Announcements

- **MP4** released - 4.1 due on Nov 2, 11, 18 @11am

- Project proposals - graded. If you didn't get 100 - find me and talk to me.

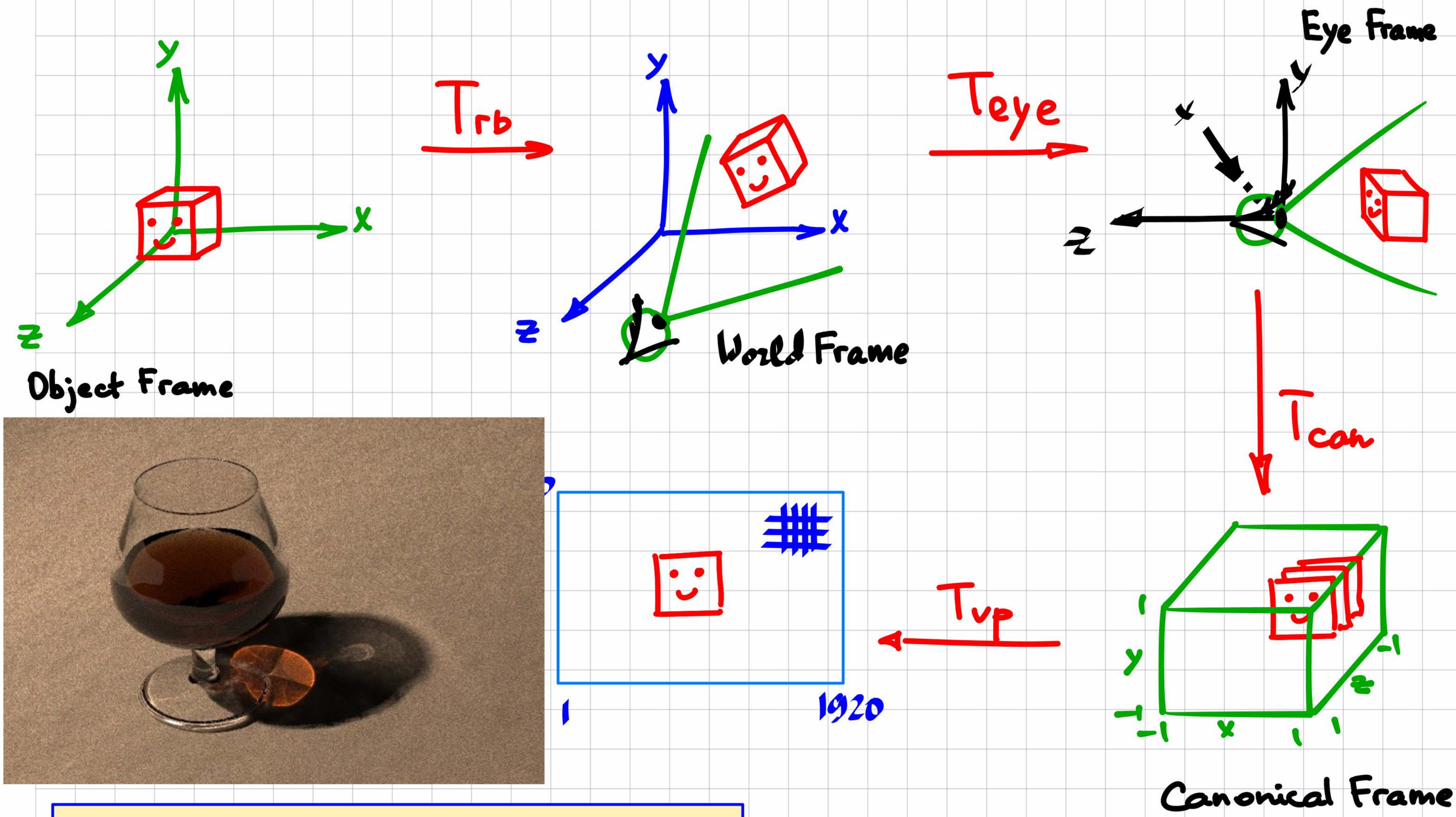
Final project upcoming deadlines:

- **Oct 31**, picture-title.

- **Nov 14**, a short video of your progress.

CS418

Visual Rendering



$$T = \overset{T_{dist}}{T_{vp}} \cdot T_{can} \cdot \overset{T_L \text{ or } T_R}{T_{eye}} \cdot T_{rB}$$

This was geometry; LaValle, Ch 3

Now, read LaValle, Ch 7

Object-Order Rending

(pixel-by-pixel)

Image-order rendering stages:

1. Ray generation
2. Ray intersection
3. Shading: assign RGB values

(triangle-by-triangle)

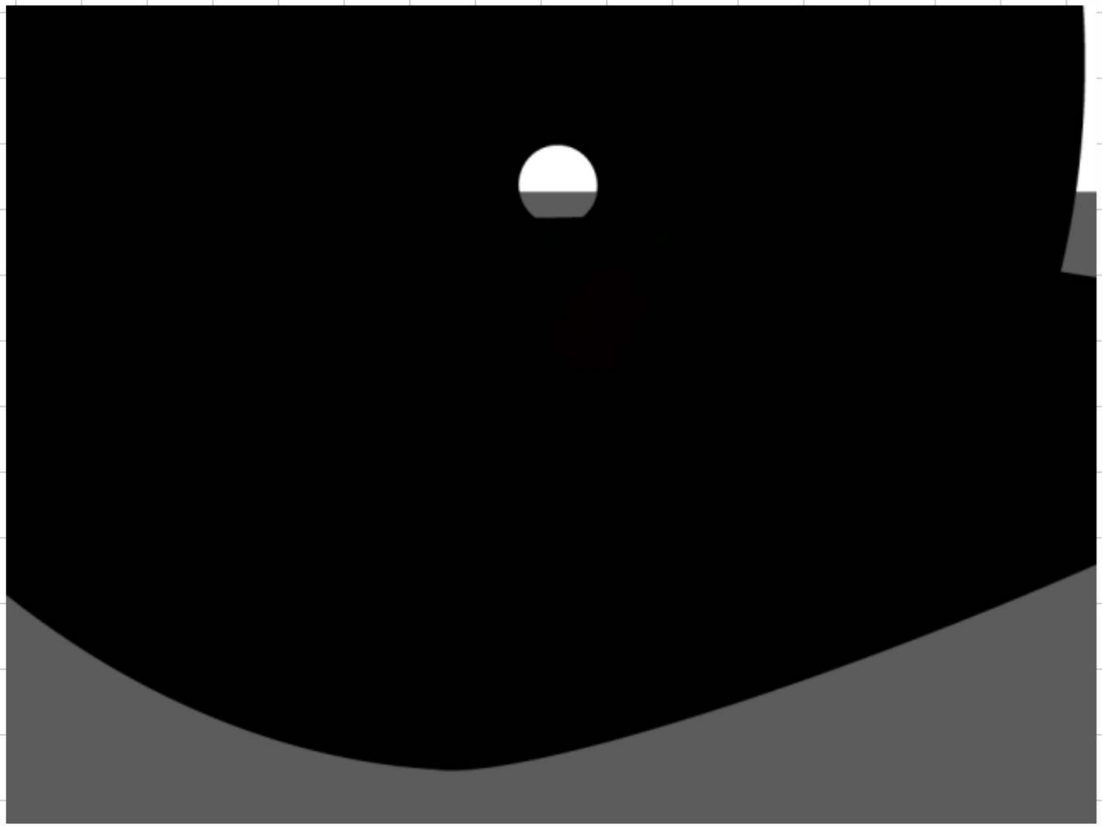
Object-order rendering stages:

1. Rasterization
2. Depth order
3. Shading: assign RGB values

Image-Order Rendering: Shading

Ambient light component:

$$L = \underbrace{k_d \cdot I \cdot \max(0, n \cdot l)}_{\text{diffuse}} + \underbrace{k_s \cdot I \cdot \max(0, n \cdot h)}_{\text{specular}} + \underbrace{k_a \cdot I_a}_{\text{ambient}}$$



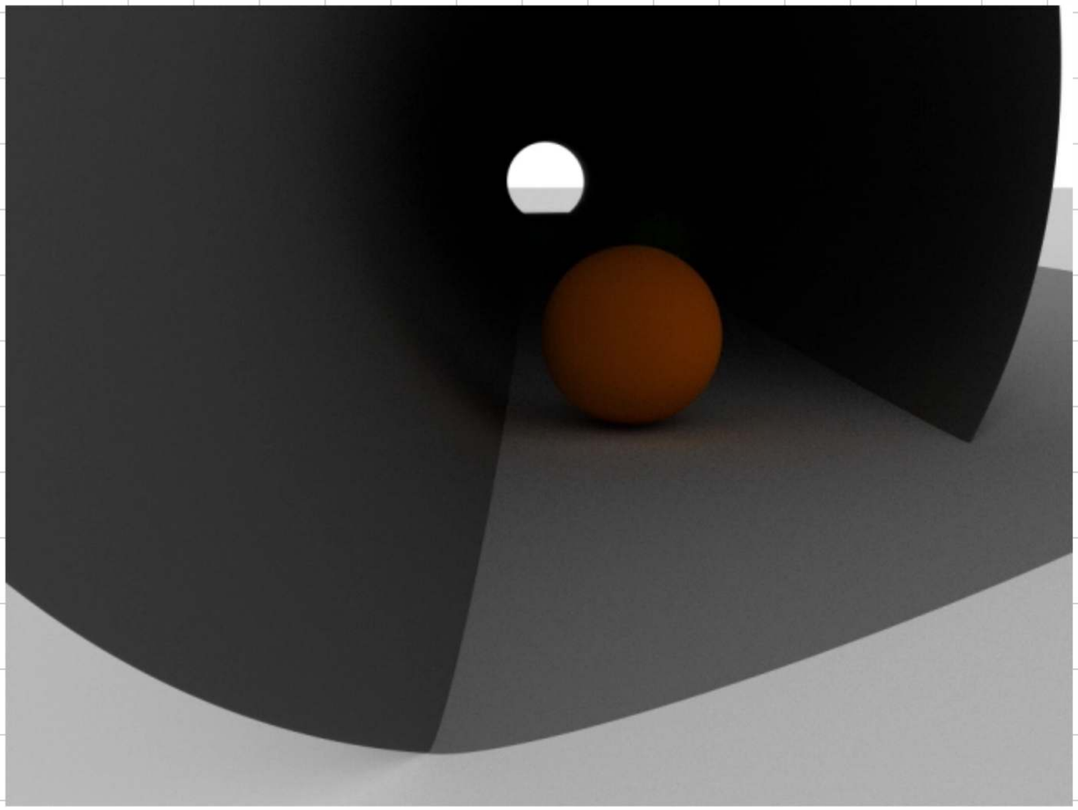
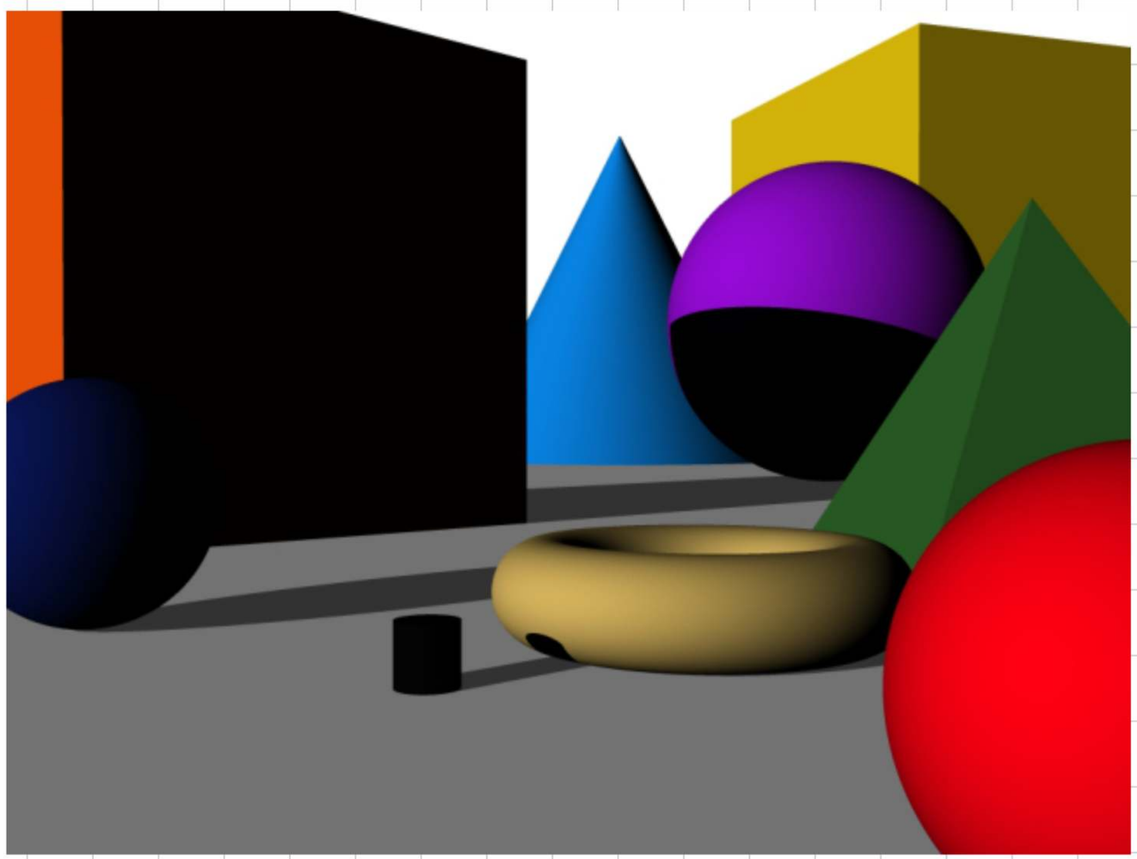
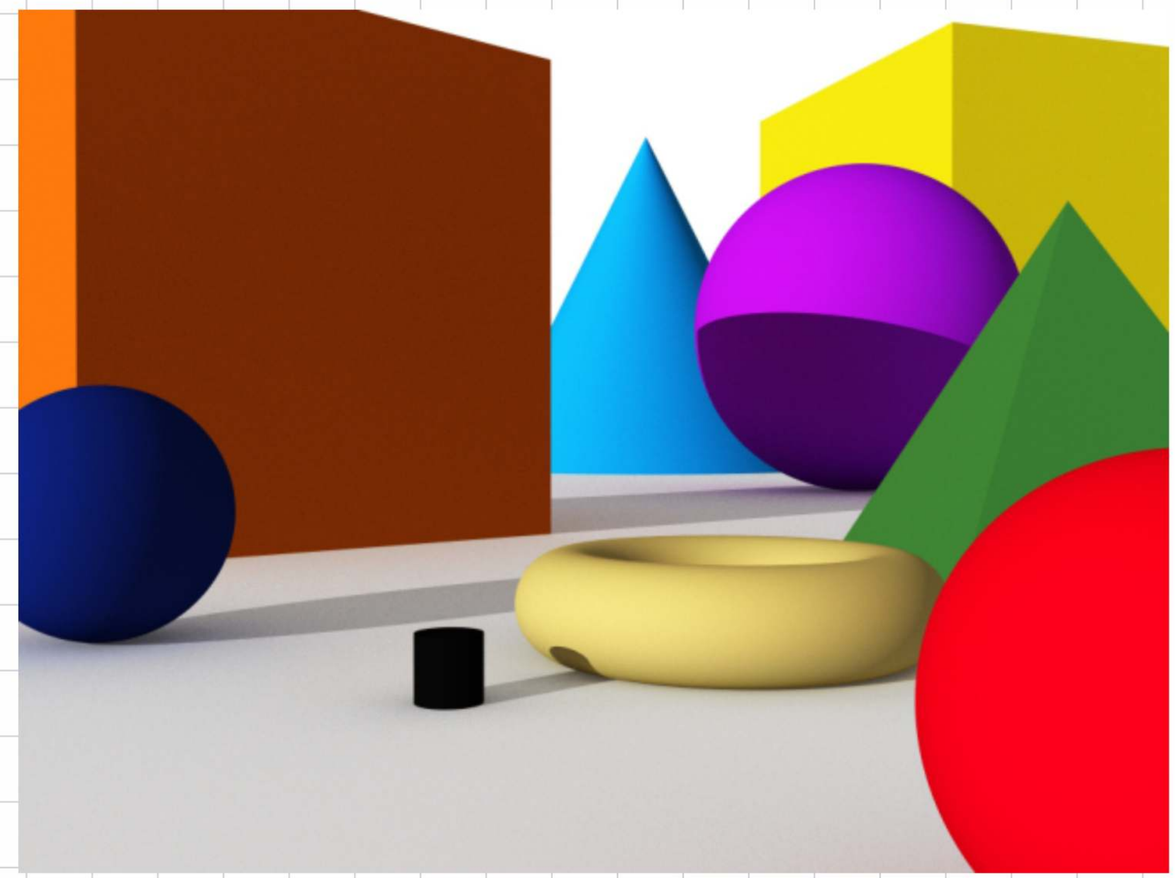


Image-Order Rendering: Shading

$$L = \underbrace{k_d \cdot I \cdot \max(0, n \cdot l)}_{\text{diffuse}} + \underbrace{k_s \cdot I \cdot \max(0, n \cdot h)}_{\text{specular}} + \underbrace{k_a \cdot I_a}_{\text{ambient}}$$





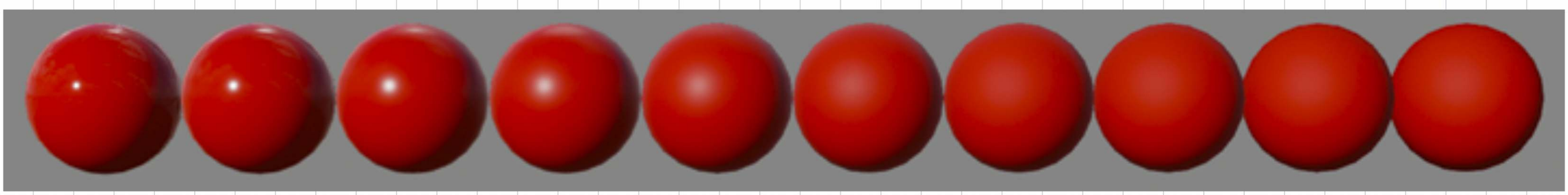


Image-Order Rending: Shading

Multiple light sources:

$$L = \sum_{i=1}^{N} \Big(\underbrace{k_d \cdot I \cdot \max(0, n \cdot l)}_{\text{diffuse}} + \underbrace{k_s \cdot I \cdot \max(0, n \cdot h)}_{\text{specular}} \Big) + \underbrace{k_a \cdot I_a}_{\text{ambient}}$$

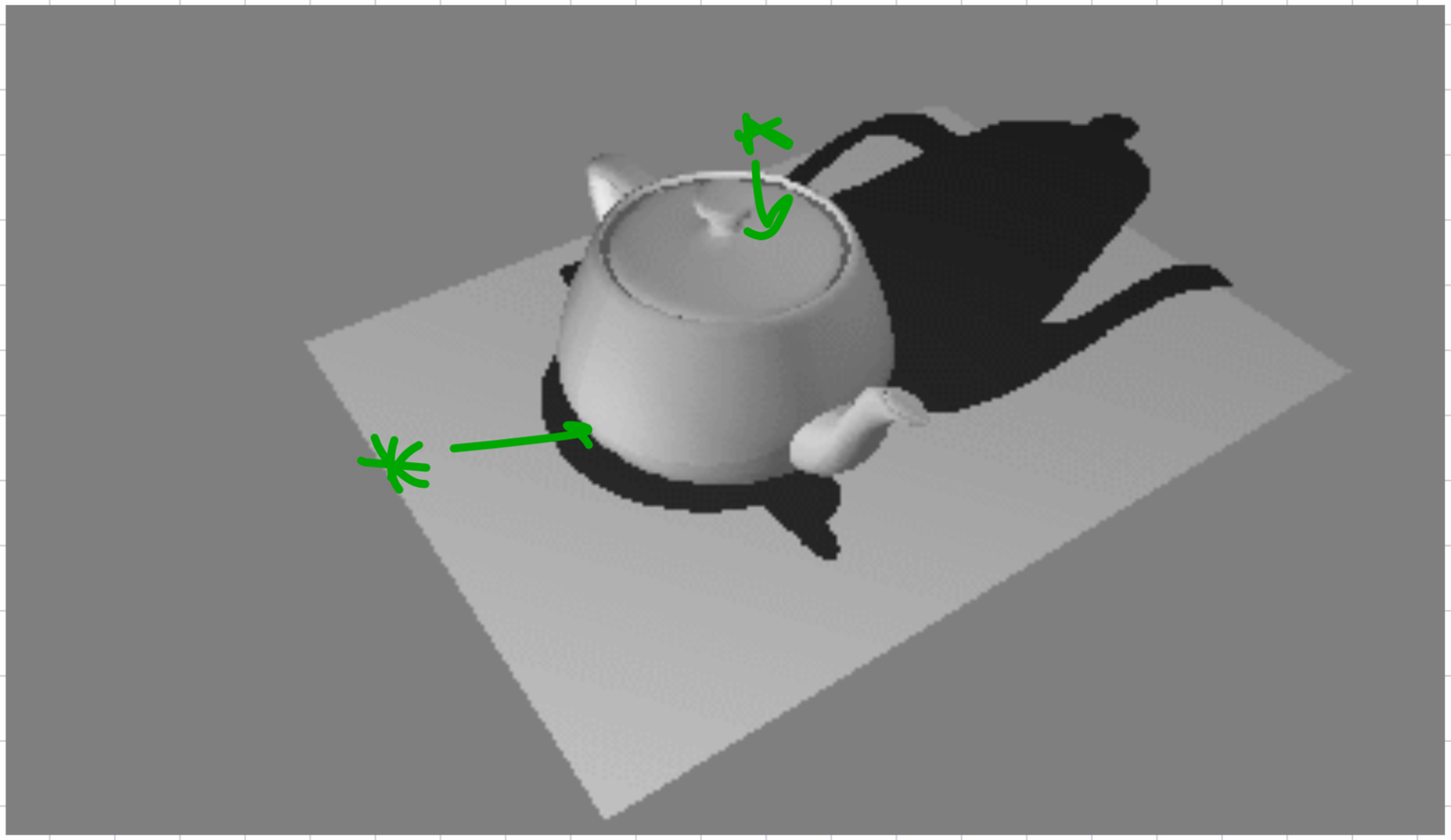
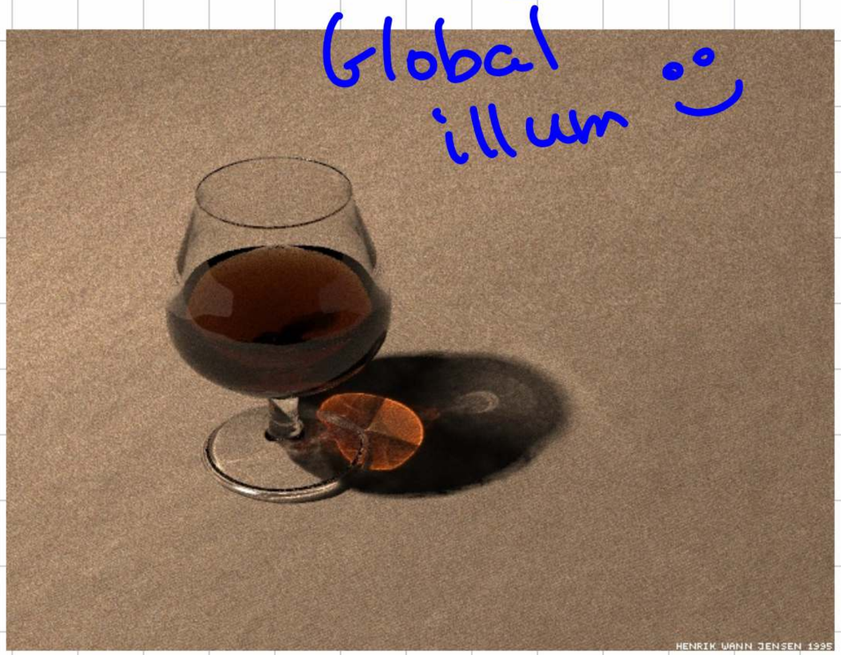


Image-Order Rending: Shading

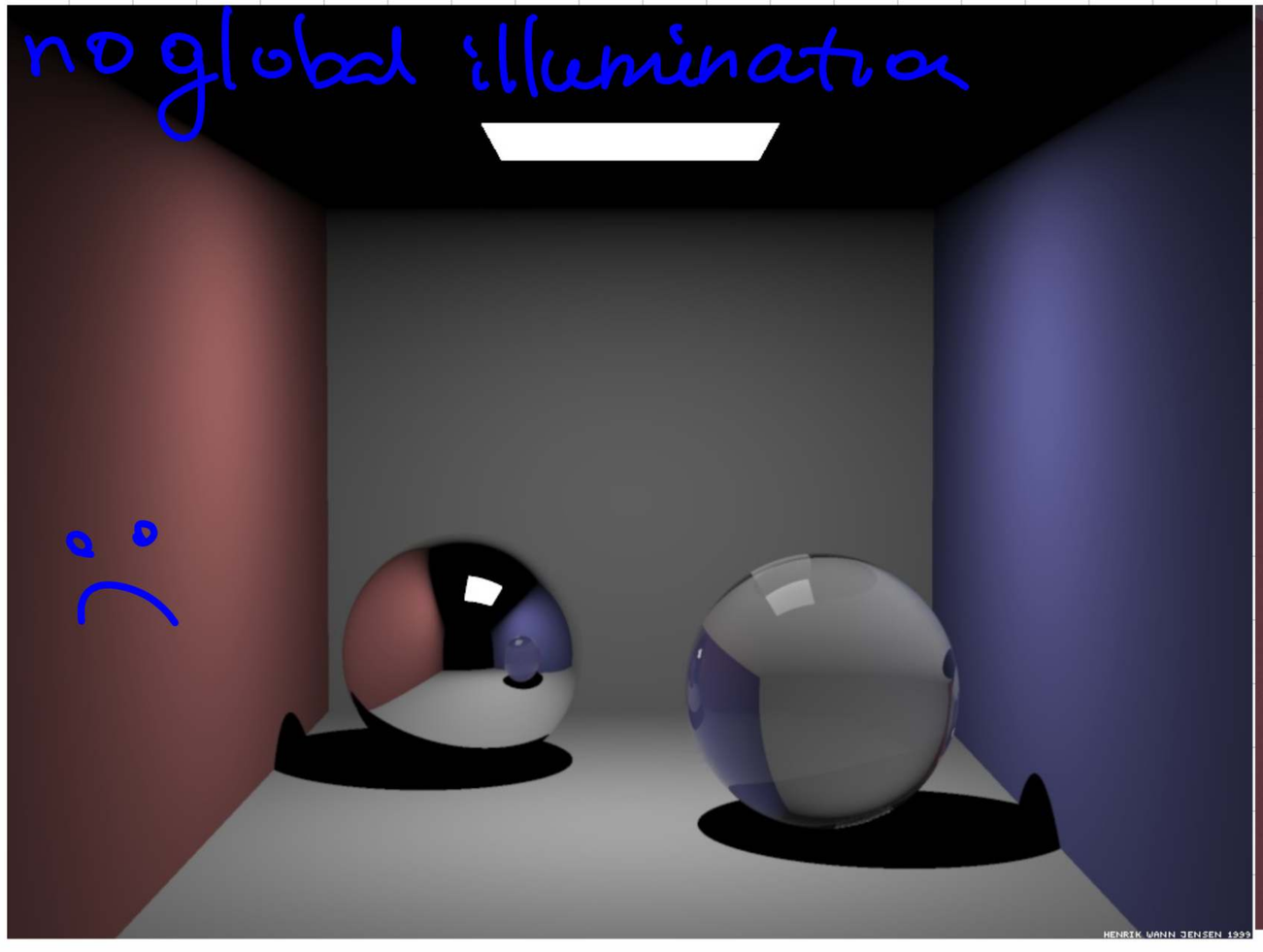
Alternative approach: global illumination

- Handling multiple reflections (how many is enough?)
- Very expensive

diffraction, refraction
transparancy, translucency



http://graphics.stanford.edu/~henrik/images/global.html



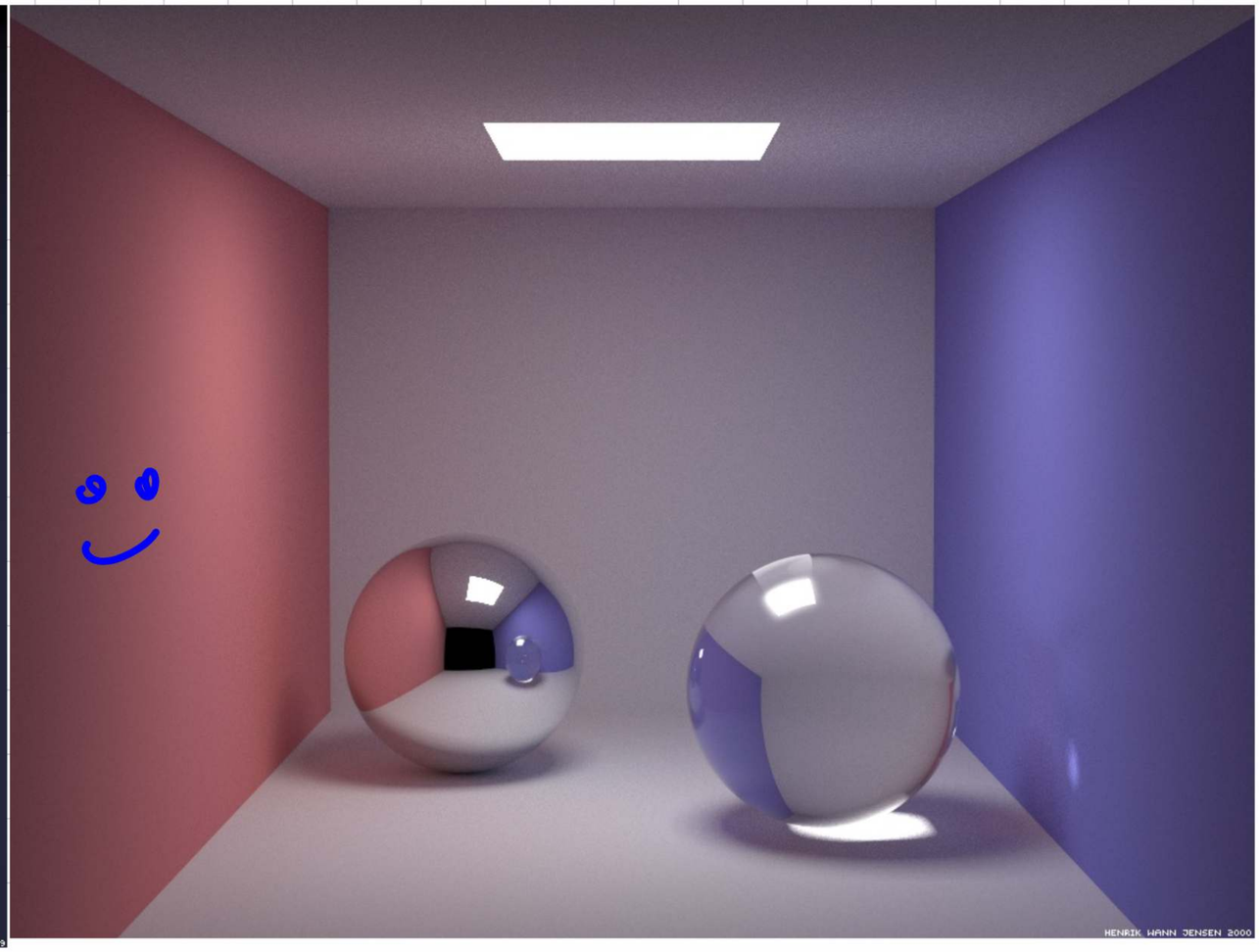
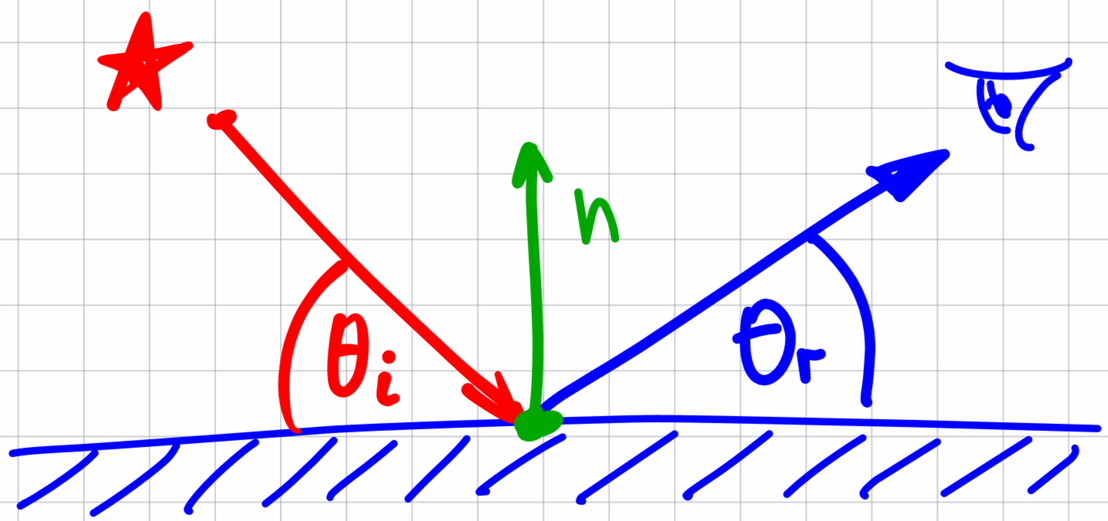
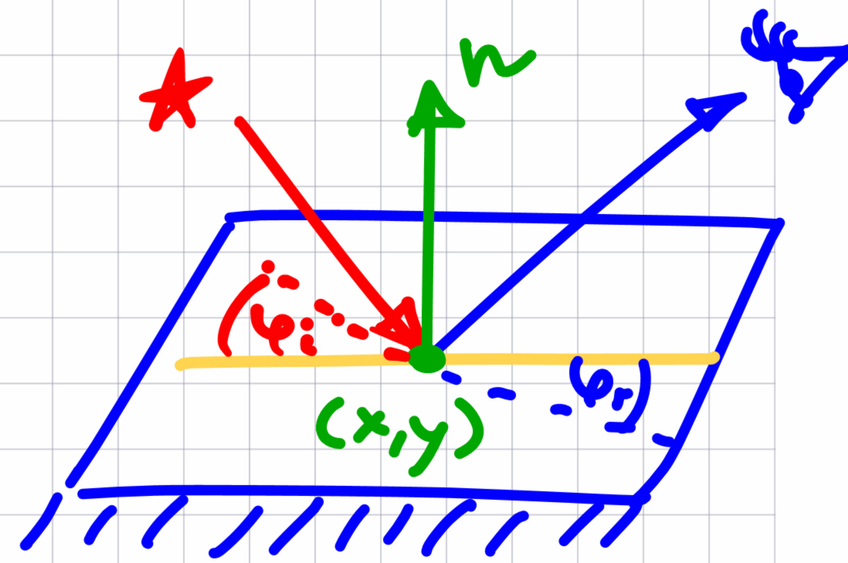


Image-Order Rending: Shading

BRDF: Bidirectional Reflectance Distribution Function

(most photorealistic rendering)





reflected energy

$$f(\theta_i, \varphi_i, \theta_r, \varphi_r) = \frac{\text{radiance}}{\text{irradiance}}$$

(x, y, λ) ?

received energy

Image-Order Rending: Shading in VR

- higher resolution !
- more FPS !
- two eyes !
- faster moving viewing point(s)
- wider FOV !

Object-Order Rending

Image-order rendering stages:

1. Ray generation
2. Ray intersection
3. Shading: assign RGB values

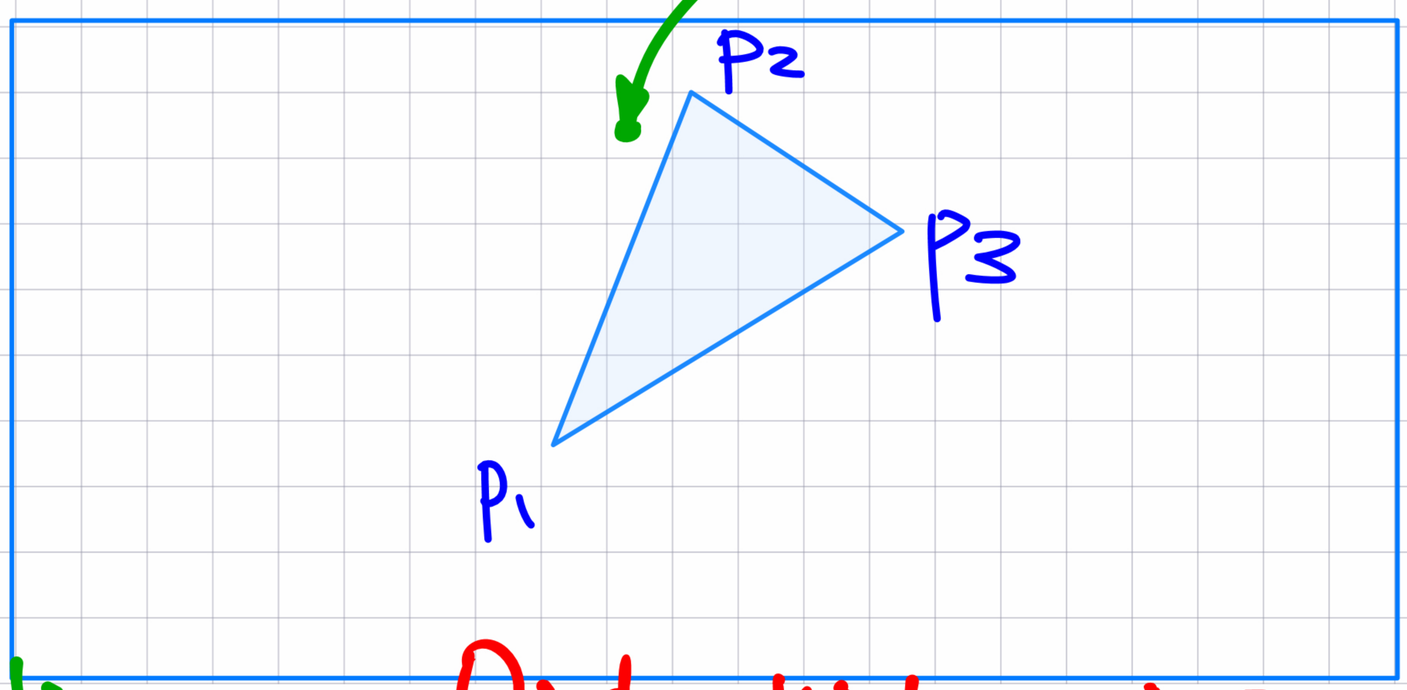
Object-order rendering stages:

1. Rasterization
2. Depth order
3. Shading: assign RGB values

inside/outside test

Rasterization:

Assign pixels to triangle



Display

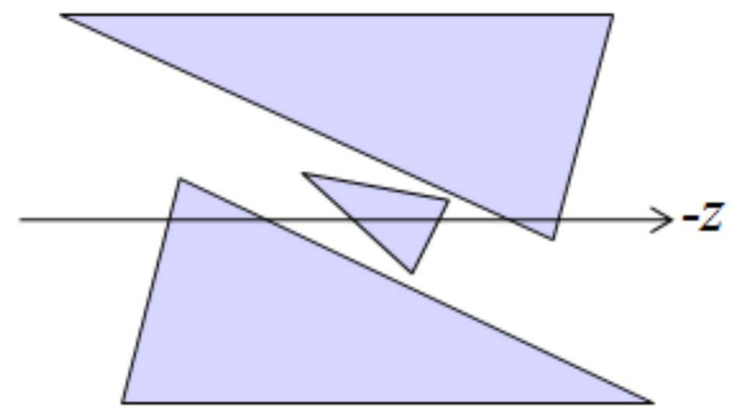
Problem:

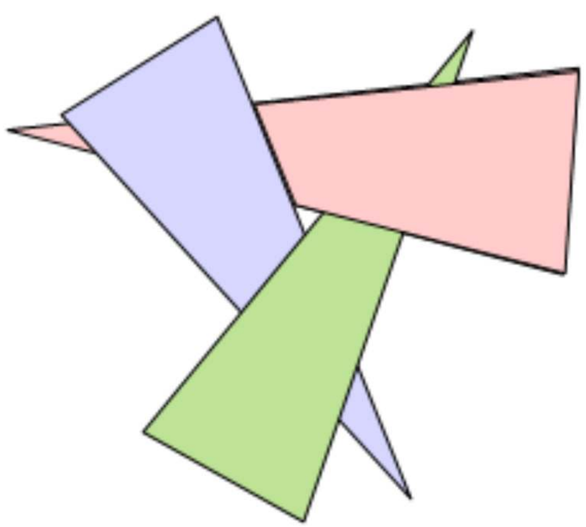
Aliasing, distortion of primitives

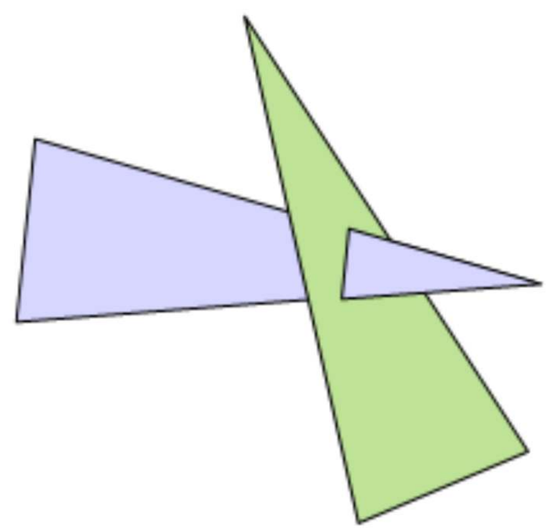
Q: does this happen in 1D rendering?

Painter's Algorithm

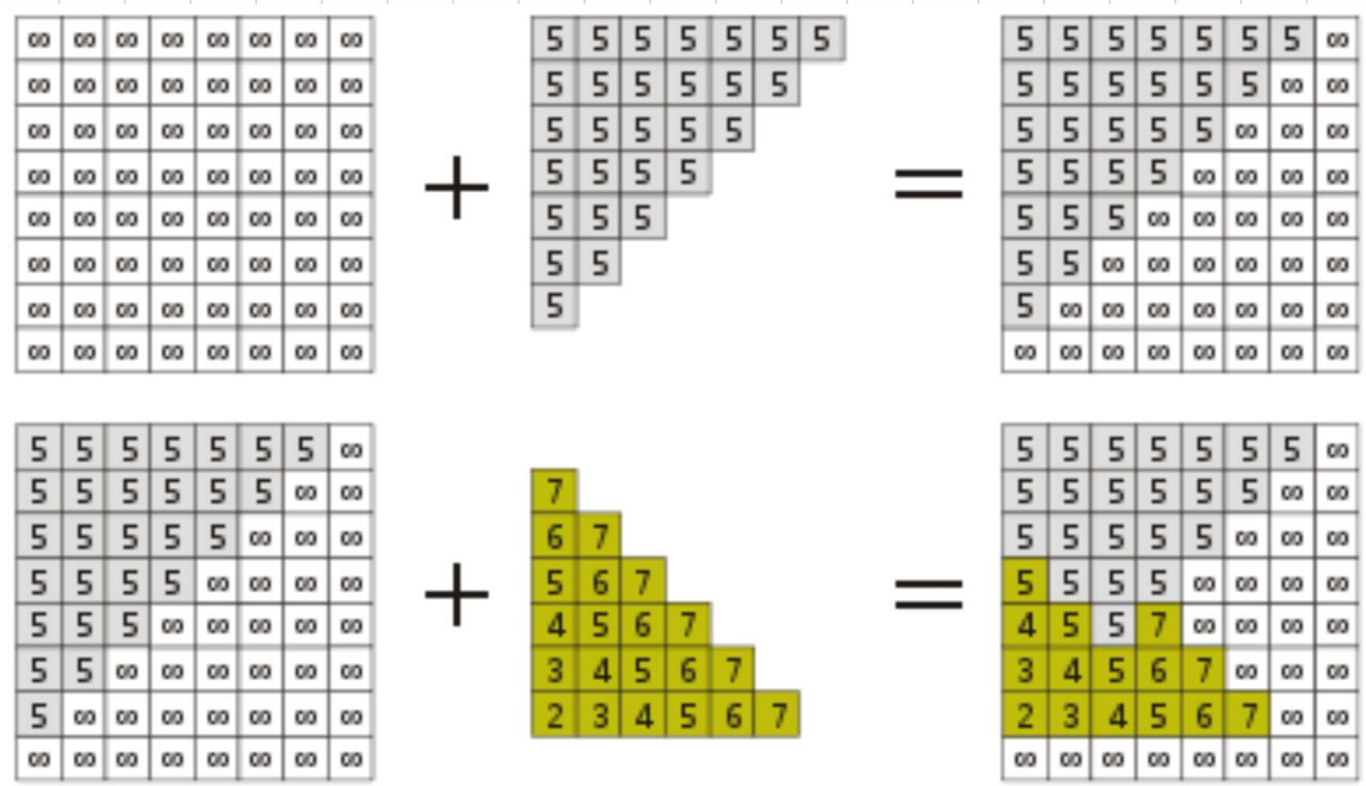
- Display polygons in back-to-front order
- Sort polygons by z-value
  - Which vertex?
  - $O(n \log n)$ (in VR? :( )
- Problems…







Z-Buffer



Key Observation: Each pixel displays color of only one triangle, ignores everything behind it

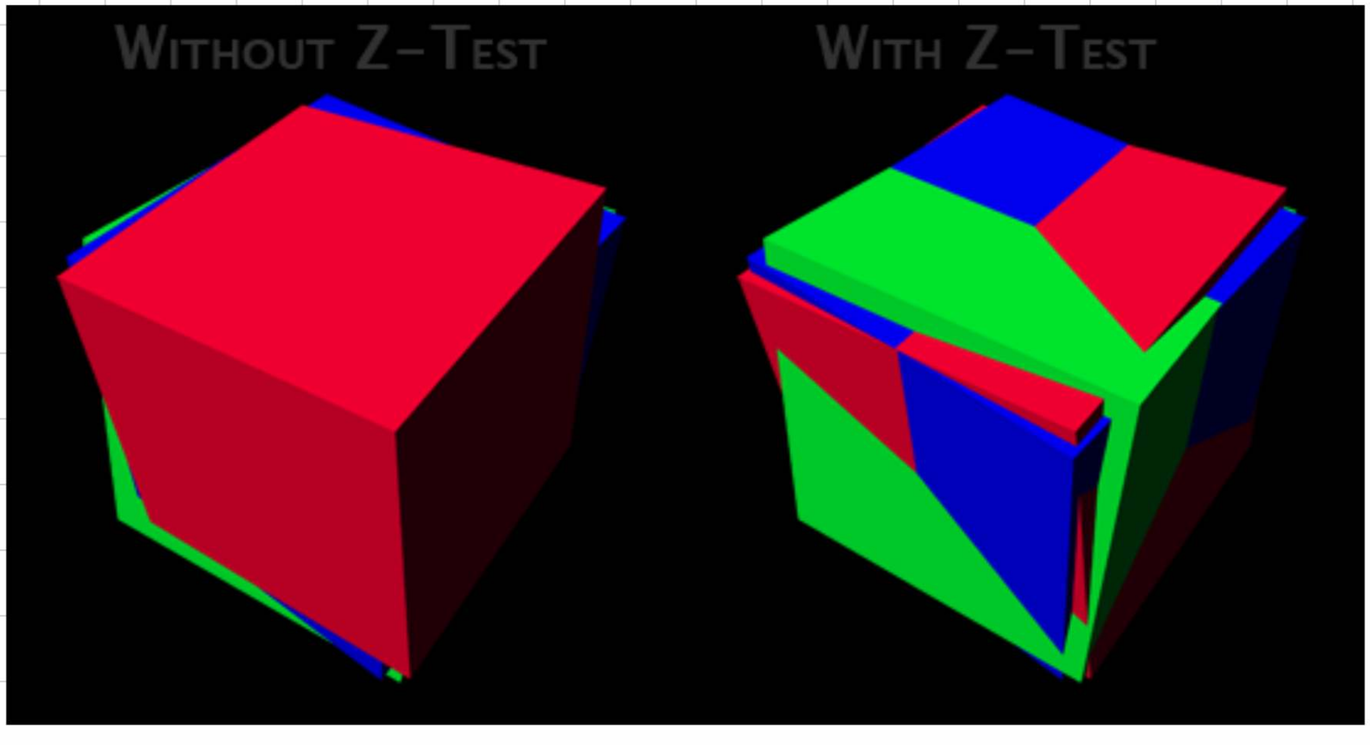
- Don't need to sort triangles, just find for each pixel the closest triangle
- Z-buffer: one fixed or floating point value per pixel
- Algorithm:

For each rasterized fragment (x,y)

If $z >$ zbuffer(x,y) then

framebuffer(x,y) = fragment color

zbuffer$(x,y) = z$



Clipping/Culling

Clipping:

- Remove triangles behind the eye

Q: Where is clipping plane in VR?

Culling:

- View Frustum: Remove triangles outside of viewing frustum.

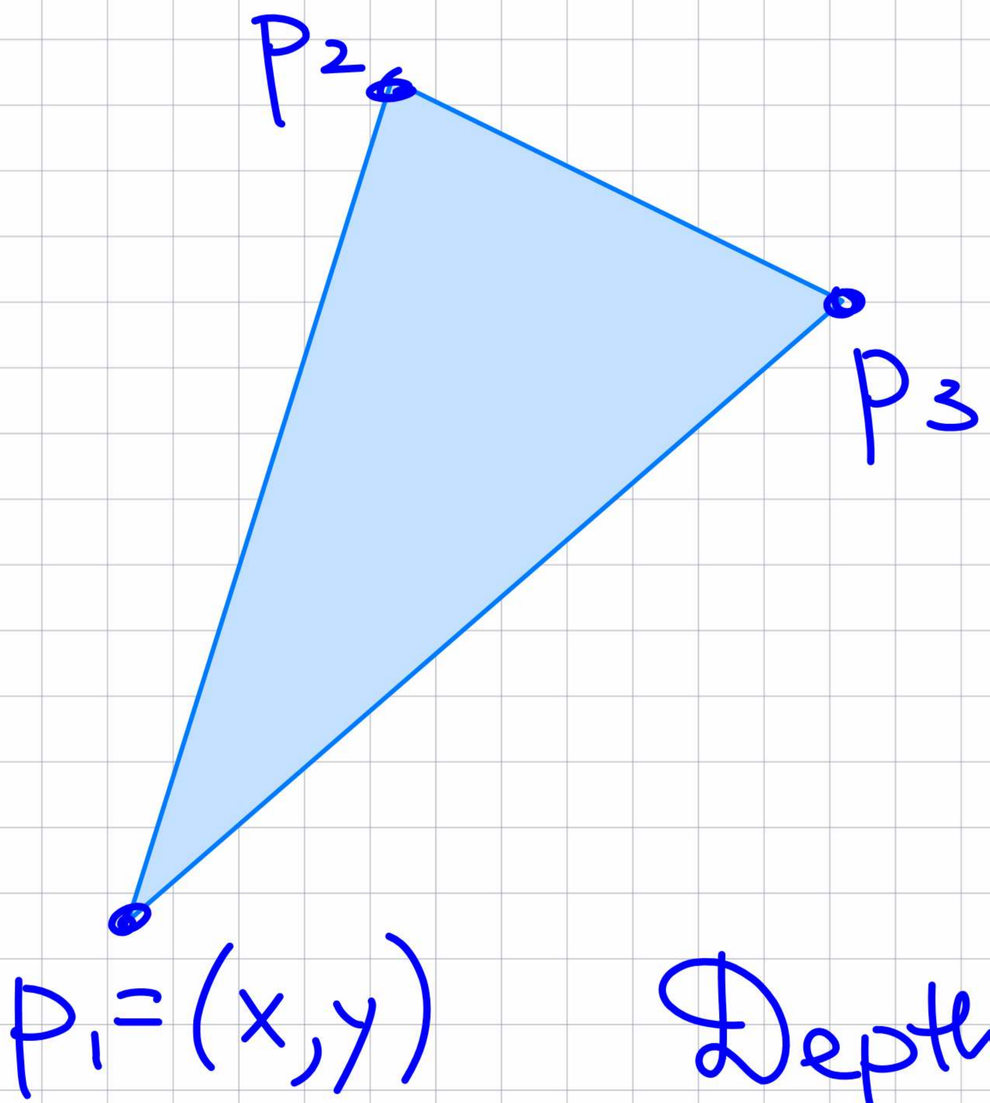
- Occlusion: Remove hidden triangles.

- Backface: Remove triangles on "back" of objects.

Interpolation and Barycentric Coordinates

Propagate color, RGB, other attributes efficiently:

Barycentric coordinates: $(\alpha_1, \alpha_2, \alpha_3)$



$$p = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3$$

$$\alpha_1, \alpha_2, \alpha_3 \in [0, 1]$$

$$\alpha_1 + \alpha_2 + \alpha_3 = 1$$

Ex1: $\alpha_1 = \alpha_2 = 0$

Ex2: $\alpha_1 = 0$

Depth calculation:

$$z = \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 z_3$$

RGB Mapping

$$R = \alpha_1 R_1 + \alpha_2 R_2 + \alpha_3 R_3$$
$$G = \alpha_1 G_1 + \alpha_2 G_2 + \alpha_3 G_3$$
$$B = \alpha_1 B_1 + \alpha_2 B_2 + \alpha_3 B_3$$

$$z = \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 z_3$$

Examples: