Rendering Algorithms

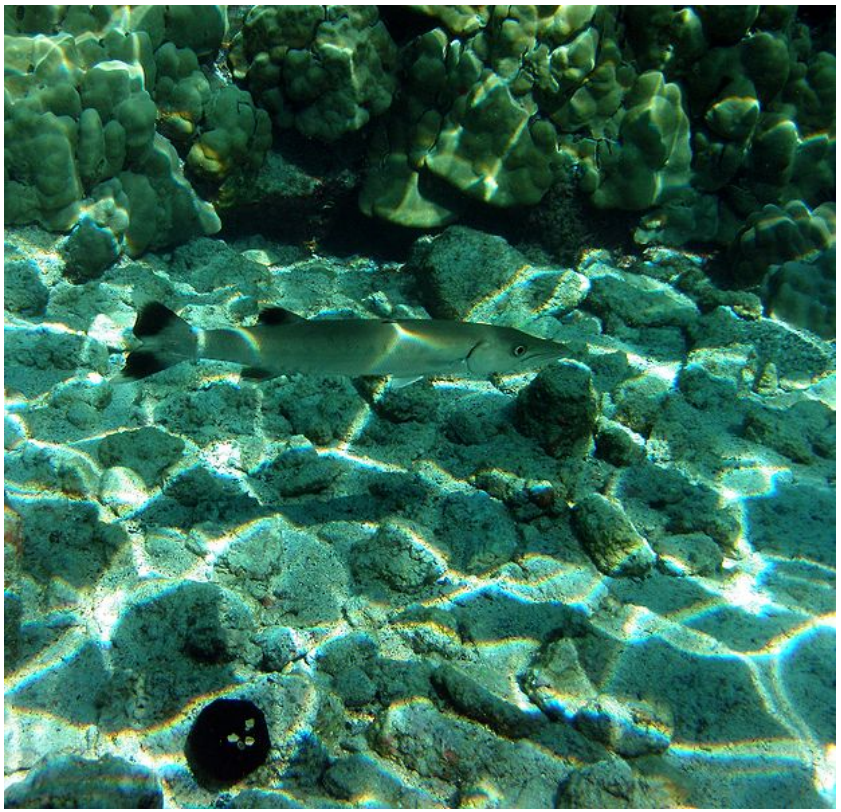
Spring 2014

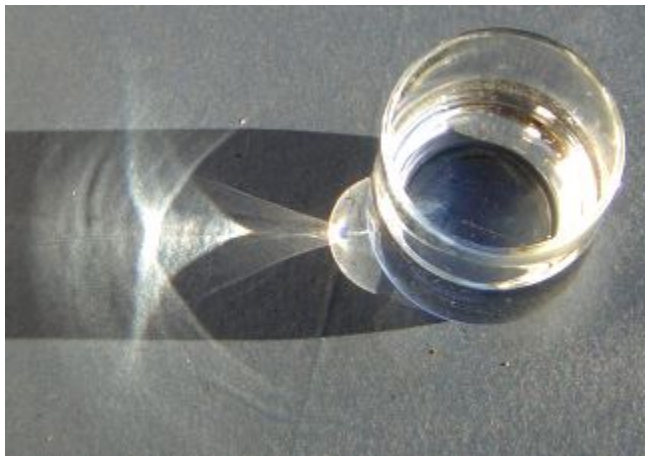
Matthias Zwicker

Universität Bern

Photon mapping

- Addresses same problem of path tracing with caustics as bidirectional path tracing





http://en.wikipedia.org/wiki/Caustic_(optics)

Photon mapping

- Very similar to bidirectional path tracing, but with a twist
  - Shoot all light subpaths first
  - Cache them using „photons“
  - Shoot eye subpaths in second step and look up photons to connect paths
- Particularly efficient for caustics

http://graphics.ucsd.edu/~henrik/images/caustics.html

Photon mapping

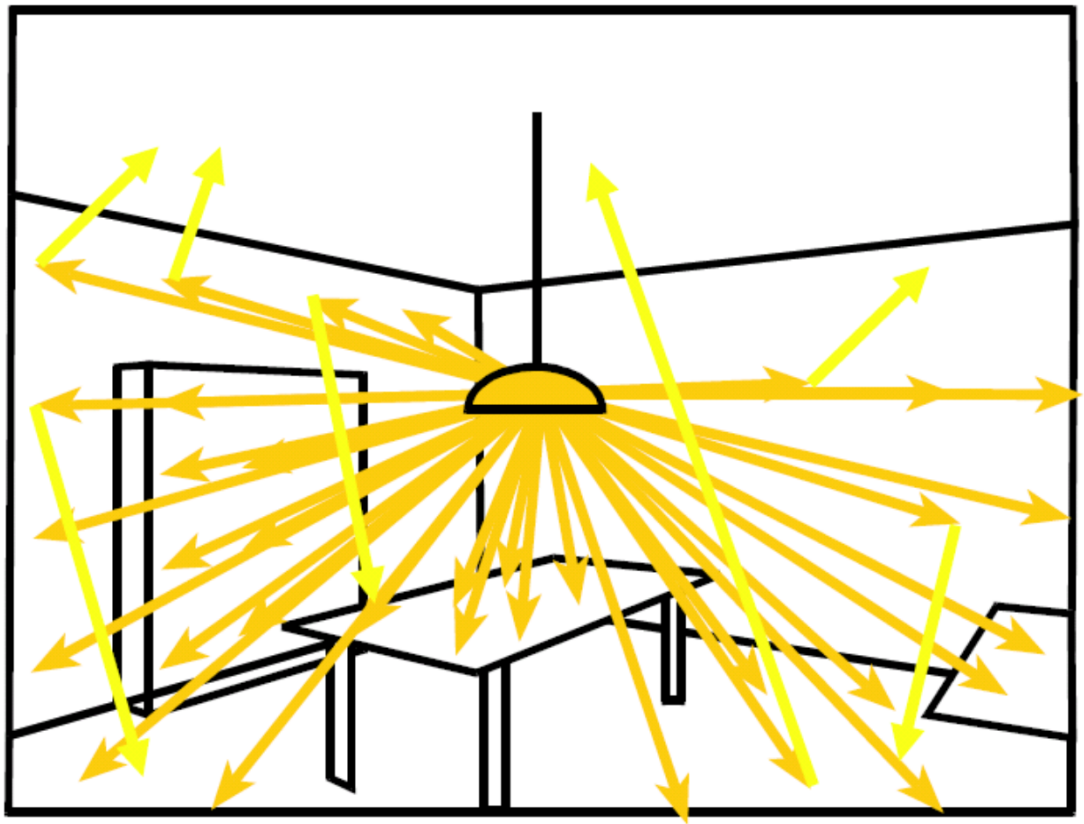
- Overview
- Emission, transport, radiance estimation
- Spatial data structure
- Effective use of photon maps

Photon mapping

1. **Light subpaths**
   Construct subpaths starting from the light sources
2. **Caching**
   Cache (store) photons distributed along light subpaths
3. **Radiance estimation**
   Connect paths from eye with photons; interpolate radiance from cached photons

Photon mapping

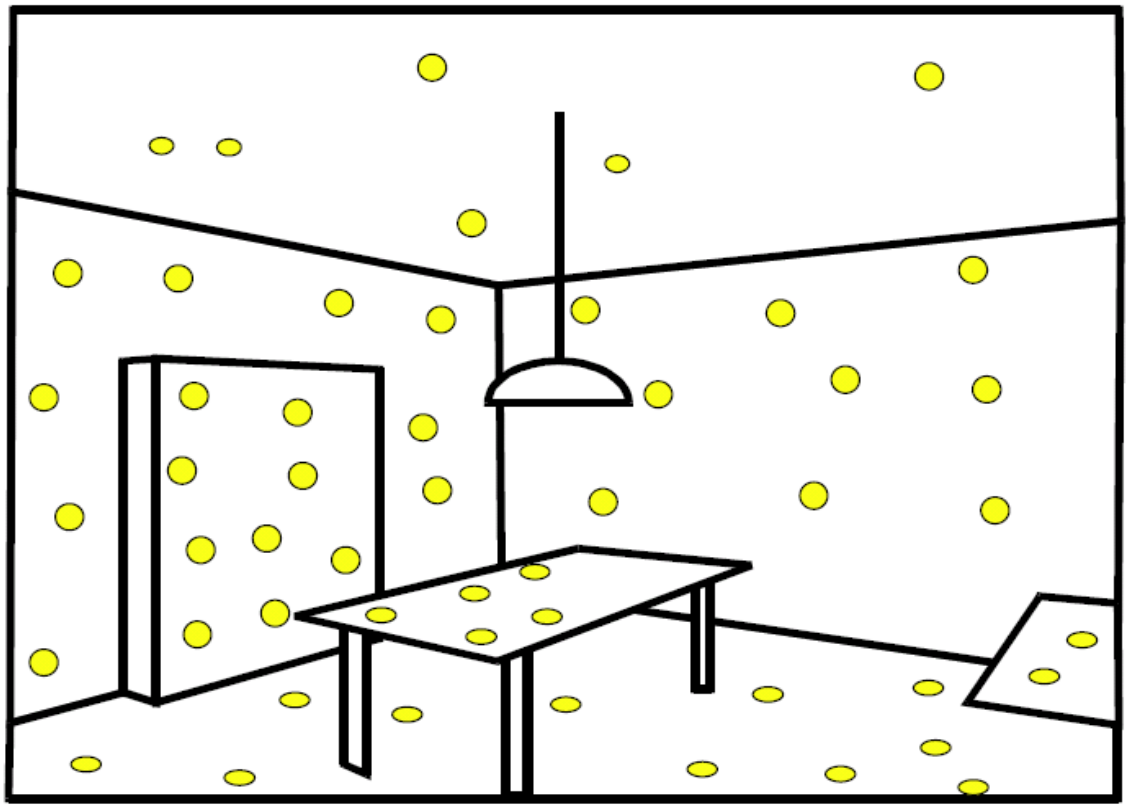
1. Light paths: photon emission & transport



[Cutler, Durand]

Photon mapping

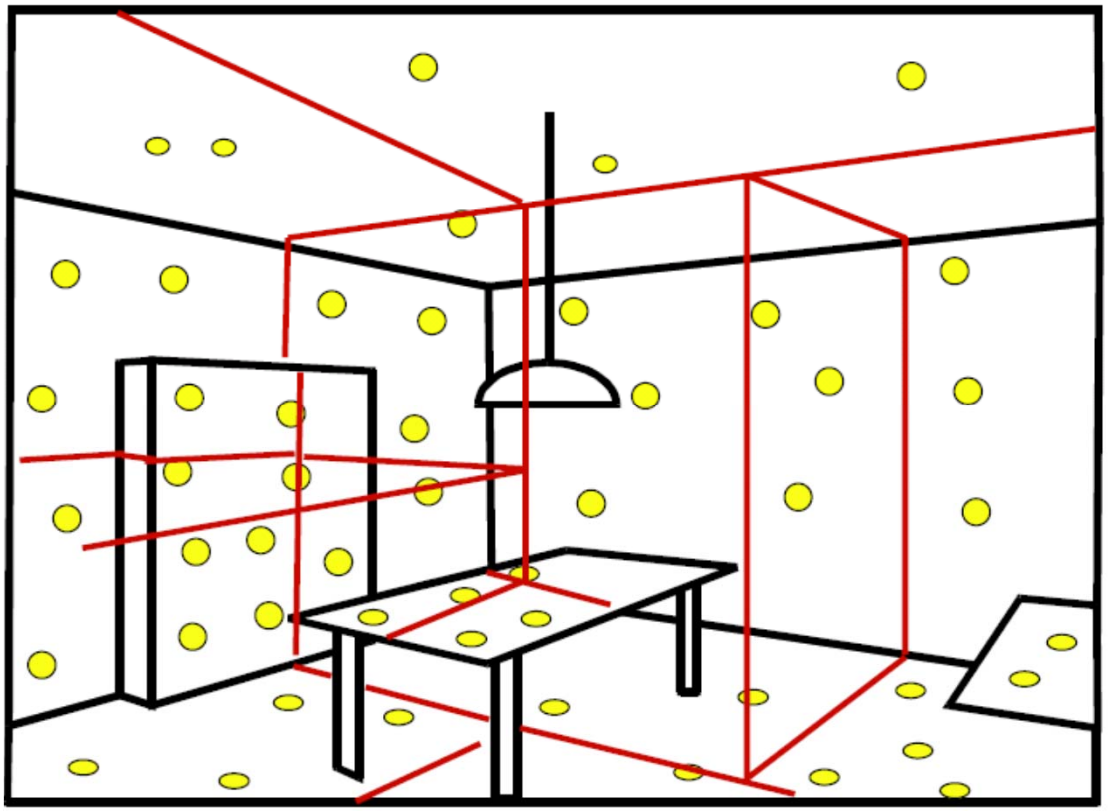
2. Caching: store photons



[Cutler, Durand]

Photon mapping

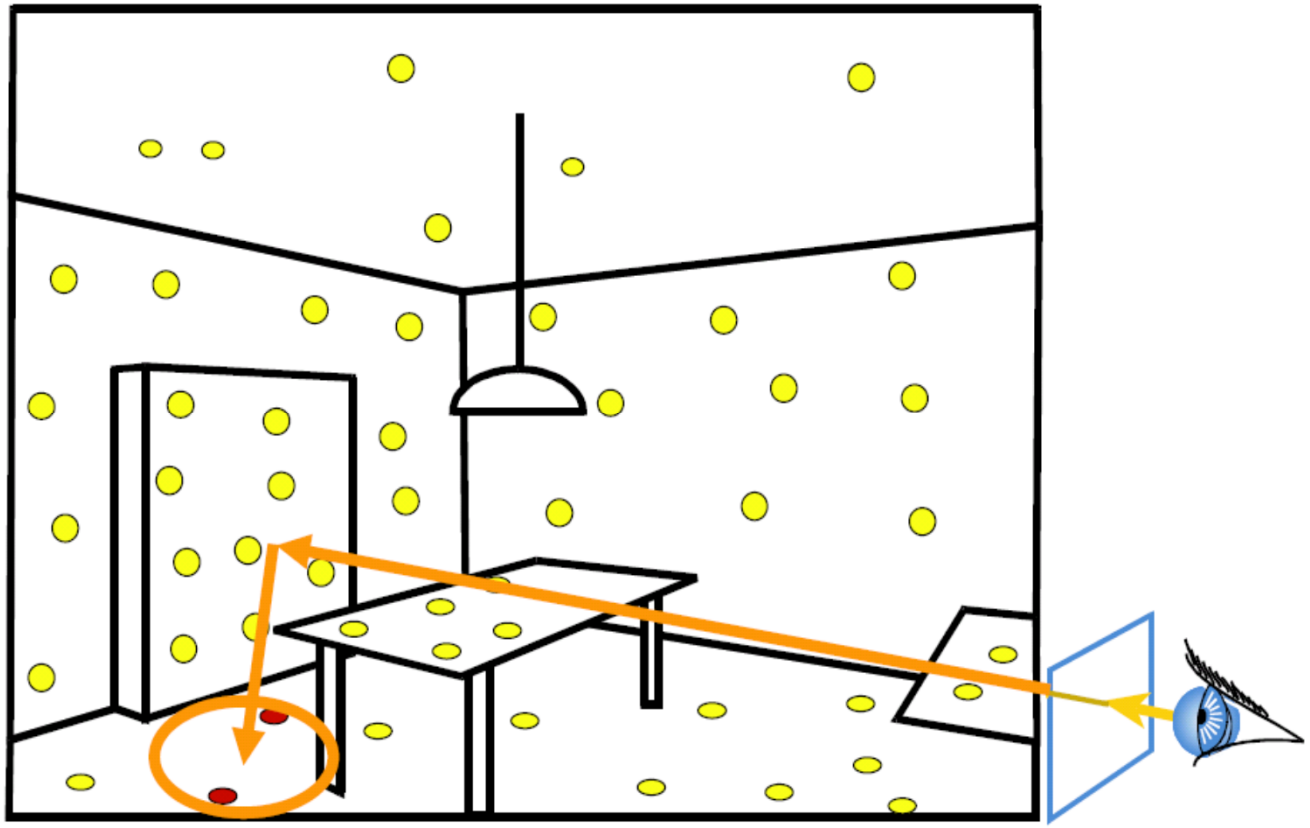
2. Caching: spatial data structure, fast access



[Cutler, Durand]

Photon mapping

3. Radiance estimation: eye paths, interpolation



[Cutler, Durand]

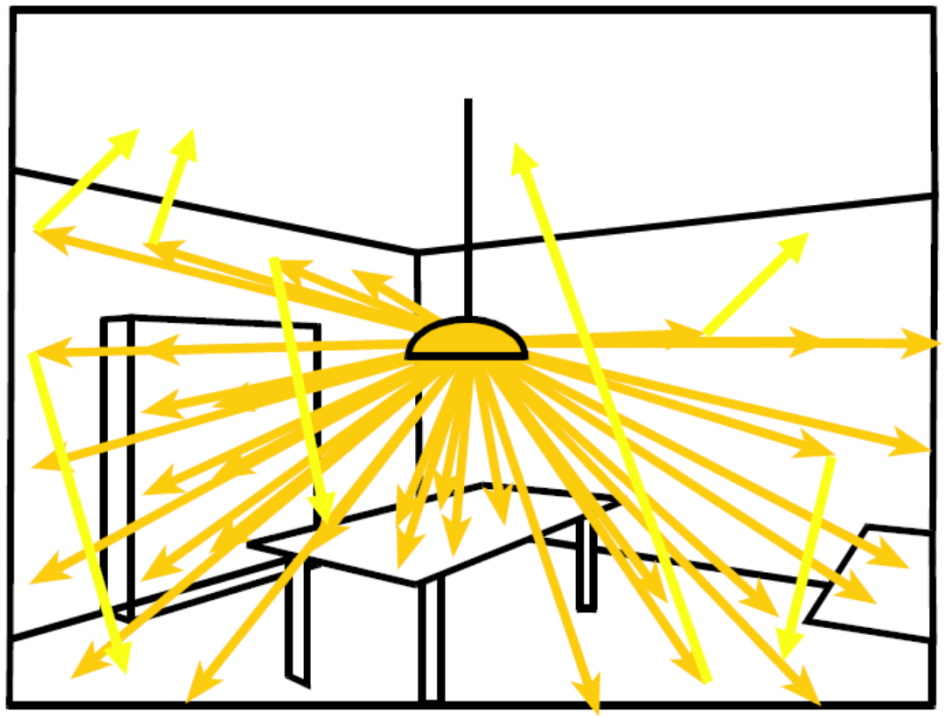
What is a “photon”?

- A photon has a
  - Location $\mathbf{x}$
  - Incident direction ω
  - Value α
- Intuition: value of photon is radiance transported along light path divided by probability for path (and some cosine factors, see later…)

```
Class Photon {
  vector3 x
  vector3 w
  vector3 alpha
}
```

Photon emission and transport

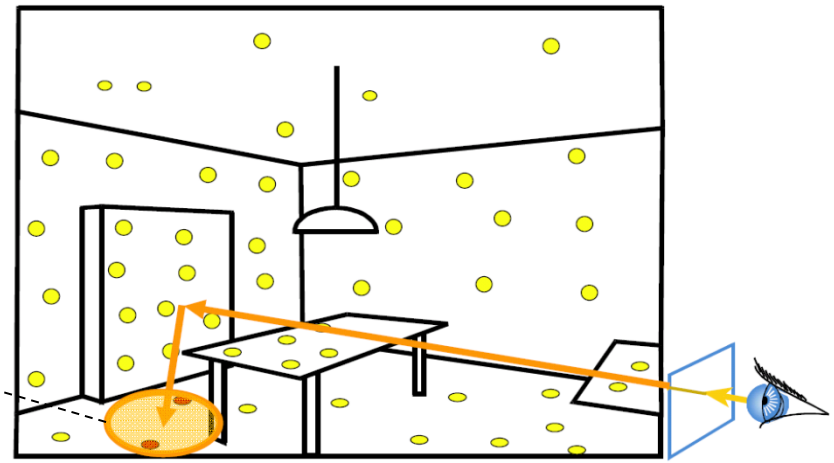
- Very much like path tracing, only starting at light sources instead of at the eye
- Same as tracing light subpath in bidirectionl path tracing



[Cutler, Durand]

Photon values α

- Goal: use photons to estimate an „average“ radiance over some region
- Formulate measurement integral on scene surfaces (instead of image plane)
- Express using integration over surface area

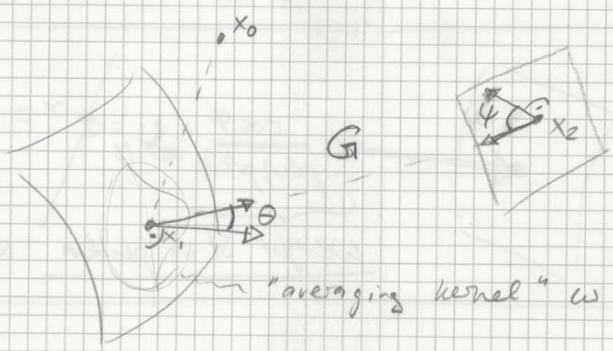


[Cutler, Durand]

Photon mapping weights

Explanation of Photon Map Weights

"Average Illumination" from area light source



"averaging kernel" ω

$$\iint W(x_1)\, f(x_2 \to x_1 \to x_0)\, L(x_1 \to x_1)\, G(x_1, x_2)\, dx_1 dx_2$$

this is called a "measurement integral"

Monte Carlo approximation:

$$\frac{1}{N} \sum_i \frac{W(x_{1,i})\, f\, L\, G}{p(x_{1,i}, x_{2,i})}$$

Sampling x_1, x_2: sample x_2 randomly on surface with some pdf $p(x_2)$; sample x_1 by shooting rays in random direction from x_2

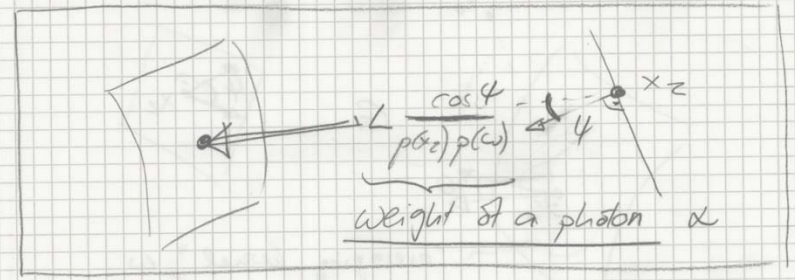
$\Rightarrow p(x_1) = p(\omega) \cdot \cos\Theta / \|x_1 - x_2\|$

$$= \frac{1}{N} \sum_i \frac{W\, f\, L\, G}{p(x_2)\, \underbrace{p(\omega) \cdot \frac{\cos\Theta}{\|x_1 - x_2\|}}_{p(x_1)}}$$

Note: $\frac{\cos\Theta}{\|x_1 - x_2\|} = \frac{G}{\cos\Psi}$

because $G = \frac{\cos\Theta \cos\Psi}{\|x_1 - x_2\|}$

$$= \frac{1}{N} \sum_i W\, f\, L\, \frac{\cos\Psi}{p(x_2)\, p(\omega)}$$



$$= \frac{1}{N} \sum_i W\, f\, \alpha_i$$

Today

Photon mapping

- Overview
- Emission, transport, radiance estimation
- Spatial data structure
- Effective use of photon maps

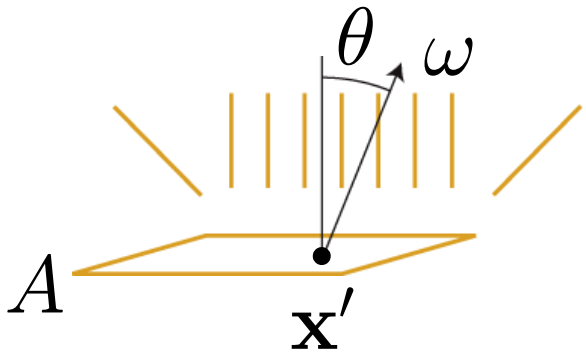
Photon emission

Rectangular diffuse light

- Power Φ, Area A
- Emitted radiance $L_e(\mathbf{x}', \omega) = \Phi/(\pi A)$
- Choose PDF $p(\mathbf{x}', \omega) = p(\mathbf{x}')p(\omega)$, e.g. $\cos(\theta)/(A\pi)$
- Initialize photon
  - In general

$$\alpha = \frac{L_e(\mathbf{x}', \omega)\cos\theta}{p(\mathbf{x}')p(\omega)}$$

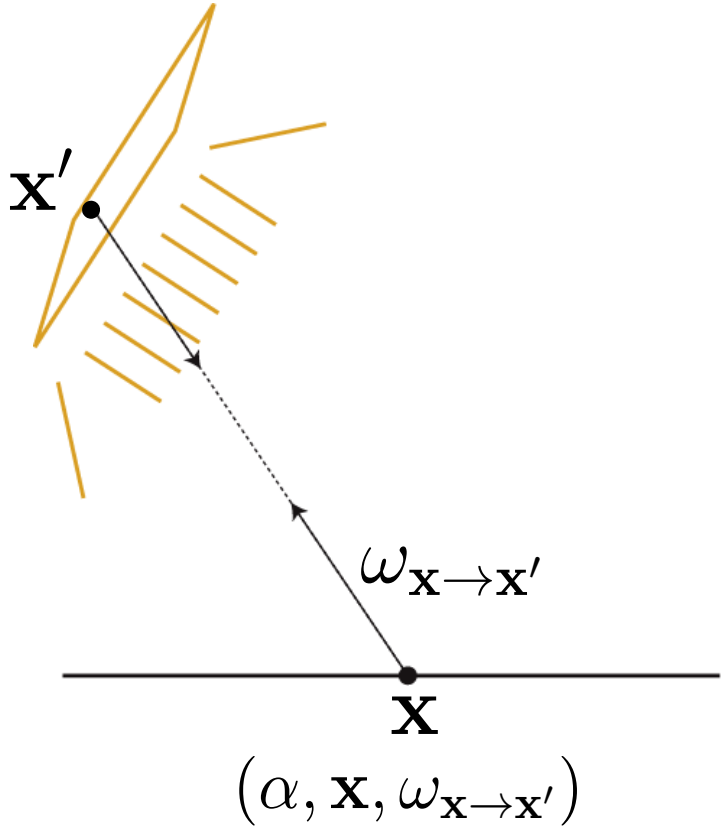
  - For $p(\mathbf{x}', \omega) = \cos(\theta)/(A\pi)$ we get $\alpha = \Phi$!



Photon transport & caching

- Store photon at first hit $\mathbf{x}$

```
photonCache->add(new Photon( a, x, w_x->x' ))
```

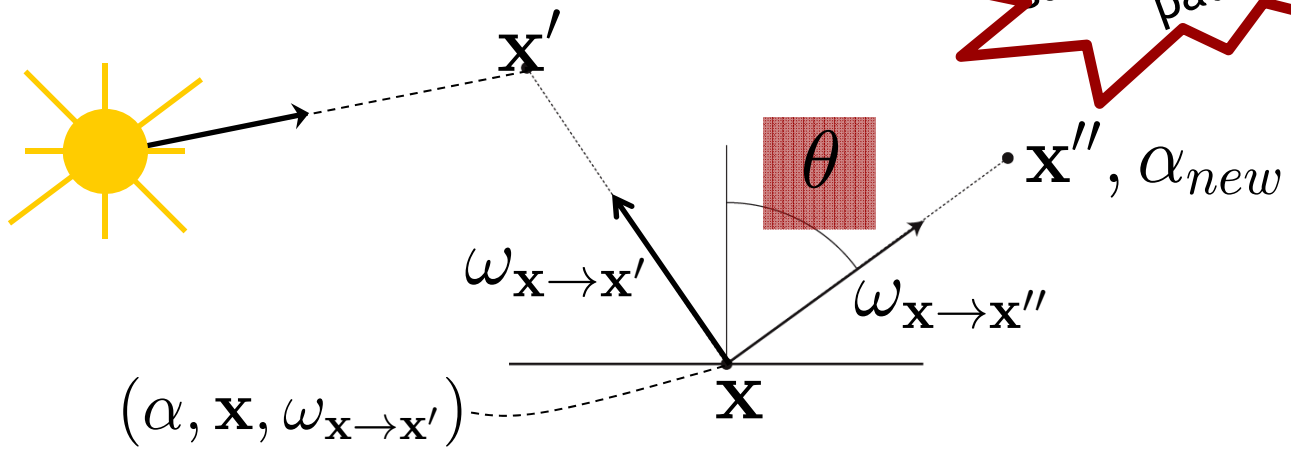


Photon transport & caching

- At each surface intersection $\mathbf{x}$
  - Store incident photon $(\alpha, \mathbf{x}, \omega_{\mathbf{x}\to\mathbf{x}'})$
  - Choose random direction $\omega_{\mathbf{x}\to\mathbf{x}''}$ with pdf $p(\omega_{\mathbf{x}\to\mathbf{x}''})$
  - Russian roulette: abort with probability q
  - Compute new value of reflected photon

$$\alpha_{new} = \alpha \frac{1}{1-q} \frac{f(\mathbf{x}'\to\mathbf{x}\to\mathbf{x}'')\cos\theta}{p(\omega_{\mathbf{x}\to\mathbf{x}''})}$$

Same as a in light subpath in bidirectional path tracing!



Choosing random directions

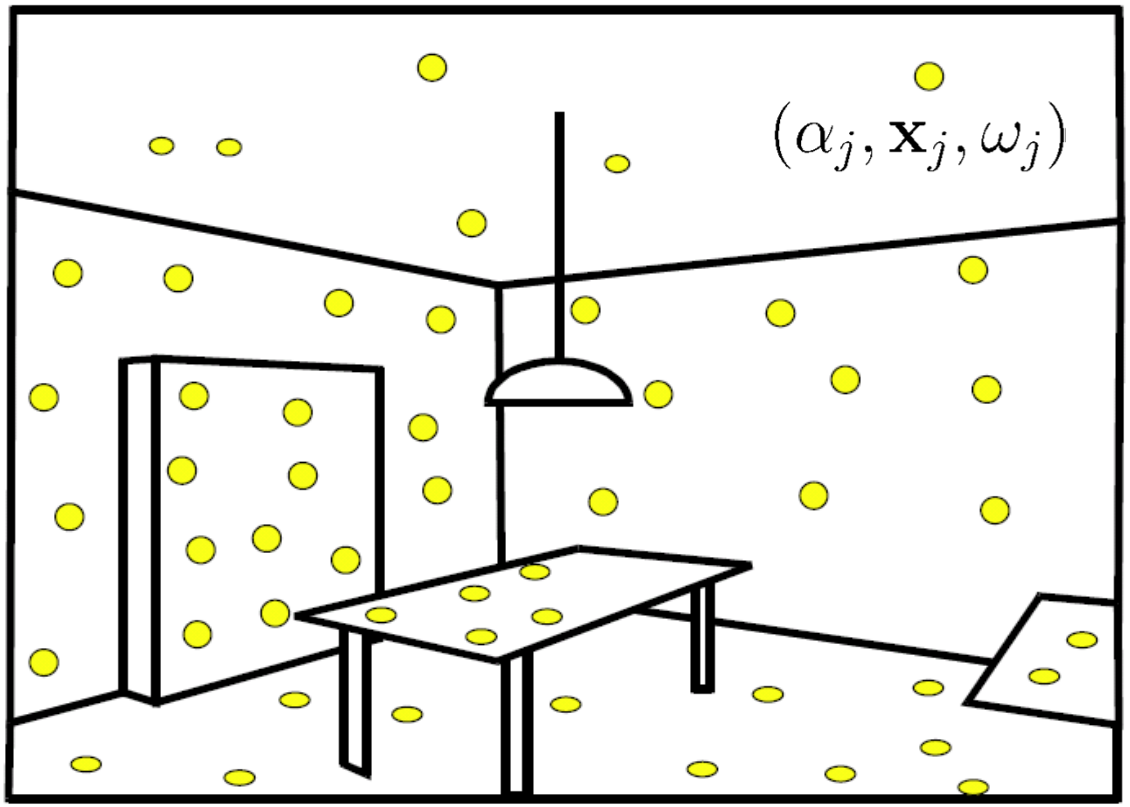
- Cosine weighted $p(\omega_{\mathbf{x}\to\mathbf{x}''}) = \cos(\theta)/\pi$
  - Cosine in pdf cancels out with $\cos\theta'$!
  - Don't forget factor $1/\pi$ in pdf

$$\alpha_{new} = \alpha \frac{1}{1-q} \frac{f(\mathbf{x}'\to\mathbf{x}\to\mathbf{x}'')\cos\theta}{p(\omega_{\mathbf{x}\to\mathbf{x}''})}$$

- Improvement: Importance sampling the BRDF
  ... as in path tracing

Photon caching

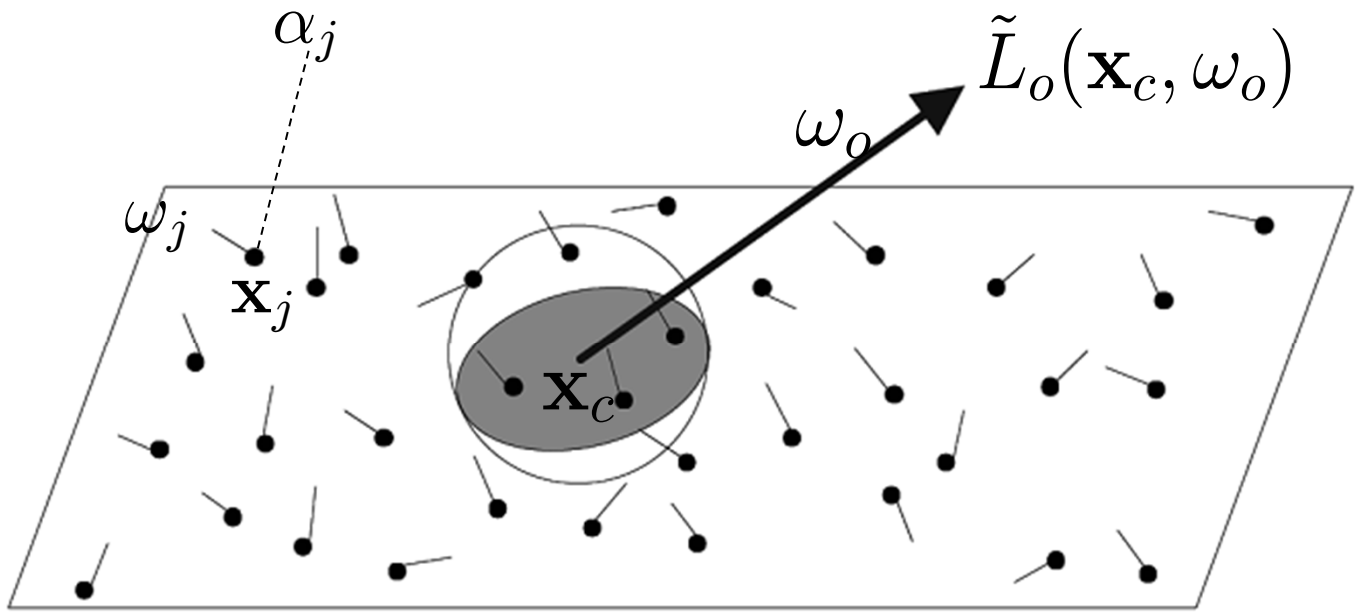
- Photon j stores values $(\alpha_j, \mathbf{x}_j, \omega_j)$



[Cutler, Durand]

Radiance estimation

- Taking radiance measurements based on stored photons
- Use the “average” incident radiance to estimate reflected radiance



Radiance estimation

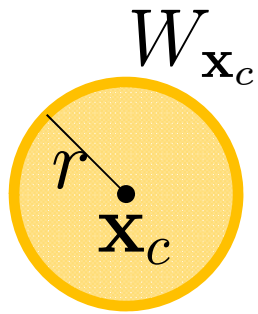
- Measurement equation with "averaging kernel" $W_{\mathbf{x}_c}$ centered at $\mathbf{x}_c$

$$\tilde{L}_o(\mathbf{x}_c, \omega_o) \approx \frac{1}{N} \sum_{j=1}^{N} W_{\mathbf{x}_c}(\mathbf{x}_j) f(\mathbf{x}_j, \omega_j, \omega_o) \alpha_j$$

- Photons
  - Positions $\mathbf{x}_j$
  - Incident directions ω_j
  - Values α_j
- Total number of emitted photons N

Radiance estimation

- Simplest kernel: constant



$$W_{\mathbf{x}_c}(\mathbf{x}_j) = 1/(\pi r^2)$$

for $\|\mathbf{x} - \mathbf{x}_c\| < r$, otherwise 0

- Kernel has unit integral!
- The estimate of the measurement is

$$\tilde{L}_o(\mathbf{x}_c, \omega_o) \approx \frac{1}{N\pi r^2} \sum_j f(\mathbf{x}_j, \omega_j, \omega_o)\alpha_j$$

where $\|\mathbf{x}_j - \mathbf{x}_c\| < r$

Radiance estimation

- How to pick the radius r?

1. Constant
   - User specified parameter
   - Don‘t know how many photons are within radius
2. Use distance to k-nearest neighbor
   - Guarantees that there are k photos in the measurement

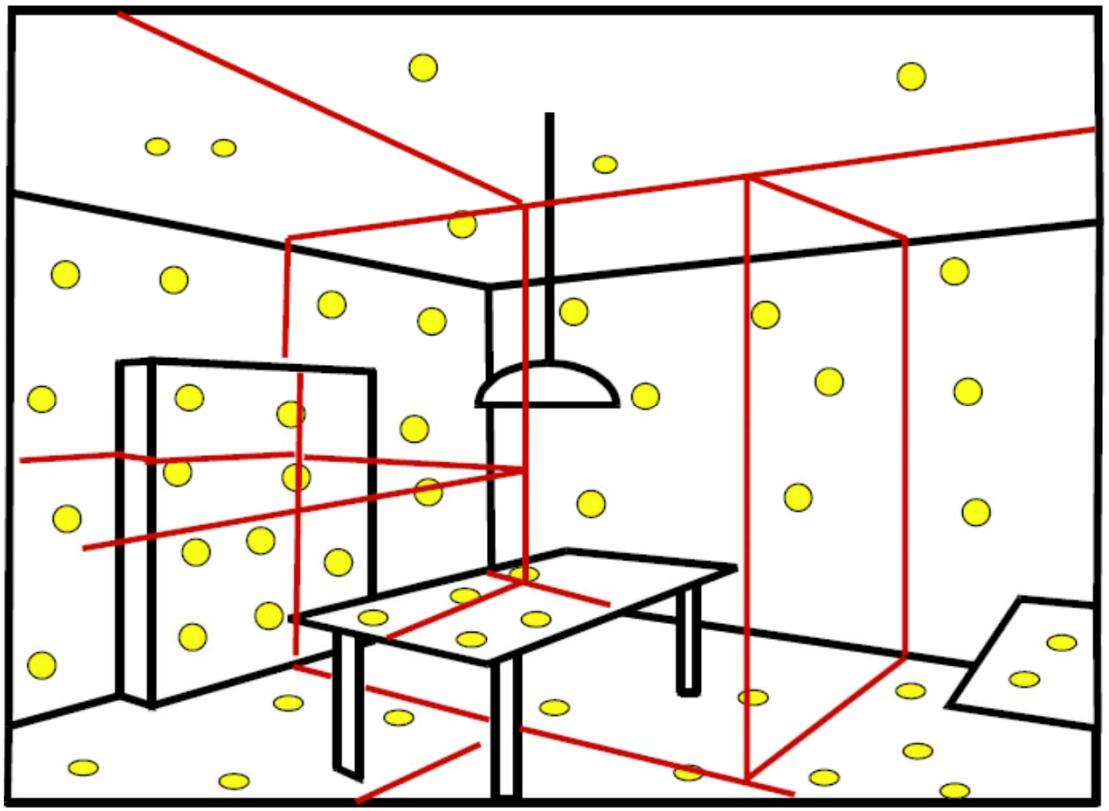
- Challenge: how to retrieve neighboring photons efficiently from photon cache?

Today

Photon mapping

- Overview
- Emission, transport, radiance estimation
- Spatial data structure
- Effective use of photon maps

Kd-tree for fast photon access



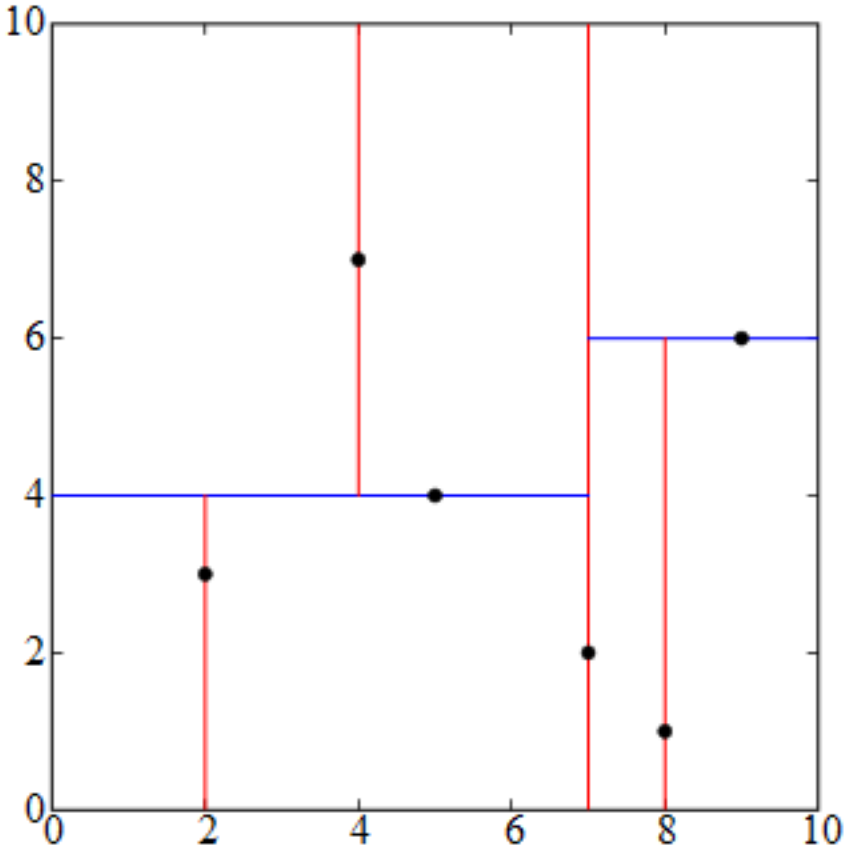
[Cutler, Durand]

Kd-tree construction

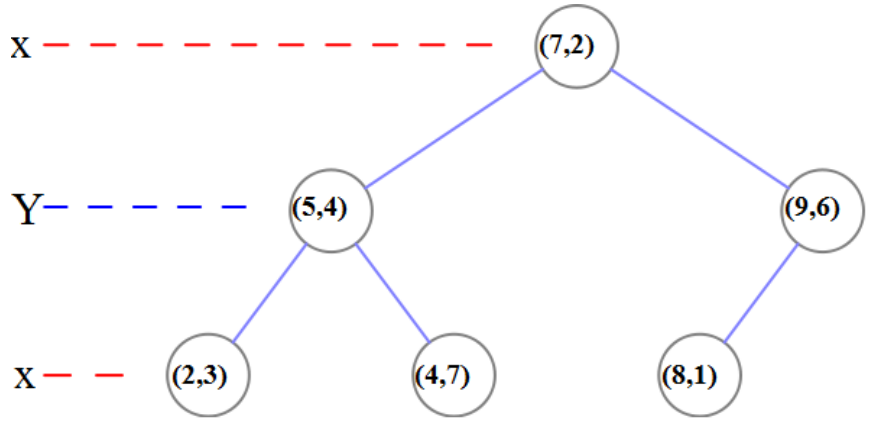
- Many options, here balanced kd-tree

```
node constructKdTree (set of points, axis)
  if set of points is empty return null;
  else {
    select median point along axis
    split set of points into set below and above median
    create node with location of median point
    node.left = kdtree(points below median, cycle(axis))
    node.right = kdtree(points above median, cycle(axis))
    return node
  }
}
```

Kd-tree construction



Set of points in 2D



Balanced kd-tree

Nearest neighbor

- Input: query point
- Output: point closest to query point
- Algorithm, see also http://en.wikipedia.org/wiki/Kd-tree
  - Traverse tree down to leaf, which contains query point
  - Leaf is candidate for nearest neighbor
  - Traverse tree back to root, at each node
    - If node is closer than previous candidate, node is new candidate
    - If nodes in other subtree of current node are potentially closer to query point than current node, traverse down subtree

K-nearest neighbors

- Maintaining k current closest points instead of just one
- Eliminate subtrees only if they can't have points closer than any of the k current bests

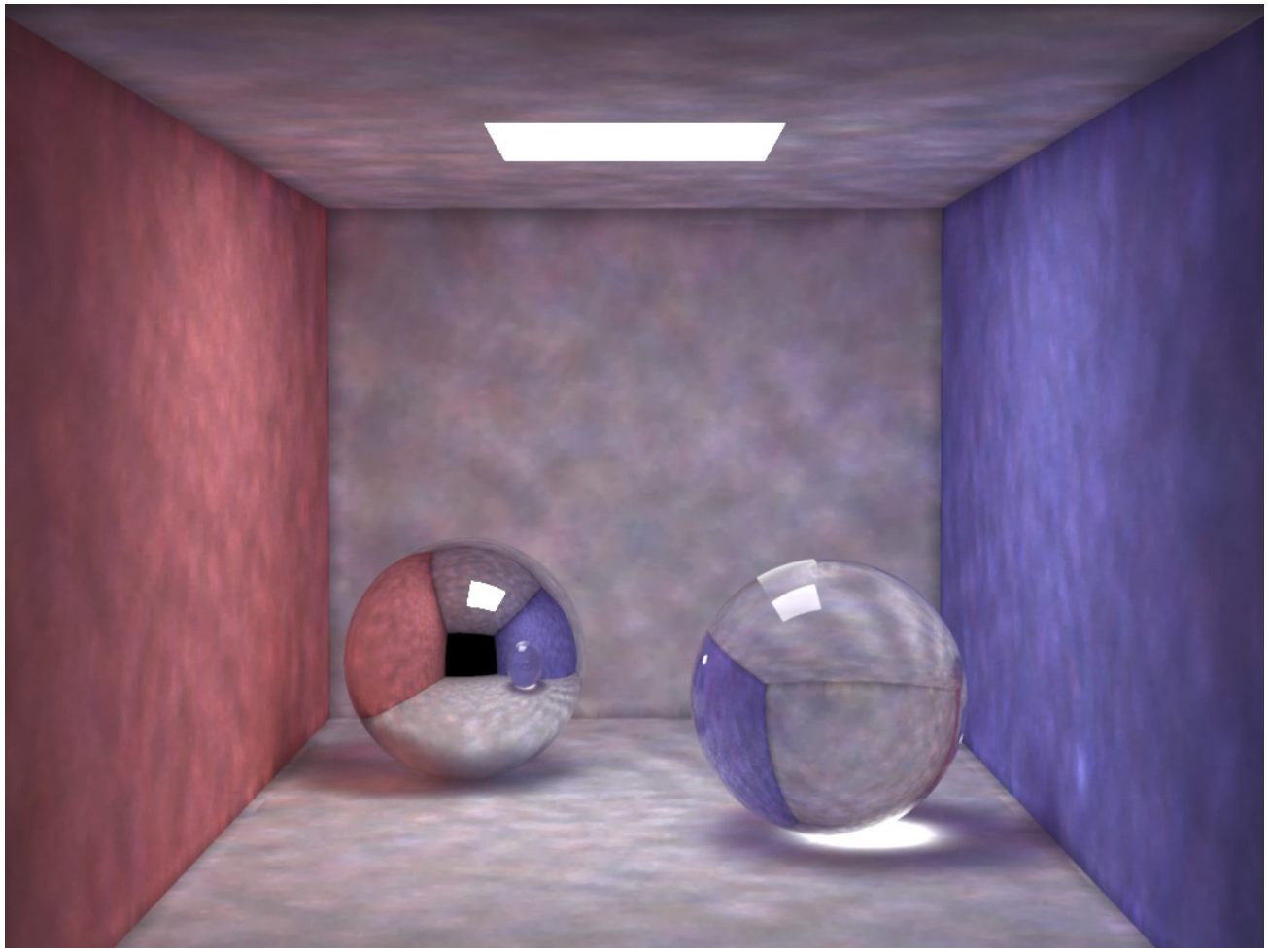
Today

Photon mapping

- Overview
- Emission, transport, radiance estimation
- Spatial data structure
- Effective use of photon maps

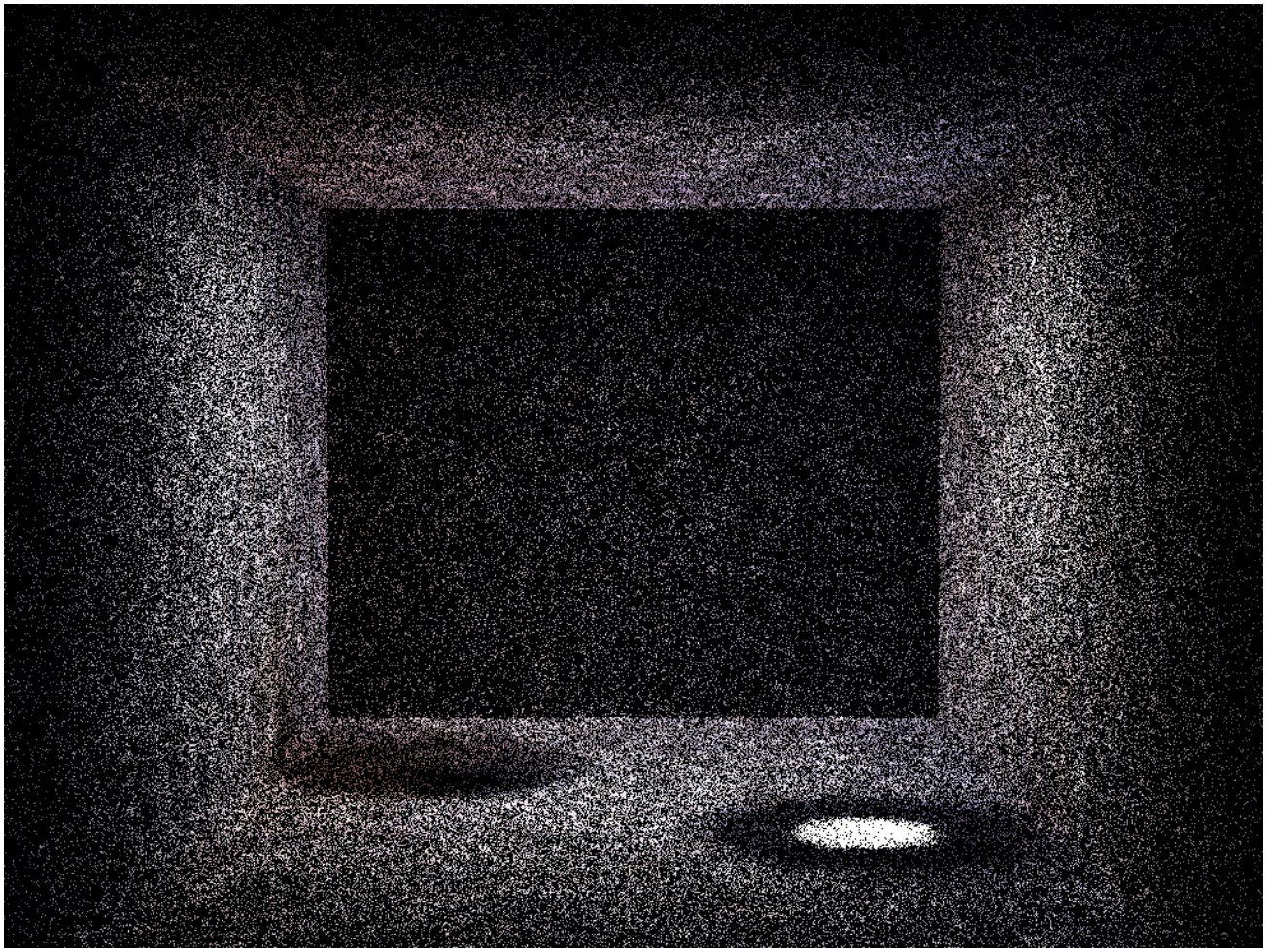
Global illumination

100000 photons, 50 photons in radiance estimate



[Wann Jensen]

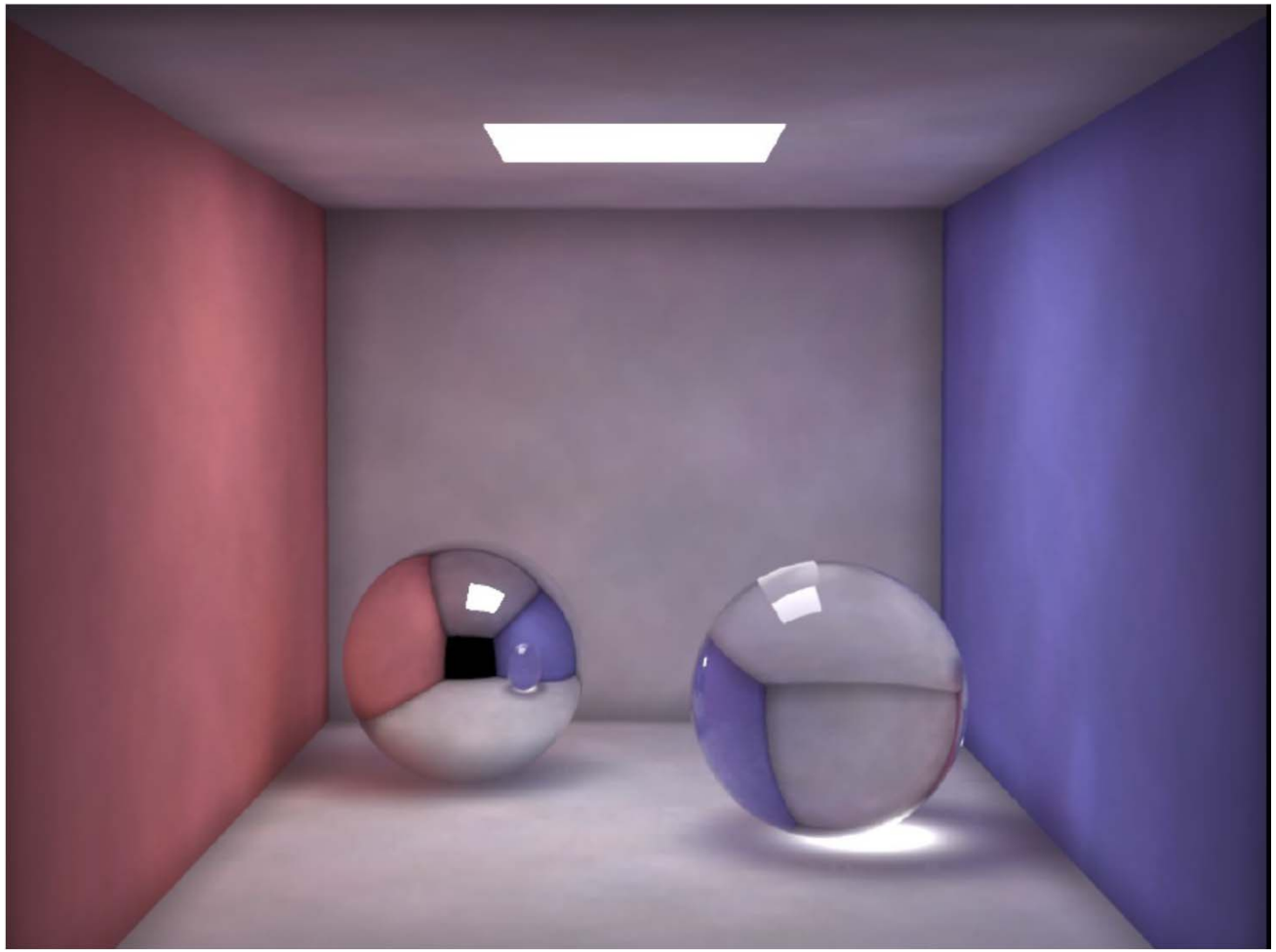
Visualization of the photons



[Wann Jensen]

Global illumination

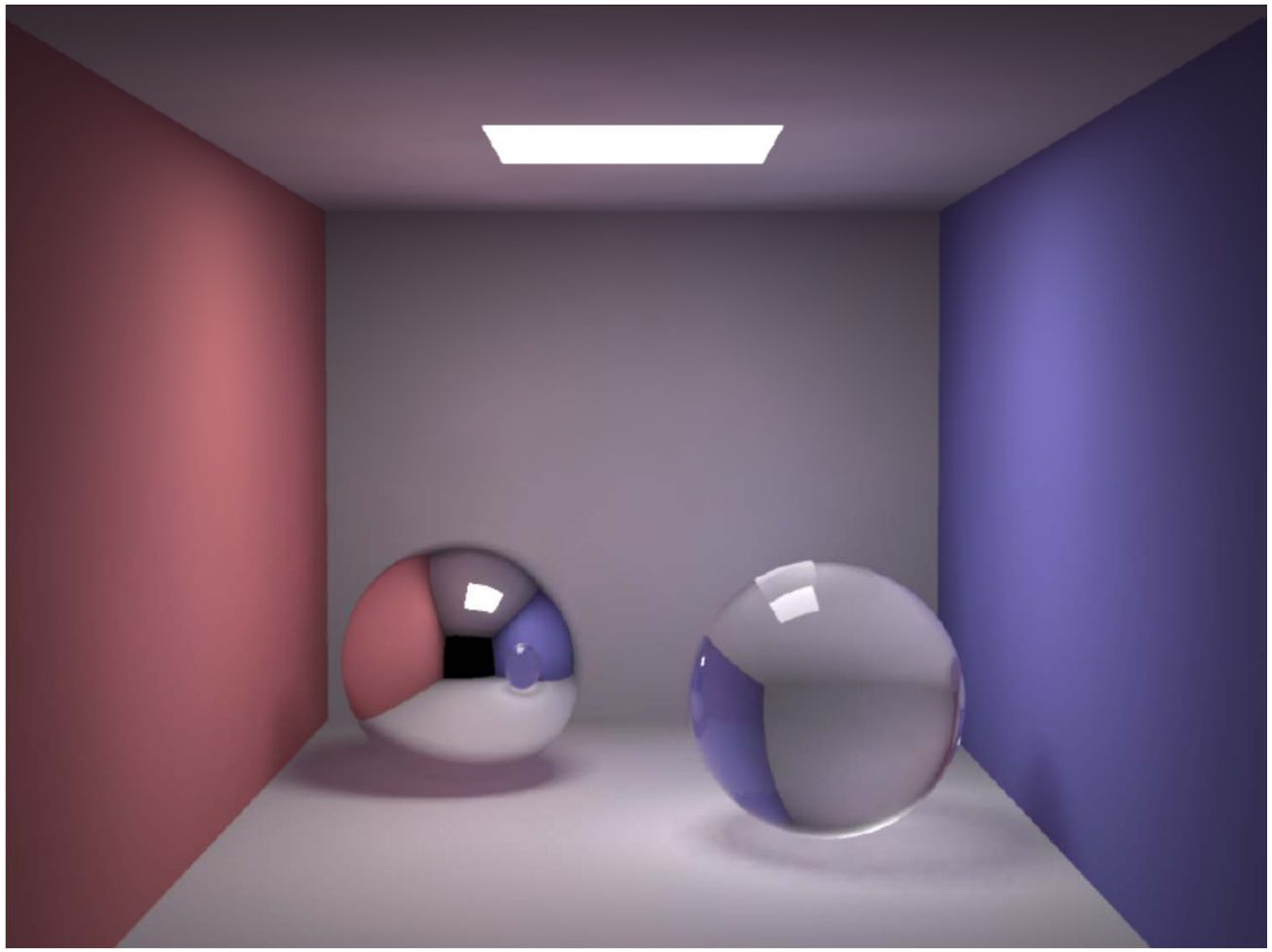
500000 photons, 500 photons in radiance estimate



[Wann Jensen]

Photons for indirect illumination

10000 photons, 500 photons in radiance estimate



[Wann Jensen]

Caustics

Photon mapping algorithm

Recap

1. Emit and transport photons
2. Build data structure for fast access
3. Rendering: use stored photons to estimate radiance

Observation

- Direct illumination requires lots of photons
- Caustics work very well

Splitting up the reflection equation

- Reflection equation

$$L_o(\mathbf{x}, \omega_o) = \int_{\mathcal{H}^2} f(\mathbf{x}, \omega_o, \omega_i) L_i(\mathbf{x}, \omega_i) \cos\theta_i d\omega_i$$

- Split up BRDF into specular and diffuse
  - Specular means mirror reflection or refraction

$$f(\mathbf{x}, \omega_o, \omega_i) = f_s(\mathbf{x}, \omega_o, \omega_i) + f_d(\mathbf{x}, \omega_o, \omega_i)$$

- Split up incoming radiance into direct, indirect from caustics, rest of indirect

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

Splitting up the reflection equation

- Reflection equation

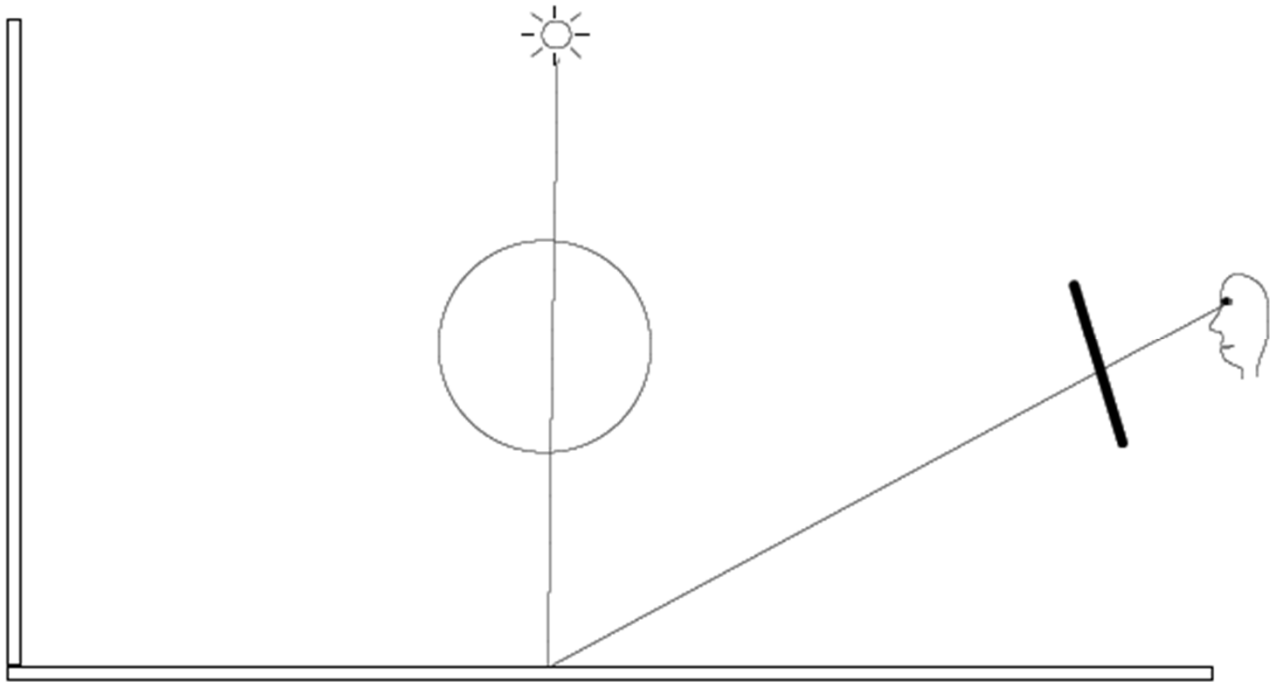
$$
\begin{aligned}
L_o(\mathbf{x},\omega_o) &= \int_{\mathcal{H}^2} f(\mathbf{x},\omega_o,\omega_i) L_i(\mathbf{x},\omega_i)\cos\theta_i d\omega_i \\
&= \int_{\mathcal{H}^2} f(\mathbf{x},\omega_o,\omega_i) L_{i,l}(\mathbf{x},\omega_i)\cos\theta_i d\omega_i && 1. \\
&+ \int_{\mathcal{H}^2} f_s(\mathbf{x},\omega_o,\omega_i)(L_{i,c}(\mathbf{x},\omega_i)+L_{i,d}(\mathbf{x},\omega_i))\cos\theta_i d\omega_i && 2. \\
&+ \int_{\mathcal{H}^2} f_d(\mathbf{x},\omega_o,\omega_i) L_{i,c}(\mathbf{x},\omega_i)\cos\theta_i d\omega_i && 3. \\
&+ \int_{\mathcal{H}^2} f_d(\mathbf{x},\omega_o,\omega_i) L_{i,d}(\mathbf{x},\omega_i)\cos\theta_i d\omega_i && 4.
\end{aligned}
$$

- Evaluate each part separately, each with different algorithm

Photon maps

- Two photon maps
  - Global photon map (all photons)
  - Caustic photon map (only indirect specular; i.e., photons whose previous intersection is on a specular surface)
- Keep track of photon type during photon transport

Direct illumination

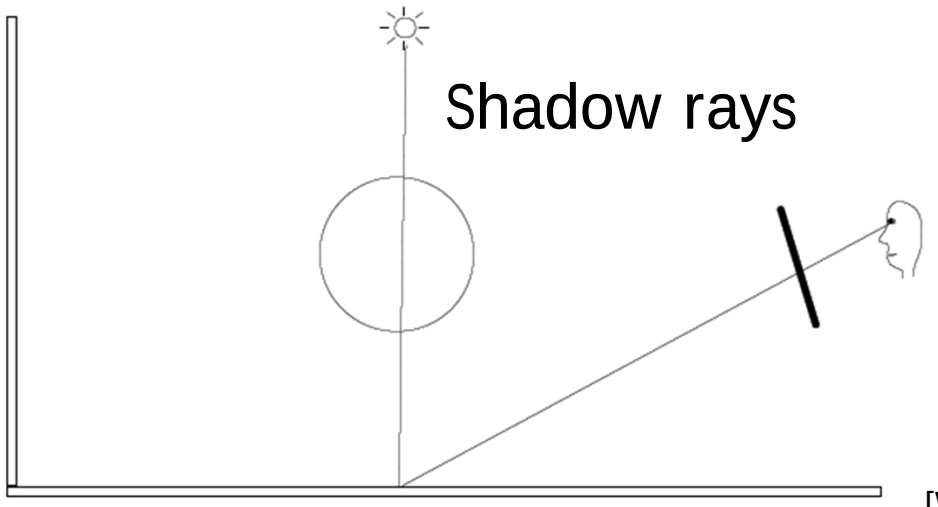


[Wann Jensen]

Direct illumination

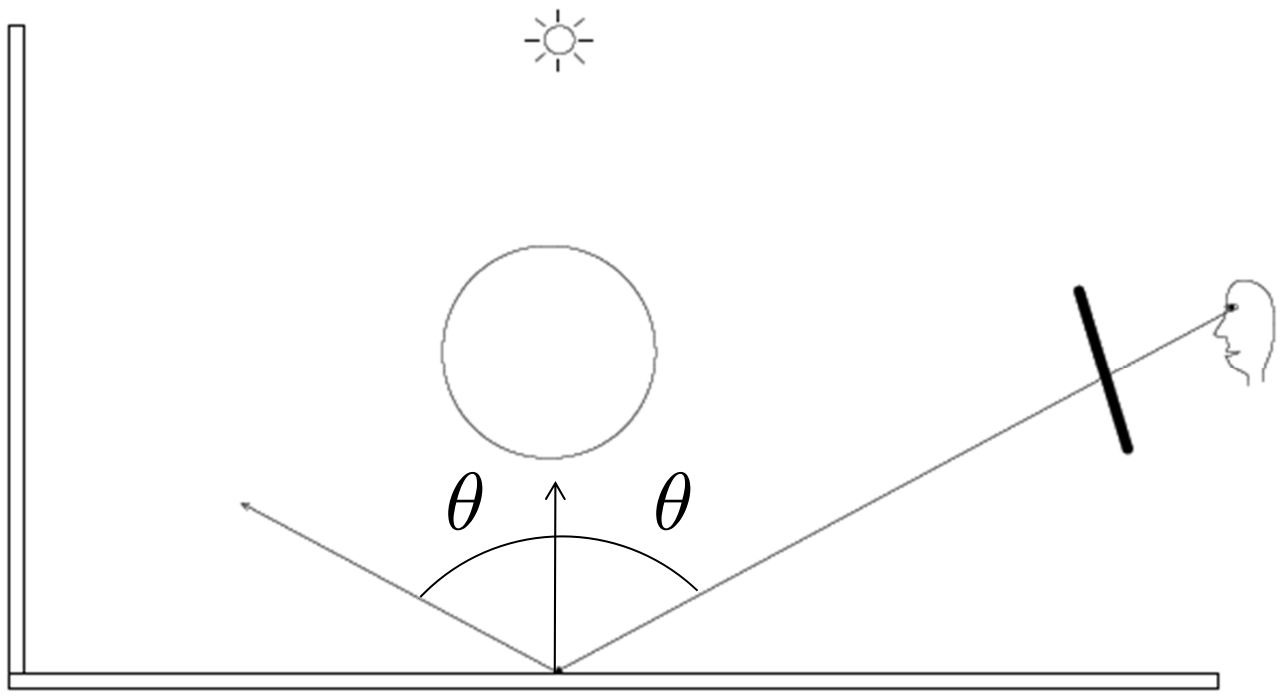
$$\int_{\mathcal{H}^2} f(\mathbf{x}, \omega_o, \omega_i) L_{i,l}(\mathbf{x}, \omega_i) \cos\theta_i d\omega_i \qquad 1.$$

- Path tracing
- Sample light sources using shadow rays
- Advanced: multiple importance sampling



[Wann Jensen]

Specular reflection/refraction



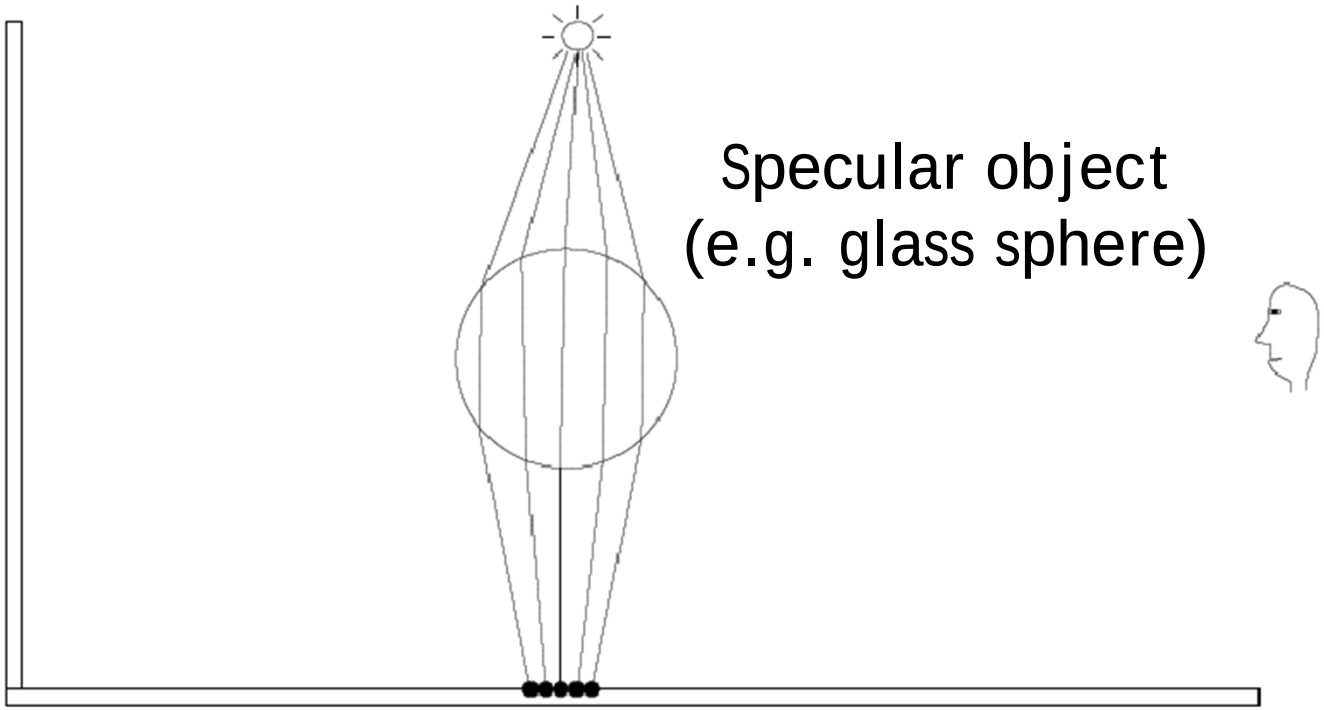
[Wann Jensen]

Specular reflection/refraction

$$\int_{\mathcal{H}^2} f_s(\mathbf{x}, \omega_o, \omega_i)(L_{i,c}(\mathbf{x}, \omega_i) + L_{i,d}(\mathbf{x}, \omega_i)) \cos\theta_i d\omega_i \qquad 2.$$

- Path tracing
- No need to store photons on purely specular (mirror, refractive) surfaces
  - We will never try to do a radiance estimate on purely diffuse surfaces

Caustics



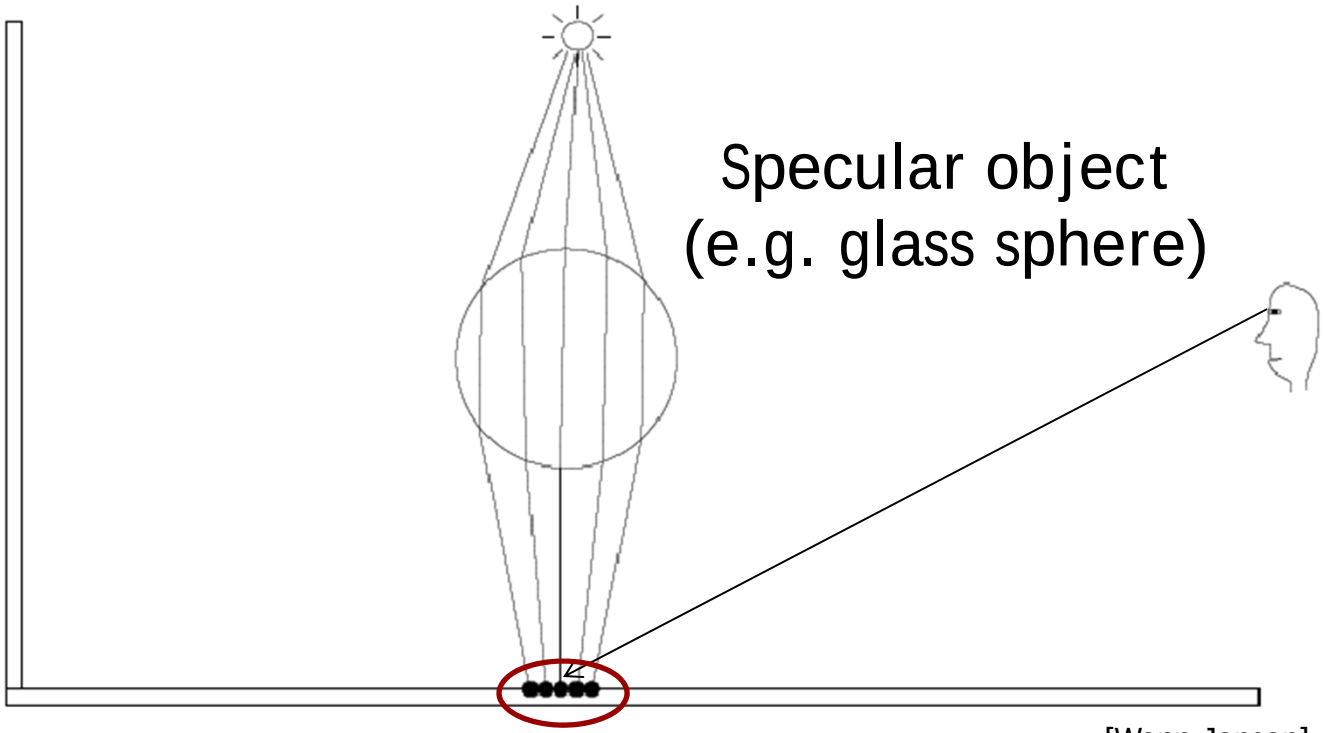
[Wann Jensen]

Caustics

$$\int_{\mathcal{H}^2} f_d(\mathbf{x}, \omega_o, \omega_i) L_{i,c}(\mathbf{x}, \omega_i) \cos\theta_i d\omega_i \qquad 3.$$

- Accounts for indirect illumination due to purely specular paths ($\mathrm{L\{D,S\}^*S^+DE}$ paths)
- Caustic photon map stores photons that
  - Are incident on surface with diffuse part
  - Previous bounce was specular reflection or refraction
- Radiance estimation using caustic photon map

Caustics


Specular object
(e.g. glass sphere)
[Wann Jensen]
Radiance estimate
from caustic photon map

Caustic photon map

- Extension: Useful to do separate photon tracing step for caustic photon map
- Direct photons only towards objects that can produce caustics (specular objects)
  - Reduces „wasted“ photons
  - Do not store photons on purely specular surfaces!

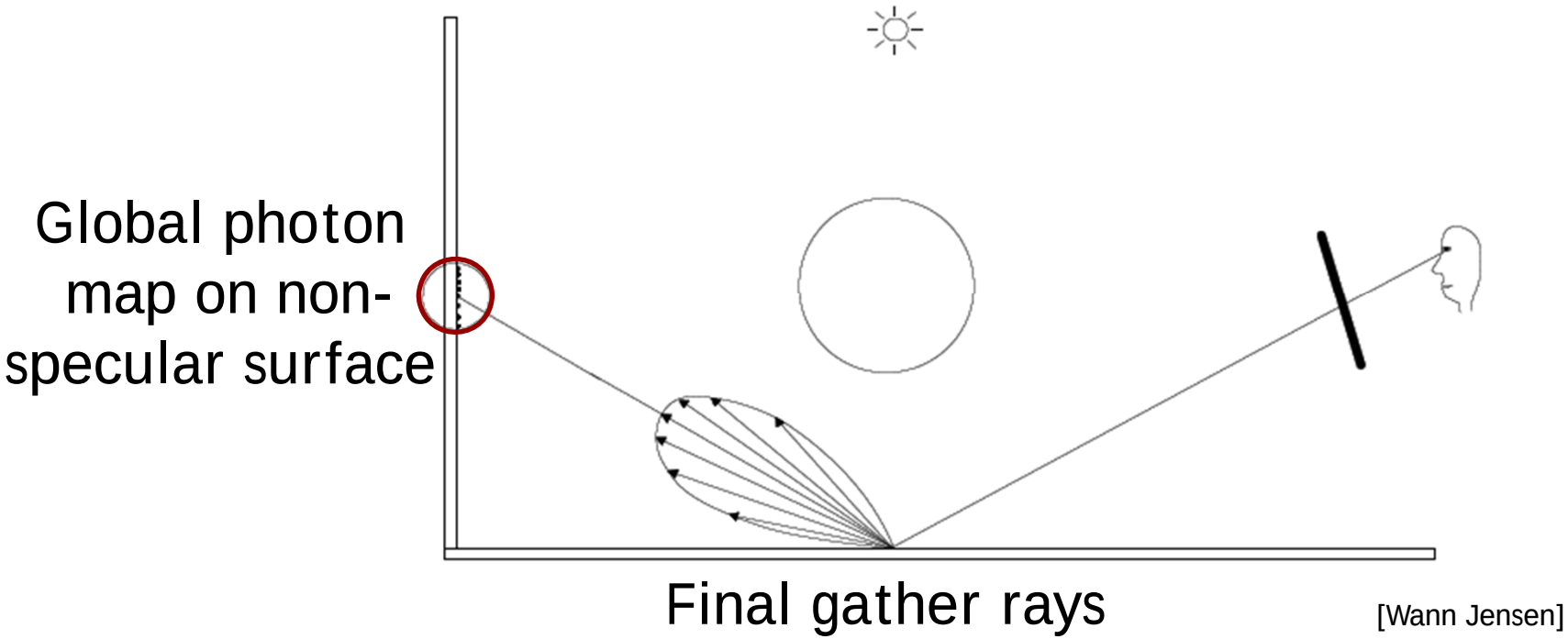
Diffuse indirect illumination

$$\int_{\mathcal{H}^2} f_d(\mathbf{x}, \omega_o, \omega_i) L_{i,d}(\mathbf{x}, \omega_i) \cos\theta_i d\omega_i \qquad 4.$$

- Indirect illumination whose last bounce was on diffuse surface
- Accurate: path tracing
- Less accurate, often good enough: final gathering using global photon map

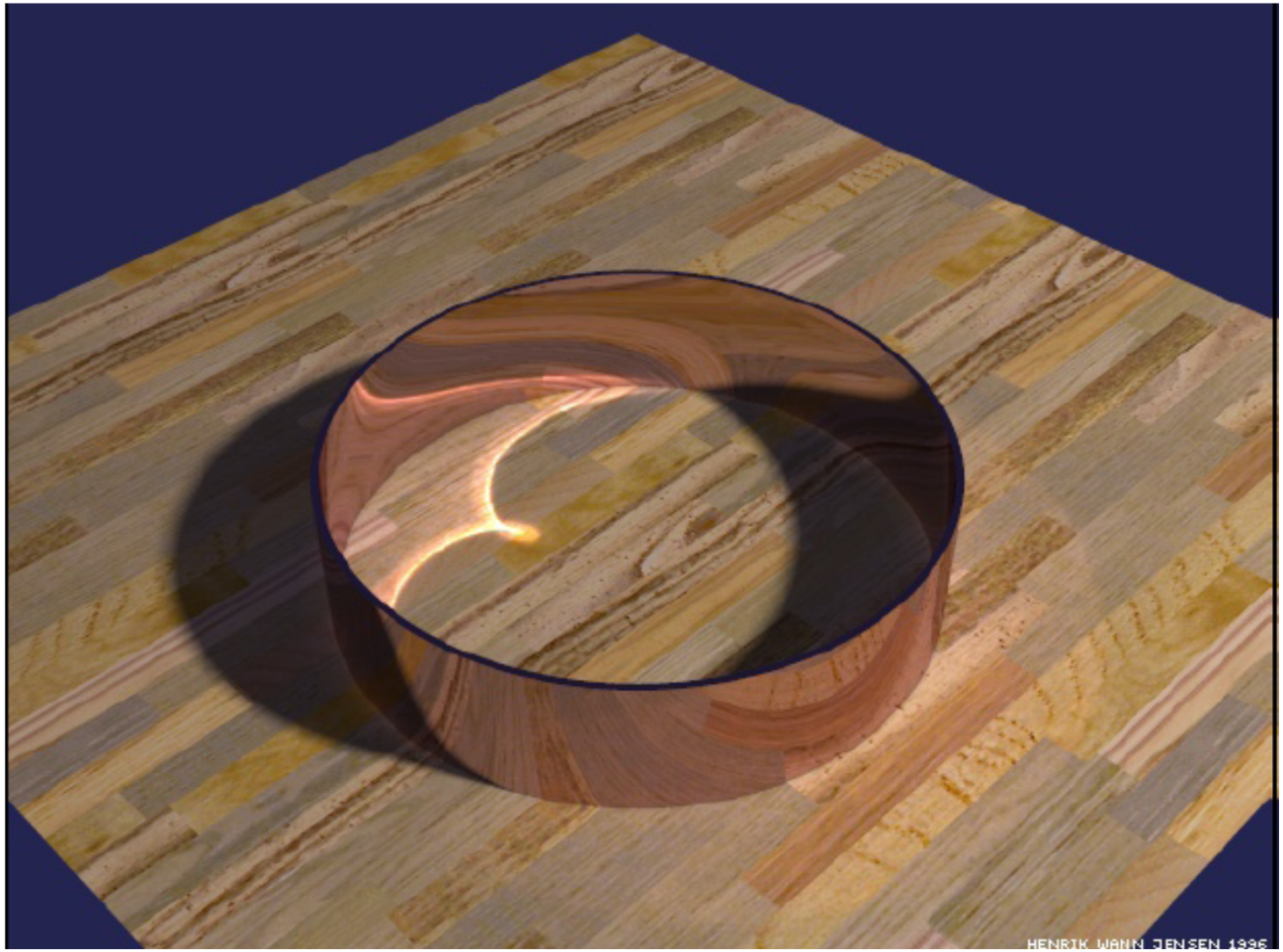
Diffuse indirect illumination

- Final gathering



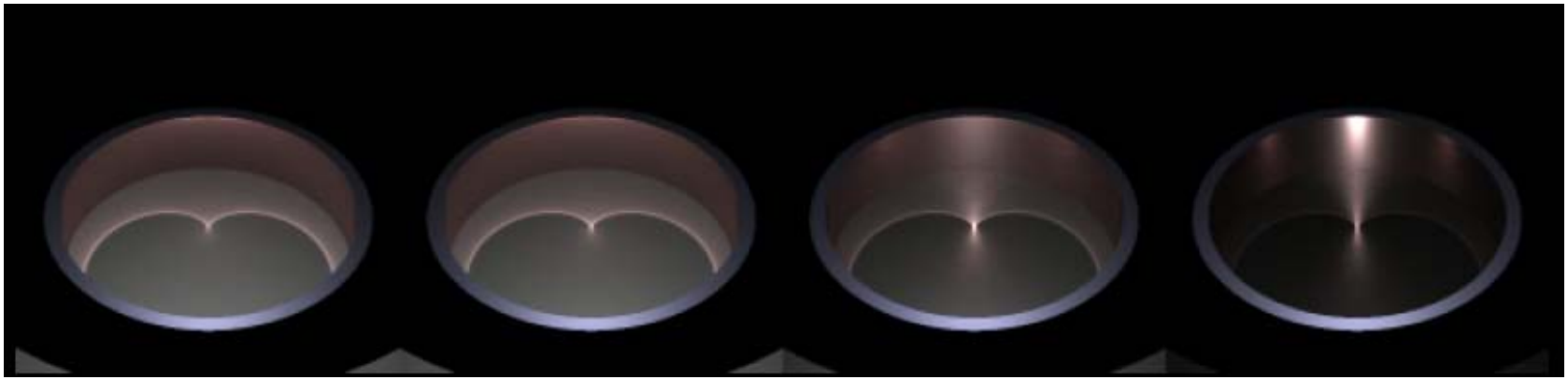
[Wann Jensen]

Reflections inside a ring



50000 caustic photons, 50 photons for radiance estimation

Caustics on glossy surfaces



[Wann Jensen]

340000 photons, ~100 photons for radiance estimation

HDR environment illumination

[Wann Jensen]

Lightprobe from www.debevec.org

Next time

- Irradiance caching