Fundamentals of Computer Graphics

Lecture 12. Photon Mapping

Yong-Jin Liu
liuyongjin@tsinghua.edu.cn

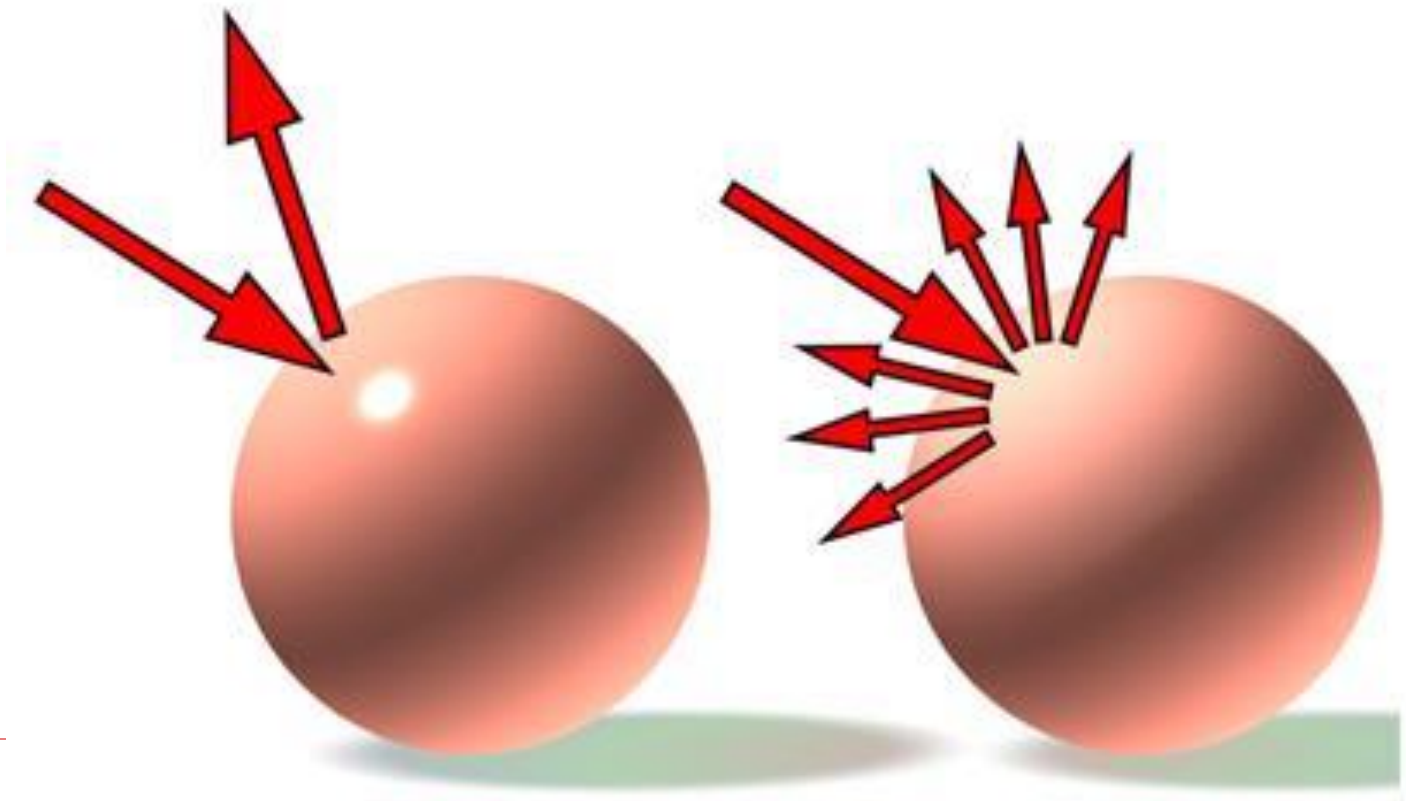
What are PM used for?

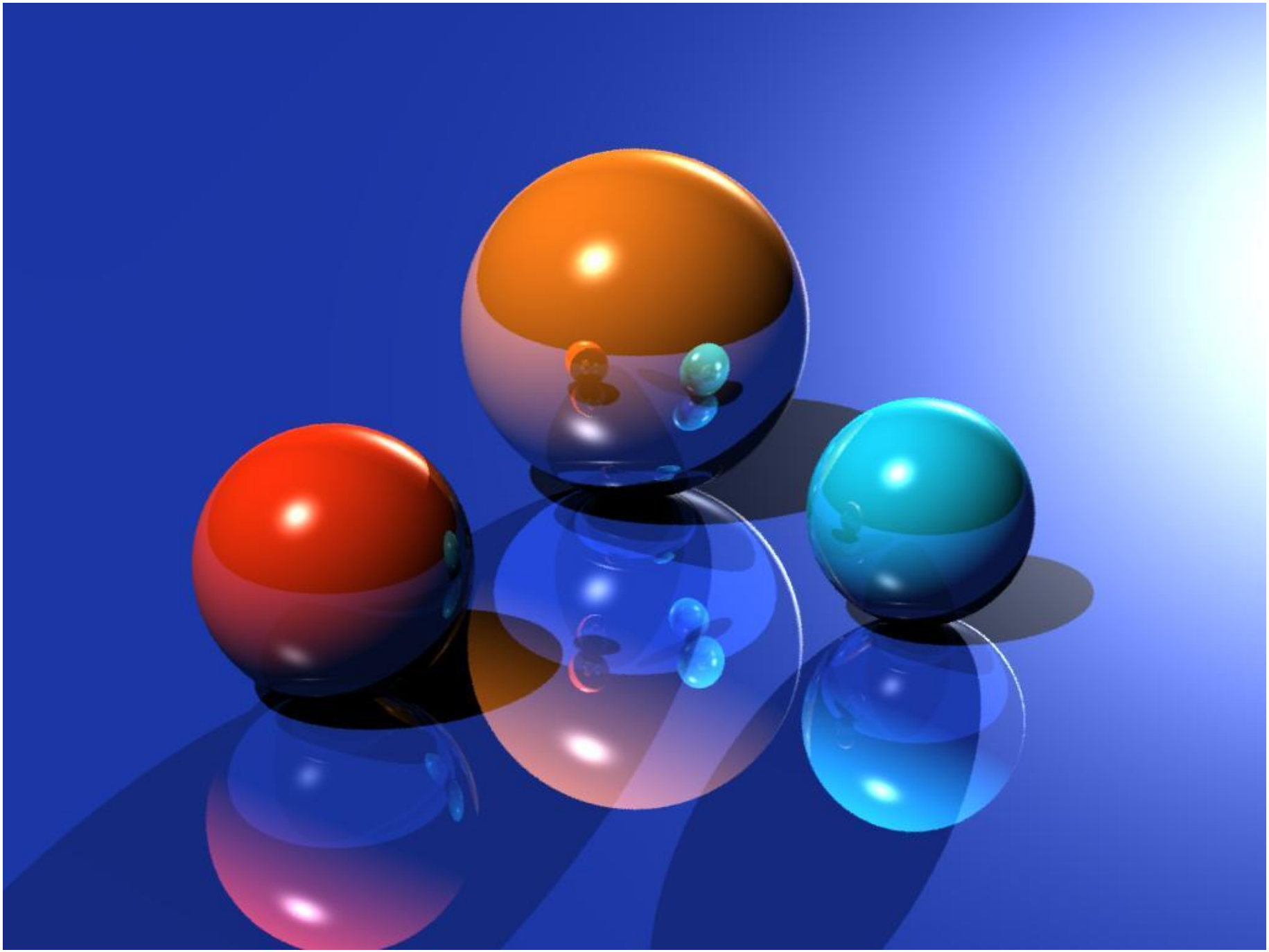
- Methods for computing the Global Illumination (GI)
- GI = Direct + Indirect Illumination (multiple bounces)
  - Ray tracing
  - Radiosity
  - Why Photo Mapping (PM)?

Why *ray-tracing* vs. *radiosity*

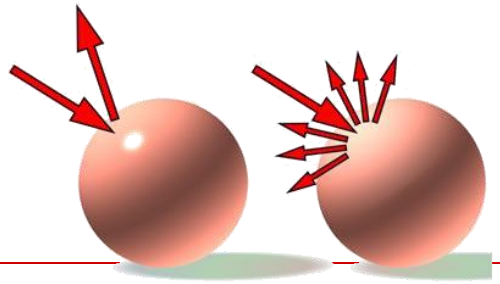
Specular reflection

Diffuse reflection





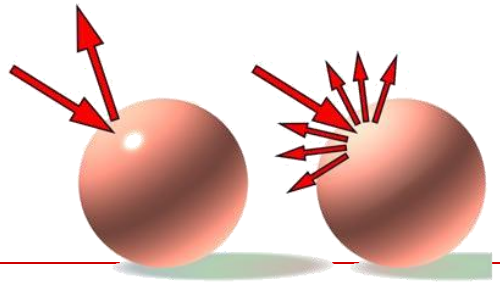
Ray tracing



- Not good for diffuse components

Ray tracing displays only specular highlights

Radiosity



- Very good for diffuse components

Notice that the walls are more naturally illuminated by reflected as well as direct light. Notice in particular, the circle of light on the left wall above the table. This is caused by light reflecting from the silver tray (with decanter and glasses) on the table.

Summary: Radiosity

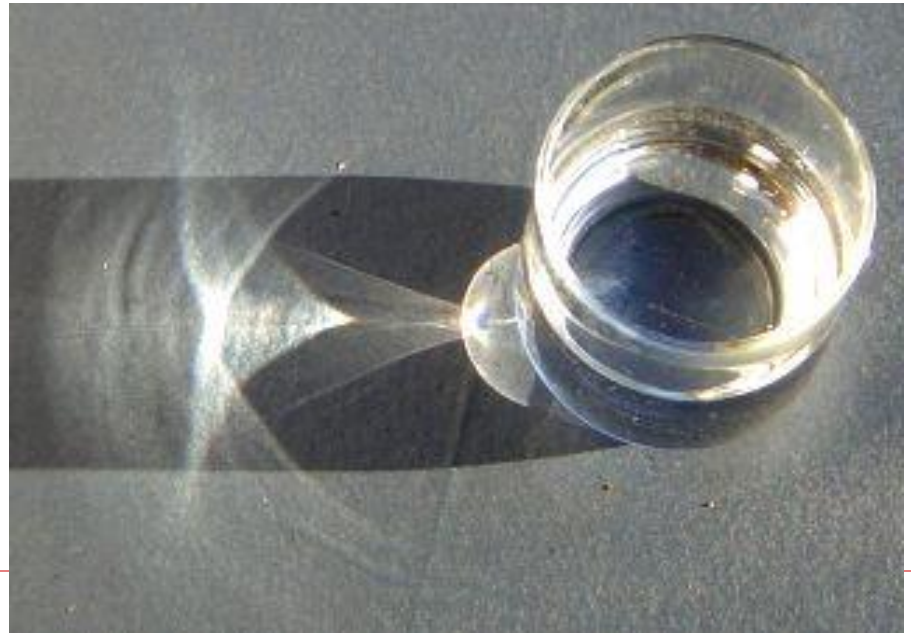
- It is an **indirect** global illumination algorithm
- Diffuse inter-reflectivity simulates many reflections of light around a scene
- The result is often **softer** more natural shadows

What are PM used for?

- Methods for computing the Global Illumination (GI)
- GI = Direct + Indirect Illumination (multiple bounces)
  - Ray tracing
  - Radiosity
  - Why Photo Mapping (PM)?

Caustics

- In optics, a **caustic** or caustic network is the envelope of light rays reflected or refracted by a curved surface or object.

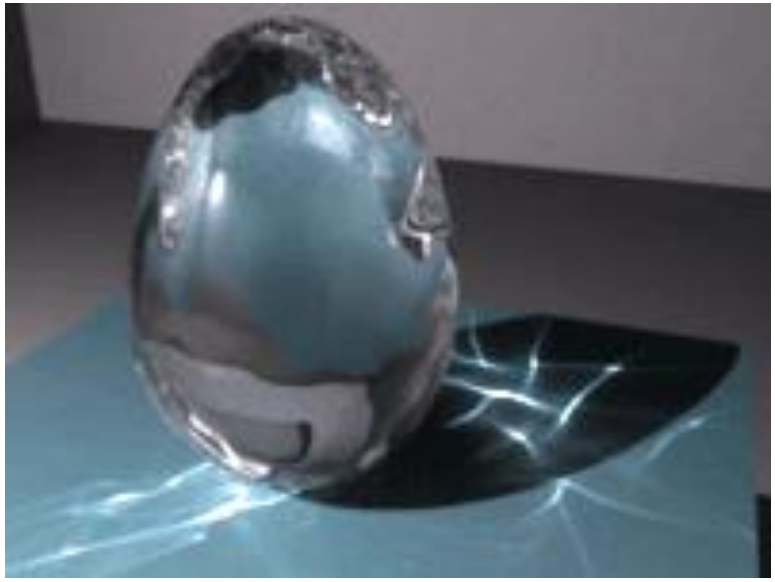


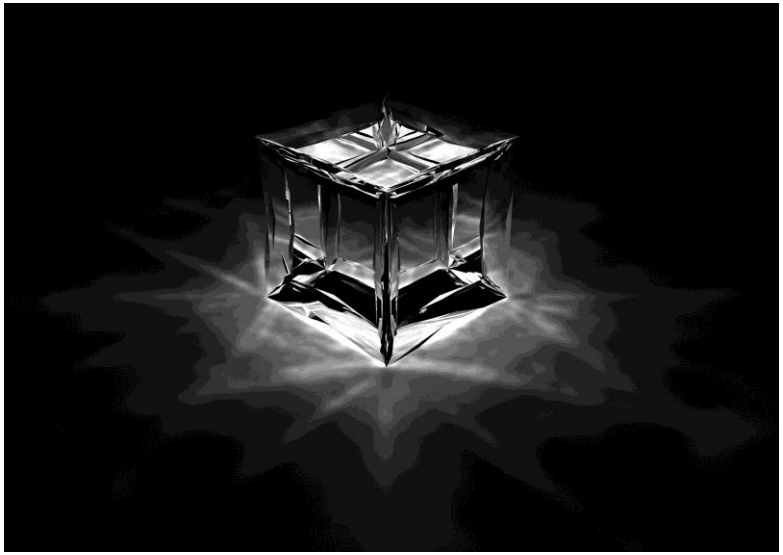
Caustics produced by a glass of water

The caustic is a curve or surface to which each of the light rays is tangent, defining a boundary of an envelope of rays as a curve of concentrated light. These shapes often have cusp singularities.

Caustics

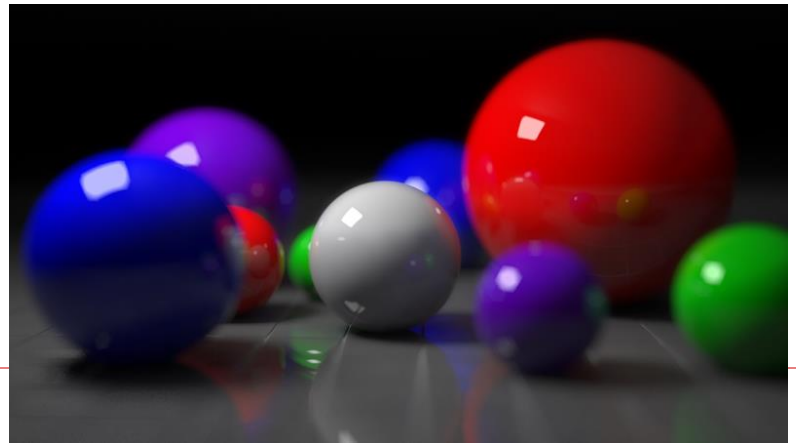
- **Caustics** are formed when light reflected from or transmitted through one or more specular surfaces strikes a diffuse surface.

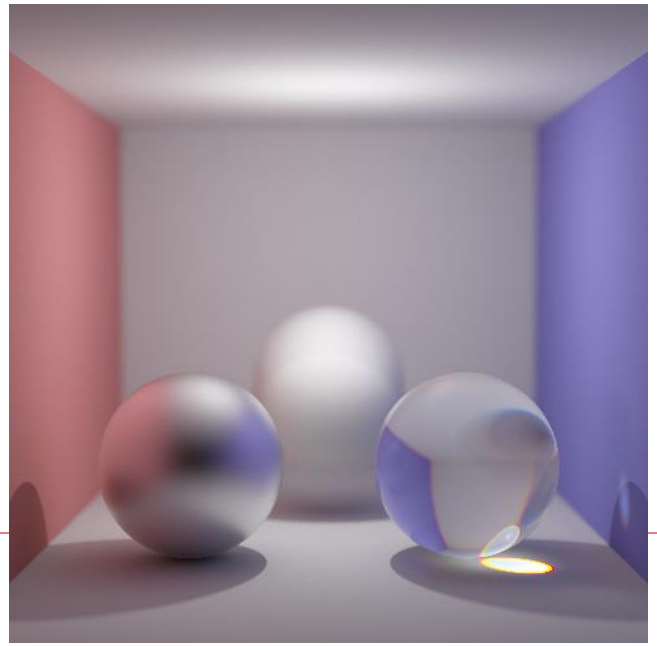




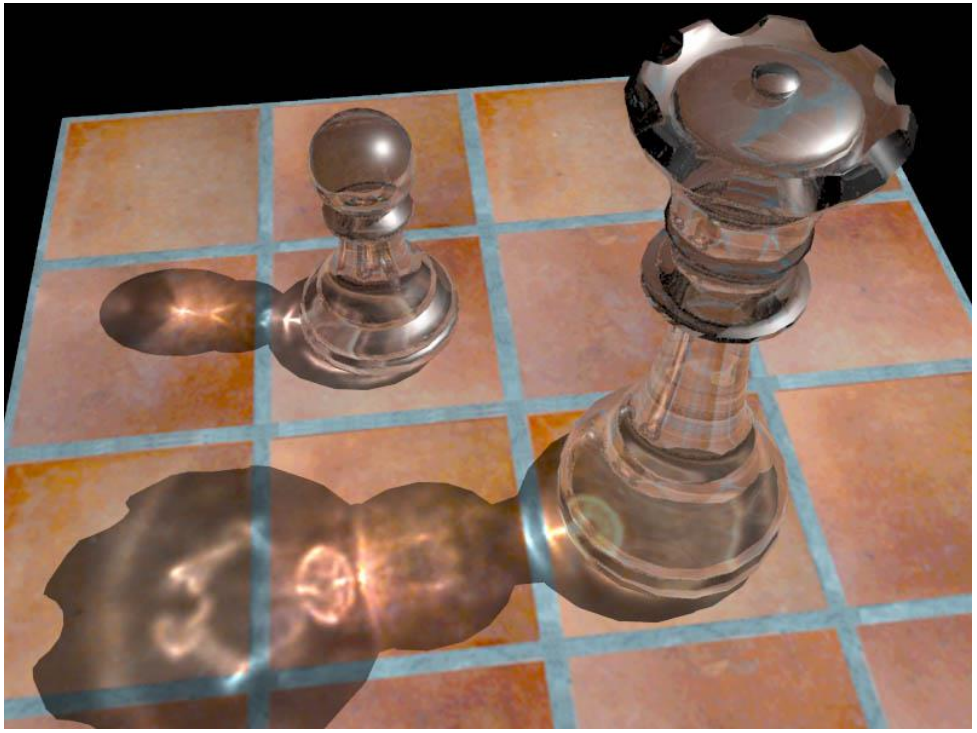
Why Photo Mapping (PM)?

- Create nice pictures with translucent objects and rendering nice effects on diffuse surfaces
  - Specular, translucent, shadow (ray tracing)
  - Diffusion (radiosity)
  - Caustics





Caustics

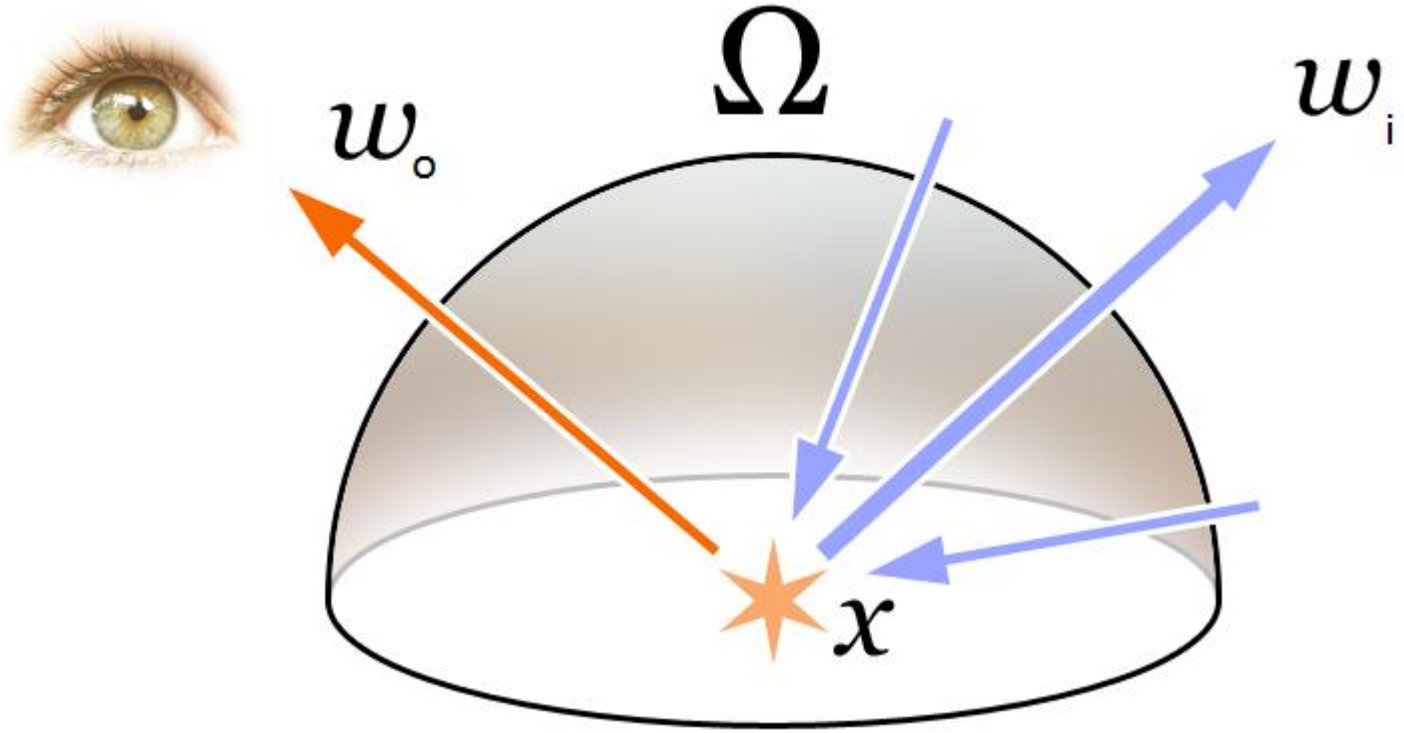


Rendering equation

- the rendering equation is an integral equation in which the equilibrium radiance leaving a point is given as the sum of emitted plus reflected radiance under a geometric optics approximation
- The various realistic rendering techniques in computer graphics attempt to (approximately) solve this equation

Rendering equation

- The physical basis for the rendering equation is the law of conservation of energy



Rendering equation

- The physical basis for the rendering equation is the law of conservation of energy
  - Assuming that L denotes radiance, we have that at each particular position and direction, the outgoing light (L_o) is the sum of the emitted light (L_e) and the reflected light.

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_\Omega f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t)\, L_i(\mathbf{x}, \omega_i, \lambda, t)\, (\omega_i \cdot \mathbf{n})\, \mathrm{d}\,\omega_i$$

Rendering equation

- The physical basis for the rendering equation is the law of conservation of energy
  - The reflected light itself is the sum of the incoming light (L_i) from all directions, multiplied by the surface reflection and cosine of the incident angle.

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_\Omega f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t)\, L_i(\mathbf{x}, \omega_i, \lambda, t)\, (\omega_i \cdot \mathbf{n})\, \mathrm{d}\,\omega_i$$

$$L_{\mathrm{o}}(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t) = L_e(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_{\mathrm{i}}, \omega_{\mathrm{o}}, \lambda, t)\, L_{\mathrm{i}}(\mathbf{x}, \omega_{\mathrm{i}}, \lambda, t)\, (\omega_{\mathrm{i}} \cdot \mathbf{n})\, \mathrm{d}\, \omega_{\mathrm{i}}$$

- λ is a particular wavelength of light
- t is time
- $\mathbf{x}$ is the location in space
- ω_{o} is the direction of the outgoing light
- ω_{i} is the negative direction of the incoming light
- $L_{\mathrm{o}}(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t)$ is the total spectral radiance of wavelength λ directed outward along direction ω_{o} at time t, from a particular position $\mathbf{x}$
- $L_e(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t)$ is emitted spectral radiance
- Ω is the unit hemisphere containing all possible values for ω_{i}
- $\int_{\Omega} \ldots \mathrm{d}\, \omega_{\mathrm{i}}$ is an integral over Ω
- $f_r(\mathbf{x}, \omega_{\mathrm{i}}, \omega_{\mathrm{o}}, \lambda, t)$ is the bidirectional reflectance distribution function, the proportion of light reflected from ω_{i} to ω_{o} at position $\mathbf{x}$, time t, and at wavelength λ
- $L_{\mathrm{i}}(\mathbf{x}, \omega_{\mathrm{i}}, \lambda, t)$ is spectral radiance of wavelength λ coming inward toward $\mathbf{x}$ from direction ω_{i} at time t
- $\omega_{\mathrm{i}} \cdot \mathbf{n}$ is the weakening factor of inward irradiance due to incident angle, as the light flux is smeared across a surface whose area is larger than the projected area perpendicular to the ray

BRDF

- Bidirectional reflectance distribution function

The **bidirectional reflectance distribution function** (**BRDF**; $f_r(\omega_i, \omega_r)$) is a four-dimensional function that defines how light is reflected at an opaque surface. The function takes a negative incoming light direction, ω_i, and outgoing direction, ω_r, both defined with respect to the surface normal $\mathbf{n}$, and returns the ratio of reflected radiance exiting along ω_r to the irradiance incident on the surface from direction ω_i. Each direction ω is itself parameterized by azimuth angle ϕ and zenith angle θ, therefore the BRDF as a whole is 4-dimensional. The BRDF has units sr^{-1}, with steradians (sr) being a unit of solid angle.

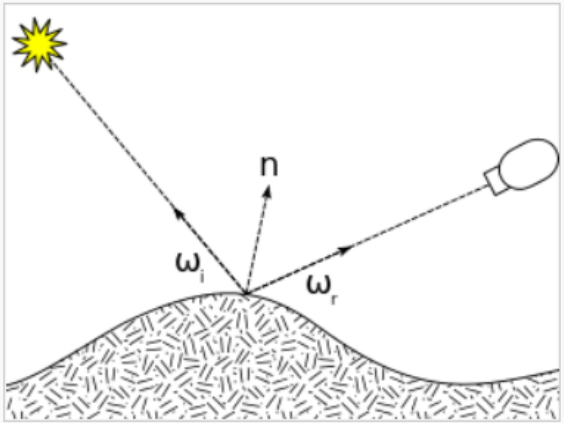


Diagram showing vectors used to define the BRDF. All vectors are unit length. ω_i points toward the light source. ω_r points toward the viewer (camera). n is the surface normal.

Rendering equation

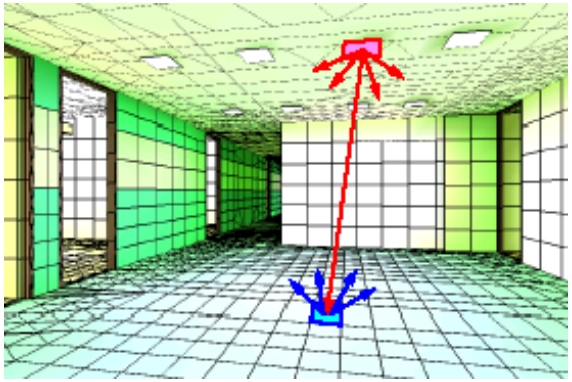
- Global illumination
  - Target: Generating synthetic images that are **indistinguishable** from images taken of the real world
  - Global illumination algorithms are designed to **approximate** the rendering equation which describes the complete transport of light within an environment.

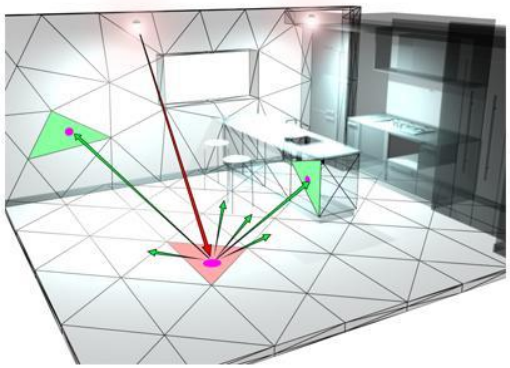
Photon Mapping – taxonomy

- Hybrid algorithm – combines “the best” of two worlds
  - Radiosity – computes radiance (radiosity) values for every mesh element (i.e. the object-space method)
  - Ray-tracing – computes radiance values for every pixel by generating paths between the pixel and light sources

I. Radiosity

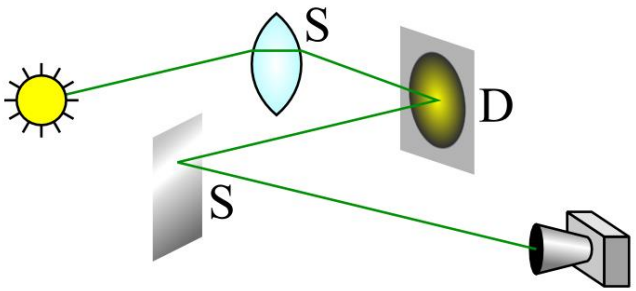
- A finite element method that computes a **view independent** GI solution as a finite mesh
  - The environment is divided up into patches that are uniform and perfectly diffuse and energy is transferred amongst these patches until equalized.

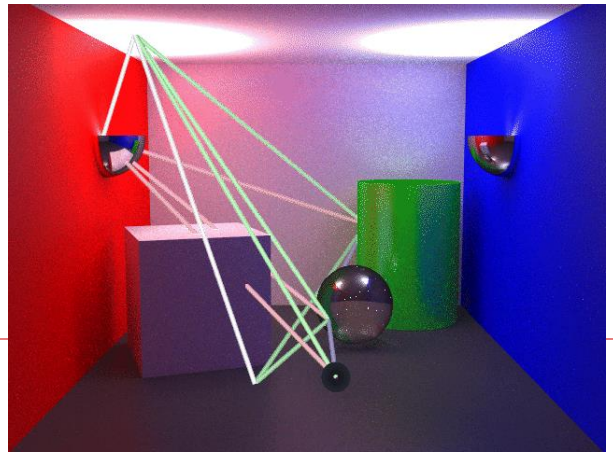




II. Path tracing

- A probabilistic **view dependent** point sampling technique that extends raytracing to approximate all light paths such as those that contribute to indirect illumination and caustics.
  - Path tracing is a random process where each path is a simple random walk and thus a Markov chain.



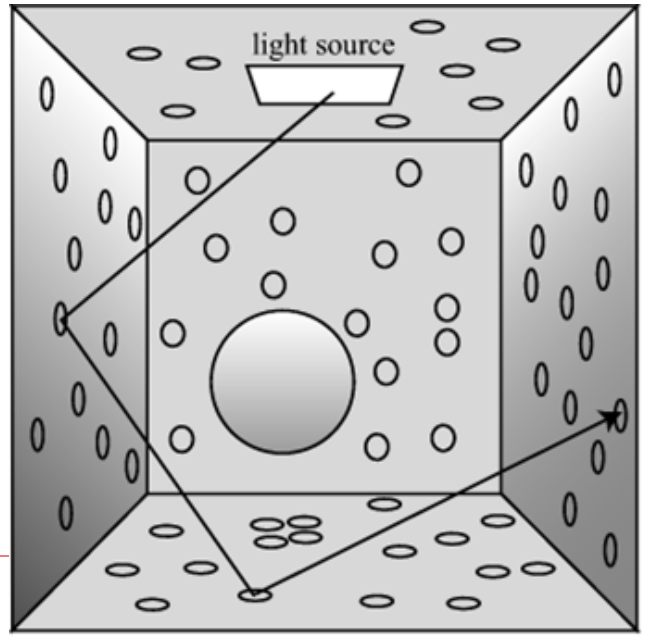


Photon Mapping – taxonomy II

- Hybrid algorithm – combines "the best" of two worlds
  - Multipass method – generally computing various parts of the light transport and then combining them together
  - Special care needs to be taken about not computing the same light transport twice (too bright parts of the scene)

Photon Mapping

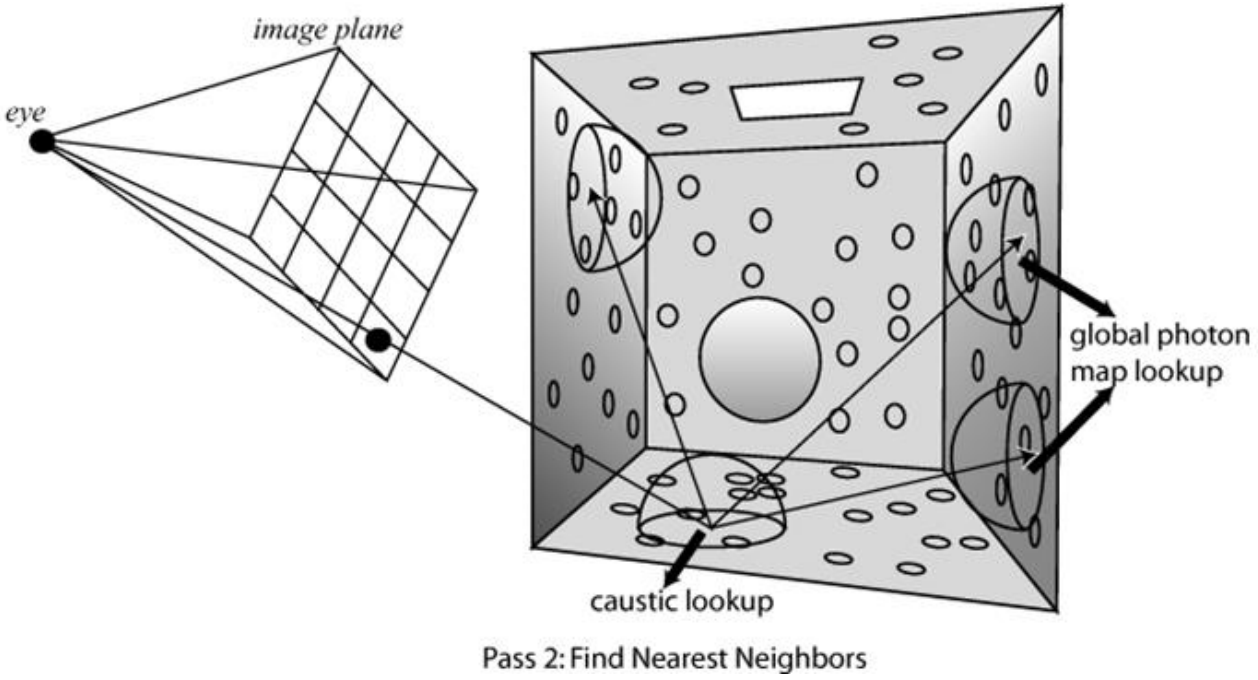
- Two passes
  1. Approximation of the light transport **in the object-space** – photon shooting (particle tracing), storing photon-surface interactions in the photon map



Pass 1: Shoot Photons

Photon Mapping

- Two passes
  2. Computing the pixel values **in the image plane** by exploiting the information stored in the photon map – final gather (ray-tracing)



Pass 2: Find Nearest Neighbors

First pass

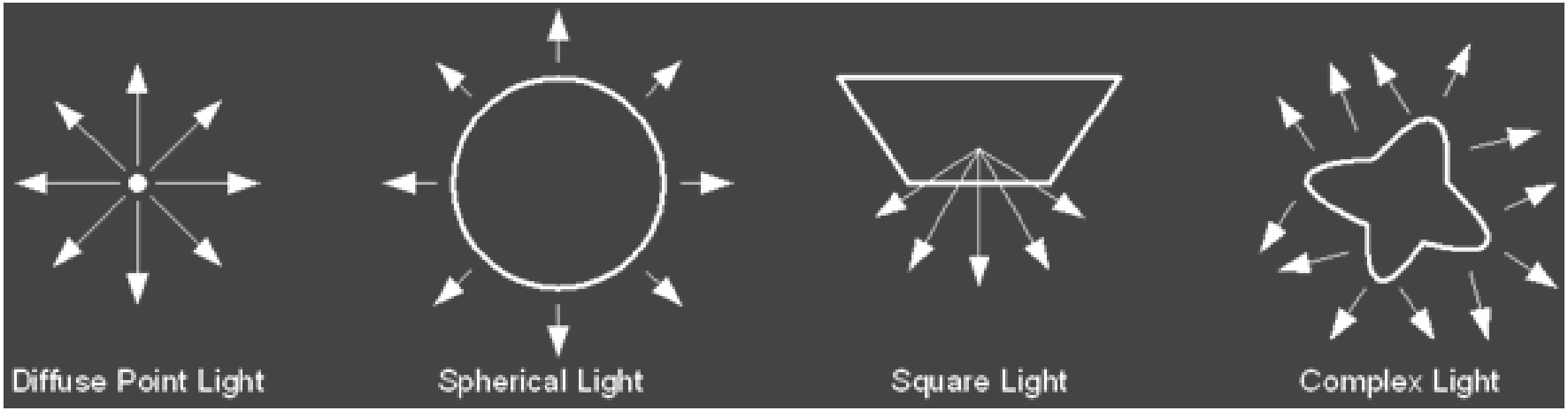
Photon Tracing

The process of emitting discrete photons from the light sources and tracing them through the scene. The primary goal of this pass is to populate the photon maps that are used in the rendering pass to calculate the reflected radiance at surfaces and out-scattered radiance in participating media.

- Photon Emission (shooting)
- Photon Scattering (tracing)
- Photon Storing (photon map)

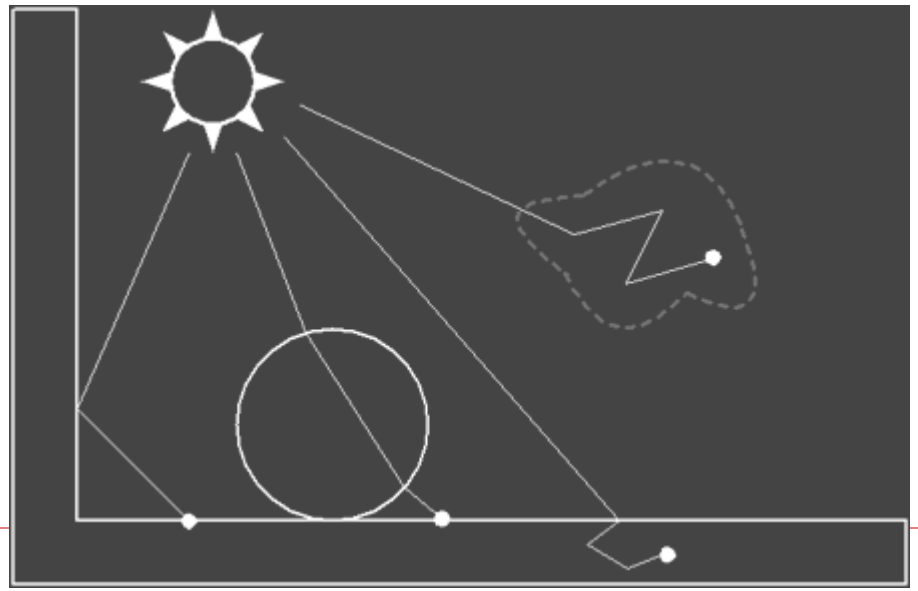
First pass

- Photon Tracing
  - Photon Emission (shooting)
  - Photon Scattering (tracing)
  - Photon Storing (photon map)



First pass

- Photon Tracing
  - Photon Emission (shooting)
  - Photon Scattering (tracing)
  - Photon Storing (photon map)



First pass

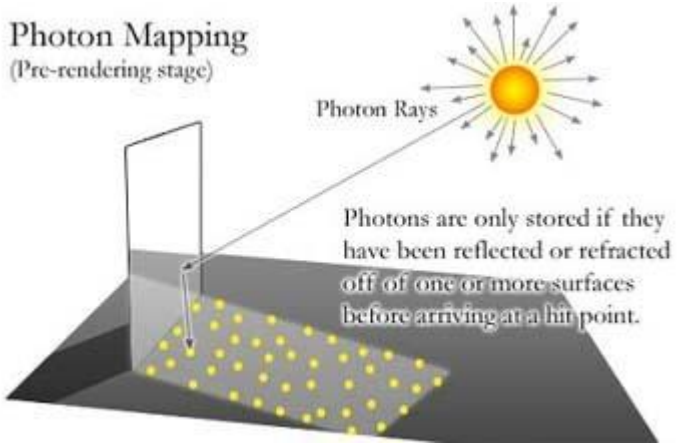
- Photon Tracing
  - Photon Emission (shooting)
  - Photon Scattering (tracing)
  - Photon Storing (photon map)

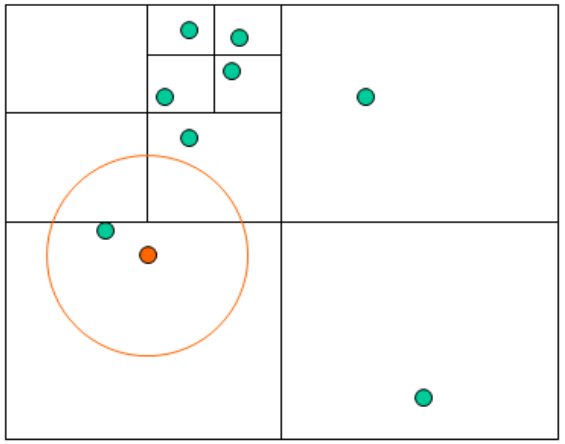
  The data structure usually for the photon map is a kd-tree, ideal for handling non-uniform distributions of photons.

```
struct photon {
    float x, y, z;          // position ( 3 x 32 bit floats )
    char p[4];              // power(rgb) packed as 4 chars
    char phi, theta;        // compressed incident direction
    short flag;             // flag used for kd-tree
}
```

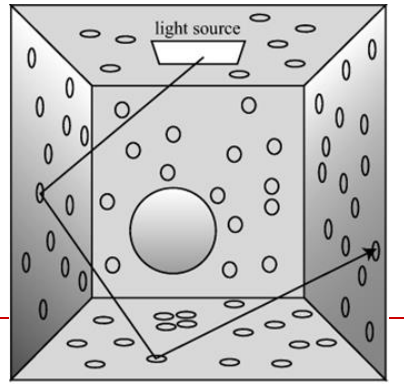
First pass

- Photon Tracing
  - Photon Emission (shooting)
  - Photon Scattering (tracing)
  - Photon Storing (photon map)





Photon shooting

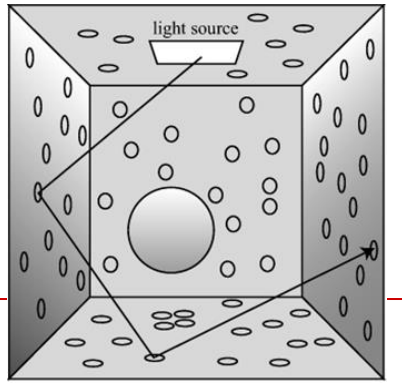


- Emitting quanta of energy (photons) from light sources
- Each photon carries the same amount of energy

$$\Phi_p = \frac{\sum_{Lights} \Phi_L}{N} \quad W/s$$

- Emission is driven probabilistically according to the spatial and directional properties of light sources and according to their brightness

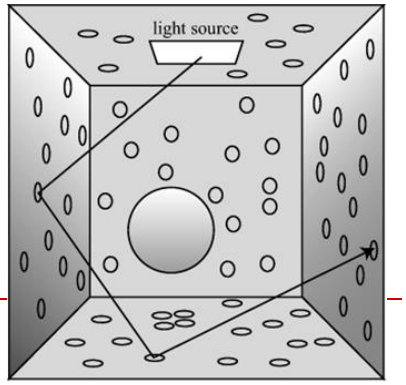
Photon tracing



- Propagating photons in the scene just as rays are in the ray-tracing
- Reflection, transmission, absorption
- Standard Monte Carlo sampling techniques

- Still, tracing of photons is not the same as tracing the rays!
  - We are "tracing the density" …

Photon tracing



- Path tracing can be very computationally expensive as a large number of samples are needed
- Standard Monte Carlo sampling techniques
  - reduce variance and achieve convergence in a finite number of steps
  - importance sampling and Russian roulette

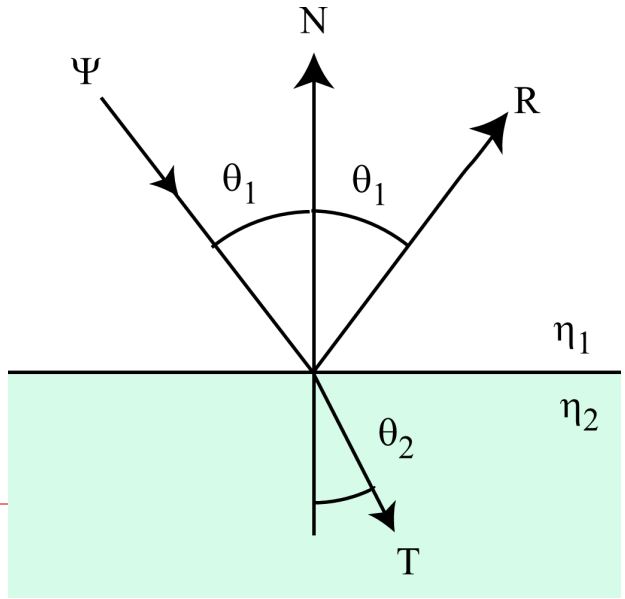
Photon tracing - refraction

- Ray-tracing & ray refraction:
  - Change of medium density bends the ray towards/outwards the normal direction
  - Results in change of energy

$$L(x,\omega) = \frac{d^2\Phi(x)}{\cos\theta \, dA \, d\omega}$$

- Compensation factor:

$$(\eta_2 / \eta_1)^2$$



Photon tracing - density

- Photon tracing
  - The photon carries the same amount of energy after refraction
- Why
  - Because we want the flux in the scene to be captured in the distribution of photons rather than in the various amounts of energy they bear
- Photons approximates the radiosity

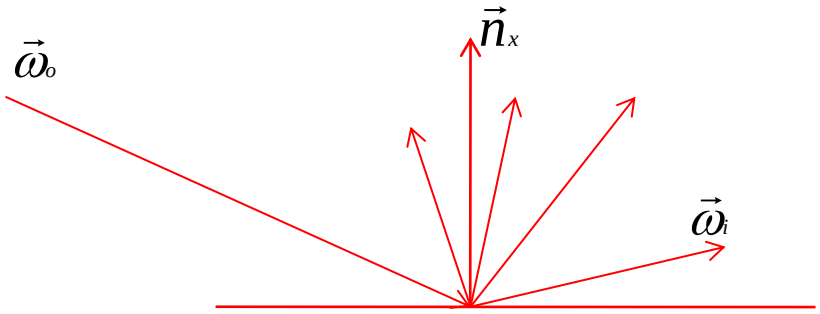
Photon tracing – Russian Roulette

- Russian Roulette: a standard Monte Carlo technique
- Use Russian Roulette to probabilistically decide whether photons are reflected, refracted or absorbed
- Allows as to keep the number of traced photons low as well as keep their energy approximately the same
- For instance instead of tracing 1000 photons with half energy after hitting the surface with albedo = 0.5 we will trace just 500 photons with the full energy
  - 500 photons were simply absorbed by material

Photon-tracing

- Solving the rendering equation by Monte Carlo distribution photon-tracing

$$L(x,\vec{\omega}_o) = L_e(x,\vec{\omega}_o) + \int_{\Omega} f_r(x,\vec{\omega}_o,\vec{\omega}_i) L(x,\vec{\omega}_i) \,|\, \vec{n}_x \cdot \vec{\omega}_i \,|\, d\omega_i$$



- Use the information stored in the photon map

Photon map query

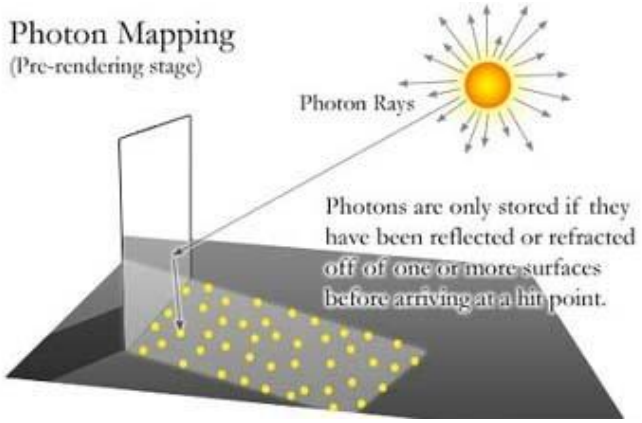
- We can query the photon map about the estimate of the reflected radiance from any surface point

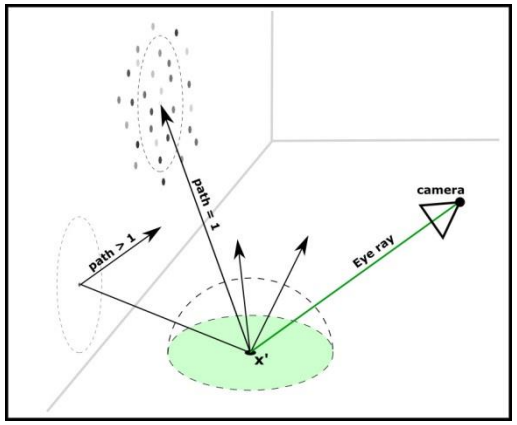
$$L(x,\vec{\omega}_o) = L_e(x,\vec{\omega}_o) + \int_{\Omega} f_r(x,\vec{\omega}_o,\vec{\omega}_i) L(x,\vec{\omega}_i) \,|\, \vec{n}_x \cdot \vec{\omega}_i \,|\, d\omega_i$$

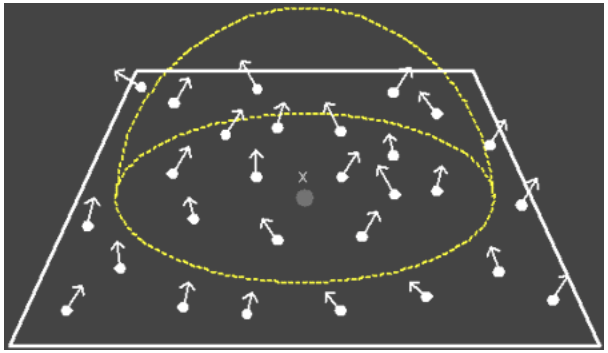
$$L(x,\vec{\omega}_o) = L_e(x,\vec{\omega}_o) + \qquad\qquad L_r(x,\vec{\omega}_o)$$

Photon map query

- We can query the photon map about the estimate of the reflected radiance from any surface point







Store in kd tree query integration

Radiance estimate – derivation I

- Relationship between radiance and flux

$$L(x,\vec{\omega}) = \frac{d^2\Phi(x,\vec{\omega})}{|\vec{n}_x \cdot \vec{\omega}|\, dA\, d\vec{\omega}}$$

- Reflected radiance equation

$$L_r(x,\vec{\omega}_o) = \int_\Omega f_r(x,\vec{\omega}_o,\vec{\omega}_i) L(x,\vec{\omega}_i) |\vec{n}_x \cdot \vec{\omega}_i|\, d\omega_i$$

$$L_r(x,\vec{\omega}_o) = \int_\Omega f_r(x,\vec{\omega}_o,\vec{\omega}_i) \frac{d^2\Phi(x,\vec{\omega}_i)}{|\vec{n}_x \cdot \vec{\omega}_i|\, dA\, d\vec{\omega}_i} |\vec{n} \cdot \vec{\omega}_i|\, d\omega_i$$

Intermezzo – "back to Radiosity"

- We consider the directionality

$$L_r(x,\vec{\omega}_o) = \int_{\Omega} f_r(x,\vec{\omega}_o,\vec{\omega}_i)\frac{d^2\Phi(x,\vec{\omega}_i)}{dA\,d\vec{\omega}_i}d\omega_i$$

- If we reduced the dimensionality of the problem by assuming the Lambertian surfaces we would get

$$L_r(x,\vec{\omega}_o) = k_x \int_{\Omega} \frac{d^2\Phi(x,\vec{\omega}_i)}{dA\,d\vec{\omega}_i}d\omega_i = k_x\ E(x)$$

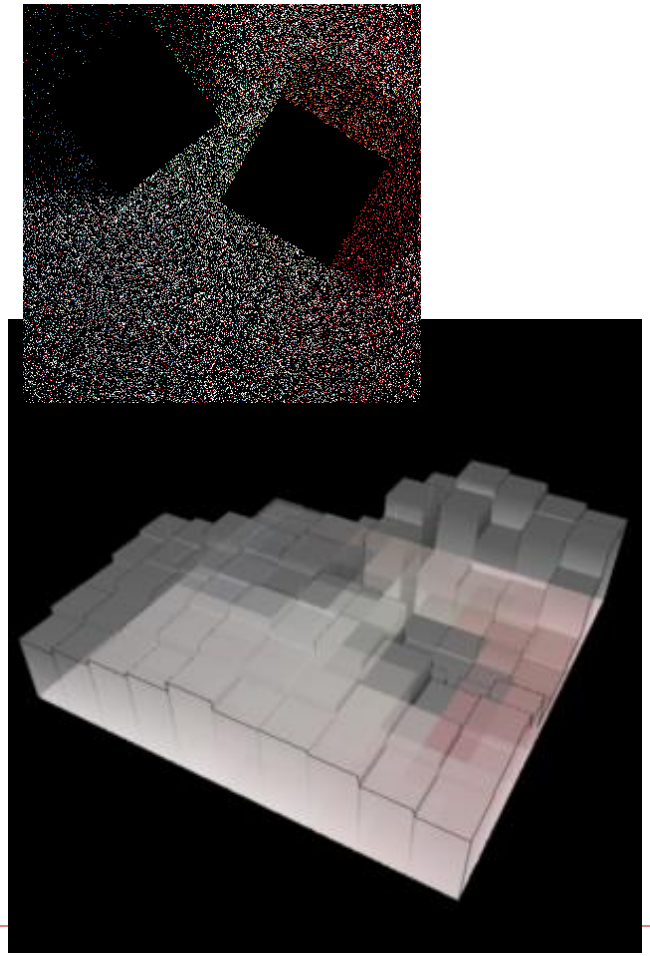
Density estimation – histogram method

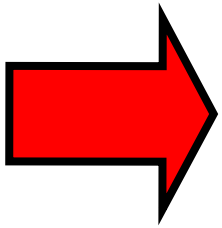
- Stochastic radiosity uses histogram-based density estimation method (known from statistics):

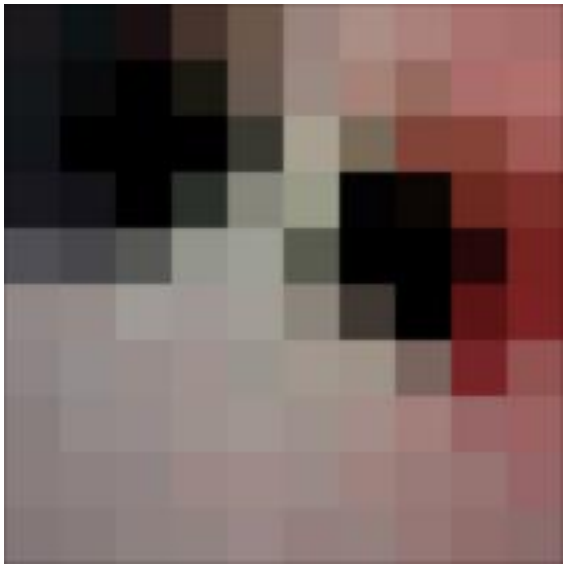
$$E(x) \approx \frac{\Delta\Phi(x)}{\Delta A}$$

- Density of flux is captured in the illumination map or in the geometry mesh itself
  - Size of the bin for capturing the flux carried by photons is fixed (texel or mesh element)
- Positions of photon hits are not stored
  - Flux in bins is accumulated incrementally

Density estimation – histogram method





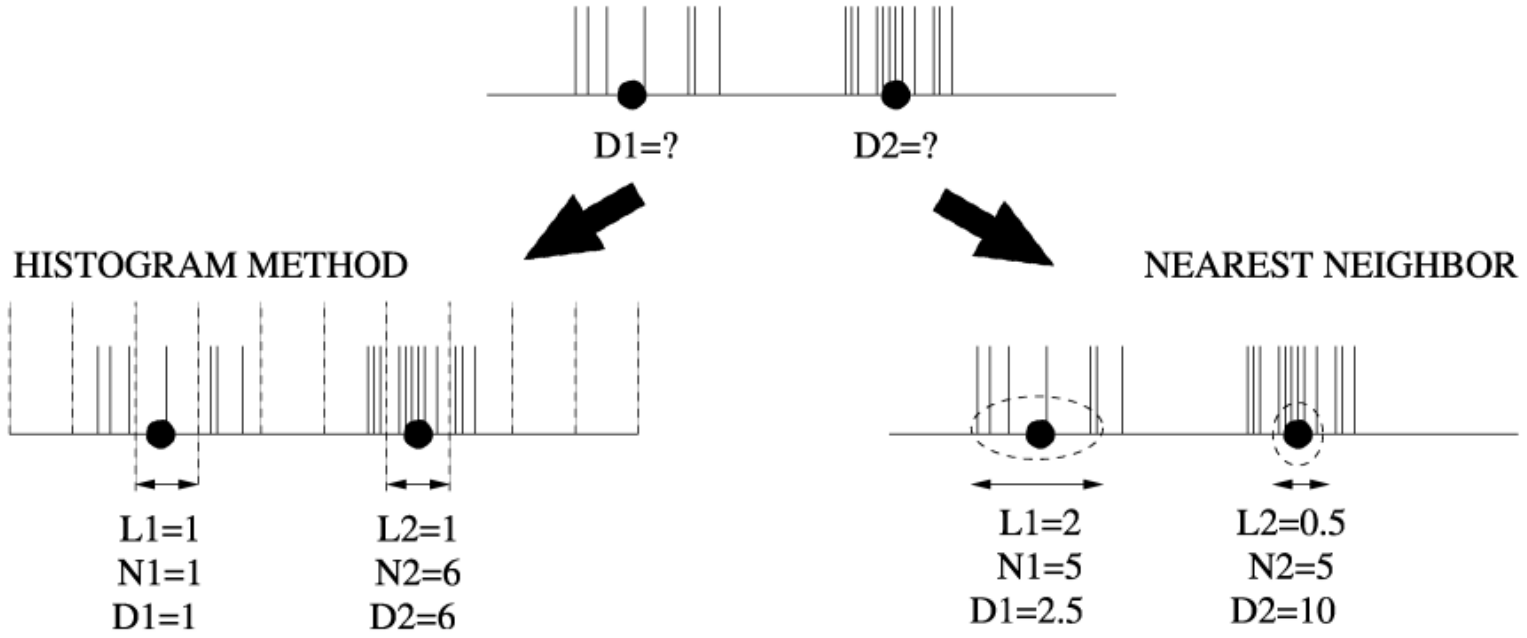


Density estimation – nearest neighbour

- What if we fixed the number of photons in density estimation (N/A) instead of fixing the surface area?
- Lets look for the N nearest photons
- Density estimation is then determined by the size of an area where we found those N photons

Density estimation

Nearest neighbour vs. histogram



Density estimation – nearest neighbor

- We need to store the positions of incoming photons
- The density estimation isn't computed "on-line"
  - Dependency on the whole set of samples => it has to be computed a posteriori
- Thus we need to store the photon hits positions
  - Why not store the incoming direction as well?

Radiance estimation – derivation II

- Photon map stores information about
  - Photons hit positions
  - Incoming photon direction
  - Flux carried by the photon

$$L_r(x,\vec{\omega}_o) = \int_{\Omega} f_r(x,\vec{\omega}_o,\vec{\omega}_i)\frac{d^2\Phi(x,\vec{\omega}_i)}{dA\,d\vec{\omega}_i}d\omega_i$$

Radiance estimation – derivation III

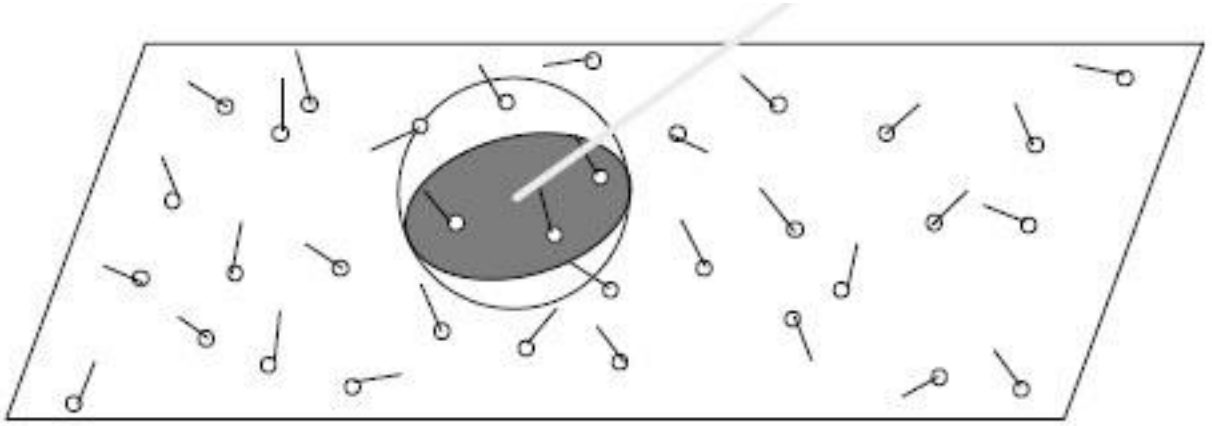
- Apply the nearest neighbour density estimation & multiply the samples by their flux and BRDF
- Assume that each photon in searched space ΔA hits the surface right at x:

$$L_r(x,\vec{\omega}_o) \approx \sum_{p=1}^{n} f_r(x,\vec{\omega}_o,\vec{\omega}_p)\frac{\Delta\Phi(x,\vec{\omega}_p)}{\Delta A}$$

- Photons look up can be imagined as expanding the sphere around x until it contains n photons
- Lets suppose that the area around the x is locally flat:

$$L_r(x,\vec{\omega}_o) \approx \frac{1}{\pi r^2}\sum_{p=1}^{n} f_r(x,\vec{\omega}_o,\vec{\omega}_p)\Delta\Phi(x,\vec{\omega}_p)$$

Radiance estimation – illustration



Final gather

- Lets split the BRDF

$$f_r(x, \vec{\omega}_o, \vec{\omega}_i) = f_{r,S}(x, \vec{\omega}_o, \vec{\omega}_i) + f_{r,D}(x, \vec{\omega}_o, \vec{\omega}_i)$$

- Lets split the incoming radiance

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

- Lets combine it into the expression for reflected radiance

$$L_r(x, \vec{\omega}_o) = \int_{\Omega} f_r(x, \vec{\omega}_o, \vec{\omega}_i) L(x, \vec{\omega}_i) \, |\vec{n}_x \cdot \vec{\omega}_i| \, d\omega_i$$

Reflected radiance equation

$$L_r(x,\vec{\omega}_o) = \int_\Omega f_r(x,\vec{\omega}_o,\vec{\omega}_i) L(x,\vec{\omega}_i) \,|\, \vec{n}_x \cdot \vec{\omega}_i \,|\, d\omega_i =$$

$$= \int_\Omega f_r(x,\vec{\omega}_o,\vec{\omega}_{in}) L_{i,l}(x,\vec{\omega}_{in}) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in} +$$

$$\int_\Omega f_{r,S}(x,\vec{\omega}_o,\vec{\omega}_{in}) (L_{i,c}(x,\vec{\omega}_{in}) + L_{i,d}(x,\vec{\omega}_{in})) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in} +$$

$$\int_\Omega f_{r,D}(x,\vec{\omega}_o,\vec{\omega}_{in}) L_{i,c}(x,\vec{\omega}_{in}) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in} +$$

$$\int_\Omega f_{r,D}(x,\vec{\omega}_o,\vec{\omega}_{in}) L_{i,d}(x,\vec{\omega}_{in}) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in}$$

Second pass

- Rendering

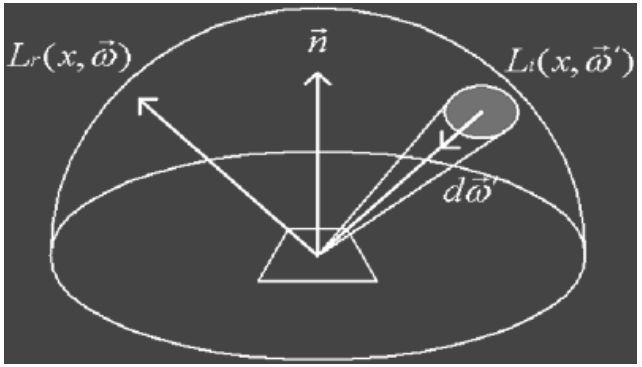
Photon mapping is used to calculate the effects of indirect lighting and caustics. Too many photons would be needed in the photon map to accurately handle specular/glossy surfaces and direct lighting.

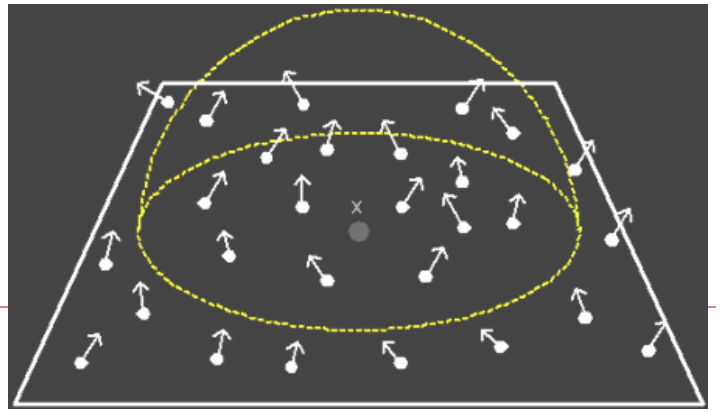
  - Approximating the Rendering Equation
  - Caustics
  - Indirect Illumination

Second pass

- Rendering
  - Approximating the Rendering Equation
  - Caustics
  - Indirect Illumination

$$L_r(x,\vec{\omega}) = \int_{\Omega} f_r(x,\vec{\omega}',\vec{\omega}) L_i(x,\vec{\omega}') \cos\theta_i d\vec{\omega}'$$





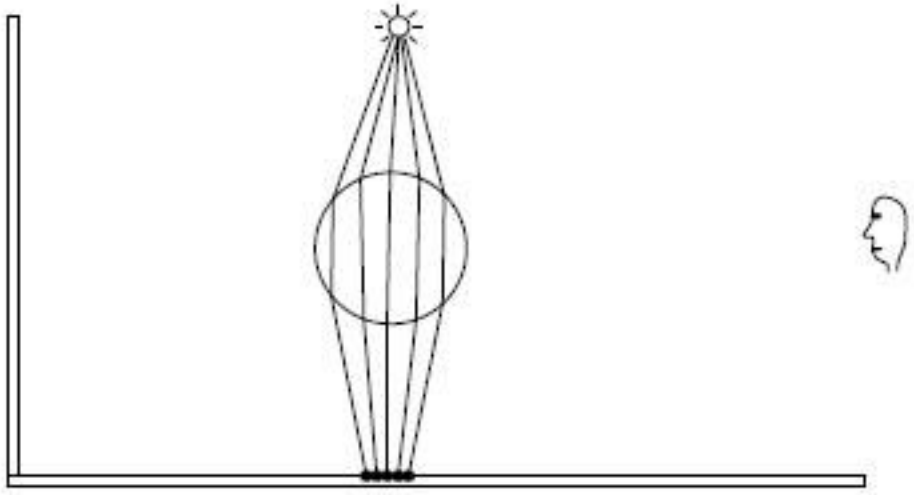
Second pass

- Rendering
  - Approximating the Rendering Equation
  - Caustics

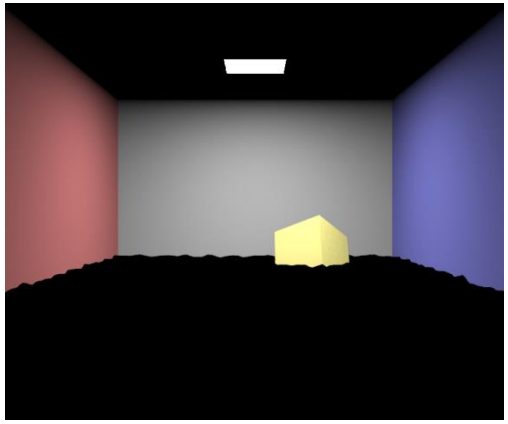
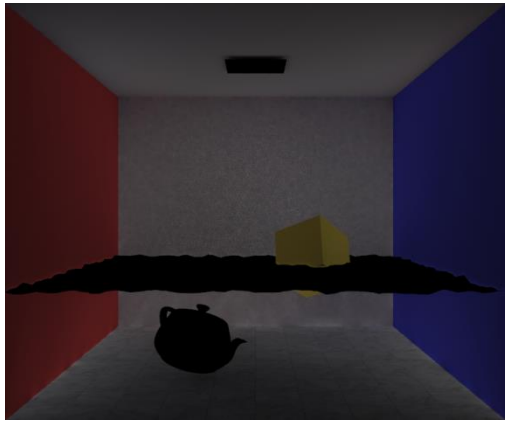
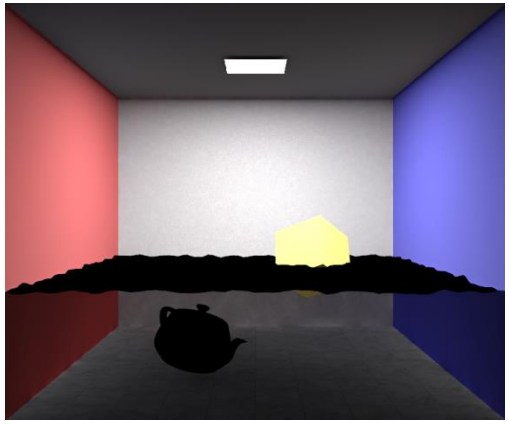
    Rendering caustics is fairly straightforward as you can directly visualize the photon map. Visualizing the photon map directly basically means we gather N nearest photons and estimate the radiance directly from that information using the last equation from the previous section. During the photon tracing phase a separate photon map is used for caustics.
  - Indirect Illumination

Creation of caustic

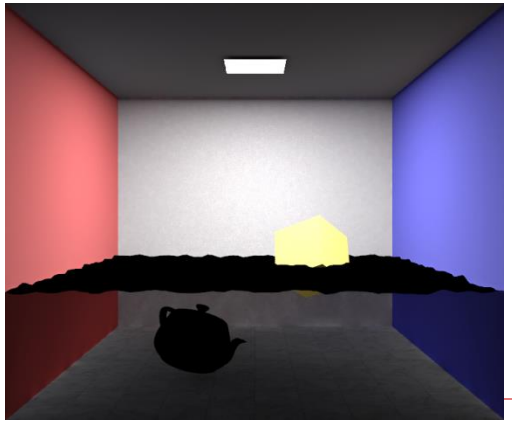
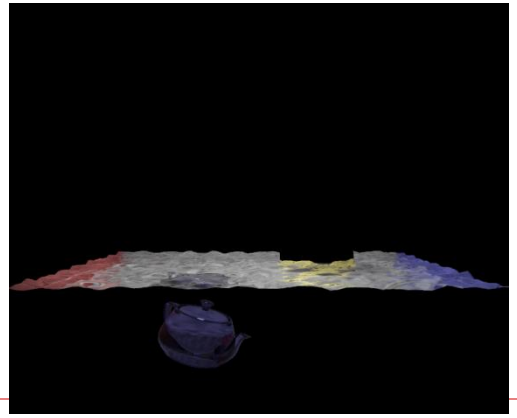
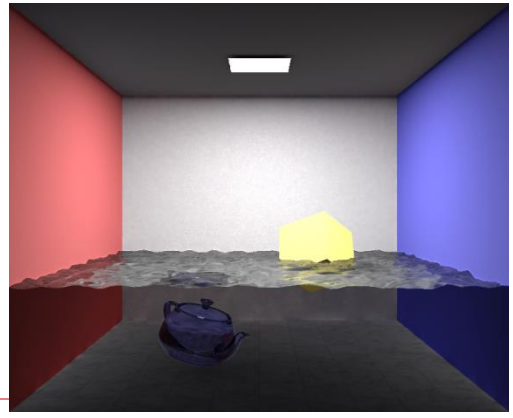
- Caustics appear on diffuse surfaces after light interaction with specular surfaces (both reflection and transmission)



Creation of caustic

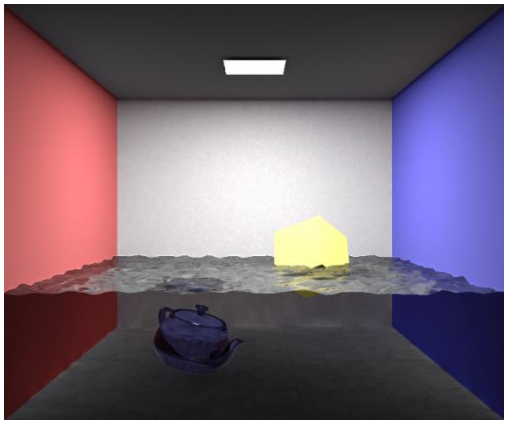

+

=


Direct Illumination Indirect Illumination

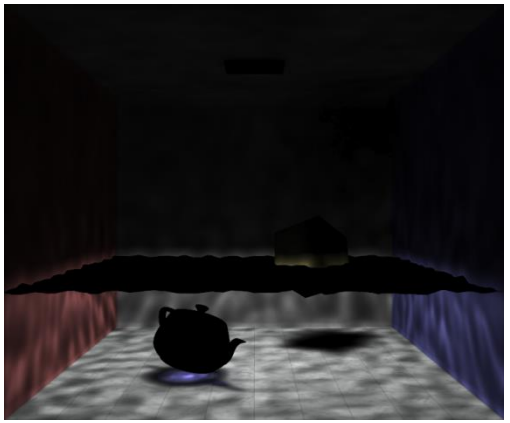

+

=


Specular Part

Creation of caustic

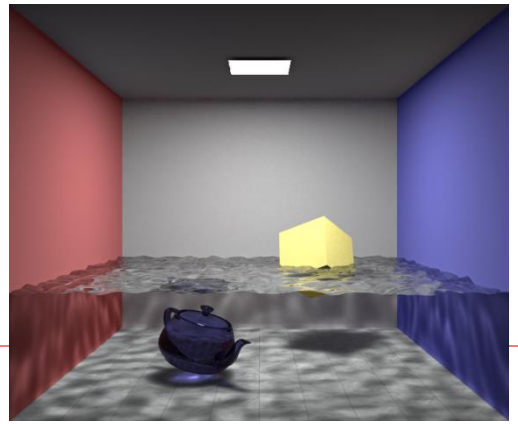


+



Caustics

=



More photon maps

- Create the special photon map for caustic photons
  - Trace them separately (store the photon on the first non-specular surface -> S+D paths)
  - Cast photons only against specular surfaces – light source projection maps
- Create the special photon map for indirect illumination
- Create the special photon map for participating media effects

Final gather II

- Direct illumination term

$$\int_{\Omega} f_r(x, \vec{\omega}_o, \vec{\omega}_{in}) L_{i,l}(x, \vec{\omega}_{in}) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in}$$

  - Solving directly by casting shadow rays (sampling the light sources)
- Specular and glossy term

$$\int_{\Omega} f_{r,S}(x, \vec{\omega}_o, \vec{\omega}_{in})(L_{i,c}(x, \vec{\omega}_{in}) + L_{i,d}(x, \vec{\omega}_{in})) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in}$$

  - Could take too much time if we waited until enough photons hit the BRDF narrow peak
- Caustics

$$\int_{\Omega} f_{r,D}(x, \vec{\omega}_o, \vec{\omega}_{in}) L_{i,c}(x, \vec{\omega}_{in}) \,|\, \vec{n}_x \cdot \vec{\omega}_{in} \,|\, d\omega_{in}$$

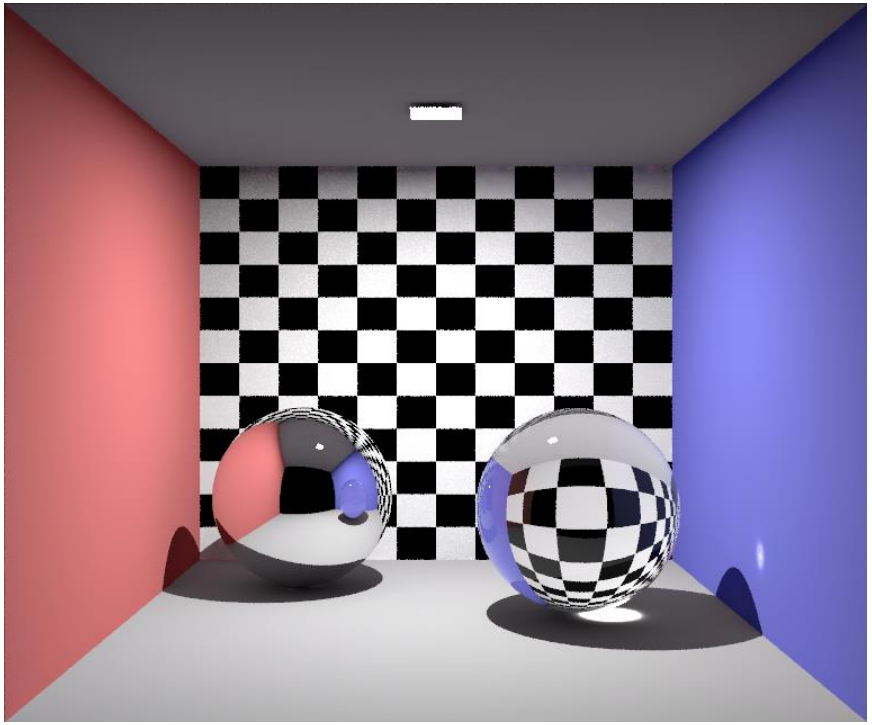
  - Direct query into the caustic photon map

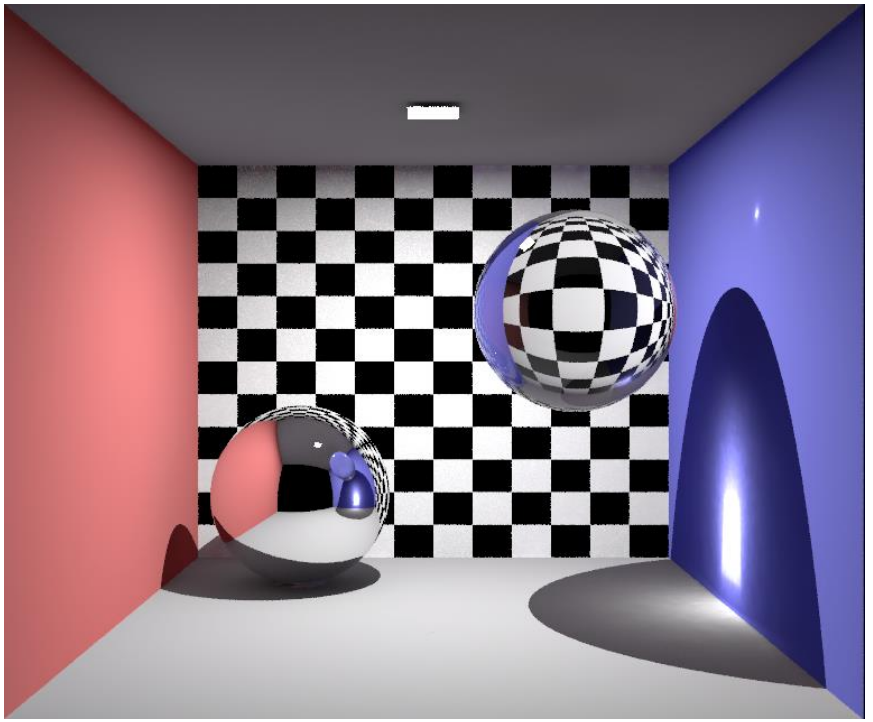
In what are the photon maps especially good at?

- Generating caustics
- Fast convergence of SDS paths

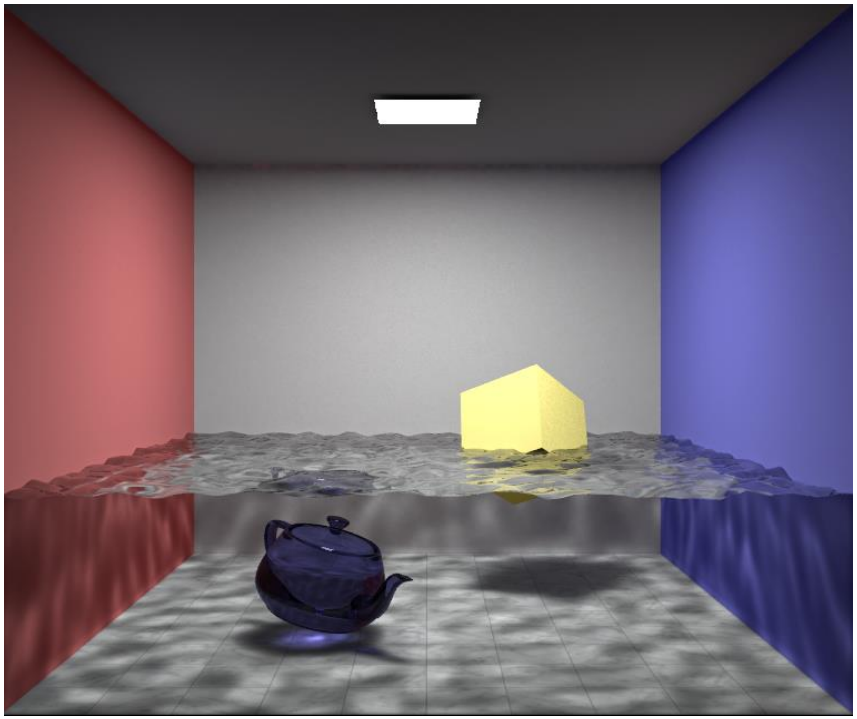
SDS - Bottom of the pool

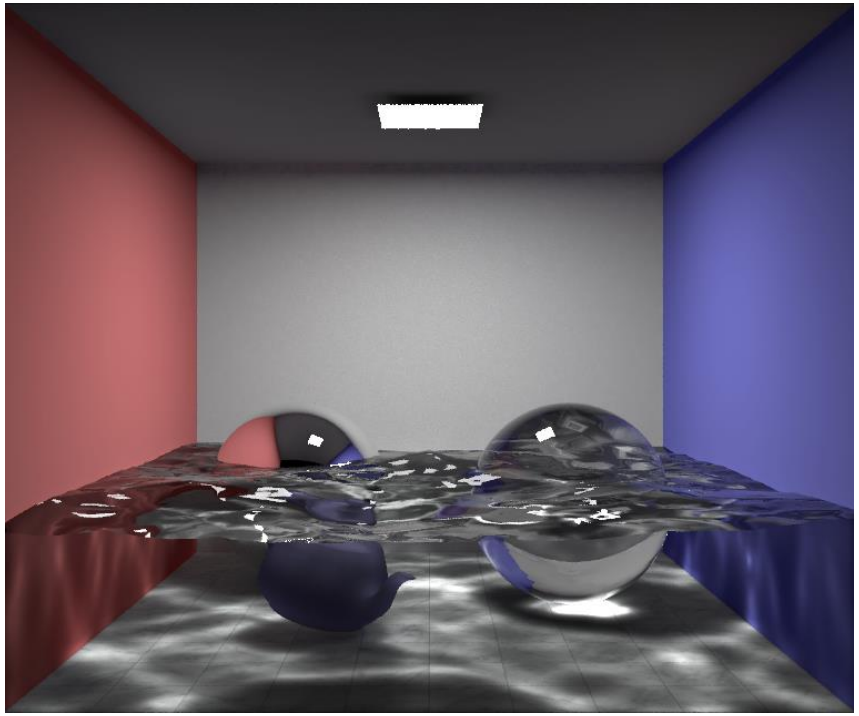
More results



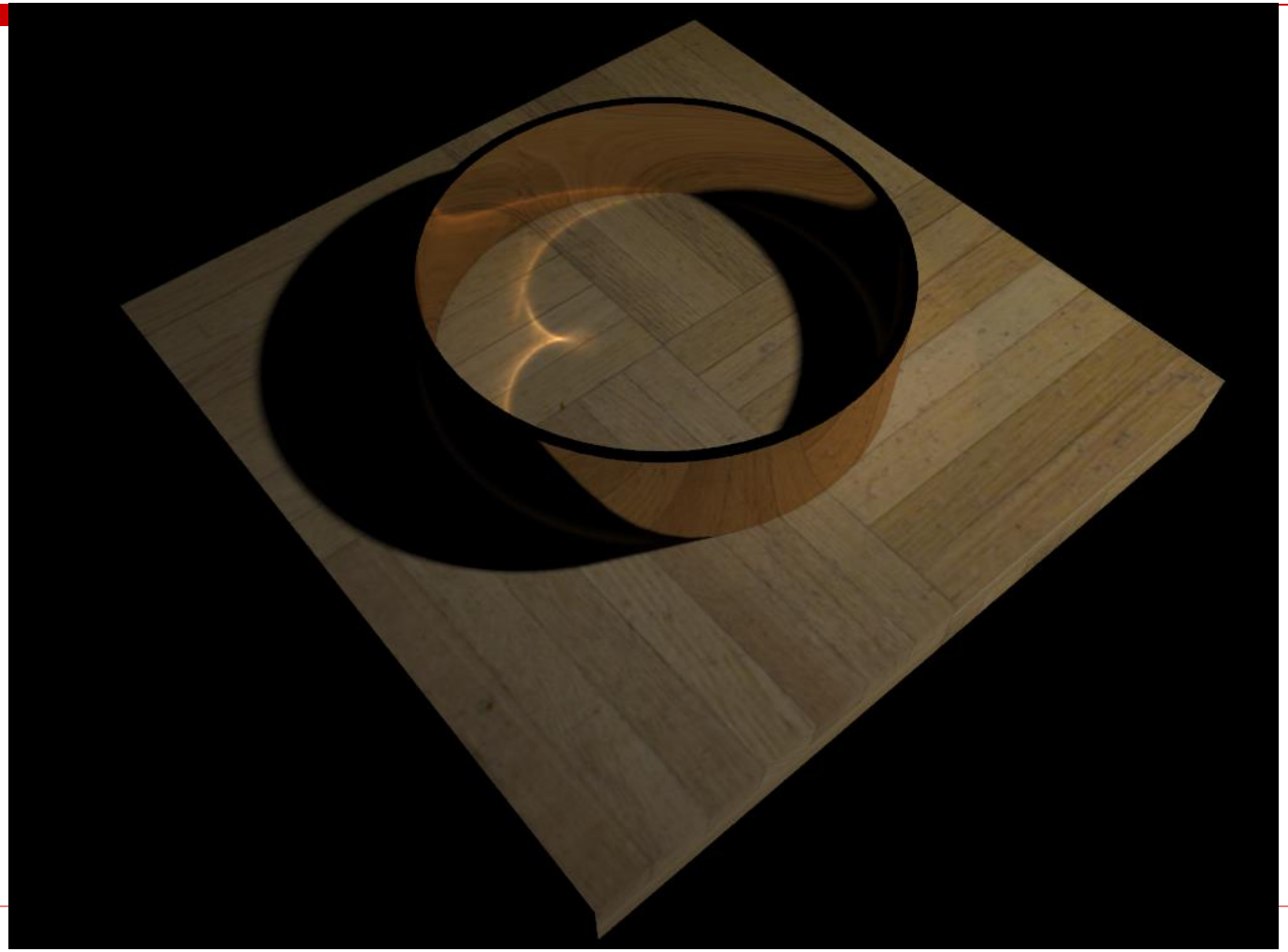


More results





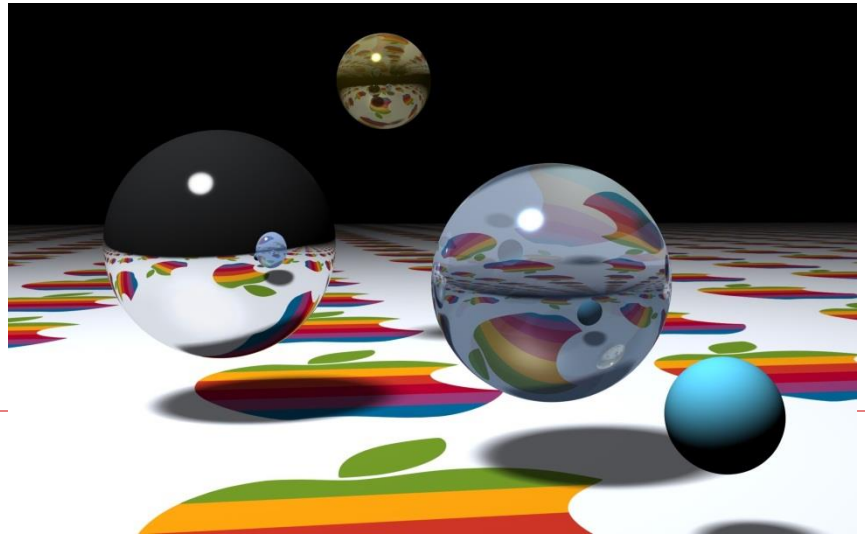
More results



Comparisons

Classical Ray Tracing

- Accurately account for specular reflections (for example, mirrors), and refraction through transparent materials
- Do not account for diffuse inter-reflections



Comparisons

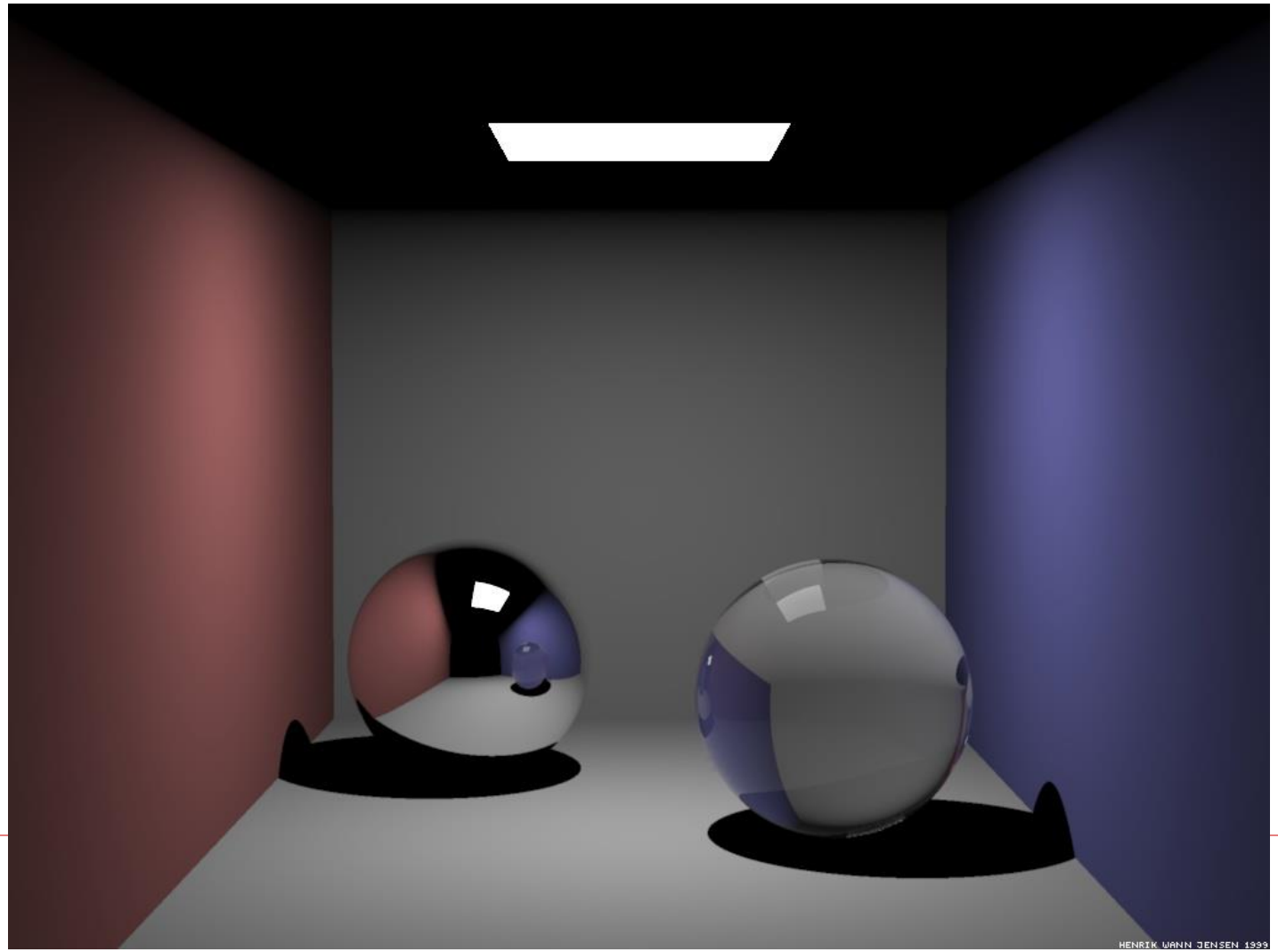
Radiosity

- Good for diffuse components
- Not suitable for rendering caustics

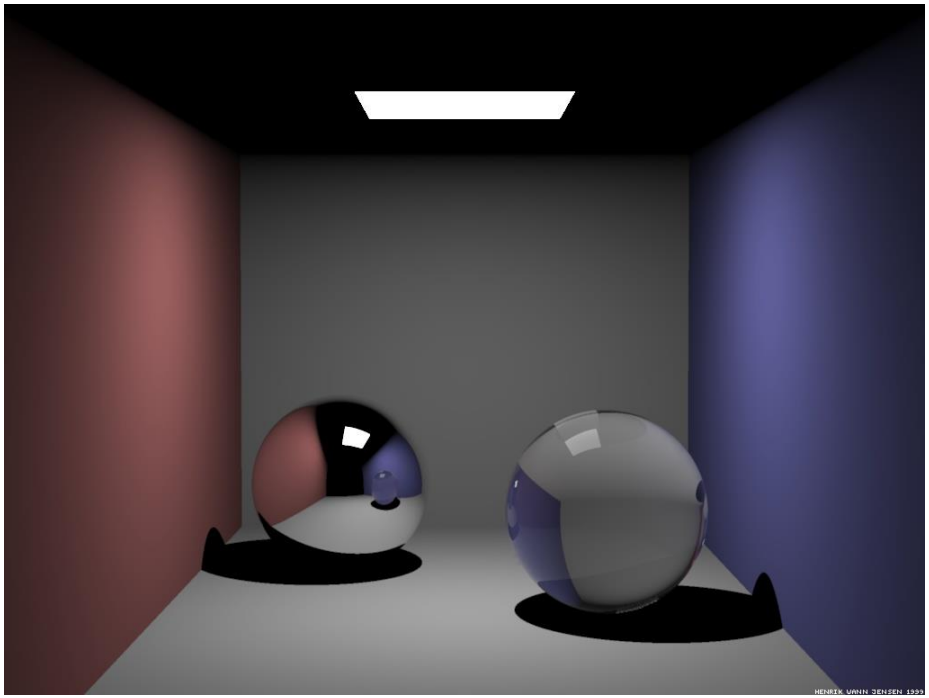
Ray tracing displays only specular highlights

Radiosity good for diffusion

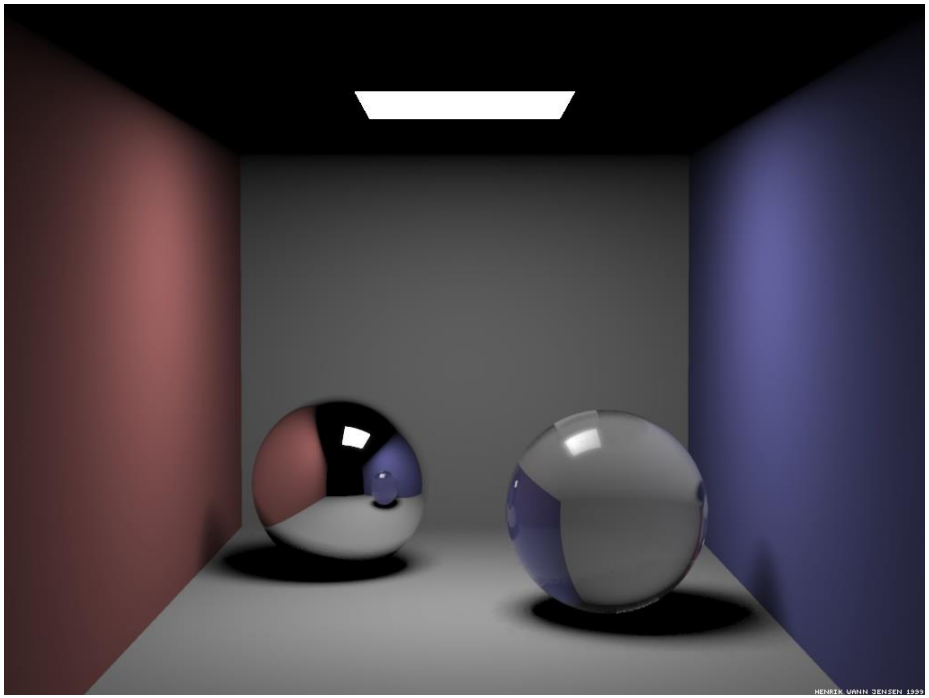
Ray tracing --- 1.5 seconds



Comparisons

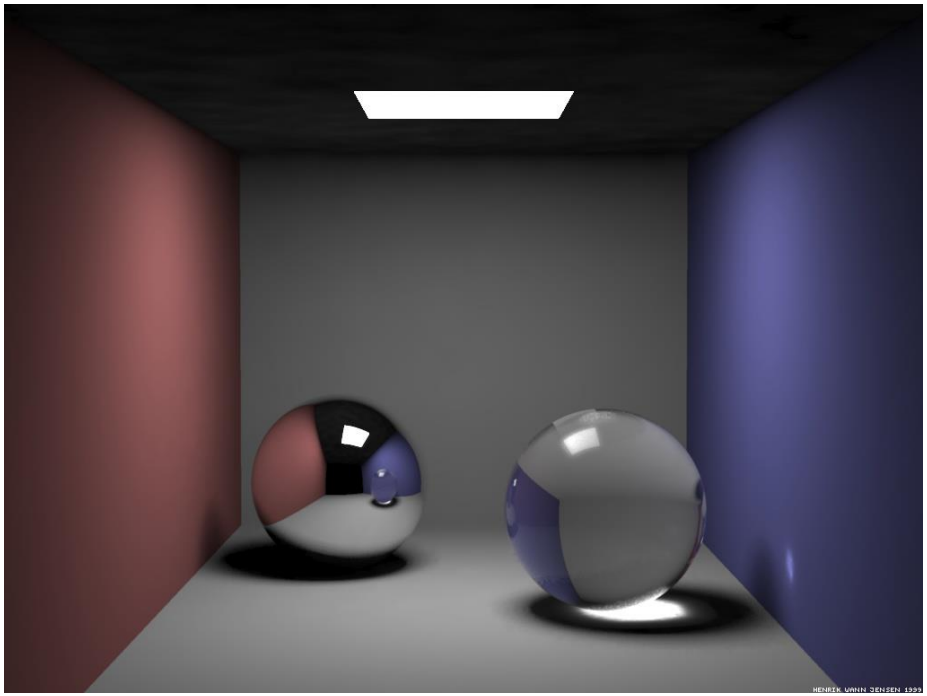


Ray tracing
1.5 seconds

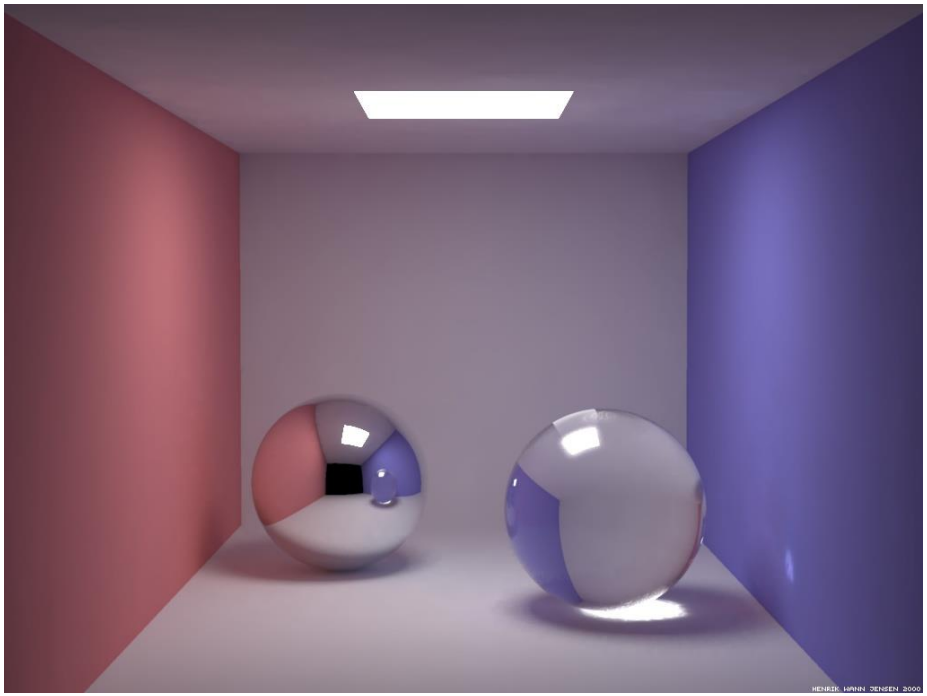


+ soft shadows
7 seconds

Comparisons

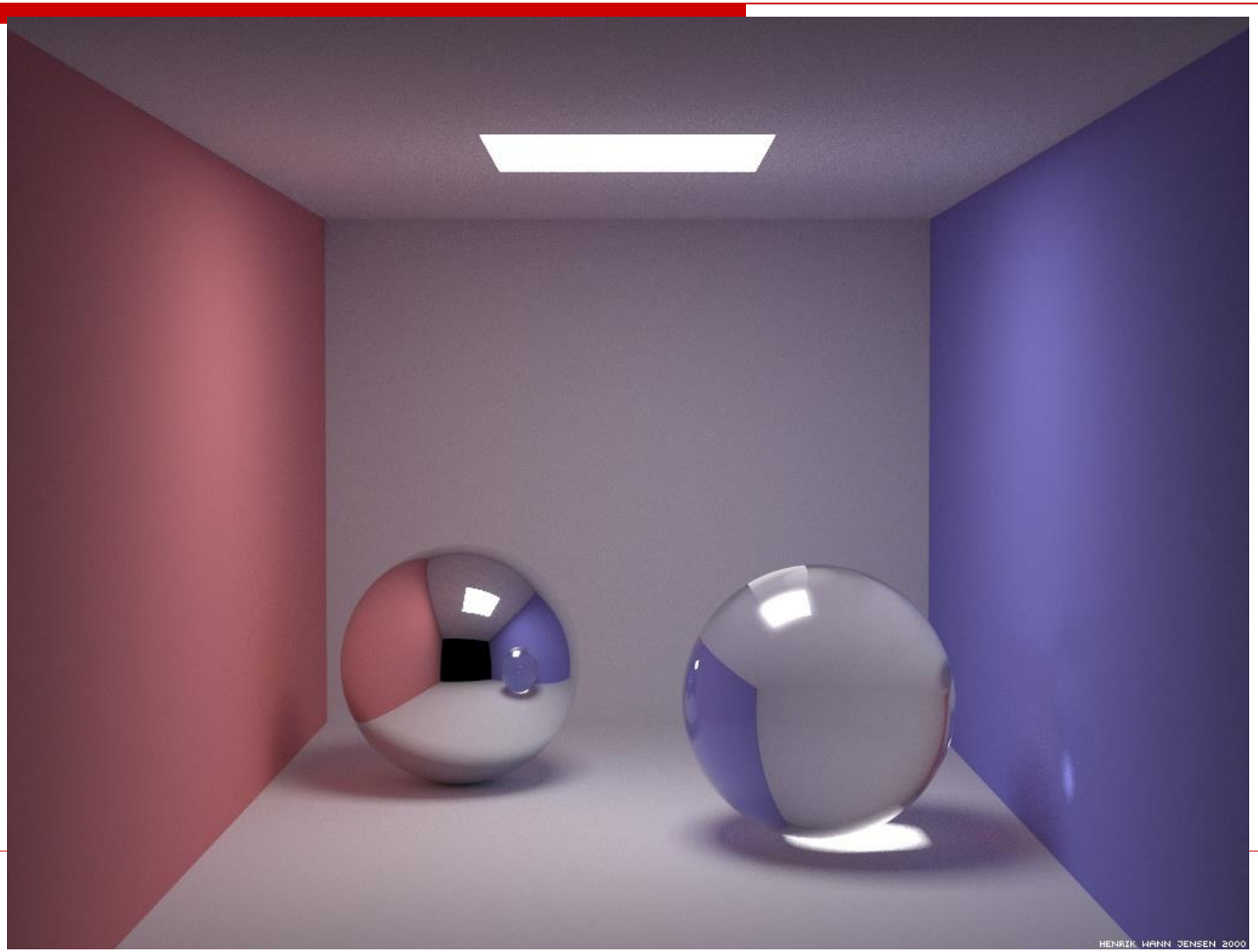


+ caustics
12 seconds



+ global illumination
15 seconds

Path tracing reference



Photon mapping simulation demo

This animation demonstrates how global illumination using photon mapping can be used to explore the light flow in a complex architectural model

References

- Henrik Wann Jensen: "Global Illumination using Photon Maps". In "Rendering Techniques '96". Eds. X. Pueyo and P. Schroder. Springer-Verlag, pp. 21-30, 1996
- Henrik Wann Jensen. Realistic Image Synthesis using Photon Mapping.
  - http://graphics.ucsd.edu/~henrik/papers/book/
  - ISBN: 1-56881-140-7. AK Peters, 2001
  - 2nd printing with corrections, March 2005

End.

Thanks