Real-Time Rendering: Lighting

Bauhaus-Universität Weimar

Deferred Shading

Real-Time Rendering: Lightning

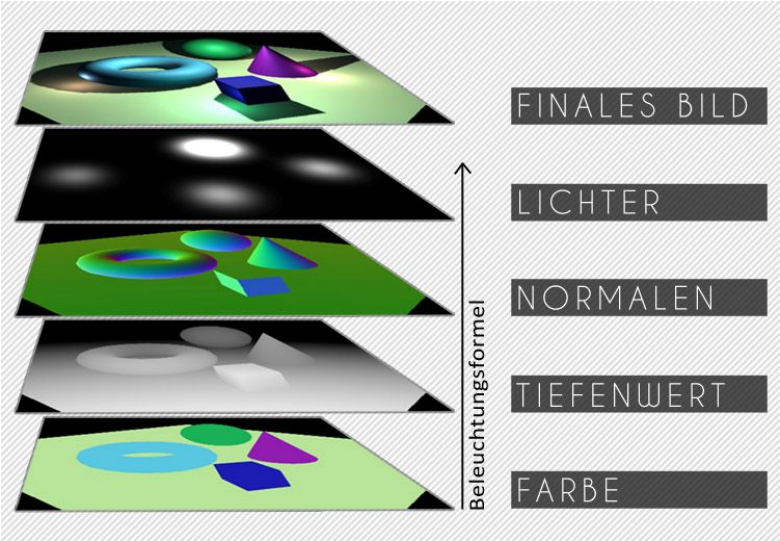
Rendering ‘The Hard Way’

➔ Vertex information:
- Color
- Normal
- Distance
- ….

➔ Calculate a vertex’ color with these,
➔ Decide what is visible,
➔ Render to screen

Rendering with Deferred Shading

- ➔ Similar to offscreen-rendering
- ➔ First pass stores available information in separate textures/buffers
- ➔ Second pass calculates illumination of each fragment with these textures



NPR vs PBR

Real-Time Rendering: Lightning

material layering

Physically-Based-Rendering

material parameters taken from real-world objects

https://www.marmoset.co/posts/physically-based-rendering-and-you-can-too/

http://www.okami-game.com/

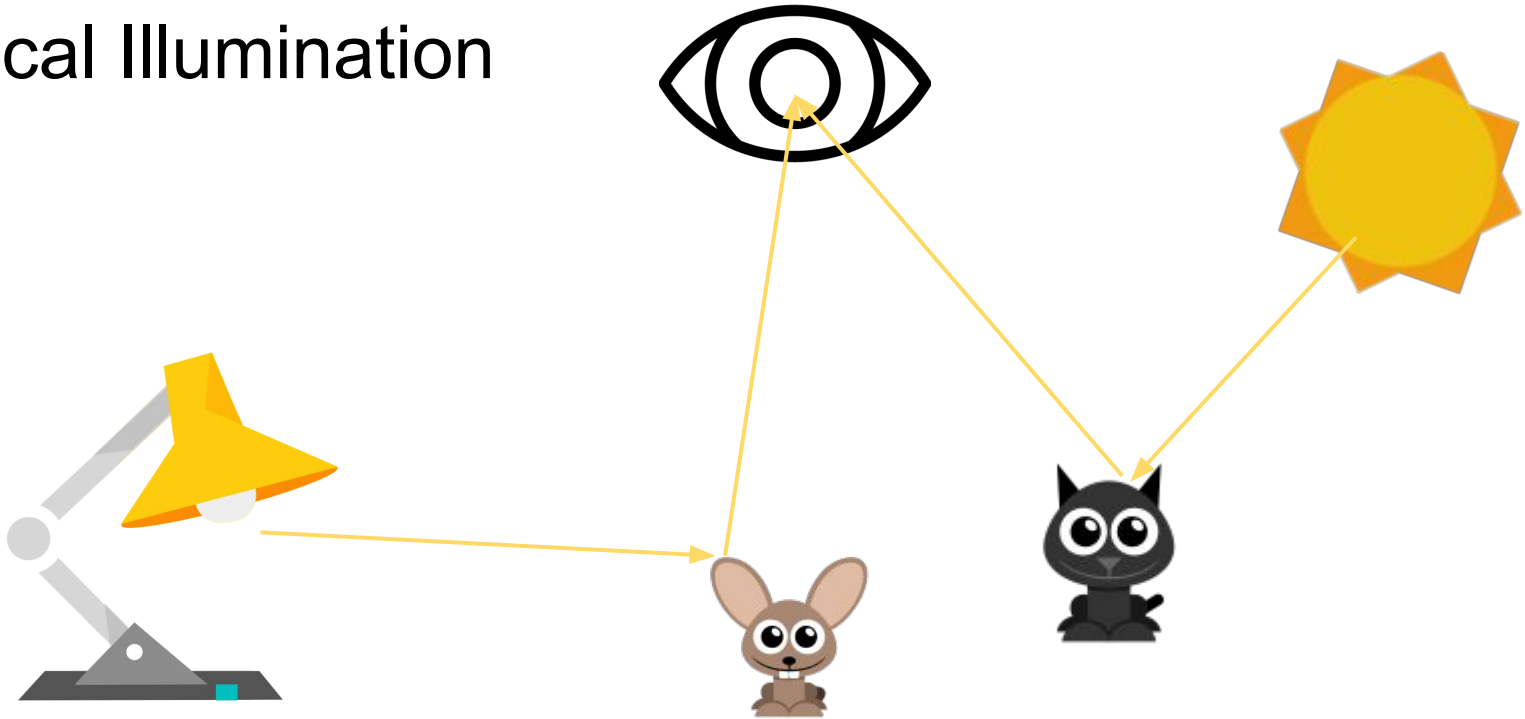
Non-Photorealistic-Rendering

e.g. Comic-Style, CAD-Software...

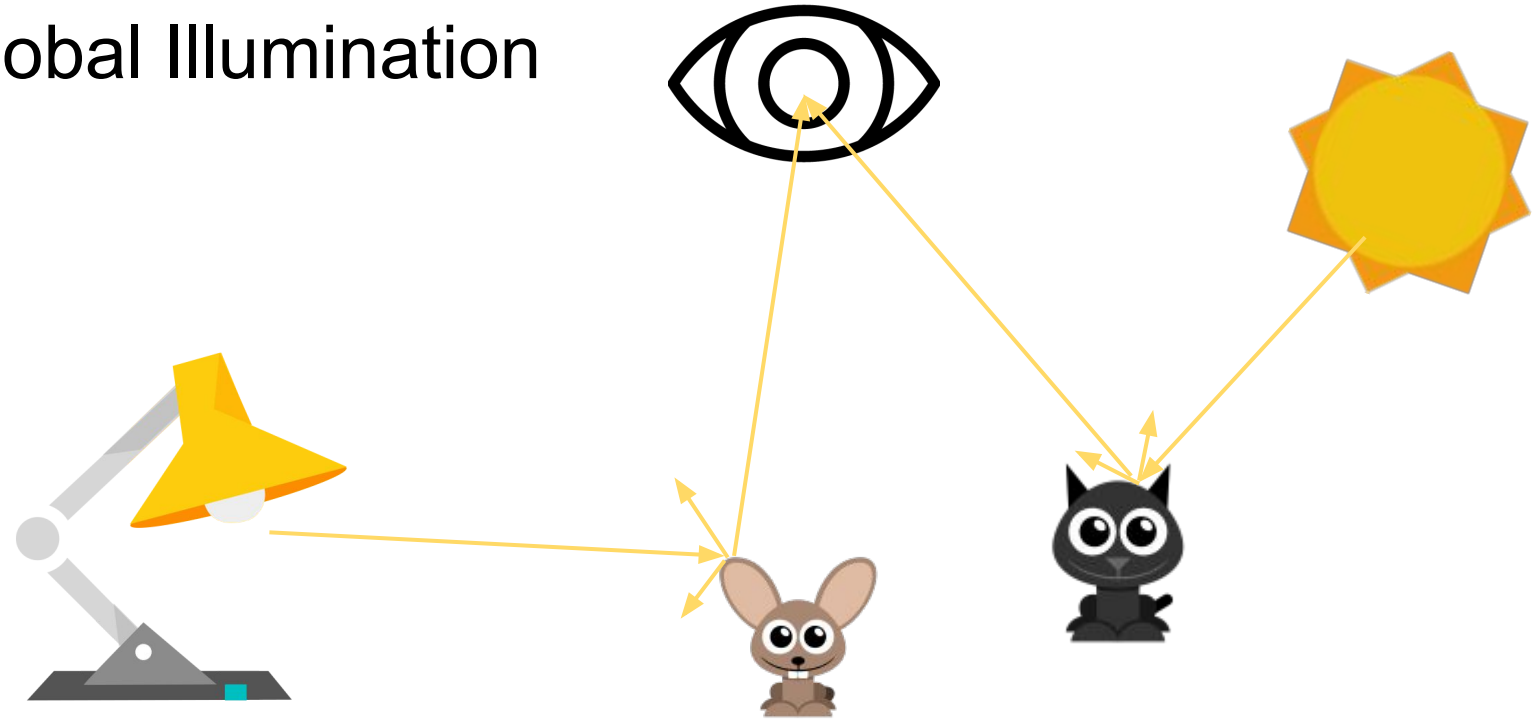
Global Illumination

Real-Time Rendering: Lightning

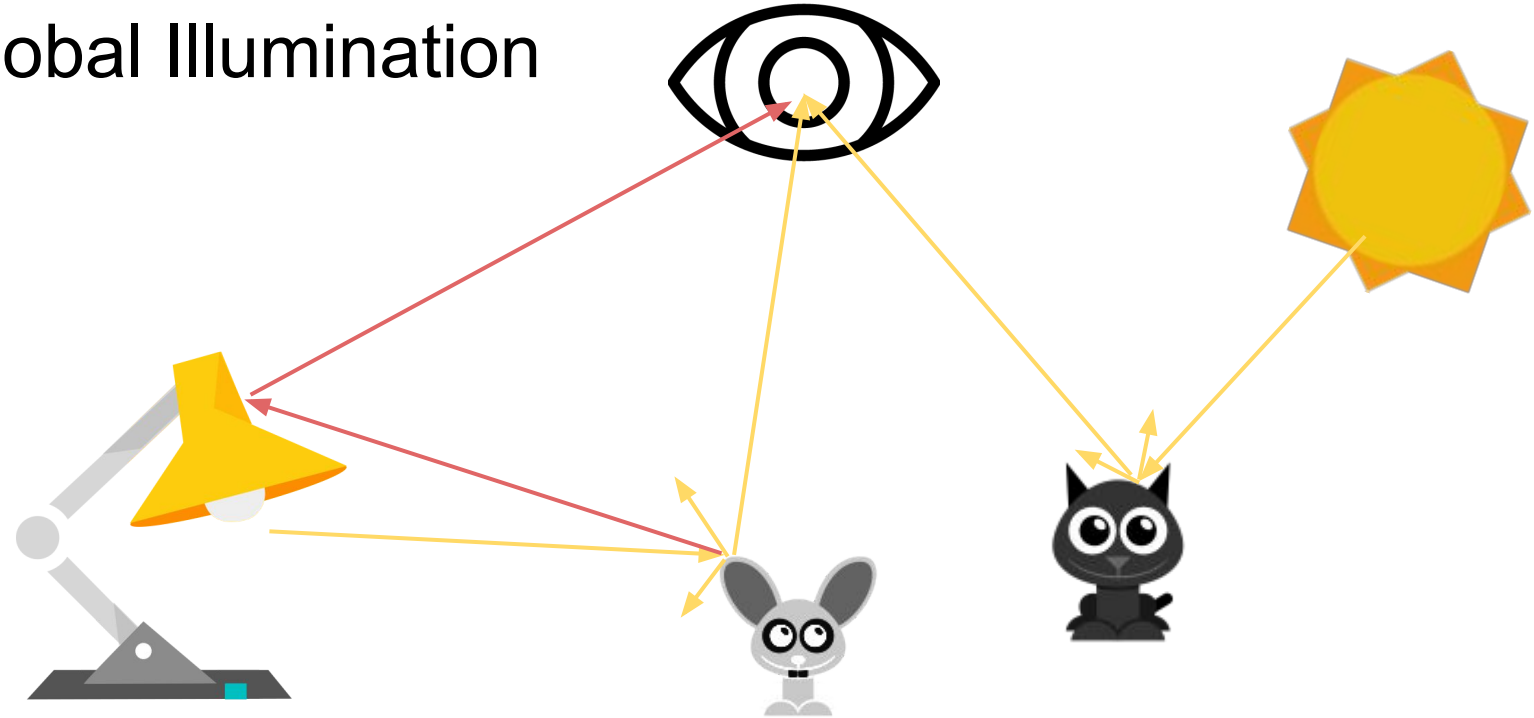
Local Illumination



Global Illumination



Global Illumination



(highly reflective bunny)

Global Illumination Equation

$$L(x, x') = g(x, x') \cdot \left(L_e(x, x') + \int_S b(x, x', x'') L(x', x'') \mathrm{d}x'' \right)$$

➔ Light from x’ to x

Global Illumination Equation

$$L(x, x') = g(x, x') \cdot \left(L_e(x, x') + \int_S b(x, x', x'') L(x', x'') \mathrm{d}x'' \right)$$

- ➔ Light from x' to x
- ➔ Distance / occlusion btw x, x'

Global Illumination Equation

$$L(x, x') = g(x, x') \cdot \left(L_e(x, x') + \int_S b(x, x', x'') L(x', x'') \mathrm{d}x'' \right)$$

- ➔ Light from x' to x
- ➔ Distance / occlusion btw x, x'
- ➔ Light emitted from x' to x

Global Illumination Equation

$$L(x, x') = g(x, x') \cdot \left(L_e(x, x') + \int_S b(x, x', x'') L(x', x'') \mathrm{d}x'' \right)$$

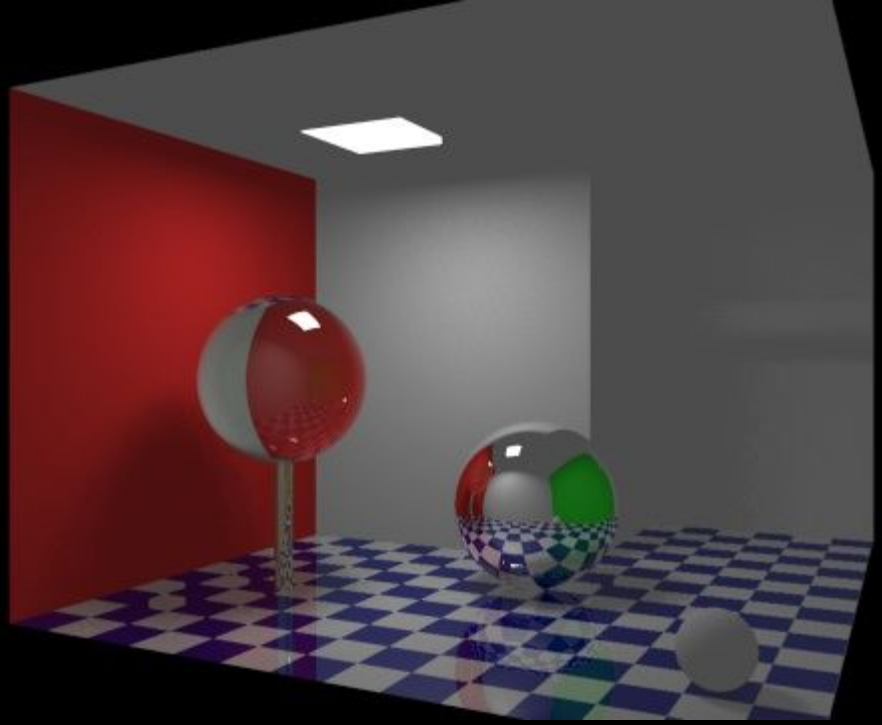
- ➔ Light from x' to x
- ➔ Distance / occlusion btw x, x'
- ➔ Light emitted from x' to x
- ➔ Light from x'' redirected to x via x' (x'': all other objects)

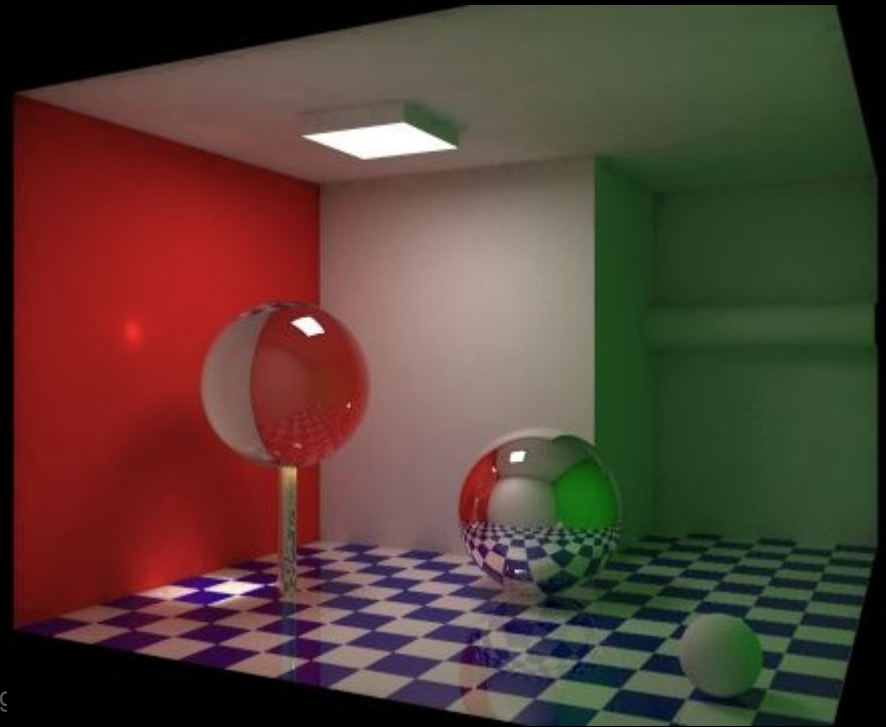
Global Illumination Equation

$$L(x, x') = g(x, x') \cdot \left(L_e(x, x') + \int_S b(x, x', x'') L(x', x'') \mathrm{d}x'' \right)$$

➔ Exact solution too complex

➔ Methods such as ray tracing approximate a solution





Sources:

ani_ch7 (Physically based Animation)
rtr_ch17 (collision_detection)
3dm_ch12 (mechanics_2)
“Approaches to destruction effects in real-time computer graphics” -)R. Hettich)
Real-time Cloth Rendering with Fiber-level Detail - (Kui Wu and Cem Yuksel)

khanacademy.org/partner-content/pixar
opengl-tutorial.org/assets/images/tuto-particules/
what-when-how.com/advanced-methods-in-computer-graphics/

Iconarchive.com
onlinewebfonts.com
giphy.com