# Some Global illumination -- Monte Carlo and others

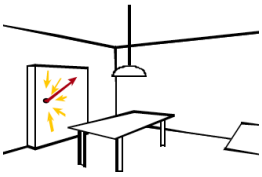
Zoë Wood

## Illumination

- Light hits a surface and is….
  - Absorbed
  - Reflected
  - Refracted

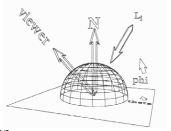
## Rendering equation

– Light leaving 1 point in 1 direction depends on
  - Integral of incoming light from every direction
  - BRDF (reflectance function)



## Rendering Equation

$$L_o(\mathbf{x},\mathbf{w}) = L_e(\mathbf{x},\mathbf{w}) + \int_\Omega f_r(\mathbf{x},\mathbf{w}',\mathbf{w}) L_i(\mathbf{x},\mathbf{w}')(-\mathbf{w}'\cdot\mathbf{n})d\mathbf{w}'$$




Lo = outgoing light in a particular direction w from a point x on the surface

Le = emitted light

Fr denotes the bidirectional reflectance distribution function

Li = the incoming light at the point x from the direction w'

Lo = outgoing light in a particular direction w from a point x on the surface

-w' dot n = attenuated light based on cosine of the angle between w' and n

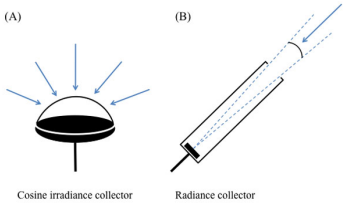
Integrated over the hemisphere over the point x for all directions w'

## Gobal illumination

- Radiance
  – Quantity passing through or emitted from a particular area (falls within a solid angle)
- Irradiance = radiant emittance = radiant exitance
  – Power of radiation incident on a surface (per unit area)
  – Includes direct and indirect

## Global illumination definitions

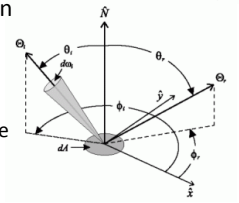
- More on radiance and irradiance



## BRDF

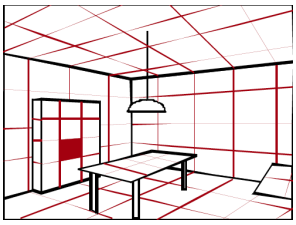
- Bidirectional Reflectance Distribution Function
- Defines how light is reflected based on
  - Incoming light
  - Outgoing light
  - Returns ratio of reflected radiance

$$\rho(\Theta_i;\Theta_r;\lambda) = \frac{dL_r(\Theta_i;\Theta_r;\lambda)}{dE_i(\Theta_i;\lambda)}$$

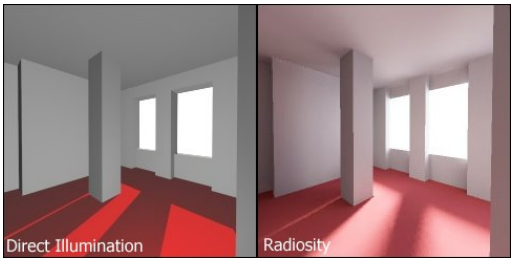


## Radiosity

- Based on heat transfer
- Subdivide the scene into discrete (finite) elements
- Elements transfer diffuse illumination
- View independent solution
- Computationally expensive

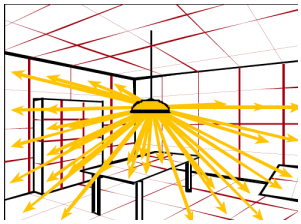


## Radiosity Example

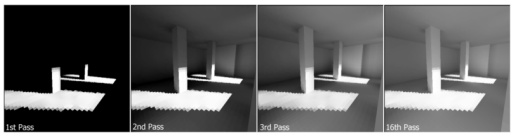



## Radiosity

- Iterative solution
- Shoot light from source
- Iterate with next luminous source



## Iterative Radiosity Solution




## Lightscape Example

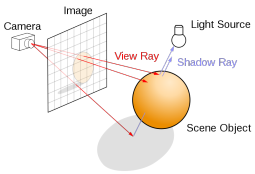
## Lightscape Example

## Lightscape example

## Lightscape

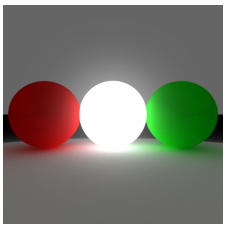
## Ray tracing

- From eye, not from light….
  - Light tracing = correct solution



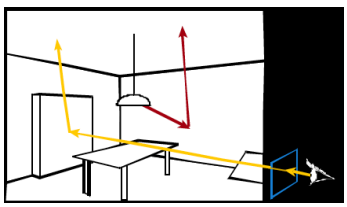

## Path tracing

- Complete global illumination
  - Infinite recursion raytracing
  - With new rays randomly sampled (within hemisphere) and traced until they hit the light…



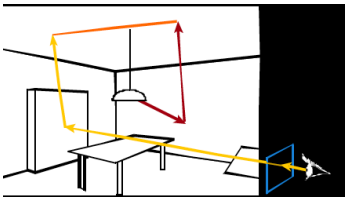
## Bi-directional

- Cast rays from light and camera



## Bi-directional

- join



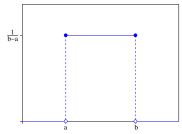
## Monte Carlo

- Repeated random sampling
- Want to gather indirect illumination
  - Without doing complete integral

Figure 4.1: *A spherical triangle with uniform samples (left) and stratified samples (right). Both sets of samples were generated using an area-preserving parametrization for spherical triangles, which we derive below.*

## Monte Carlo General

- Probability background
  - Continuous random variable x
    - Randomly takes some value
    - Its behavior described by the distribution of values it takes

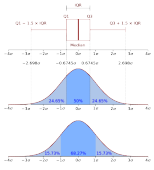


Uniform distribution

## Monte Carlo General

- Probability background
  - Continuous random variable x
  - Probability density function x~p
    - Describes likelihood of x taking on certain values

$$\text{Probability}(x \in [a,b]) = \int_a^b p(x)dx.$$



Pdf of normal function

## Monte Carlo

Expected value

- Value you would expect to find
- The average value for function, *f* (of a 1D random variable), with pdf, *p*

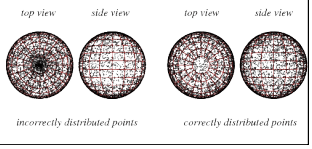
$$E(f(x)) = \int f(x)p(x)dx.$$

## Probability

- Everything similar for higher dimension (ie over sphere)

$$\text{Probability}(x \in S_i) = \int_{S_i} p(x)d\mu$$

$$(d\mu = dA = dxdy)$$



## Probability

- Expected value (summation):
  - Expected value of sum of two random variables is the sum of the expected variables

$$E(x+y) = E(x) + E(y),$$

- Key idea to Monte Carlo methods

## Probability

We want to use a Monte Carlo approach to compute an expected value of an integral

$$E(f(x)) = \int_{x \in S} f(x)p(x)d\mu \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i).$$

Law of large numbers = when expected value exists – it can be reached in limit

$$\text{Probability}\left[E(x) = \lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N} x_i\right] = 1.$$

## Probability

- Or more intuitively – the expected value can be reached by averaging many results

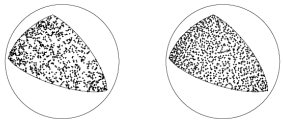
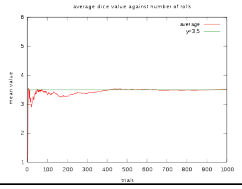


Figure 4.1: *A spherical triangle with uniform samples (left) and stratified samples (right). Both sets of samples were generated using an area-preserving parametrization for spherical triangles, which we derive below.*

## Monte Carlo

- Challenge is the number of samples, N…
  - (try 128-256)
  - Must quadruple points to halve the error




## Monte Carlo

- the *canonical random variable* has a probability density function

$$q(\xi) = \begin{cases} 1 & \text{if } 0 \le \xi \le 1 \\ 0 & \text{otherwise} \end{cases}$$

## Probability

- Variance

## Monte Carlo

- Its awkward to have a product in the integral, rewrite as:

$$\int_{x \in S} g(x) d\mu \approx \frac{1}{N} \sum_{i=1}^{N} \frac{g(x_i)}{p(x_i)}.$$

- Challenge is the number of samples, N…
  - (try 128-256)
  - Must quadruple points to halve the error

## Monte Carlo

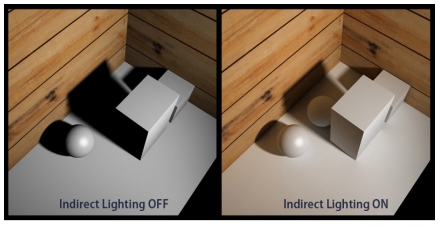
- Its awkward to have a product in the integral, rewrite as:

$$\int_{x \in S} g(x) d\mu \approx \frac{1}{N} \sum_{i=1}^{N} \frac{g(x_i)}{p(x_i)}.$$

- Also want the 'shape' of p to match that of g = importance sampling

## Notice Indirect light smooth

- It can be estimated fairly well




Artist: Ted Czotter

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi$$

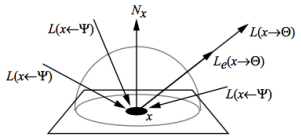



Figure 8.1: Rendering equation

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = \boxed{L_e(x \to \Theta)} + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi$$

- Emitted radiance (only non-zero if a light source)

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = L_e(x \to \Theta) + \boxed{\int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi}$$

- Contribution from all of the other surfaces in the scene

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi$$

- Radiance at point x from Ψ

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi$$

- Scale radiance at point x from Ψ in the direction theta by the BRDF

## Rendering equation

- Radiance leaving point x is a direction theta

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(x \leftarrow \Psi) cos(N_x, \Psi) d\omega_\Psi$$

- Scale radiance based on geometric relationship

## Rendering equation

- Equivalently we can re-write this to instead integrate over the surfaces

**Exitant radiance, integration over the hemisphere**

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta) L(y \to -\Psi) \cos(N_x, \Psi) d\omega_\Psi$$

with

$$y = r(x, \Theta)$$

## Rendering equation

- Algorithm will integrate over hemisphere (cast ray)

$$L(x \to \Theta) = L_e(x \to \Theta) + \int_A f_r(x, \Psi \leftrightarrow \Theta) L(y \to \overrightarrow{yx}) V(x, y) G(x, y) dA_y$$

with

$$G(x, y) = \frac{cos(N_x, \Psi) cos(N_y, \Psi)}{r_{xy}^2}$$

- V(x, y) evaluates the visibility between points x and y

## Monte Carlo

- Simple stochastic ray tracing – given the rendering equation

$$\begin{aligned} L(x \to \Theta) &= L_e(x \to \Theta) + L_r(x \to \Theta) \\ &= L_e(x \to \Theta) + \int_{\Omega_x} L(x \leftarrow \Psi) f_r(x, \Theta \leftrightarrow \Psi) \cos(\Psi, N_x) d\omega_\Psi \end{aligned}$$

re-write

$$\langle L_r(x \to \Theta) \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{L(x \leftarrow \Psi_i) f_r(x, \Theta \leftrightarrow \Psi_i) \cos(\Psi_i, N_x)}{p(\Psi_i)}$$

Note the incident radiance isn't known = recursive procedure

## Monte Carlo

- Or break rendering into:
  - Direct + indirect
  - Evaluate direct then add:

$$L_{indirect}(x \rightarrow \Theta) = \int_{\Omega_x} L_r(r(x,\Psi) \rightarrow -\Psi) f_r(x, \Theta \leftrightarrow \Psi) \cos(\Psi, N_x) d\omega_\Psi$$

## Monte Carlo

- replace the indirect integral with sum

$$L_{indirect}(x \rightarrow \Theta) = \int_{\Omega_x} L_r(r(x,\Psi) \rightarrow -\Psi) f_r(x, \Theta \leftrightarrow \Psi) \cos(\Psi, N_x) d\omega_\Psi$$

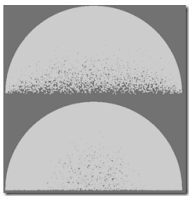
$$\langle L_{indirect}(x \rightarrow \Theta) \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{L_r(r(x,\Psi_i) \rightarrow -\Psi_i) f_r(x, \Theta \leftrightarrow \Psi_i) \cos(\Psi_i, N_x)}{p(\Psi_i)}$$

## Hemisphere sampling

- Given two random numbers u1, u2 in [0,1]
- Sample Disk
  - sampleDisk(u1, u2, Dx, Dy) {
    float r = sqrt(u1);
    float theta = 2.0*M_PI*u2;
    Dx = r*cosf(theta);
    Dy =r*sinf(theta);}

## Hemisphere sampling

- Cosine-weighted
  - sampleDisk(u1, u2, Dx, Dy)
  - Z = sqrt(max(0, 1-Dx*Dx-Dy*Dy



## Monte Carlo Integration

- Usually: $F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$

  For uniform sampling p(x) = 1/(2*PI)
- Diffuse: $L_o \approx \frac{2c}{N} \sum_{i=1}^{N} L_i \cos\theta$
  - With cosine weighting = $L_o \approx \frac{c}{\pi N} \sum_{i=1}^{N} \frac{L_i \cos\theta}{\frac{\cos\theta}{\pi}}$
    - thus

      $L_o \approx \frac{c}{N} \sum_{i=1}^{N} L_i$

## Monte Carlo Integration

- Usually: $F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$

  For uniform sampling p(x) = 1/(2*PI)
- Diffuse: $L_o \approx \frac{2c}{N} \sum_{i=1}^{N} L_i \cos\theta$
  - With cosine weighting = $L_o \approx \frac{c}{\pi N} \sum_{i=1}^{N} \frac{L_i \cos\theta}{\frac{\cos\theta}{\pi}}$
    - thus

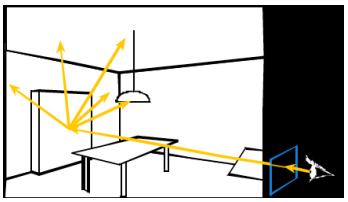
      $L_o \approx \frac{c}{N} \sum_{i=1}^{N} L_i$

## Hemisphere sampling

```
Vec3 sampleToHCoord ( float us , float ts ) {
   const float r = sqrt ( 1 . - us ) ;
   const float theta = 2 * PI * ts ;
   const float x = r * cosf ( theta ) ;
   const float y = r * sinf ( theta ) ;
   return Vec3 (x , y , sqrt ( us ) ) ;
}
```

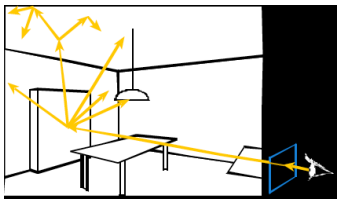
## Monte Carlo

- Cast a ray from the eye thru each pixel
- Cast random rays from the visible point



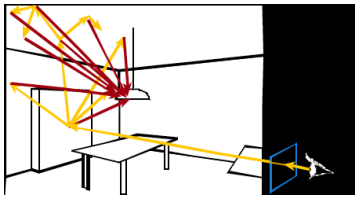
## Monte Carlo

- Cast a ray from the eye thru each pixel
- Cast random rays from the visible point
- Recurse



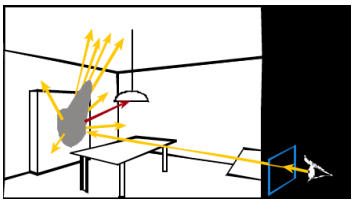
## Monte-Carlo

- Systematically sample primary light



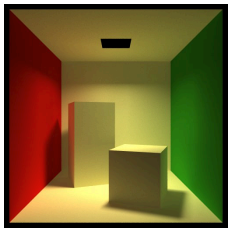
## Monte Carlo

- Take BRDF into account
  - Multiple incoming light
  - Consider sampling density



## Monte Carlo

- Monte Carlo integration can also be used for direct illumination to get soft shadows
  - Sample light



Ryan Schmitt

## References

- Monte Carlo Ray tracing course note from siggraph
  - Shirley probability chapter
  - The rendering equation and path tracing (Dutre)
  - Siggraph 2013
  - *Robust Adaptive Photon Tracing Using Photon-Path Visibility*
  - *Adaptive Progressive Photon Mapping*
  - *Gradient-Domain Metropolis Light Transport*
  - *Axis-Aligned Filtering for Interactive Physically Based Diffuse Indirect Lighting*
  - http://s2013.siggraph.org/attendees/technical-papers/session/global-illumination

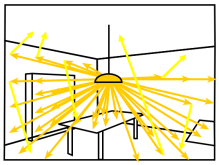
## Path tracing

- Monte Carlo methods can also be used in path tracing
  - Russian roulette
  - Metropolis light transport
    - Mutated paths

Bi-direction vs metropolis for same compute time:
http://graphics.stanford.edu/papers/metro/

## Photon Mapping

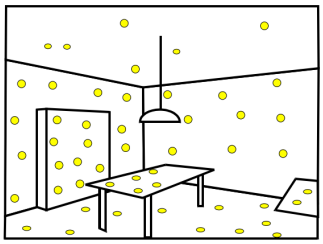
- Photon is a packet of energy
- Two step algorithm
  - 1st pass: Construct photon map from light to objects
    - Global and caustic map
    - Cast from light source and store



MIT EECS 6.837, Cutler and Durand

## Photon Mapping

- Store photons (position + light power + incoming direction)
  - Use kd-tree



MIT EECS 6.837, Cutler and Durand

## Photon Mapping

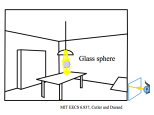
2nd pass: Render using a ray tracer

-Cast primary rays

-for secondary rays

- Reconstruct irradiance over a number of photons
- Only send shadow rays when 'needed'

– Note – caustic map is focused

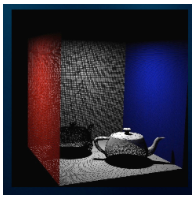



## Photon Mapping

Caustics are modeled well with Photon mapping

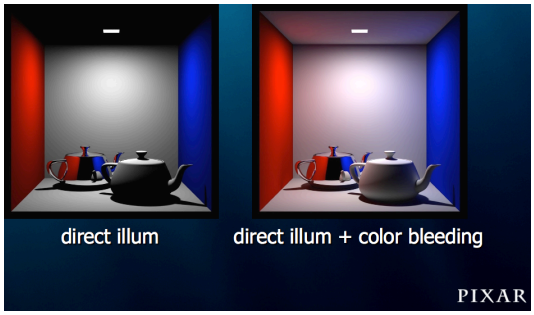
## Pointed based color bleeding

- Pixar, Sony, Dream Works, etc. (Per Christensen)
  - Compute & store direct lighting
  - Render by gathering light (from that stored)



Pixar

## Point based color bleeding




## Real-time ray tracing?

- Shader implementations
- Intel demoed 'real-time' raytracing on Larrabee in September 2009
- New gpus +CUDA….


http://techreport.com/discussions.x/17641