

Photon-Mapping

© 2009-2024 Josef Pelikán
CGG MFF UK Praha

pepca@cgg.mff.cuni.cz
<https://cgg.mff.cuni.cz/~pepca/>



Basics of Photon-mapping

Based on **light/photon tracing**

- arbitrary **scene geometry**
- utilization of efficient (long-term optimized) **ray-casting libraries, acceleration techniques**, etc.

Light is traced both **forwards** (from sources) and **backwards** (from the camera)

- camera represents importance (potential)
- lights are sources of photons

Separation of scene geometry and representation of light

- 3D scene can be very complex
- light representation can be optimized independently



Photon Map

Data structure – **impacts of individual photons**

- represents even very varying light conditions
- completely separated from scene geometry
- memory efficient

“Caching paths of bi-directional Path-tracing”

- light estimate is free of HF noise
- ... much faster than classical Monte Carlo techniques (equal quality)

Biased method !

- but **consistent** (converges to better result)



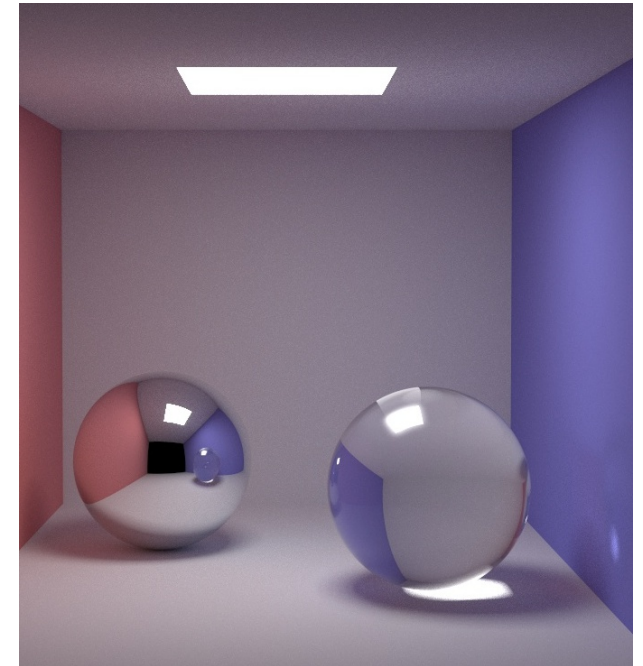
Algorithm scheme

Photon-tracing

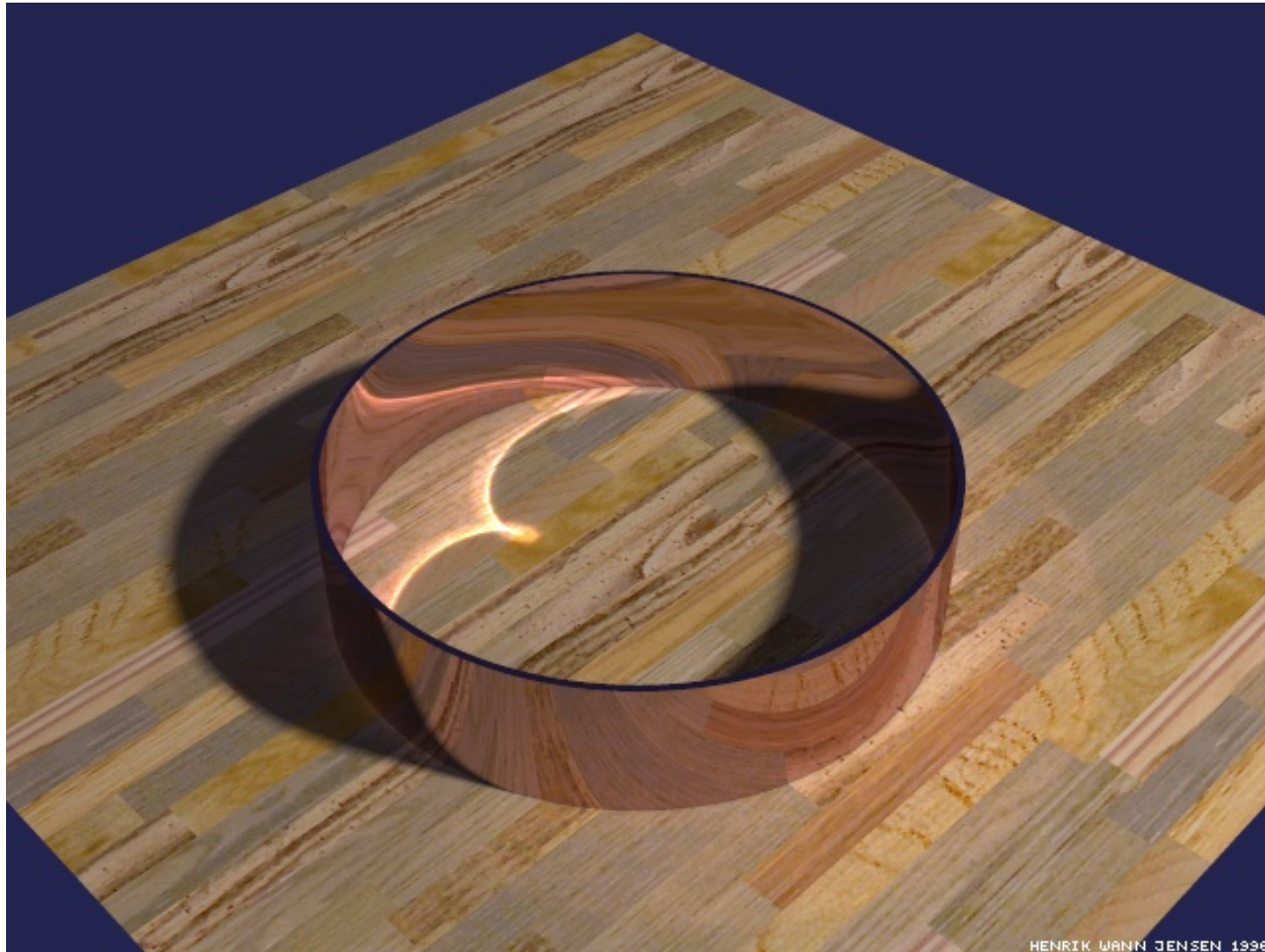
- photons are generated by light sources,
- propagated to the scene (Monte-Carlo)
- and finally stored in photon maps
(**global maps** for smooth changes
and **caustic maps** for sharp edges)

Rendering

- photon maps are used for efficient rendering of the scene
- plain Ray-tracing or
- a Monte Carlo method (Path-tracing)



Photon-mapping – examples



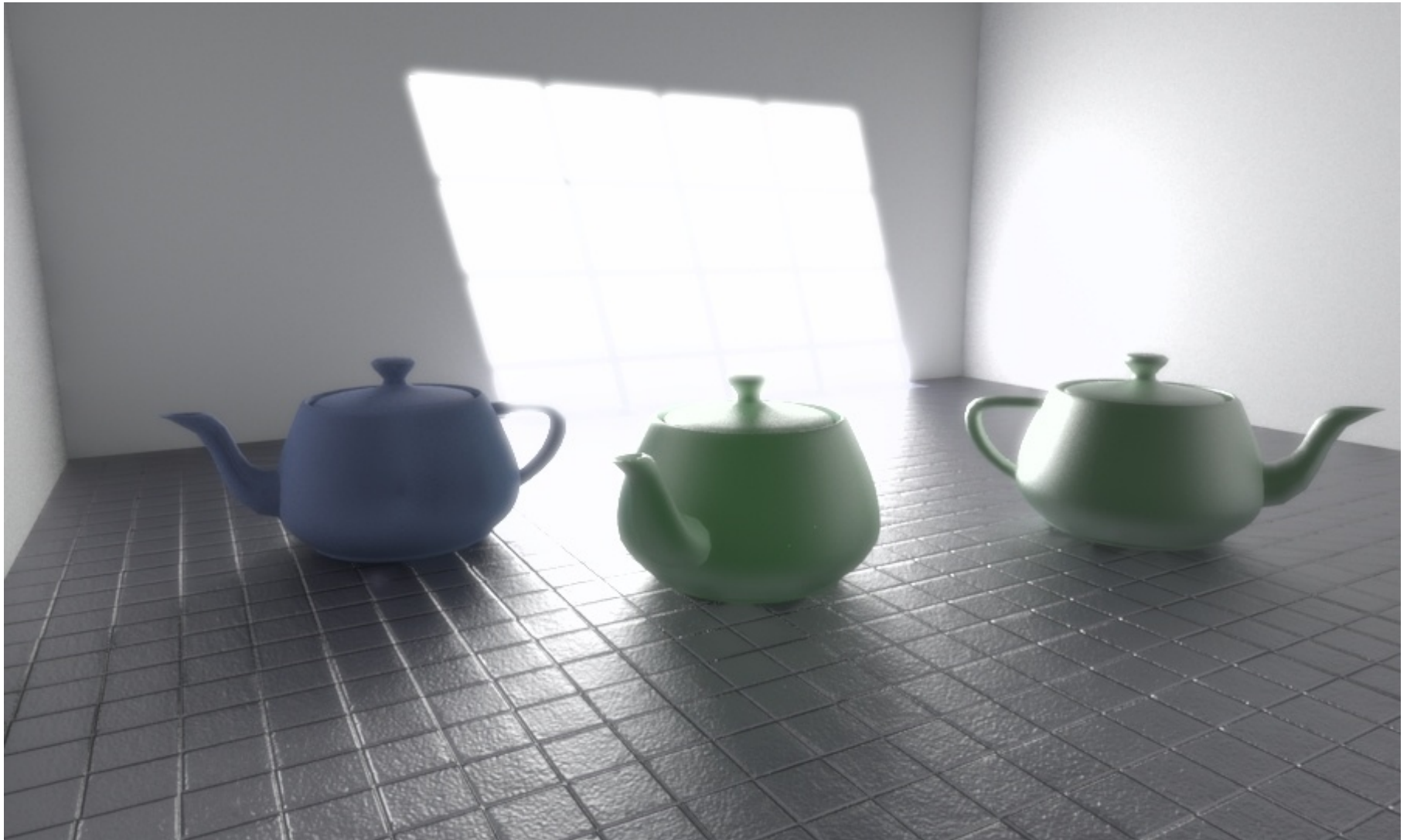
Photon-mapping – examples



Photon-mapping – examples



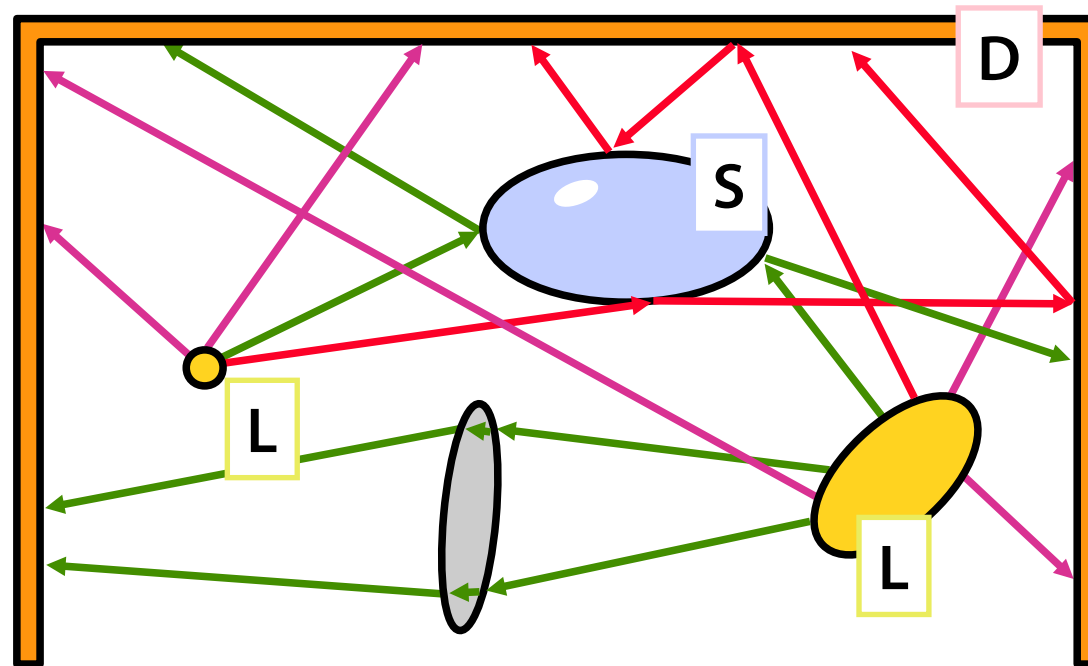
Photon-mapping – examples





Photon-tracing

Photons are generated by light sources,
Randomly propagated through the scene and
Stored in photon maps





Photon generation

The most elegant approach – each photon carries **the same light energy**

Random sampling of light sources

- "rejection sampling" for difficult distributions

More light sources...

- distribution among them (based on their total contribution)

Efficient sampling

- pre-computed **projection maps** (see acceleration of Ray-tracing)



Photon scattering

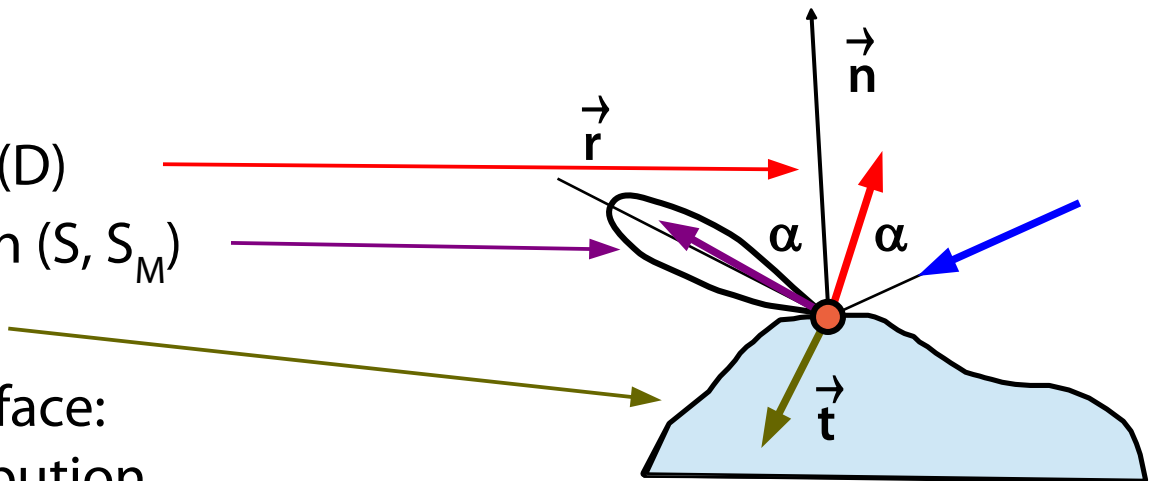
Refraction and reflection should alter (reduce)

photon energy

- photon map would contain nonequivalent entries

Keeping constant photon energy ... **Russian roulette**

- photon is (randomly) **bounced with original energy** or **terminated**
- **three options**
 - » 1. diffuse reflection (D)
 - » 2. specular reflection (S, S_M)
 - » 3. refraction
 - » on every diffuse surface:
photon-map contribution





Data structure for photon map

Photon

- impact **position** (float[3])
- impact **direction** (float[2] or compression into int8[2])
- photon **energy** (RGB, spectrum or RGBE = int8[4])
- tree construction attributes/flags (e.g. “splitting plane”)

The photon map has to be **very fast** event for very high **number of records**

- 10^5 to 10^7 individual records
- crucial operation: **fast nearest neighbor lookup**
 - » K nearest or all records in the given radius R
- **KD-trees** work well (binary trees, data in all nodes, optimally balanced)



In the construction phase records are only gathered, it needs to be **balanced before actual usage**

Optimization for geometric lookup

- **splitting plane** can be determined from maximum range or variance
- stored in an array – **without pointers !**

à la Jensen

- heap-like system (descendants have indices: $2i$, $2i+1$)

à la Hooley ("cache-friendly")

- the median is fixed, two segments are heap-sorted



Nearest neighbor lookup

The heap for branches not yet visited

Pruning based on

- current distance of K-nearest neighbor photon (KNN approach)
- required radius of interest R



Radiance estimate I

Emitted radiance from $\mathbf{x}$

$$L_r(\mathbf{x}, \omega_o) = \int_{\Omega} f_r(\mathbf{x}, \omega_i \rightarrow \omega_o) \cdot \underline{L_i(\mathbf{x}, \omega_i)} \cdot \cos \theta_i \, d\omega_i$$

Expressed using **radiant flux**

$$L_r(\mathbf{x}, \omega_o) = \int_{\Omega_x} f_r(\mathbf{x}, \omega_i \rightarrow \omega_o) \cdot \frac{\partial^2 \Phi_i(\mathbf{x}, \omega_i)}{\partial A_i}$$



Radiance estimate II

Radiance estimate from **photon map** surrounding **x**
(looking for **n** nearest photons)

$$L_r(x, \omega_o) \approx \sum_{p=1}^n f_r(x, \omega_p \rightarrow \omega_o) \cdot \frac{\Delta \Phi_p(x, \omega_p)}{\Delta A}$$

For **circular** neighborhood (n-th photon has distance **r**)

$$L_r(x, \omega_o) \approx \frac{1}{\pi r^2} \sum_{p=1}^n f_r(x, \omega_p \rightarrow \omega_o) \cdot \Delta \Phi_p(x, \omega_p)$$



Filtering on photon map

If the number of photons is too low, radiance estimate is blurry ("box filter")

- problem in case of a caustic map

Filter can accent samples in the middle

- cone filter
- Gaussian filter
- **differential control** – monitoring change of average value (or variance) while adding more photons, stop the process if changes are marginal



Global illumination I

Survey of previous formulae

$$L_o(x, \omega_o) = L_e(x, \omega_o) + L_r(x, \omega_o)$$

Reflected radiance

$$L_r(x, \omega_o) = \int_{\Omega_x} f_r(x, \omega_i, \omega_o) \cdot L_i(x, \omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i$$

BRDF components

$$f_r(x, \omega_i, \omega_o) = f_{r,d}(x, \omega_i, \omega_o) + f_{r,s}(x, \omega_i, \omega_o)$$



Global illumination II

Incoming radiance classification L_i

$L_{i,l}(x, \omega_i)$ direct light from light source L

$L_{i,c}(x, \omega_i)$ caustics – light from source concentrated by reflection/refraction $L S^+$

$L_{i,d}(x, \omega_i)$ indirect light reflected diffusely at least once $L S^* D (D|S)^*$

$$L_i(x, \omega_i) = L_{i,l}(x, \omega_i) + L_{i,c}(x, \omega_i) + L_{i,d}(x, \omega_i)$$



Global illumination III

Reflected radiance (bounce point **x** was left out)

$$\begin{aligned} L_r(\omega_o) = & \int_{\Omega_x} f_r(\omega_i, \omega_o) \cdot L_{i,l}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i + \\ & \int_{\Omega_x} f_{r,s}(\omega_i, \omega_o) \cdot (L_{i,c}(\omega_i, \omega_o) + L_{i,d}(\omega_i, \omega_o)) \cdot \cos \theta_i \, d\omega_i + \\ & \int_{\Omega_x} f_{r,d}(\omega_i, \omega_o) \cdot L_{i,c}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i + \\ & \int_{\Omega_x} f_{r,d}(\omega_i, \omega_o) \cdot L_{i,d}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i \end{aligned}$$



Accuracy

“Accurate” computation

- if point x is visible (in the result image) ... or
- if it is visible via a couple of specular reflections ... or
- if the ray is too short (eliminates “color bleeding”)

Approximate computation

- in all other cases
- ... if there is at least one diffuse reflection
- ... if the ray has too small performance/importance (accumulated reflection coefficient)



Direct light

Light coming directly from light sources

$$\int_{\Omega_x} f_r(\omega_i, \omega_o) \cdot L_{i,l}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i$$

In Ray-tracing “shadow rays” are used

- multiple test rays for area light sources (“distributed R-T”)

Accurate case: shadow rays or photon map

- speedup ... photon map can store “**shadow photons**”

Approximate case: only global photon map is used

- no secondary rays



Mirror and specular reflection

Indirect light from specular BRDF component

$$\int_{\Omega_x} f_{r,s}(\omega_i, \omega_o) \cdot (L_{i,c}(\omega_i, \omega_o) + L_{i,d}(\omega_i, \omega_o)) \cdot \cos \theta_i \, d\omega_i$$

Classical Monte Carlo technique (“distributed R-T”)

- accuracy is sufficient even in demanding situations (direct visibility)
- for reasonable accuracy only a few reflected rays need to be computer

Caustics



Light from light source concentrated on diffuse surface

$$\int_{\Omega_x} f_{r,d}(\omega_i, \omega_o) \cdot L_{i,c}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i$$

Accurate case: caustic photon map

- very high density of photons, accuracy is high (sharp caustics)

Approximate case: using global photon map



Multiple diffuse reflection

Light reflected diffusely multiple times

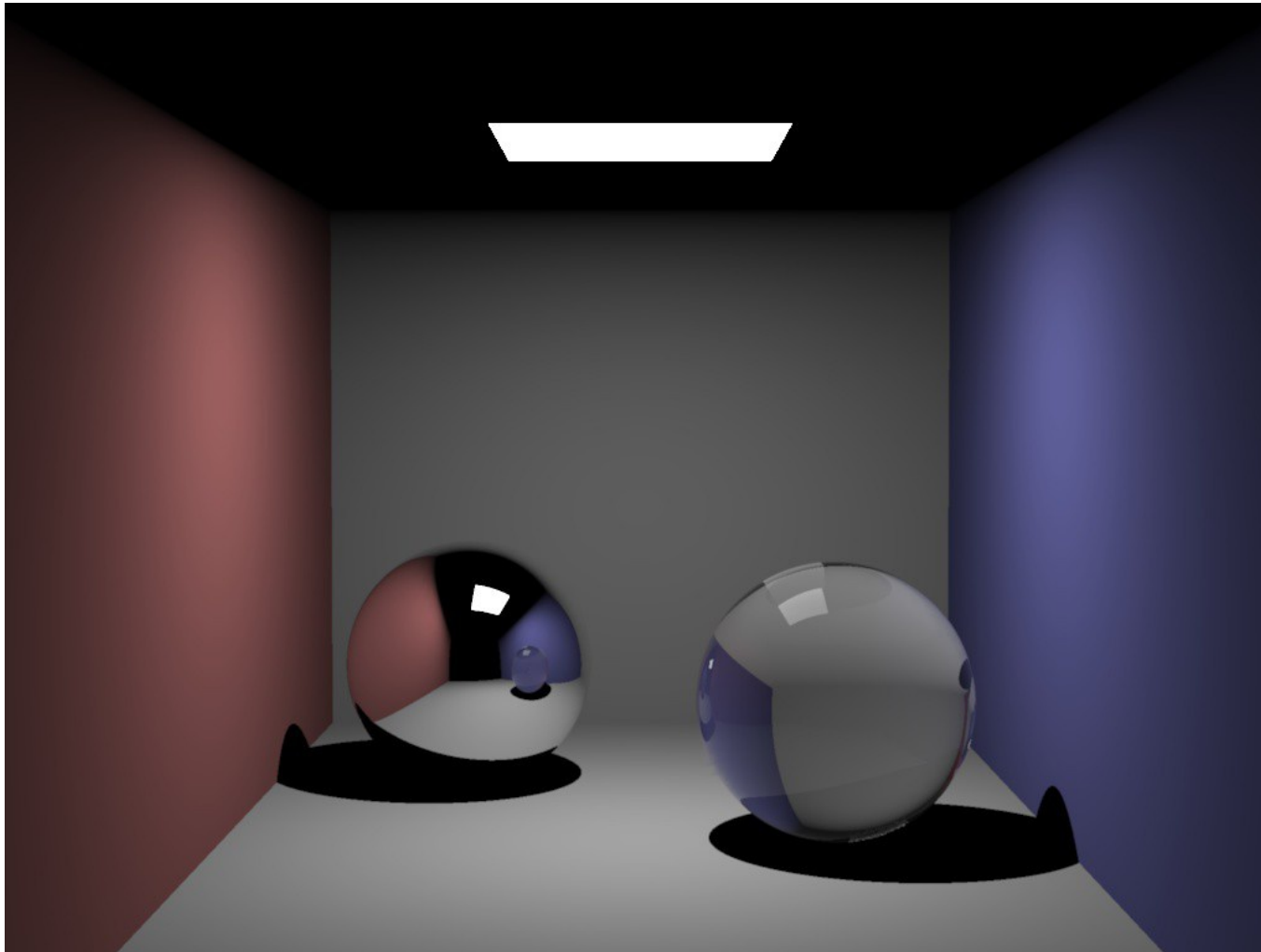
$$\int_{\Omega_x} f_{r,d}(\omega_i, \omega_o) \cdot L_{i,d}(\omega_i, \omega_o) \cdot \cos \theta_i \, d\omega_i$$

Accurate case: “distributed Ray-tracing” (Monte Carlo)

- sampling optimized using global photon map (distribution of impact directions is known in the neighborhood)
- more speedup: “Irradiance caching” (Ward 1988)

Approximate case: using global photon map

Samples – Ray-Tracing

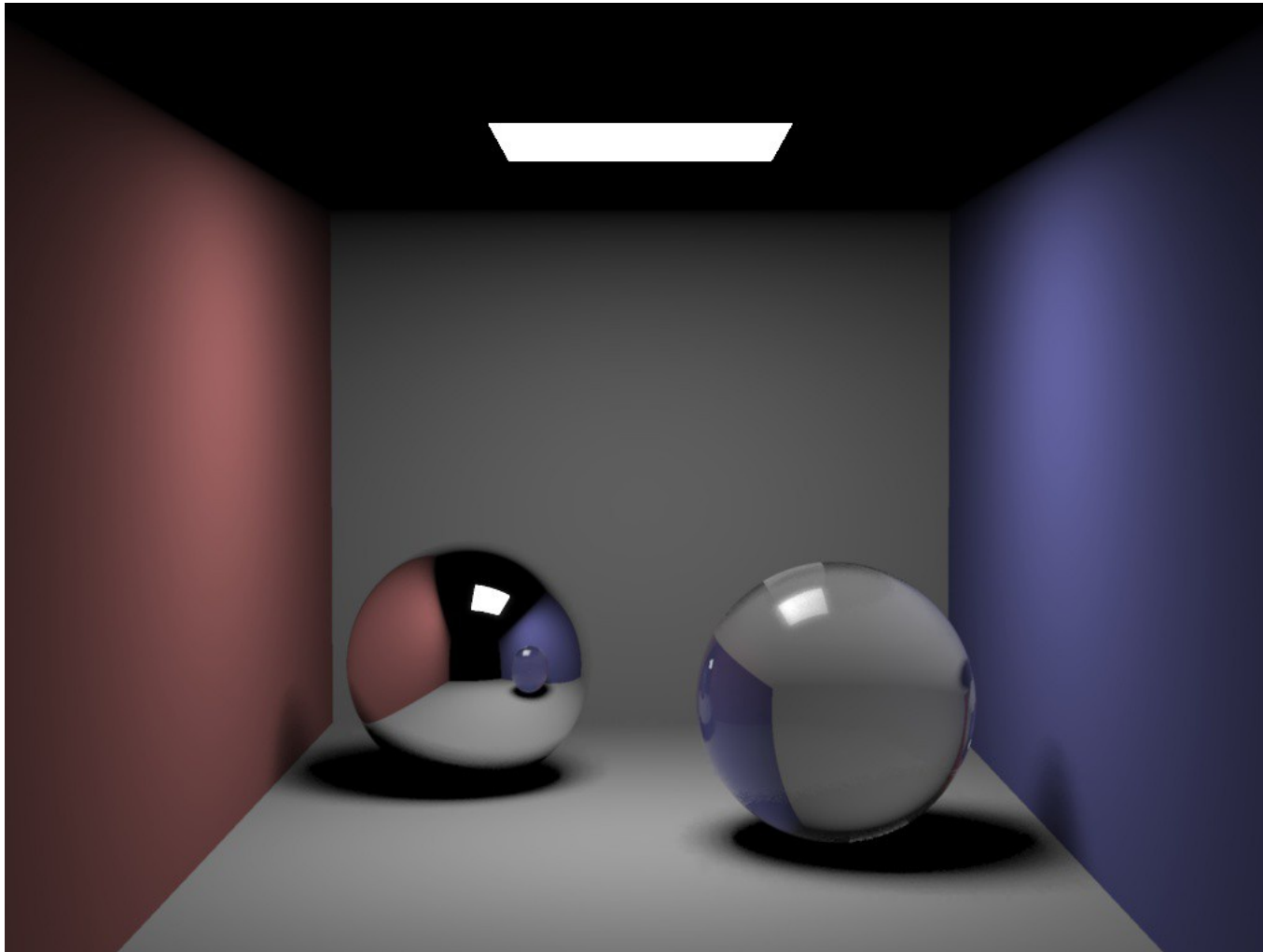


3.5 sec (dual Pentium II, 90's...)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...



Ray-Tracing with soft shadows

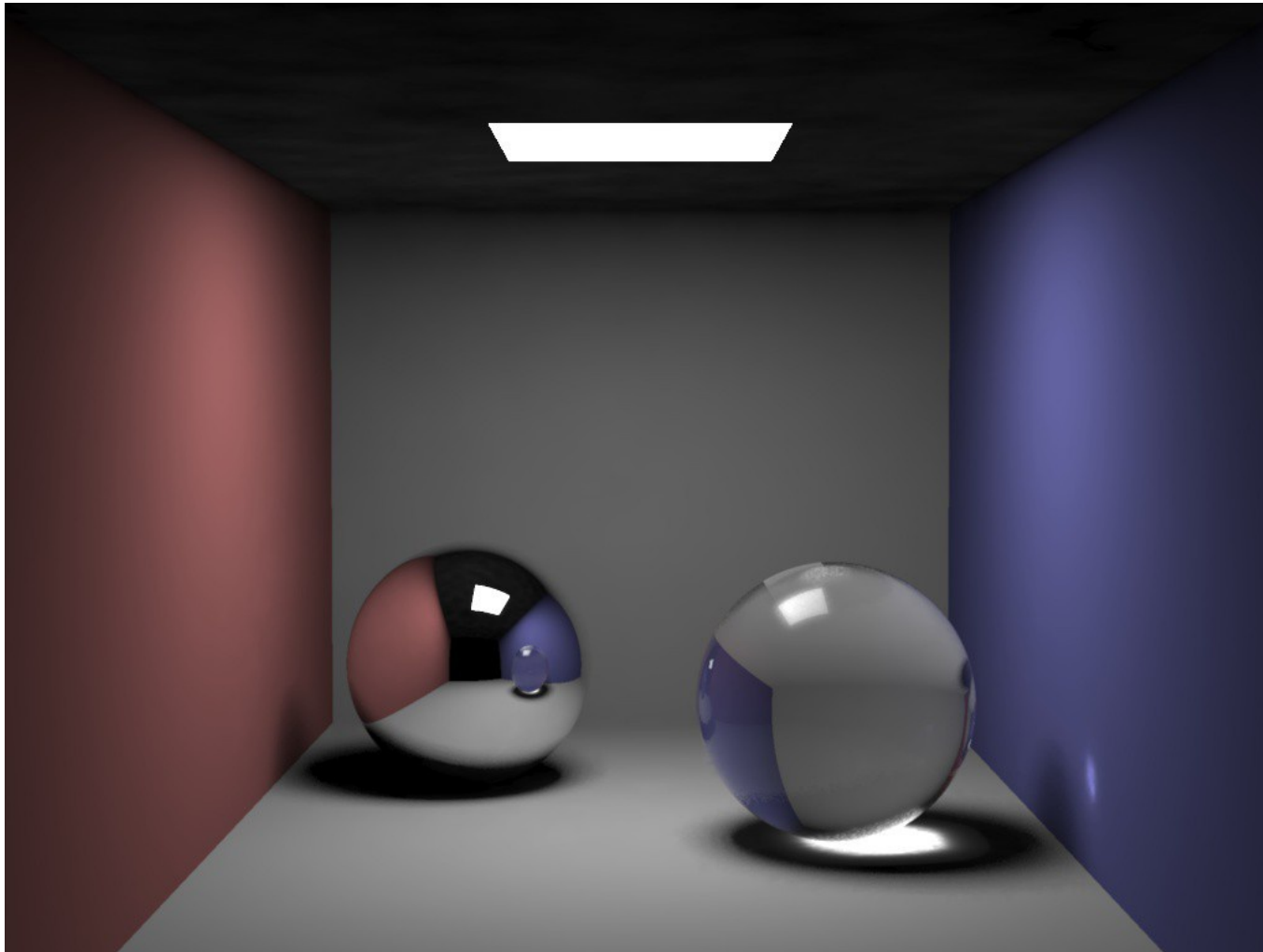


21 sec

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...



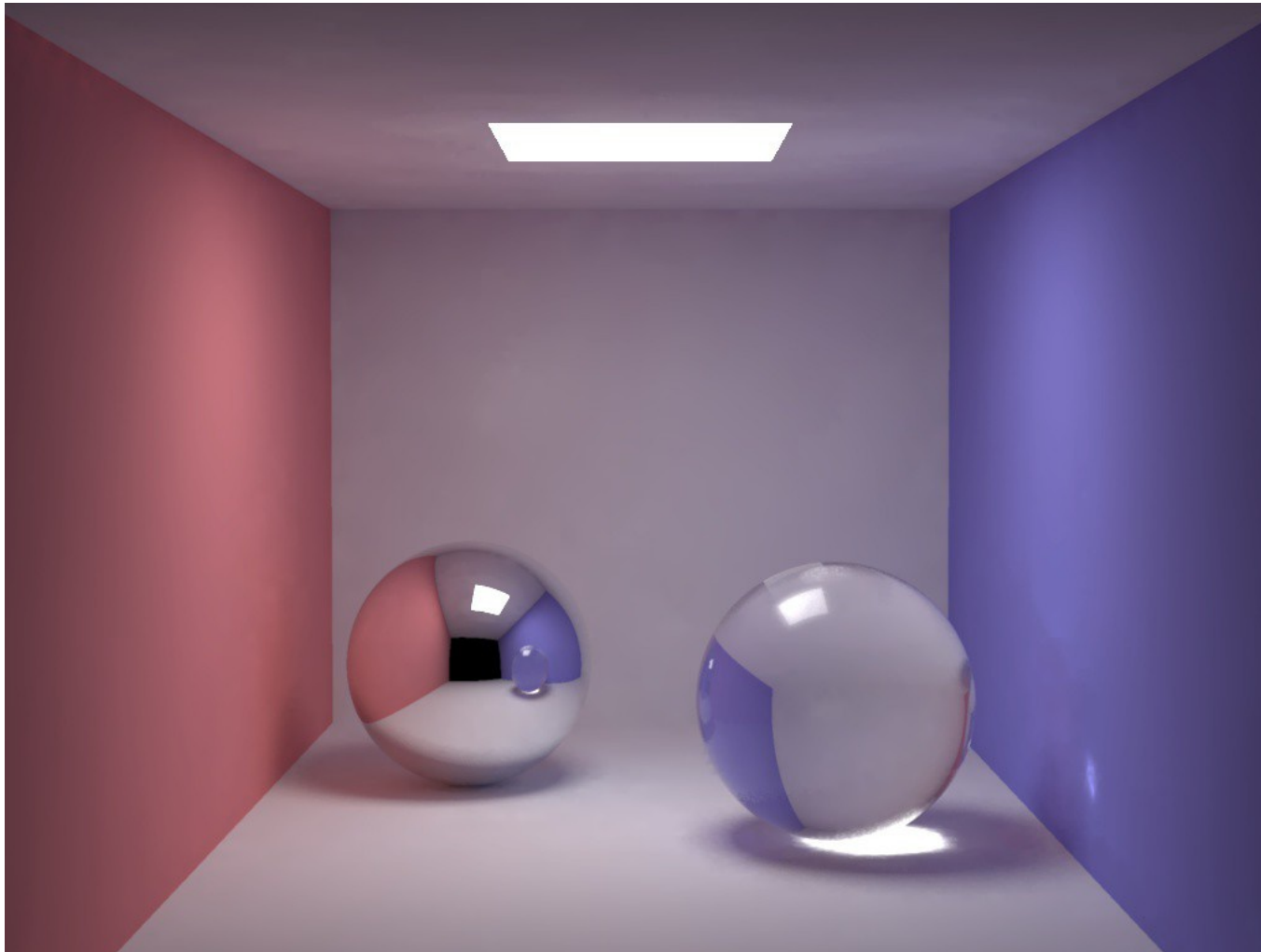
Ray-Tracing with caustics



2 + 34 sec (50k photons)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Photon-Mapping – Global illumination



4 + 66 sec (50k caustics + 200k global)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

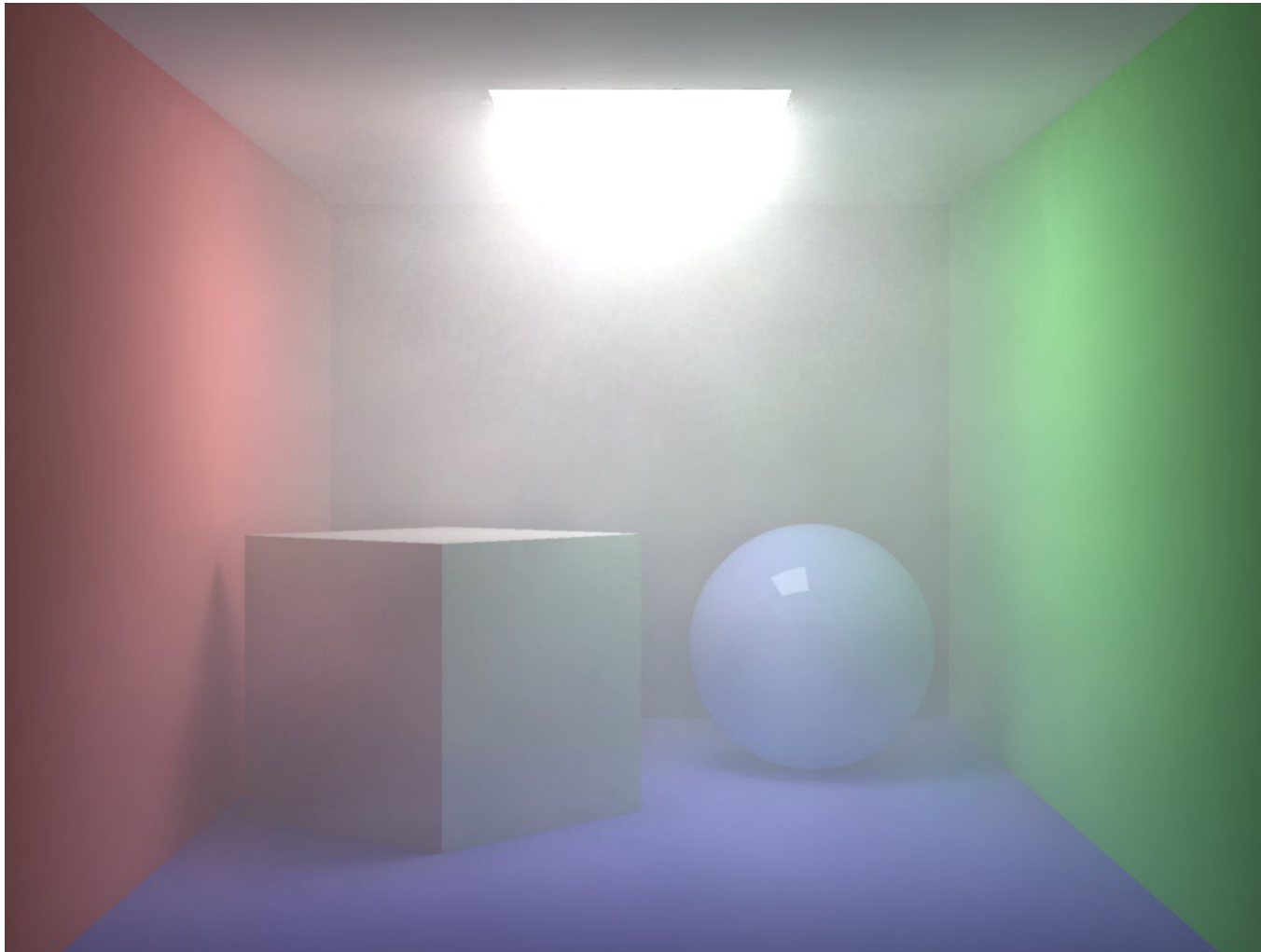
Photon-Mapping + Water



11 min (water – 500k photons)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Photon-Mapping + Fog

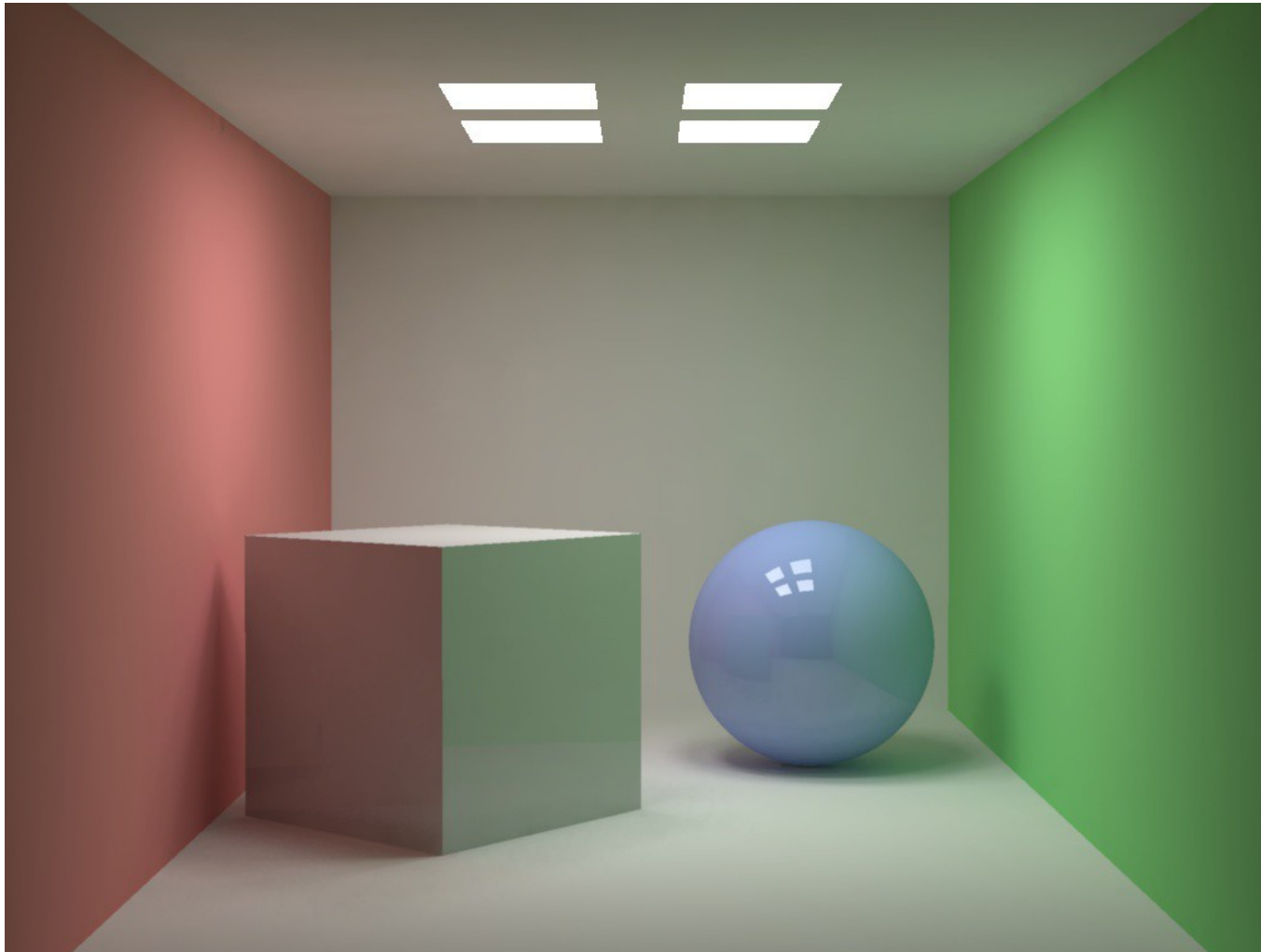


44 min (100k global + 150k volumetric)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...



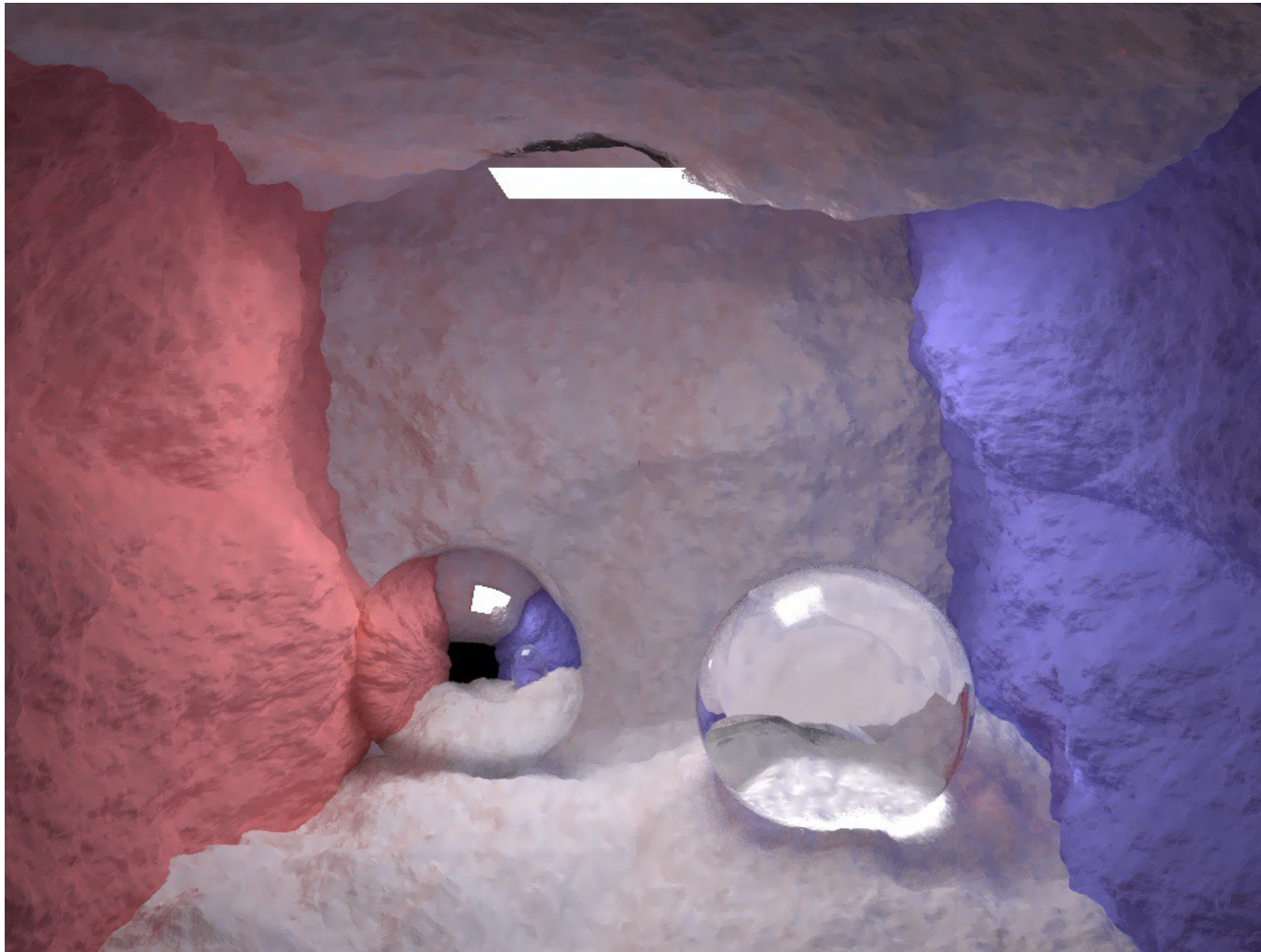
Photon-Mapping – Four light sources



19 sec (100k photons for each light)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Photon-Mapping – Fractal surfaces



14 min (50k caustics + 200k global)

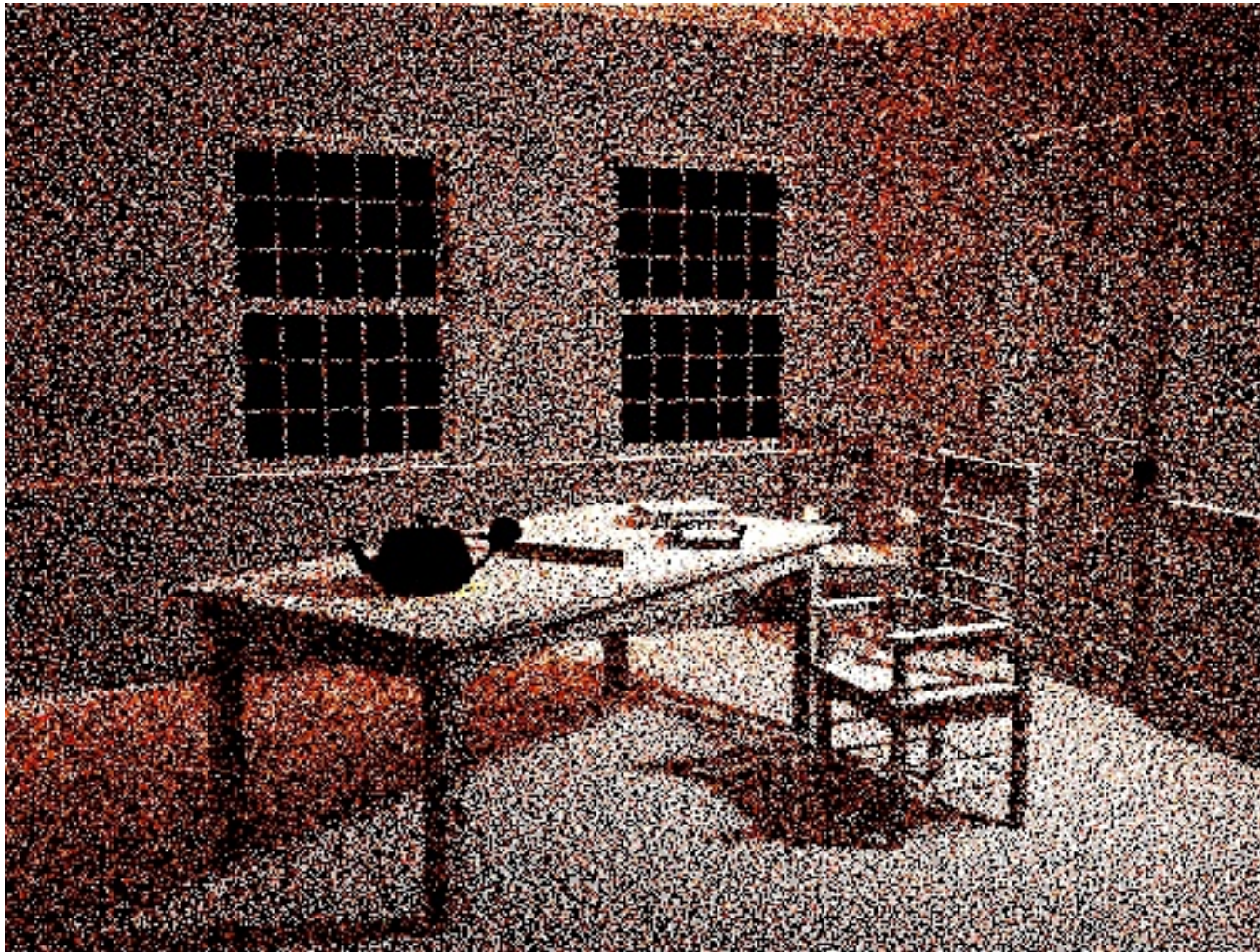
Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Example – Ray-tracing



Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Example – Photon map



Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Example – Radiance estimation (25% photons)



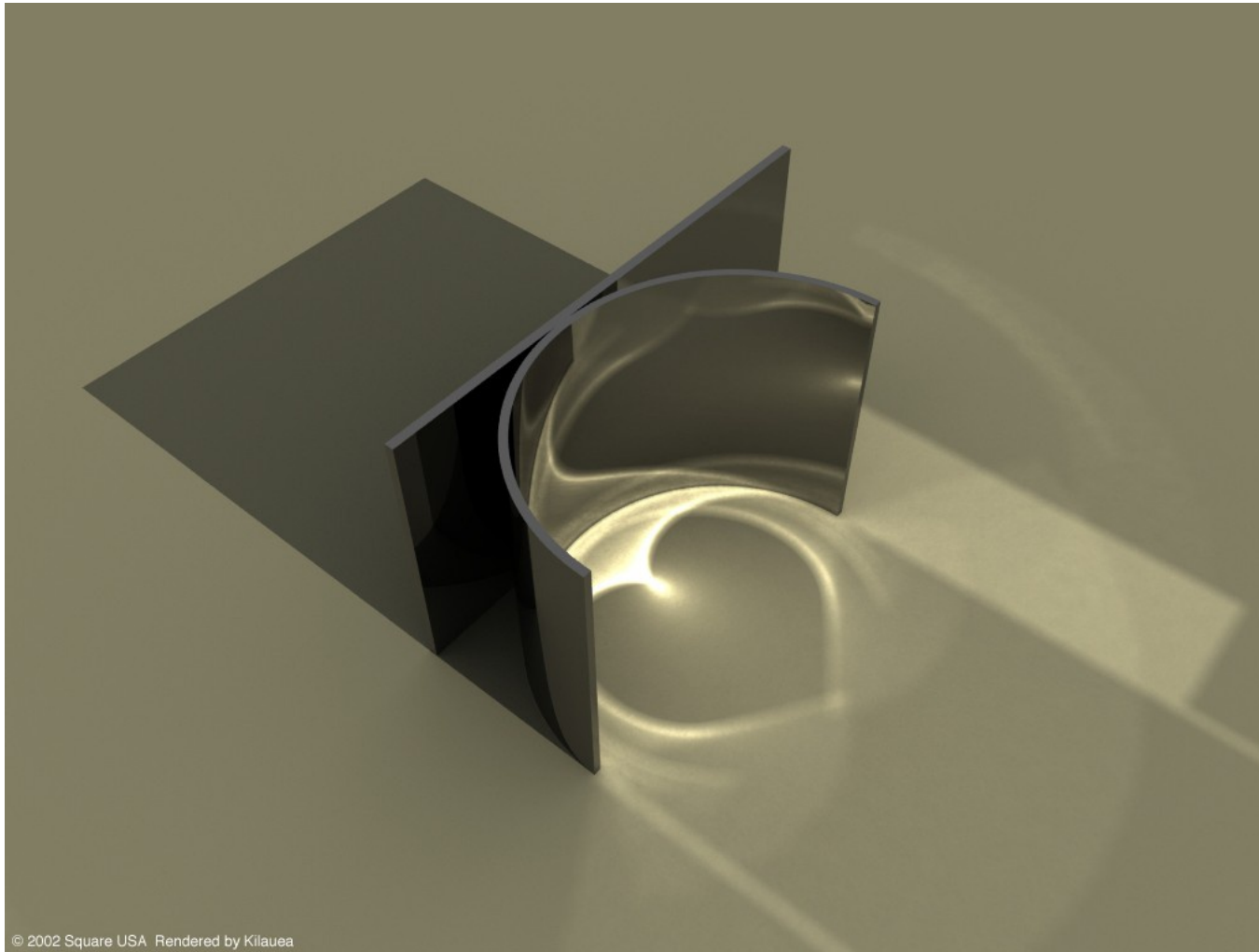
Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Example – Complete global illumination



Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Photon-Mapping – Caustics (old code)



10 min (92k caustics + 152k global) ... probably Pentium 100MHz

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

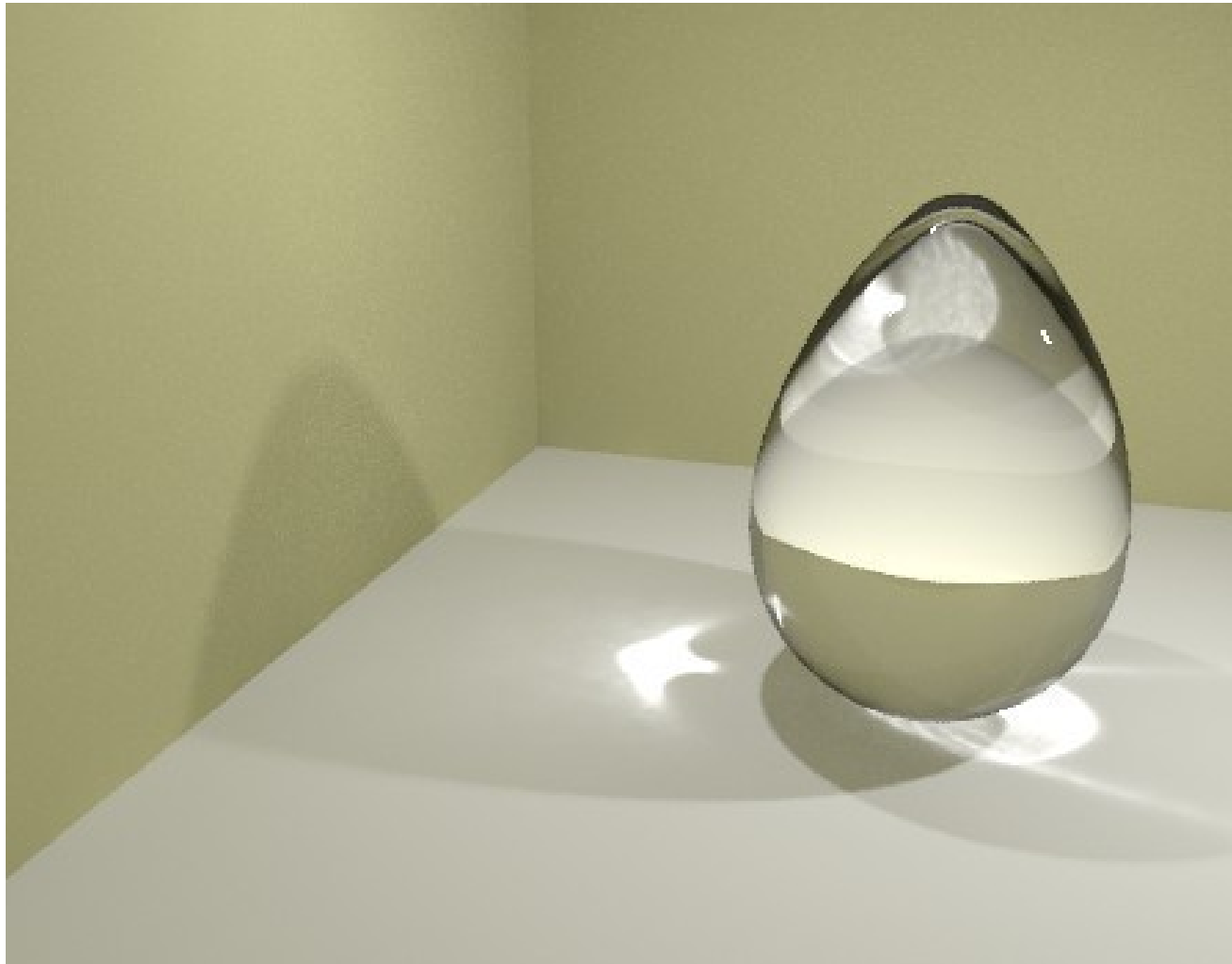
Photon-Mapping – Cognac (old code)



8.2 min (200k caustics)

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Glass egg



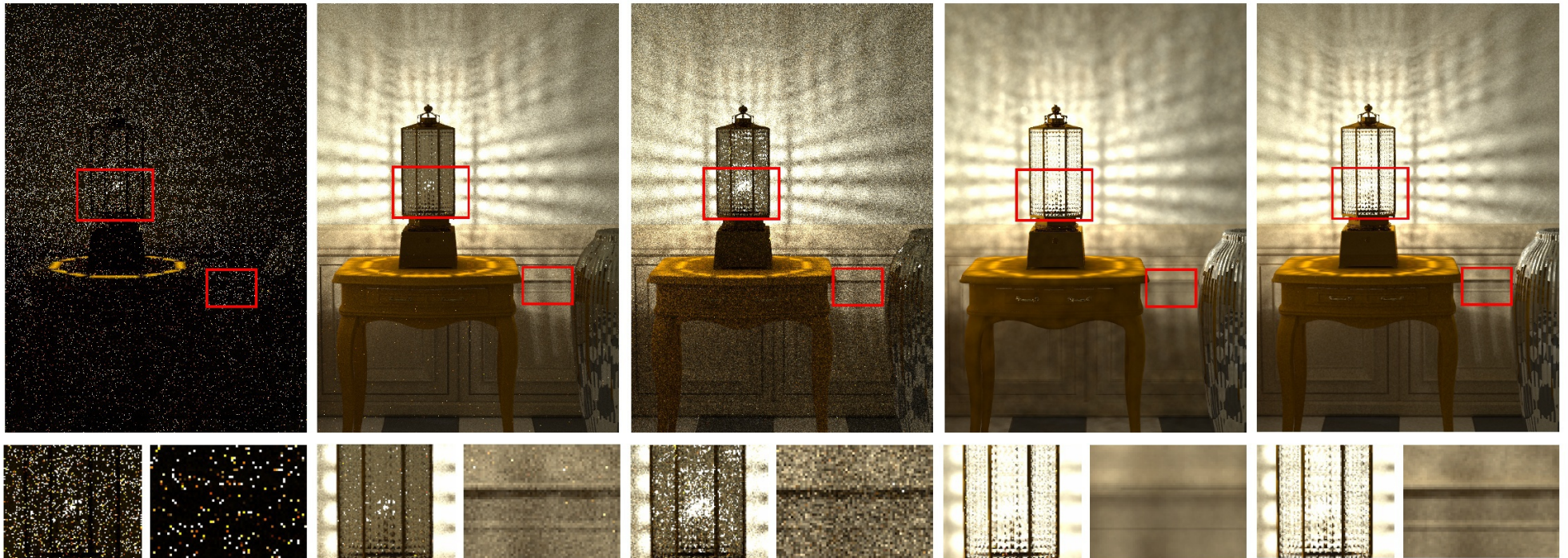
80 min

Images © Henrik Wann Jensen, James Arvo, Eric Veach, Leonidas Guibas...

Progressive Photon Mapping



© 2008, T. Hachisuka, S. Ogaki, H. Wann Jensen, SIGGRAPH Asia 2008



Path tracing

Bidirectional path tracing

Metropolis light transport

Photon mapping

Progressive photon mapping



References

Henrik Wann Jensen: *Realistic Image Synthesis Using Photon Mapping*, A K Peters, 2001

Henrik Wann Jensen et al.: *A Practical Guide to Global Illumination using Photon Mapping*, SIGGRAPH 2002 Course

Matt Pharr, Wenzel Jakob, Greg Humphreys: *Physically Based Rendering: From Theory To Implementation*, online version, 2018

Philip Dutre, Kavita Bala, Philippe Baekert: *Advanced Global Illumination*, A K Peters, 2006