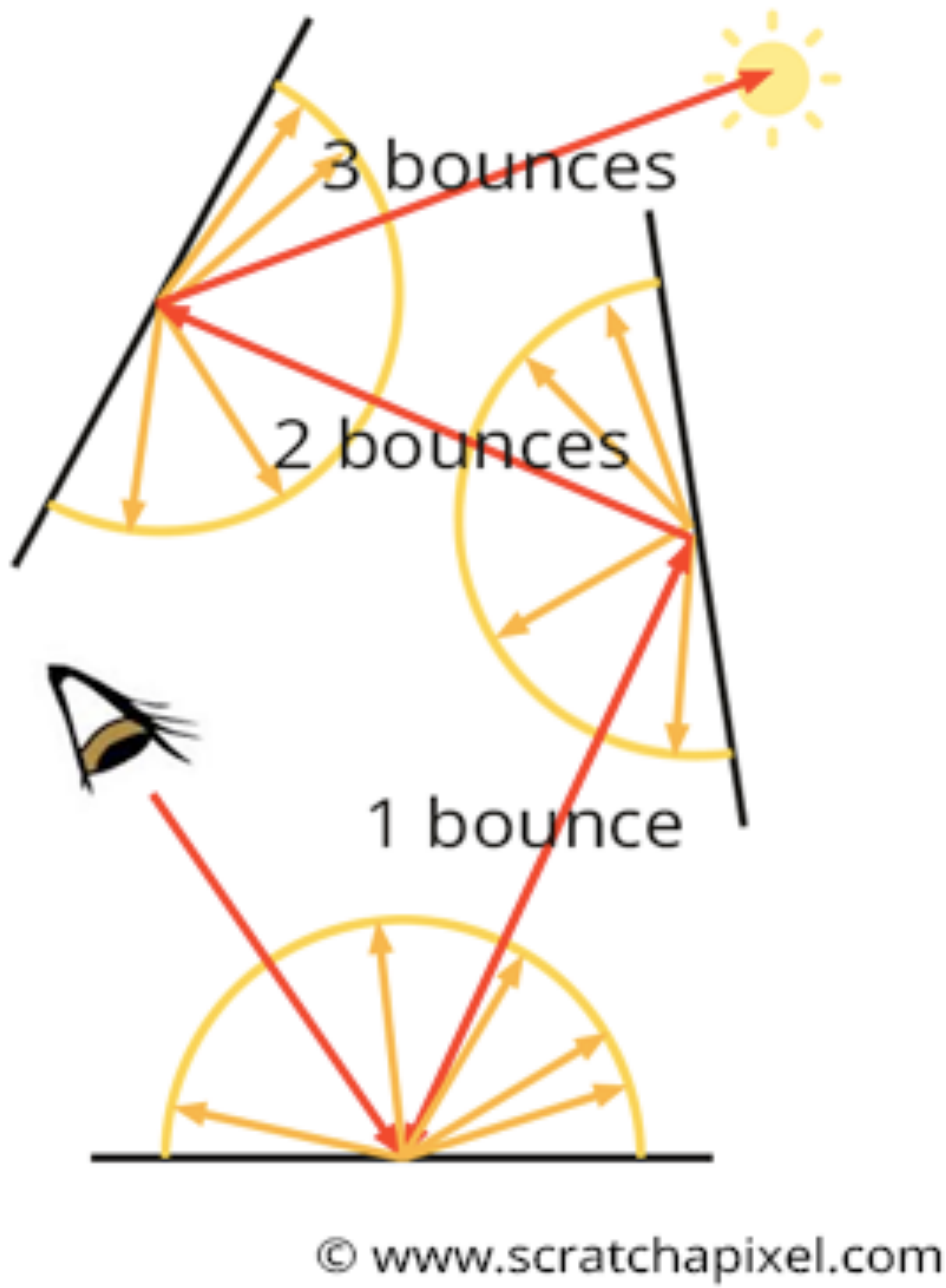
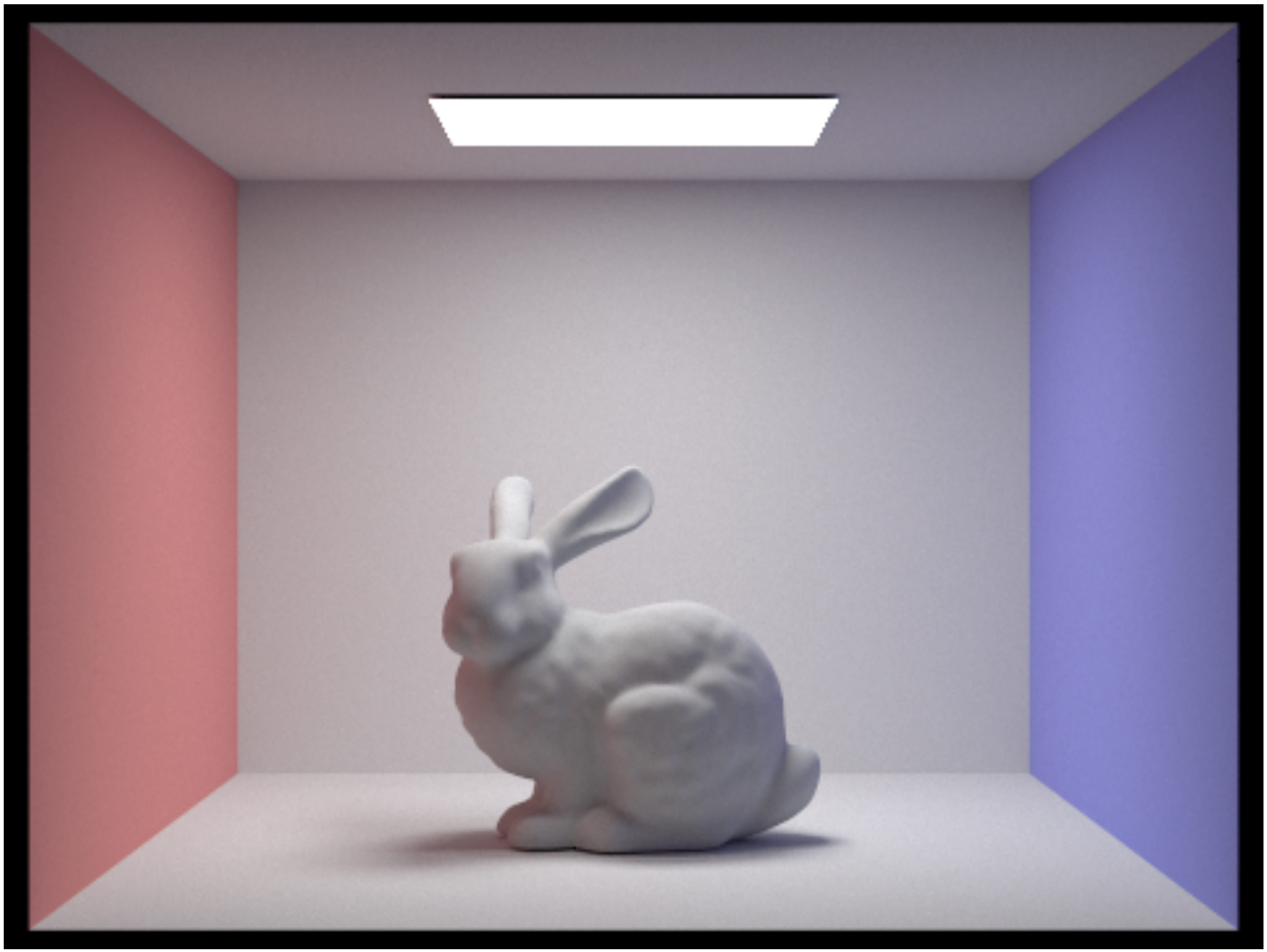
Advanced Rendering Techniques

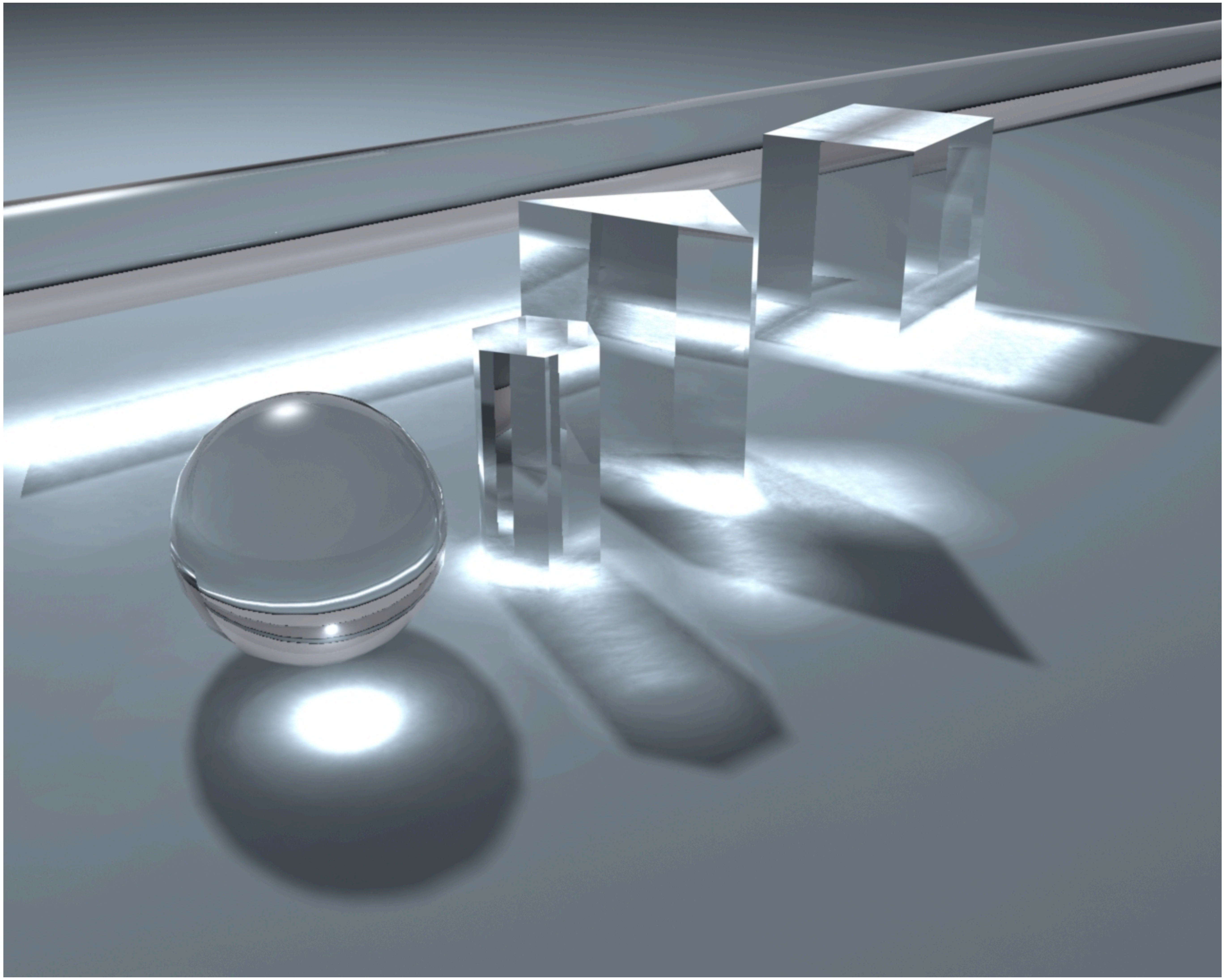
Computer Graphics and Imaging
UC Berkeley CS184

Path tracing

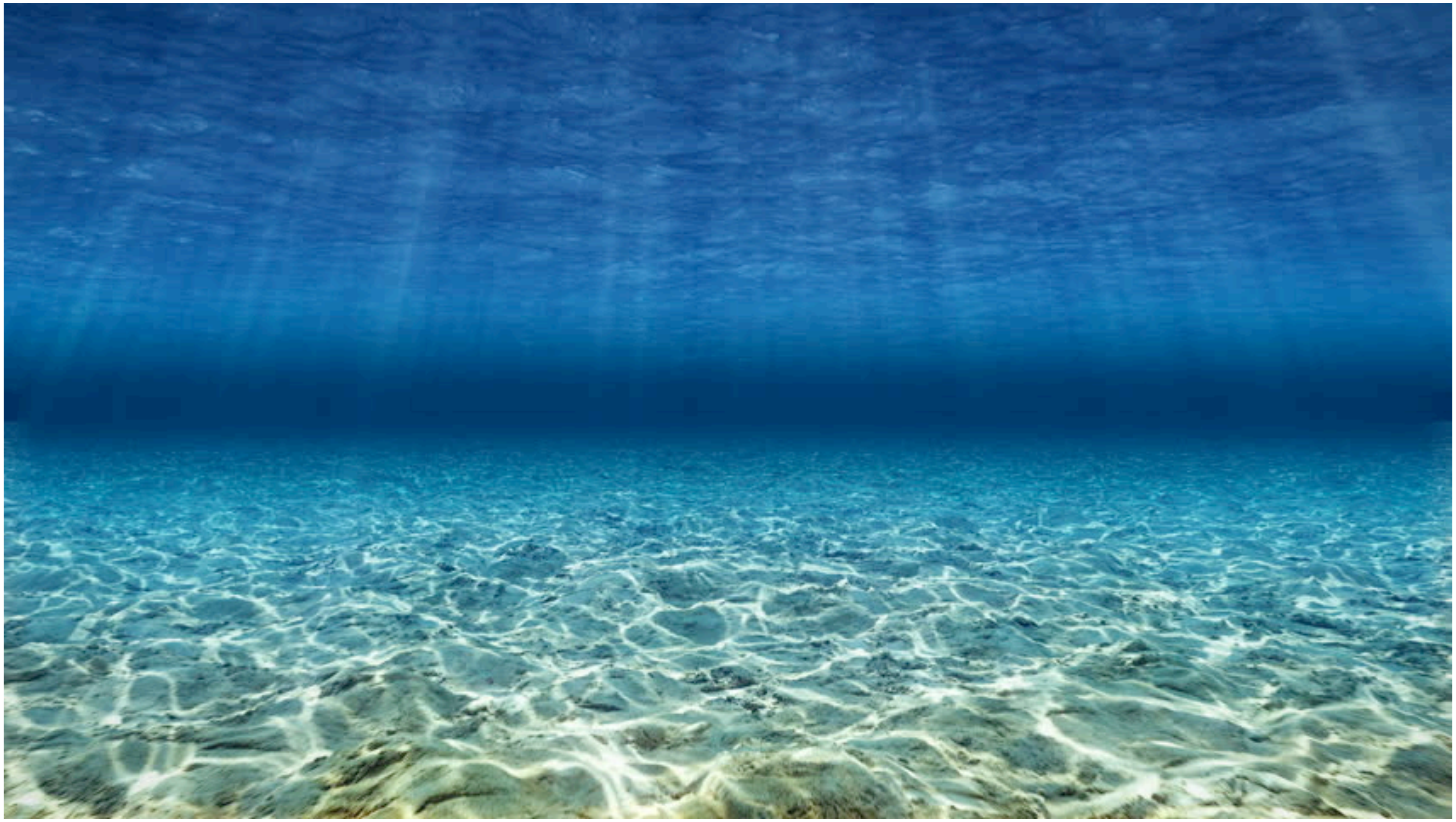




Caustics



Caustics



Complex Visibility

Volume Scattering

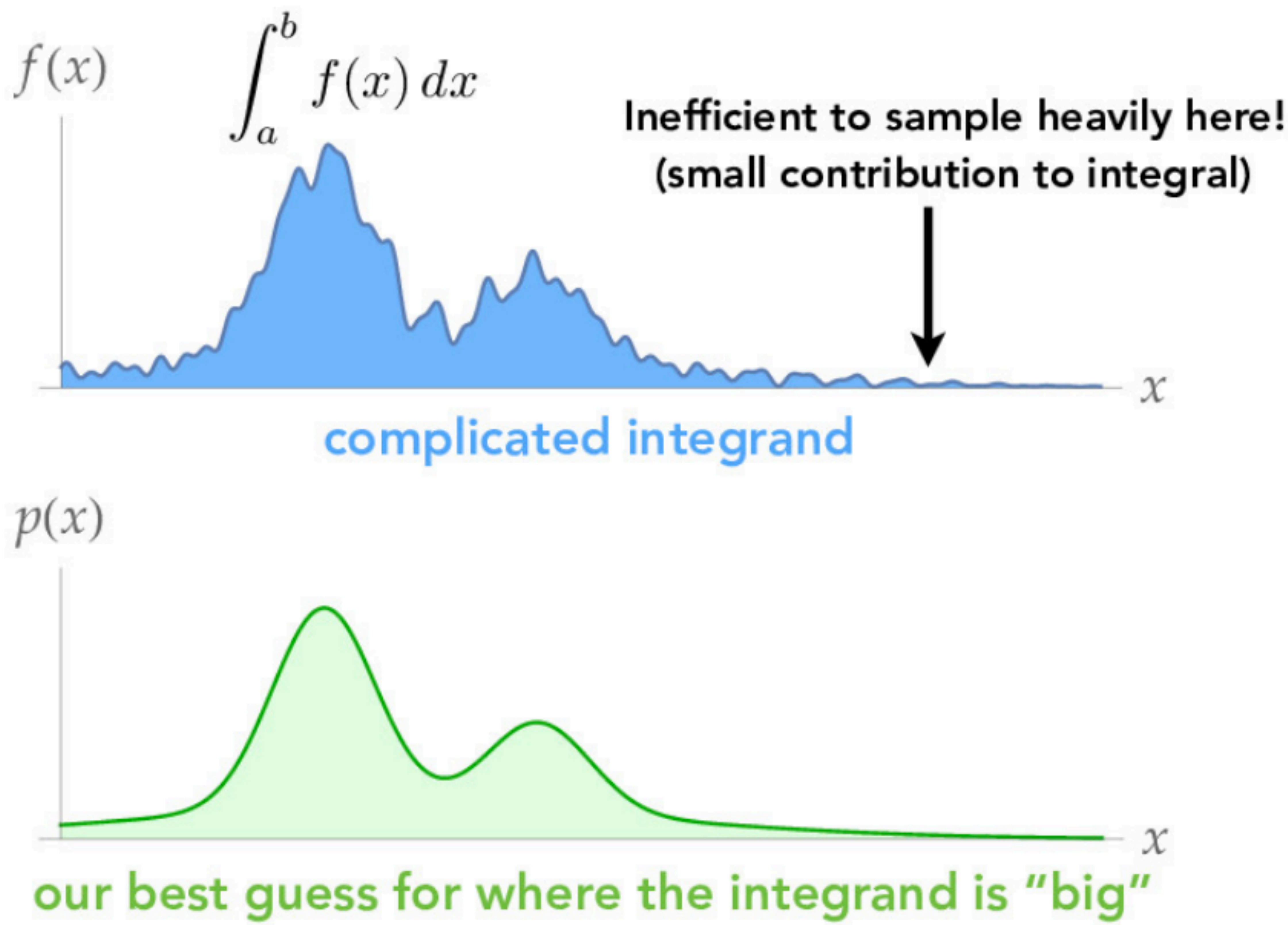
God Rays

Rich visual effects blow up render time

- Caustics
    - Liquids, Glass
- Complex visibility
    - Lights hidden in objects, lights blocked by walls, etc…

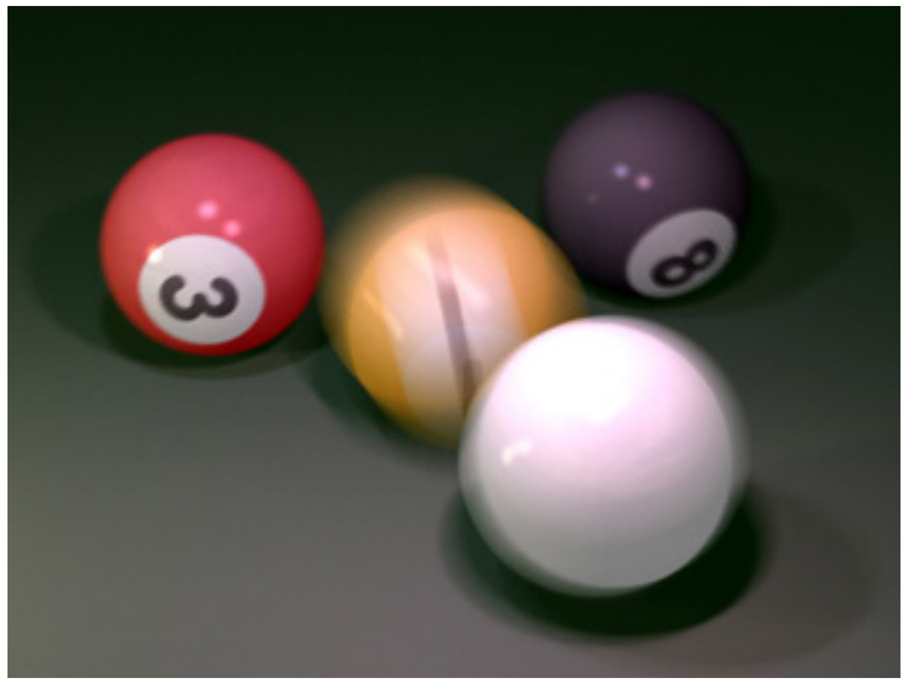
- Lighting from specular bounces
    - Complex metallic objects
- Translucent materials
    - Backlit cloth/paper, Materials with Subsurface
- Volume scattering
    - Fog, God Rays, Dust

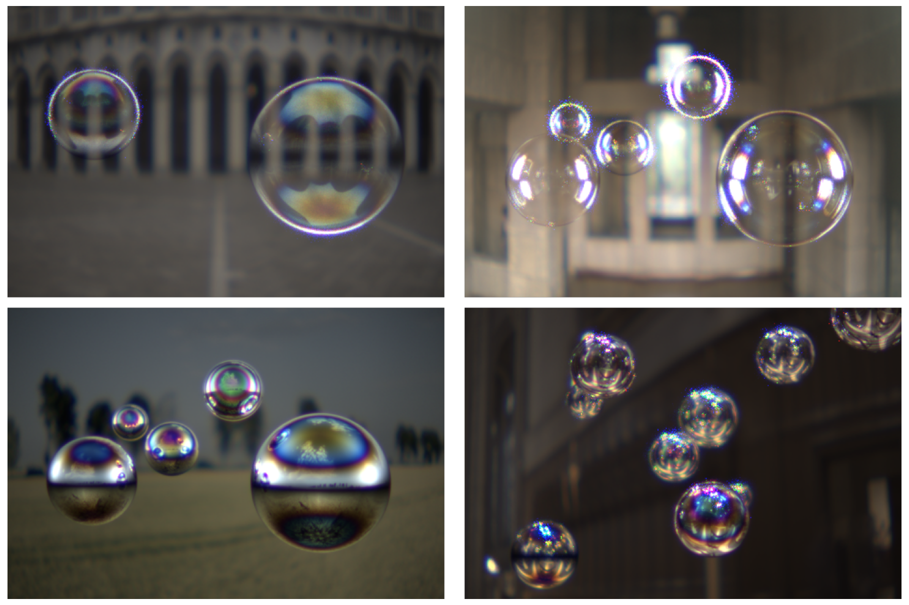
Path tracing takes a long time to converge!



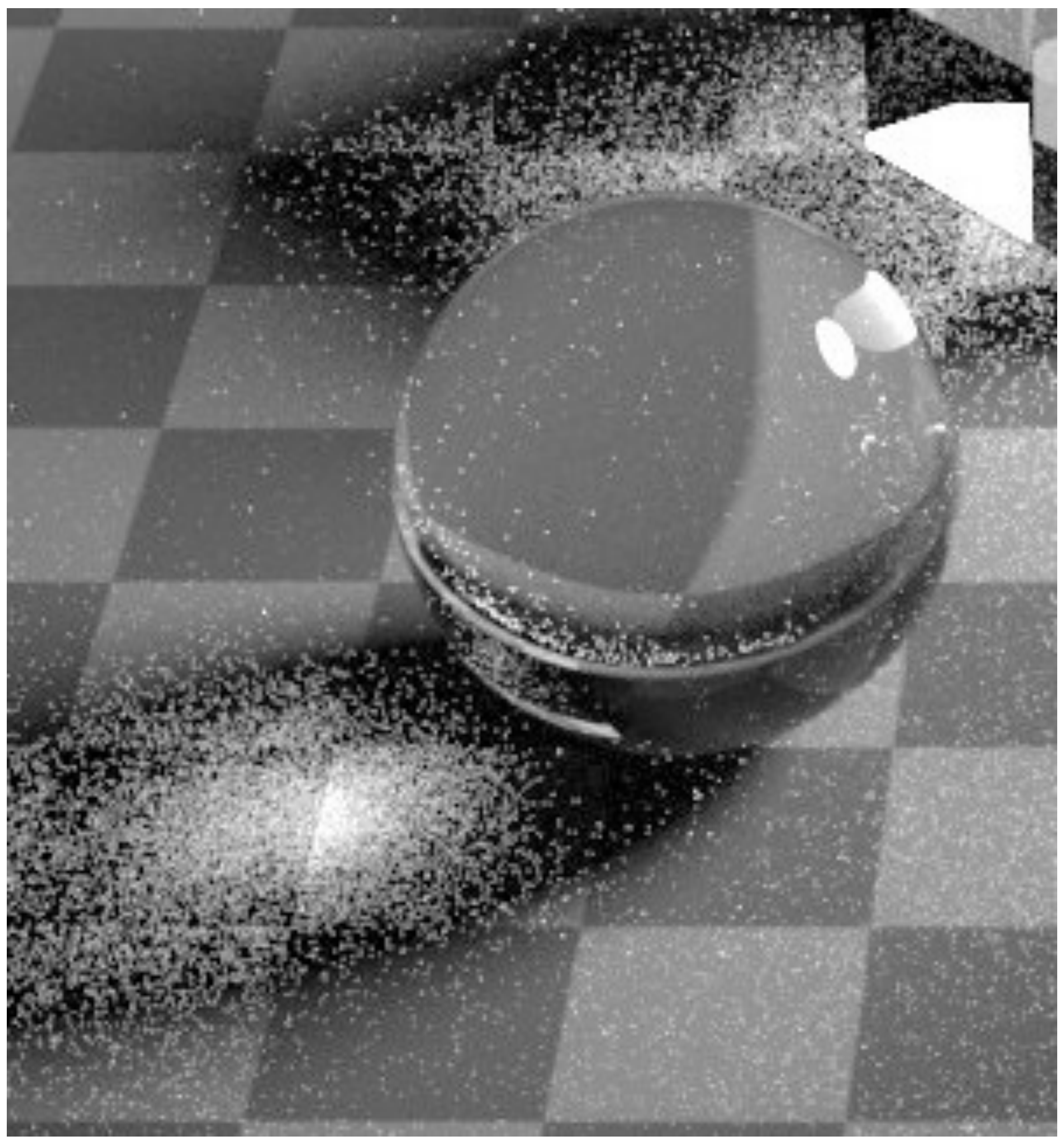
Everything is sampling

- Across the pixel (we did this in 3-1)
    - Reduces aliasing
- Across the camera aperture (we will do this in 3-2)
    - Creates DoF effects
- Across time
    - Motion blur
- Across wavelengths
    - Creates spectral effects
- Across solid angle of light sources (we did this in 3-1)
    - Penumbras & soft shadows
- ...and many more!!

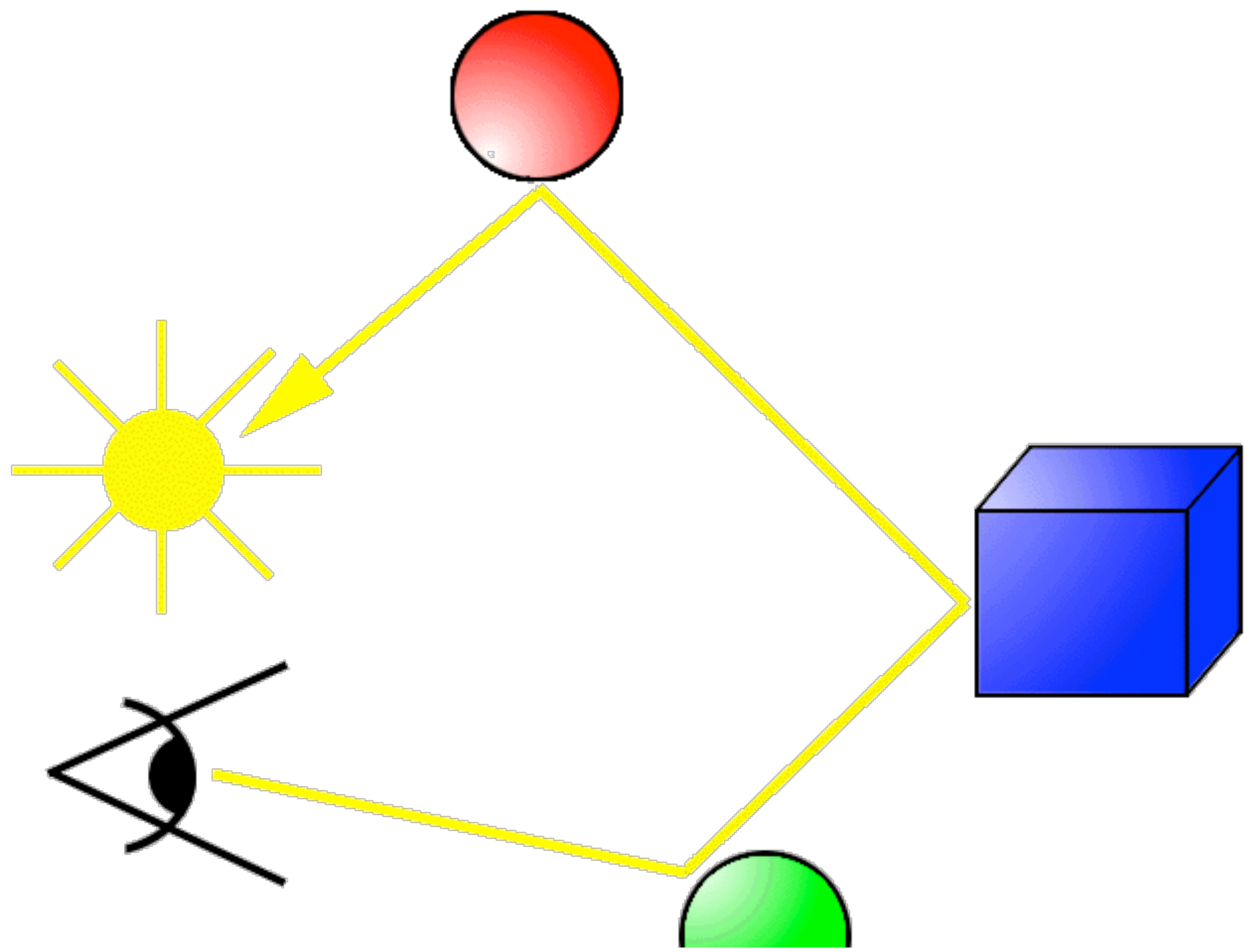




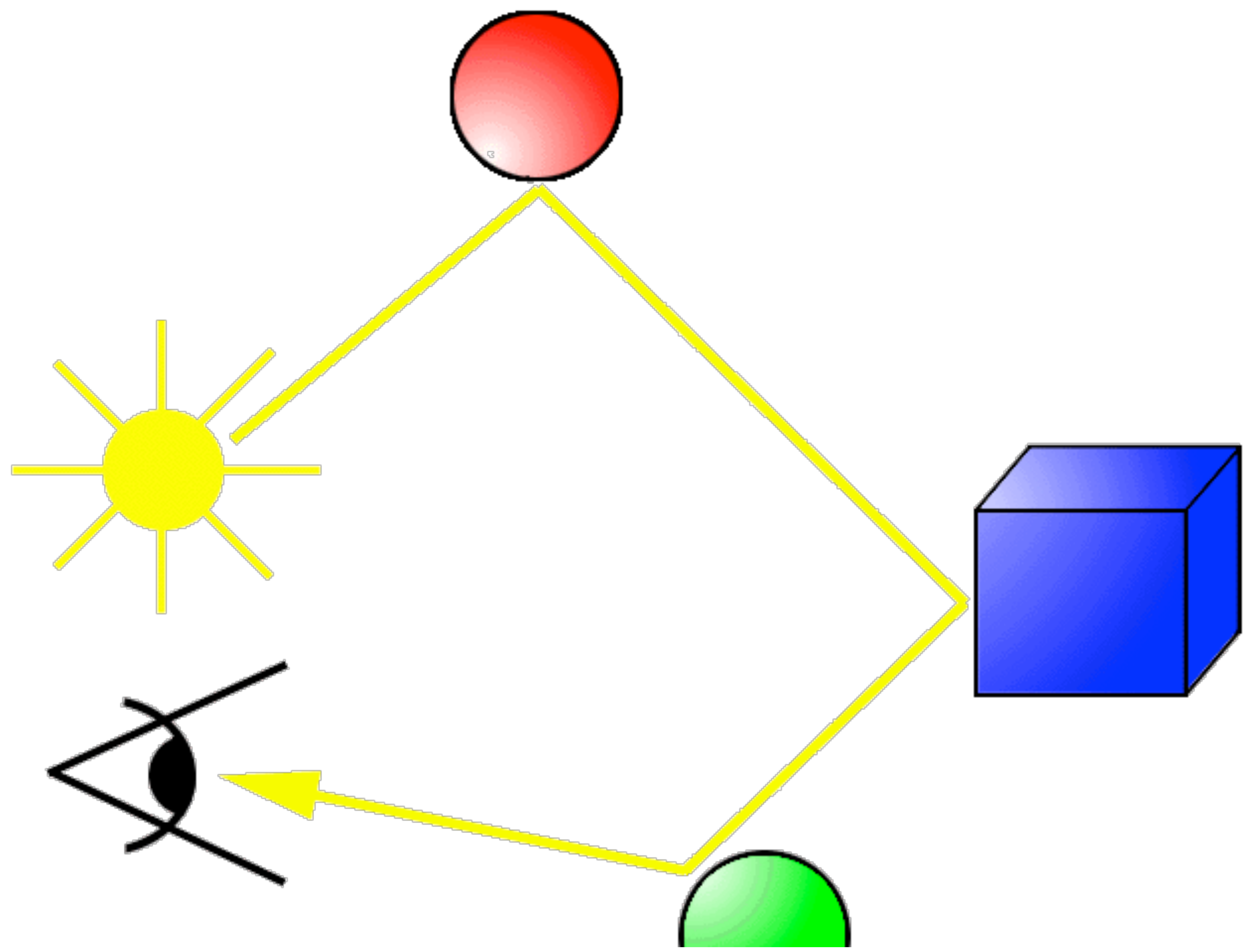
Path tracing failures



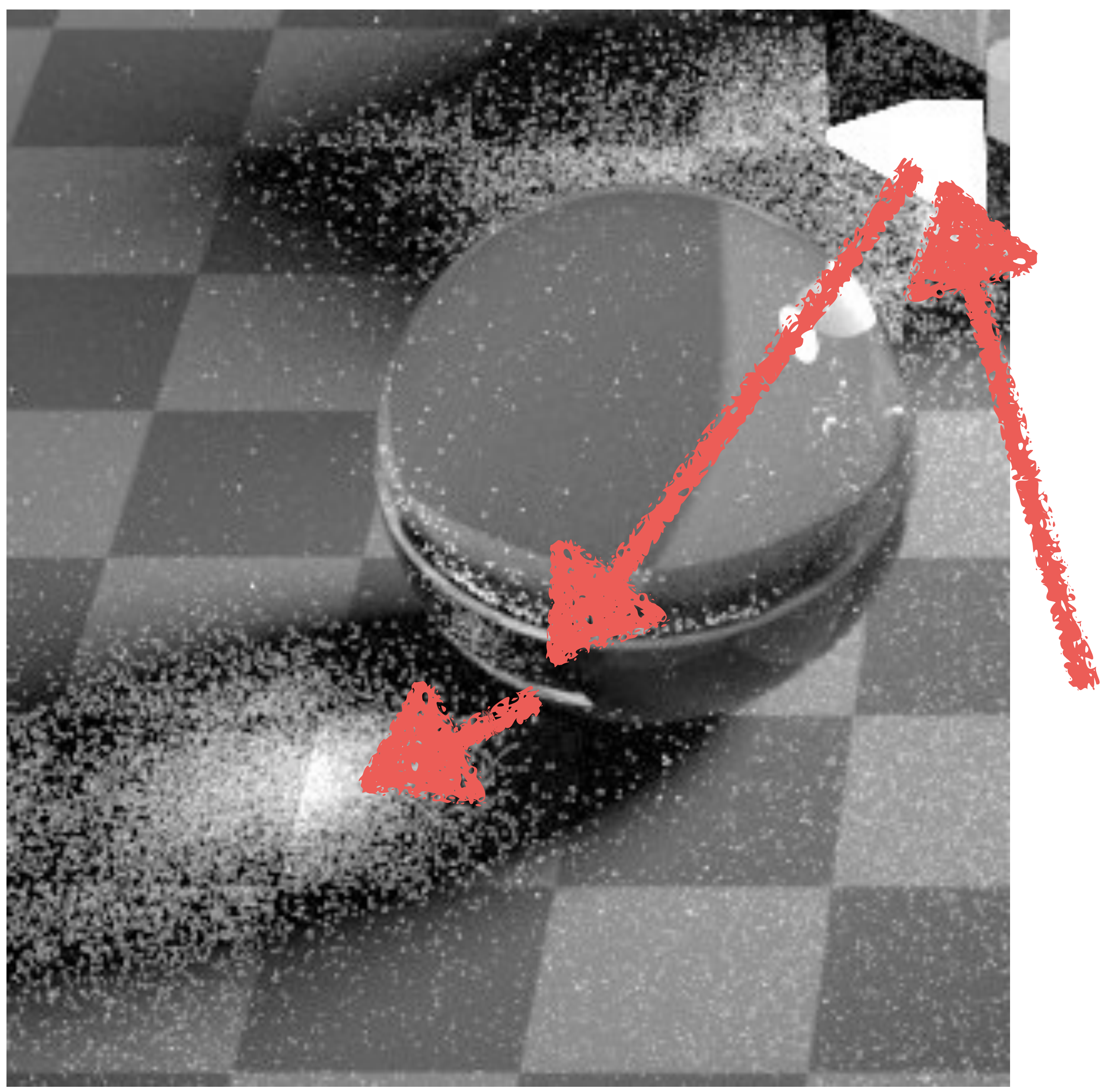
What we've been doing

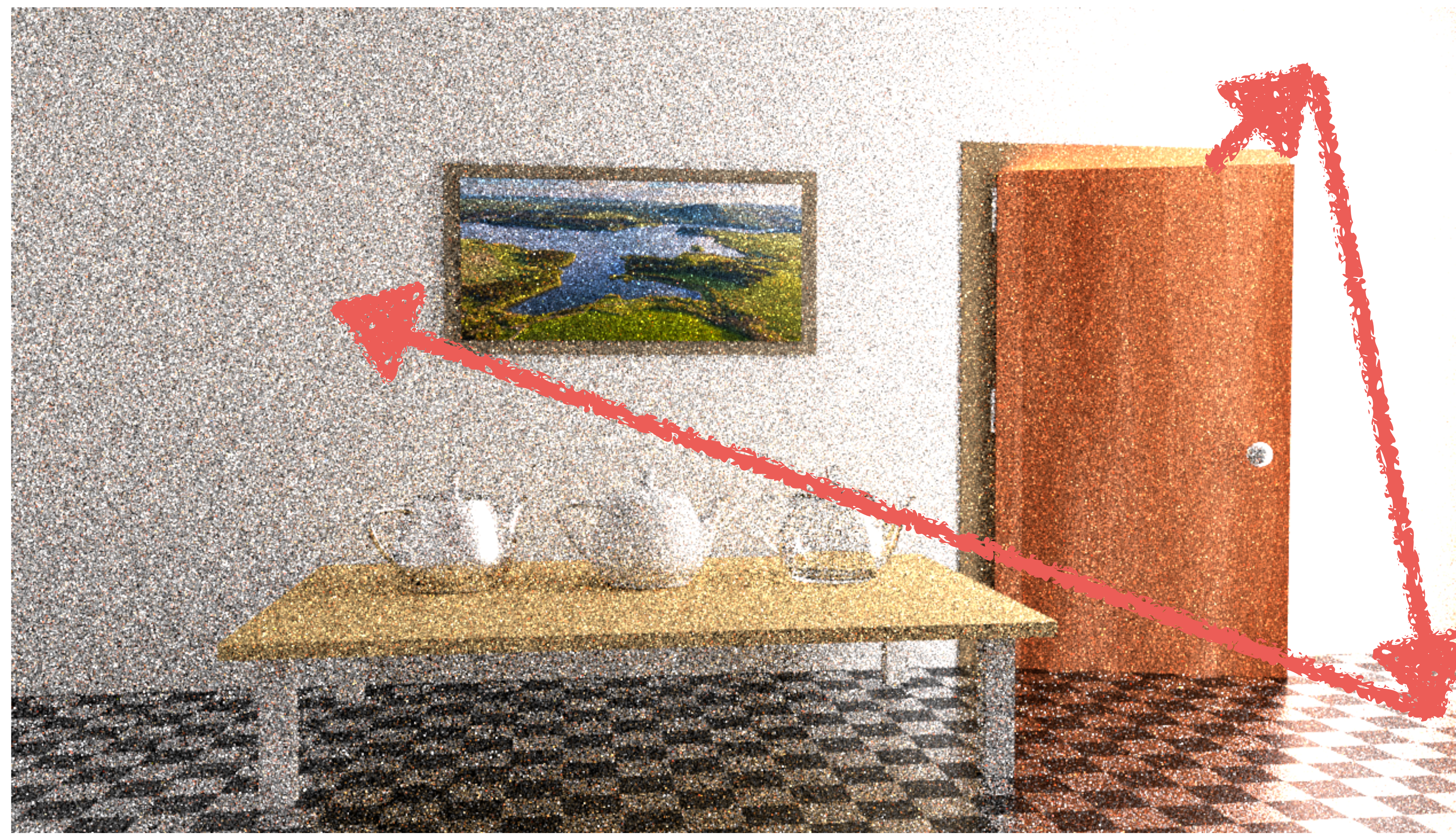


How the world actually works

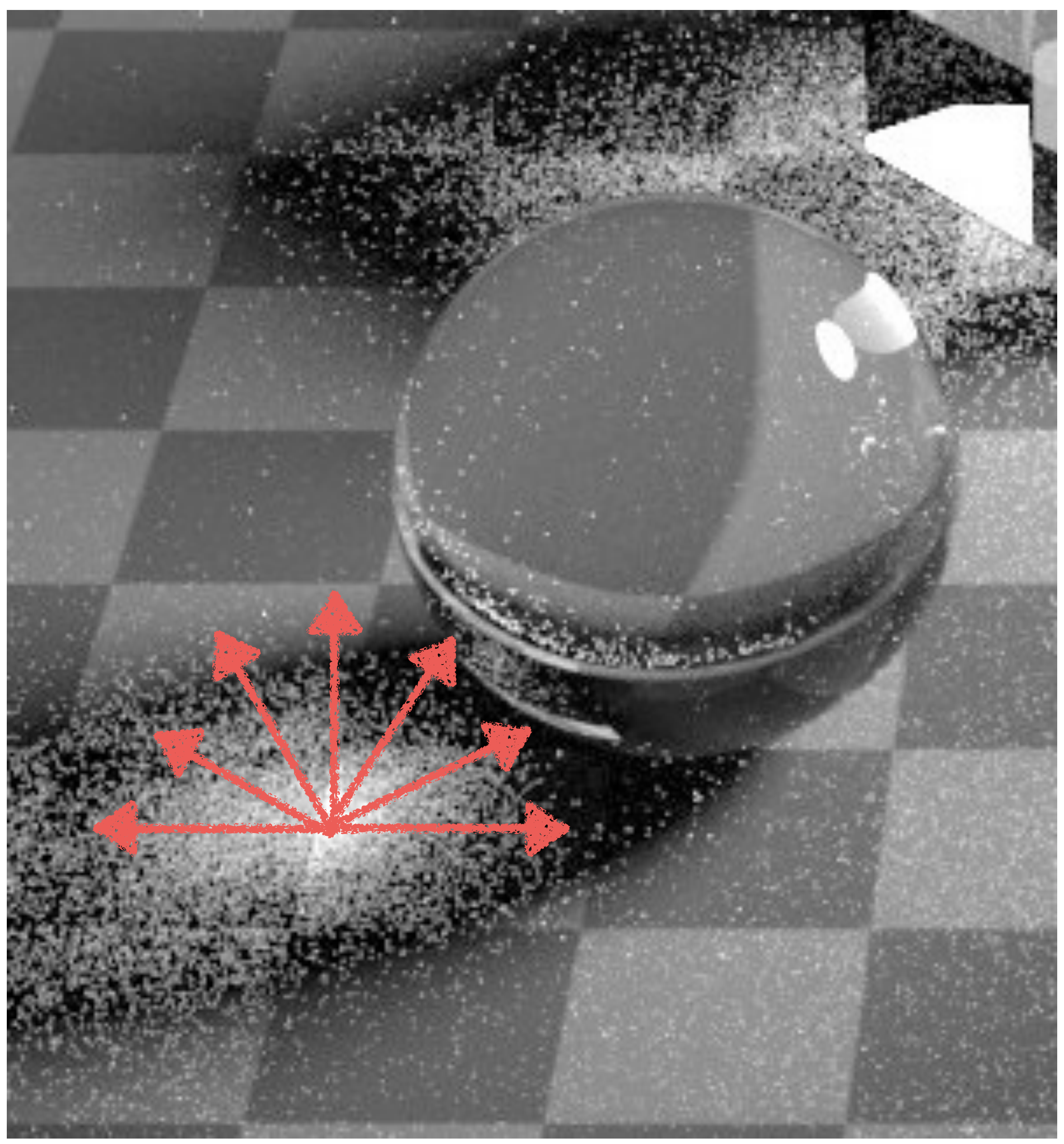


Path tracing failures



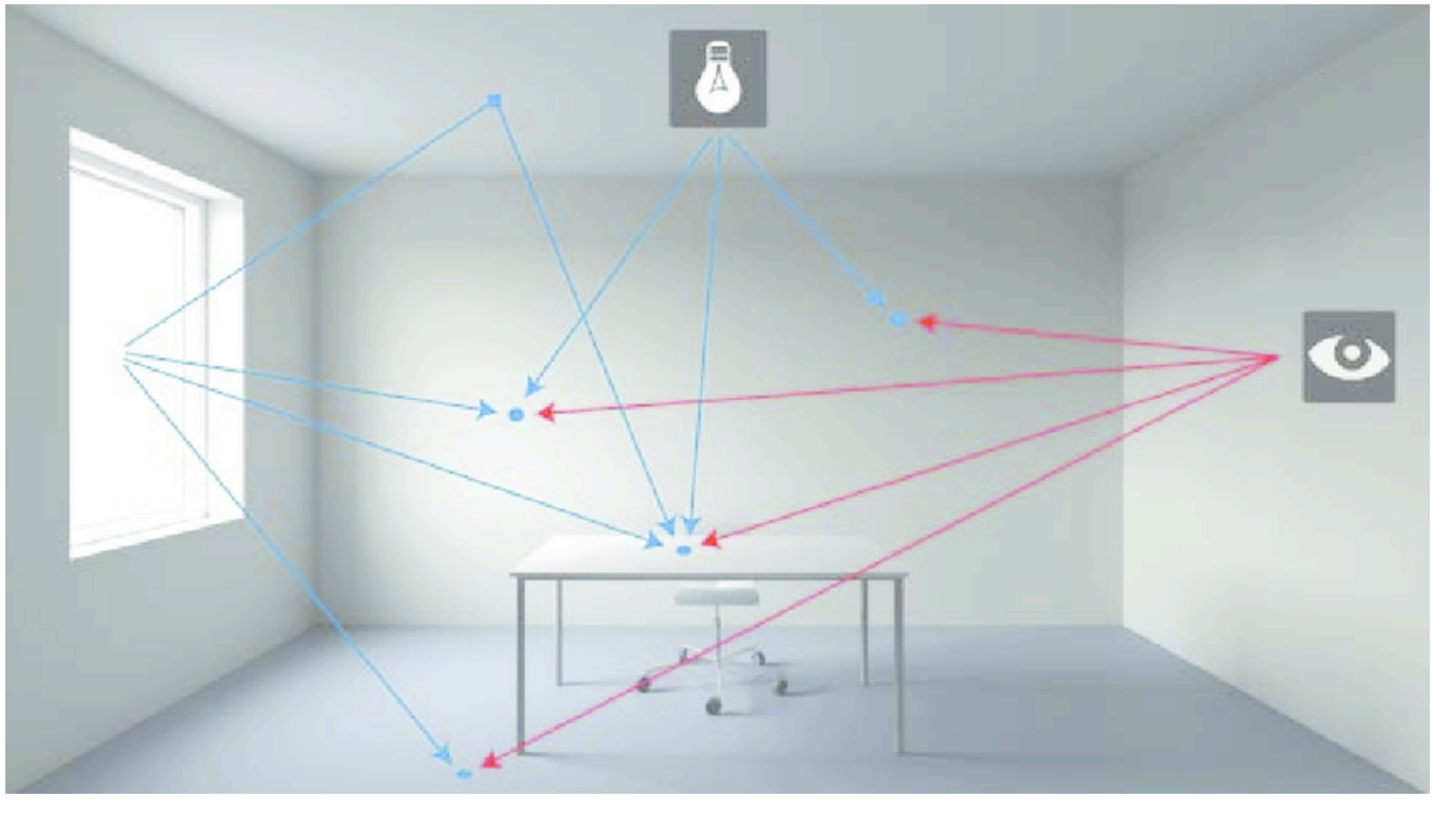


Path tracing failures



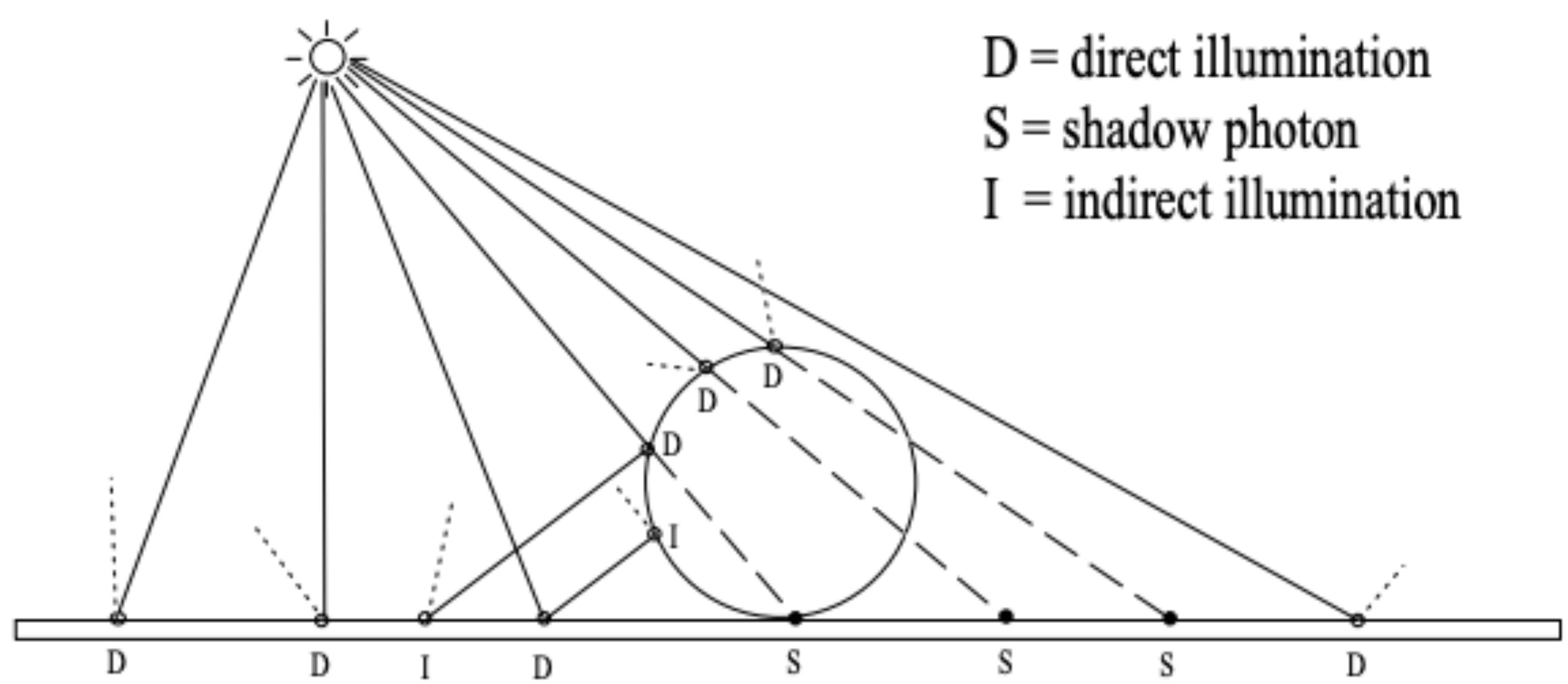
Photon Mapping

- Two-pass technique
    - First, trace 'photons' and store emission in 2 maps
        - Store a separate map for caustics specifically
    - Second, render
- "Global Illumination Using Photon Maps", Henrik Wann Jensen



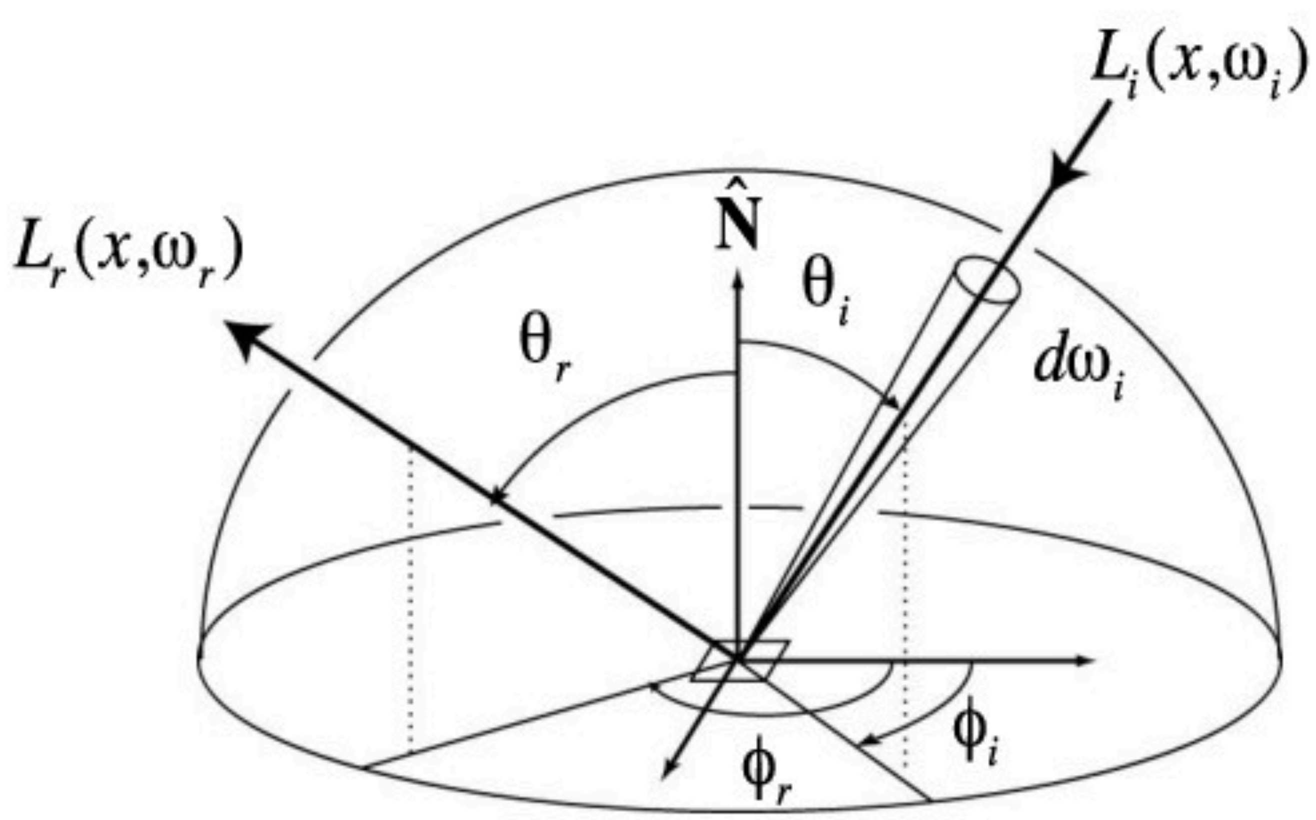
First pass: photon scattering

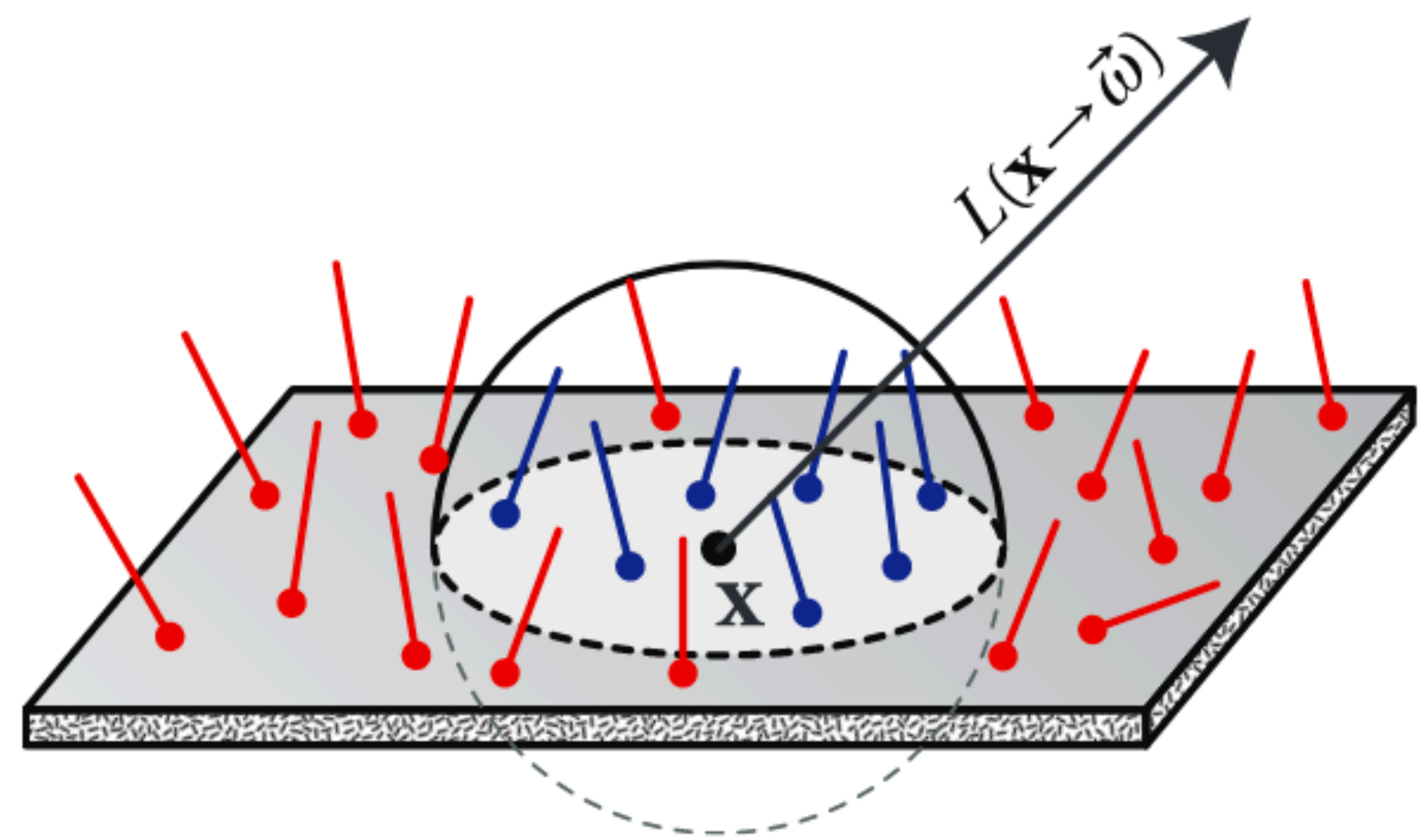
- "Global" photon map computed similarly to standard path tracing



- Separate caustics map is computed by emitting photons towards specular objects and storing when they hit diffuse surfaces

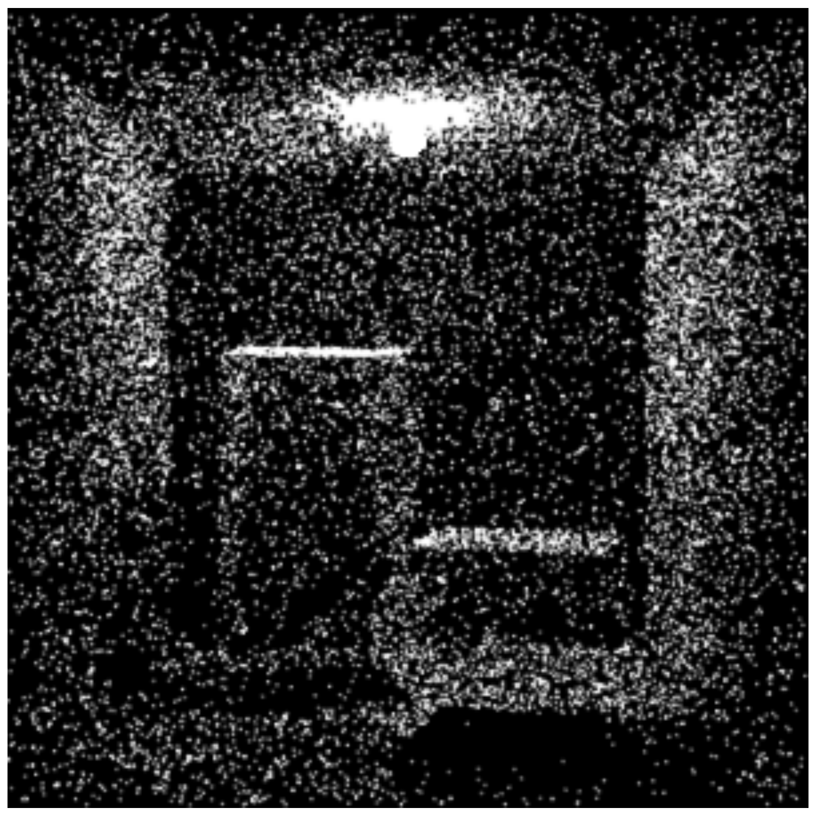
Using the photon map



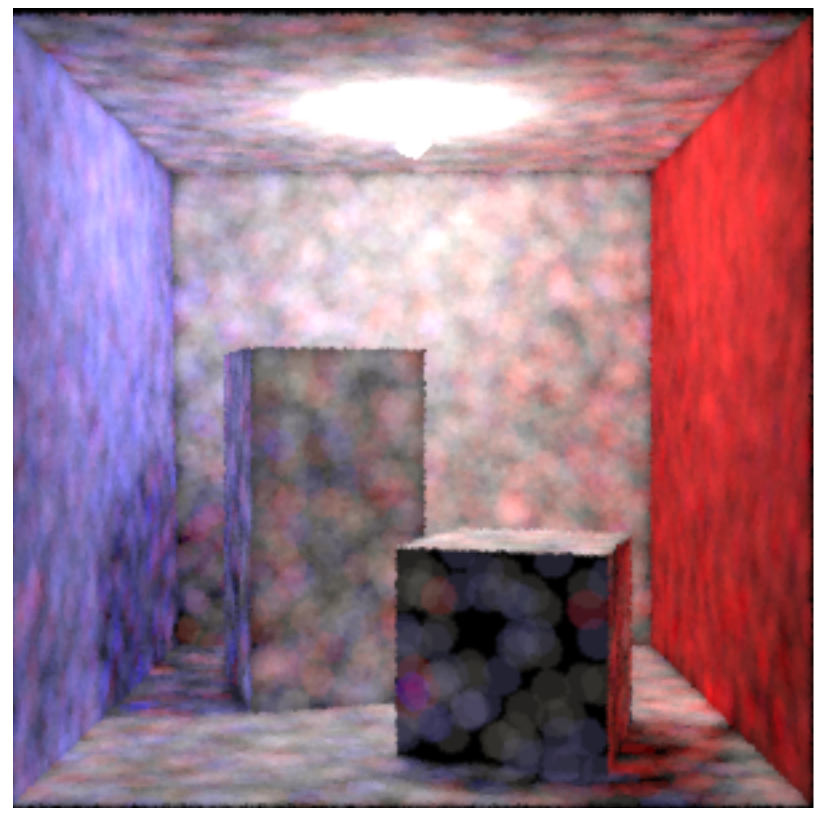


Visualizations

Photon map

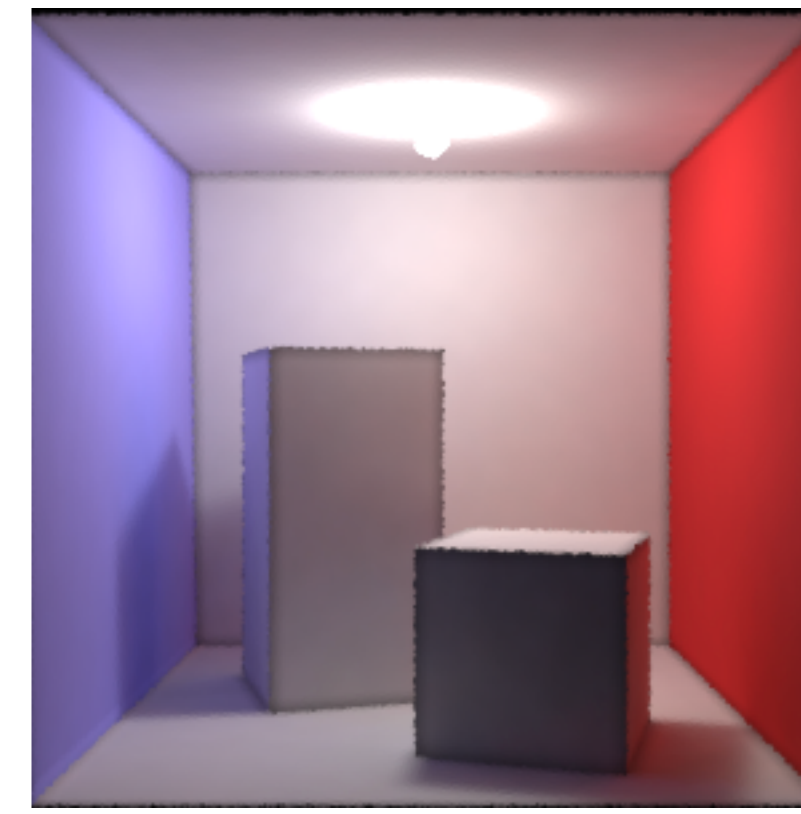


est. radiance (100k photons)

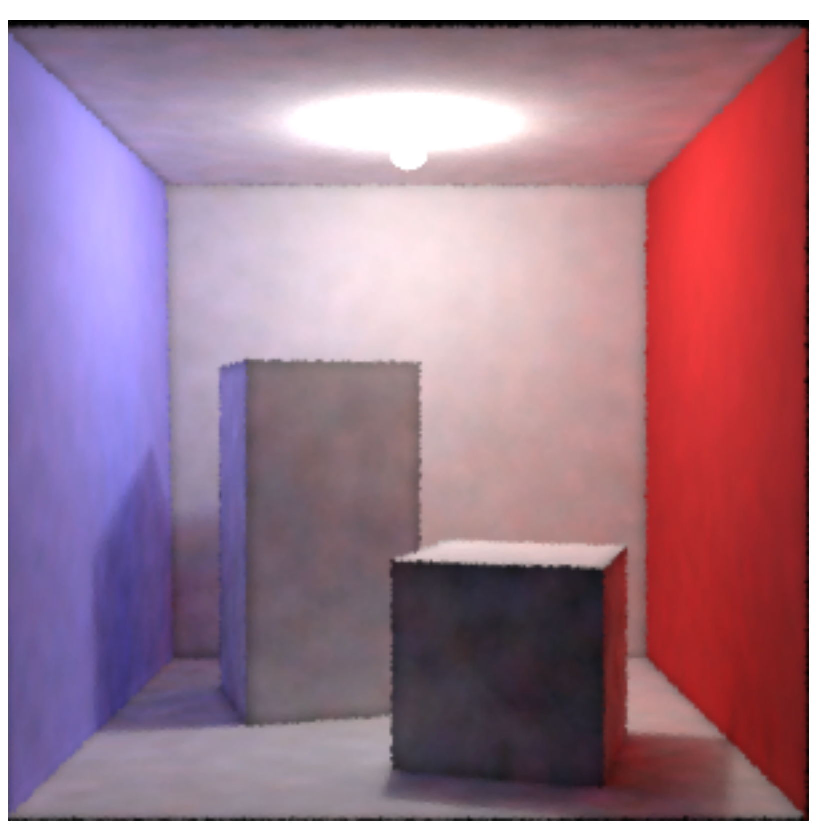


est. radiance (1M photons)

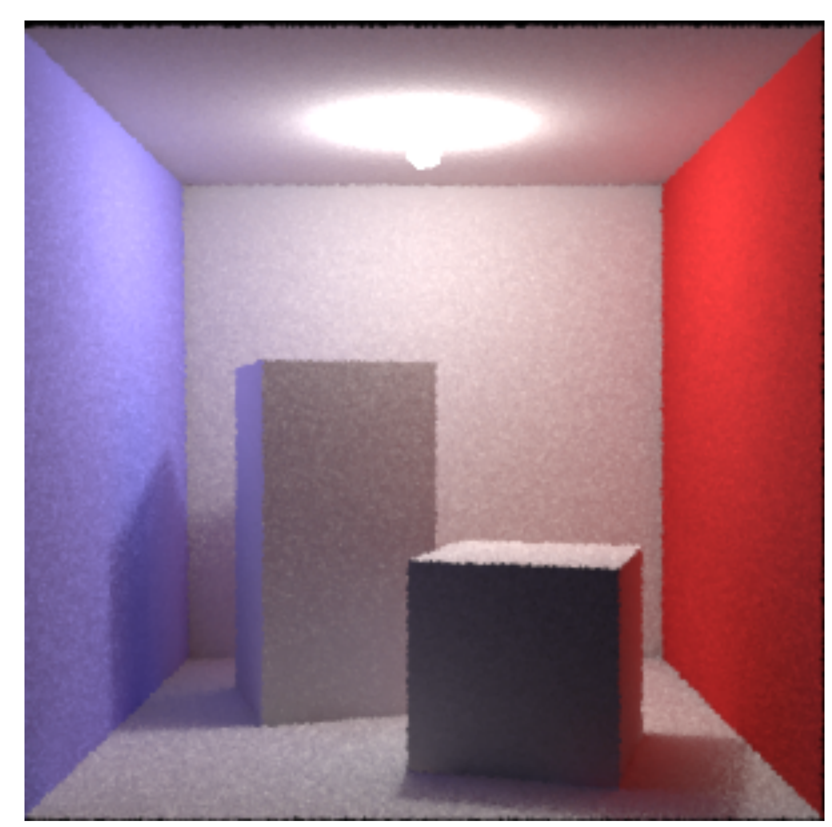
est. radiance (50M photons)



Normal raytrace for direct + photon map (1M photons)

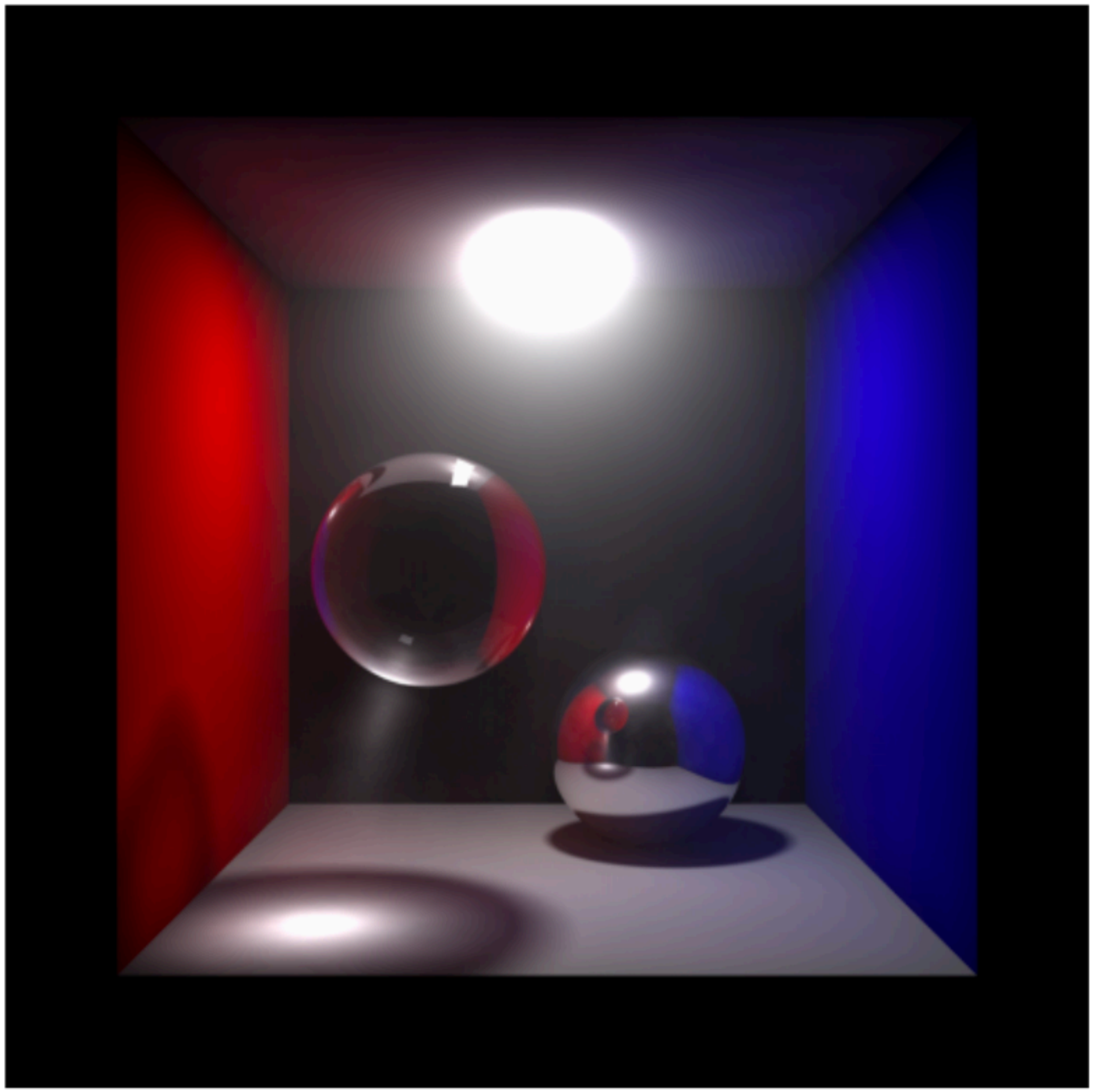
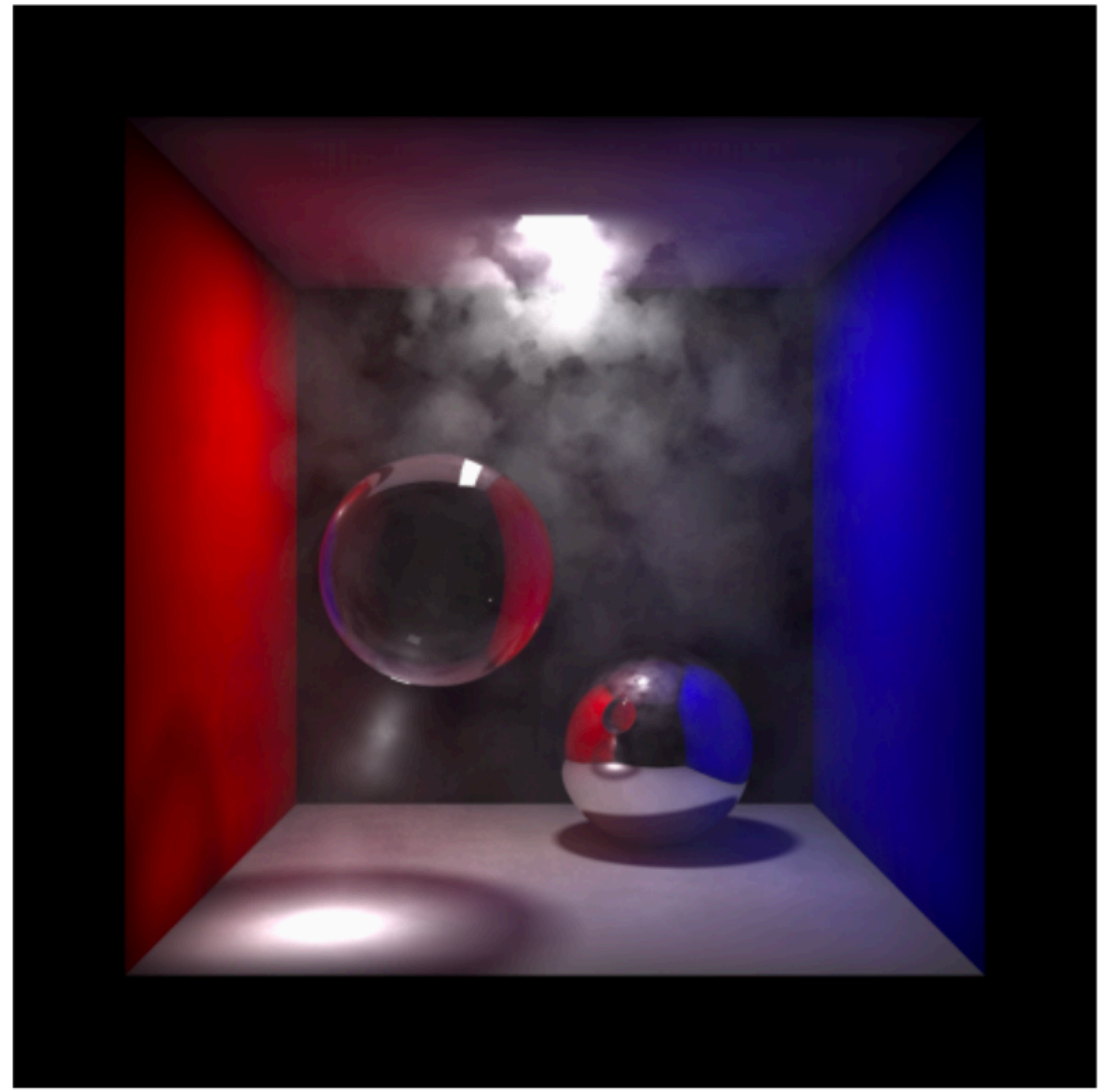


photon map estimate for direct (50k photons)



Source: McGill University ECSE 689


HENRIK WANN JENSEN 1996

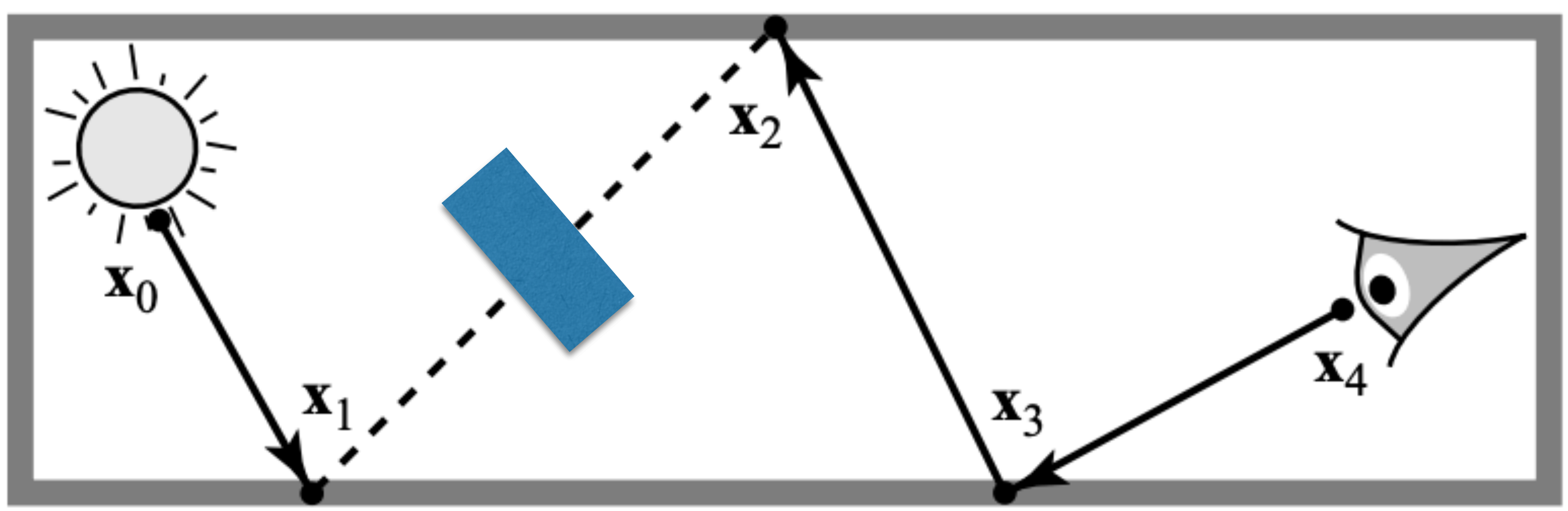
Unbiased? Consistent?

- Unbiased: a method is unbiased if it produces the correct answer on average.
    - "If I rendered the same image millions of times using different random numbers, would averaging the results give me the right answer?"
- Consistent: if an approximation approaches the correct solution as computation time increases, then the method is consistent
- Photon mapping is <u>biased</u>, but <u>consistent</u>!
    - <u>Need</u> to store a high resolution caustics map for crispy caustics

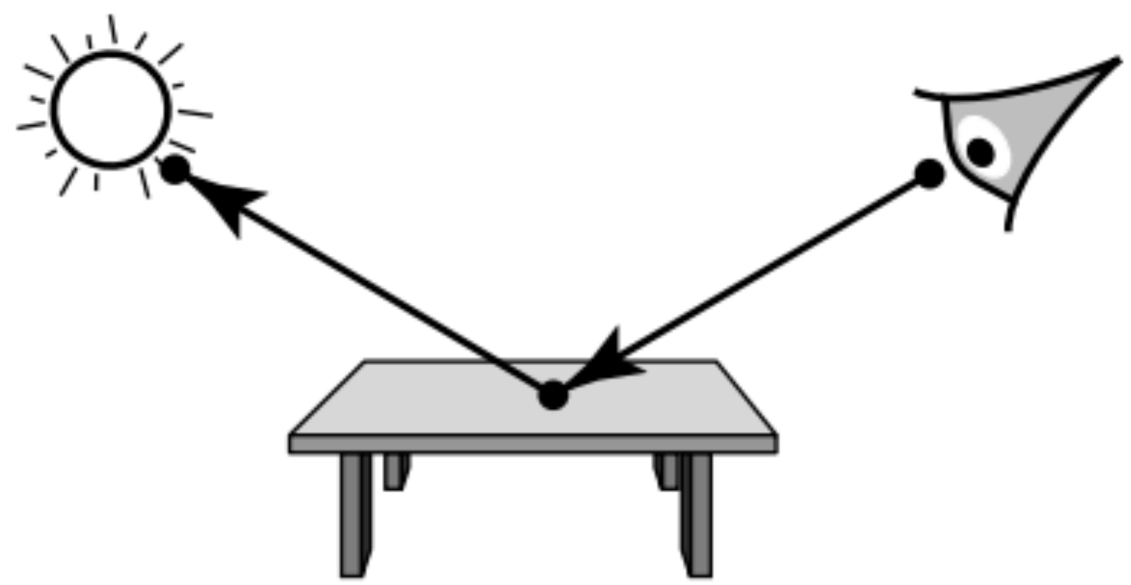
Keenan Crane, "Bias in Rendering"

Bidirectional Path Tracing

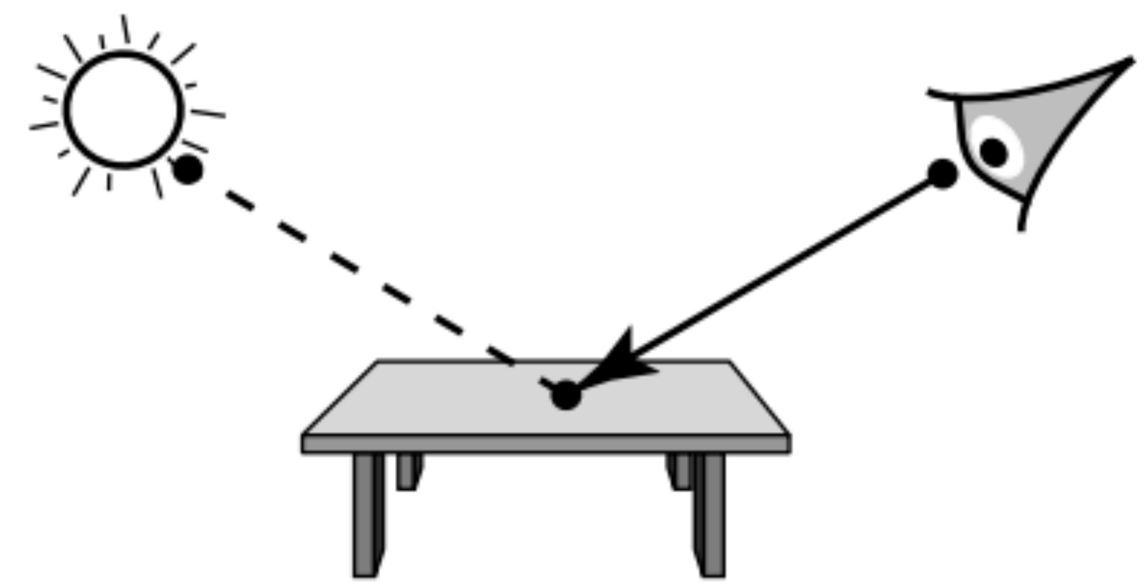
- Recall: path tracing is sampling the space of all possible light paths of all possible lengths
- Bidirectional path tracing: connect two independently generated pieces
- Unbiased and consistent



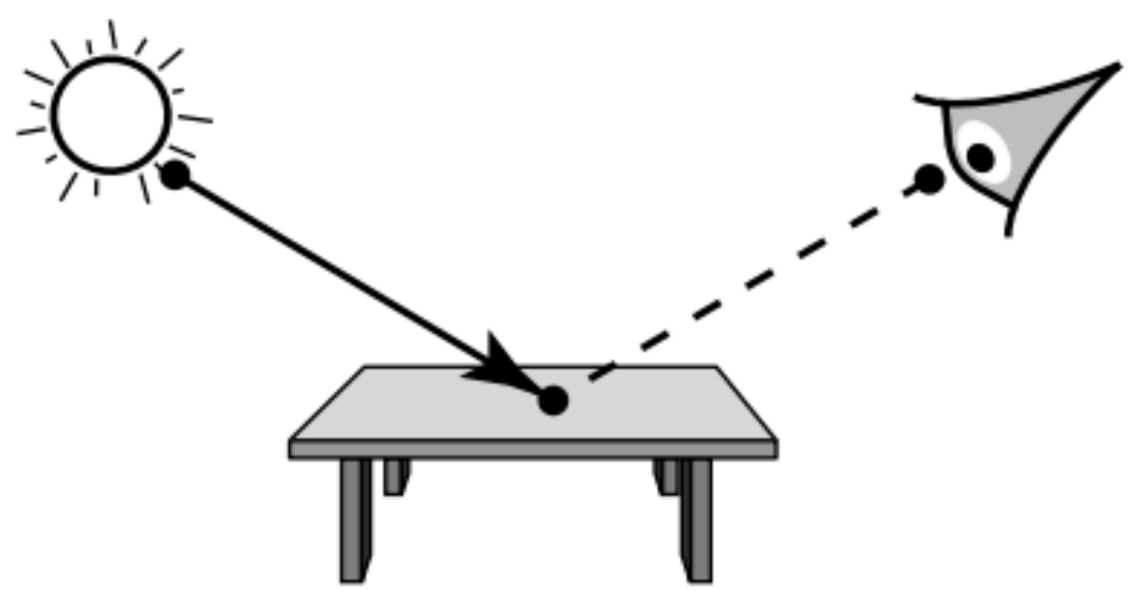
Source: Eric Veach



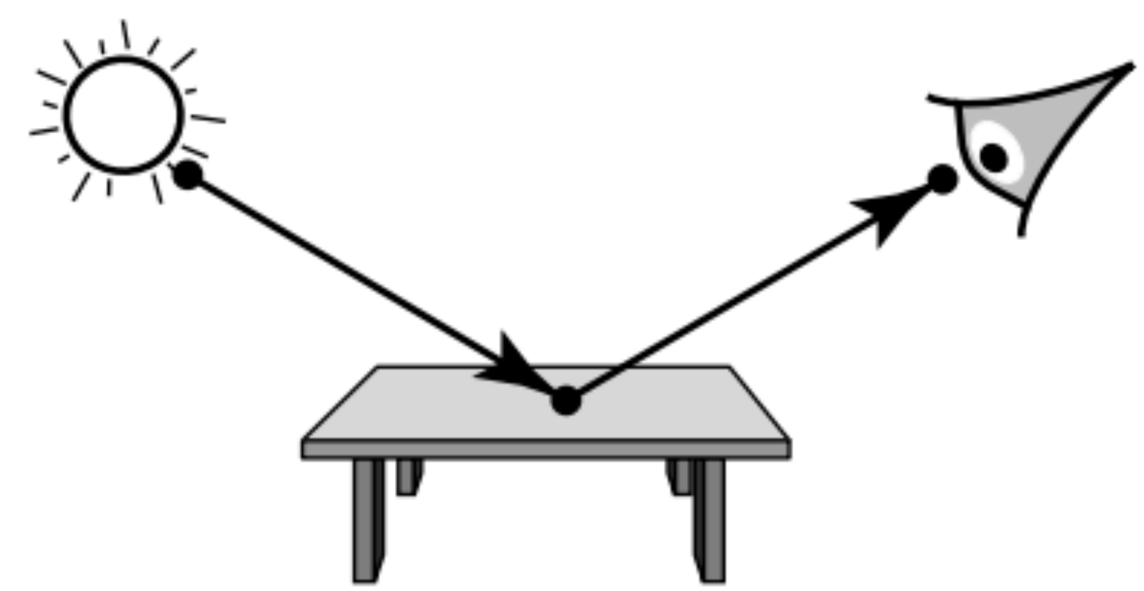
(a) $s = 0, t = 3$



(b) $s = 1, t = 2$



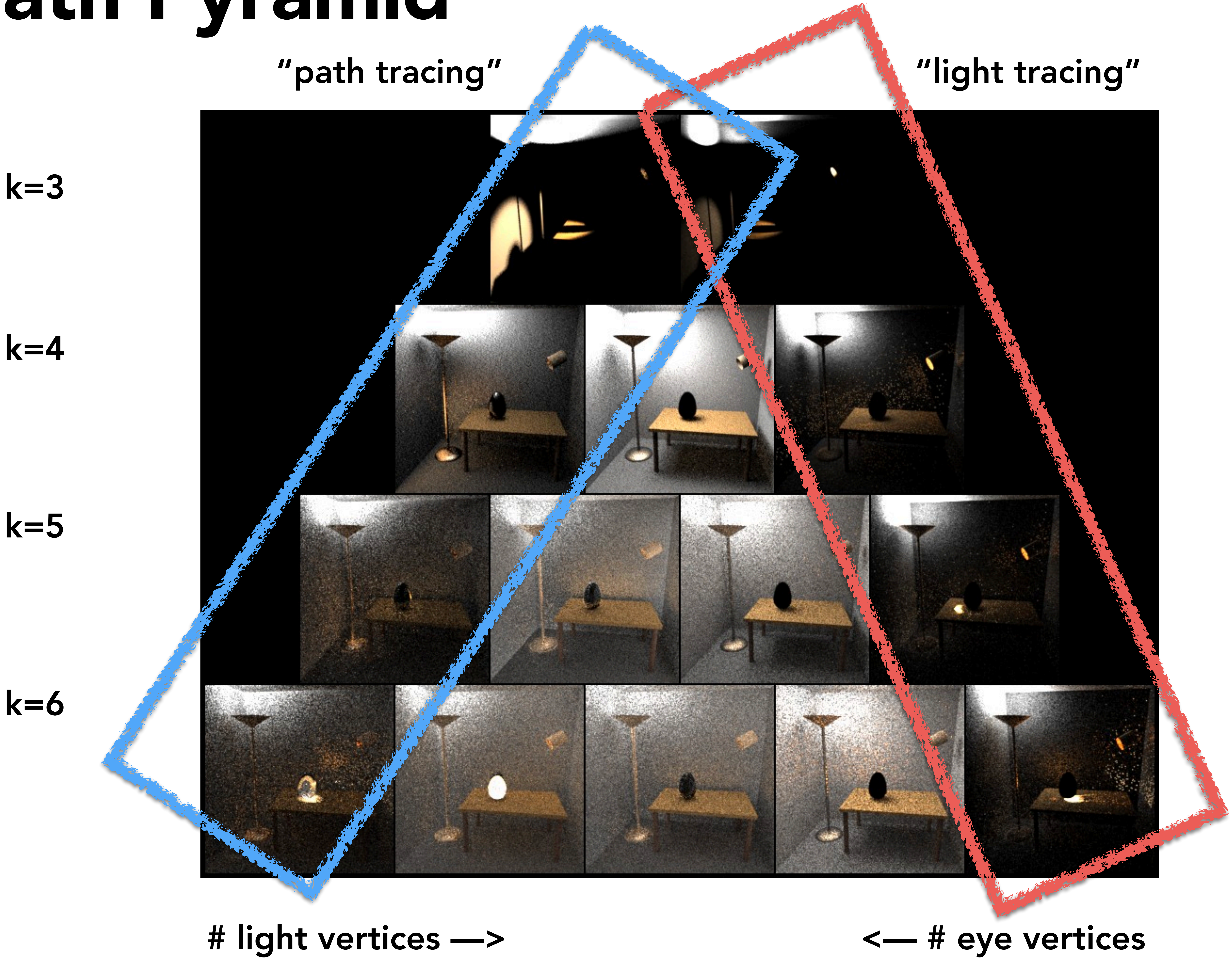
(c) $s = 2, t = 1$



(d) $s = 3, t = 0$

Source: Eric Veach

Path Pyramid



(a) Bidirectional path tracing with 25 samples per pixel

(b) Standard path tracing with 56 samples per pixel (the same computation time as (a))

Metropolis Light Transport

- Typically used to address the complex visibility problem
- Key point: Generate initial paths using bidirectional path tracing, then explore local mutations to that path
    - Exploit scene coherence to handle difficult light problems if needed
- Unbiased and consistent

Metropolis-Hastings

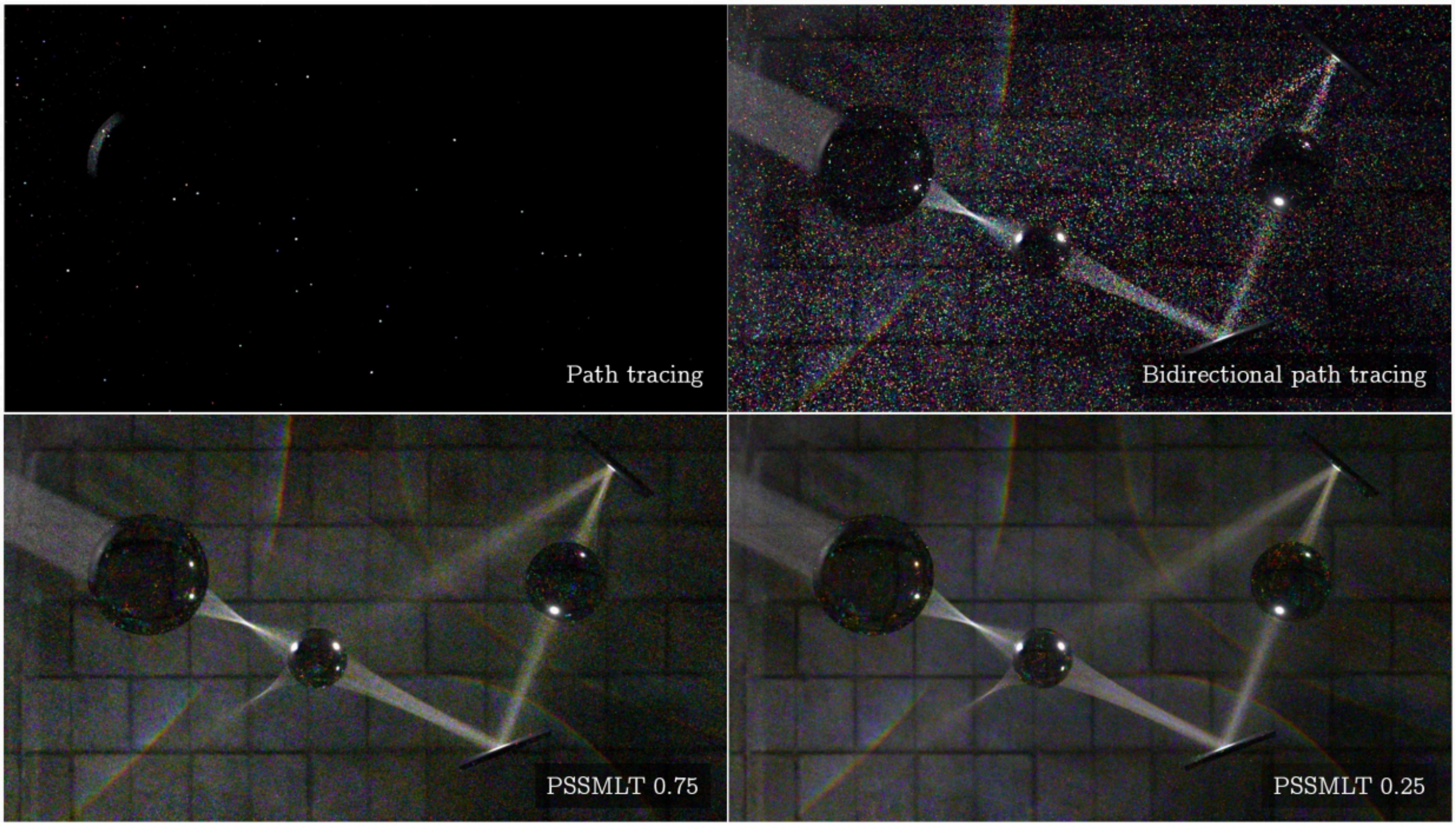
- Alternate Monte Carlo method
- The Algorithm:
  - Setup: define some distribution $f(x)$ that is proportional to the desired distribution, $P(x)$
  - Choose an arbitrary initial point x, and an arbitrary initial density $g(x|y)$ (usually Gaussian)
  - Iterate: Given a current point x, pick a new candidate x' from the distribution $g(x'|x)$
  - Accept the new candidate with probability $f(x')/f(x)$
  - If accepted, set x to x', otherwise keep x
- Intuitively, we want to generate (and keep generating) samples from high-density regions of $P(x)$
- How might this help us with particular problems in pathtraced rendering?

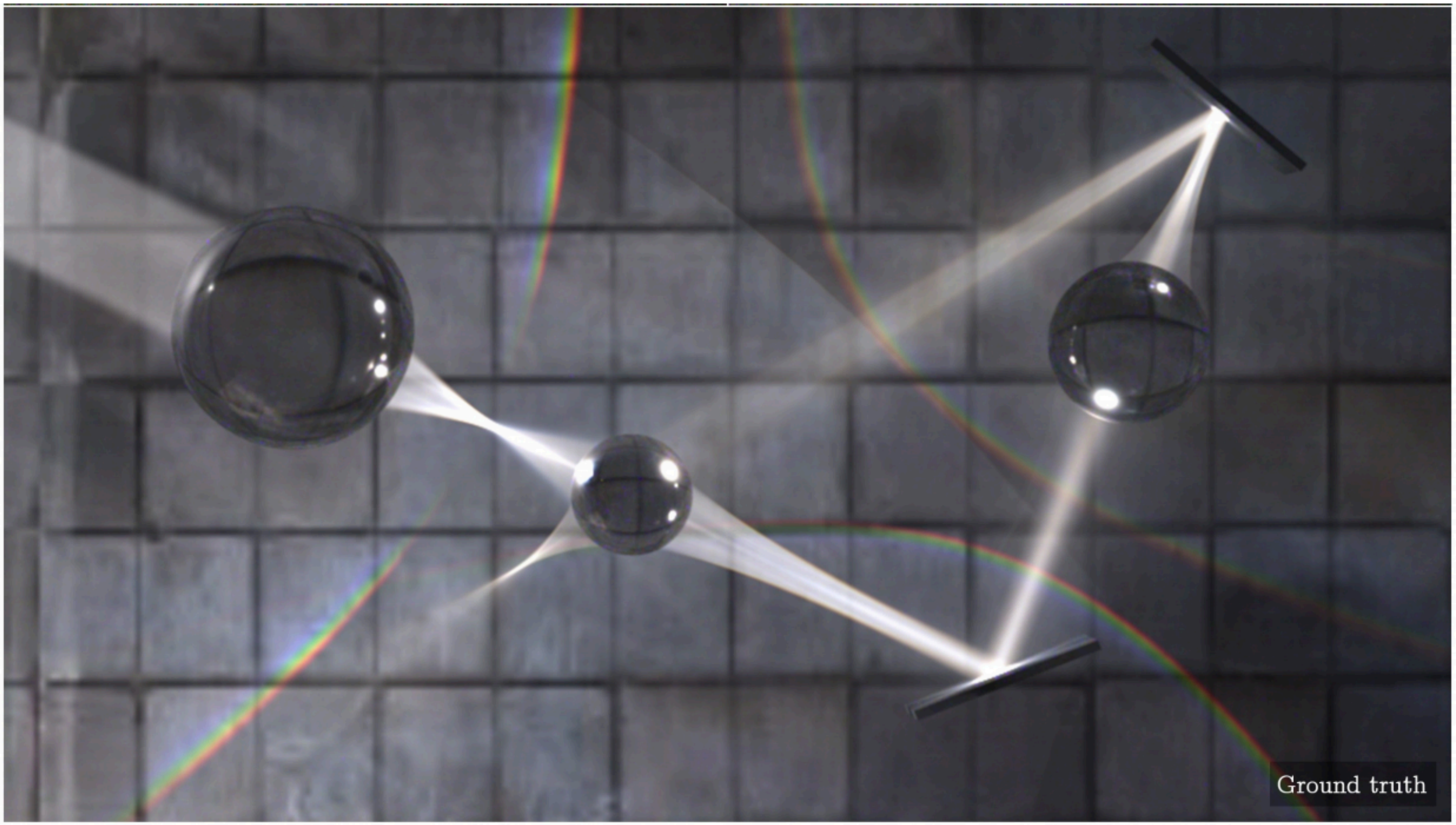
Metropolis Light Transport

- Again, treat the space of paths as a random variable we are trying to sample
    - An image is just a collection of such paths
- Ideally we'd like to find paths that are "light-to-eye" -- these contribute the most to our image!
- Randomly start with some "seed" paths
    - Use Metropolis-Hastings to perturb these paths to generate new ones
    - These will intuitively be concentrated in the "important" regions!
- Along a path, when it intersects with the scene, map the intersection location to the image and write that color in

Mutating Paths

- Ways to mutate paths
  - Add vertex
  - Delete vertex
  - Move vertex
- Some desirable properties of mutations
  - High acceptance probability of a new mutation
    - Otherwise too many of the same sample
  - Large changes to the path
    - Otherwise samples too highly correlated
  - … and more


Path tracing
Bidirectional path tracing
PSSMLT 0.75
PSSMLT 0.25


Ground truth

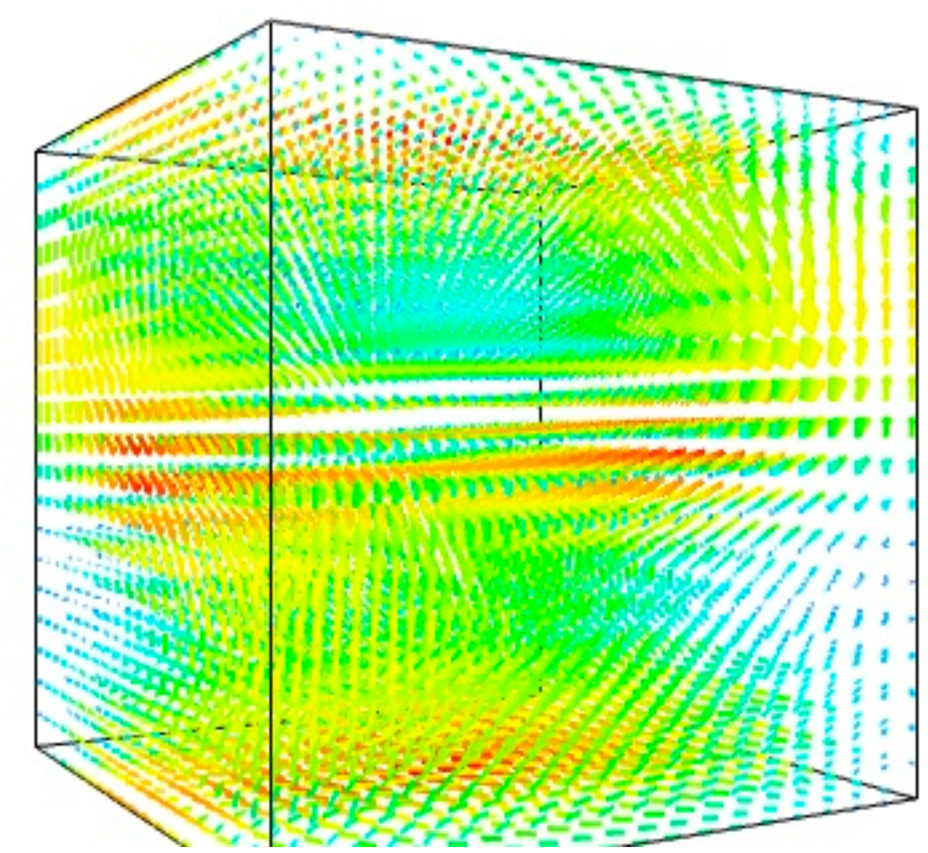
Bidirectional Path Tracing

Metropolis Light Transport

Volume Rendering

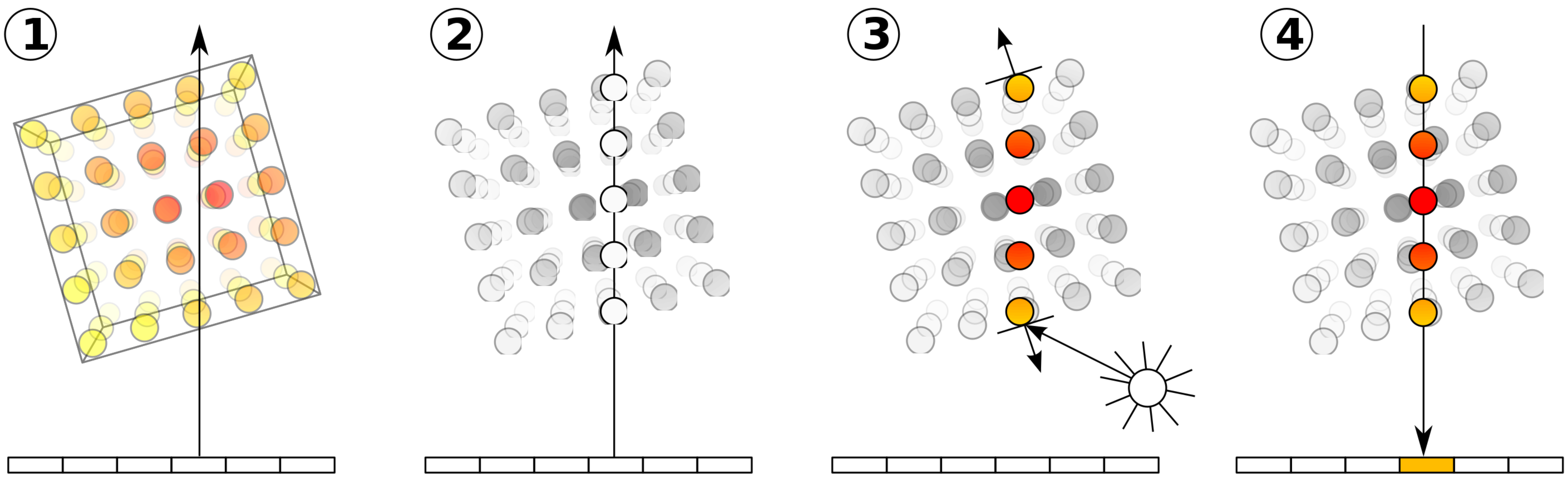
Voxels

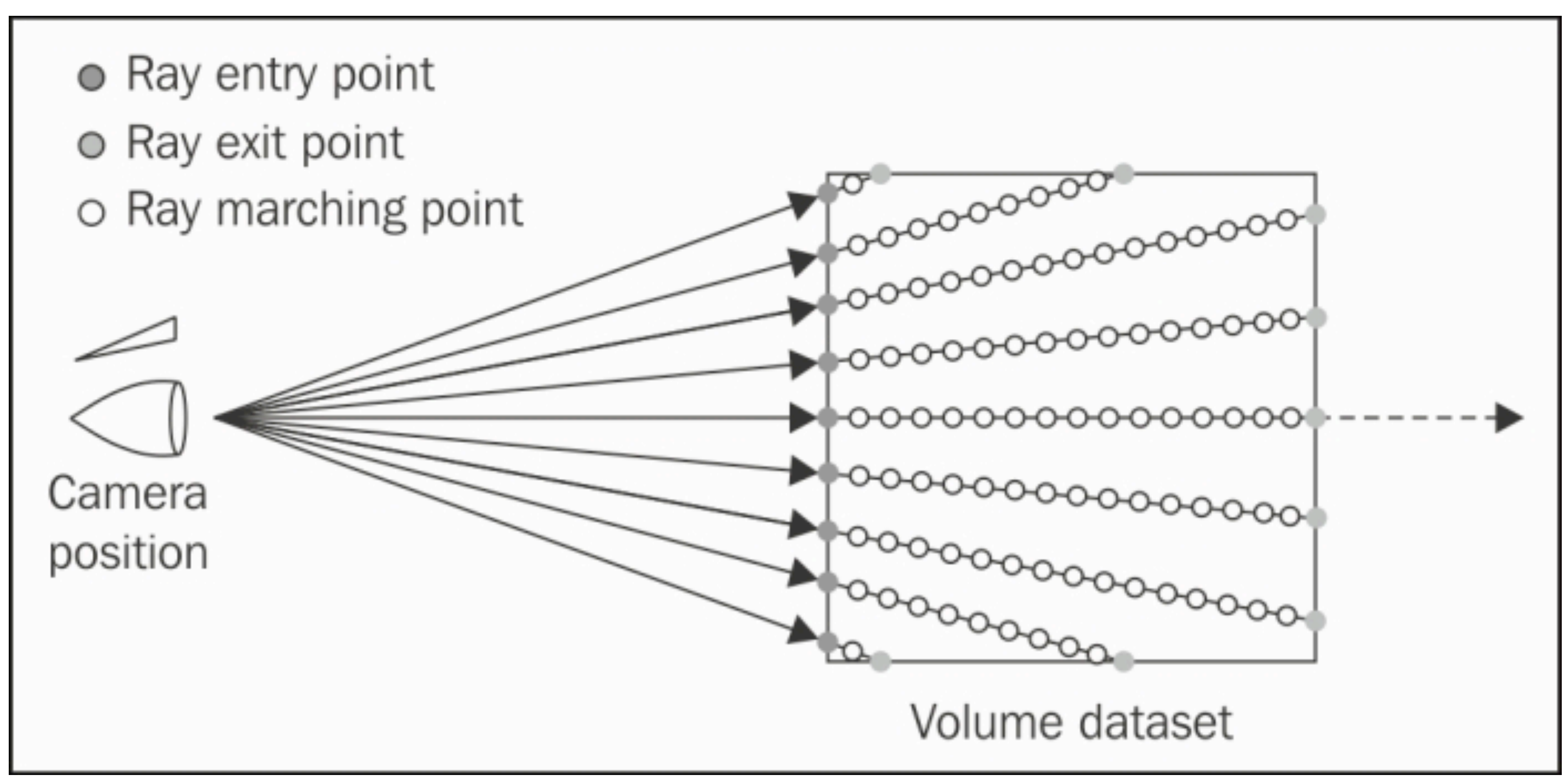
Tomography



3D Scalar Fields

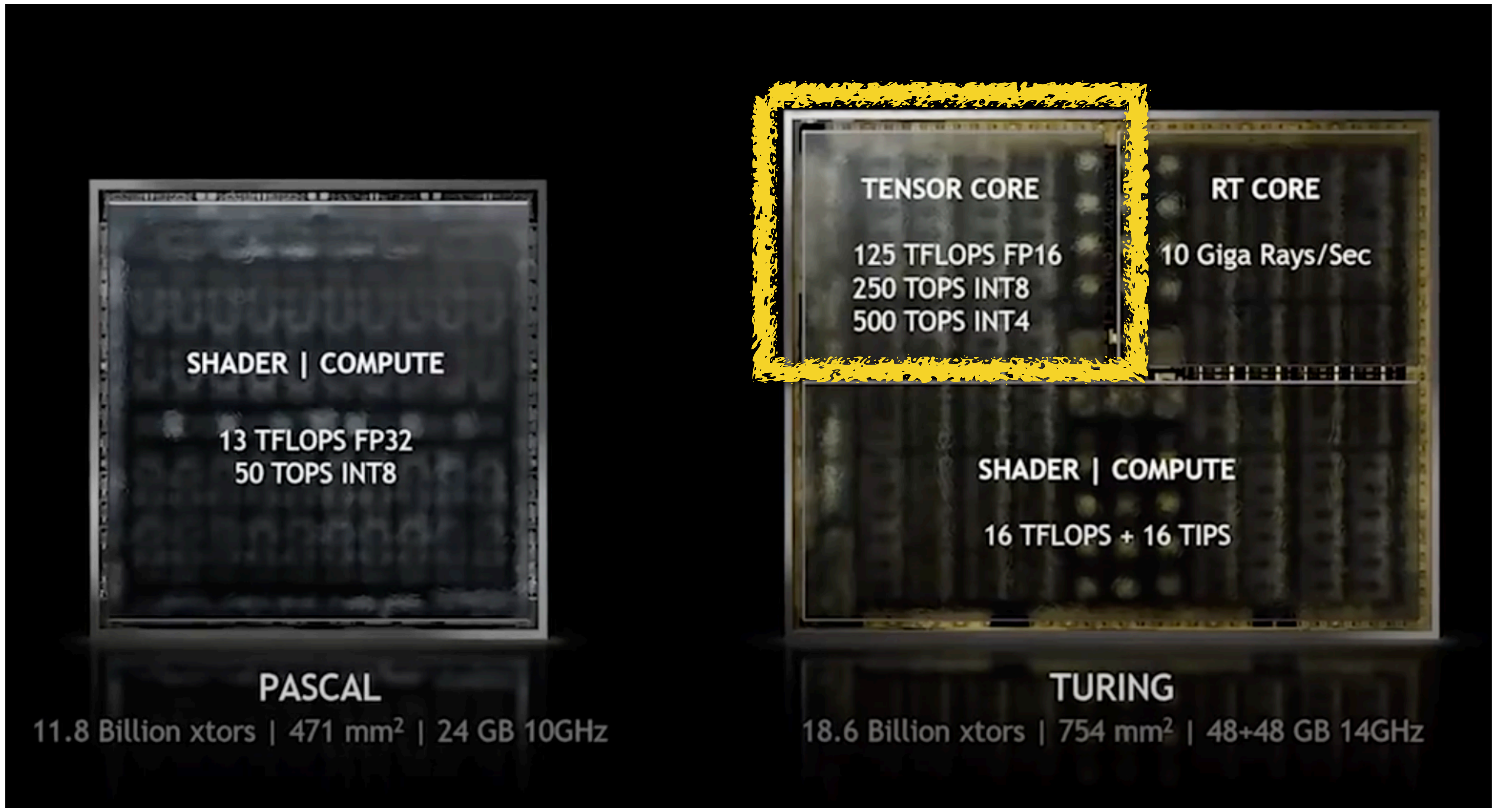
Volume Ray Casting



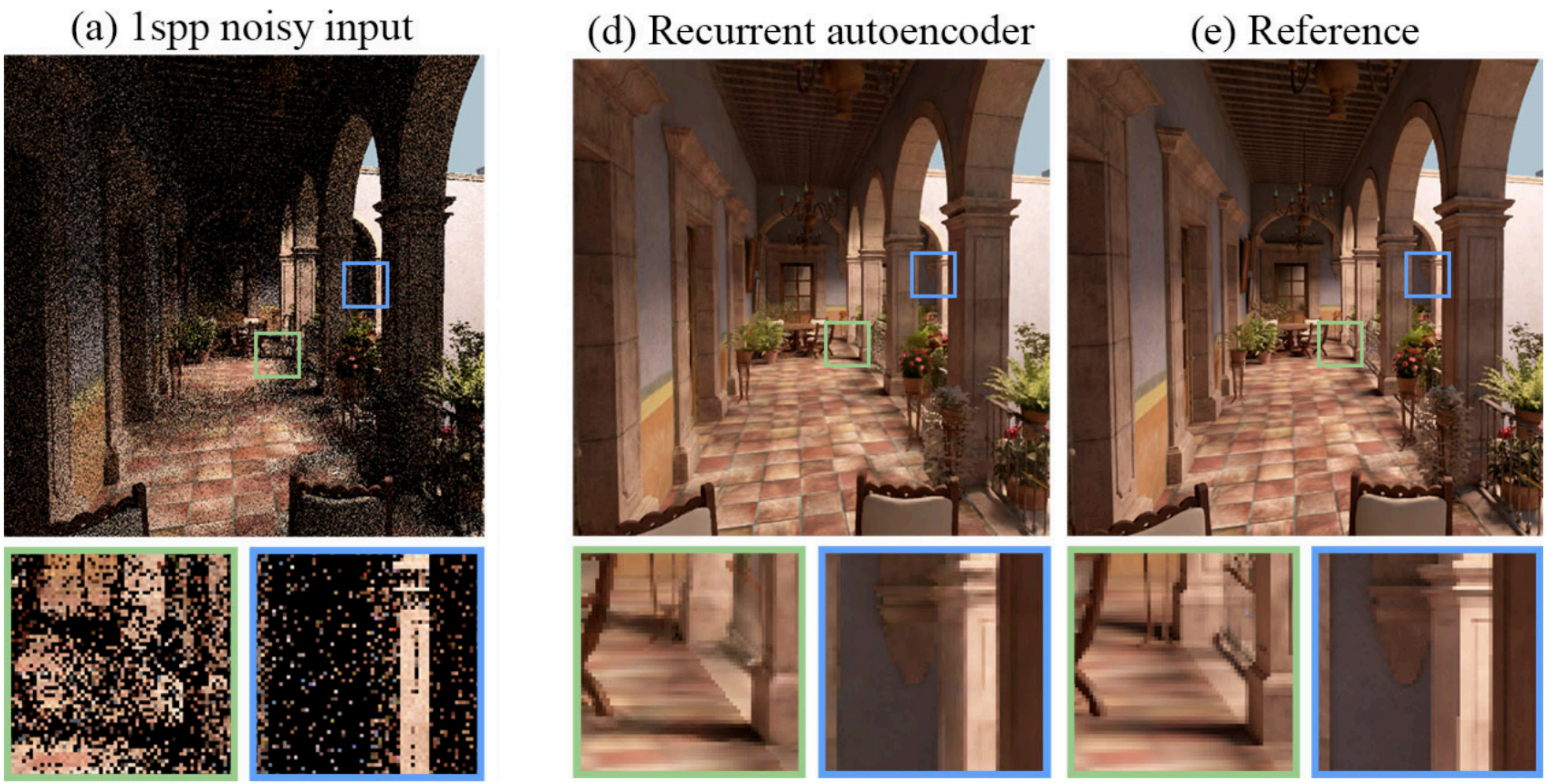


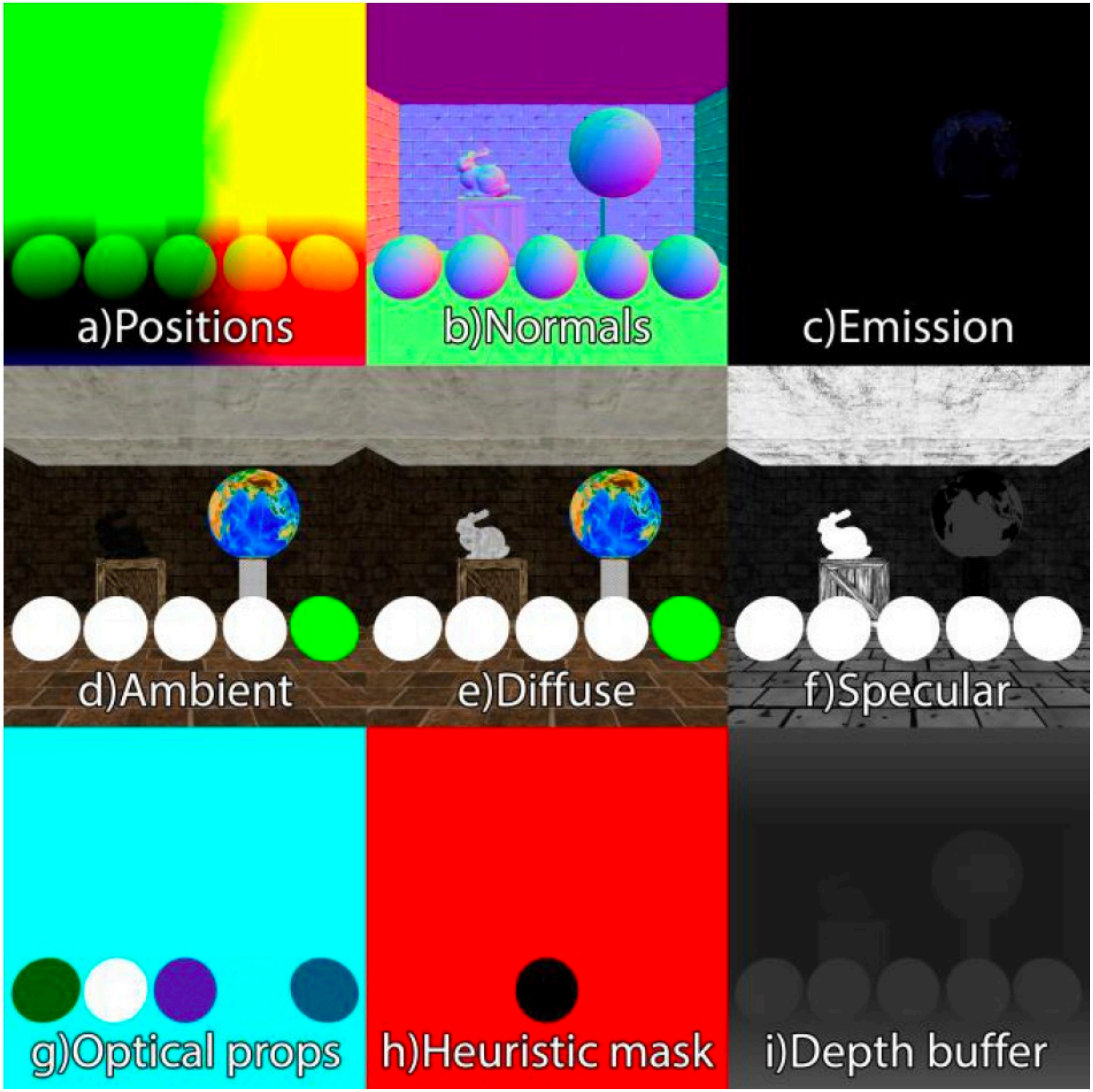
Denoising Renderings

- "Interactive Reconstruction of Monte Carlo Image Sequences using a Recurrent Denoising Autoencoder", Chaitanya 2017 (NVIDIA)
    - Probably what part of DLSS is based off of
        - DLSS is meant to handle both low ray counts and low image resolution



Denoising Renderings




a)Positions
b)Normals
c)Emission
d)Ambient
e)Diffuse
f)Specular
g)Optical props
h)Heuristic mask
i)Depth buffer

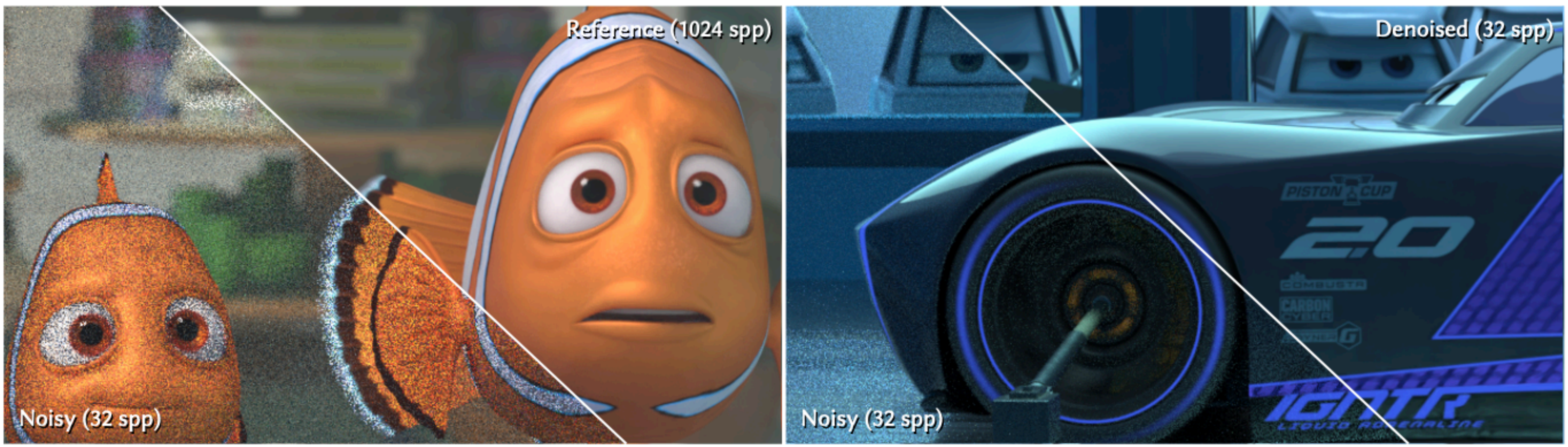
Training Network

- Input:
  - "1 sample” image in HDR (1 ray per pixel, up to two bounces) (RGB)
  - Buffers for 2D normal, depth, material roughness
  - Total = 7 scalar values per pixel
- Training data:
  - Get ~1000 frames per scene. 10 different renders per frame.
  - Target image is 2000 or 4000 sample image (!)
  - Pass in a sequence of inputs

Challenges

- In the paper, retrained for every scene
- In practice (e.g. for DLSS), needs to be retrained for every game.
    - Especially challenging moments: when objects suddenly appear in the frame

Other Pathtracing Denoisers



TRAINING

TEST

"Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings", Bako 2017