Biased Monte Carlo Ray Tracing
Filtering, Irradiance Caching, and Photon Mapping

Henrik Wann Jensen
Stanford University
May 23, 2002
Unbiased and Consistent

Unbiased estimator:

$$E\{X\} = \int \ldots$$

Consistent estimator:

$$\lim_{N \to \infty} E\{X\} \to \int \ldots$$
Unbiased estimator:

$$\frac{1}{N} \sum_{i=1}^{N} f(\xi_i)$$

Consistent estimator:

$$\frac{1}{N + 1} \sum_{i=1}^{N} f(\xi_i)$$
Unbiased Methods

- Variance (noise) is the only error
- This error can be analyzed using the variance (i.e. 95% of samples are within 2% of the correct result)
Path Tracing (Unbiased)

10 paths/pixel
Path Tracing (Unbiased)

10 paths/pixel
Path Tracing (Unbiased)

100 paths/pixel
How Can We Remove This Noise
The World is Diffuse!

Arnold Rendering
The World is Diffuse!

Arnold Rendering
The World is Diffuse!

Arnold Rendering
Noise Reduction/Removal

- More samples (slow convergence, $\sigma \propto 1/\sqrt{N}$)
Noise Reduction/Removal

- More samples (slow convergence, $\sigma \propto 1/\sqrt{N}$)
- Better sampling (stratified, importance, qmc etc.)
Noise Reduction/Removal

- More samples (slow convergence, $\sigma \propto 1/\sqrt{N}$)
- Better sampling (stratified, importance, qmc etc.)
- Adaptive sampling
Noise Reduction/Removal

- More samples (slow convergence, $\sigma \propto 1/\sqrt{N}$)
- Better sampling (stratified, importance, qmc etc.)
- Adaptive sampling
- Filtering
Noise Reduction/Removal

- More samples (slow convergence, $\sigma \propto 1/\sqrt{N}$)
- Better sampling (stratified, importance, qmc etc.)
- Adaptive sampling
- Caching and interpolation
Stratified Sampling

Latin Hypercube: 10 paths/pixel
Quasi Monte-Carlo

Halton-Sequence: 10 paths/pixel
Fixed (Random) Sequence

10 paths/pixel
Filtering: Idea

- Noise is high frequency
Filtering: Idea

- Noise is high frequency
- Remove high frequency content
Unfiltered Image

10 paths/pixel
3x3 Lowpass Filter

10 paths/pixel
Unfiltered Image

10 paths/pixel
3x3 Median Filter

10 paths/pixel
Energy Preserving Filters
Energy Preserving Filters

- Distribute noisy energy over several pixels
Energy Preserving Filters

- Distribute noisy energy over several pixels

- Adaptive filter width
  [Rushmeier and Ward 94]

- Diffusion style filters
  [McCool99]

- Splatting style filters
  [Suykens and Willems 00]
Problems With Filtering

- Everything is filtered (blurred)
  - Textures
  - Highlights
  - Caustics
  - ...
Caching Techniques
Caching Techniques

Irradiance caching:
Compute irradiance at selected points and interpolate.

Photon mapping:
Trace “photons” from the lights and store them in a photon map, that can be used during rendering.
Box: Direct Illumination
Box: Global Illumination
Box: Indirect Irradiance
Irradiance Caching: Idea


Idea: Irradiance changes slowly → interpolate.
Irradiance Sampling

\[ E(x) = \int_{2\pi} L'(x, \omega') \cos \theta \, d\omega' \]
\[ E(x) = \int_{2\pi} L'(x, \omega') \cos \theta \, d\omega' \]
\[ = \int_{0}^{2\pi} \int_{0}^{\pi/2} L'(x, \theta, \phi) \cos \theta \sin \theta \, d\theta \, d\phi \]
Irradiance Sampling

\[ E(x) = \int_{2\pi} L'(x, \omega') \cos \theta \, d\omega' \]

\[ = \int_0^{2\pi} \int_0^{\pi/2} L'(x, \theta, \phi) \cos \theta \sin \theta \, d\theta \, d\phi \]

\[ \approx \frac{\pi}{TP} \sum_{t=1}^T \sum_{p=1}^P L'(\theta_t, \phi_p) \]

\[ \theta_t = \sin^{-1} \left( \sqrt{\frac{t - \xi}{T}} \right) \quad \text{and} \quad \phi_p = 2\pi \frac{p - \psi}{P} \]
\[
\epsilon(x) \leq \left| \frac{\partial E}{\partial x}(x - x_0) + \frac{\partial E}{\partial \theta}(\theta - \theta_0) \right|
\]
\[ \epsilon(x) \leq \left| \frac{\partial E}{\partial x}(x - x_0) + \frac{\partial E}{\partial \theta} (\theta - \theta_0) \right| \]

\[ \leq E_0 \left( \frac{4 \| x - x_0 \|}{\pi x_{avg}} \right) + \sqrt{2 - 2 \vec{N}(x) \cdot \vec{N}(x_0)} \]

**position**

**rotation**

**position**

**rotation**
Irradiance Interpolation

\[ w(x) = \frac{1}{\epsilon(x)} \approx \frac{1}{\frac{||x-x_0||}{x_{avg}} + \sqrt{1 - \vec{N}(x) \cdot \vec{N}(x_0)}} \]

\[ E_i(x) = \frac{\sum_i w_i(x)E(x_i)}{\sum_i w_i(x)} \]
Find all irradiance samples with $w(x) > q$

if (samples found)
    interpolate
else
    compute new irradiance sample
Box: Irradiance Caching

1000 sample rays, $w > 10$
Box: Irradiance Cache Positions

1000 sample rays, \( w > 10 \)
Box: Irradiance Caching

1000 sample rays, \( w > 20 \)
Box: Irradiance Cache Positions

1000 sample rays, $w > 20$
Box: Irradiance Caching

5000 sample rays, \( w > 10 \)
Box: Irradiance Cache Positions

5000 sample rays, $w > 10$
Caustics

Pathtracing – 1000 paths/pixel
A simple test scene
Rendering
Photon Tracing
Photons
Radiance Estimate

\[ L(x, \vec{\omega}) = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) L'(x, \vec{\omega}') \cos \theta' \, d\omega \]
Radiance Estimate

\[ L(x, \vec{\omega}) = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) L'(x, \vec{\omega}') \cos \theta' \, d\omega \]

\[ = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) \frac{d\Phi^2(x, \vec{\omega}')}{d\omega} \cos \theta' \, dA \cos \theta' \, d\omega \]
Radiance Estimate

\[ L(x, \vec{\omega}) = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) L'(x, \vec{\omega}') \cos \theta' \, d\omega \]

\[ = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) \frac{d\Phi^2(x, \vec{\omega}')}{d\omega} \cos \theta' \, dA \]

\[ = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) \frac{d\Phi^2(x, \vec{\omega}')}{dA} \]
Radiance Estimate

\[ L(x, \vec{\omega}) = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) L'(x, \vec{\omega}') \cos \theta' \, d\omega \]

\[ = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) \frac{d\Phi^2(x, \vec{\omega}')}{d\omega \cos \theta' \, dA} \cos \theta' \, d\omega \]

\[ = \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) \frac{d\Phi^2(x, \vec{\omega}')}{dA} \]

\[ \approx \sum_{p=1}^{n} f_r(x, \vec{\omega}'_p, \vec{\omega}) \frac{\Delta \Phi_p(x, \vec{\omega}'_p)}{\pi r^2} \]
Radiance Estimate
The photon map datastructure

The photons are stored in a left balanced kd-tree

```c
struct photon = {
    float position[3];
    rgbe power;      // power packed as 4 bytes
    char phi, theta; // incoming direction
    short flags;
};
```
Rendering: Caustics
Caustic from a Glass Sphere

Photon Mapping: 10000 photons / 50 photons in radiance estimate
Caustic from a Glass Sphere

Path Tracing: 1000 paths/pixel
Sphereflake Caustic
Reflection Inside A Metal Ring

50000 photons / 50 photons in radiance estimate
Caustics On Glossy Surfaces

340 000 photons / $\approx$ 100 photons in radiance estimate
HDR environment illumination

Using lightprobe from www.debevec.org
Cognac Glass
Cube Caustic
Global Illumination

100000 photons / 50 photons in radiance estimate
Global Illumination

500,000 photons / 500 photons in radiance estimate
Fast estimate

200 photons / 50 photons in radiance estimate
Indirect illumination

10000 photons / 500 photons in radiance estimate
Global Illumination
Global Illumination

Global photon map

Caustics photon map
Photon tracing

- Photon emission
- Photon scattering
- Photon storing
Photon emission

Given $\Phi$ Watt lightbulb.

Emit $N$ photons.

Each photon has the power $\frac{\Phi}{N}$ Watt.

- Photon power depends on the number of emitted photons. Not on the number of photons in the photon map.
What is a photon?

• Flux (power) - not radiance!

• Collection of physical photons
  ✴ A fraction of the light source power
  ✴ Several wavelengths combined into one entity
Diffuse point light

Generate random direction
Emit photon in that direction

// Find random direction
do {
    x = 2.0*random()-1.0;
    y = 2.0*random()-1.0;
    z = 2.0*random()-1.0;
} while ( (x*x + y*y + z*z) > 1.0 );
Example: Diffuse square light

- Generate random position \( p \) on square
- Generate diffuse direction \( d \)
- Emit photon from \( p \) in direction \( d \)

// Generate diffuse direction
\[
\begin{align*}
\mathbf{u} &= \text{random}(); \\
\mathbf{v} &= 2\pi \times \text{random}(); \\
\mathbf{d} &= \text{vector}( \cos(\mathbf{v})\sqrt{\mathbf{u}}, \sin(\mathbf{v})\sqrt{\mathbf{u}}, \sqrt{1 - \mathbf{u}} );
\end{align*}
\]
Surface interactions

The photon is

- Stored (at diffuse surfaces) and
- Absorbed ($A$) or
- Reflected ($R$) or
- Transmitted ($T$)

$$A + R + T = 1.0$$
Photon scattering

The simple way:

Given incoming photon with power $\Phi_p$

Reflect photon with the power $R \ast \Phi_p$

Transmit photon with the power $T \ast \Phi_p$
Photon scattering

The simple way:

Given incoming photon with power $\Phi_p$

Reflect photon with the power $R \times \Phi_p$

Transmit photon with the power $T \times \Phi_p$

- Risk: Too many low-powered photons - wasteful!
- When do we stop (systematic bias)?
- Photons with similar power is a good thing.
Russian Roulette

- Statistical technique
- Known from Monte Carlo particle physics
- Introduced to graphics by Arvo and Kirk in 1990
Russian Roulette

Probability of termination: $p$
Russian Roulette

Probability of termination: $p$

$E\{X\}$
Russian Roulette

Probability of termination: \( p \)

\[
E\{X\} = p \cdot 0
\]
Russian Roulette

Probability of termination: $p$

$$E\{X\} = p \cdot 0 + (1 - p)$$
Russian Roulette

Probability of termination: \( p \)

\[ E\{X\} = p \cdot 0 + (1 - p) \cdot \frac{E\{X\}}{1 - p} \]
Russian Roulette

Probability of termination: $p$

$$E\{X\} = p \cdot 0 + (1 - p) \cdot \frac{E\{X\}}{1 - p} = E\{X\}$$
Russian Roulette

Probability of termination: \( p \)

\[
E\{X\} = p \cdot 0 + (1 - p) \cdot \frac{E\{X\}}{1 - p} = E\{X\}
\]

Terminate un-important photons and still get the correct result.
Russian Roulette Example

Surface reflectance: $R = 0.5$
Incoming photon: $\Phi_p = 2$ W

```plaintext
r = random();
if ( r < 0.5 )
    reflect photon with power 2 W
else
    photon is absorbed
```
Russian Roulette Intuition

Surface reflectance: $R = 0.5$

200 incoming photons with power: $\Phi_p = 2$ Watt

Reflect 100 photons with power 2 Watt instead of 200 photons with power 1 Watt.
Russian Roulette

- Very important!
- Use to eliminate un-important photons
- Gives photons with similar power :)
Sampling a BRDF

\[ f_r(x, \vec{\omega}_i, \vec{\omega}_o) = w_1 f_{r,1}(x, \vec{\omega}_i, \vec{\omega}_o) + w_2 f_{r,2}(x, \vec{\omega}_i, \vec{\omega}_o) \]
Sampling a BRDF

\[ f_r(x, \bar{\omega}_i, \bar{\omega}_o) = w_1 \cdot f_{r,d} + w_2 \cdot f_{r,s} \]

\[ r = \text{random}() \cdot (w_1 + w_2) ; \]

if ( \( r < w_1 \) )

reflect diffuse photon

else

reflect specular
Rendering
Direct Illumination
Specular Reflection
Caustics
Indirect Illumination
\[ L_r(x, \bar{\omega}) = \int_{\Omega_x} f_r(x, \omega', \bar{\omega}) L_i(x, \bar{\omega}') \cos \theta_i \, d\omega_i' \]

\[ = \int_{\Omega_x} f_r(x, \omega', \bar{\omega}) L_{i,l}(x, \bar{\omega}') \cos \theta_i \, d\omega_i' + \]

\[ \int_{\Omega_x} f_r(x, \omega', \bar{\omega}) (L_{i,c}(x, \bar{\omega}') + L_{i,d}(x, \bar{\omega}')) \cos \theta_i \, d\omega_i' + \]

\[ \int_{\Omega_x} f_r(x, \omega', \bar{\omega}) L_{i,c}(x, \bar{\omega}') \cos \theta_i \, d\omega_i' . \]
Features

- Photon tracing is unbiased
  - Radiance estimate is biased but consistent
  - The reconstruction error is local
- Illumination representation is decoupled from the geometry
Box

200,000 global photons, 50,000 caustic photons
Box: Global Photons

200000 global photons
Fractal Box

200000 global photons, 50000 caustic photons
Cornell Box
Indirect Illumination
Little Matterhorn
Mies house (swimmingpool)
Mies house (3pm)
Mies house (6pm)
The creation of realistic three-dimensional images is central to computer graphics. Photon mapping, an extension of ray tracing, makes it possible to efficiently simulate global illumination in complex scenes. Photon mapping can simulate caustics (focused light, such as shimmering waves at the bottom of a swimming pool), diffuse inter-reflections (e.g., the ‘bleeding’ of colored light from a red wall onto a white floor, giving the floor a reddish tint), and participating media (e.g., clouds or smoke). This book is a practical guide to photon mapping; it provides both the theory and the practical insight necessary to implement photon mapping and simulate all types of direct and indirect illumination efficiently.

http://graphics.stanford.edu/~henrik
henrik@graphics.stanford.edu