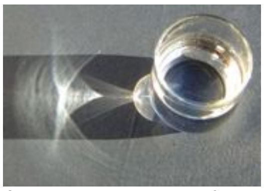
Introduction to Photon Mapping

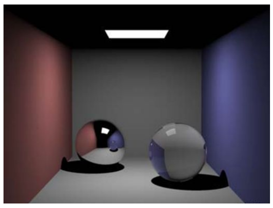


Caustics generated by a glass of water
Source: Wikipedia
Uploader: Heiner Otterstedt

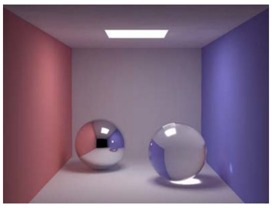
Caustics generated in a swimming pool
Source: epod.usra.edu
Photograph by: Menashe Davidson

- Ray tracing cannot produce caustics.
- Radiosity cannot produce caustics.

Why is ray tracing insufficient?

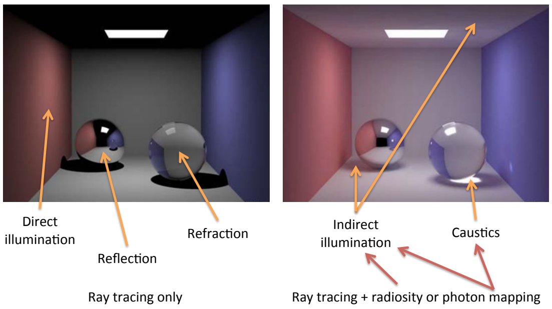


Basic ray traced solution



Ray traced image using information from photon mapping

What can we use to improve the ray tracing method?

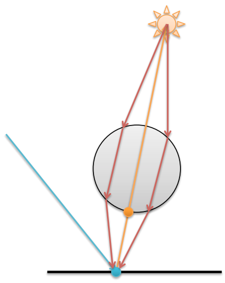


Ray tracing only

Ray tracing + radiosity or photon mapping

Shadows, refraction and caustics

- The problem:
  - the shadow ray strikes a transparent, refractive object
  - the ray tracing algorithm cannot trace backwards through objects to find light
  - so it returns "surface cannot see light"
- This destroys the validity of the boolean shadow test
- Similar problems occur with reflective objects



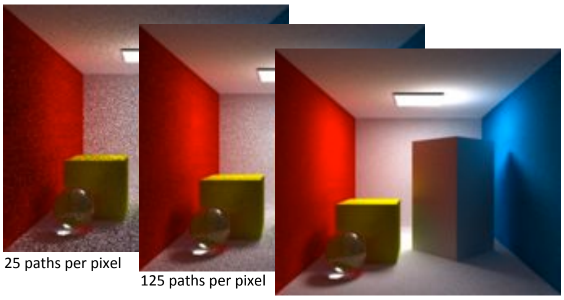
Some early solutions

- Shadow attenuation
  - Assume transparent objects do not refract!
  - Does not produce caustics
  - But looks close to right for objects with little refraction.
- Distribution ray tracing (Cook *et al.*, 1984)
  - Spawn a ray from the intersection point to randomly sample the BRDF
  - Need a lot (1,000 to 10,000) of rays per pixel to get this to give a good result.

Some more early solutions

- Backwards ray tracing (Arvo, 1986)
  - Trace rays from the lights ("backwards") rather than from the camera ("forewards")
  - Store the resulting intensity in texture maps attached to each object
  - Computationally expensive (lots of rays)
- Path tracing (Kajiya, 1986, Lafortune & Willems, 1993)
  - Do both backwards and forwards ray tracing
  - Connect the two together
  - Still computationally expensive

Why so expensive?



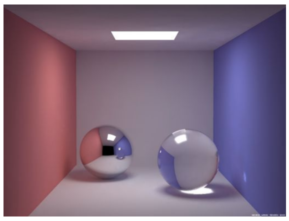
25 paths per pixel

125 paths per pixel

625 paths per pixel

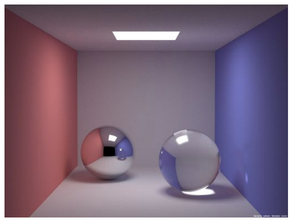
Images from http://www.thepolygoners.com/tutorials/GIIntro

Side-by-side comparison



Ray tracing + photon mapping

Faster (30 seconds)



Path tracing with 1000 rays (paths) per pixel

Slower (30 minutes)

Images from http://graphics.ucsd.edu/~henrik/

Global Illumination: Photon mapping

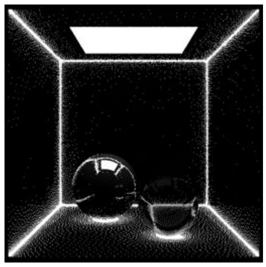
- emit photons into a scene
- trace their paths probabilistically
- build a "photon map": a data structure that describes the illumination of the scene independent of its geometry
- combine the photon map data with ray tracing to compute the global illumination of the scene.

Image by Henrik Jensen (2000)

Photon mapping—Algorithm (1/2)

1. Photon scattering

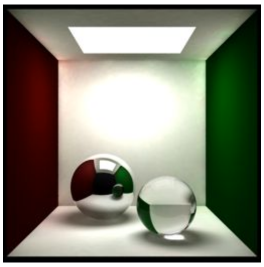
- Photons are fired from each light source, scattered in randomly-chosen directions. The number of photons per light is a function of its surface area and brightness.
- Photons fire through the scene (use your ray-tracing code). Where they strike a surface they are either absorbed, reflected or refracted.
- Wherever energy is absorbed, cache the 3D location, the direction and the 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 common.



Photon mapping algorithm (2/2)

2. Rendering

- Ray trace the scene from the point of view of the camera.
- 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.
- Compute radiant illumination by summing the contribution along the eye ray of all photons within a sphere of radius *r* of *P*.
- Caustics can be calculated directly here from the photon map. For speed, the caustic map is usually distinct from the radiance map.



Images from http://web.cs.wpi.edu/~emmanuel/courses/cs563/

Photon mapping

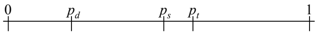
- This method is an example of Monte Carlo integration, in which a difficult integral (the lighting equation) is simulated by randomly sampling values from within the integral's domain until enough samples average out to about the right answer.
- This means that you're going to be firing millions of photons. Your data structure is going to have to be very space-efficient!

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

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

Photon mapping: some details

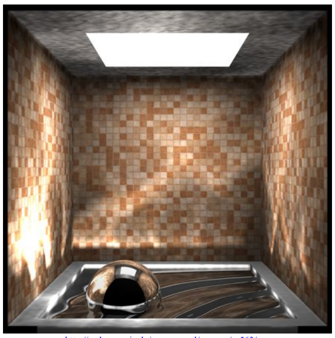
- Initial photon direction is random. Constrained by light shape, but random.
- What exactly happens each time a photon hits a solid also has a random component:
- Based on the diffuse reflectance, specular reflectance and transparency of the surface, compute probabilities p_d, p_s and p_t where $(p_d+p_s+p_t)\leq1$. 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

References

- Radiosity
  - nVidia: http://http.developer.nvidia.com/GPUGems2/gpugems2_chapter39.html
  - Cornell: http://www.graphics.cornell.edu/online/research/
  - Wallace, J. R., K. A. Elmquist, and E. A. Haines. 1989, "A Ray Tracing Algorithm for Progressive Radiosity." In Computer Graphics (Proceedings of SIGGRAPH 89) 23(4), pp. 315–324.
  - Buss, "3-D Computer Graphics: A Mathematical Introduction with OpenGL" (Chapter XI), Cambridge University Press (2003)
- Photon mapping
  - Henrik Jenson, "Global Illumination using Photon Maps", http://graphics.ucsd.edu/~henrik/
  - Zack Waters, "Photon Mapping", http://web.cs.wpi.edu/~emmanuel/courses/cs563/write_ups/zackw/photon_mapping/PhotonMapping.html