Realistic Image Synthesis

Bidirectional Path Tracing & Reciprocity

Philipp Slusallek
Karol Myszkowski
Gurprit Singh
Path Sampling Techniques

- Different techniques of sampling paths from both sides
  - Numbers in parenthesis are # of vertices traced from light/camera, resp.
  - See later, for many light methods
Results from Different Techniques

- a) Unidirectional (eye tracing)
- b) Unidirectional + next event
- c) Next event (light tracing)

• Results from tracing 40 paths per pixel
Results from Different Techniques

- Results from tracing 40 paths per pixel
  - f): “Problem of insufficient techniques“ for sampling SDS paths
BIDIRECTIONAL PATH TRACING
Light & Path Tracing

**Problem:**
- Probability of hitting the camera from the light sources is almost zero
- Probability of hitting the light source is often also very small
  - Next Event Estimator: Try to find a direct connections
    - Non-optimal (e.g. on mirror surface)
    - Ignores secondary light sources (e.g. via mirror, at caustics)

**Approaches:**
- Bidirectional Path Tracing
  - Combination of eye and light paths
  - Weighted MC sampling for best results
  - Includes Vertex Connection and Merging (VCM, later)
- Metropolis-Sampling [Veach´1997] (see later)
  - Random variation and mutations of bidirectional paths
  - Very well suited for very complex light paths
  - Unbiased but relatively complex algorithms
  - Uneven convergence
Bidirectional Path-Tracing

- **Idea: Combine Paths from Both Sides**
  - Generate path from the light sources and the camera
  - Connect paths deterministically (every pair of two hit points)
    - Different probabilities of generating paths
  - Compute weighted sum of contributions (→ MIS)

- **References:**
  - Lafortune et al., Bidirectional Path-Tracing, [CompuGraphics`93]
  - Veach, Guibas, Bidirectional Estimators for LightTransport, [EGRW´94, Siggraph´95]
Solving the Rendering Equation

- Von Neumann Expansion of Measurement Equation

\[
I_p = \int_{S \times S} L_e(x \rightarrow x') G(x \rightarrow x') W_p(x \rightarrow x') dA(x) dA(x') + \\
+ \int_{S \times S \times S} L_e(x \rightarrow x') G(x \rightarrow x') f_r(x \rightarrow x' \rightarrow x'') G(x' \rightarrow x''') W_p(x') \\
\rightarrow x''') dA(x) dA(x') dA(x''') + \ldots \text{ with } G(x, y) = \frac{\cos \theta_x \cos \theta_y}{||x - y||^2}
\]

- Independent estimation of all paths with fixed lengths
- Bidirectional generation of paths
- Weighted MC integration for each term (MIS)
- More efficient by reusing costly paths (i.e. visibility samples) multiple times
- Typically: One pair of paths per pixel sample
Bidirectional Path-Tracing

• Notation
Bidirectional Path-Tracing

• Generating Light Paths (example)
  – On the light source

\[
p(x, \Theta_x) = \frac{L_e(x, \Theta_x) \| \Theta_x \cdot N_x \|}{\Phi}
\]

\[
\Phi = \int_{\Omega_x^+} \int_{A} L_e(x, \Theta_x) \| \Theta_x \cdot N_x \| d\Theta_x dA_x
\]

• Generating Eye Paths (example)
  – On the eye/camera (via point in the scene)

\[
p(y, \Theta_y) = \frac{g(y, \Theta_y) W(y, \Theta_y) \| \Theta_y \cdot N_y \|}{G}
\]

\[
G = \int_{\Omega_y^+} \int_{A} g(y, \Theta_y) W(y, \Theta_y) \| \Theta_y \cdot N_y \| d\Theta_y dA_y
\]

– \( g() \): 1, if point is visible in this direction
Bidirectional Path-Tracing

- **Extension of Paths at Hit Points**
  - Identical for both directions
    - Reciprocity of BRDF under reflection
    - Use whatever BRDF sampling technique suits best
      - But must be a joint probability (conditioned on the previous point)
        - This does include uniform probability on any surface
        - (But not a point generated from some other point, e.g. due to occlusion.)
  - E.g.

\[
p(\Theta) = f_r(\Theta_{x_i}, x_{i+1}, \Theta) \parallel \Theta_{x_{i+1}} \cdot N_{x_{i+1}}
\]
Bidirectional Probabilities

• Probabilities of Paths $\pi$ in Bidirectional Path Tracing
  – Different locations of *vertex connections* (see VCM later)
  – $k$: length of paths (# of transports or segments)
  – $m$: # of vertices generated from light source
    • $0$: None
    • $1$: Vertex on light source
    • $2$: Vertex on light source and directional sample
    • Etc.
  – Similar for paths from the eye
  – $p_{k,m}(\pi)$: Probability to choose path $\pi$ with method $(k,m)$

\[
\begin{align*}
p_{2,0}(\pi) & \\
p_{2,1}(\pi) & \\
p_{2,2}(\pi) & \\
p_{2,3}(\pi) & \\
\end{align*}
\]
Mathematical Formulation

- Rendering Equation with Area Parametrization

\[ L(p_1 \rightarrow p_0) = \int_{A} L_e(p_2 \rightarrow p_1) f(p_2 \rightarrow p_1 \rightarrow p_0) G(p_2 \rightarrow p_1) dA(p_2) + \]
\[ \int_{A} \int_{A} L_e(p_3 \rightarrow p_2) f(p_3 \rightarrow p_2 \rightarrow p_1) G(p_3 \rightarrow p_2) f(p_2 \rightarrow p_1 \rightarrow p_0) G(p_2 \rightarrow p_1) dA(p_3) dA(p_2) + \cdots \]

where \[ G(p_2 \rightarrow p_1) = \frac{V(p_1, p_2) \cos(\theta_{p_1}) \cos(\theta_{p_2})}{|p_2 - p_1|^2} \]
Mathematical Formulation

• **Path Formulation**
  
  - $\pi_i$: Path of length $i$
  
  \[
  L(p_1 \rightarrow p_0) = \sum_{i=1}^{\infty} L(\pi_i(p_1, p_0)) = \sum_{i=1}^{\infty} L(\pi_i)
  \]
  
  \[
  L(\pi_i) = \int \int \cdots \int \frac{L_e(p_i \rightarrow p_{i-1})}{A} \left( \prod_{j=1}^{i-1} G(p_{j+1} \rightarrow p_j) f(p_{j+1} \rightarrow p_j \rightarrow p_{j-1}) \right) dA(p_2) \cdots dA(p_i)
  \]

• **Connection Throughput $T(\pi)$ of a path $\pi$**

  \[
  T(\pi_i) = \prod_{j=1}^{i-1} G(p_{j+1} \rightarrow p_j) f(p_{j+1} \rightarrow p_j \rightarrow p_{j-1})
  \]
  
  \[
  L(\pi_i) = \int \int \cdots \int L_e(p_i \rightarrow p_{i-1}) T(\pi_i) dA(p_2) \cdots dA(p_i)
  \]

• **With Measurement**

  \[
  I = \int \int A A_{\text{pixel}} L(p_1 \rightarrow p_0) G(p_1 \rightarrow p_0) W(p_1 \rightarrow p_0) dA(p_0) dA(p_1)
  \]
  
  \[
  I = \sum \int \int \cdots \int L_e(p_i \rightarrow p_{i-1}) T(\pi_i) G(p_1 \rightarrow p_0) W(p_1 \rightarrow p_0) dA(p_0) \cdots dA(p_i)
  \]
Mathematical Formulation

- **Path Tracing with Russian Roulette**

\[ L(p_1 \rightarrow p_0) = \sum_{i=1}^{\infty} L(\pi_i) = L(\pi_1) + \frac{1}{1-q_2} \sum_{i=2}^{\infty} L(\pi_i) \]

- And similar for higher path lengths

- **How to choose the probabilities of sample points**
  - Whatever works, from wherever (!!!), e.g.
    - Area (uniform):
    - Solid angle, depending on direction from previous sample:
    - Any other joint probability that integrates to one over all surfaces and is non-zero where there could be a contribution
    - Must be a conditional probability, based on the previous point

- **Splitting of BRDFs or Emissions**
  - Make sure all path are accounted for!
  - Make sure no path is counted multiple times, either!
Example

- Light tracing (one eye ray, 1st generation only)
Example

- Standard MC Path Tracing (same number of paths)
Example

• Contribution of Different Paths

[Not shown: direct connection eye to light + all from light]

One reflection
One step from the eye (plus direct connection to light)

Two reflections
l: Two steps from the eye
r: One step from the eye, one step from light source

Three reflections
l: Three steps from the eye
m: Two steps from the eye, one from light source
r: One step from the eye, two from the light sources
Bidirectional Path-Tracing

• Combination of Estimators
  – Every option of generating a specific path \( \pi \) defines its own estimator with given \( p_{k,m}(\pi) \)
  – Weighted MC sampling provides new combined estimator of a bi-directionally generated path
    \[
    \bar{C} = \sum_{i=1}^{N_e} \sum_{j=1}^{N_l} w_{ij} \langle C_{ij} \rangle
    \]
  – \( N_e \): # reflections on eye paths
  – \( N_l \): # reflections on light paths
  – \( w_{ij} \): weights for combination
Combination of Estimators

- **Example:**
  - Four paths between LS and eye
  - Weighted with three estimators
    - A, B, C
  - Selection with maximum heuristics
    - Choose $p_X(\pi)$ maximum
  - Area of rectangles is constant across A, B, C
    - $f/p*\pi$
  - Width corresponds to $p_X(\pi)$
Implementation

Example: Maximums Heuristics

\[ S = 0 \]
\[ P = \text{GenerateBiDirPaths()} \]
for light\_segs = 0 to \( P.\text{max\_light\_segments} \)
  for eye\_segs = 0 to \( P.\text{max\_eye\_segments} \)
    \[ SP = \text{ChooseSubPath}(P, \text{eye\_segs}, \text{light\_segs}) \]
    // Compute best estimator (Max-Heuristics)
    \[ p = 0; \text{segments} = \text{eye\_segs} + \text{light\_segs}; \]
    // Iterate over different estimators:
    // assuming \( j \) segments generated
    // from camera
    for estimator = 0 to \( \text{segments} \)
      \[ p_t = \text{Probability}(SP, \text{estimator}) \]
      if \((p_t > p)\) \( p = p_t \)
    \[ S = S + SP.f/p \]
return \( S \)
Example

Bidirectional Path Tracing

Path Tracing
Contributions of Different Paths

More camera segments

\( p_{2,x} \)

\( p_{3,x} \)

\( p_{4,x} \)

\( p_{5,x} \)

More light segments (right: \( n-1 \))
Comparison w/ Path Tracing

- **Brute Force Method**
  - Only use $p_{n,0}$ method to generate paths
    - No points sampled from light source
  - Highly inefficient:
    - Probability of hitting the light is almost zero
    - Especially for point lights :)  

- **Path Tracing with Direct Lighting Optimization**
  - Next Event Estimation
  - Use $p_{n,0}$ and $p_{n,1}$ paths only
    - Path from the eye/camera plus direct connection to point sampled on light source
NON-SYMMETRIC SCATTERING IN LIGHT TRANSPORT ALGORITHMS
Use of Shading Normals

- **Shading Normals**
  - It is common to shade with respect to arbitrary normals
    - E.g. specified as normals at each triangle vertex
  - Allow many neat tricks
    - Smooth surface even though real surface is tessellated
    - Bump mapping, normal mapping, …

- **Problem**
  - Use of shading normals $\theta'$ is generally not energy conserving

\[
L_r = \int_{\Omega_+} f_r(\omega_o, x, \omega_i) \cos \theta_i \, d\omega_i
\]
\[
= \int_{\Omega_+} f'_r(\omega_o, x, \omega_i) \frac{\cos \theta'_i}{\cos \theta_i} \cos \theta_i \, d\omega_i
\]

- Can “generate” energy
Use of Shading Normals

- **Energy “Generator”**
  - Light is received by an apparently small surface $\rightarrow$ some density

- And emitted from an apparently much larger one, w/ same density
Use of Shading Normals

Correct results

Wrong results
Use of Shading Normals

• Solution
  – Unfortunately there seems to be no good solution to the problem
  – Except not using shading normals :-(
    • Or making them differ as little as possible from geometric normals
Power versus Radiance

- **Light tracing and Refraction**
  - Distribution of “photons” carrying a certain energy/power
  - Power/energy does not change when photon is refracted

- **Ray Tracing and Refraction**
  - Consider
    - uniform illumination
    - a point below a refracting surface
  - If no light is absorbed at the surface then the same power comes through a smaller solid angle
  \[ \text{increased radiance} \]

\[
L_t = \frac{\eta_i^2}{\eta_i^2} L_i
\]
Power versus Radiance

Correct image rendered with particle tracing

Incorrect image rendered assuming the BRDF is symmetric also for refraction