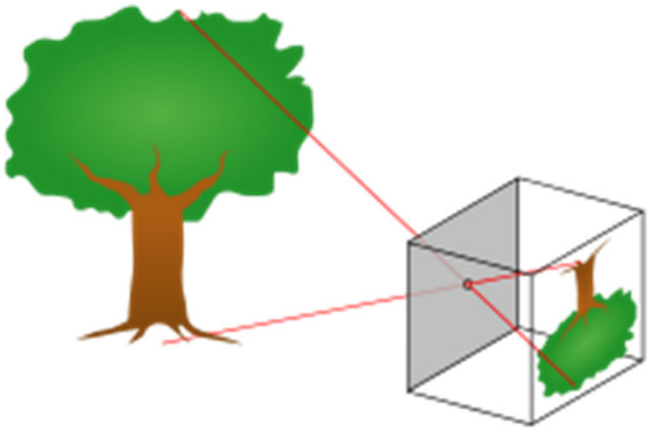
# Ray Casting



# Basic Components of Ray Casting

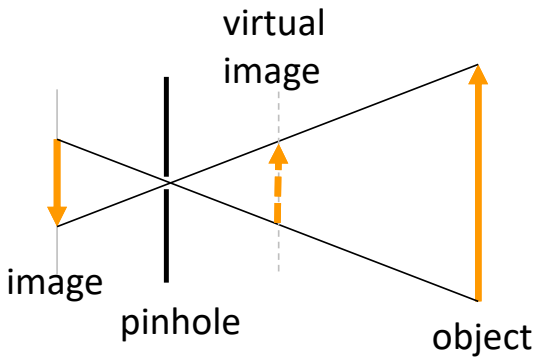
- Ray
- Camera
- Intersection Tests

# Camera model: camera obscura

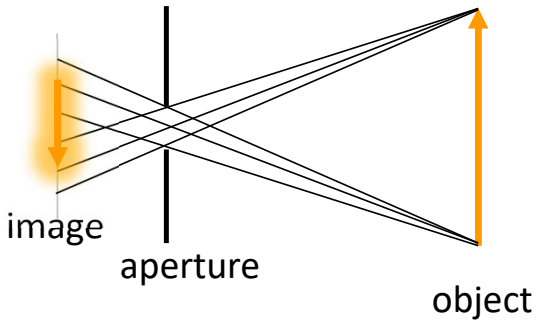


# Camera model

Ideal pinhole camera

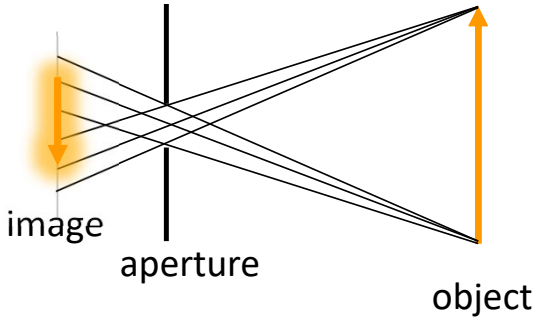



Real pinhole camera

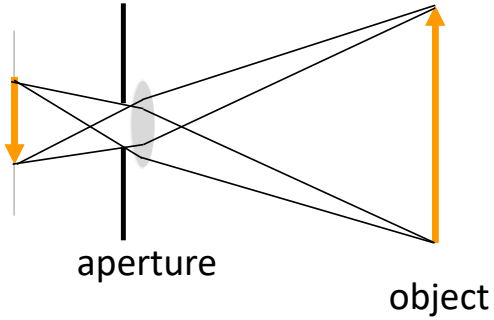


# Camera model

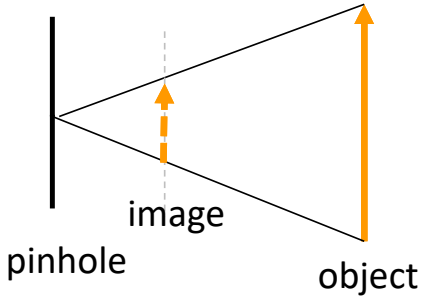
Real pinhole camera




Camera with a lens




Virtual camera

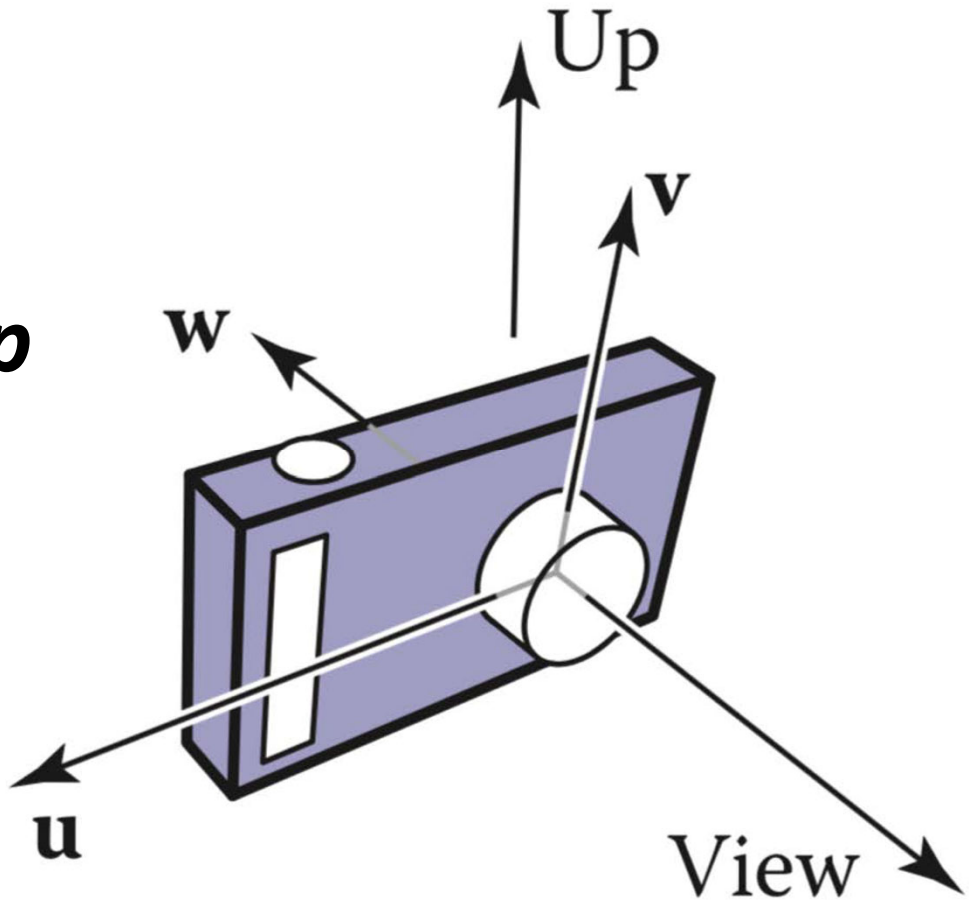


## Camera

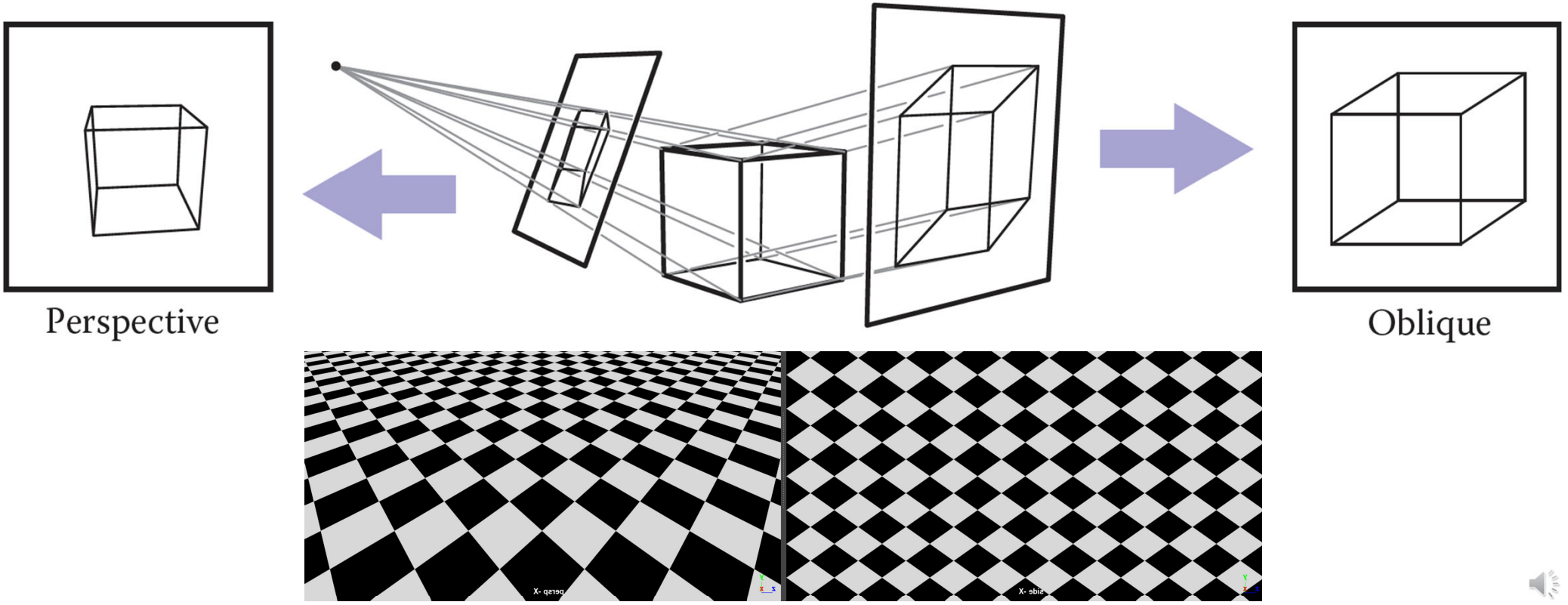
$w = -view$

$u = view \times up$

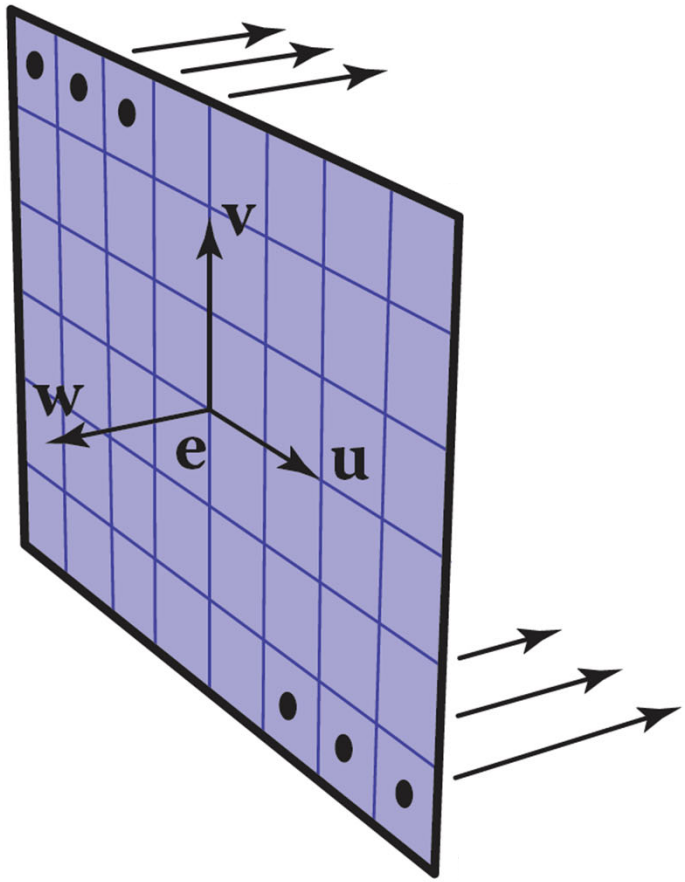
$v = w \times u$



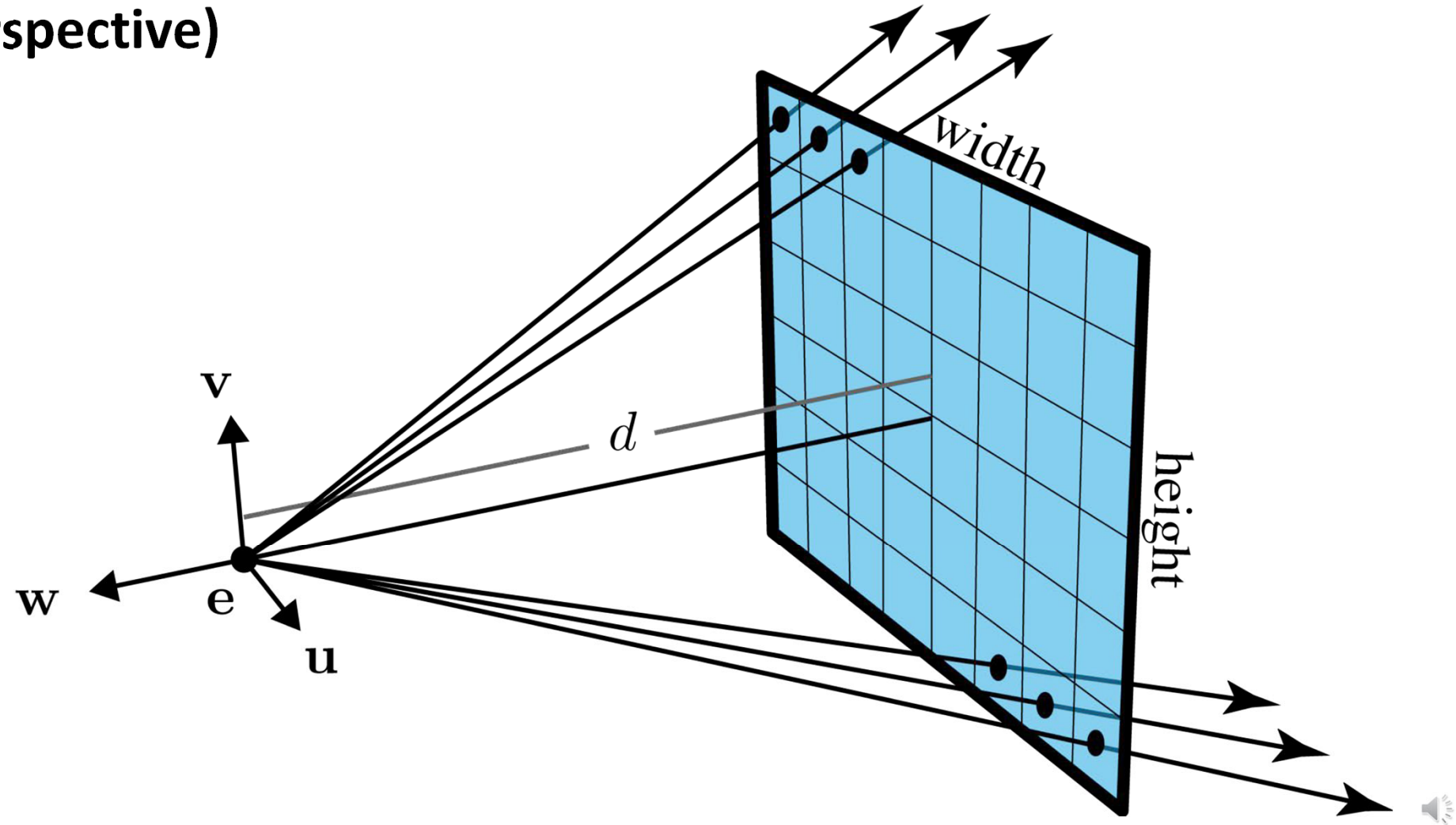
# Orthographic v Perspective Projection



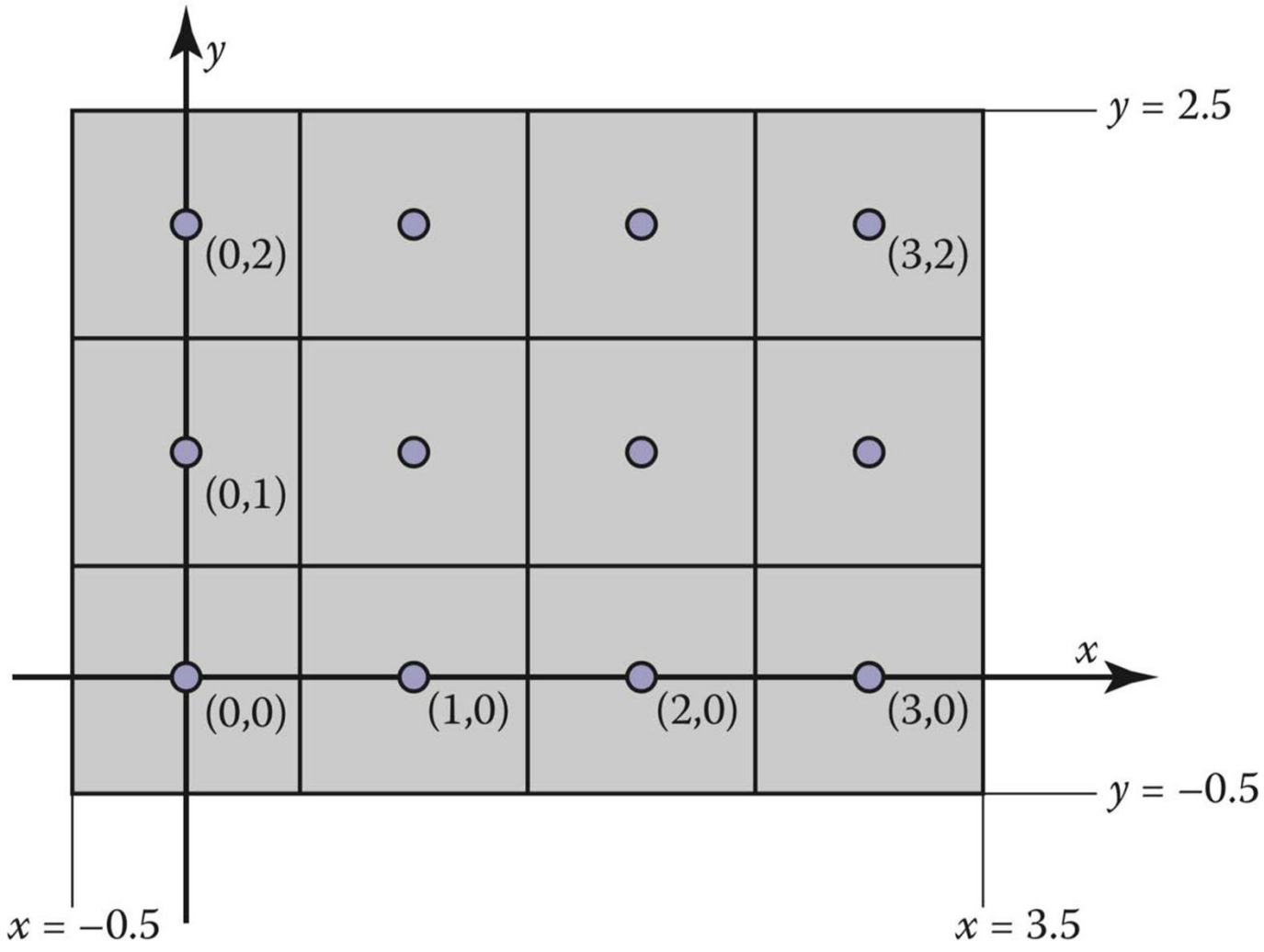
# Generating Rays (Orthographic)

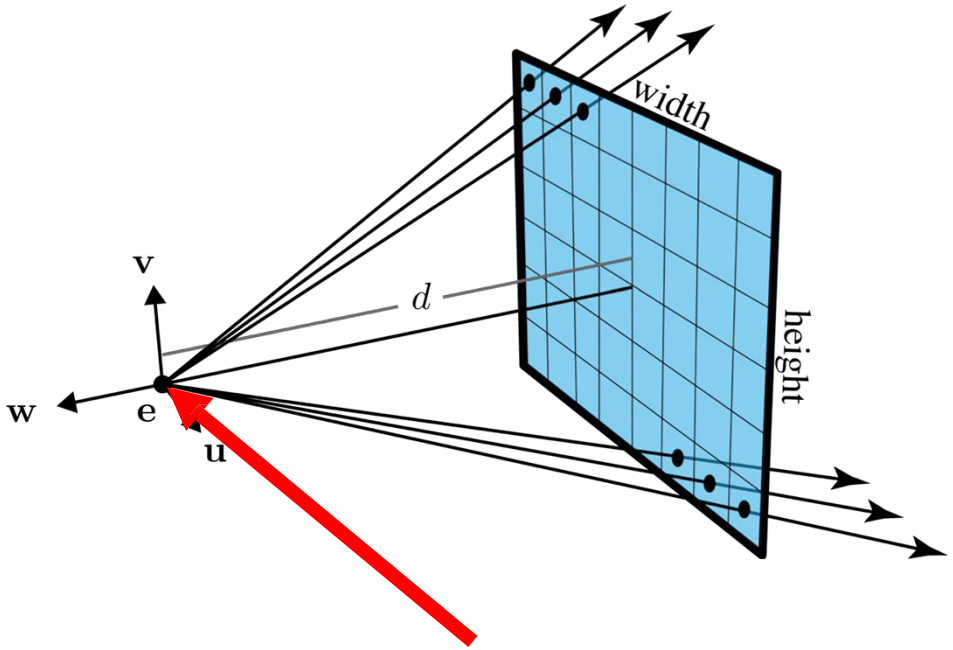


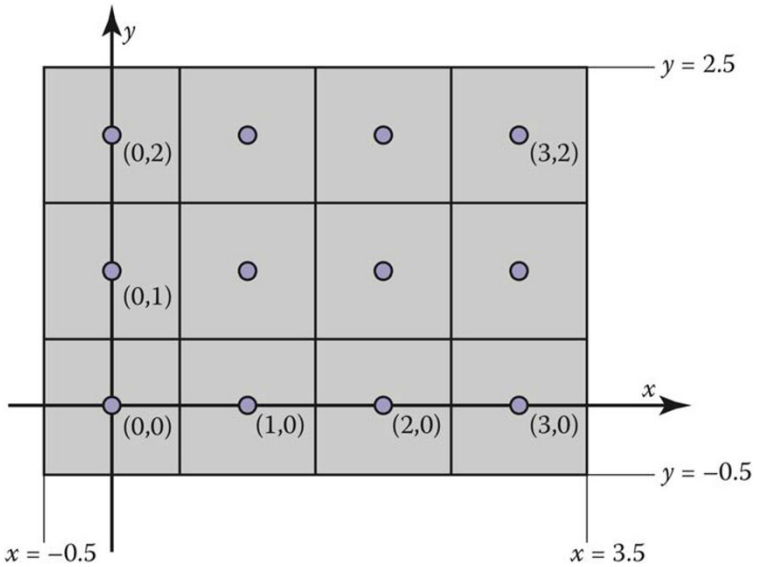
# Generating Rays (Perspective)



# Recall: Standard Pixel Coordinate System

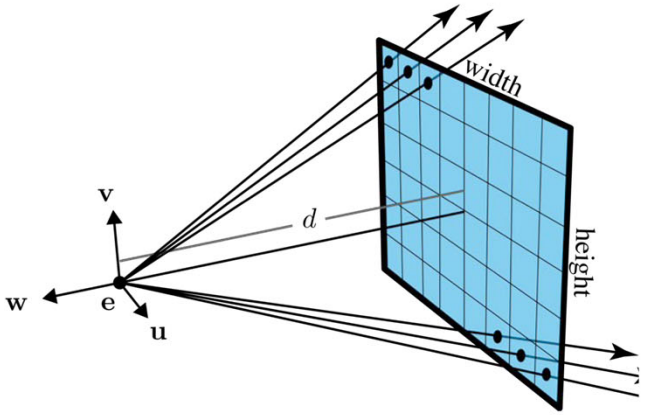




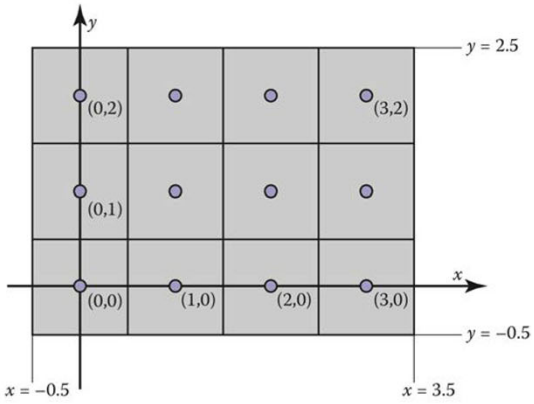




Origin of camera frame (the eye)

What are the *(u,v)* coordinates for pixel $(i, j)$ in the camera frame?




Camera space




Pixel space

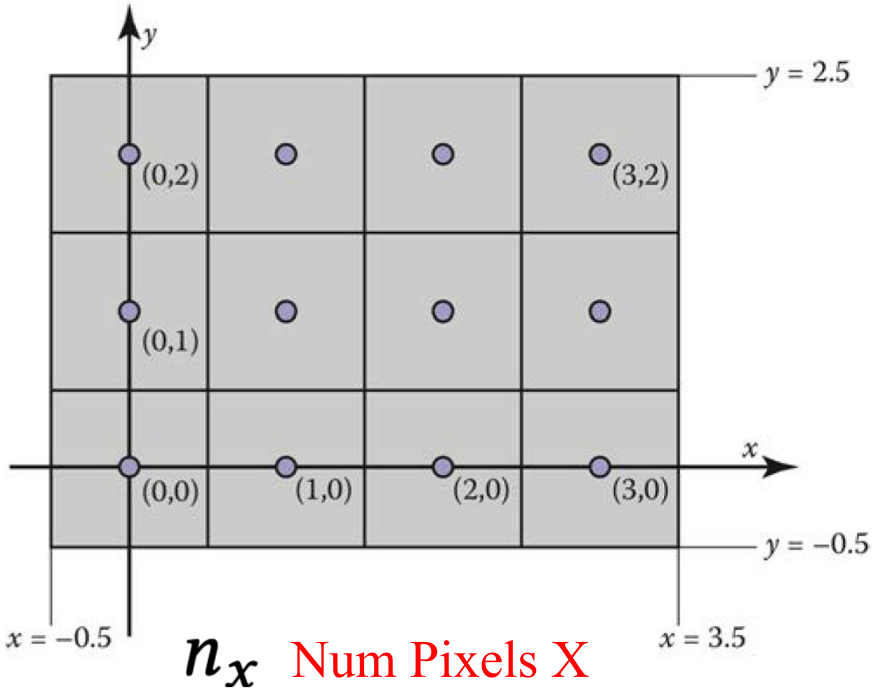
Bottom Left Corner $(i, j)$ : ?

Top Right Corner $(i, j)$: ?

Bottom Left Corner $(u, v)$ : ?

Top Right Corner $(u, v)$ : ?

$n_y$ Num Pixels Y




$n_x$ Num Pixels X

Bottom Left Corner $(i, j)$ : $(-\frac{1}{2}, -\frac{1}{2})$

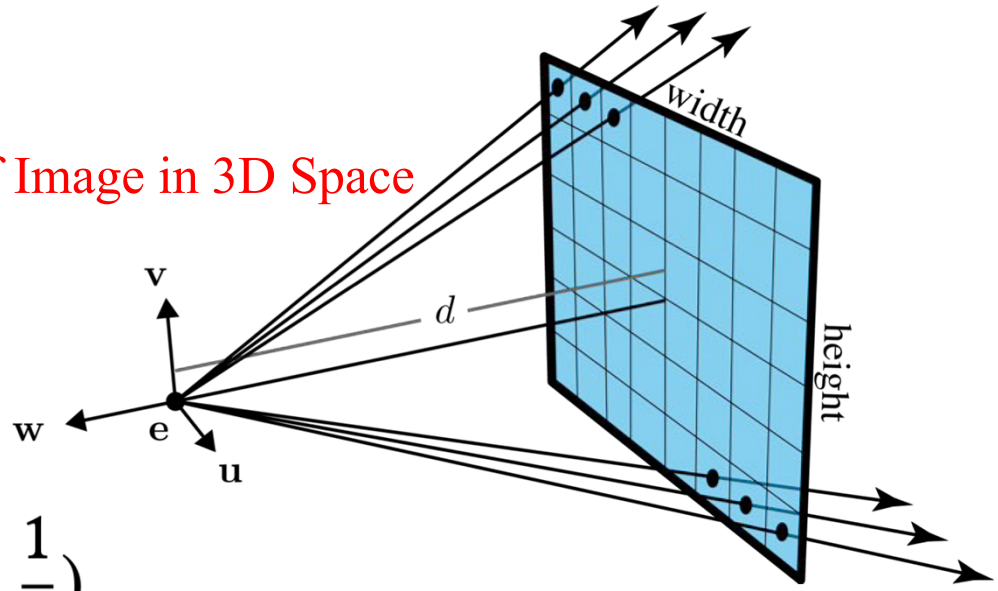
Top Right Corner $(i, j)$ : $(n_x - \frac{1}{2}, n_y - \frac{1}{2})$

Bottom Left Corner $(u, v)$ :

Top Right Corner $(u, v)$ :

Physical Width of Image in 3D Space

Physical Height of Image in 3D Space



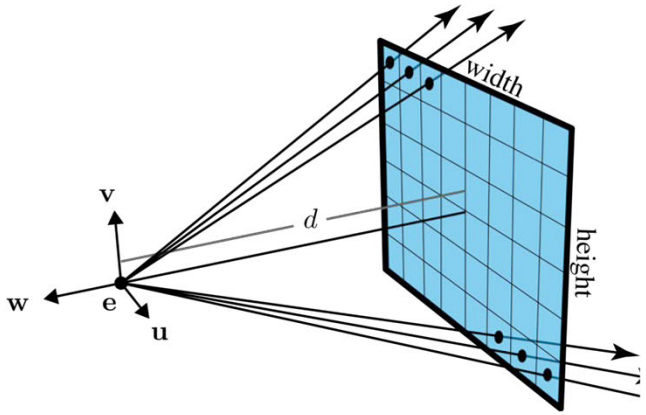

Bottom Left Corner $(i, j)$ : $(-\frac{1}{2}, -\frac{1}{2})$

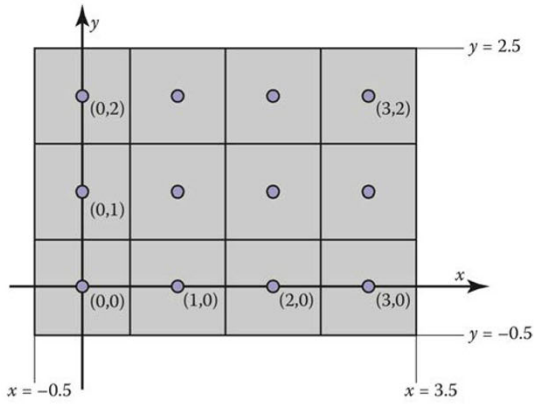
Top Right Corner $(i, j)$ : $(n_x - \frac{1}{2}, n_y - \frac{1}{2})$

Bottom Left Corner $(u, v)$ : $(-\frac{\text{width}}{2}, -\frac{\text{height}}{2})$

Top Right Corner $(u, v)$ : $(\frac{\text{width}}{2}, \frac{\text{height}}{2})$

pixel at position $(i, j)$ in the raster image has the position:




Physical Width of Image in 3D Space

$$u = \frac{\text{width}}{n_x} \cdot \left(i + \frac{1}{2}\right) - \frac{\text{width}}{2}$$

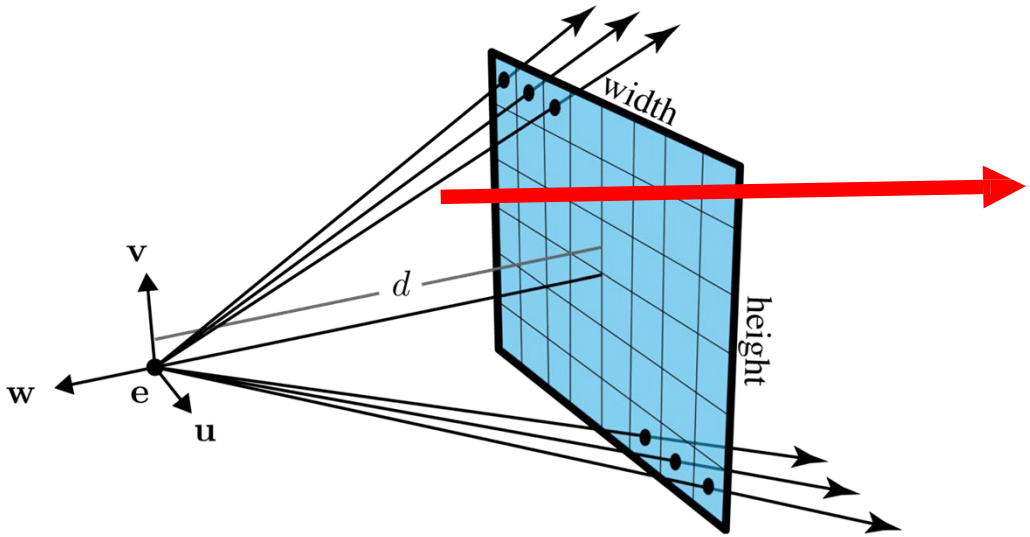
Num Pixels X

Physical Height of Image in 3D Space

$$v = \frac{\text{height}}{n_y} \cdot \left(j + \frac{1}{2}\right) - \frac{\text{height}}{2}$$

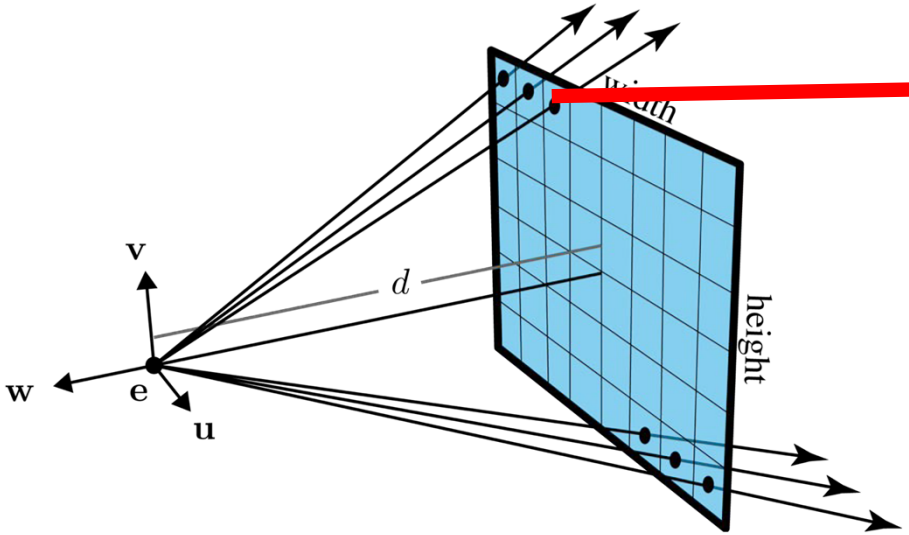
Num Pixels Y

# Ray Equation in Camera Space




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

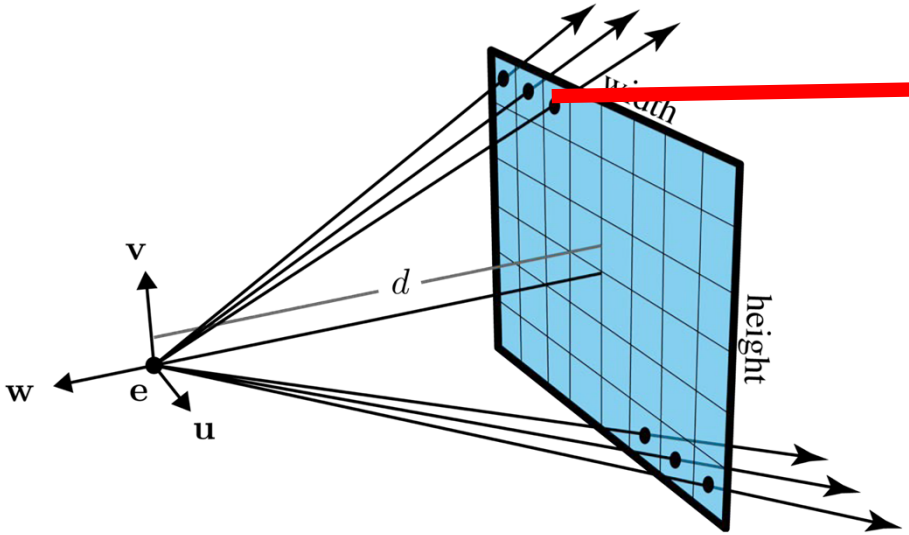
# Ray Equation in Camera Space




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t\left(\begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right)$$

# Ray Equation in Camera Space




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

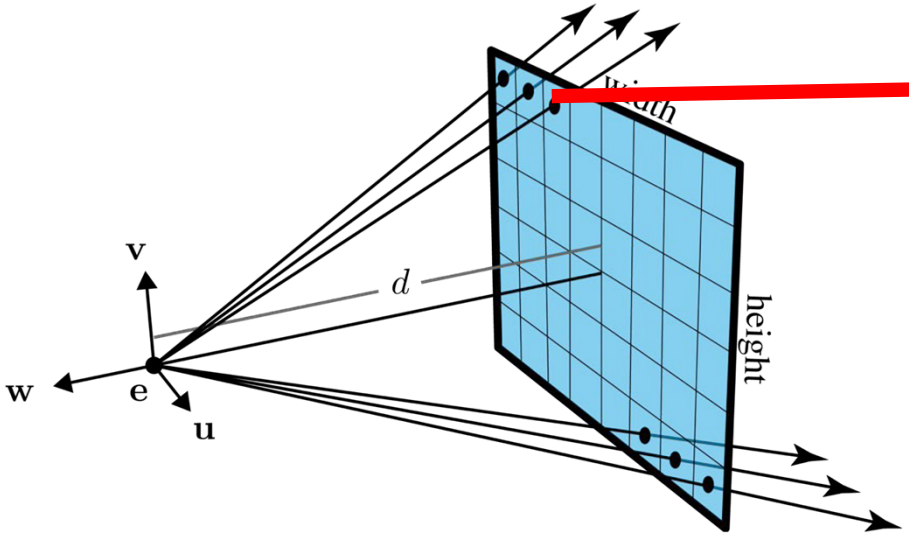
$$\mathbf{p}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t\left(\begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right)$$

$$\mathbf{p}(t) = t\begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix}$$

$$u = \frac{\text{width}}{n_x} \cdot \left(i + \frac{1}{2}\right) - \frac{\text{width}}{2}$$

$$v = \frac{\text{height}}{n_y} \cdot \left(j + \frac{1}{2}\right) - \frac{\text{height}}{2}$$

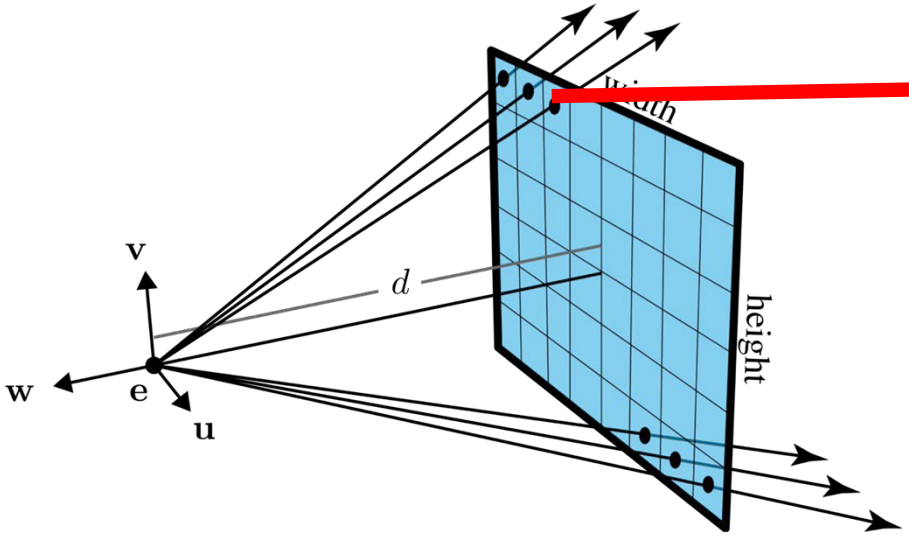
# Ray Equation in World Space




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(t) = \mathbf{e} + t\left(u(i)\mathbf{u} + v(j)\mathbf{v} + -d\mathbf{w}\right)$$

# Ray Equation in World Space




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(t) = \mathbf{e} + t \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{bmatrix} \begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix}$$

*Camera Transformation Matrix*

# Ray Casting

```
for each pixel in the image {
    Generate a ray
    for each object in the scene {
        if (Intersect ray with object){
            Set pixel colour
        }
    }
}
```

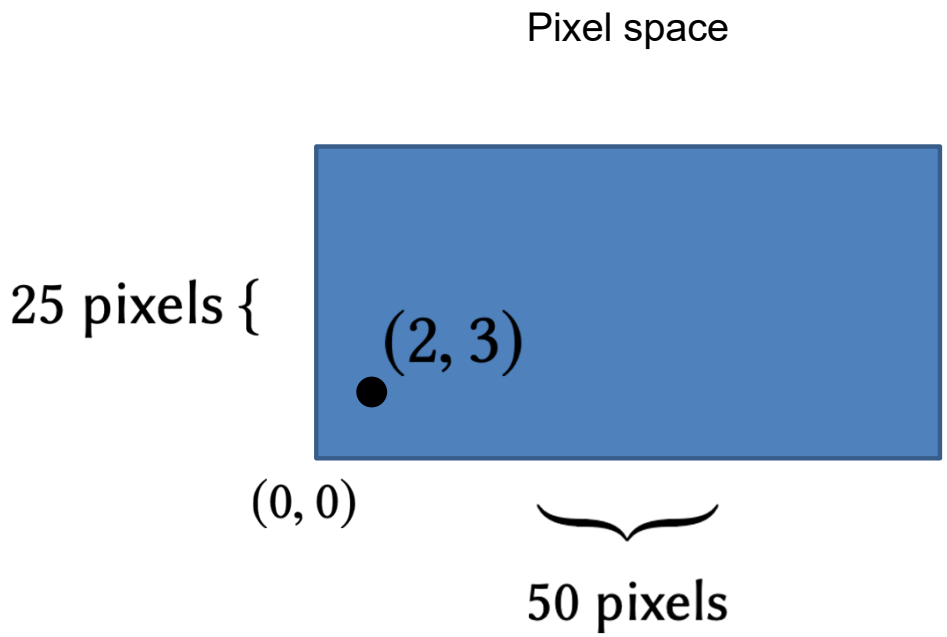
# Example – Pixel Space

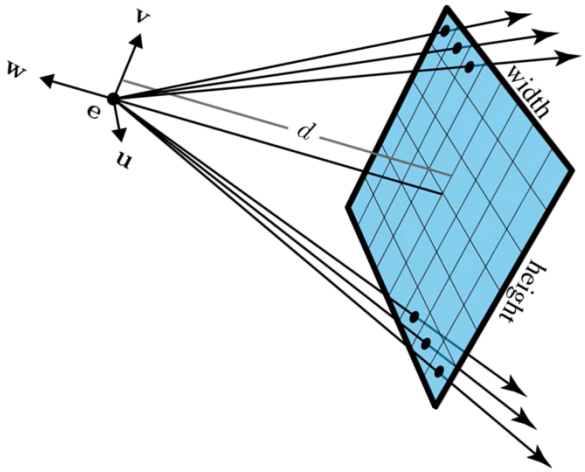
$\text{width} = 2$

$\text{height} = 1$

$n_x = 50$

$n_y = 25$



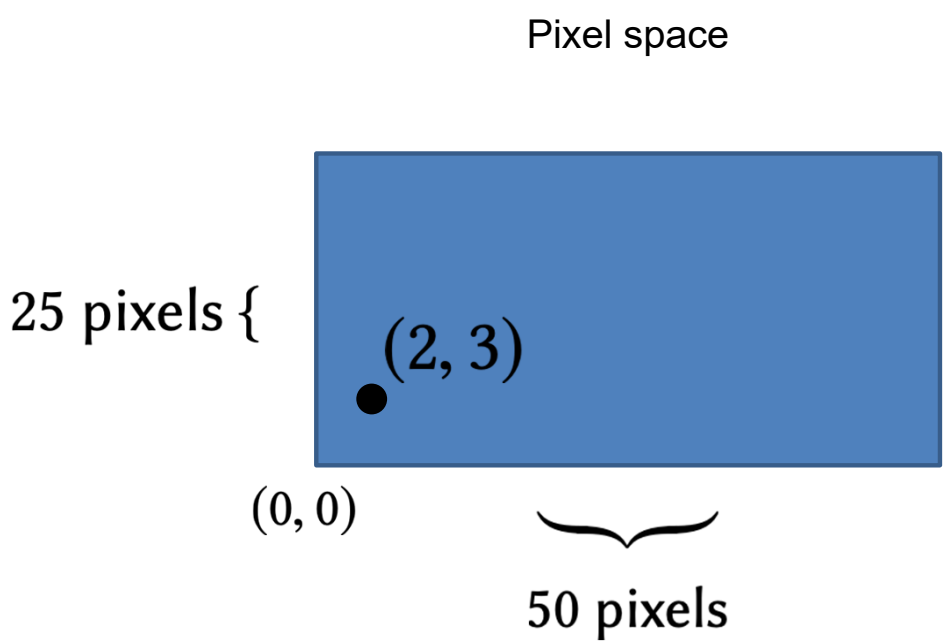
# Example – Pixel Space

$$\text{width} = 2$$

$$\text{height} = 1$$

$$n_x = 50$$

$$n_y = 25$$




$$u = \frac{\text{width}}{n_x} \cdot \left(i + \frac{1}{2}\right) - \frac{\text{width}}{2}$$

$$v = \frac{\text{height}}{n_y} \cdot \left(j + \frac{1}{2}\right) - \frac{\text{height}}{2}$$

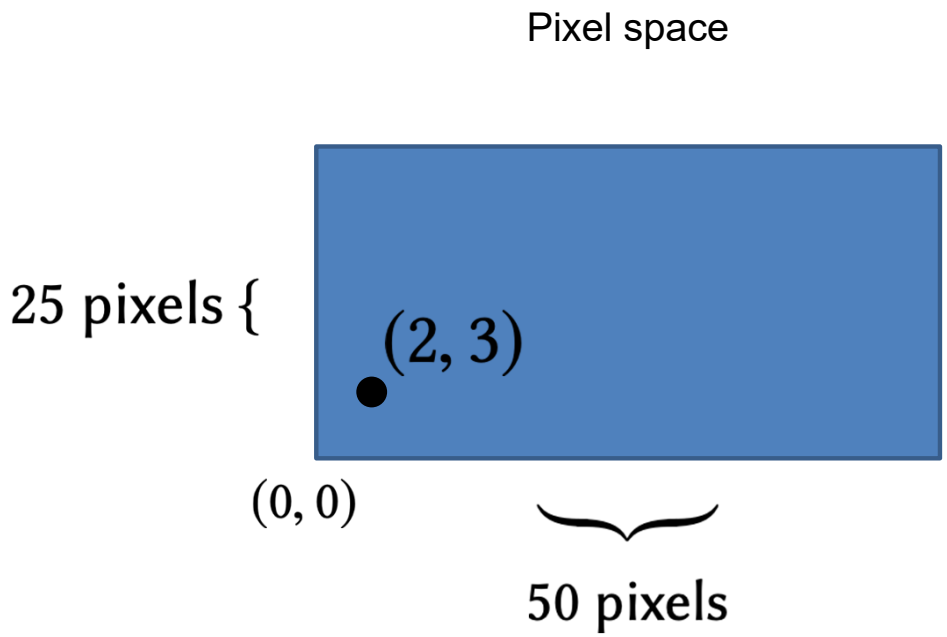
# Example – Pixel Space

$$\text{width} = 2$$

$$\text{height} = 1$$

$$n_x = 50$$

$$n_y = 25$$




$$u = \frac{2}{50} \cdot \left(2 + \frac{1}{2}\right) - \frac{2}{2}$$

$$v = \frac{1}{25} \cdot \left(3 + \frac{1}{2}\right) - \frac{1}{2}$$

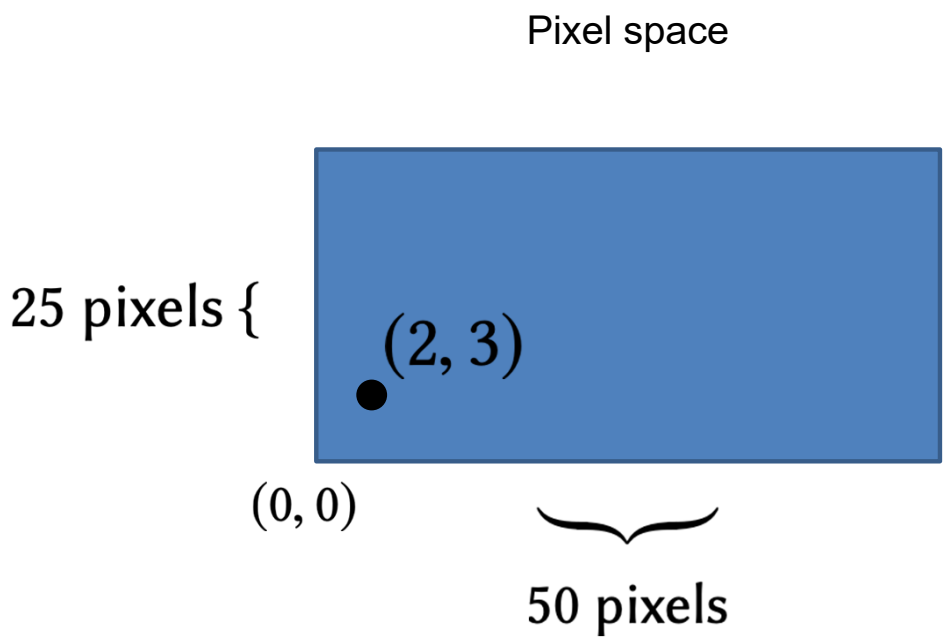
# Example – Pixel Space

$\text{width} = 2$

$\text{height} = 1$

$n_x = 50$

$n_y = 25$

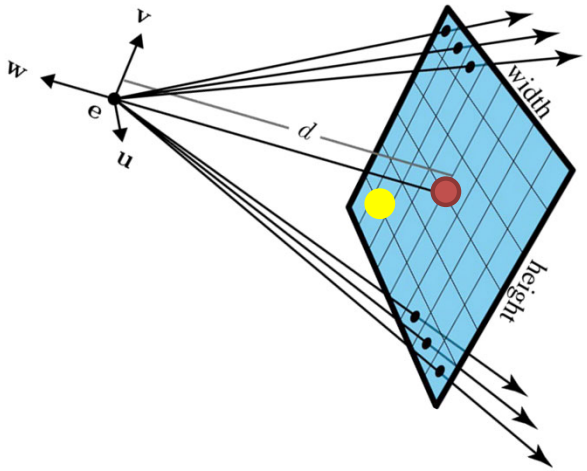



$u = -0.9$

$v = -0.36$

# Example – Pixel Space

Camera space



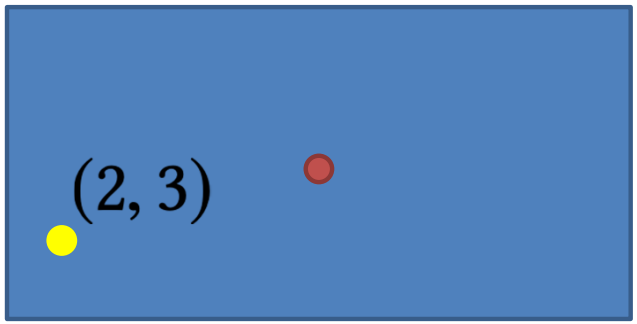

$\text{width} = 2$

$\text{height} = 1$

$n_x = 50$

$n_y = 25$

Pixel space




$u = -0.9$

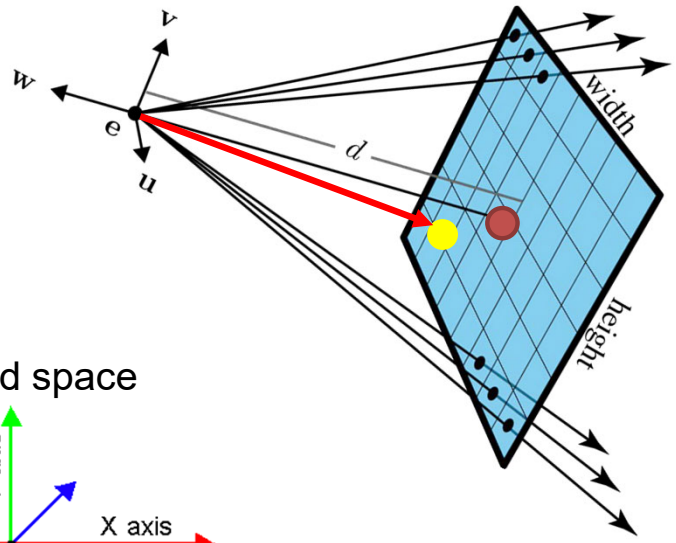
$v = -0.36$

# Example – Camera Space

$$\mathbf{u}, \mathbf{v}, \mathbf{w}$$

are the basis vectors for camera space

Camera space



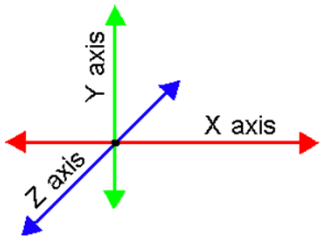

$$d = 10$$

d is the distance from the viewpoint to the image plane (focal length)

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

our ray equation

World space



# Example – Camera Space

$d = 10$

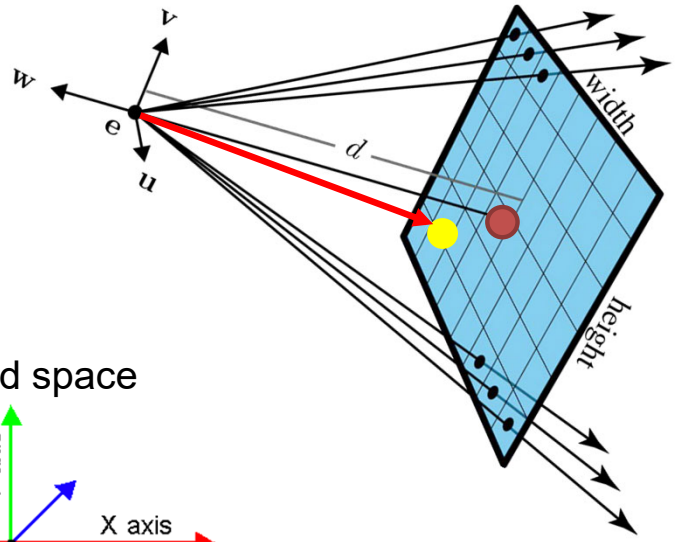
**u, v, w**

are the basis vectors for camera space

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

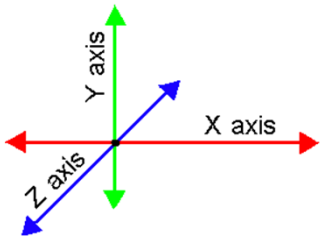
our ray equation

Camera space




$$\mathbf{p}(t) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t\left(\begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right)$$

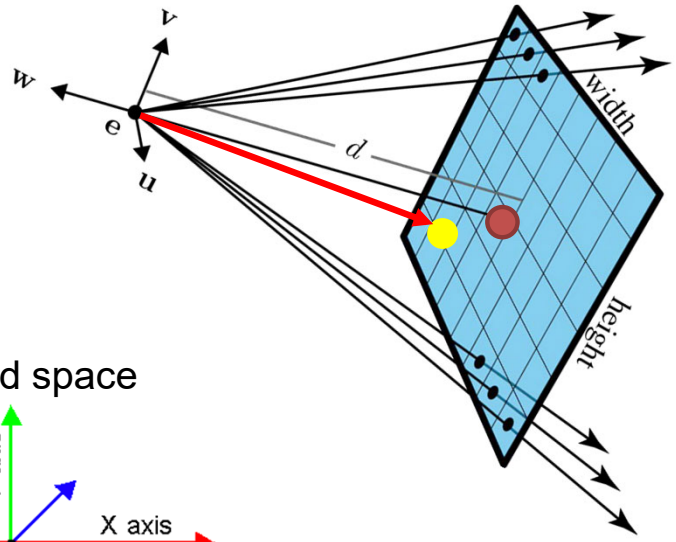
World space



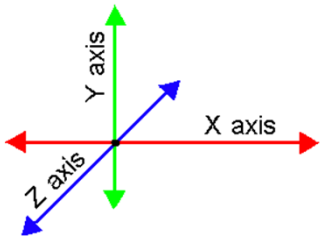
# Example – Camera Space

$d = 10$

Camera space




World space




**u, v, w**

are the basis vectors for camera space

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

our ray equation

for (i,j) = (2,3)

$$u(2) = -0.9, v(3) = -0.36$$

$$\mathbf{p}(t) = t \begin{bmatrix} -0.9 \\ -0.36 \\ -10 \end{bmatrix}$$

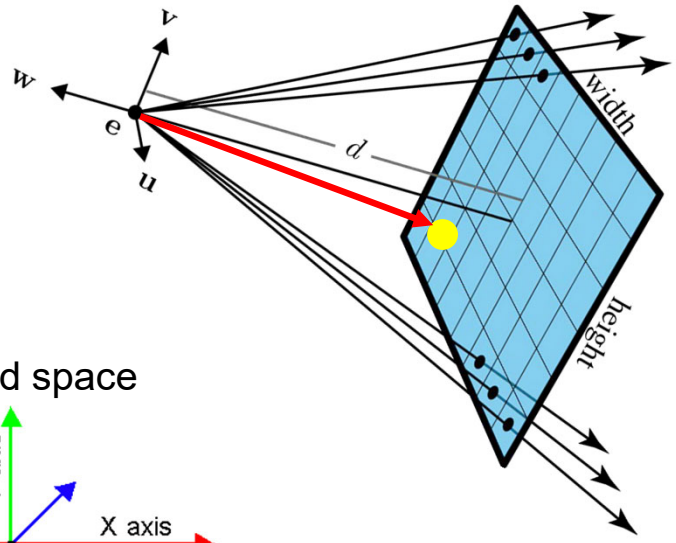
# Example – World Space

$\mathbf{u}, \mathbf{v}, \mathbf{w}$

are the basis vectors for camera space

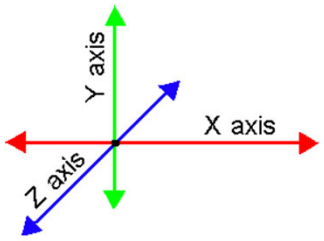
$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

our ray equation




$$\mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, \mathbf{w} = \begin{bmatrix} 0 \\ -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

World space




$$\mathbf{p}(t) = \mathbf{e} + t\left(u(i)\mathbf{u} + v(j)\mathbf{v} + -d\mathbf{w}\right)$$

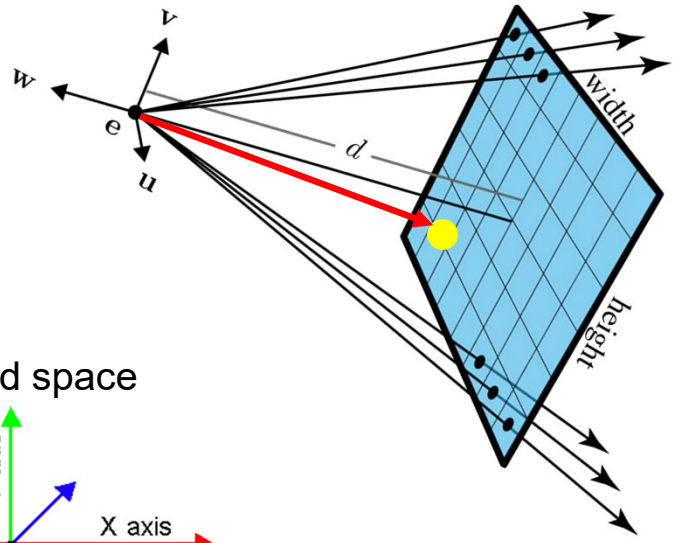
# Example – World Space

$\mathbf{u}, \mathbf{v}, \mathbf{w}$

are the basis vectors for camera space

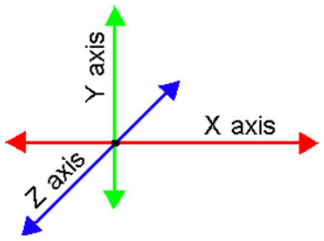
$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

our ray equation




$$\mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, \mathbf{w} = \begin{bmatrix} 0 \\ -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

World space




$$\mathbf{p}(t) = \mathbf{e} + t \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{bmatrix} \begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix}$$

# Example – World Space

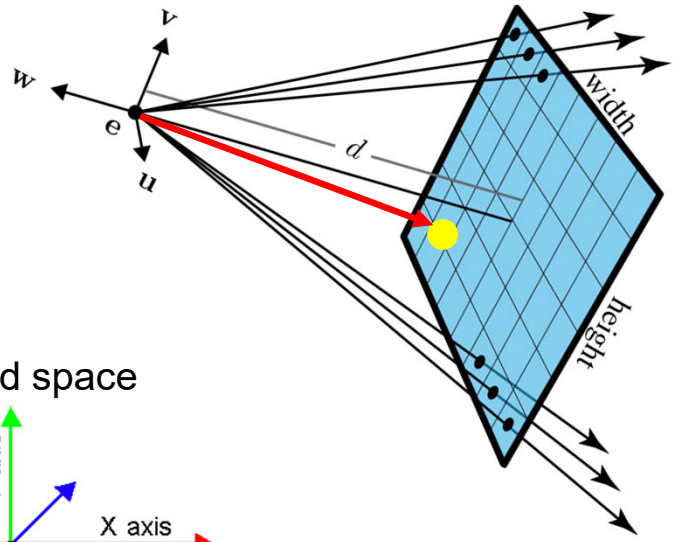
$$\mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, \mathbf{w} = \begin{bmatrix} 0 \\ -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

**u, v, w**

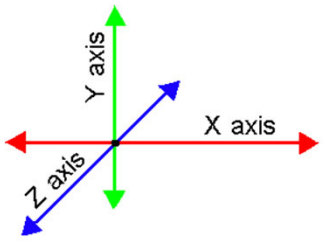
are the basis vectors for camera space

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

our ray equation




World space




$$\mathbf{p}(t) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + t \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} -0.9 \\ -0.36 \\ -10 \end{bmatrix}$$

https://mathworld.wolfram.com/ChangeofCoordinatesMatrix.html

# Ray Casting

Some Slides/Images adapted from Marschner and Shirley and David Levin

## Basic Components of Ray Casting

Ray

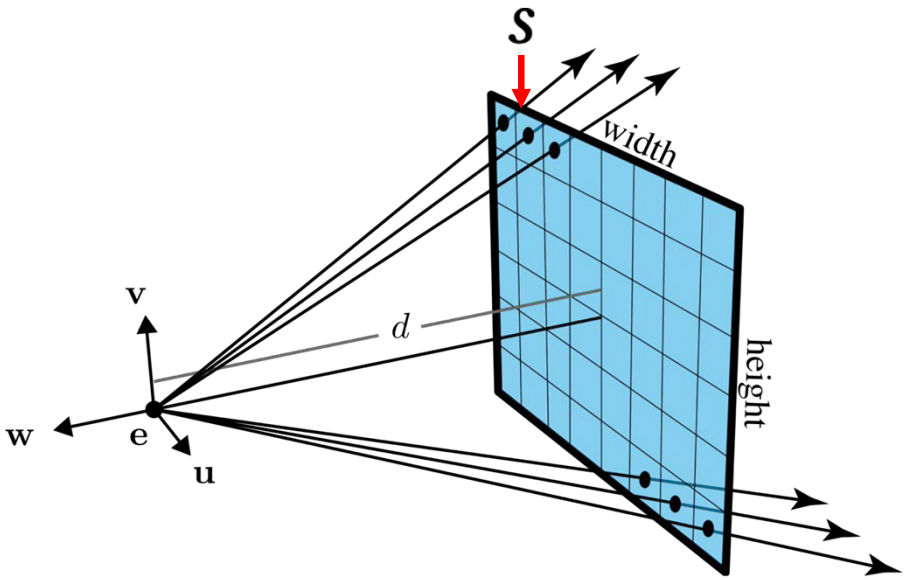
Camera

Intersection Tests

## Intersection Tests

- Plane
- Sphere
- Triangle

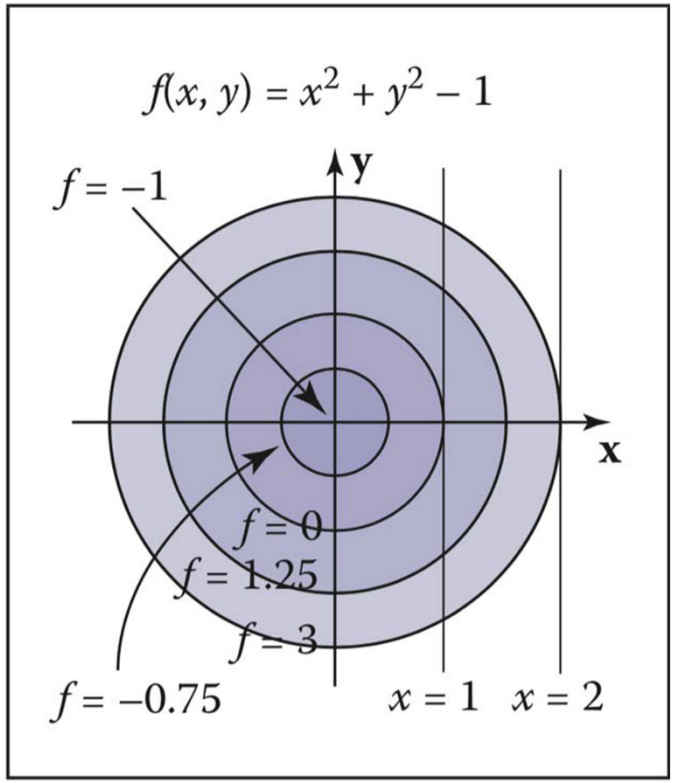
# Ray Equation




$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

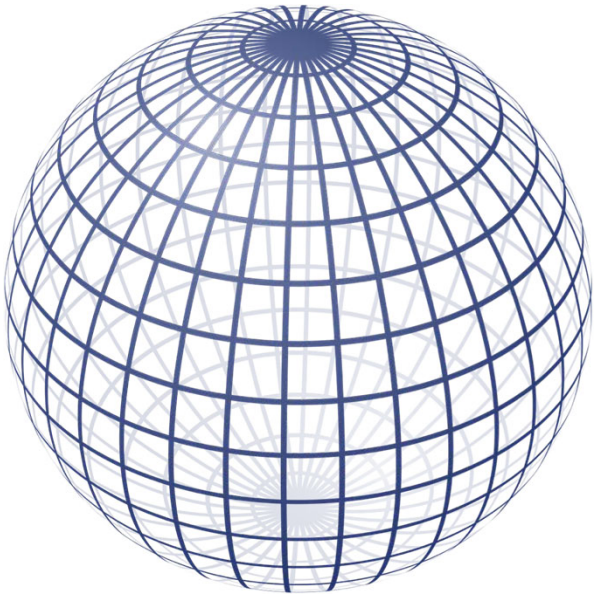
# Aside: Types of Surface

## Implicit Surface




## Parametric Surface

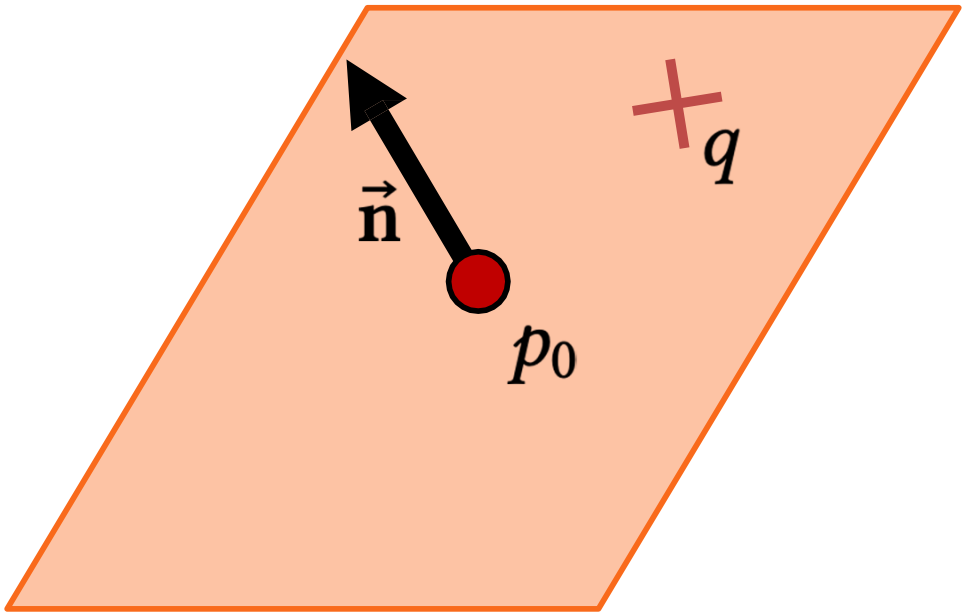
$$x = r \cos\phi \sin\theta,$$
$$y = r \sin\phi \sin\theta,$$
$$z = r \cos\theta.$$



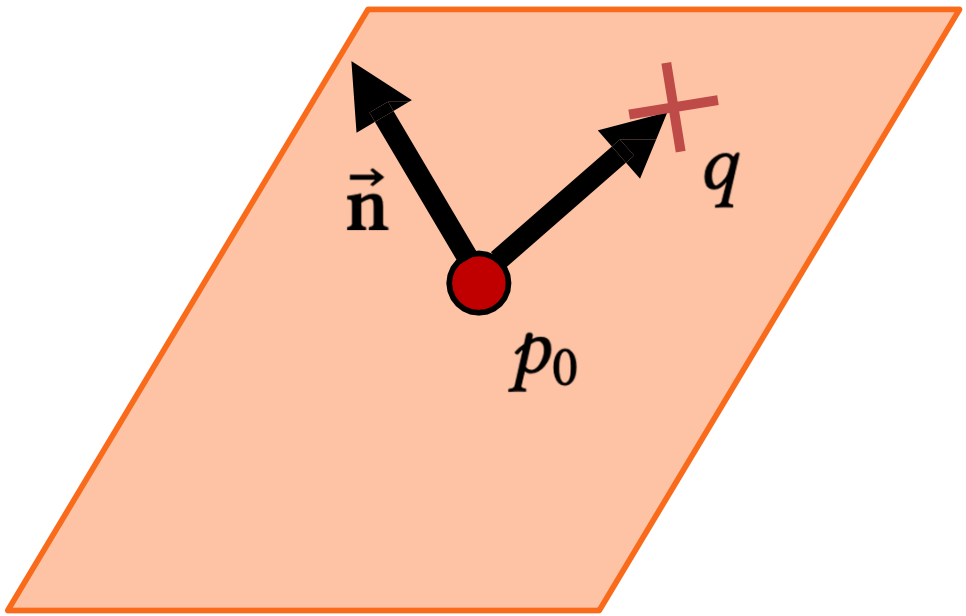
## Intersection Tests

- Plane
- Sphere
- Triangle

# Plane Equation

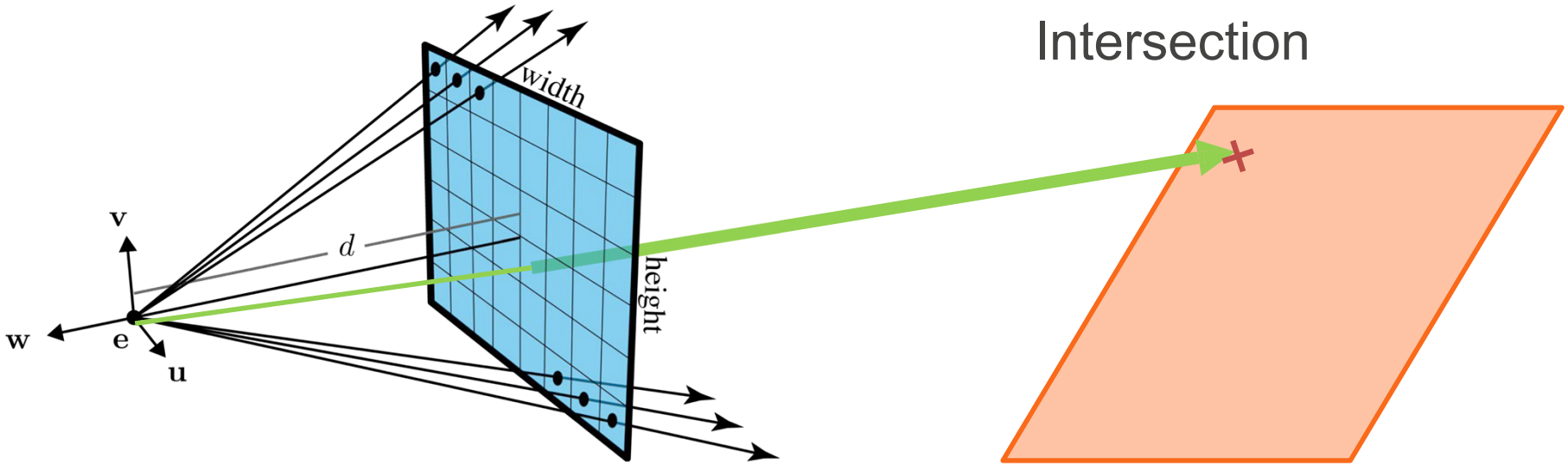


# Plane Equation




$$\vec{n} \cdot (q - p_0) = 0$$

# Ray-Plane Intersection

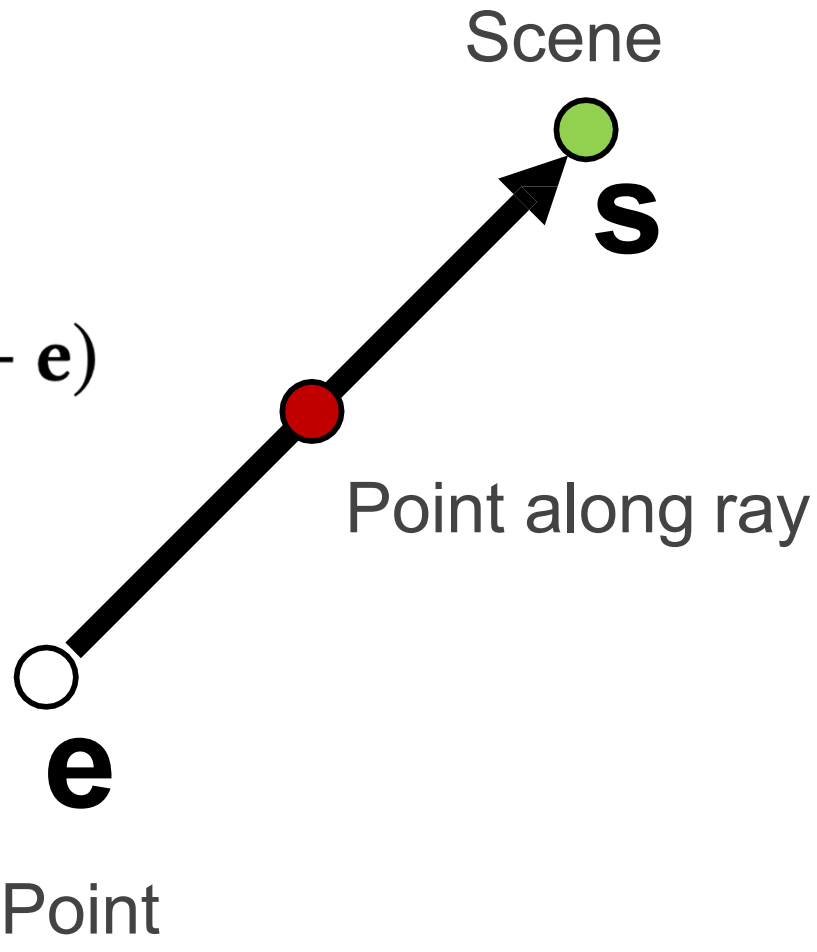


# Ray-Plane Intersection

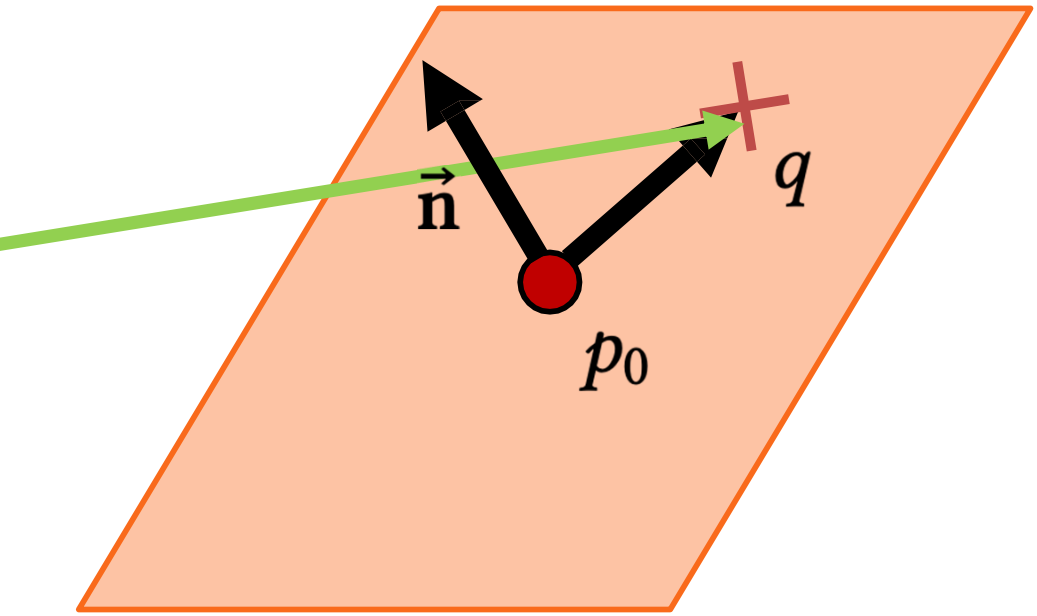
Ray equation

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(t) = \mathbf{e} + t\vec{\mathbf{d}}$$



# Plane Equation

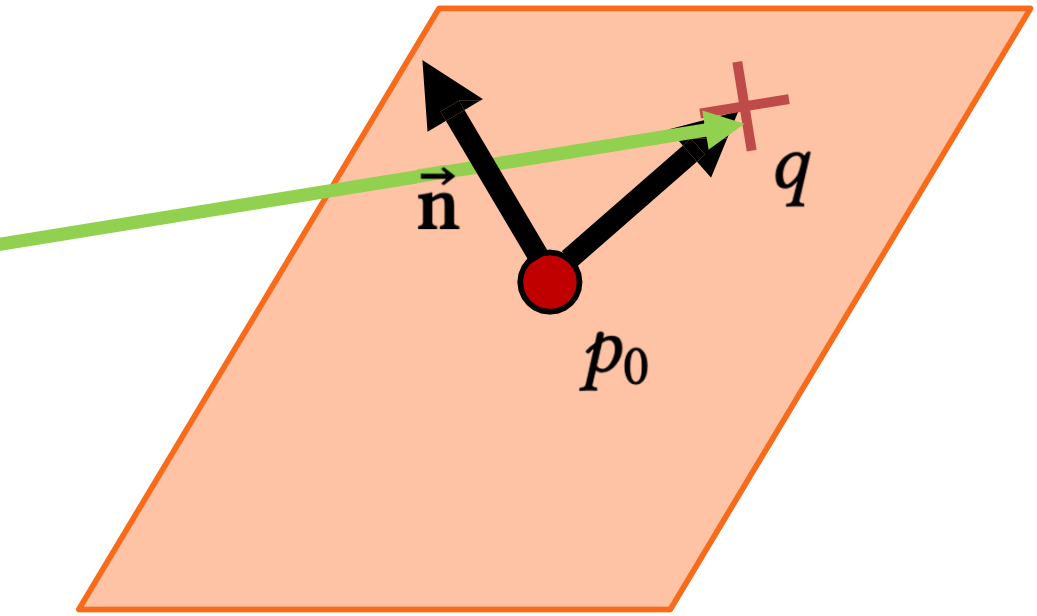



Plane equation
Substitute ray equation into it

$$\vec{\mathbf{n}} \cdot (\mathbf{p}(t) - p_0) = 0$$

$$\vec{\mathbf{n}} \cdot ((\mathbf{e} + t\vec{\mathbf{d}}) - p_0) = 0$$

# Plane Equation




Plane equation
Substitute ray equation into it

$$\vec{\mathbf{n}} \cdot (\mathbf{p}(t) - p_0) = 0$$

$$\vec{\mathbf{n}} \cdot ((\mathbf{e} + t\vec{\mathbf{d}}) - p_0) = 0$$

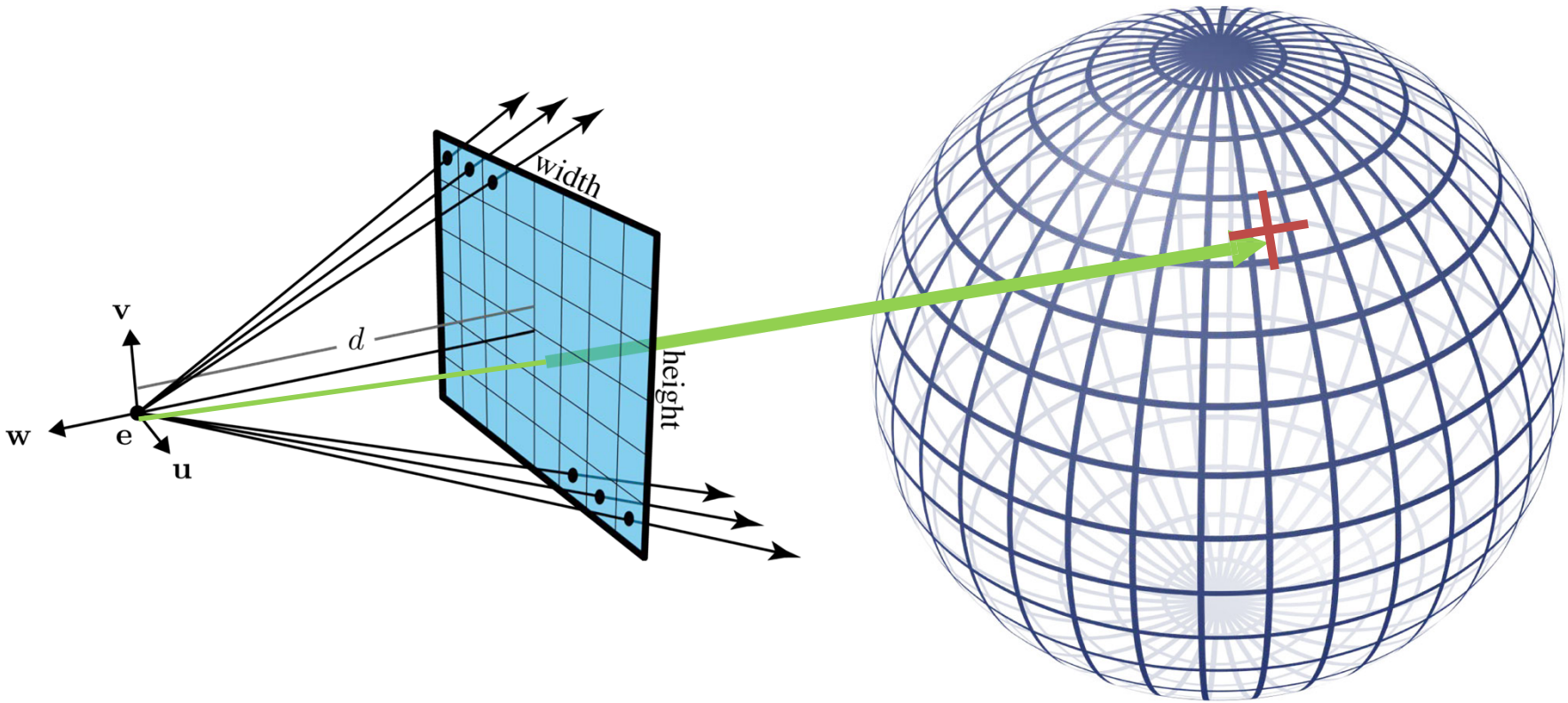
Solve for t

$$t = \frac{-\vec{\mathbf{n}} \cdot (\mathbf{e} - p_0)}{\vec{\mathbf{n}} \cdot \vec{\mathbf{d}}}$$

## Intersection Tests

- Plane
- Sphere
- Triangle

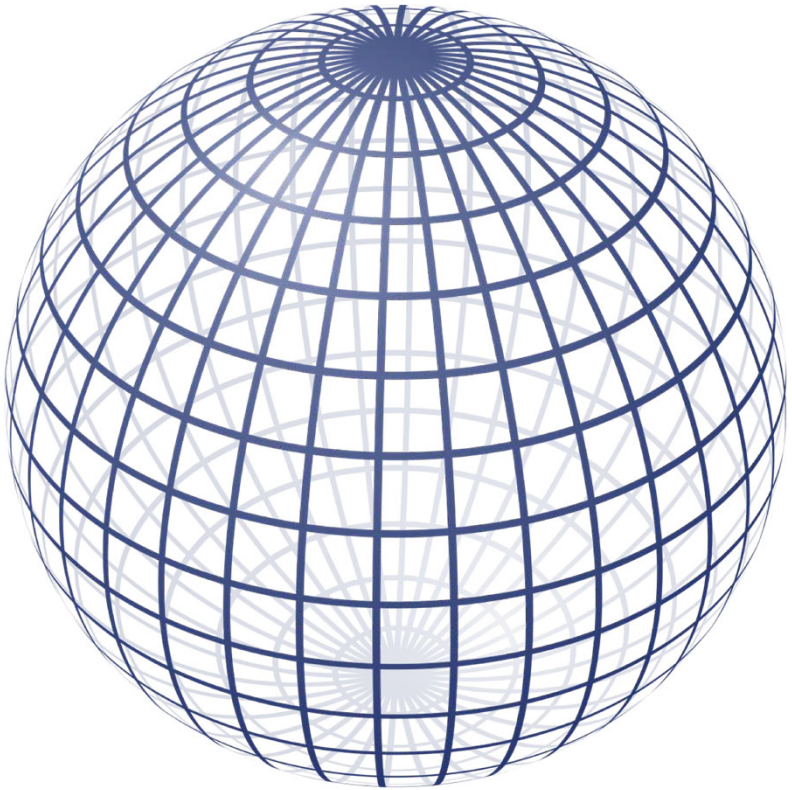
# Ray-Sphere Intersection



# Implicit Equation of a Sphere

$$(q - \mathbf{c}) \cdot (q - \mathbf{c}) - r^2 = 0$$

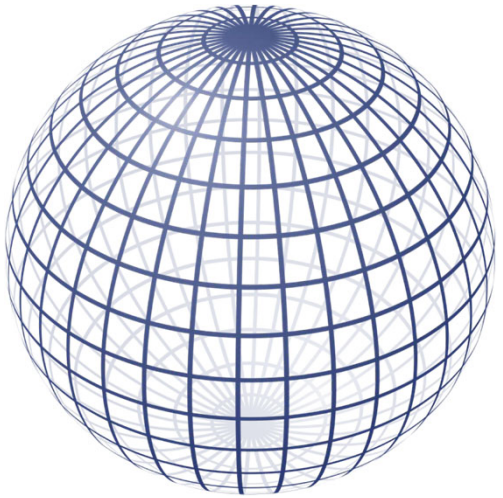
*Sphere centered at **c** with radius r*



# Ray-Sphere Intersection

Substitute ray equation into implicit equation for sphere

$$(\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) \cdot (\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) - r^2 = 0$$



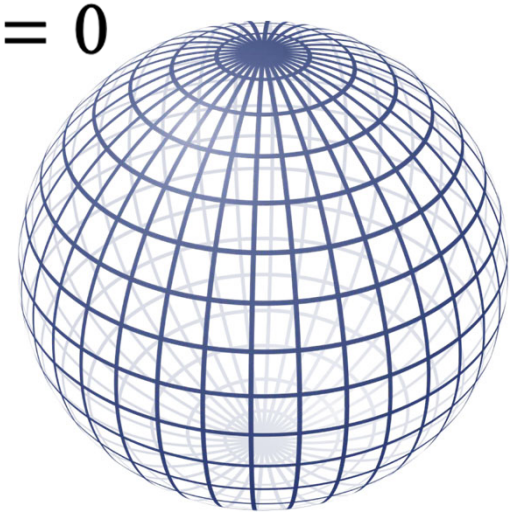
# Ray-Sphere Intersection

Substitute ray equation into implicit equation for sphere

$$(\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) \cdot (\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) - r^2 = 0$$

Rearrange

$$(\vec{\mathbf{d}} \cdot \vec{\mathbf{d}})t^2 + 2\vec{\mathbf{d}} \cdot (\mathbf{e} - \mathbf{c})t + (\mathbf{e} - \mathbf{c}) \cdot (\mathbf{e} - \mathbf{c}) - r^2 = 0$$



# Ray-Sphere Intersection

Substitute ray equation into implicit equation for sphere

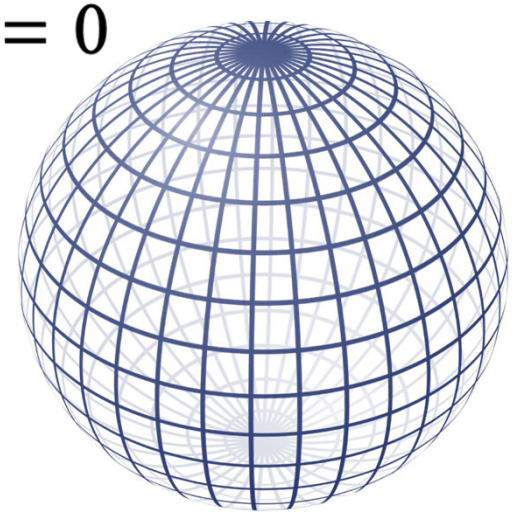
$$(\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) \cdot (\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) - r^2 = 0$$

Rearrange

$$(\vec{\mathbf{d}} \cdot \vec{\mathbf{d}})t^2 + 2\vec{\mathbf{d}} \cdot (\mathbf{e} - \mathbf{c})t + (\mathbf{e} - \mathbf{c}) \cdot (\mathbf{e} - \mathbf{c}) - r^2 = 0$$

Looks familiar…

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



# Ray-Sphere Intersection

Substitute ray equation into implicit equation for sphere

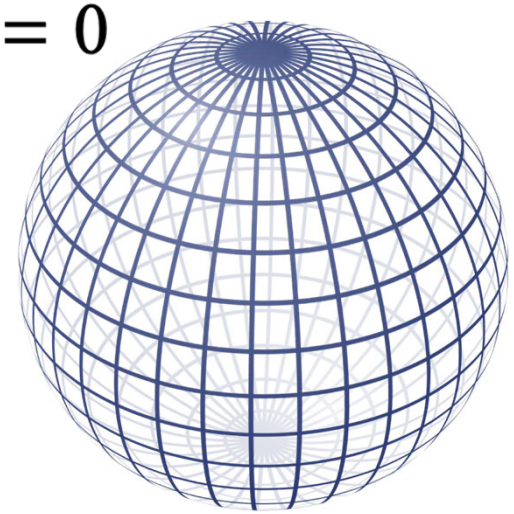
$$(\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) \cdot (\mathbf{e} + t\vec{\mathbf{d}} - \mathbf{c}) - r^2 = 0$$

Rearrange

$$(\vec{\mathbf{d}} \cdot \vec{\mathbf{d}})t^2 + 2\vec{\mathbf{d}} \cdot (\mathbf{e} - \mathbf{c})t + (\mathbf{e} - \mathbf{c}) \cdot (\mathbf{e} - \mathbf{c}) - r^2 = 0$$

Looks familiar…

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



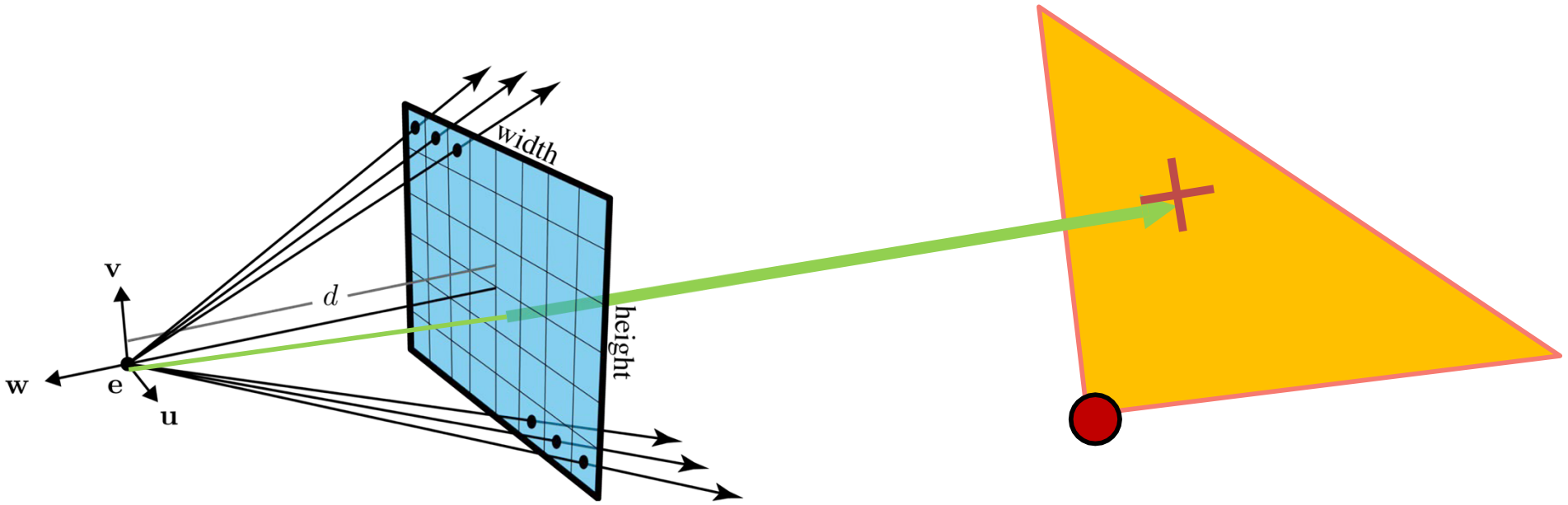
It’s a quadratic! (can use the quadratic equation)

Hint: the discriminant tells us what kinds of roots the equation has.

## Intersection Tests

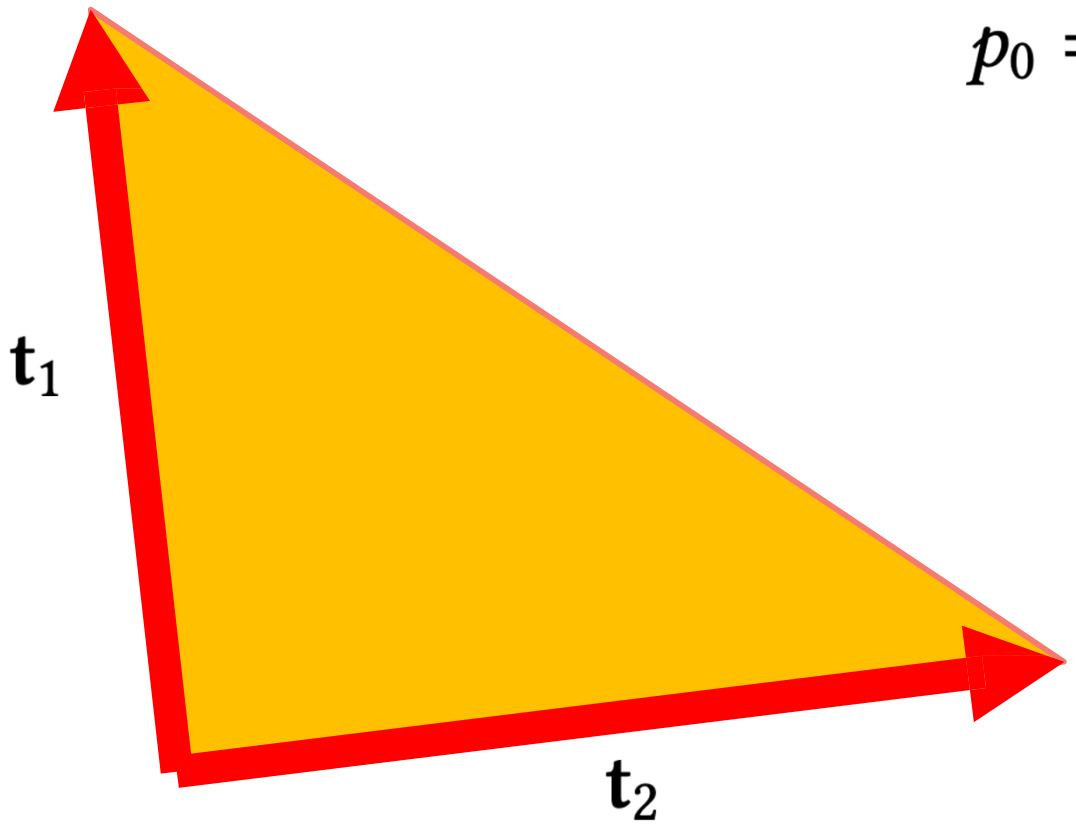
- Plane
- Sphere
- Triangle

# Ray-Triangle Intersection

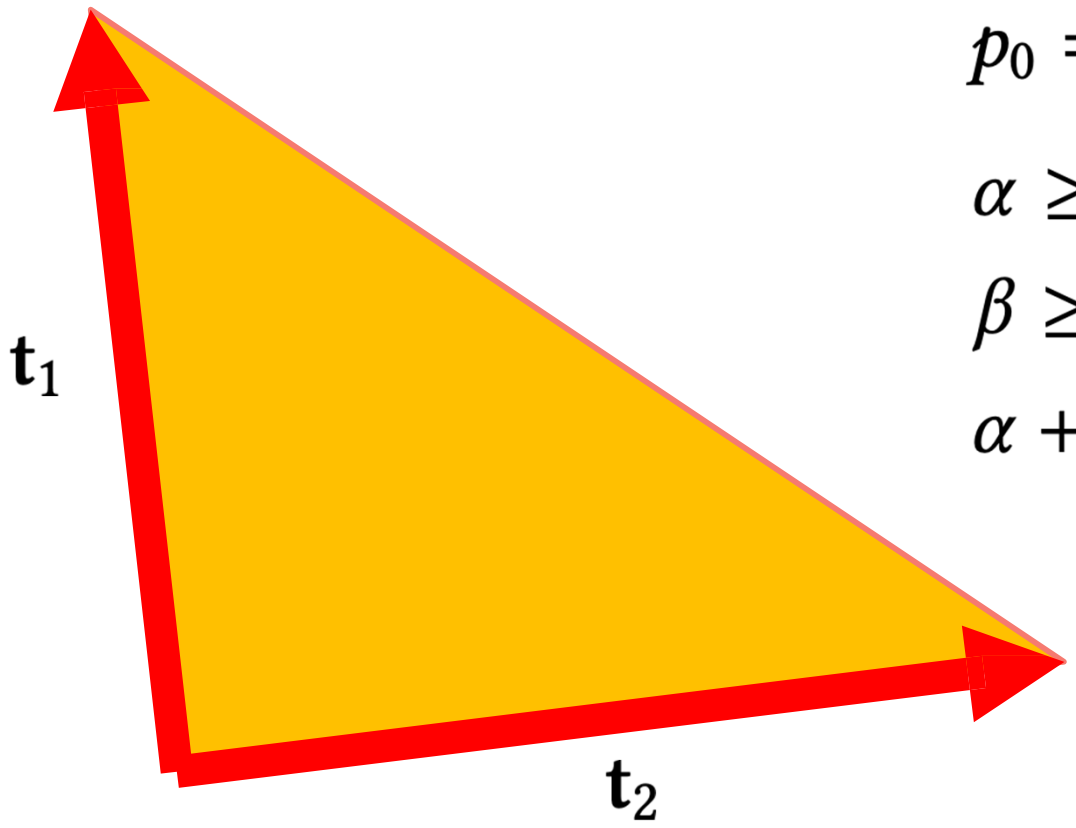


# Equations for a Triangle

$$p_0 = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2$$



# Equations for a Triangle




$$p_0 = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2$$

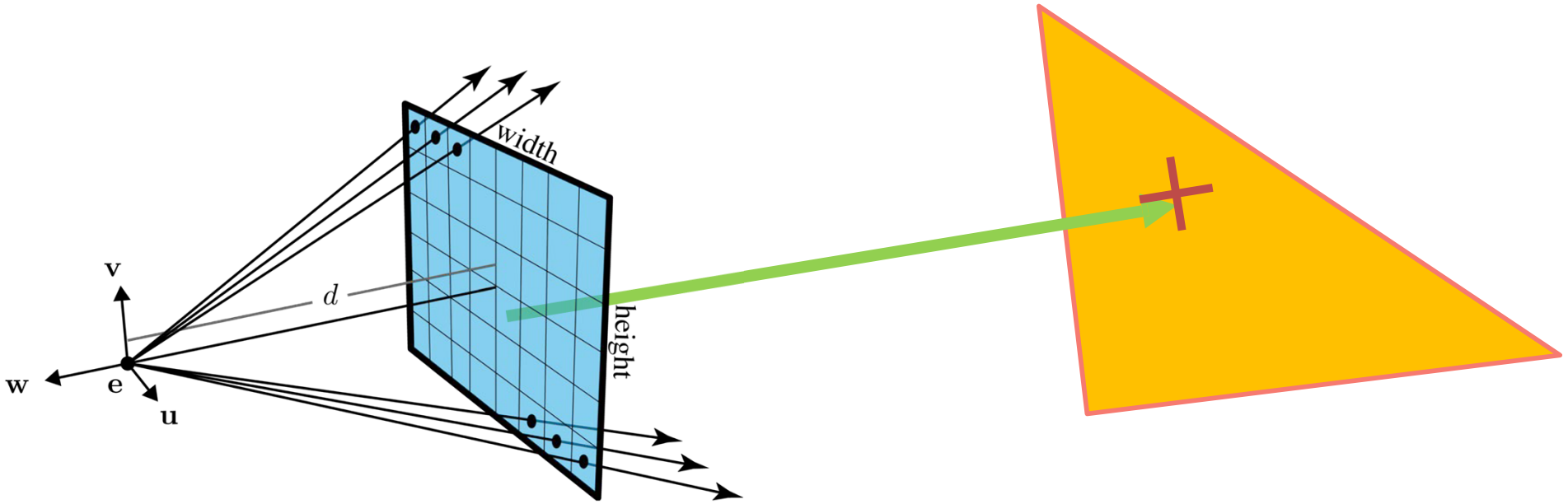
$$\alpha \geq 0$$

$$\beta \geq 0$$

$$\alpha + \beta \leq 1$$

# Intersection with a Triangle (Parametric Surface)

Check via equating point on surface with point on ray



## Intersection with a Triangle (Parametric Surface)

Check via equating point on surface with point on ray

$$\mathbf{p}(t) = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2$$

$$\mathbf{e} + t\vec{\mathbf{d}} = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2$$

$$\mathbf{e} = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2 - t\vec{\mathbf{d}}$$

# Intersection with a Triangle (Parametric Surface)

Check via equating point on surface with point on ray

$$\mathbf{e} = \alpha \mathbf{t}_1 + \beta \mathbf{t}_2 - t\vec{\mathbf{d}}$$

$$\mathbf{e} = \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & -\mathbf{d} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ t \end{bmatrix}$$

Check values of $\alpha, \beta, t$

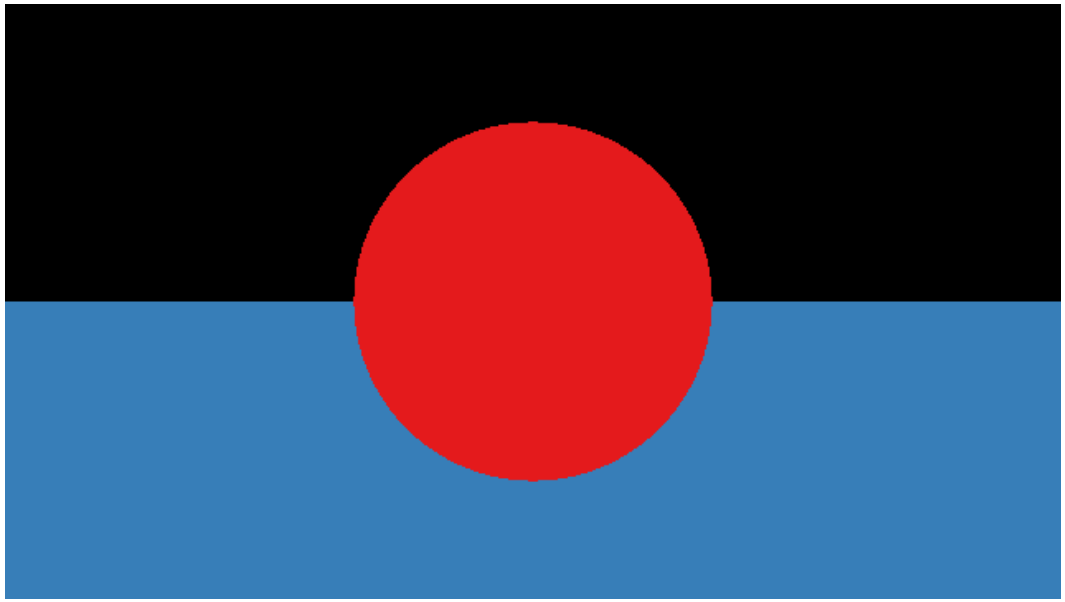
# Ray Casting

```
for each pixel in the image {
    Generate a ray
    for each object in the scene {
        if (Intersect ray with object)
            {  Set pixel colour
        }
    }
}
```

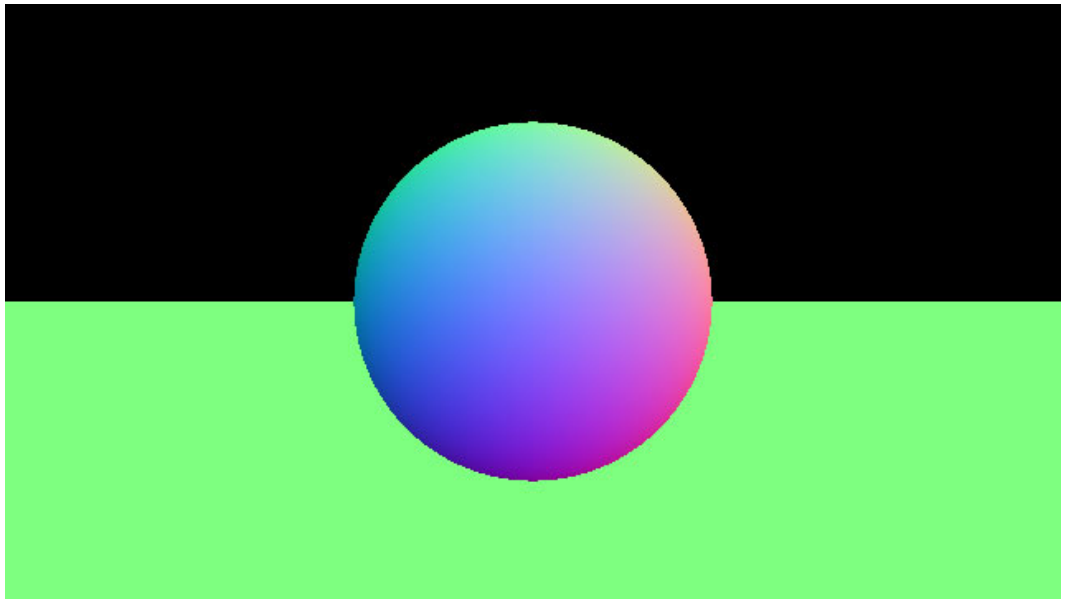
## Output Type

- Object ID
- Surface Normal
- Depth

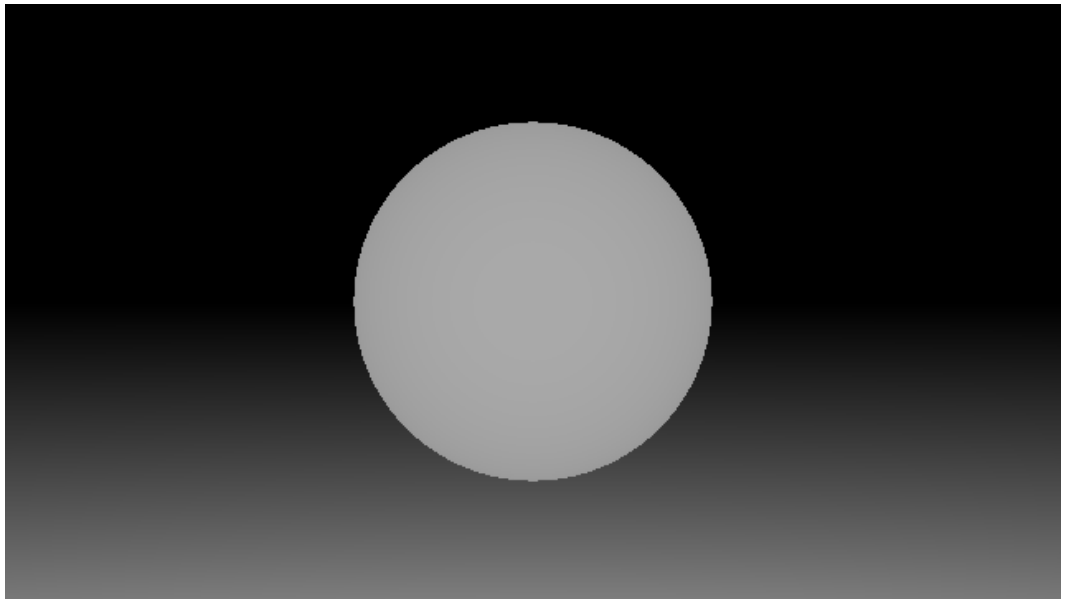
# Output Type: Object ID



# Output Type: Surface Normal



# Output Type: Depth



# Assignment #2 tasks

### `src/viewing_ray.cpp`

Construct a viewing ray given a camera and subscripts to a pixel.

### `src/first_hit.cpp`

Find the first (visible) hit given a ray and a collection of scene objects

### `Sphere::intersect_ray` in `src/Sphere.cpp`

Intersect a sphere with a ray.

### `Plane::intersect_ray` in `src/Plane.cpp`

Intersect a plane with a ray.

### `Triangle::intersect_ray` in `src/Triangle.cpp`

Intersect a triangle with a ray.

### `TriangleSoup::intersect_ray` in `src/TriangleSoup.cpp`

Intersect a triangle soup with a ray.

## Next Week

Ray Tracing