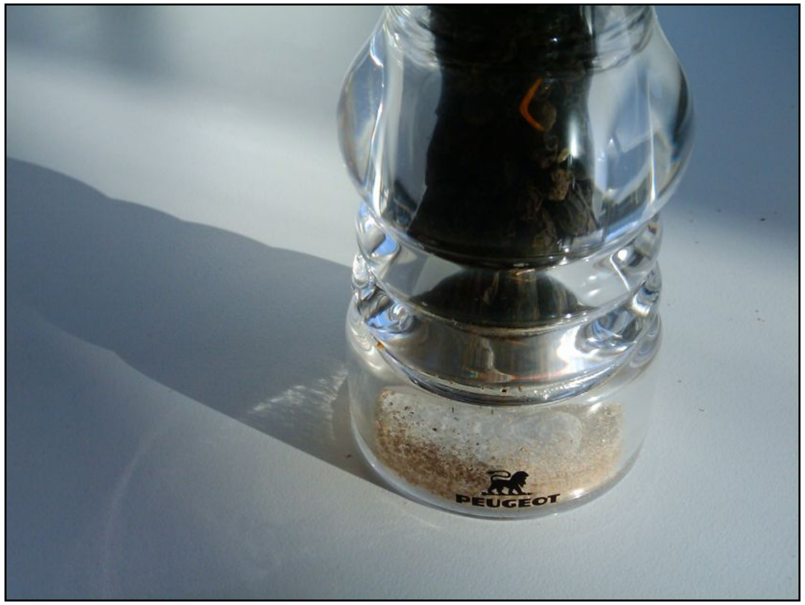
Appendix E: Photon Mapping

- Problem: shadow ray strikes transparent, refractive object.
  - Refracted shadow ray will now miss the light.
  - This destroys the validity of the boolean shadow test.
- Problem: light passing through a refractive object will sometimes form *caustics* (right), artifacts where the envelope of a collection of rays falling on the surface is bright enough to be visible.



This is a photo of a real pepper-shaker.
Note the caustics to the left of the shaker, in and outside of its shadow.
Photo credit: Jan Zankowski

Shadows, refraction and caustics

- Solutions for shadows of transparent objects:
  - Backwards ray tracing (Arvo)
    - *Very* computationally heavy
    - Improved by stencil mapping (Shenya et al)
  - Shadow attenuation (Pierce)
    - Low refraction, no caustics

- More general solution:
  - *Photon mapping* (Jensen)→

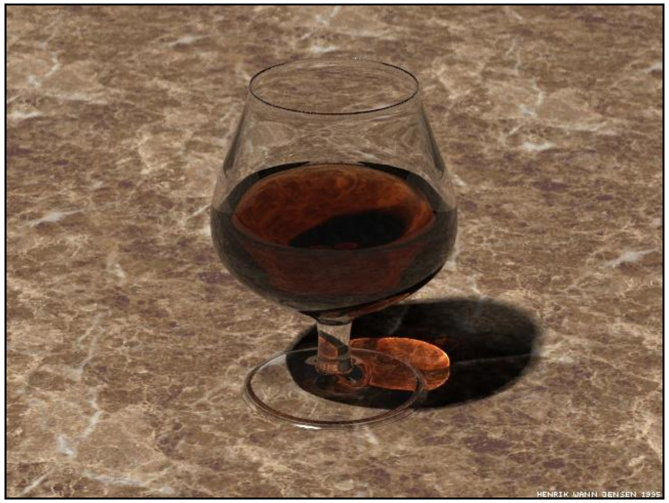


Image from http://graphics.ucsd.edu/~henrik/
Generated with photon mapping

Photon mapping

Photon mapping is the process of emitting photons into a scene and tracing their paths probabilistically to build a *photon map*, a data structure which describes the illumination of the scene independently of its geometry.

This data is then combined with ray tracing to compute the global illumination of the scene.

RENDERED USING DALI - HENRIK WANN JENSEN 2000

Image by Henrik Jensen (2000)

Photon mapping—algorithm (1/2)

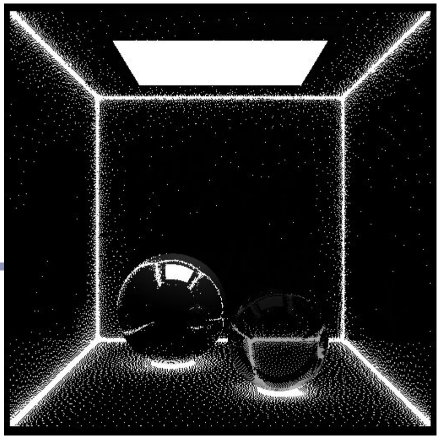


Image by Zack Waters

Photon mapping is a two-pass algorithm:

1. Photon scattering
   A. Photons are fired from each light source, scattered in randomly-chosen directions. The number of photons per light is a function of its surface area and brightness.
   B. Photons fire through the scene (re-use that raytracer, folks.) Where they strike a surface they are either absorbed, reflected or refracted.
   C. Wherever energy is absorbed, cache the location, direction and energy of the photon in the *photon map*. The photon map data structure must support fast insertion and fast nearest-neighbor lookup; a *kd-tree* is often used.

Photon mapping—algorithm (2/2)

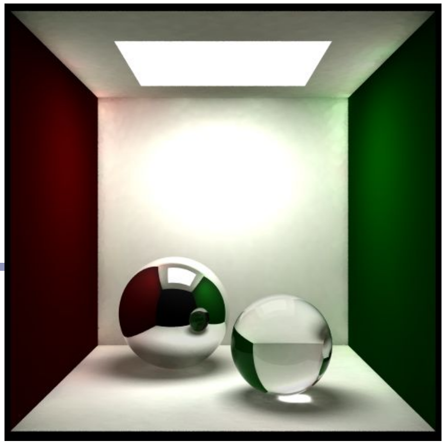


Image by Zack Waters

Photon mapping is a two-pass algorithm:

2. Rendering
   A. Ray trace the scene from the point of view of the camera.
   B. For each first contact point *P* use the ray tracer for specular but compute diffuse from the photon map and do away with ambient completely.
   C. Compute radiant illumination by summing the contribution along the eye ray of all photons within a sphere of radius *r* of *P*.
   D. Caustics can be calculated directly here from the photon map. For speed, the caustic map is usually distinct from the radiance map.

Photon mapping is probabilistic

This method is a great example of *Monte Carlo integration*, in which a difficult integral (the lighting equation) is simulated by randomly sampling values from within the integral's domain until enough samples average out to about the right answer.

- This means that you're going to be firing *millions* of photons. Your data structure is going to have to be <u>very</u> space-efficient.

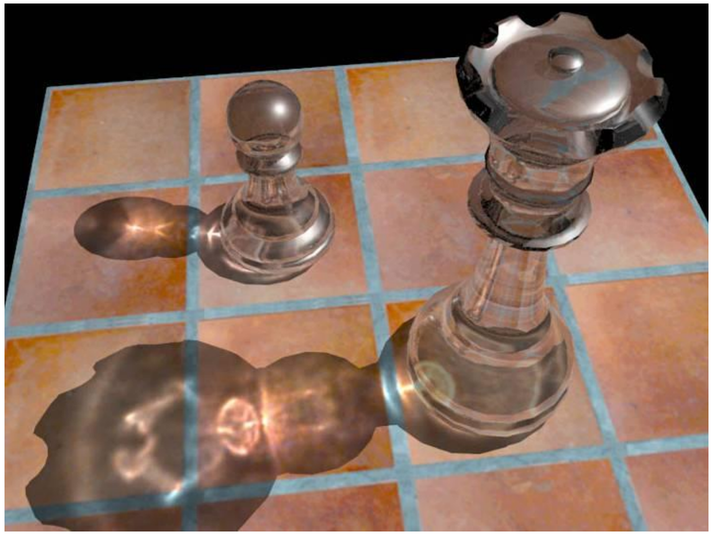
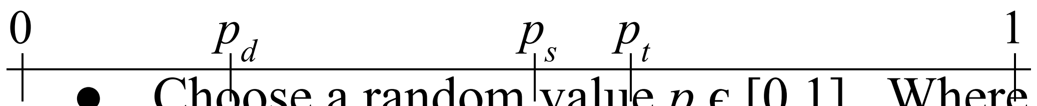


Image credit: http://www.okino.com/conv/imp_jt.htm

Photon mapping is probabilistic

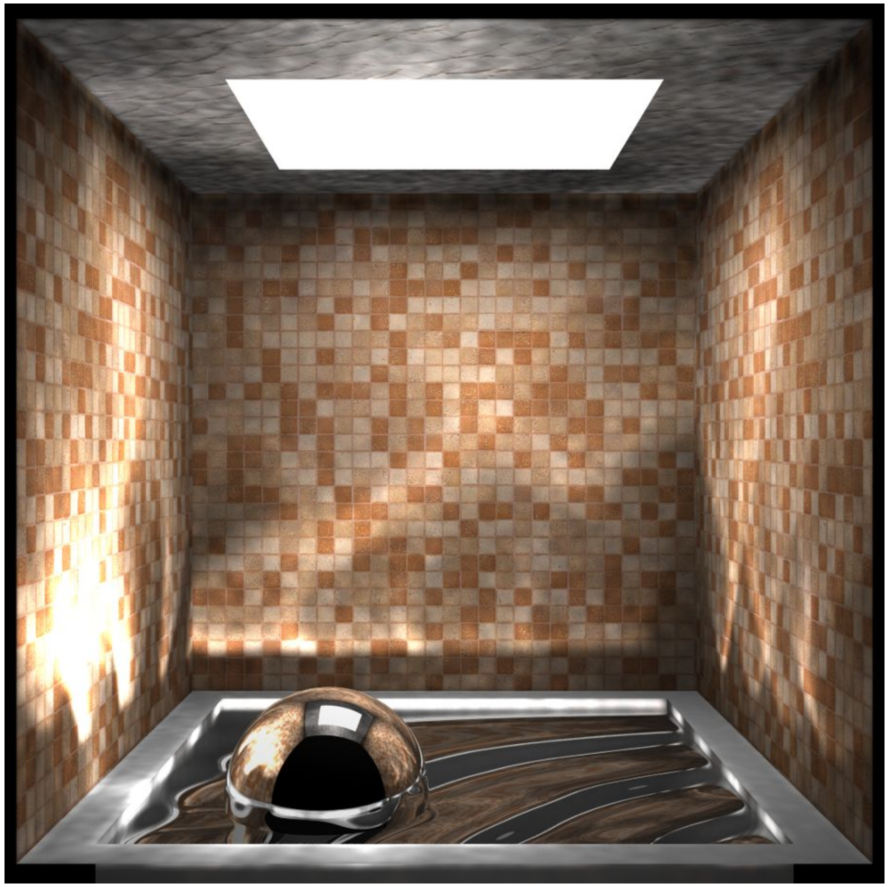
- Initial photon direction is random. Constrained by light shape, but random.
- What exactly happens each time a photon hits a solid also has a random component:
  - Based on the diffuse reflectance, specular reflectance and transparency of the surface, compute probabilities p_d, p_s and p_t where $(p_d+p_s+p_t)\leq 1$. This gives a probability map:



This surface would have minimal specular highlight.

  - Choose a random value $p \in [0,1]$. Where p falls in the probability map of the surface determines whether the photon is reflected, refracted or absorbed.

Photon mapping gallery



http://web.cs.wpi.edu/~emmanuel/courses/cs563/write_ups/zackw/photon_mapping/PhotonMapping.html

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

http://www.pbrt.org/gallery.php

Photon Mapping - References

- Henrik Jensen, “Global Illumination using Photon Maps”: http://graphics.ucsd.edu/~henrik/
- Henrik Jensen, “Realistic Image Synthesis Using Photon Mapping”
- Zack Waters, “Photon Mapping”: http://web.cs.wpi.edu/~emmanuel/courses/cs563/write_ups/zackw/photon_mapping/PhotonMapping.html