

Path Based Rendering

Prof. Dr. Stefan Gumhold

Chair of Computer Graphics and
Visualization, TU Dresden



Overview

Path Based Rendering

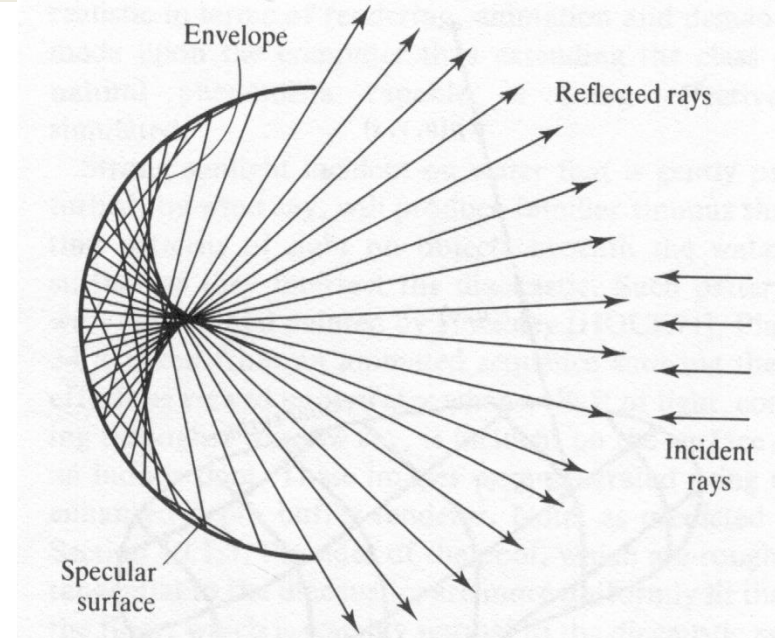
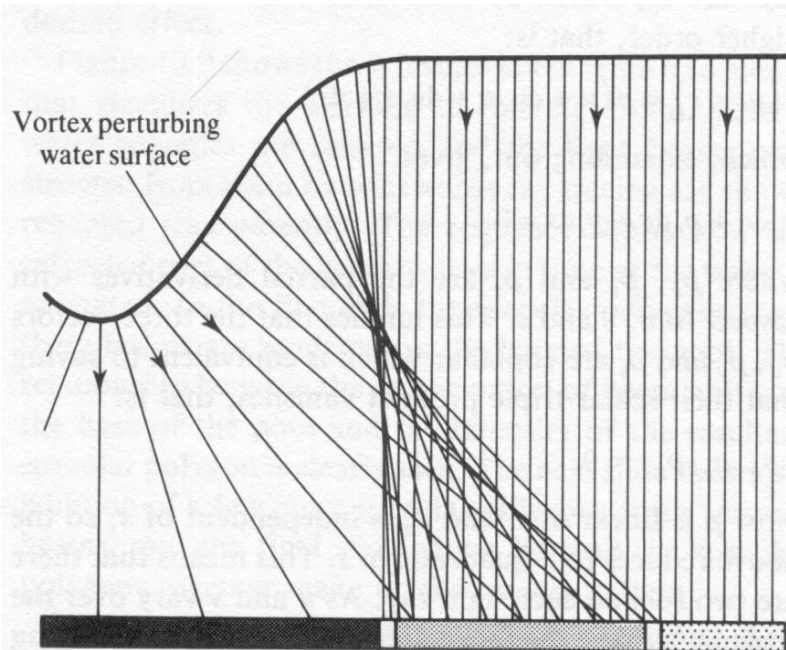
- ◆ Overview of Path Based Approaches
- ◆ Implementation of Bidirectional Path Tracing
- ◆ Metropolis Light Transport



OVERVIEW OF PATH BASED APPROACHES

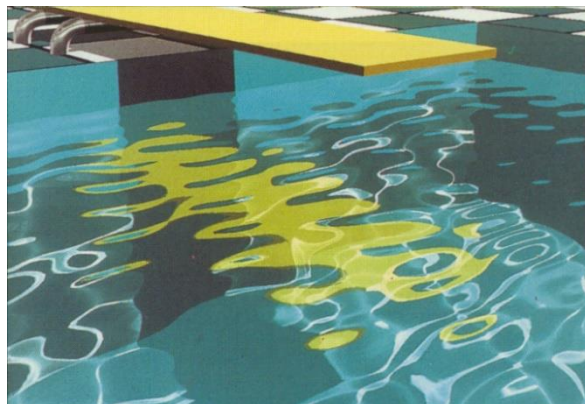
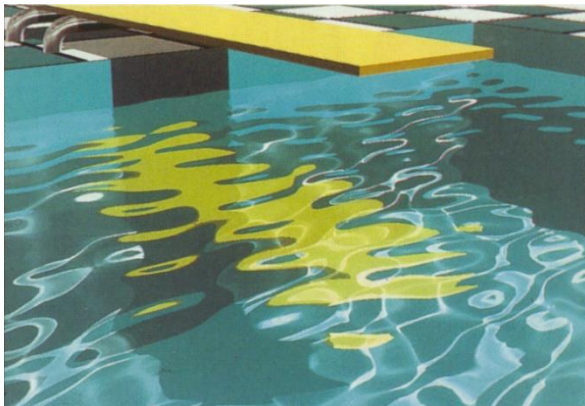
Light Tracing – caustics

- Curved mirror reflecting and refracting surfaces bundle light, which becomes visible when reflected from second surface
- The resulting bright areas are called **caustics**.



Light Tracing – caustics

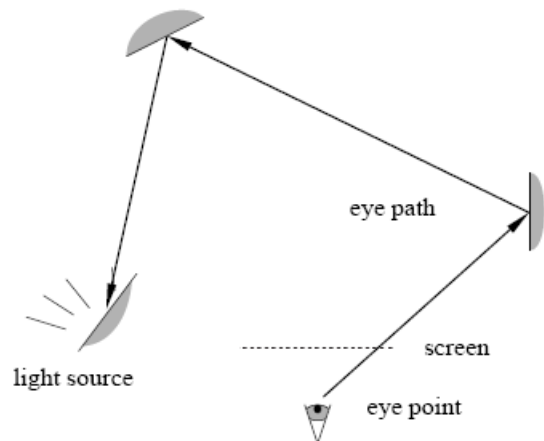
- ◆ This happens frequently in water, where the caustics are seen through the refracting water surface, making correct rendering even more difficult



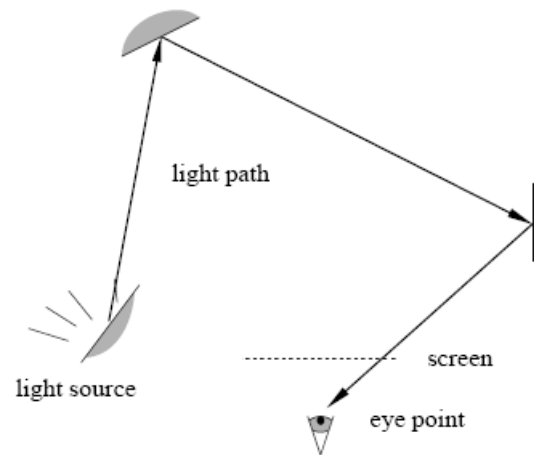
Watt & Watt, Advanced Animation and Rendering Techniques
(different approximation techniques)



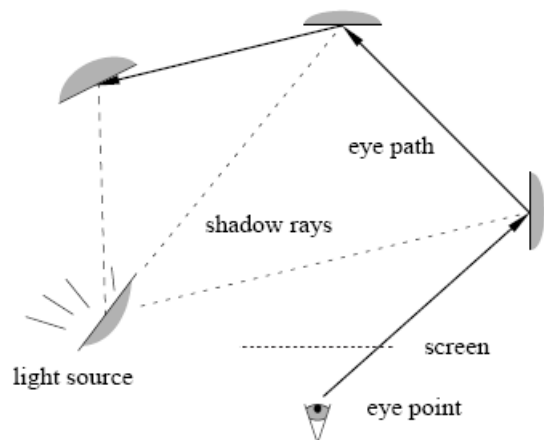
Path Tracing vs Light Tracing



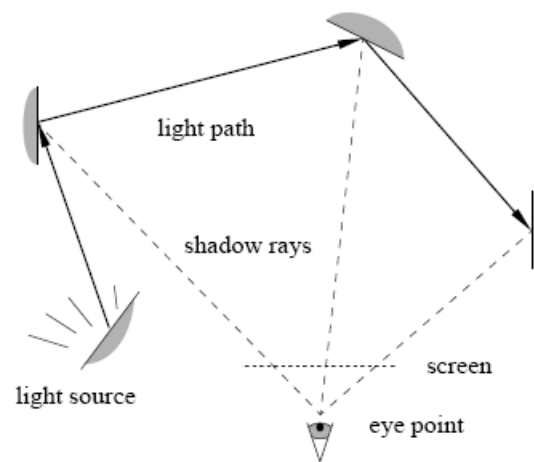
Path Tracing



Light Tracing



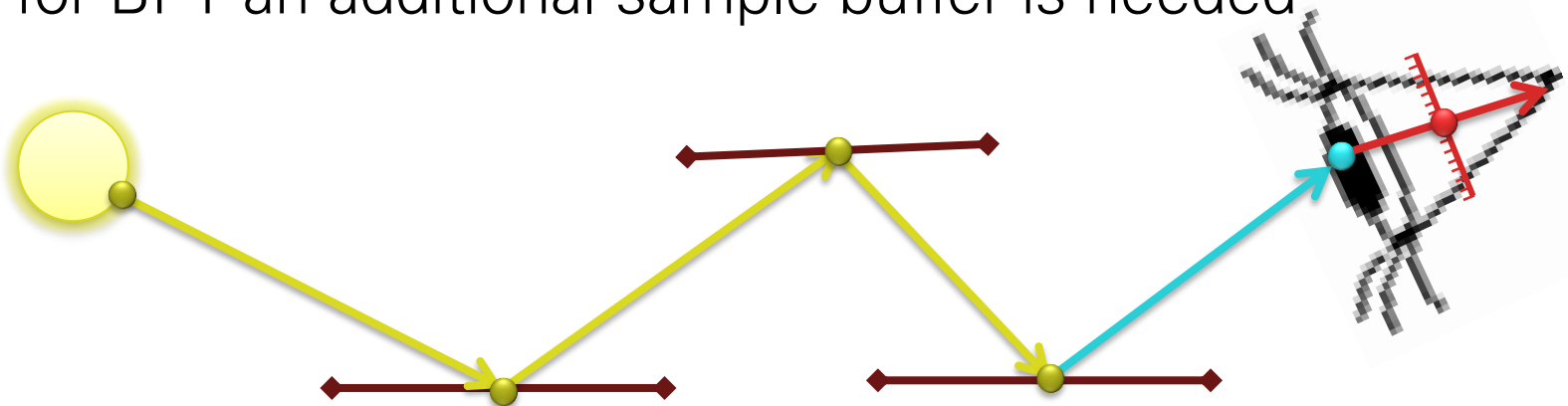
Path Tracing with Direct Lighting



Light Tracing with Eye Rays

Light Tracing – Image Formation

- ◆ light tracing samples paths from the light sources and connects them to the eye by **direct eye sampling** after Russian Roulette terminated a path
- ◆ direct eye sampling connects to the eye point or chooses a point on the lense
- ◆ In both cases the pixel location depends on the previous path point
- pixels get different number of samples or even none!
- for BPT an additional sample buffer is needed





Light Tracing – Algorithm

light tracing algorithm

- construct sample buffer with per pixel accumulator A_{ij} and count c_{ij}
- loop over light paths
 - **generate path** x and compute $p(x)$
 - determine pixel location (i, j)
 - compute pixel contribution $f(x)$
 - add $f(x)/p(x)$ to accumulator A_{ij}
 - increment pixel count c_{ij}
- compute final image
$$I_{ij} = A_{ij}/c_{ij}$$

generate path

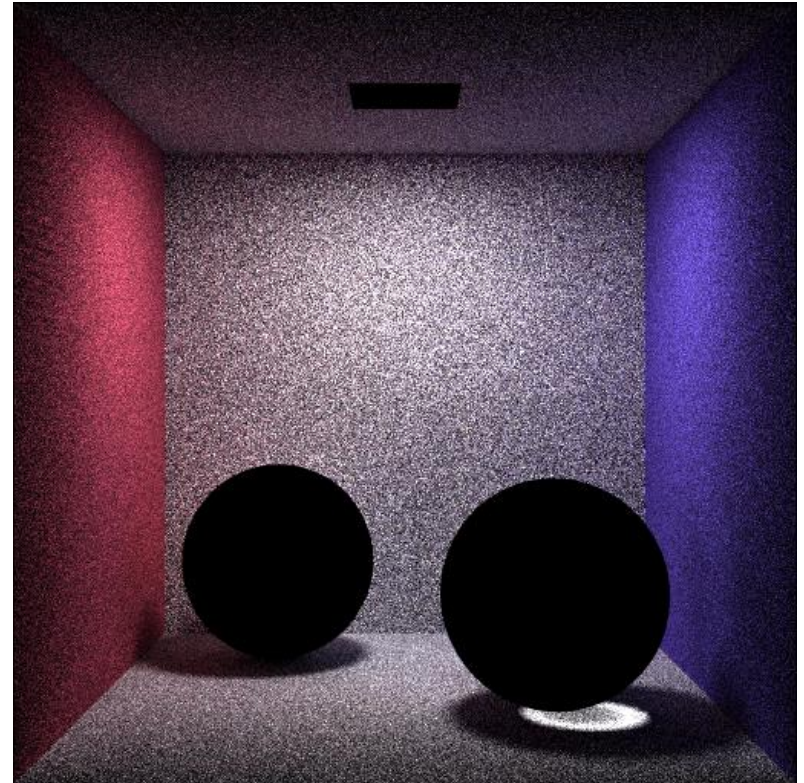
- sample initial ray on light source
- iterate over path length
 - intersect ray with scene
 - split BRDF and sample reflection type
 - in case of diffuse/specular perform RR
 - in case of termination perform **direct eye sampling** and **return** generated path
 - otherwise sample outgoing direction according to BRDF times cosine term, construct new ray and iterate

direct eye sampling

- sample position on aperture
- construct ray to eye and check visibility
- reject complete path if check fails
- otherwise compute light contribution and return generated path

Light Tracing

- ◆ although caustics are reproduced better with light tracing, all other effects are worse than with path tracing
- ◆ therefore light tracing should be only used in combined approaches

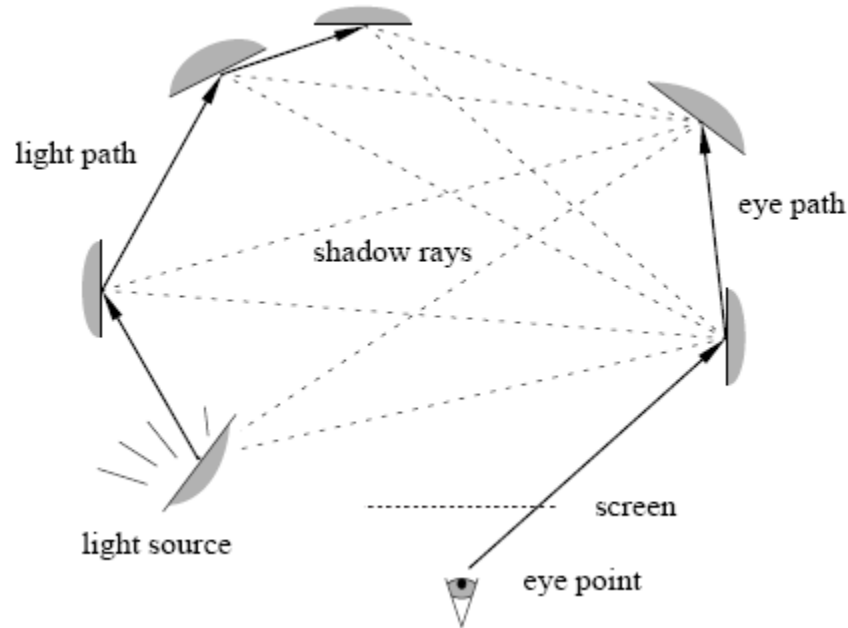


Combined Methods Overview

- ◆ To combining methods without introducing bias, it is important to use multiple importance sampling (MIS) on paths that can be generated in multiple ways
- ◆ possible combinations:
 - ◆ Path $L(D|S|M)^*E$ + Light Tracing $L(D|S|M)^*E$
 - ◆ Radiosity LD^*E + Raytracing $L(D|S)M^*E$
- ◆ Caching
 - ◆ further efficient approaches through caching of results from one method, such that the same calculations can be reused several times by a second method often called **Final Gathering**.
 - ◆ Examples for methods that allow caching:
 - ◆ Radiosity
 - ◆ Irradiance Mapping
 - ◆ Photon Mapping

Bidirectional Path Tracing

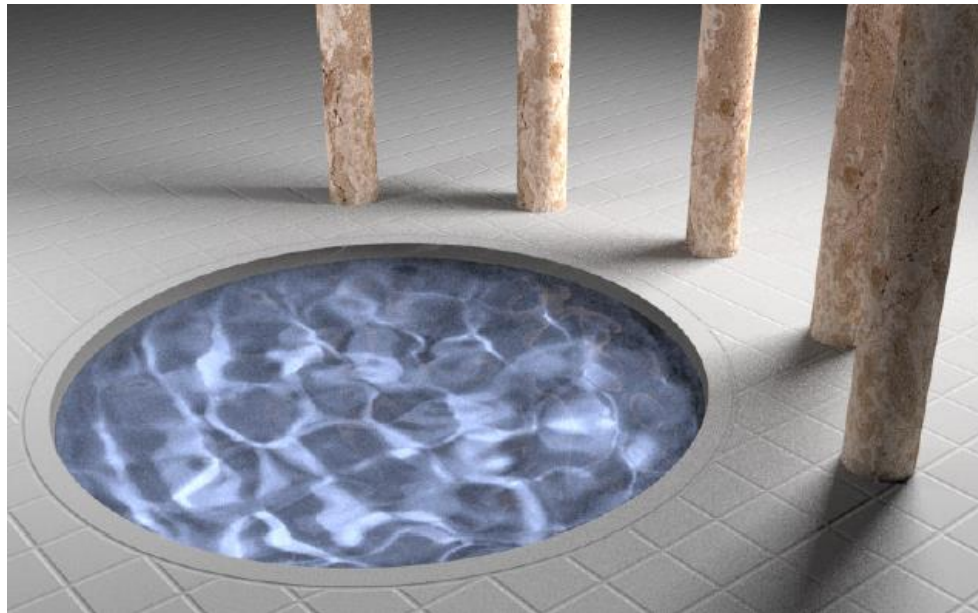
1. construct individual paths from eye and from light both terminated through Russian Roulette
2. interconnect paths for all possible connections between all possible truncated subpaths
3. compute whether connection is blocked
4. accumulate contribution with MIS to avoid bias



Bidirectional Path Tracing

Literature

- Eric P. Lafortune and Yves D. Willems. Bi-directional path-tracing. In Computer Graphics (SIGGRAPH '93 Proceedings), pages 145–153, Alvor, 1993,
- Eric Veach and Leonidas Guibas. Bidirectional estimators for light transport. In Computer Graphics (SIGGRAPH '95 Proceedings), pages 419–428, 1995.)

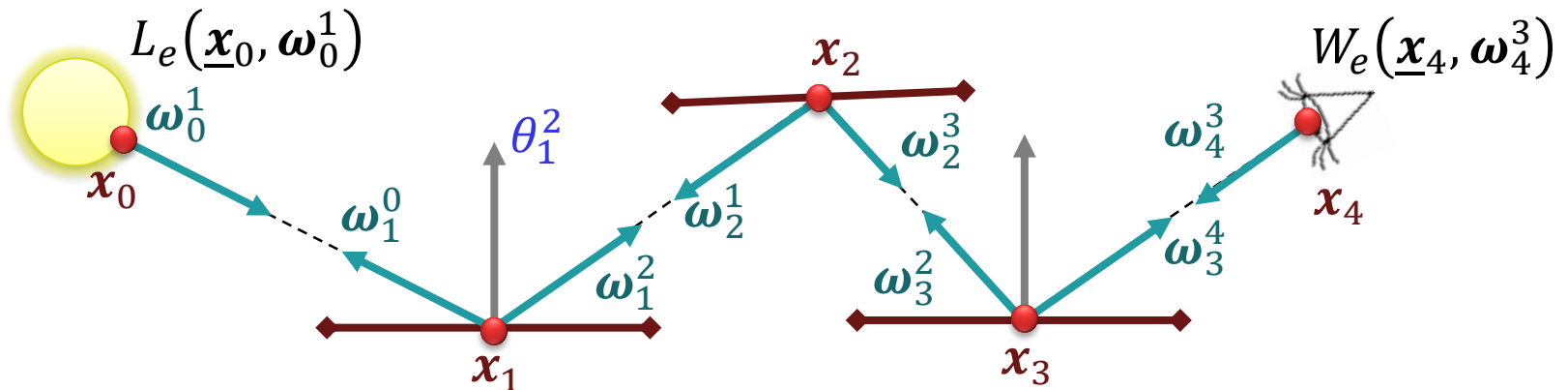




IMPLEMENTATION OF BIDIRECTIONAL PATH TRACING

Implementing BPT

- Veach 1997 introduces light paths $x = \underline{x}_0 \underline{x}_1 \dots \underline{x}_k$ for all **edge lengths** $k \geq 1$ from a light source location $\underline{x}_0$ to a location on the lense of the camera $\underline{x}_k$
- outgoing direction ω_0^1 of emission needed to evaluate $L_e(\underline{x}_0, \omega_0^1)$ is given by $\omega_0^1 \propto \underline{x}_1 - \underline{x}_0$.
- pixel index j and subpixel location are given from $\omega_k^{k-1} \propto \underline{x}_k - \underline{x}_{k-1}$ and used to evaluate sensitivity function W_e



Path Measure fixed k

- simplest approach: all path locations $\mathbf{x}_i$ are sampled according to the surface area dA_i .
- On the set Ω_k of paths of length $k \geq 1$ the product measure $d\mu_k = \prod_{i=0}^k dA_i$ is defined

Example: Average path length for $k = 2$:

$$\begin{aligned} \bar{l}_k &= \frac{1}{\mu_k(\Omega_k)} \int_{\Omega_k} \|\underline{\mathbf{x}}_1 - \underline{\mathbf{x}}_0\| + \|\underline{\mathbf{x}}_2 - \underline{\mathbf{x}}_1\| d\mu_k \\ &= \frac{1}{A_{\text{scn}}^2 A_{\text{lens}}} \int_{A_{\text{scn}}} \int_{A_{\text{scn}}} \int_{A_{\text{lens}}} \|\underline{\mathbf{x}}_1 - \underline{\mathbf{x}}_0\| + \|\underline{\mathbf{x}}_2 - \underline{\mathbf{x}}_1\| dA_2 dA_1 dA_0 \end{aligned}$$

with scene surface area A_{scn} and lense area A_{lens}

- No visibility is accounted for this example.



Path Measure

- ◆ To allow integration over paths of arbitrary length Veach introduces the set Ω of all paths and defines the measure μ from the μ_k , where each μ_k is applied to the subset Ω_k of paths with length k .

$$\Omega := \bigcup_{k \geq 1} \Omega_k, \quad \mu(\Omega) := \sum_{k \geq 1} \mu_k(\Omega_k)$$

Example: Average path length of paths of length $\leq k$:

$$\begin{aligned} \bar{l}_{\leq k} &= \frac{1}{\mu(\bigcup_{i \leq k} \Omega_i)} \int_{\bigcup_{i \leq k} \Omega_i} \sum_{j=1}^i \|\underline{x}_j - \underline{x}_{j-1}\| d\mu \\ &= \frac{1}{\sum_{i=1}^k \mu_i(\Omega_i)} \sum_{i=1}^k \int_{\Omega_i} \sum_{j=1}^i \|\underline{x}_j - \underline{x}_{j-1}\| d\mu_i \end{aligned}$$

- ◆ Overall average length: $\bar{l} = \lim_{k \rightarrow \infty} \bar{l}_{\leq k}$

Monte Carlo approximation

- Given some path sampling scheme like path tracing or light tracing that provides a path pdf $p(x)$ a MC estimator for a pixel j is

$$I_j = \int_{x \in \Omega} f_j(x) d\mu(x) \approx \hat{I}_{j,N} = \frac{1}{N} \sum_i^N \frac{f_j(x_i)}{p(x_i)}$$

- The path pdf $p(x_i)$ depends on the sampling scheme and can involve pixel filter sampling, Roussian Roulette, splitting of BRDF in components, cosine term sampling, BRDF sampling, direct light sampling, or MIS.
- Each of these processes is assumed to be independent of the others and therefore the **path pdf is just the product of the process pdfs.**

Path Throughput

- Each pixel value is integrated as $I_j = \int_{x \in \Omega} f_j(x) d\mu(x)$
- here $f_j(x)$ is the contribution of path $x = \underline{x}_0 \underline{x}_1 \dots \underline{x}_k$ to pixel j and split into

$$f_j(x = \underline{x}_0 \underline{x}_1 \dots \underline{x}_k) = L_e(\underline{x}_0, \omega_0^1) T(x) W_e(\underline{x}_k, \omega_k^{k-1})$$

- with path throughput

$$T(x) = G(\underline{x}_0, \underline{x}_1) \prod_{i=1}^{k-1} \rho(\underline{x}_i, \omega_i^{i-1}, \omega_i^{i+1}) G(\underline{x}_i, \underline{x}_{i+1})$$

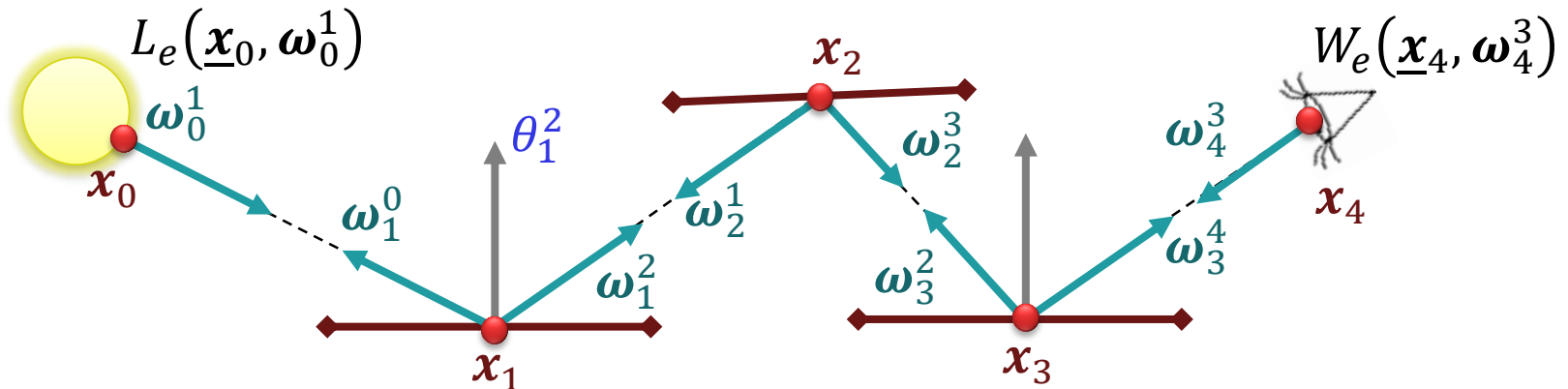
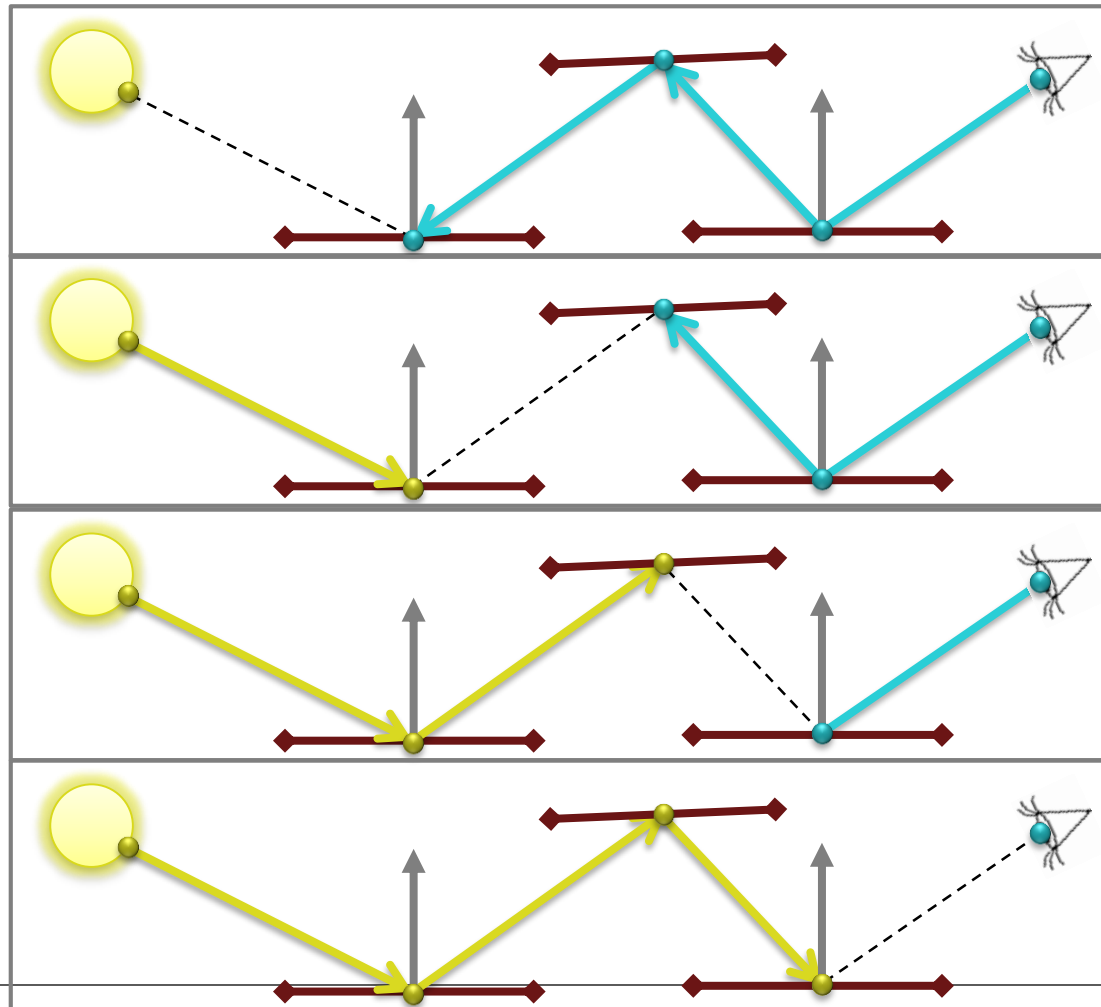


Image formation

- allocate an eye image I^E for regular contributions and a light image I^L for contributions of direct eye samples each with M pixels.
- generate $N = nM$ eye-light-path pairs with RR ($n \approx 25$)
 - eye path: sample pixel and lens location, and then trace to a path z terminated with RR at edge length t and compute $p_e(z)$
 - light path: sample light source location and then trace path y terminated with RR at edge length s and compute $p_l(y)$
 - interconnect all possible path prefixes α to paths x_α of edge length $k \leq s + t + 1$ and compute $p(x_\alpha) = p_l(y_\alpha)p_e(z_\alpha)$
 - do visibility test on edge that connects paths
 - discard path if edge is blocked as $f_j = 0$.
 - otherwise compute throughput $f_{j,\alpha}$ and use MIS estimator to add contribution to image pixels

Bidirectional Path Tracing (BPT)

- Careful!** The same path can be generated in several different ways: $(s, t) \in \{(0, k-1), (1, k-2), \dots, (k-1, 0)\}$



Path pdfs

- The same path can be created with light or path tracing with different probabilities:
- **Light Tracing** generates the path from $\underline{x}_0$ to $\underline{x}_k$ such that

$$p_l(x_i) = p_l(\underline{x}_0)p_l(\underline{x}_1|\underline{x}_0) \cdot \dots \cdot p_l(\underline{x}_k|\underline{x}_0\underline{x}_1 \dots \underline{x}_{k-1})$$
 - Note the dependency on preceding path vertices.
 - The last pdf $p_l(\underline{x}_k|\underline{x}_0\underline{x}_1 \dots \underline{x}_{k-1})$ describes sampling a point on the lens where the dependency on $\underline{x}_{k-1}$ implements the visibility test $V(\underline{x}_{k-1}, \underline{x}_k)$.
- **Path Tracing** generates paths from $\underline{x}_k$ to $\underline{x}_0$ such that

$$p_e(x_i) = p_e(\underline{x}_k)p_e(\underline{x}_{k-1}|\underline{x}_k) \cdot \dots \cdot p_e(\underline{x}_0|\underline{x}_1 \dots \underline{x}_k)$$
 - again we have significant dependencies on preceding path vertices.
 - The last pdf $p_e(\underline{x}_0|\underline{x}_1 \dots \underline{x}_k)$ implements direct light sampling.



From Area to Direction sampling

- Area sampling is used only for initial light source samples $\underline{x}_0$ and direct light sampling
- In order to support **sampling of directions** ω_i^{i+1} the direction pdfs need to be converted to area pdfs.
- comparing the area with directional formulation of the rendering equation demands for a geometry factor:

$$p(\underline{x}_{i+1} | \underline{x}_0 \underline{x}_1 \dots \underline{x}_i) = p(\omega_i^{i+1} | \underline{x}_0 \underline{x}_1 \dots \underline{x}_i) \cdot G(\underline{x}_i, \underline{x}_{i+1})$$

- Here the visibility term V is always one:

$$p(\underline{x}_{i+1} | \underline{x}_0 \underline{x}_1 \dots \underline{x}_i) = p(\omega_i^{i+1} | \underline{x}_0 \underline{x}_1 \dots \underline{x}_i) \cdot \frac{\cos \theta_i^{i+1} \cos \theta_{i+1}^i}{\|\underline{x}_i - \underline{x}_{i+1}\|^2}$$

scene points $\underline{x}_{i+1}$ not visible from $\underline{x}_i$ are implicitly assigned $p(\underline{x}_{i+1} | \underline{x}_0 \underline{x}_1 \dots \underline{x}_i) = 0$ in direction sampling.



Bidirectional Path Tracing (BPT)

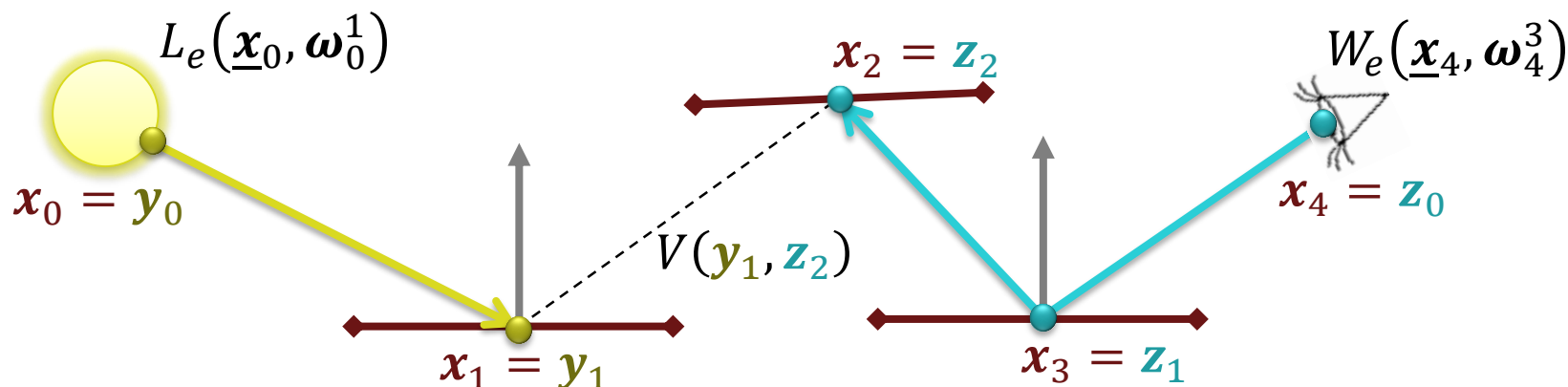
- BPT connects light and path tracing by generating one path in each direction and interconnecting them
- light path: $y = y_0 \dots y_s$, reversed eye path: $z = z_t \dots z_0$
- if $V(y_s, z_t) = 1$ one can connect paths:

$$x = x_0 \dots x_k = y^{\circ} z = y_0 \dots y_s z_t \dots z_0$$

- here s and t are edge lengths and connected to path of edge length $k = s + t + 1$.

- y and z are sampled independently: $p_{s,t}(x) = p_{l,s}(y)p_{e,t}(z)$

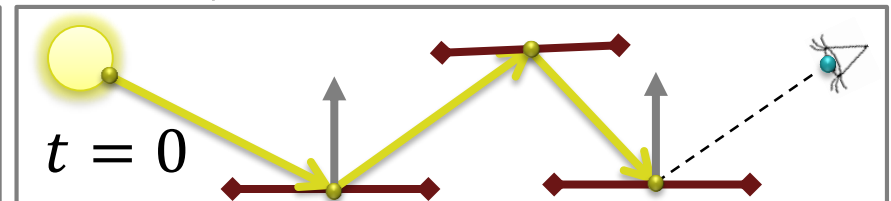
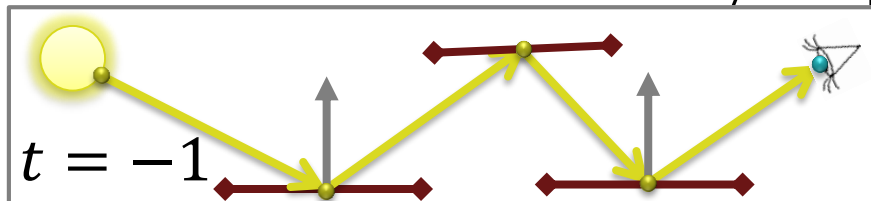
$$\begin{aligned} &= p_{l,s}(y)p_{e,t}(z) = p_l(y_0) \cdot \dots \cdot p_l(y_s) \cdot p_e(z_t) \cdot \dots \cdot p_e(z_0) \\ &= p_l(x_0) \cdot \dots \cdot p_l(x_s) \cdot p_e(x_{s+1}) \cdot \dots \cdot p_e(x_k) \end{aligned}$$



BPT Sampling techniques

- ◆ each tuple $\alpha = (s, t)$ with $s + t + 1 = k$ defines different sampling strategy p_α of path x with the special cases
- ◆ path tracing:
 - ◆ $s = -1$... no light path, eye path hits light [path tracing]
($\rightarrow$ light sources must be part of the scene, **ignored here**)
 - ◆ $s = 0$... light path consists of one light source point
(these paths correspond to direct light sampling)
- ◆ light tracing
 - ◆ $t = -1$... no eye path, light path hits eye [light tracing]
($\rightarrow$ eye lens must be part of scene, **also ignored here**)
 - ◆ $t = 0$... eye path consists of one lens point
(these paths correspond to direct eye sampling.
reason for secondary sample buffer)

important for
caustics!





BPT meets MIS

- Given a path x of edge length k we use multiple importance sampling over all strategies that can generate x to avoid bias. Recap: power heuristics:

$$\hat{f}_N^{MIS} = \sum_i \frac{1}{n_i} \sum_{j=1}^{n_i} \omega_i(x_{i,j}) \frac{f(x_{i,j})}{p_i(x_{i,j})}, \quad \omega_i^\beta(x) = \frac{(n_i p_i(x))^\beta}{\sum_k (n_k p_k(x))^\beta}$$

- Let us define $S_k = \{(0, k-1), (1, k-2), \dots, (k-1, 0)\}$
- We choose $n_\alpha \equiv 1$ for all $\alpha \in S_k$:

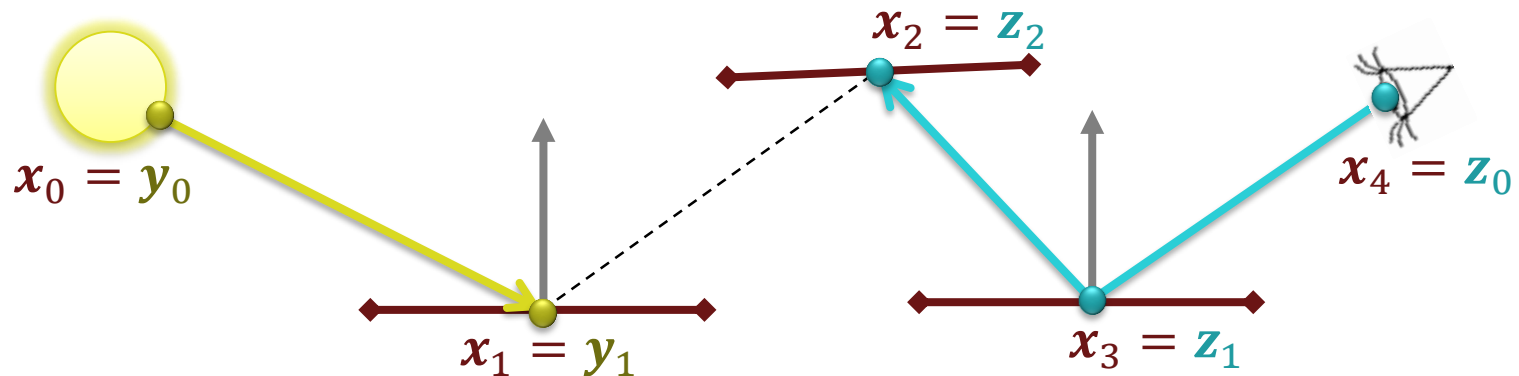
$$\hat{I}_{j,MIS} = \sum_{\alpha \in S_k} \omega_\alpha \frac{f_j(x)}{p_\alpha(x)}, \quad \omega_\alpha = \frac{p_\alpha^\beta}{\sum_{\tilde{\alpha} \in S_k} p_{\tilde{\alpha}}^\beta}$$

typically
 $\beta = 2$

- Conclusion:** for a given path sampled according to $\alpha \in S_k$ we therefore need to be able also to compute all other probabilities $p_{\tilde{\alpha}}(x)$ with $\tilde{\alpha} \in S_k$.

Weight Computations Example

- For a given prefix path combination $x_{1,2}$

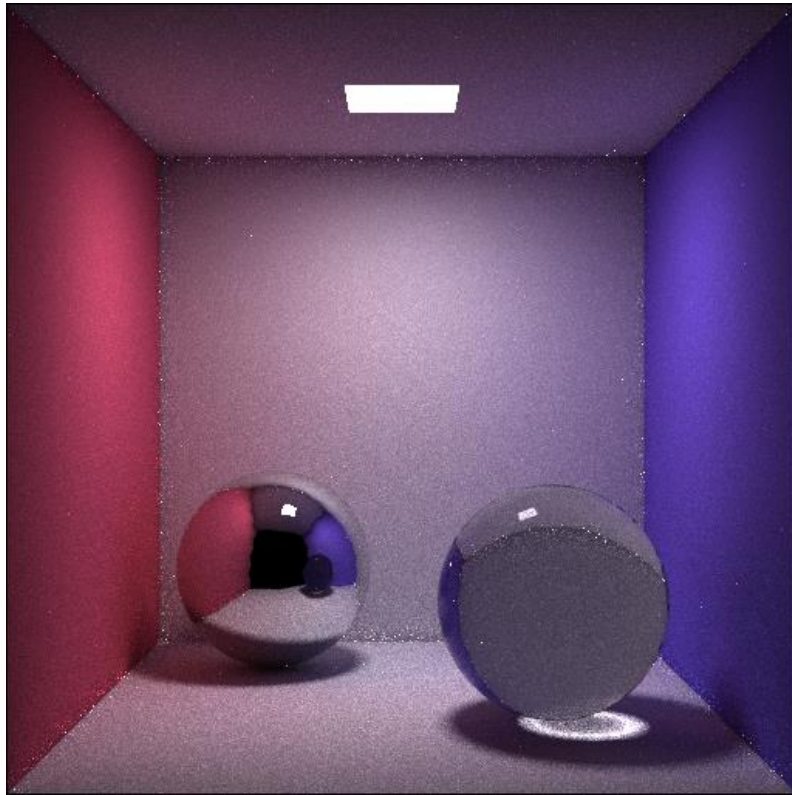


- we need to compute the pdfs of all other combinations

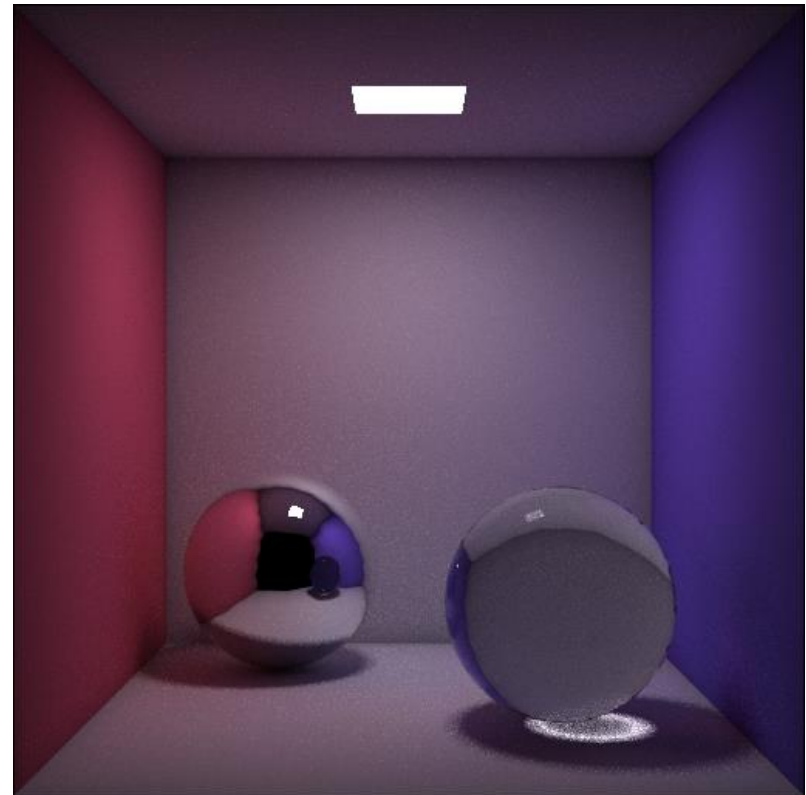
$p_{3,0}$	↑	• $x_0 = y_0 \rightarrow x_1 = y_1 \rightarrow x_2 = y_2 \rightarrow x_3 = y_3 \leftrightarrow x_4 = z_0$
$p_{2,1}$	↑	• $x_0 = y_0 \rightarrow x_1 = y_1 \rightarrow x_2 = y_2 \leftrightarrow x_3 = z_1 \leftarrow x_4 = z_0$
$p_{1,2}$	●	• $x_0 = y_0 \rightarrow x_1 = y_1 \leftrightarrow x_2 = z_2 \leftarrow x_3 = z_1 \leftarrow x_4 = z_0$
$p_{0,3}$	↓	• $x_0 = y_0 \leftrightarrow x_1 = z_3 \leftarrow x_2 = z_2 \leftarrow x_3 = z_1 \leftarrow x_4 = z_0$

BPT meets MIS

- Implementation of BPT without multiple importance sampling has additional artefacts where faces meet as the geometry term $1/r^2$ can become very large



without MIS



with MIS



Russian Roulette in BPT

- Russian Roulette is used to terminate the light and the eye paths if a coin flip with success probability $s_{\text{light},i} / s_{\text{eye},i}$ fails.
- To account for Russian Roulette we multiply in each tracing step s_{light} (s_{eye}) to the path probabilities:

$$p_l^{RR}(\underline{x}_0) = p_l(\underline{x}_0)$$

$$p_l^{RR}(\underline{x}_1 | \underline{x}_0) = s_{l,0} \cdot p_l(\underline{x}_1 | \underline{x}_0)$$

$$p_l^{RR}(\underline{x}_{i+1} | \underline{x}_0 \dots \underline{x}_i) = s_{l,i} \cdot p_l(\underline{x}_{i+1} | \underline{x}_0 \dots \underline{x}_i)$$

(similarly for the eye path)

- finally we combine: $p_{s,t}^{RR}(\underline{x}) = p_{s,l}^{RR}(\underline{y}) p_{t,e}^{RR}(\underline{z})$
- Veach proposes to set success probabilities $s_{*,i}$ to one for small i and for larger i one first samples the next direction ω_i^{i+1} and then computes

$$s_{*,i} = \min \left\{ 1, \frac{\rho(\underline{x}_i, \omega_i^{i-1}, \omega_i^{i+1})}{p(\underline{x}_{i+1} | \underline{x}_0 \dots \underline{x}_i)} \right\}$$

$$p_{s,t}(x) = p_l(x_0) \cdot \dots \cdot p_l(x_s) \cdot p_e(x_{s+1}) \cdot \dots \cdot p_e(x_k)$$



Incremental Weight Computation

- One can rearrange the weight computation:

$$\omega_\alpha = \frac{(p_\alpha^{RR}(x))^\beta}{\sum_{\tilde{\alpha}} (p_{\tilde{\alpha}}^{RR}(x))^\beta} = 1 / \sum_{\tilde{\alpha} \in S_k} \left(\frac{p_{\tilde{\alpha}}^{RR}(x)}{p_\alpha^{RR}(x)} \right)^\beta$$

- Thus only fractions $q_{\tilde{\alpha}} = \frac{p_{\tilde{\alpha}}^{RR}(x)}{p_\alpha^{RR}(x)}$ are relevant, which can be generated incrementally from $q_\alpha = \frac{p_\alpha^{RR}(x)}{p_\alpha^{RR}(x)} = 1$.

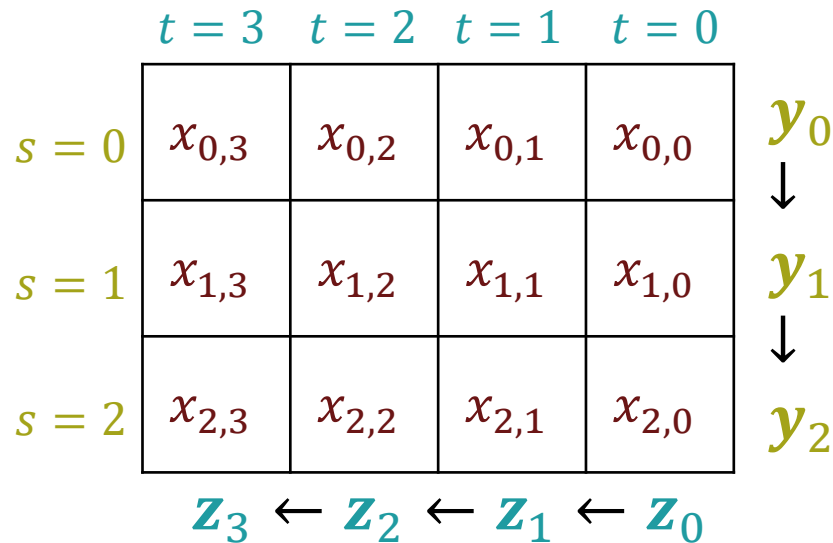
- lengthen light path: $q_{s+1,t-1} = q_{s,t} \cdot \frac{p_l^{RR}(x_{s+1} | x_0 \dots x_s)}{p_e^{RR}(x_{s+1} | x_{s+2} \dots x_k)}$

- lengthen eye path: $q_{s-1,t+1} = q_{s,t} \cdot \frac{p_e^{RR}(x_s | x_{s+1} \dots x_k)}{p_l^{RR}(x_s | x_0 \dots x_{s-1})}$



Efficient Sampling with path groups

- More efficient generation of path samples can be achieved by forming them in groups
- after sampling light and eye paths with RR one can combine all possible path prefixes with $\tilde{s} \leq s, \tilde{t} \leq t$



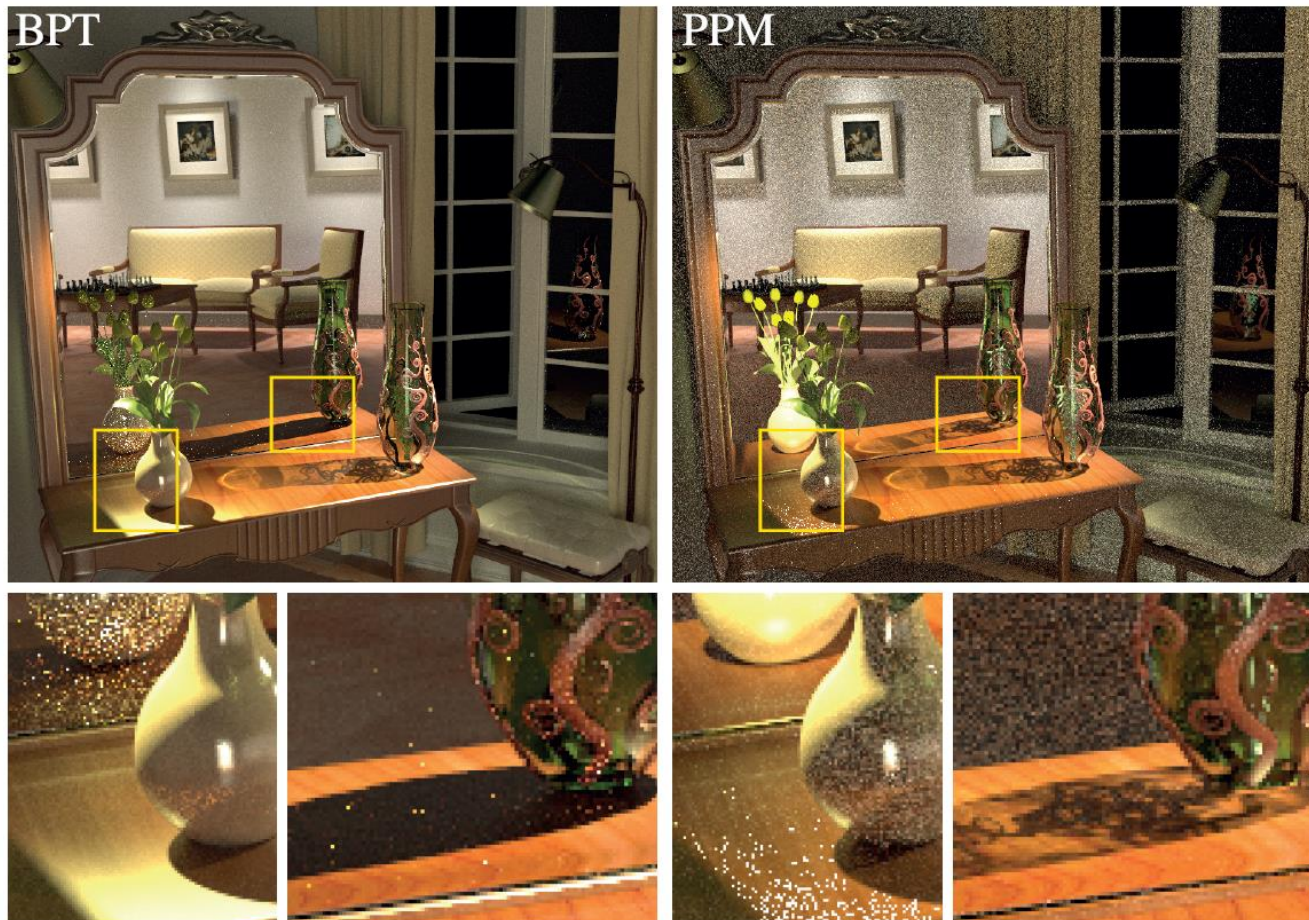
- again incremental computation helps

- For each path sample x we need to compute the contribution

$$C_\alpha = \omega_\alpha C_\alpha^* = \omega_\alpha \frac{f_j(x)}{p_\alpha^{RR}(x)}$$

- Each describes an efficient scheme to compute the weights ω_α and the unweighted contribution C_α^* incrementally during generation of all path samples x from a sample group.
- Another even more efficient scheme is described in Georgiev 2012 (see script on cache based methods)
- In both approaches most geometry factors in $f_j(x)$ cancel out with the pdf conversion geometry factors in $p_\alpha^{RR}(x)$ such that the directional pdfs can be used directly.

Is BPT perfect?



- ◆ no there are still problems with LSDSE-paths
- ◆ to solve this we need cache based methods



METROPOLIS LIGHT TRANSPORT

- ◆ **goal:** generate 2D pixel locations $\underline{x}_i$ distributed according to $p(\underline{x}) \propto f(\underline{x})$ with $f(\underline{x})$ given as grey scale image
- ◆ **idea:** start with random location $\underline{x}_0$ and then randomly permute by choosing $\underline{x}_{i+1}$ in pixel neighborhood of $\underline{x}_i$
- ◆ **question:** How to reject new sample such that resulting **chain of samples** is distributed according to $p(\underline{x})$?



$f(\underline{x})$



- ◆ **input:** $f: \Omega \rightarrow \mathbb{R}^+$

- ◆ **output:** sample chain

$$\Omega \ni x_i \sim p(x) = \frac{f(x)}{F}$$

with $F = \int_{\Omega} f(x) dx$.

Ansatz:

- ◆ start with $x_0 \in \Omega$

- ◆ define random process
from **process probability**
 $K(y|x)$

- ◆ generate chain

$$x_{i+1} \sim K(x_{i+1}|x_i)$$

Analysis:

- ◆ **propagation** of pdf:

$$p_{i+1}(y) = \int_{\Omega} K(y|x)p(x)dy$$

- ◆ in case of **convergence** we
arrive at equilibrium pdf p_{∞}

$$p_{\infty}(y) = \int_{\Omega} K(y|x)p_{\infty}(x)dy$$

- ◆ equilibrium is **stable** if
exchange of probability
between two points x and
 y is stable:
 $\text{prob}(x \rightarrow y) = \text{prob}(y \rightarrow x)$

$$\Leftrightarrow$$

$$K(y|x)p(x) = K(x|y)p(y)$$



Metropolis Sampling

- ◆ **output:** sample chain

$$\Omega \ni x_i \sim p(x) = \frac{f(x)}{F}$$

- ◆ equilibrium is **stable** if
 $K(y|x)p(x) = K(x|y)p(y)$

Approach:

- ◆ with **transition probability**
 $T(y|x)$
- ◆ & **acceptance probability**
 $a(y|x)$
- ◆ split process $K(y|x)$ into transition and acceptance
 $K(y|x) = T(y|x) \cdot a(y|x)$

Acceptance:

- ◆ derive $a(y|x)$ from equilibrium condition:
- ◆ $K(y|x)p(x) = K(x|y)p(y)$
- ◆ $T(y|x)a(y|x)f(x)/F = T(x|y)a(x|y)f(y)/F$

- ◆
$$\frac{a(y|x)}{a(x|y)} = \frac{T(x|y)f(y)}{T(y|x)f(x)}$$

- ◆ fast **convergence** has been shown for

$$a(y|x) := \min \left\{ 1, \frac{T(x|y)f(y)}{T(y|x)f(x)} \right\}$$

Lena Image example:

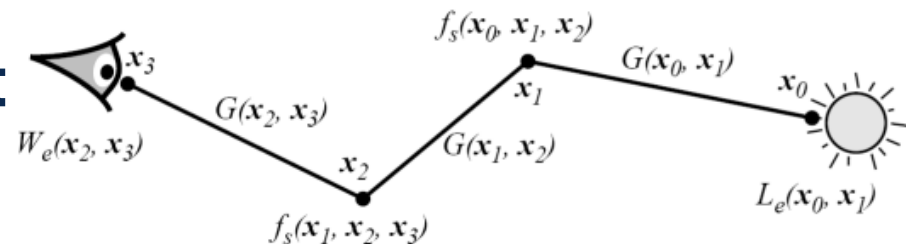
- ◆ $T(y|x)$... const on $n \times n$ neigh
- ◆ $a(y|x) = \min\{1, f(y)/f(x)\}$

Discussion of Metropolis Sampling

- ◆ Transition $x \rightarrow y$ is interpreted as mutation
- ◆ Samples in Markov chain are distributed according to f/F only after convergence. Before there is a bias.
- ◆ How many samples i need to be skipped at head of chain?
- ◆ Surrounding of samples with high contribution are explored by Metropolis sampler automatically.
- ◆ Variance can be minimized if acceptance probability is maximized. The choice of $T(y|x)$ significantly influences $a(y|x)$. Theoretically, the perfect choice is

$$T(y | x) = f(y) / F \Rightarrow a(y | x) = \min \left\{ 1, \frac{f(y)T(x | y)}{f(x)T(y | x)} \right\} \equiv 1$$

Metropolis Light Transport



Path Integral Formulation

- Starting point is measurement equation for pixel j

$$\Phi_j = \int_{A_p} \int_{A_q} W_{\text{emi}}(\underline{p} \leftarrow \underline{q}) L(\underline{p} \leftarrow \underline{q}) G(\underline{p}, \underline{q}) dA_q dA_p$$

- Recursively, plugging in tracing and reflection into incoming light, the integral can be reparameterized over paths. The integration domain is union of sets of all paths over increasing length k :

$$\Omega = \bigcup_{k=1}^{\infty} \Omega_k$$

- Over Ω_k a product measure is defined over one area measure dA of the scene surfaces per vertex in paths of length k : $d\mu_k = dA_1 \dots dA_k$
- From this a measure $d\mu = \sum_k \mu_k$ on the set of all paths Ω is defined
- This yields

$$f_j(x) = \omega_j f(x) \quad \omega_j(x) = W_{\text{emi}}(\underline{p}_k \leftarrow \underline{p}_{k-1})$$

$$\Phi_j = \int_{\Omega} f_j(x) d\mu$$

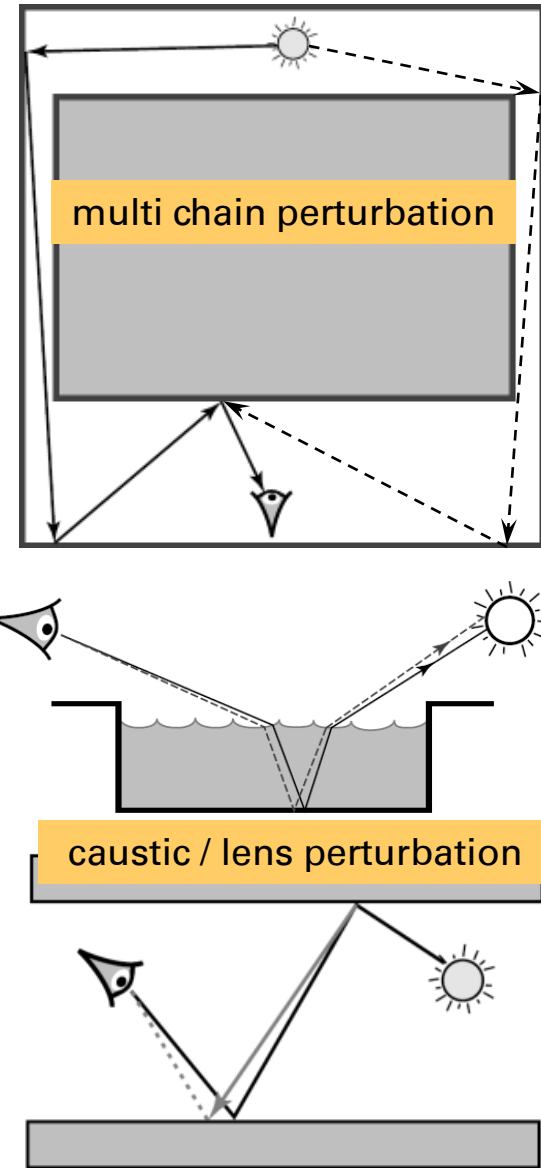
$$f(x) = L_{\text{emi}}(\underline{p}_2 \leftarrow \underline{p}_1) \prod_{i=1}^{k-1} f_r(\underline{p}_{i+1} \leftarrow \underline{p}_i) G(\underline{p}_{i+1}, \underline{p}_i)$$

Application of MLT to Pathtracing

- ◆ each Markov Chain generates a complete image
- ◆ mutations rearrange paths by
 - ◆ Replacement of parts. Important: all possible paths must be reachable (Ergodicity)
 - ◆ Movement of path points to explore caustics
- ◆ Monte-Carlo Integration is simply done by counting the contributions per pixel j :

$$\Phi_j = \int \omega_j f(x) d\mu \Rightarrow$$
$$\Phi_j^* = \frac{1}{N} \sum_{i=1}^N \frac{\omega_j(x_i) f(x_i)}{p(x_i)} = \frac{F}{N} \sum_{i=1}^N \omega_j(x_i)$$

- ◆ F is the total light power over the whole image and can be estimated for example by bidirectional pathtracing.



Algorithmus

- ◆ Bidirectional Pathtracing
 - ◆ estimate F
 - ◆ choose n representative seed paths for Markov Chains
- ◆ Metropolis sampling of each Chain
 - ◆ **input:** $x = x_0$, compute $f(x)$
 - ◆ Initialize to black image
 - ◆ **iterate N times**
 - ◆ $[y, T(y|x), T(x|y)] = \mathbf{mutate}(x)$, compute $f(y)$
 - ◆ $a = \min \left\{ 1, \frac{T(x|y) f(y)}{T(y|x) f(x)} \right\}$
 - ◆ **if** (**random**() $< a$) $[x, f(x)] = [y, f(y)]$
 - ◆ **record sample x in image**
- ◆ termination criterion is defined based on image comparison
- ◆ result is average over per chain images

Ergebnisse

- ◆ Comparison of Metropolis Light Transport (MLT) with Pathtracing (PT) and Bidirectional Pathtracing (BPT)
- ◆ Compute different error norms (L_1 , L_2 and L_∞) with respect to reference image of different methods for fixed computation time
- ◆ The following tables shows error values of other methods (PT and BPT) over error value of MLT:

Test Case	PT vs. MLT			BPT vs. MLT		
	l_1	l_2	l_∞	l_1	l_2	l_∞
Fig. 5	7.7	11.7	40.0	4.2	6.5	6.1
Fig. 6 (top)	2.4	4.8	21.4	0.9	2.1	13.7
Fig. 7	3.2	4.7	5.0	4.2	6.5	6.1

Figure 5

BPT



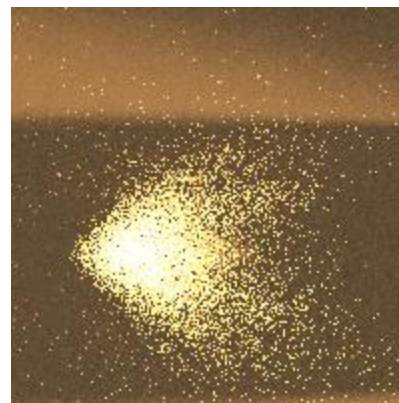
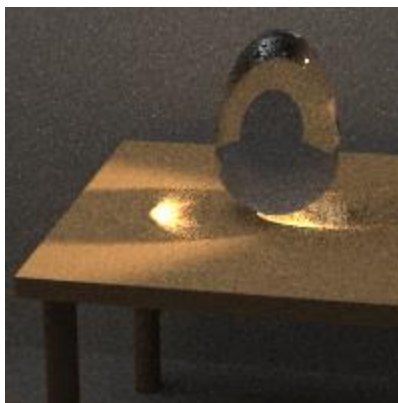
BPT vs. MLT		
l_1	l_2	l_∞
5.2	4.9	13.2

MLT



Figure 6

BPT



BPT vs. MLT

l_1

l_2

l_∞

0.9

2.1

13.7

MLT

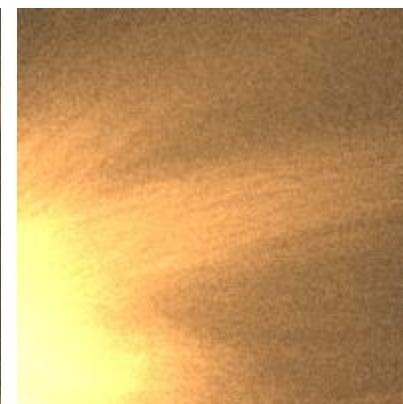
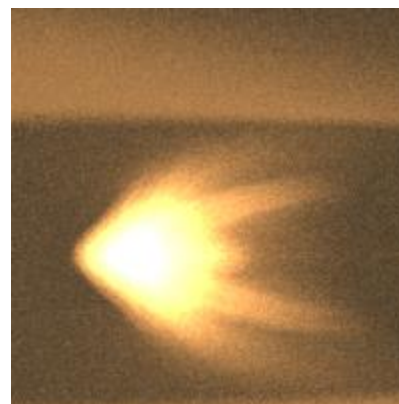
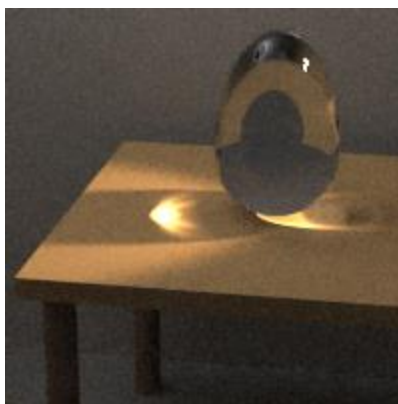
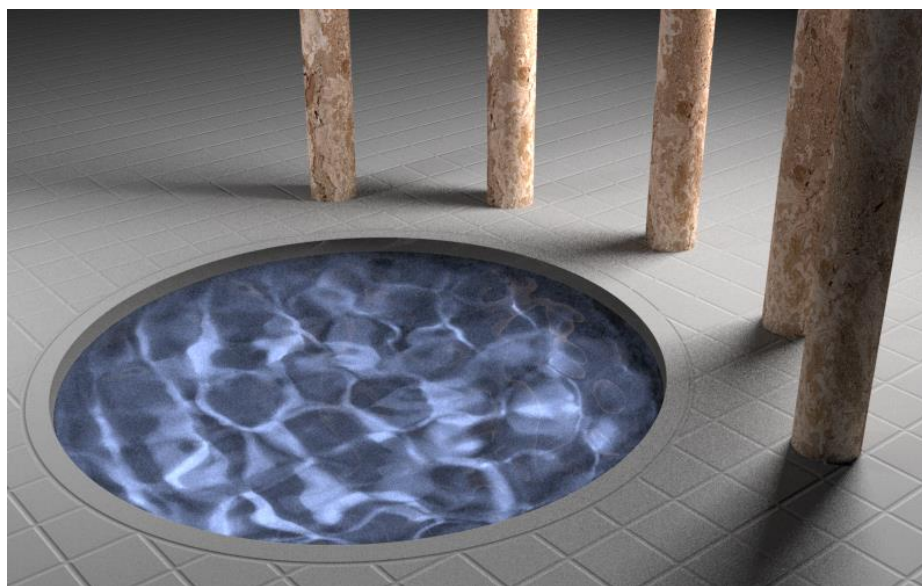
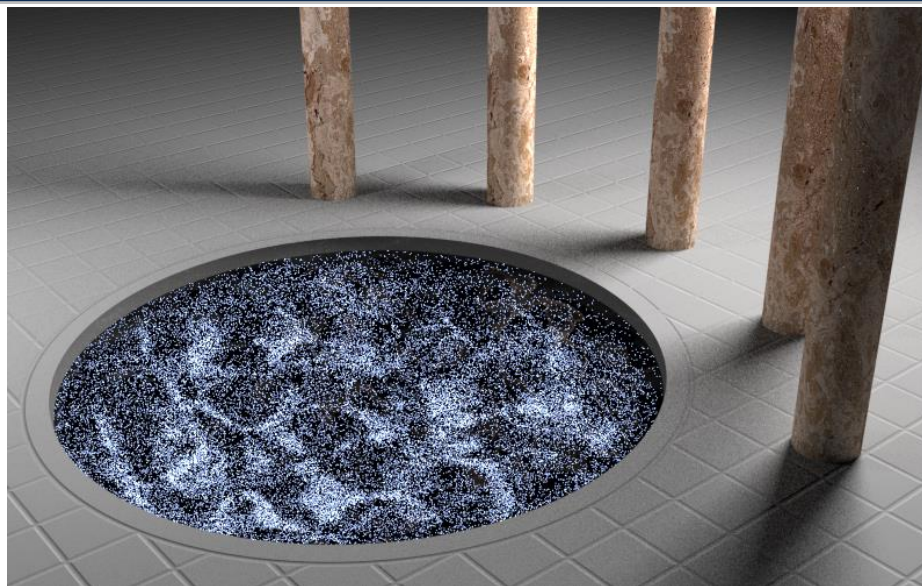


Figure 7

BPT

BPT vs. MLT		
l_1	l_2	l_∞
4.2	6.5	6.1

MLT



Literature

- ◆ Metropolis Light Transport, [Eric Veach](#) and [Leonidas J. Guibas](#), *SIGGRAPH 97 Proceedings* (August 1997), Addison-Wesley, pp. 65-76 ([pdf](#))