Fall 2015

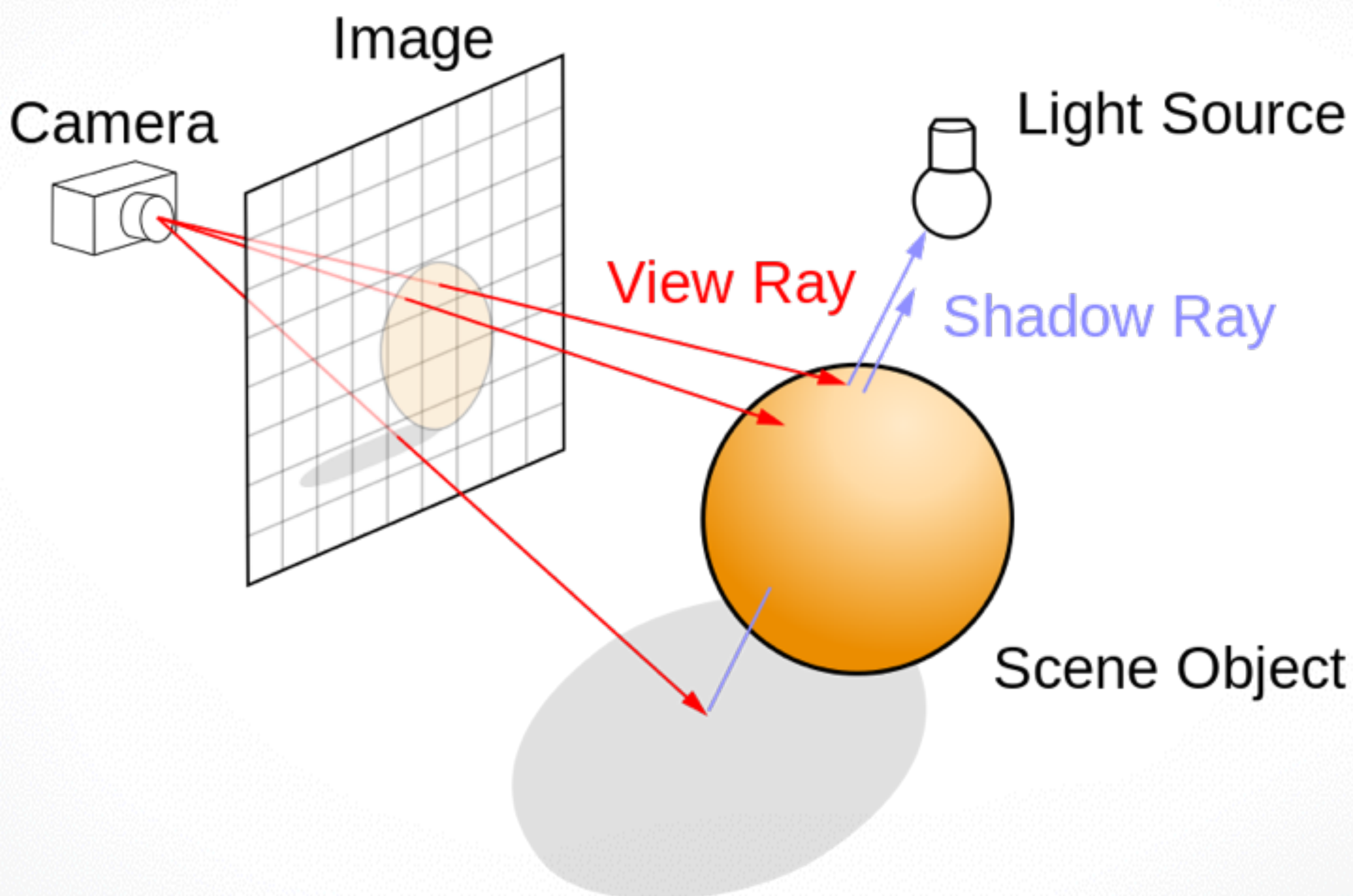
CSCI 420: **Computer Graphics**

Exercise 3. Ray Tracing

Hao Li

http://cs420.hao-li.com

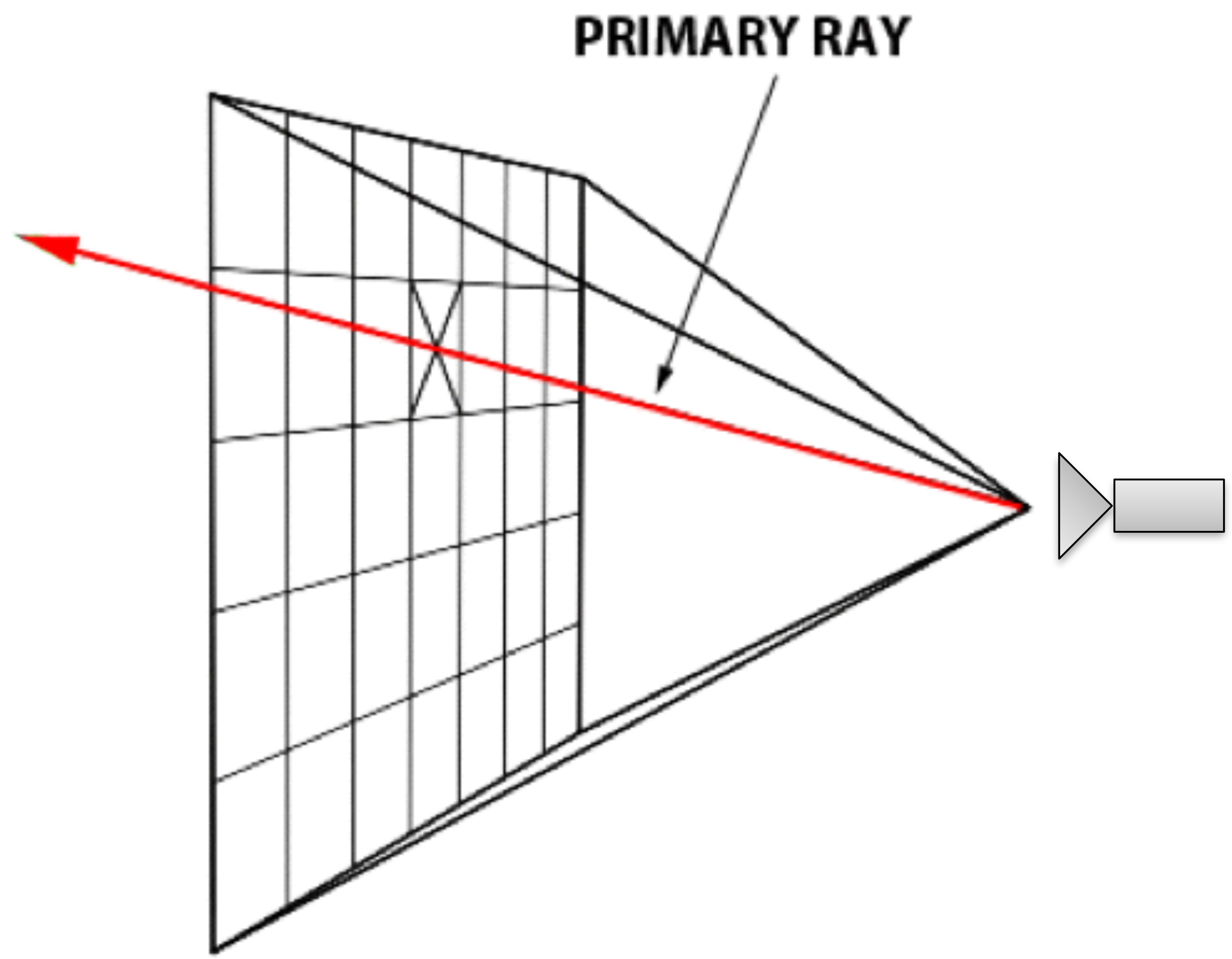
Ray Tracing



Ray Tracing

- Level 1: sent out rays
- Level 2: intersection
- Level 3: illumination

Level 1: Sent out rays



Level 1: Sent out rays

camera position: (0, 0, 0)

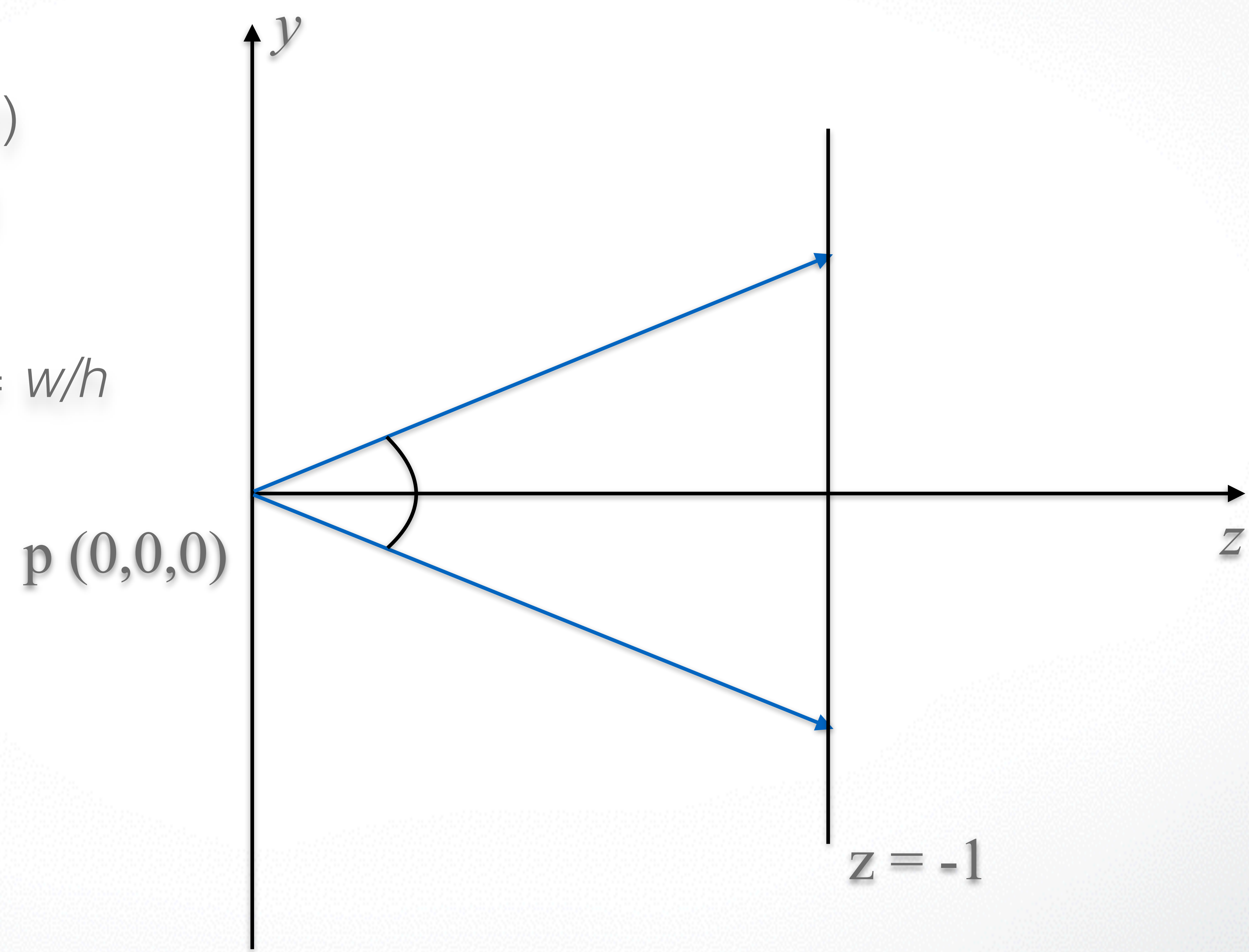
look at: (0, 0, -1)

up vector: (0, 1, 0)

near plane: z = -1

FOV: 60 degree

a = aspect ratio = *w/h*

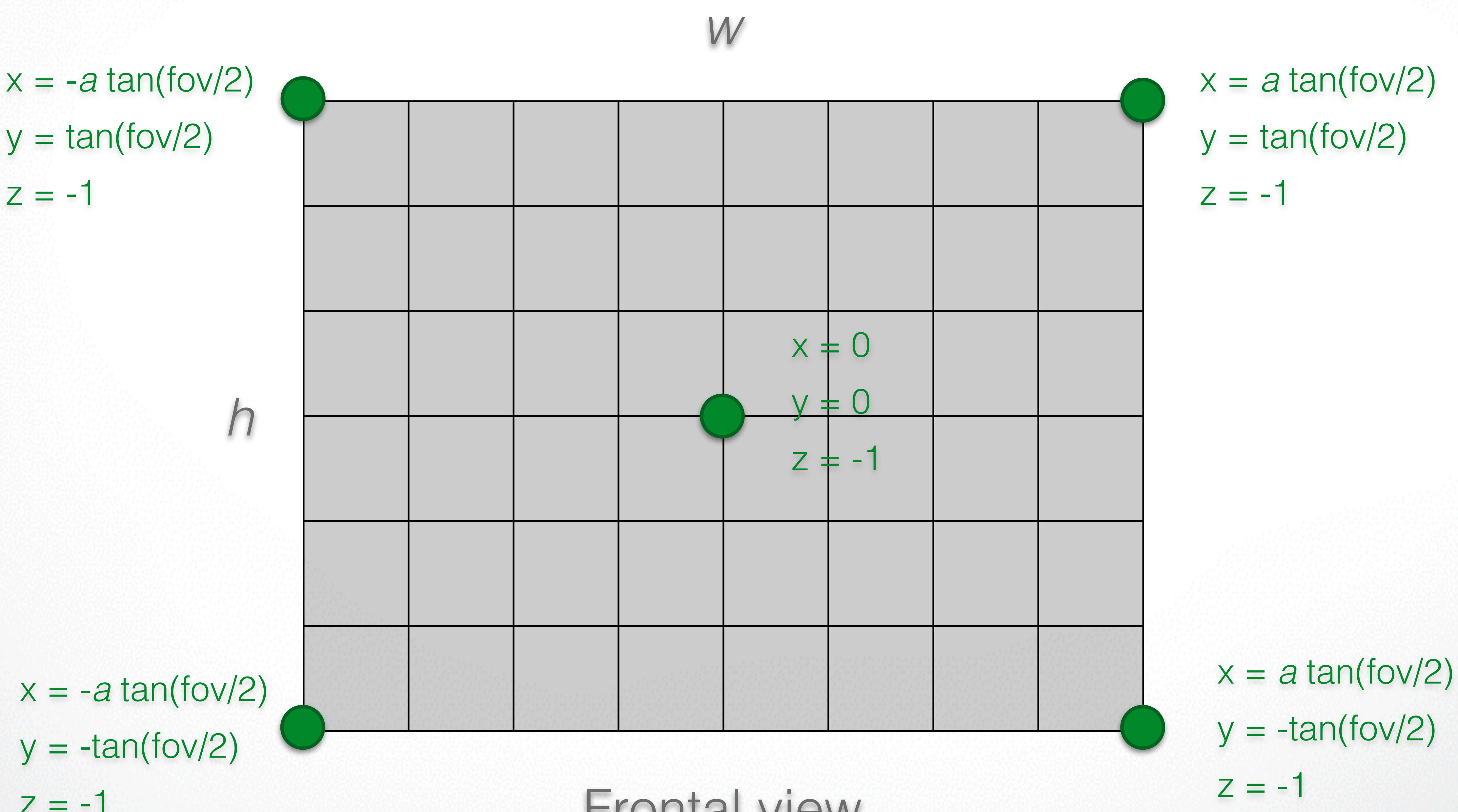


Side view

Level 1: Sent out rays

FOV: 60 degree

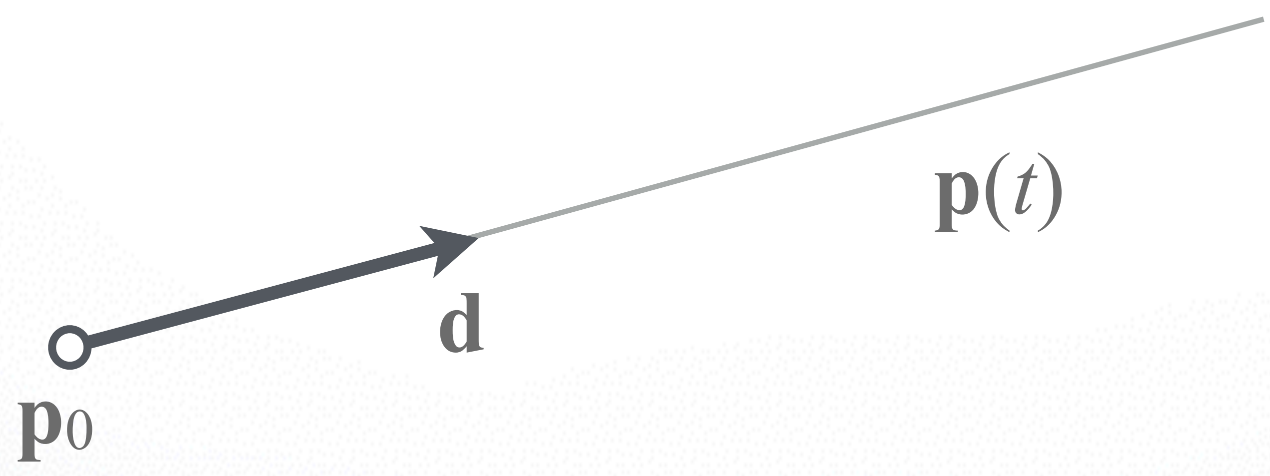
a = aspect ratio = *w*/*h*



Frontal view

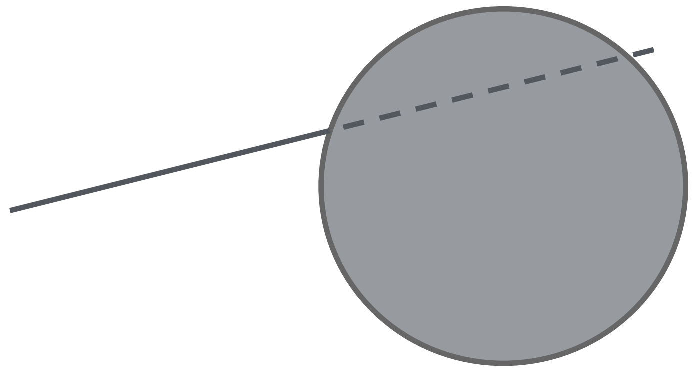
Level 1: Sent out rays

- Ray in parametric form
  - Origin $\mathbf{p}_0 = [x_0 \;\; y_0 \;\; z_0]^T$
  - Direction $\mathbf{d} = [x_d \;\; y_d \;\; z_d]^T$
  - Assume $\mathbf{d}$ is normalized: $x_d \cdot x_d + y_d \cdot y_d + z_d \cdot z_d = 1$
  - Ray $\mathbf{p}(t) = \mathbf{p}_0 + \mathbf{d}t$ for $t > 0$

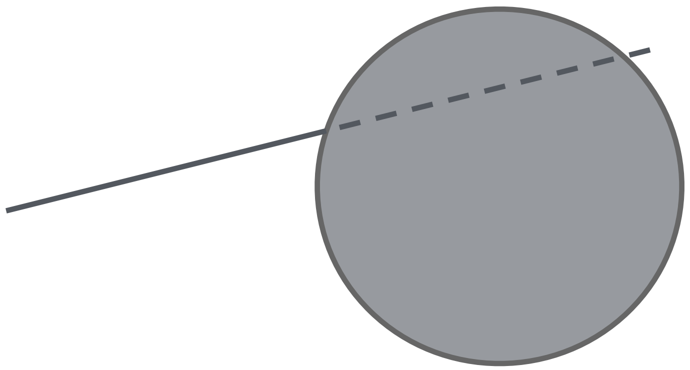


Level 2: Ray-Sphere Intersection

- Define sphere by
  - Center $\mathbf{c} = [x_c \;\; y_c \;\; z_c]^T$
  - Radius r
  - Implicit surface $f(\mathbf{q}) = (x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 - r^2 = 0$



Level 2: Ray-Sphere Intersection



- Define sphere by
  - Center $\mathbf{c} = [x_c \;\; y_c \;\; z_c]^T$
  - Radius r
  - Implicit surface $f(\mathbf{q}) = (x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 - r^2 = 0$
- Plug in ray equations for x, y, z

$$x = x_0 + x_d\, t, \qquad y = y_0 + y_d\, t, \qquad z = z_0 + z_d\, t$$

- Obtain a scalar equation for t

$$(x_0 + x_d\, t - x_c)^2 + (y_0 + y_d\, t - y_c)^2 + (z_0 + z_d\, t - z_c)^2 - r^2 = 0$$

Level 2: Ray-Sphere Intersection

- Simplify to $at^2 + bt + c = 0$

  where
$$a = x_d^2 + y_d^2 + z_d^2 = 1 \qquad \text{since } |d| = 1$$
$$b = 2(x_d(x_0 - x_c) + y_d(y_0 - y_c) + z_d(z_0 - z_c))$$
$$c = (x_0 - x_c)^2 + (y_0 - y_c)^2 + (z_0 - z_c)^2 - r^2$$

- Solve to obtain t_0, t_1
$$t_{0,1} = \frac{-b \pm \sqrt{b^2 - 4c}}{2}$$

- Calculate $b^2 - 4c$, abort if negative
- Check if $t_0, t_1 > 0$. Return $min(t_0, t_1)$

Level 2: Ray-Triangle Intersection

- Method 1:
  - Find intersection of the ray and the plane which the triangle lies on.
  - Determine the ray-plane intersection point is in/out of the triangle.
- Method 2:
  - Fast, Minimum Storage Ray/Triangle Intersection [Moller et al. 1997].

Level 2: Ray-Triangle Intersection

- Ray: $p(t) = p + dt \;\; (t > 0)$
- Triangle (barycentric coordinates):

$$p(u, v) = (1 - u - v) * p_0 + u * p_1 + v * p_2$$

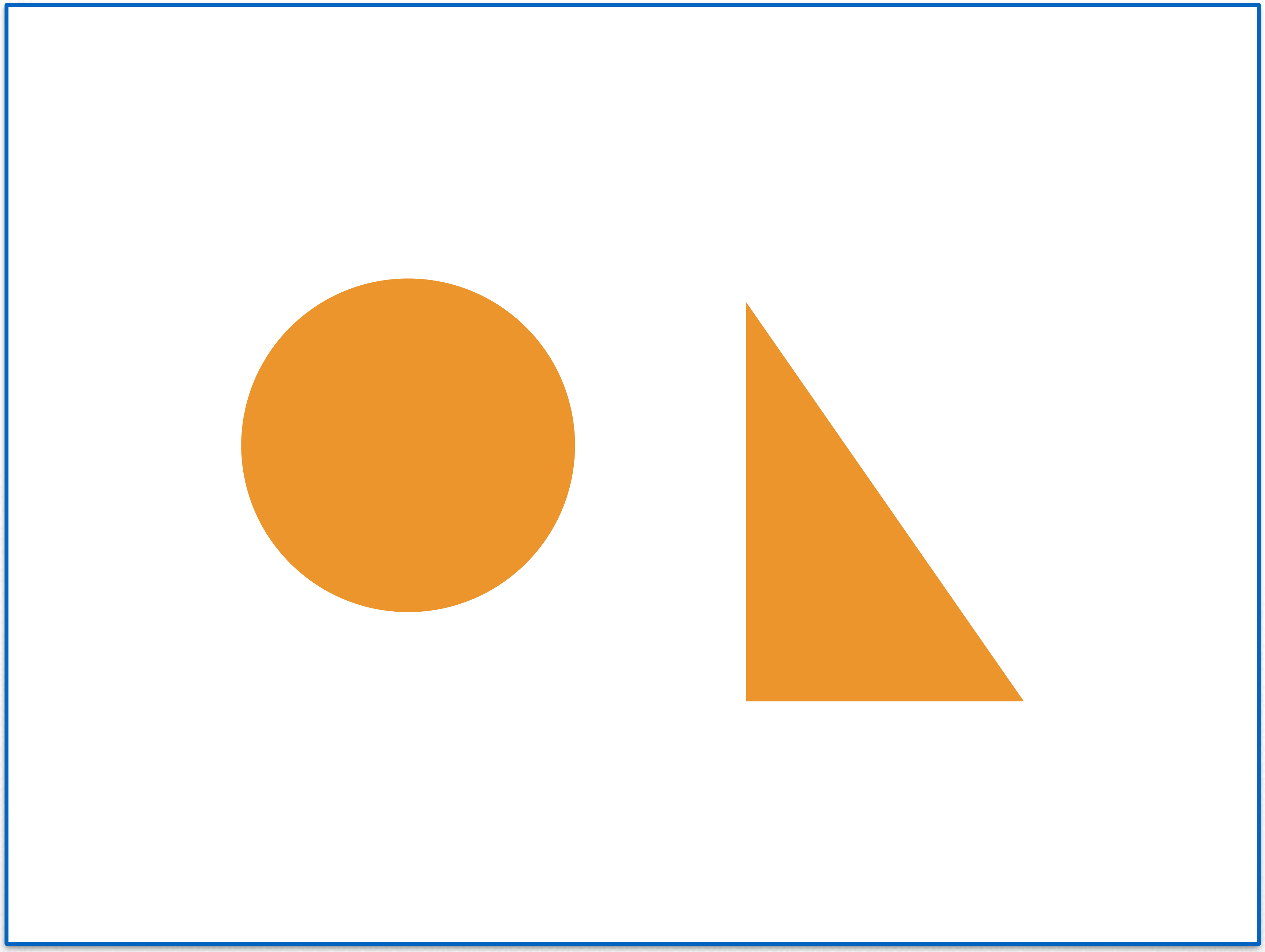
$$(u \geq 0, v \geq 0, u + v \leq 1)$$

$$p + dt = (1 - u - v) * p_0 + u * p_1 + v * p_2$$

$$[-d, p_1 - p_0, p_2 - p_0] \begin{bmatrix} t \\ u \\ v \end{bmatrix} = p - p_0$$

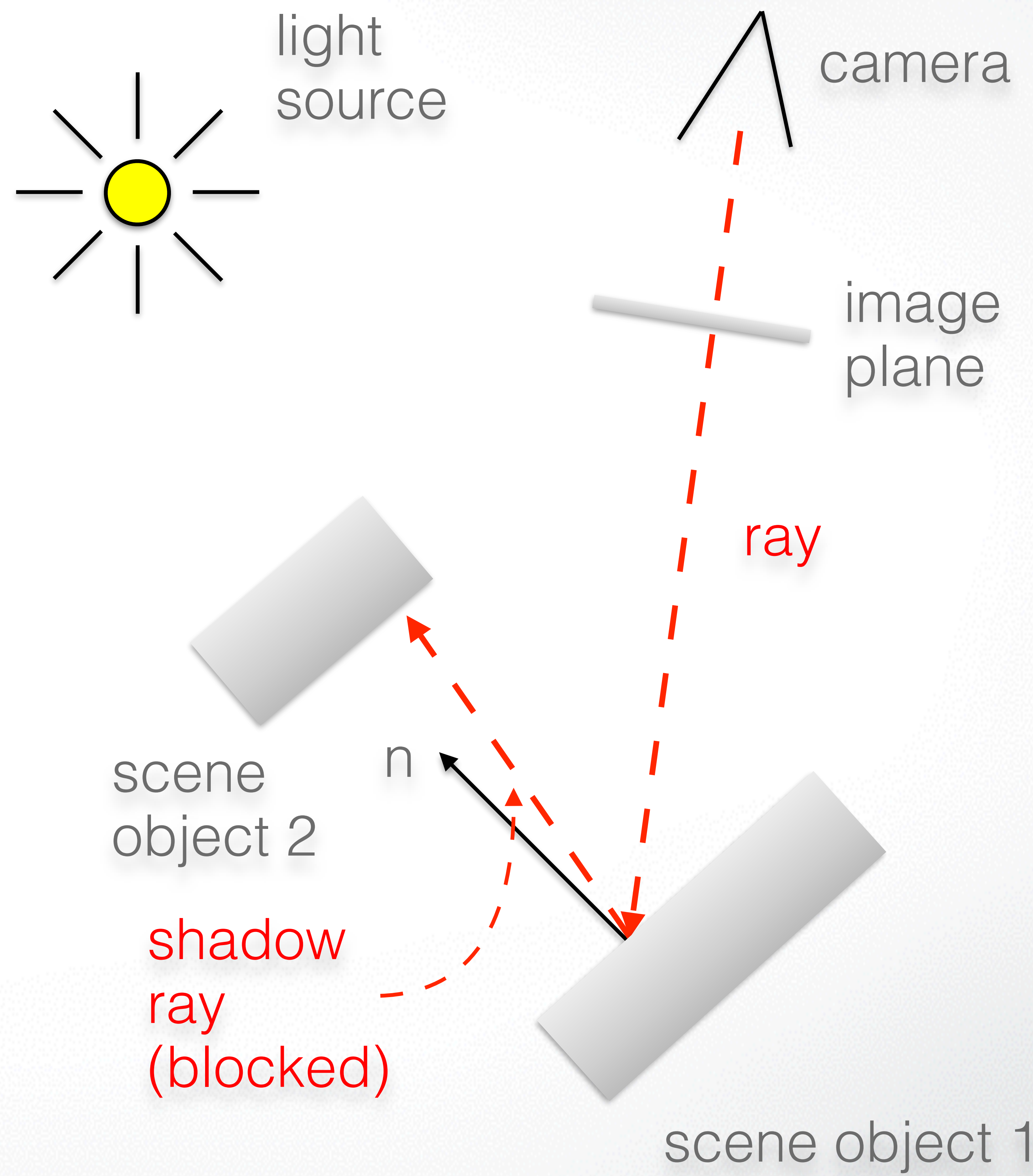
Level 2: Intersection

- Test your intersection code before illumination computation
- Use small image size to test

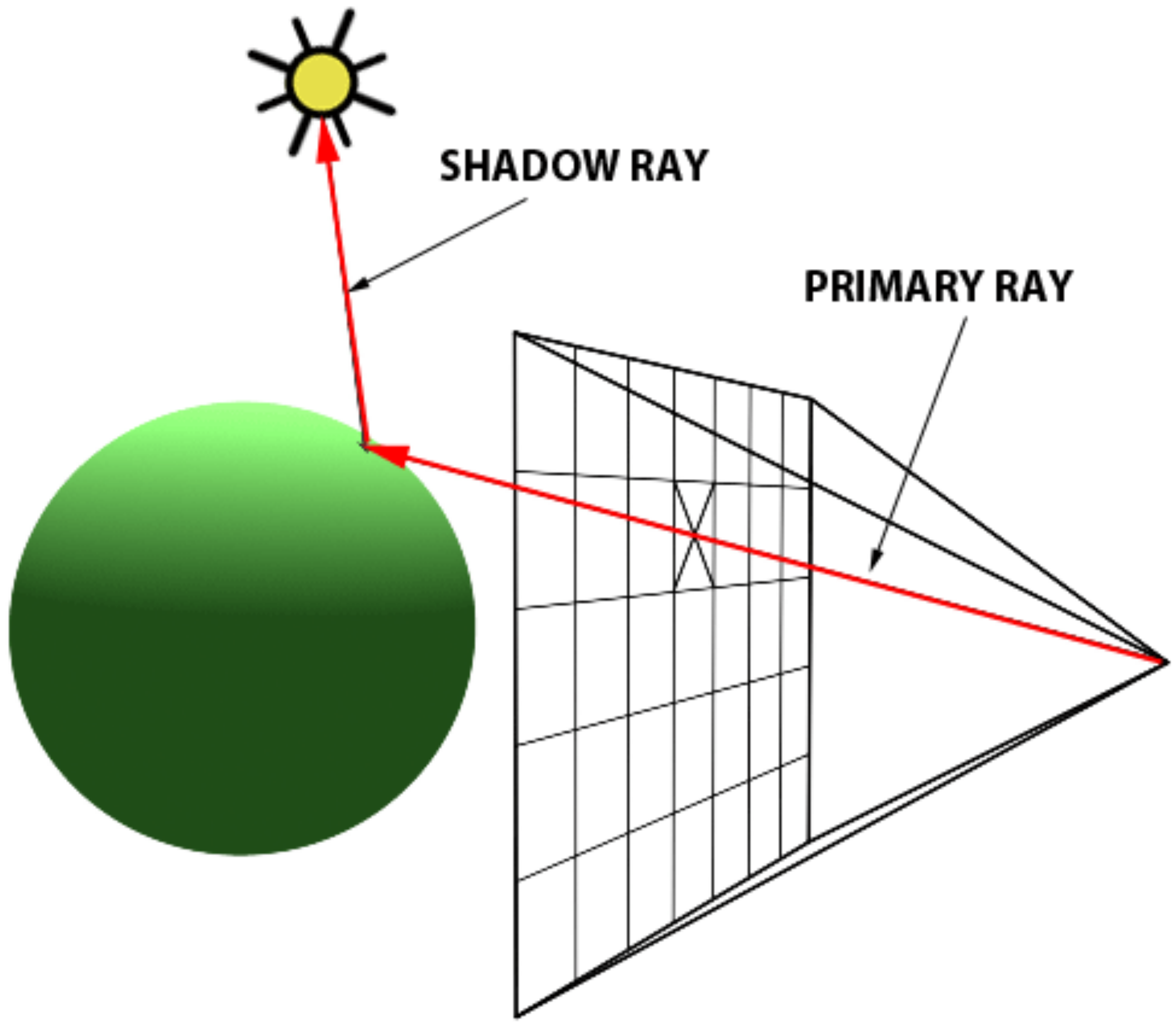


Level 3: Illumination - shadow rays

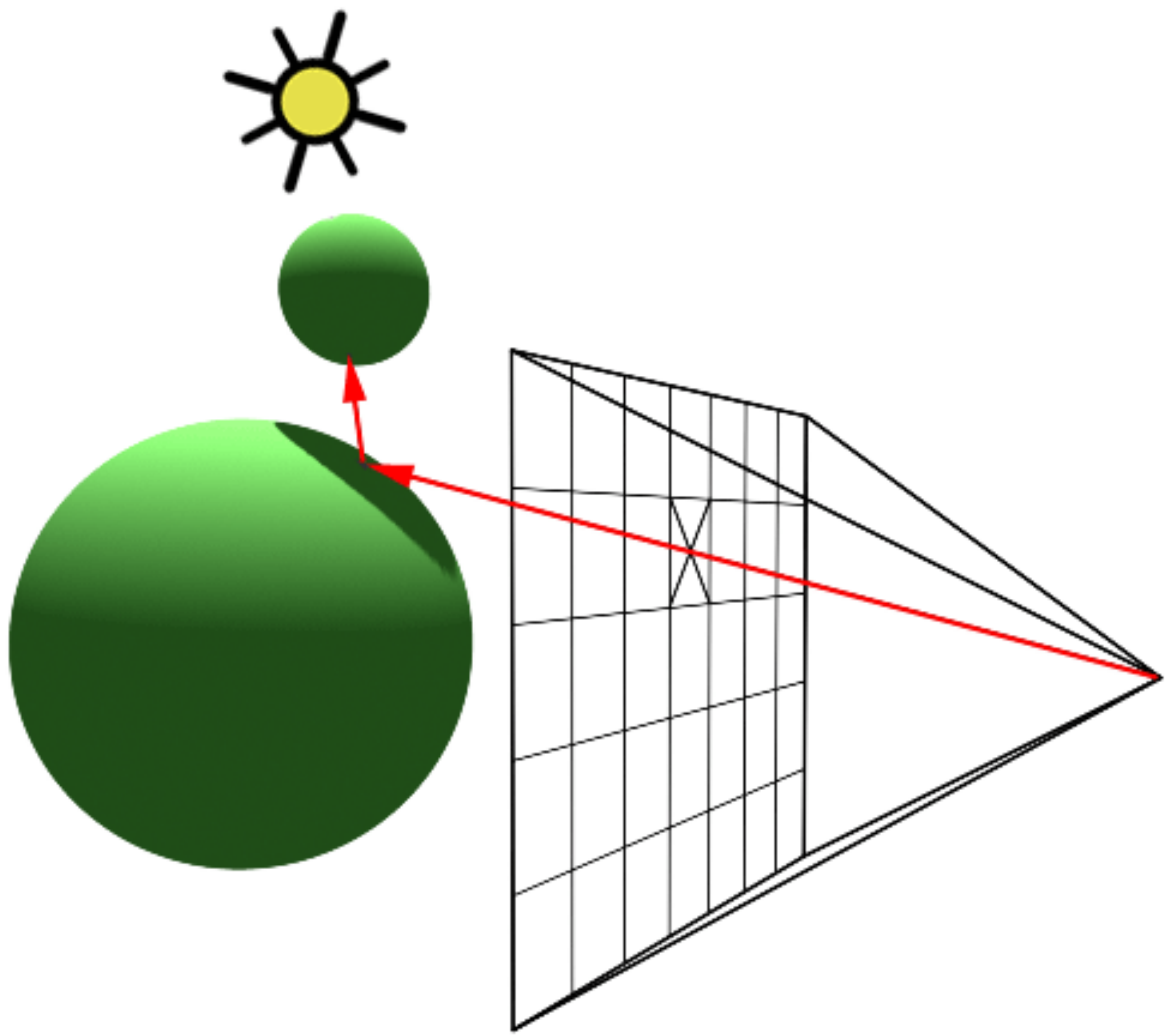
- Determine if light "really" hits surface point
- Cast shadow ray from surface point to each light
- If shadow ray hits opaque object, no contribution from that light



Level 3: Illumination - shadow rays



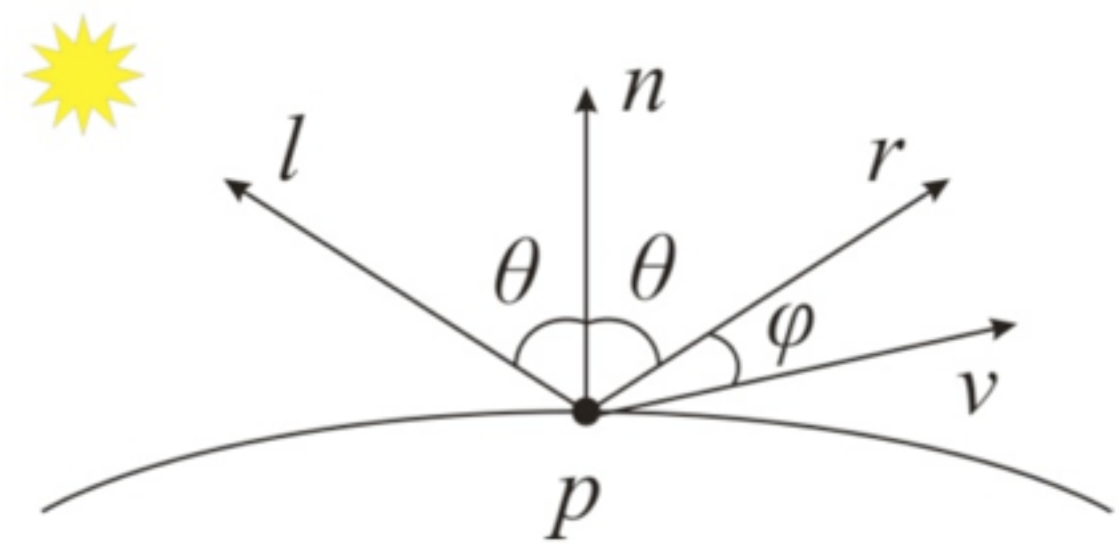
Level 3: Illumination - shadow rays



Level 3: Illumination - Phong shading

$$I = L(k_d(l \cdot n) + k_s(r \cdot v)^\alpha)$$

- L: light coefficient
- l: dirToLight
- n: normal
- v: dirToCamera
- r: reflectDir = $2(l \cdot n)n - l$



Level 3: Illumination - Phong shading

- Sphere normal:

$$n = \frac{1}{r}[(x_i - x_c) \quad (y_i - y_c) \quad (z_i - z_c)]^T$$

- Triangle normal:

$$p(u, v) = (1 - u - v) * p_0 + u * p_1 + v * p_2$$

$$(u \geq 0, v \geq 0, u + v \leq 1)$$

$$n(u, v) = (1 - u - v) * n_0 + u * n_1 + v * n_2$$

Notice

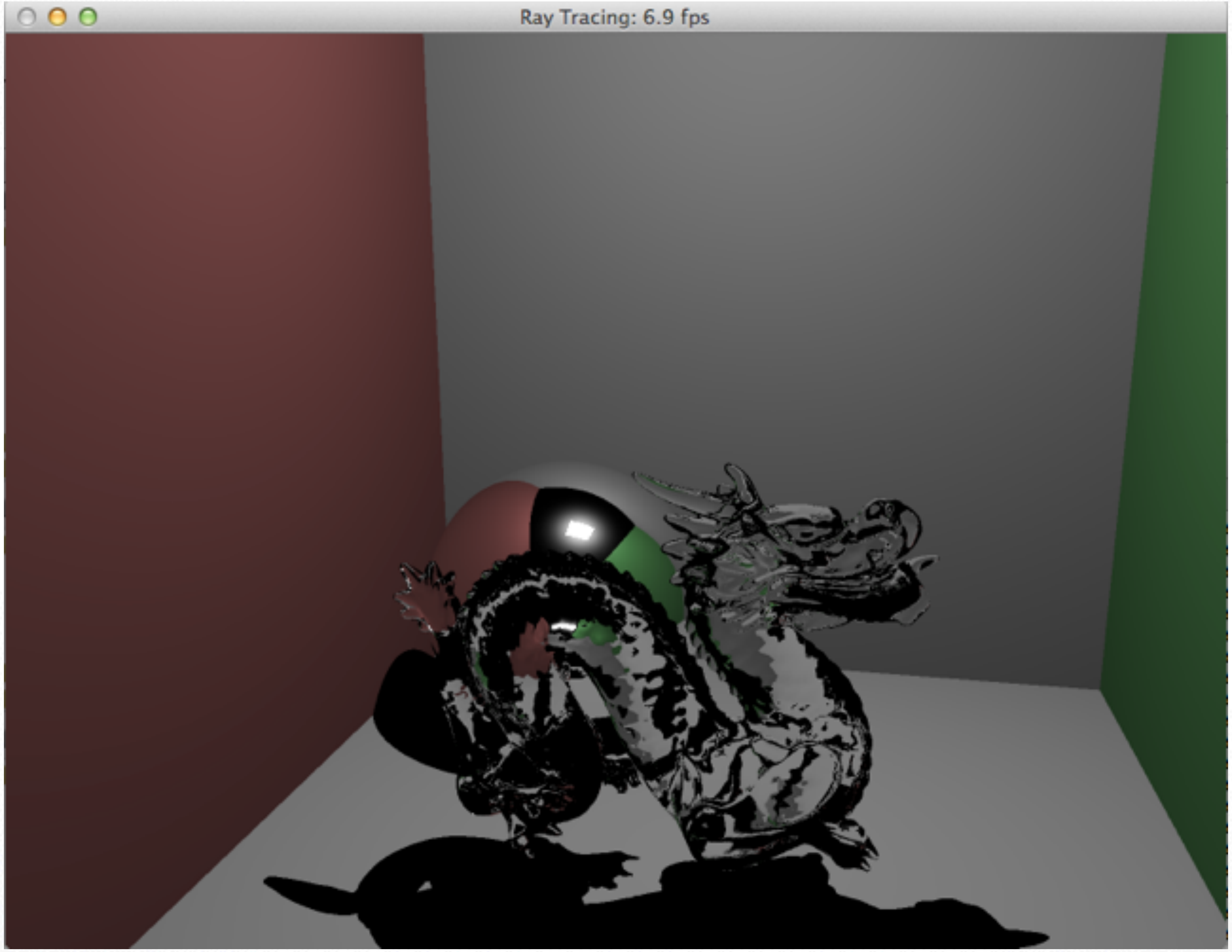
- Ensure B!=0 when doing A/B
- Before call sqrt(A), make sure A>=0
- Remember to normalize direction vector
- Remember to check len(dir)!=0 before normalize dir
- Floating-point operations are not accurate

```
if (a>0.0f)
#define EPS 10e-8f
if (a>EPS)
```

Extra credit

- Recursive reflection
- Recursive refraction
- Antialiasing
- Soft shadows
- Animation
- Motion blur
- Using Spatial structure to accelerate
- Parallel computing to accelerate
- …

Demo



Submission

- Deadline: **Tues. Nov 24, 2015 11:59 pm**
- **Start this assignment as soon as you can**
- Upload a .zip compressed file named “Exercise3-YourName.zip” to blackboard
- Include your code with comments
- Include a readme file
- Include output still images

Contact

- Emails: olszewsk@usc.edu, lingyu.wei@usc.edu
- When you sent emails, add “CSCI420” in the title, and suggest to sent both of us
- Highly recommended to post your questions on blackboard

Enjoy it!