Monte Carlo Path Tracing

Today
- Path tracing starting from the eye
- Path tracing starting from the lights
- Which direction is best?
- Bidirectional ray tracing
- Random walks and Markov chains

Next
- Irradiance caching
- Photon mapping

The Rendering Equation

\[ L(x, \omega) = L_e(x, \omega) + \int_{H^2} f_r(x, \omega' \rightarrow \omega) L(x^*(x, \omega'), -\omega') \cos \theta' \, d\omega' \]

\[ L = L_e + K \circ L \]
Solving the Rendering Equation

Rendering Equation

\[ L = L_e + K \circ L \]
\[ (I - K) \circ L = L_e \]

Solution

\[ L = (I - K)^{-1} \circ L_e \]
\[ = (I + K + K^2 + K^3 + \cdots) \circ L_e \]
\[ = (I + K(I + K(I + K \cdots))) \circ L_e \]

Successive Approximation
Light Path

\[ S(x_0, x_1) = L_e(x_0, x_1) \]

\[ L_S(x_0, x_1, x_2, x_3) \]

Light Path

\[ S(x_0, x_1) \]
\[ f_r(x_1, x_2, x_3) \]

\[ G(x_0, x_1) \]
\[ G(x_1, x_2) \]

\[ f_r(x_0, x_1, x_2) \]
\[ f_r(x_1, x_2, x_3) \]

\[ L_S(x_0, x_1, x_2, x_3) = S(x_0, x_1)G(x_0, x_1)f_r(x_0, x_1, x_2)G(x_1, x_2)f_r(x_1, x_2, x_3) \]
Solving the Rendering Equation

One path

\[ L_3(x_0, x_1, x_2, x_3) = S(x_0, x_1)G(x_0, x_1)f_r(x_0, x_1, x_2)G(x_1, x_2)f_r(x_1, x_2, x_3) \]

Solution is the integral over all paths

\[ L(x_{k-1}, x_k) = \sum_{k=1}^{\infty} \int_{M^2} \cdots \int_{M^2} L_3(x_0, \cdots, x_{k-2}, x_{k-1}, x_k) dA(x_0) \cdots dA(x_{k-2}) \]

Solve using Monte Carlo Integration

Question: How to generate a random path?

Path Tracing from the Eye
Path Tracing: From Camera

Step 1. Choose a camera ray \( r \) given the \((x,y,u,v,t)\) sample

\[
\text{weight} = 1;
\]

Step 2. Find ray-surface intersection

Step 3.

if hit light

\[
\text{return weight} \times \text{Le}(r);
\]

else

\[
\text{weight} \times= \text{reflectance}(r)
\]

Choose new ray \( r' \sim \text{BRDF}(O|I) \)

Go to Step 2.

---

Path Tracing

10 paths / pixel
M. Fajardo Arnold Path Tracer

Works well for diffuse surfaces
And large hemispherical light sources
How Many Bounces?

Avoid path that carry little energy
   Terminate when the weight is low

Photons with similar power is a good thing
   Think of importance sampling
   Integrand is $f(x)/p(x)$ which is constant
Russian Roulette

Terminate photon with probability $p$
Adjust weight of the result by $1/(1-p)$

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

Intuition:
Reflecting from a surface with $R=.5$
100 incoming photons with power 2 W
1. Reflect 100 photons with power 1 W
2. Reflect 50 photons with power 2 W

Path Tracing: Include Direct Lighting

Step 1. Choose a camera ray $r$ given the $(x,y,u,v,t)$ sample

- weight = 1;
- $L = 0$

Step 2. Find ray-surface intersection

Step 3.

- $L += \text{weight} \times L(r)$
- weight *= reflectance($r$)
- Choose new ray $r' \sim \text{BRDF pdf}(r)$

Go to Step 2.
Penumbra: Trees vs. Paths

4 eye rays per pixel
16 shadow rays per eye ray

64 eye rays per pixel
1 shadow ray per eye ray

Variance Decreases with N

10 rays per pixel
100 rays per pixel

From Jensen, Realistic Image Synthesis Using Photon Maps
Fixed Sampling (Not Random Enough)

10 paths / pixel

Light Ray Tracing
Classic Ray Tracing

Forward (from eye): \( E S^* (D|G) L \)

Photon Paths

Radiosity

Caustics

From Heckbert
Early Example [Arvo, 1986]

“Backward“ ray tracing

Path Tracing: From Lights

Step 1. Choose a light ray.

Choose a ray from the light source distribution function

\[ x \sim p(x) \]

\[ d \sim p(d|x) \]

\[ r = (x, d) \]

weight = \Phi;}
Path Tracing: From Lights

Step 1. Choose a light ray
Step 2. Find ray–surface intersection
Step 3. Reflect or transmit
  \[ u = \text{Uniform()} \]
  if \( u < \text{reflectance}(x) \)
    Choose new direction \( d \sim \text{BRDF}(O|I) \)
    goto Step 2
  else \( u < \text{reflectance}(x) + \text{transmittance}(x) \)
    Choose new direction \( d \sim \text{BTDF}(O|I) \)
    goto Step 2
  else // absorption=\(1-\text{reflectance}-\text{transmittance}\)
    terminate on surface; deposit energy
Symmetric Light Path

\[ M = S(x_0, x_1)G(x_0, x_1)f_r(x_0, x_1, x_2)G(x_1, x_2)f_r(x_1, x_2, x_3)G(x_2, x_3)R(x_2, x_3) \]
Symmetric Light Path

\[ S(x_0, x_1) \]
\[ f_r(x_1, x_2, x_3) = f_r(x_3, x_2, x_1) \]
\[ G(x_1, x_2) \]
\[ x_2 \]
\[ G(x_2, x_3) = G(x_3, x_2) \]
\[ x_3 \]
\[ f_r(x_0, x_1, x_2) \]
\[ R(x_2, x_3) \]

\[ M = S(x_0, x_1)G(x_0, x_1)f_r(x_0, x_1, x_2)G(x_1, x_2)f_r(x_1, x_2, x_3)G(x_2, x_3)R(x_2, x_3) \]

Symmetric Light Path

\[ S(x_1, x_0) \]
\[ f_r(x_3, x_2, x_1) \]
\[ x_0 \]
\[ G(x_2, x_1) \]
\[ x_2 \]
\[ G(x_3, x_2) \]
\[ x_3 \]
\[ G(x_1, x_0) \]
\[ x_1 \]
\[ f_r(x_2, x_1, x_0) \]
\[ R(x_3, x_2) \]

\[ M = R(x_3, x_2)G(x_3, x_2)f_r(x_3, x_2, x_1)G(x_2, x_1)f_r(x_2, x_1, x_0)G(x_1, x_0)S(x_1, x_0) \]
Bidirectional Ray Tracing

\[ k = l + e \]

\[ l = 0, \ e = 3 \]
\[ l = 1, \ e = 2 \]
\[ l = 2, \ e = 1 \]
\[ l = 3, \ e = 0 \]
\[ k = 3 \]

Path Pyramid

From Veach and Guibas

\[ k = 3 \]
\[ k = 4 \]
\[ k = 5 \]
\[ k = 6 \]

\[ (l = 2, e = 1) \]
\[ (l = 5, e = 1) \]

CS348B Lecture 14

Pat Hanrahan, Spring 2009
Comparison

Same amount of time

Bidirectional path tracing
25 rays per pixel

Path tracing
56 rays per pixel

From Veach and Guibas

Which Direction?

Solve a linear system \( Mx = b \)

Solve for a single \( x_i \)?

Solve the reverse equation

Source \( x_i \)

Estimator \( < (x_i + Mx_i + M^2 x_i + \cdots), b > \)

More efficient than solving for all the unknowns
[von Neumann and Ulam]
Discrete Random Walk

Discrete Random Process

Assign probabilities to each process

\[ p_i^0 : \text{probability of creation in state } i \]
\[ p_{i,j} : \text{probability of transition from state } i \rightarrow j \]
\[ p_i^* : \text{probability of termination in state } i \quad p_i^* = 1 - \sum_j p_{i,j} \]
Discrete Random Process

Equilibrium number of particles in each state

\[ P_i = \sum_j p_{i,j} p_j + p_i^0 \quad M_{i,j} = p_{i,j} \]

\[ P = MP + p^0 \]

Equilibrium Distribution of States

Total probability of being in states \( P \)

Solve this equation

\[ (I - M)P = p^0 \]

\[ P = (I - M)^{-1} p^0 \]

\[ = \left( I + M + M^2 + \cdots \right)p^0 \]
Discrete Random Walk

1. Generate random particles from sources.
2. Undertake a discrete random walk.
3. Count how many terminate in state $i$
   [von Neumann and Ulam; Forsythe and Leibler; 1950s]

Monte Carlo Algorithm

Define a random variable on the space of paths

Path: $\alpha_k = (i_1, i_2, \ldots, i_k)$
Probability: $P(\alpha_k)$
Estimator: $W(\alpha_k)$

Expectation:

$$E[W] = \sum_{k=1}^{\infty} \sum_{\alpha_k} P(\alpha_k)W(\alpha_k)$$
Monte Carlo Algorithm

Define a random variable on the space of paths

**Probability:** \[ P(\alpha_k) = p_{i_1}^0 \times p_{i_1, i_2} \times p_{i_{k-1}, i_k} \times p_{i_k}^* \]

**Estimator:** \[ W_j(\alpha_k) = \frac{\delta_{i_k, j}}{p_{i_k}^*} \]

Estimator

Count the number of particles terminating in state \( j \)

\[
E[W_j] = \sum_{k=1}^{\infty} \sum_{i_k} \cdots \sum_{i_1} (p_{i_1}^0 \cdot p_{i_1, i_2} \cdots p_{i_{k-1}, i_k} \cdot p_{i_k}^*) \frac{\delta_{i_k, j}}{p_{i_k}^*} \\
= \left[ p^0 \right]_j + \left[ Mp^0 \right]_j + \left[ M^2 p^0 \right]_j + \cdots
\]
Equilibrium Distribution of States

Total probability of being in states $P$

$$P = (I + M + M^2 + \cdots) p^0$$

Note that this is the solution of the equation

$$(I - M)P = p^0$$

Thus, the discrete random walk is an unbiased estimate of the equilibrium number of particles in each state.