Reconstructing the Indirect Light Field for Global Illumination

Jaakko Lehtinen¹  Timo Aila¹  Samuli Laine¹  Frédo Durand²
¹ NVIDIA Research ² MIT CSAIL
Indirect Illumination is Important
Rendering

For each visible point, look at the incident lighting from the hemisphere, multiply with BRDF, integrate over directions.
Light Paths

How do you get the incident lighting?
Monte Carlo Path Tracing

sample light paths stochastically
do this lots of times for each pixel
Light Paths

How do you get the incident lighting?
- Direct
- 1 bounce
Light Paths

How do you get the incident lighting?

- Direct
- 1 bounce
- 2 bounces
Light Paths

How do you get the incident lighting?

- Direct
- 1 bounce
- 2 bounces
- 3 bounces
- ...

Infinitely complex
Monte Carlo Path Tracing

calculate light paths stochastically

do this lots of times for each pixel
Monte Carlo Path Tracing

sample light paths stochastically
do this lots of times for each pixel
Monte Carlo Path Tracing

sample light paths stochastically

do this lots of times for each pixel
Monte Carlo Path Tracing

sample light paths stochastically
do this lots of times for each pixel
Our goal
Related work

image space + auxiliary info (depths, normals, etc.)
- [McCool99], [Meyer06]
- Cross-bilateral & variants [Segovia06, Dammertz09, Sen12]
- [Ward88, Kontkanen04]

reusing paths with scene geometry
- [Bala99, Bekaert02]

point-based methods that use actual geometry for visibility
- Photon mapping, Lightcuts [Walter05]

point-based global illumination [Christensen08]
given the samples taken by a path tracer, let’s see how well we can exploit them across pixels

no adaptive sampling
no knowledge of geometry
Observation

taken over all pixels, the path tracer fires a large set of ray segments in the scene.

lots of information on light flow and visibility

let’s exploit it
for each visible point, reconstruct the hemisphere from the sparse samples
for each visible point, reconstruct the hemisphere from the sparse samples
each sample is a ray segment with a start and end point, radiance, normals, etc.
Our Approach

1. first render quickly at low sampling rate
   - sample and store incident lighting to visible points

2. upsample incident lighting
   - reconstruct incident radiance hemisphere from samples
   - apply BRDF
Key Challenges / Contributions

how to reconstruct radiance from just ray segments?

if we have no geometry, how to determine visibility?
No direct

use known techniques
Coherence
what about radiance incident to a nearby point?
if visibility and gloss are accounted for, we can reuse the radiance. It just comes from a slightly different direction due to the geometry between and.
Light Field

parameterize rays by two planes [Levoy96]
Light Field

parameterize rays by two planes [Levoy96]
Light Field

parameterize rays by two planes [Levoy96]
Light Field

parameterize rays by two planes [Levoy96]
Light Field

parameterize rays by two planes [Levoy96]
Light Field

changing ● while leaving

unchanged
Light Field

changing • while leaving

unchanged
Anisotropy

rate of change of radiance along depends on the glossiness of [Durand05]
Anisotropy

in particular, for diffuse the radiance is constant along
Reconstruction

known slope allows reasoning about radiance for all rays on \( \text{by looking along it} \)
Reconstruction

previously: motion blur, depth of field, shadows

− [Hachisuka08, Egan09, Egan11, Lehtinen11]
Reconstruction

query ray
Reconstruction

take all input samples
Reconstruction

take all input samples
Reconstruction

take all input samples
Reconstruction

take all input samples

reproject to query ray's x coordinate
Reconstruction

take all input samples
reproject to query ray's x coordinate
determine visibility***
Reconstruction

take all input samples

reproject to query ray’s x coordinate

determine visibility***

apply filter to remaining samples
Primal Domain
to reconstruct radiance for a query ray
Primal Domain

to reconstruct radiance for a query ray

reproject all input samples,
Primal Domain

to reconstruct radiance for a query ray

reproject all input samples,
Primal Domain

to reconstruct radiance for a query ray

reproject all input samples,
Primal Domain

to reconstruct radiance for a query ray

reproject all input samples,
determine visibility,
filter radiance***
Filtering

diffuse is easy
  - just average the radiance

glossy is more challenging
  - weighting and non-uniform sampling density cause "mass shifting" that results in creeping highlights
  - solution: pick nearest neighbor
Problem #1: Gloss

diffuse is easy
Problem #1: Gloss

diffuse is easy
Problem #1: Gloss

what if \( \text{●} \) lies on a glossy surface?
Problem #1: Gloss

what if • lies on a glossy surface?
Problem #1: Gloss

what if \( \bullet \) lies on a glossy surface?

solution: we store the angular BRDF bandwidth of the sender along with the sample
  - von Mises-Fisher distribution
  - used as weight
Problem #2: Visibility
Problem #2: Visibility
Problem #2: Visibility

occluder
Problem #2: Visibility
Problem #2: Visibility

solution:
we build a point-based model out of the input segment endpoints
Point-based model

we promote the starting points of the ray segments to splats
How to Size?

we use $k$ nearest neighbors to determine the sizes (with fixed $k$)
Point-based model

we use $k$ nearest neighbors to determine the sizes (with fixed $k$)

the splats are all that remain of the geometry
Ray Tracing

we build a BVH from the splats

– streaming algorithm of Kontkanen et al. [2011]

simple to walk the tree and gather all splats that intersect a query ray
Ray Tracing

we build a BVH from the splats – streaming algorithm of Kontkanen et al. [2011]

simple to walk the tree and gather all splats that intersect a query ray
Ray Tracing

this implements the anisotropic lookup in the light field by following the slopes

—doesn’t account for gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss

everything is fine
if surface is diffuse
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss
Visibility and Gloss

solution: account for angular bandwidth in k-NN search

splat sizes adapt to angular distribution
All Done?
All Done?

clearly, replacing the scene with splats changes occlusion
Occlusion Trouble

- Ground truth
- Just splats
Occlusion Overestimation
Occlusion Overestimation
Occlusion Overestimation
Occlusion Overestimation
Occlusion Overestimation
Occlusion Overestimation

key to solution: we have all input rays, and we know this one did not terminate here!
Solution: Shrinking

key to solution: we have all input rays, and we **know** this one did not terminate here!
Solution: Shrinking

key to solution: we have all input rays, and we know this one did not terminate here!

loop over all input rays and shrink splats that violate known visibility
Occlusion Trouble No More

ground truth

after shrink
Occlusion Trouble No More

no shrink

after shrink
Diffuse Results, 1 sample/pixel

PBRT (2048 spp)  Input (1 spp)  Our result
Diffuse Results, 1 sample/pixel

PBRT (2048 spp)  Input (1 spp)  Our result
Temporal Coherence

every frame sampled and reconstructed independently
Results, Glossy

Reference (8192 spp)

Input (8 spp)

Our result (8 spp, 512 query rays/pixel)
Results, Glossy

PBRT (8192 spp)  Input (8 spp)  Our result (8 spp, 512 query rays/pixel)
Ambient Occlusion

input, 4spp

our result
Discussion, Performance

weak dependence on scene structure
- San Miguel (10Mtri) only 29% slower than Cornell box (34 tri)

diffuse and AO are fast
- 20x perf increase on GPU for San Miguel

glossy works, but is slow
Conclusion

we’ve sped up global illumination by reconstructing an up-sampled the indirect light field from sparse input
sparse path segments are surprisingly rich in information
Thank You

sparse path segments are surprisingly rich in information

code will be available

acknowledgments / thanks

- Guillermo M. Leal Llaguno for the San Miguel scene
- National Science Foundation
- NVIDIA and MIT colleagues, anonymous reviewers