LIGHT TRANSPORT SIMULATION FOR PRODUCTION RENDERING: CURRENT TRENDS AND CHALLENGES
Image synthesis (Rendering)

- Create a 2D picture of a 3D world
- Visibility and Visual Appearance
- Illumination
  - Shading
    - BSDF (Bidirectional Scattering Distribution Function)
      - BRDF (reflectance)
      - BTDF (transmittance)
  - Light Transport
    - Direct illumination
    - Indirect (global) illumination
Realism

- Shadows
- Reflections (Mirrors)
- Transparency
- Inter-reflections
- Detail (Textures...)
- Realistic Materials
- And many more
Rendering strategies

- Rasterization-based
- Ray Tracing-based
Ray tracing-based rendering

- uses physics to simulate the interaction between matter and light (shading) as well as the light transport around the scene.

When measurable/predictive photorealism is the primary goal....
Visual consistency
I – Light Transport
Ray Tracing-based light transport

Archviz

Automotive

Movies

Product Design

Advertising
Major players

- Arnold
- Renderman
- V-Ray
- iRay
- Maxwell
- Hyperion
- Manuka
- ...

...
Ray Tracing-based Light transport

- **Arnold renderer**
  - 2006, Monster House (Sony Imageworks)
  - Path Tracing algorithm

*Movies*

Image courtesy of Columbia Pictures.
© 2006 Columbia Pictures Industries, Inc.
Light transport algorithms in production

“Integrators”
  - Path Tracing
  - Bidirectional path tracing (BDPT)
  - Vertex Connection and Merging (VCM)
The energy leaving $x$ in direction $\omega_o$ is the sum of:

- The energy $L_e$ it emits on its own, without external influence; plus...
- The energy $L$ arriving at $x$, from all directions $\omega_i$ around it, that gets transformed to direction $\omega_o$.
  - The properties of the material $f_r$ (BRDF) and its geometry $N_x$ are the factors that transform that energy.

\[
L(x \rightarrow \omega_o) = L_e(x \rightarrow \omega_o) + \int_{\Omega_x} f_r(x, \omega_i, \omega_o) L(x \leftarrow \omega_i)(N_x \cdot \omega_i) \, d\omega_i
\]
Monte Carlo integration

- General approach to numerical evaluation of integrals

Integral:

\[ I = \int f(x) \, dx \]

Monte Carlo estimate of \( I \):

\[ \langle I \rangle = \frac{f(x_i)}{p(x_i)}, \quad x_i \overset{\text{i.i.d.}}{\sim} p(x) \]

Correct "on average" (unbiased):

\[ E[\langle I \rangle] = I \]
Estimating the Rendering Eq.

\[ L(x \rightarrow \omega_o) = \int_{\Omega_x} f_r(x, \omega_i, \omega_o) L(x \leftarrow \omega_i) (N_x \cdot \omega_i) \, d\omega_{\Omega_x} \]

- Instead of considering all of the directions, we generate \( N \) random directions \( \omega_i \) with pdf \( p(\omega_i) \).
- Then we evaluate the integrand for each direction \( \omega_i \) and divide it by \( p(\omega_i) \).
- In the end, we average everything.

\[ L(x \rightarrow \omega_o) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f_r(x, \omega_i, \omega_o) L(x \leftarrow \omega_i) (N_x \cdot \omega_i)}{p(\omega_i)} \]
**Which pdf?**

- **Importance sampling** means to choose the most efficient sampling (less variance) method for a situation.
- Monte Carlo estimator converges more quickly if the samples are taken from a distribution pdf(x) that is similar to the function f(x) in the integrand.
Importance sampling in the rendering equation

\[ L(x \rightarrow \omega_o) \approx \frac{1}{N} \sum_{i}^{N} \frac{f_r(x, \omega_i, \omega_o) L(x \leftarrow \omega_i)(N_x \cdot \omega_i)}{p(\omega_i)} \]

- The integrand is a product of three functions that depend on \( \omega_i \)
- Importance sampling means either:
  - BRDF sampling
  - Light sampling
  - Cosine-weighted sampling
Path Tracing

- Generate N random primary rays over a pixel area (performs antialiasing as well)
- The pixel radiance is calculated by using the MC integration of N stochastic paths
- Path building: ray tracing recursively, for each intersection:
  - Compute direct illumination with ONE shadow ray (next event estimation)
  - Shoot stochastically ONE secondary ray according to the material properties (BRDF) of the surface (importance sampling)
- Support paths of the form $L(D|S)*E$ (Eckbert notation)
Monte Carlo Integration: variance as noise

Variance $\propto \frac{1}{\sqrt{N}}$

Cornell box Path Tracing 1spp;

Cornell box Path Tracing 4spp;

Cornell box Path Tracing 16spp;

Cornell box Path Tracing 512spp;
Russian Roulette

- Probabilistic way of stopping recursion – unbiased MC method
- Define the probability $\alpha$ of continuing the recursion and
- Before tracing a secondary ray pick an uniform random number, $\xi$ (both between $[0, 1]$)
  - If $\xi \leq \alpha$ then shoot the secondary ray and divide its contribution by $\alpha$
  - else $\xi > \alpha$ stop the recursion

$$L(p \neg Y) \overset{1}{\overset{\text{?}}{\leftarrow}} L(p \neg Y_i) : 0$$
Multiple Importance Sampling (MIS)

Direct illumination

- Diffuse BRDF

- Glossy BRDF
Two Poor Estimators and a Good One

Sampling the light sources

Sampling the BSDF

MIS: Combining both estimators
Path tracing finally catches up

Advanced Path Tracing in RenderMan

Per Christensen
Pixar Animation Studios
SIGGRAPH 2017, Los Angeles
Path tracing finally catches up

- Sci-Tech Oscars
  - 2014
    - PBRT (Physically Based Ray Tracer) from Matt Pharr and Greg Humphreys
  - 2017
    - CGI Studio (Blue Sky Studios)
    - Arnold (Solid Angle)
    - V-Ray (Chaos Group)
Path Tracing – Why?

- Physics-based light transport simulation
  - Accuracy
  - Visual consistency
- Ease of use
- Single pass (no baking)
- Network rendering
- Progressive rendering (quick preview)
Path Tracing: limitations

- A lot of indirect illumination in the scene

- Solution: BDPT – Bidirectional Path Tracing
BDPT – Bidirectional Path Tracing

\[ L_p = \sum_{s=0}^{N_E} \sum_{t=0}^{N_L} w_{s,t} V(s, t) C_{s,t} \]
Results

BDPT, 25 samples per pixel

PT, 56 samples per pixel
Reference solution

Bidirectional path tracing
BDPT - Handling Specular surfaces

- Some paths sampled with zero (or very small) probability – Reflection of caustics problem
BDPT vs Photon mapping (PM)

Monte Carlo sampling and Vertex connection

Bidirectional path tracing

Density estimation

Photon mapping
Vertex connection & merging (VCM)
Georgiev et al. [2012,2013]

**Stage 1: Light sub-path sampling**
- a) Trace sub-paths
- b) Connect to eye
- c) Build search structure

**Stage 2: Eye sub-path sampling**
- a) Vertex connection
- b) Vertex merging
- c) Continue sub-path
Vertex Connection & Merging (VCM)

Georgiev, Křivánek, Davidovič, Slussalek; SIGGRAPH Asia 2012
II - Ray Tracing Acceleration
Intersection testing

Two Approaches:

- **Space partitions**: Decompose space into cells and assign primitives to the cells in which they fall.
  - A primitive may appear in multiple cells

- **Object partitions**: Group objects into clusters
  - Cluster volumes may overlap
Subdivision Techniques

Bounding Volume Hierarchy

Grid

KD-tree
Our Approach on GPUs with Grids

Vasco Costa, PhD [2015]
Grid Pros and Cons

Pros
- fast $O(N)$ construction speed
  - fast previews, dynamic geometry

Cons
- For typical scenes Grid does not adapt to empty space and local complexity
  - slow rendering speed
    - Time is wasted tracing the ray through empty grid cells
    - Local complexity (too many objects in each grid cell): unbalanced computing load
- empty grid cells memory footprint
Rectilinear grids partition the scene along regions of similar geometric complexity while a regular grid partitions using equal sized areas.
Results

**Rectilinear grids** provided better load balancing for non-homogeneous scenes than a **regular grid** improving overall **rendering performance**

![Regular Grid](image1.png)

67 million r/t isects

![Rectilinear Grid](image2.png)

26 million r/t isects
1. Assign Morton codes
2. Sort primitives
3. Generate hierarchy
4. Fit bounding boxes

$p_x = 0.1010$
$p_y = 0.0111$
$p_z = 0.1100$
1. Assign Morton codes
2. Sort primitives
3. Generate hierarchy
4. Fit bounding boxes

$p_x = 0.1\underline{0}100 1 0$
$p_y = 0.01\underline{0}1 1 1 1$
$p_z = 0.110\underline{0} 1 0 0$
LBVH - Lauterbach et al. [2009]

1. Assign Morton codes
2. Sort primitives
3. Generate hierarchy
4. Fit bounding boxes

\[ p_x = 0.1011010 \]
\[ p_y = 0.11110010 \]
\[ p_z = 0.111000 \]
LBVH - Lauterbach et al. [2009]

1. Assign Morton codes
2. Sort primitives
3. Generate hierarchy
4. Fit bounding boxes
LBVH - Garanzha et al. [2011]

1. Assign Morton codes
2. Sort primitives
3. Generate hierarchy
4. Fit bounding boxes
Garanzha et al. [2011]

Level 0
1 node

Level 1
2 nodes

Level 2
3 nodes

Level 3
1 node

0 1 2 3 4 5 6 7
0 0 0 0 1 1 1 1
0 0 1 1 0 0 0 0
0 1 0 0 1 1 0 1
0 1 0 0 1 1 0 1
Treelet Restructuring Bounding Volume Hierarchy (TRBVH)

Karras and Aila [2013]

Improvements:

- More efficient parallelism: process the internal nodes all in parallel
- Higher tree quality
  - treelet reconstruction by using the surface area heuristic (SAH)
Our compressed TRBVH
Lousada, Costa, Pereira [2017]

- Reduction of memory footprint (50%) and bandwidth (12-22%) through BVH compression
- Reduced number of memory accesses through mesh compression leading to increase in performance (FPS)
Some results

<table>
<thead>
<tr>
<th>ASIAN DRAGON</th>
<th>Karras</th>
<th>Node Compression</th>
<th>Node + Mesh Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel</td>
<td>Time (ms)</td>
<td>Time (ms)</td>
<td>Time (ms)</td>
</tr>
<tr>
<td>BVH Creation</td>
<td>424.639</td>
<td>424.639</td>
<td>424.639</td>
</tr>
<tr>
<td>Node compression</td>
<td>0</td>
<td>16.328</td>
<td></td>
</tr>
<tr>
<td>Mesh Compression</td>
<td>0</td>
<td>0</td>
<td>24.433</td>
</tr>
<tr>
<td>Rendering</td>
<td>246.357</td>
<td>194.436</td>
<td>160.952</td>
</tr>
</tbody>
</table>
Efficient GPU Cone Tracing of Human Hair

[Martins, Costa, Pereira 2017]

Acceleration techniques:
• Optimized TRBVH-based acceleration data structure
• Traversal with a cone-based solution
Ray Classification on GPUs
Costa, Pereira, Jorge [2015] - Prémio José Luís Encarnação

- Rational:
  - the cost of ray sorting should be offset by an increase in performance
- Ray sorting by exploiting spatial coherence
  - Minimize thread divergence on GPU
- Survey of different ray classification techniques to accelerate stochastic ray-tracing for Ambient Occlusion
- Best results shown with Compress-Sort-Decompress (CSD) scheme from Garanzha & Loop 2010 used for soft shadows
Compress-Sort-Decompress (CSD)

- 5D rays hashing (quantization) into 32-bit unsigned integer keys (24 bits for the origin and 8 bits for the direction)
- Compression via run-length-encoding (RLE)
Hybrid Whitted ray-tracer (rasterization with G-buffers)

Ray sorting to store spatially coherent rays in consecutive memory locations – use of CSD scheme

Improved cone-sphere hierarchy from D. Roger et al. 2007

Results shown reduction of number of intersections around 50% for shadow rays and 25% for reflection rays
Hierarchy Construction

- Bottom-up
Hierarchy Construction

• Bottom-up
Hierarchy Construction

- Bottom-up
Hierarchy Traversal

Intersection
Hierarchy Traversal

Intersection

Triangle
Hierarchy Traversal

Intersection

Triangle
III - Challenges
Open issues in light transport simulation

- Robustness

  - None of the existing algorithms works for ALL scenes

  - Future direction: Metropolis Light Transport-based techniques
Open issues in light transport simulation

- Interactive Path-Tracing
  - 1080p – 30Hz only possible with few (short) paths per pixel
  - Very noisy images
- Deep learning for Denoising?
Milestone in Denoising MC Image Sequences
- Chaitanya et al. [2017] - Interactive Reconstruction of Monte Carlo Image Sequences using a Recurrent Denoising Autoencoder

Training sequences

SponzaDiffuse  SponzaGlossy  Classroom
Ray-Tracing: a consumer’s graphics revolution?

- DirectX Raytracing (DXR) from Microsoft is the first step toward a consumer’s graphics revolution.
Ray tracing is the future, and it always will be!
Obrigado!