Monte Carlo Ray Tracing & Irradiance Caching & Photon Mapping

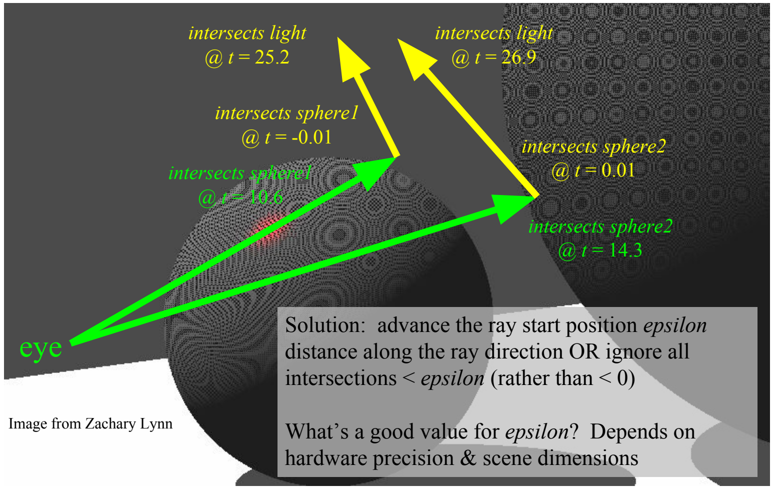
Announcements: Quiz

- On Friday (3/10), in class
- One 8.5x11 sheet of notes allowed
- Sample quiz (from a previous year) on website
- Focus on "reading comprehension" and material for Homeworks 0, 1, & 2

Announcements: Final Projects

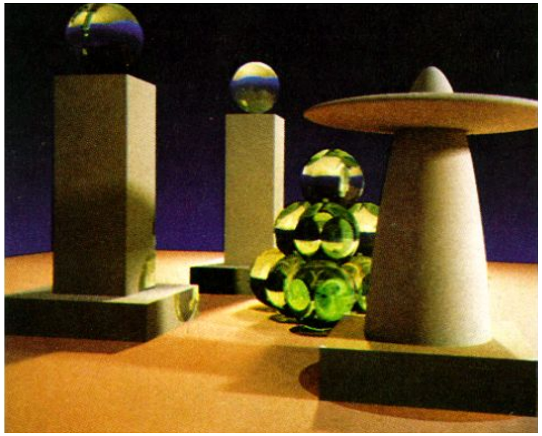
- *Everyone* should post two ideas for a final project on LMS ("due" Monday 3/20 @ 11:59pm)
- Connect with potential teammates (teams of 2 strongly recommended)
- Start finding & reading background papers
- Proposal & summary of background research are due April 3rd
- *See webpage for details on brainstorming post, proposal, and overall project requirements*

HW3: Raytracing & Epsilon



Reading for Today

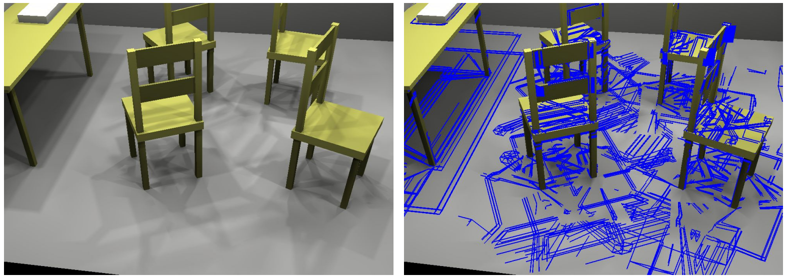
- "The Rendering Equation", Kajiya, SIGGRAPH 1986



Reading for Today

"Implicit Visibility and Antiradiance for Interactive Global Illumination"
Dachsbacher, Stamminger, Drettakis, and Durand
Siggraph 2007

Reading for Today



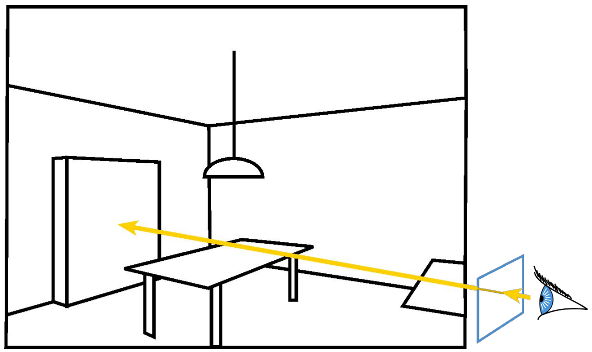
"Fast and Accurate Hierarchical Radiosity Using Global Visibility"
Durand, Drettakis, & Puech 1999

Today

- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
- Importance Sampling

Ray Casting

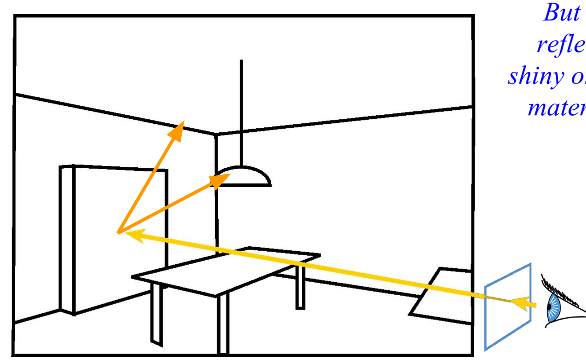
- Cast a ray from the eye through each pixel



Ray Tracing

- Cast a ray from the eye through each pixel
- Trace secondary rays (light, reflection, refraction)

But only reflect off shiny or glossy materials...

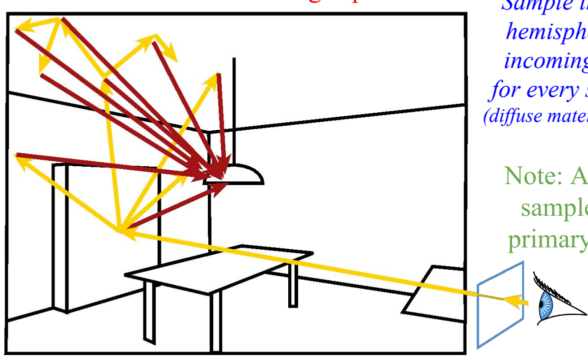


Monte Carlo Ray Tracing

- Cast a ray from the eye through each pixel
- Cast random rays to accumulate radiance contribution
  - Recurse to solve the Rendering Equation

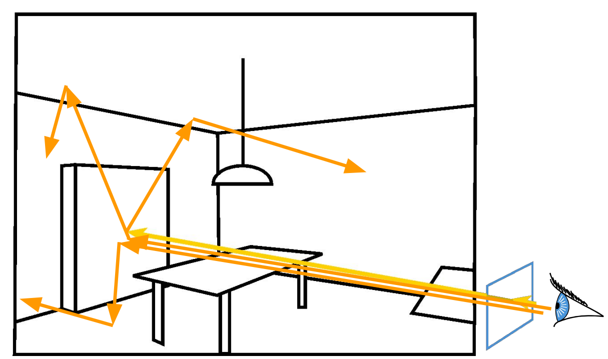
Sample the full hemisphere of incoming light for every surface
(diffuse materials too!)

Note: Always sample the primary light

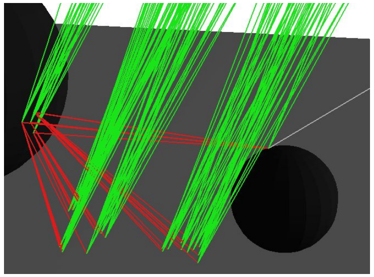


(Monte Carlo) Path Tracing

- Trace only one secondary ray per recursion
- But send many primary rays per pixel (performs antialiasing as well)



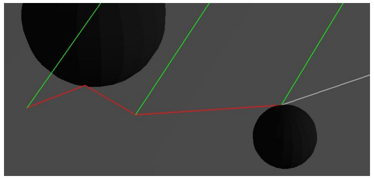
Ray Tracing vs. Path Tracing



2 bounces
5 glossy samples
5 shadow samples

How many rays cast per pixel?

1 main ray + 5 shadow rays +
5 glossy rays + 5x5 shadow rays +
5*5 glossy rays + 5x5x5 shadow rays
= 186 rays



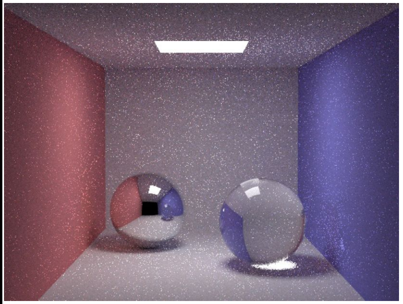
How many 3 bounce paths can we trace per pixel for the same cost?

186 rays / 8 ray casts per path
= ~23 paths

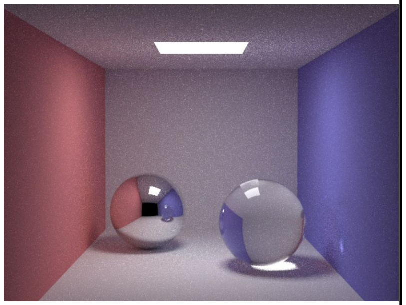
Which will probably have less error?

Questions?

10 paths/pixel



100 paths/pixel



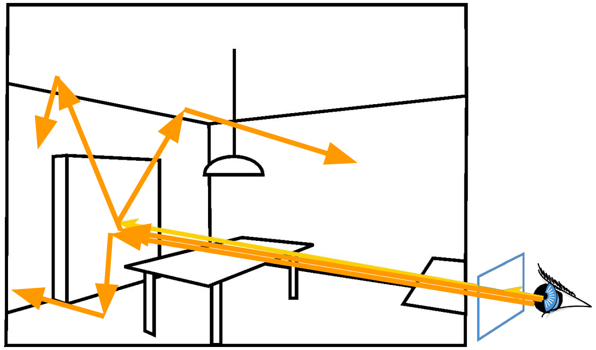
Images from Henrik Wann Jensen

Today

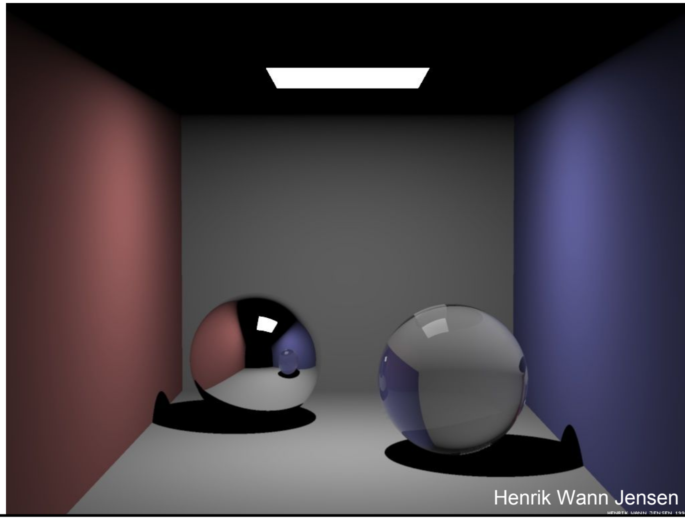
- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
- Importance Sampling

Path Tracing is costly

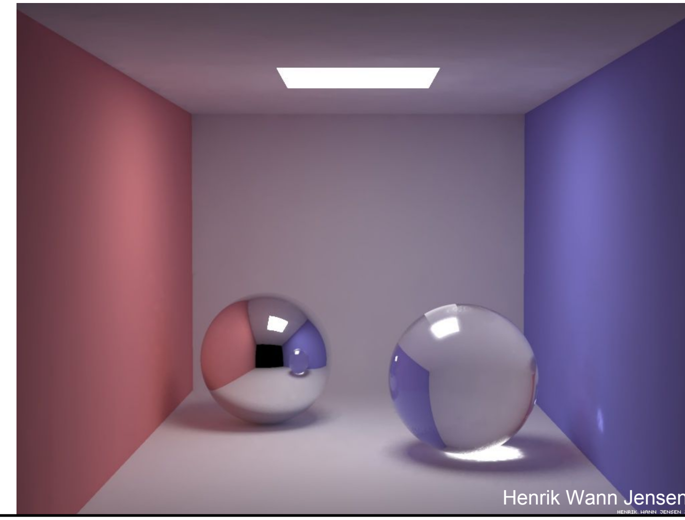
- Needs tons of rays per pixel



Direct Illumination



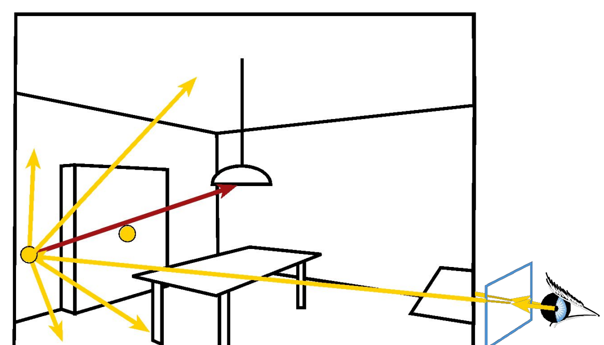
Global Illumination



Indirect Illumination: smooth

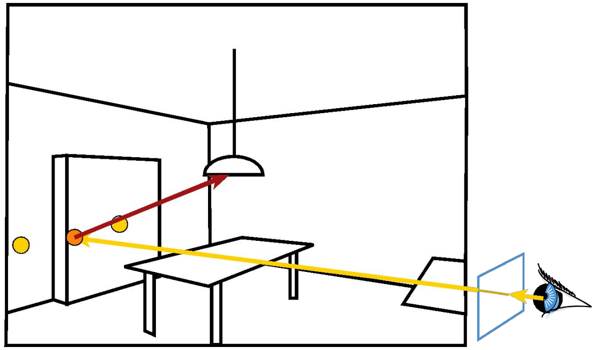
Irradiance Cache

- The indirect illumination is smooth
- Store the indirect illumination

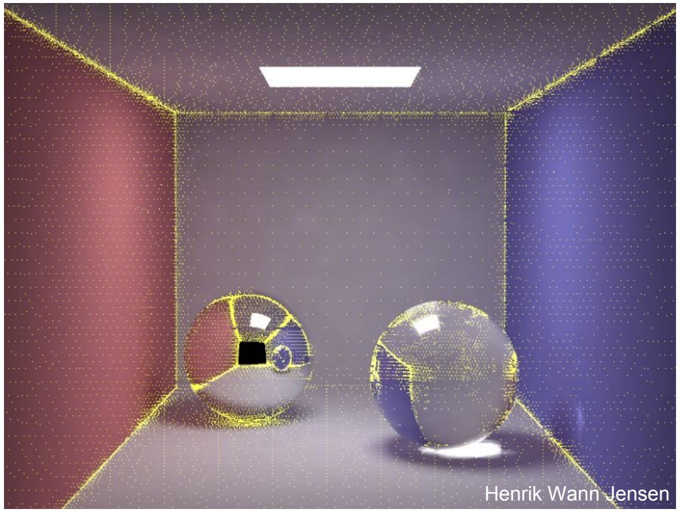


Irradiance Cache

- Interpolate nearby cached values
- But do full calculation for direct lighting

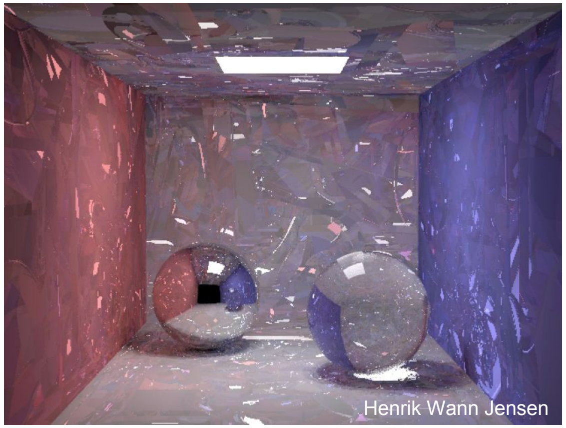


Irradiance Cache



Questions?

- Why do we need "good" random numbers?
  - With a fixed random sequence, we see the structure in the error

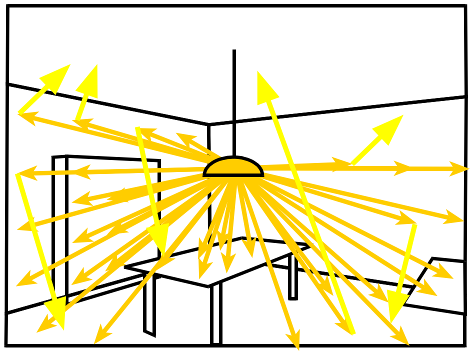


Today

- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
- Importance Sampling

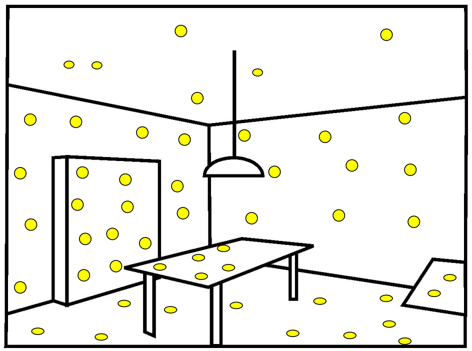
Photon Mapping

- Preprocess: cast rays from light sources
  - independent of viewpoint



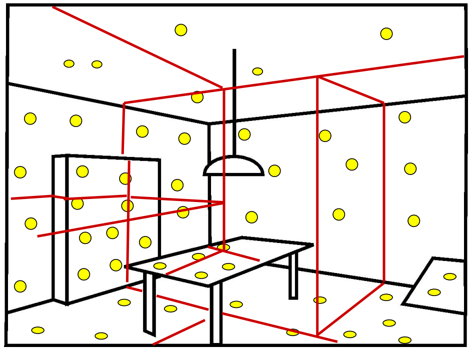
Photon Mapping

- Store photons
  - position + light power + incoming direction



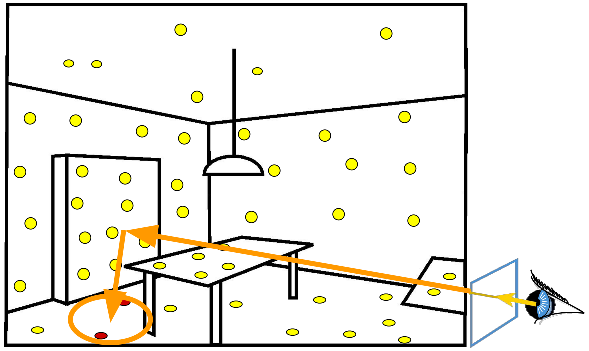
Storing the Photon Map

- Efficiently store photons for fast access
- Use hierarchical spatial structure (kd-tree)

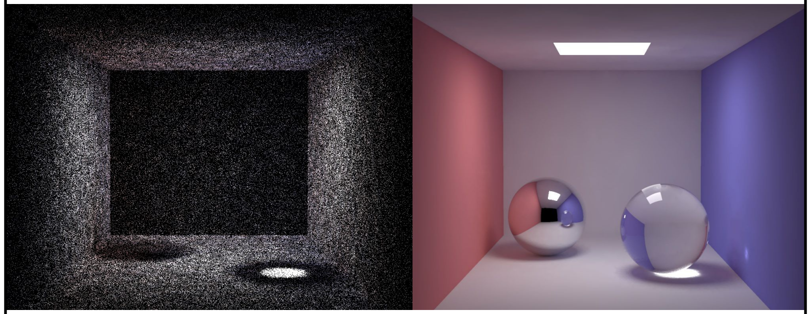


Rendering with Photon Map

- Cast primary rays
- For secondary rays: reconstruct irradiance using k closest photons
- Combine with irradiance caching and other techniques

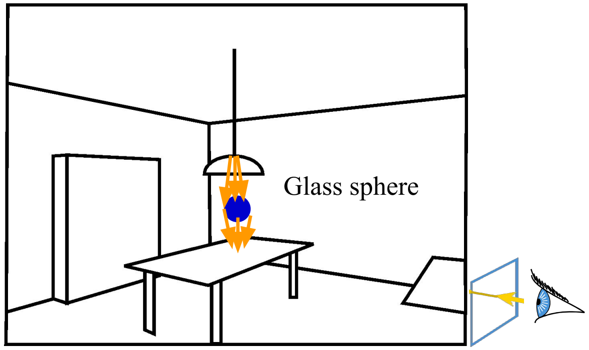


Photon Map Results



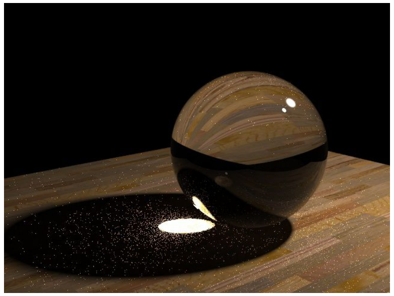
Photon Mapping - Caustics

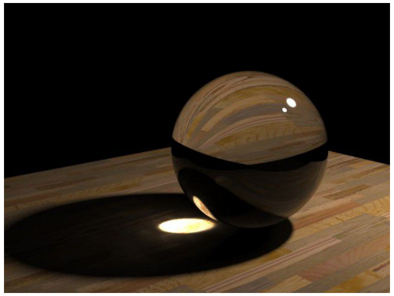
- Special photon map for specular reflection and refraction



Comparison

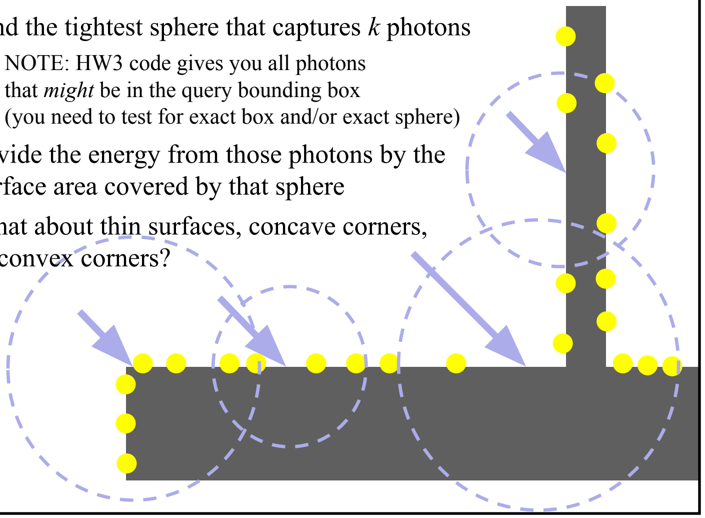
Path Tracing 1000 paths/pixel	Photon mapping



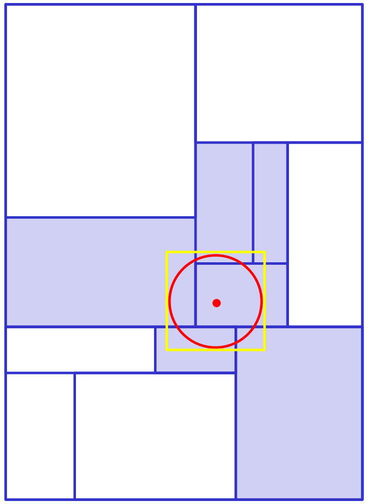


Closest Photon Details

- Find the tightest sphere that captures *k* photons
  - NOTE: HW3 code gives you all photons that *might* be in the query bounding box (you need to test for exact box and/or exact sphere)
- Divide the energy from those photons by the surface area covered by that sphere
- What about thin surfaces, concave corners, & convex corners?



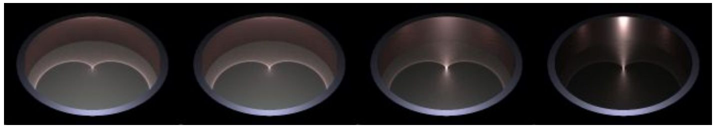
HW3: Photons in the k-d tree



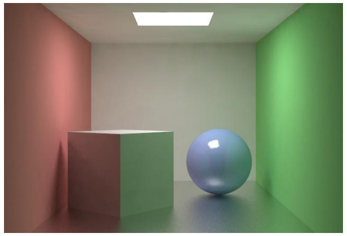
- You start with query point & radius (red)
- You give the `KDTree::CollectPhotonsInBox` function a bounding box (yellow)
- The algorithm finds all k-d tree cells that overlap with bounding box (blue)
- The function returns all photons in those cells
- *You need to discard all photons not in your original query radius*

Readings for Tuesday after break:

- "Rendering Caustics on Non-Lambertian Surfaces", Henrik Wann Jensen, *Graphics Interface* 1996.



- "Global Illumination using Photon Maps", Henrik Wann Jensen, *Rendering Techniques* 1996.

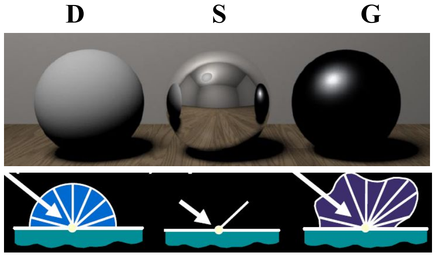


Today

- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
- Importance Sampling

Ray Grammar

- Classify local interaction:
  - E = eye
  - L = light
  - S = perfect specular reflection or refraction
  - G = glossy scattering
  - D = diffuse scattering

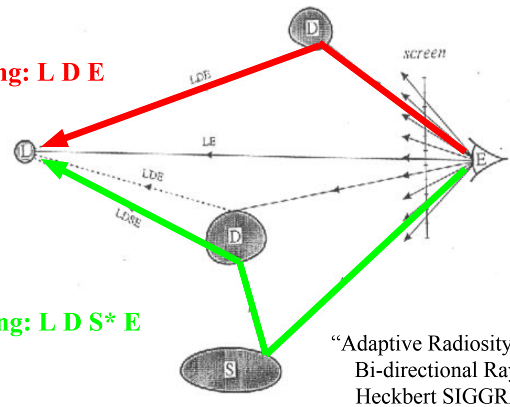


From Dutre et al.'s slides

Classic Ray Casting/Tracing

Ray casting: L D E

Ray tracing: L D S* E

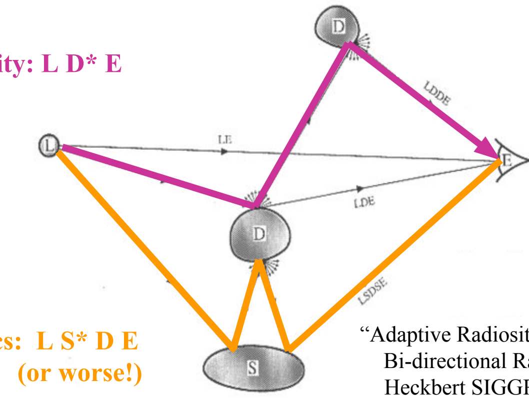


"Adaptive Radiosity Textures for Bi-directional Ray Tracing" Heckbert SIGGRAPH 1990

Photon Tracing

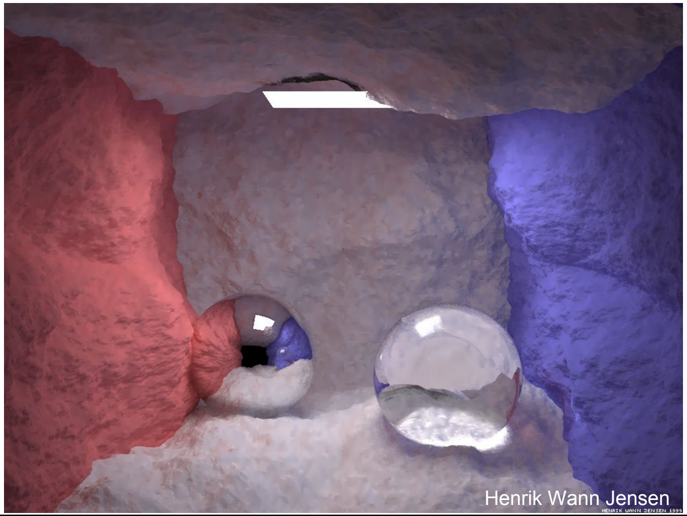
Radiosity: L D* E

Caustics: L S* D E (or worse!)



"Adaptive Radiosity Textures for Bi-directional Ray Tracing" Heckbert SIGGRAPH 1990

Questions?

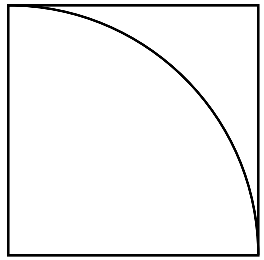


Today

- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
  - Probabilities and Variance
  - Analysis of Monte-Carlo Integration
- Importance Sampling

Monte-Carlo Computation of π

- Take a random point (x,y) in unit square
- Test if it is inside the ¼ disc
  - Is $x^2 + y^2 < 1$?
- Probability of being inside disc?
  - area of ¼ unit circle / area of unit square = π /4

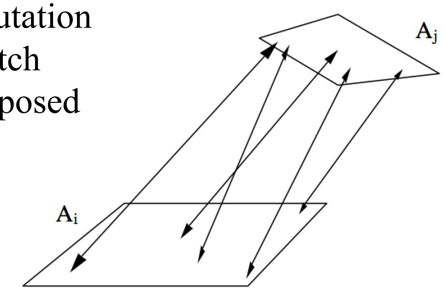


- π ≈ 4 * number inside disc / total number
- The error depends on the number or trials

Use MC to calculate Form Factor

- Cast *n* rays between the two patches
  - Compute visibility (what fraction of rays do not hit an occluder)
  - Integrate the point-to-point form factor

- Permits the computation of the patch-to-patch form factor, as opposed to point-to-patch



Convergence & Error

- Let's compute 0.5 by flipping a coin:
  - 1 flip: 0 or 1
    → average error = 0.5
  - 2 flips: 0, 0.5, 0.5 or 1
    → average error = 0. 25
  - 4 flips: 0 (*1),0.25 (*4), 0.5 (*6), 0.75(*4), 1(*1)
    → average error = 0.1875
- Unfortunately, doubling the number of samples does not double accuracy

Review of (Discrete) Probability

- Random variable can take discrete values x_i
- Probability p_i for each x_i

  $0 < p_i < 1, \quad \Sigma\, p_i = 1$
- Expected value $$E(x) = \sum_{i=1}^{n} p_i x_i$$
- Expected value of function of random variable
  - $f(x_i)$ is also a random variable

$$E[f(x)] = \sum_{i=1}^{n} p_i f(x_i)$$

Variance & Standard Deviation

- Variance σ^2: deviation from expected value
- Expected value of square difference

$$\sigma^2 = E[(x - E[x])^2] = \sum_i (x_i - E[x])^2 p_i$$

- Also

$$\sigma^2 = E[x^2] - (E[x])^2$$

- Standard deviation σ:
  square root of variance (notion of error, RMS)

Monte Carlo Integration

- Turn integral into finite sum
- Use *n* random samples
- As *n* increases…
  - Expected value remains the same
  - Variance decreases by *n*
  - Standard deviation (error) decreases by $\frac{1}{\sqrt{n}}$
- Thus, converges with $\frac{1}{\sqrt{n}}$

Advantages of MC Integration

- Few restrictions on the integrand
  - Doesn't need to be continuous, smooth, ...
  - Only need to be able to evaluate at a point
- Extends to high-dimensional problems
  - Same convergence
- Conceptually straightforward
- Efficient for solving at just a few points

Disadvantages of MC Integration

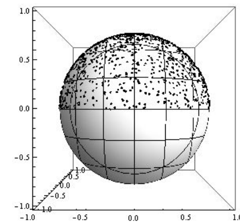
- Noisy
- Slow convergence
- Good implementation is hard
  - Debugging code
  - Debugging math
  - Choosing appropriate techniques
- Punctual technique, no notion of smoothness of function (e.g., between neighboring pixels)

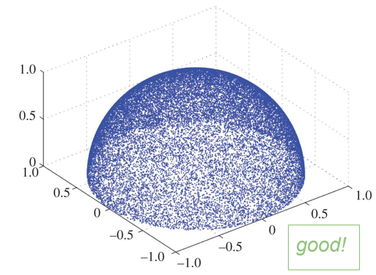
Today

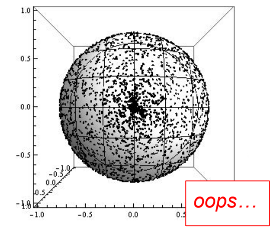
- Ray Casting vs. Ray Tracing vs. Monte-Carlo Ray Tracing vs. Path Tracing
- Irradiance Caching
- Photon Mapping
- Ray Grammar
- Monte-Carlo Integration
- Importance Sampling
  - Stratified Sampling
  - Importance Sampling

Domains of Integration

- Pixel, lens (Euclidean 2D domain)
- Time (1D)
- Hemisphere: Work needed to ensure *uniform* probability



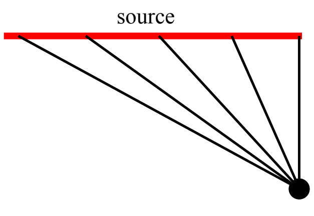




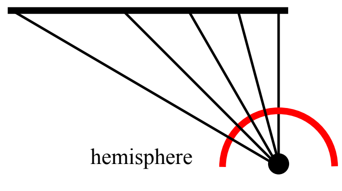
Example: Light Source

- We can integrate over surface *or* over angle
- But we must be careful to get probabilities and integration measure right!

Sampling the source uniformly

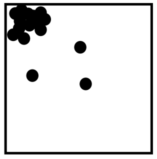


Sampling the hemisphere uniformly

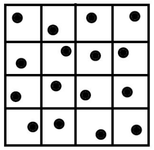


Stratified Sampling

- With uniform sampling, we can get unlucky
  - E.g. all samples in a corner



- To prevent it, subdivide domain Ω into non-overlapping regions Ω_i
  - Each region is called a stratum
- Take one random samples per Ω_i



Stratified Sampling Example

$f(x) = e^{\sin(3x^2)}$

N	I
1	2.75039
10	1.9893
100	1.79139
1000	1.75146
10000	1.77313
100000	1.77862

Unstratified

$O(1/\sqrt{N})$

$f(x) = e^{\sin(3x^2)}$

N	I
1	2.70457
10	1.72858
100	1.77925
1000	1.77606
10000	1.77610
100000	1.77610

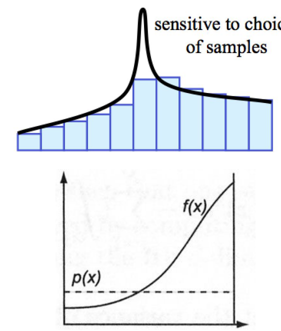
Stratified

$O(1/N)$

Slide from Henrik Wann Jensen

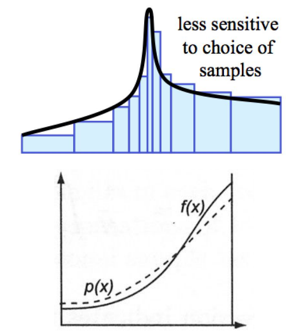
Sampling

uniform sampling
(or uniform random)



all samples
weighted equally

dense sampling where
function has greater magnitude

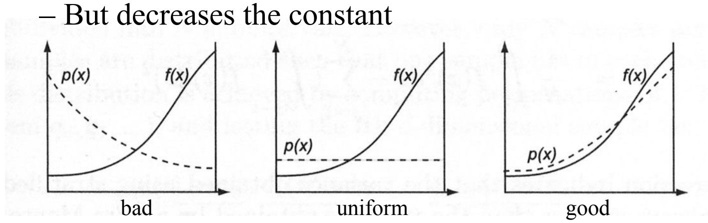


weights (width) for dense
samples are reduced

Importance Sampling

$$\langle I \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$$

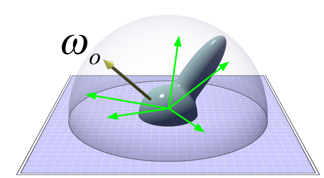
- Choose *p* wisely to reduce variance
  - Want to use a *p* that resembles *f*
  - Does not change convergence rate (still sqrt)
  - But decreases the constant

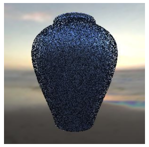


Uniform vs. Importance Sampling

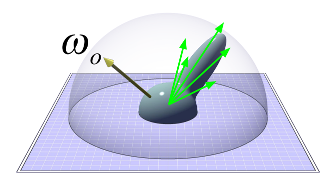
5 Samples/Pixel

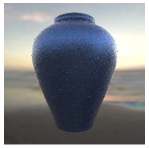
$U(\omega_i)$





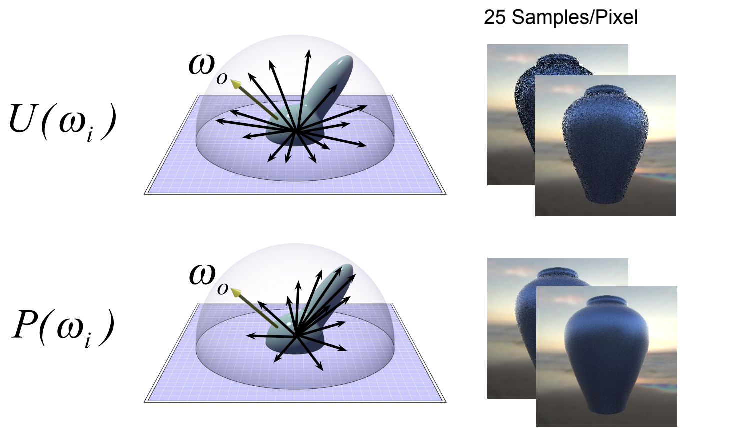
$P(\omega_i)$





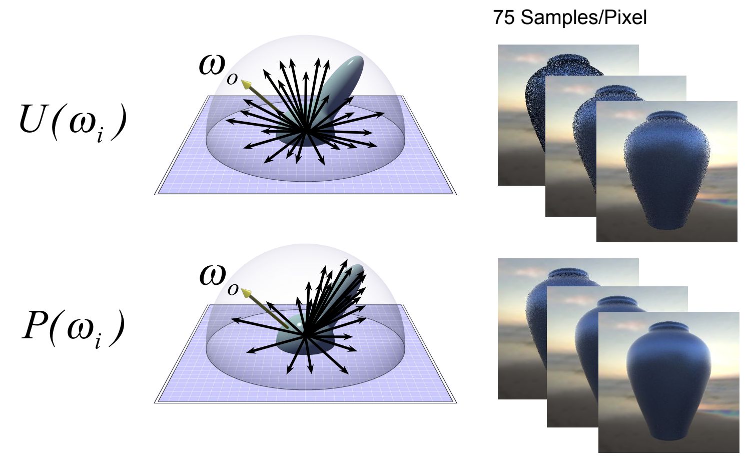
Slide from Jason Lawrence

Uniform vs. Importance Sampling



Slide from Jason Lawrence

Uniform vs. Importance Sampling



Slide from Jason Lawrence

Bidirectional Path Tracing

- "A Theoretical Framework for Physically Based Rendering", Lafortune and Willems, Computer Graphics Forum, 1994.

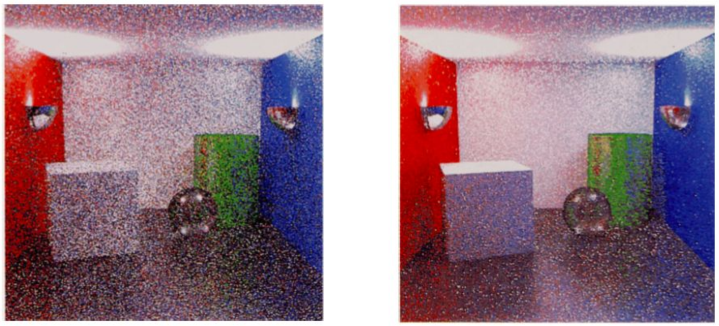


Figure B: *An indirectly illuminated scene rendered using path tracing and bidirectional path tracing respectively. The latter method results in visibly less noisefor the same amount of work.*

Questions?

Naïve sampling strategy | Optimal sampling strategy

Veach & Guibas "Optimally Combining Sampling Techniques for Monte Carlo Rendering" SIGGRAPH 95