Monte Carlo Ray Tracing

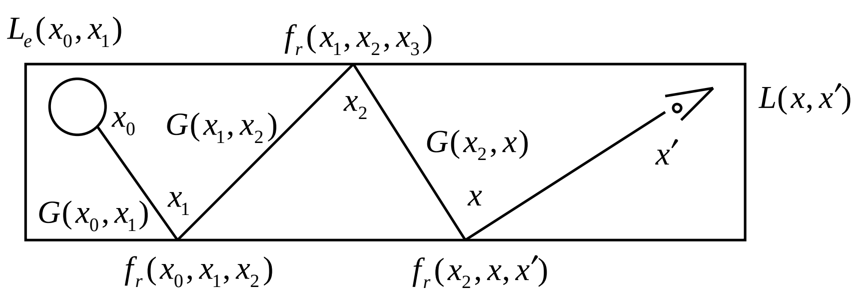
Today

- **Path tracing**
- **Particle tracing and Markov chains**
- **Adjoint equations: forward and backward ray tracing**
- **Bidirectional ray tracing**

Thursday – Henrik Wann Jensen

- **Density estimation**
- **Irradiance caching**
- **Photon mapping**

Light Paths



$$L(x,x') = \sum_{k=1}^{\infty} \int_{M^2} \int_{M^2} \cdots \int_{M^2} L_e(x_0,x_1) G(x_o,x_1)\, f_r(x_0,x_1,x_2) \cdots f_r(x_k,x,x')\; dA_0\, dA_1 \ldots dA_k$$

Light transport: Integrate over paths with k bounces

- **Sample space of paths**
- **Find good estimators**

Path Tracing: From Camera

Step 1. Choose a ray given (x,y,u,v,t)

weight = 1;

Step 2. Trace ray and find nearest point of intersection

Step 3. Randomly decide whether to compute Le or Lr

Step 3a. If Le,

return weight * Le;

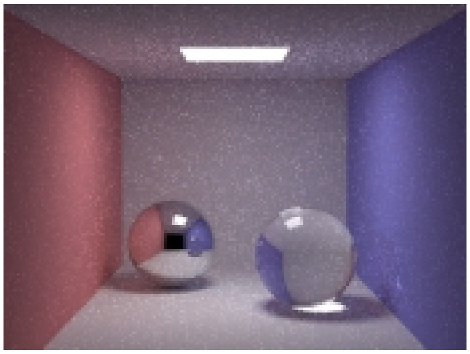
Step 3b. If Lr,

weight *= reflectance;

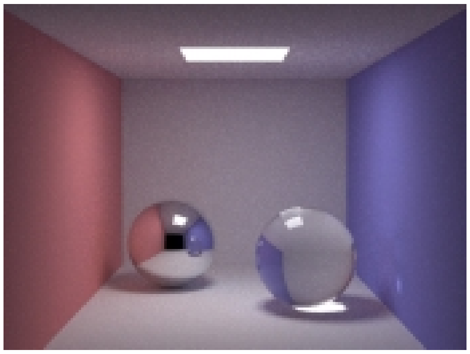
Randomly scatter the ray according to the BRDF pdf

Go to Step 2.

Cornell Box



10 rays per pixel



100 rays per pixel

Source: Jensen, Realistic Image Synthesis Using Photon Maps

Path Tracing: From Lights

Step 1. Choose a light source according to the light source power distribution function.

Choose a ray from the light source radiance (area) or intensity (point) distribution function

weight = 1;

Step 2. Trace ray and find nearest point of intersection

Step 3. Randomly decide whether to absorb or reflect the ray

Step 3a. If reflected,

weight *= reflectance;

Randomly scatter the ray according to the BRDF pdf

Go to Step 2.

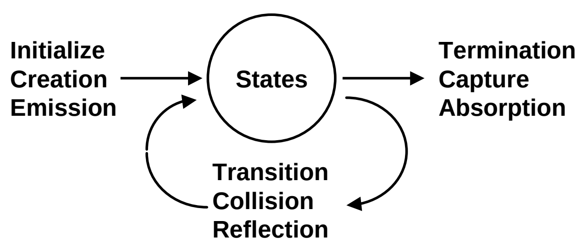
Step 3b. If absorbed at the camera,

Record weight at x, y

Go to Step 1;

Particle Simulation

Discrete Random Walk



von Neumann and Ulam; Forsythe and Leibler ('50)

1. Generate random particle paths from source (receiver).
2. Count how many terminate in state i.

Markov Process

Assign probabilities to each process

- **Creation:** p_i^0 : probability of particle being created in state i
- **Transition:** $p_{i,j}$: probability of transition from state $i \rightarrow j$
- **Termination:** p_i^* : probability of termination in state i

$$p_i^* = 1 - \sum_j p_{i,j}$$

Compute steady state probability of being in state *i*

$$P_i^0 = p_i^0$$
$$P_i^1 = \sum_j p_{j,i} P_j^0$$
$$\vdots$$
$$P_i^n = \sum_j p_{j,i} P_j^{n-1}$$

$$P_i = \sum_{k=0}^{\infty} P_i^k$$

But this is the solution of

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

Unbiased Solution

Define a random variable on the space of paths

Path: $\alpha_k = (i_1, i_2, \ldots, i_k)$

Expectation: $E[W] = \sum_{\alpha} P(\alpha) W(\alpha) = \sum_{k=1}^{\infty} \sum_{\alpha_k} P(\alpha_k) W(\alpha_k)$

Estimator: $W_j(\alpha_k) = \dfrac{\delta_{i_k,j}}{p_{i_k}^*}$

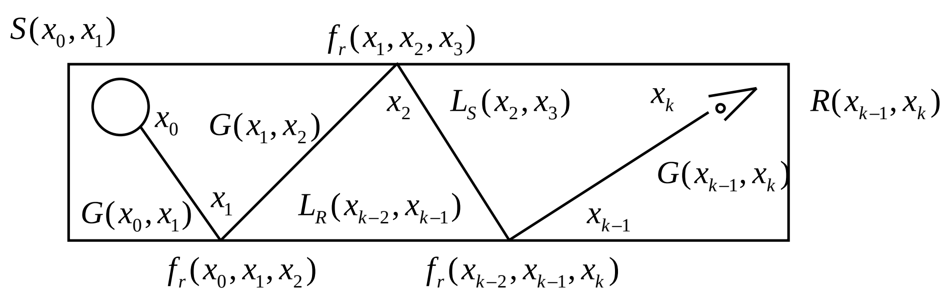
Unbiased Solution

Count the number of particles terminating in state *j*

$$P(\alpha_k) = p^0_{i_1} p_{i_1,i_2} \cdots p_{i_{k-1},i_k} p^*_{i_k}$$

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

Measurement



$$M = \sum_{k=1}^{\infty} \int_{M^2} \int_{M^2} \ldots \int_{M^2} S(x_0, x_1) G(x_o, x_1) f_r(x_0, x_1, x_2) \cdots G(x_{k-1}, x_k)\ R(x_{k-1}, x_k)\, dA_0\, dA_1 \ldots dA_k$$

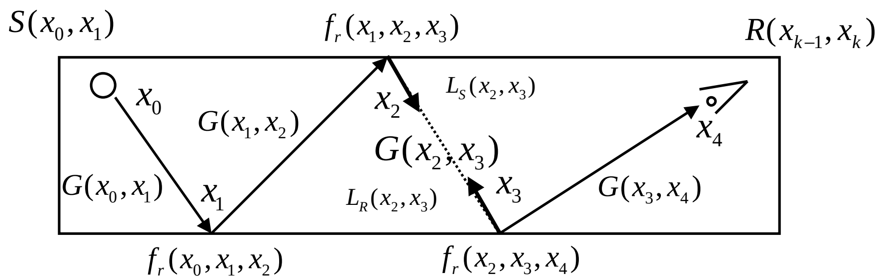
Note the symmetry between source and receiver

$$f_r(x_{k-1}, x_k, x_{k+1}) = f_r(x_{k+1}, x_k, x_{k-1})$$

$$G(x_{k-1}, x_k) = G(x_k, x_{k-1})$$

“Importance”

Importance is the adjoint solution



$$L_S(x_m, x_{m+1}) = \int_{M^2}\int_{M^2}\cdots\int_{M^2} S(x_0, x_1)G(x_0, x_1)f_r(x_0, x_1, x_3)\cdots G(x_{m-1}, x_m)f_r(x_{m-1}, x_m, x_{m+1})dA_0\,dA_1\ldots dA_{m-1}$$

$$L_R(x_m, x_{m+1}) = \int_{M^2}\int_{M^2}\cdots\int_{M^2} f_r(x_m, x_{m+1}, x_{m+2})G(x_{m+1}, x_{m+2})\cdots f_r(x_m, x_{m+1}, x_{m+2})G(x_{k-1}, x_k)R(x_{k-1}, x_k)\;\; dA_{m+2}\,dA_{m+3}\cdots$$

$$M = \int_{M^2}\int_{M^2} L_R(x_m, x_{m+1})G(x_m, x_{m+1})L_S(x_m, x_{m+1})\; dA_m dA_{m+1}$$

Adjoint Equations

Estimated quantity

$$< f, g >= \int f(x)g(x)\,dx$$

Original equation

$$K \circ g = \int K(x, y)g(y)\,dy$$

Estimated quantity

$$< f, Kg >= \int f(x)\left(\int K(x, y)g(y)\,dy\right)dx$$

$$= \int\left(\int f(x)K(x, y)dx\right)g(y)\,dy$$

$$=< K^+ f, g >$$

Adjoint equation

$$K^+ \circ f = \int K(x, y)\,f(x)\,dx$$

Forward=Backward Estimate

Source equation

$$K \circ L_S = S$$

Receiver equation

$$K \circ L_R = R$$

Inner product

$$M = < L_S, L_R > = \int_{M^2} \int_{M^2} L_S(x_{m-1}, x_m) L_R(x_{m-1}, x_m)\, G(x_{m-1}, x_m)\, dA_{m-1}\, dA_m$$

Reciprocity = Self adjoint

$$K^+ \circ L^+ = K \circ L$$

Forward = backward estimate

$$< R, L_S > = < K \circ L_R, L_S > = < L_R, K^+ \circ L_S > = < L_R, K \circ L_S > = < L_R, S >$$

Justification for eye ray tracing!

Three Consequences

1. **Forward estimate equal backward estimate**
2. **Solve for small subset of the answer**
3. **Importance sampling paths**

Example

Solve a linear system $Mx = b$

Solve for a single x_i?

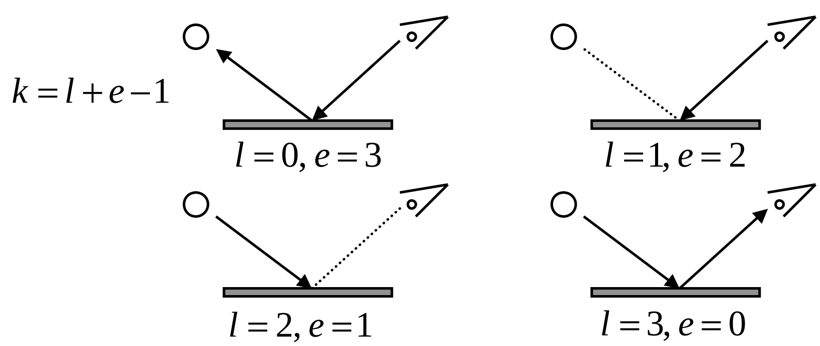
von Neumann and Ulam: Solve the adjoint equation

- **Source** x_i
- **Estimator** $<(x_i + Mx_i + M^2 x_i + \cdots), b>$

More efficient than solving the whole system of equations

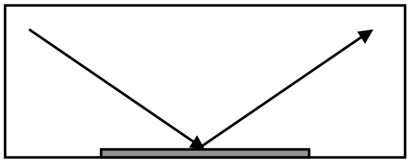
Applicable to image synthesis! Don't solve if not seen

Bidirectional Ray Tracing

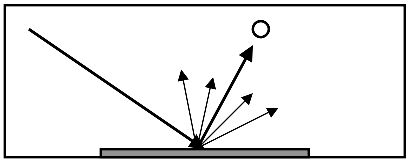


Delta Functions

Mirrors (Caustics)



Point lights



LS*DS*E particularly problematic

CS348B Lecture 15 Pat Hanrahan, Spring 2000

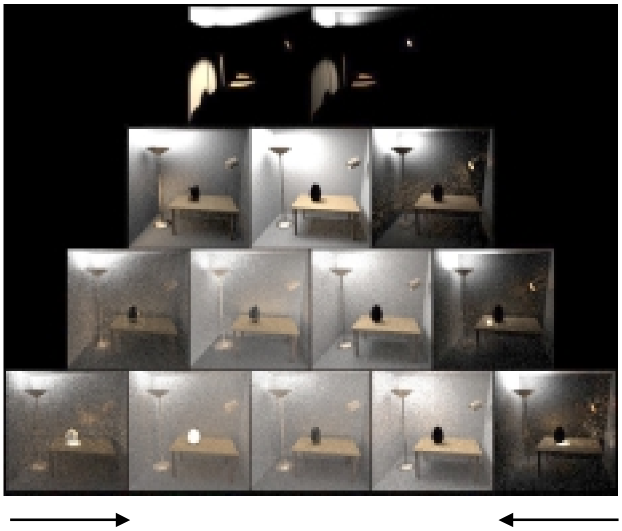
Path Pyramid

$k = 3$ $(l = 2, e = 1)$

$k = 4$

$k = 5$

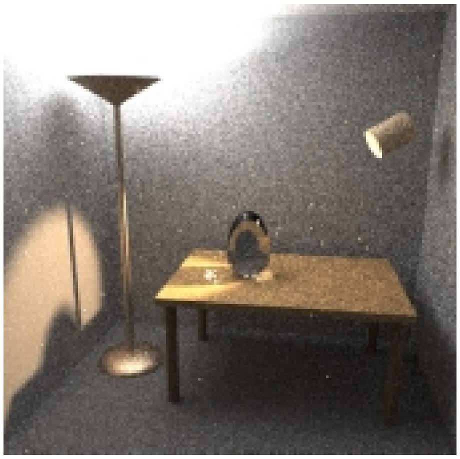
$k = 6$ $(l = 5, e = 1)$



l **From Veach and Guibas** e

CS348B Lecture 15 Pat Hanrahan, Spring 2000

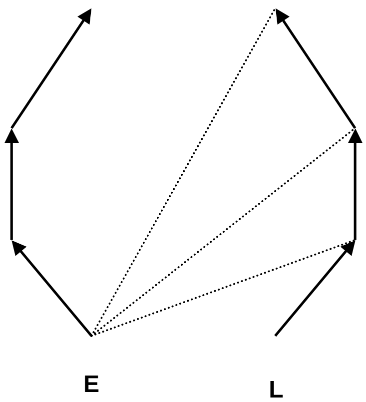
Comparison



Bidirectional ray tracing | **Path tracing**

From Veach and Guibas

Tree of Paths



Efficiently generate a collection of bidirectional paths