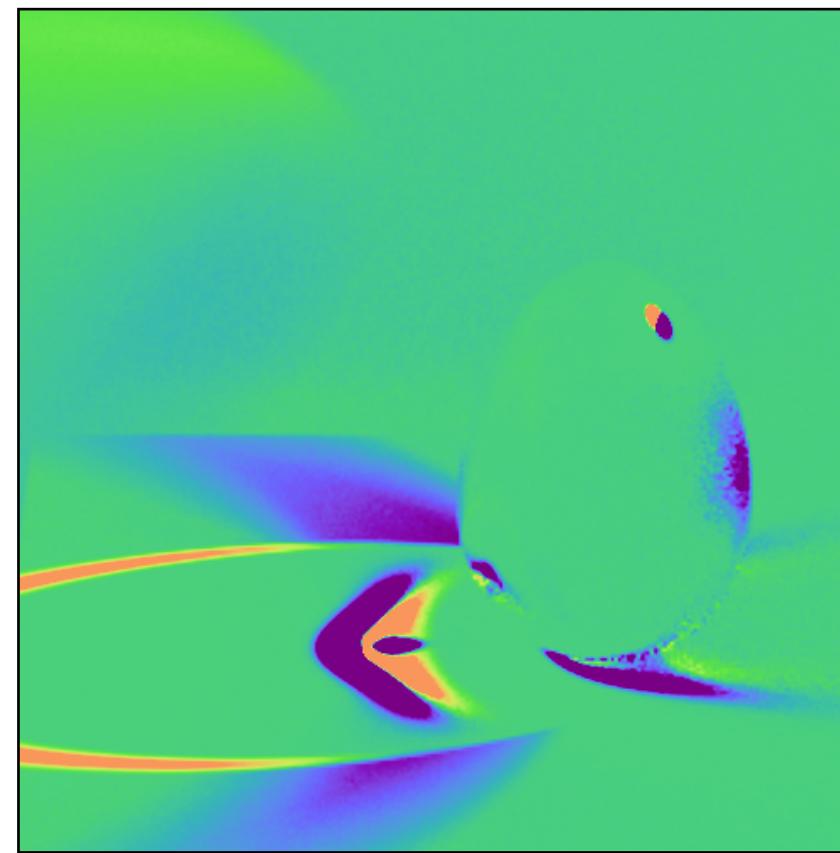
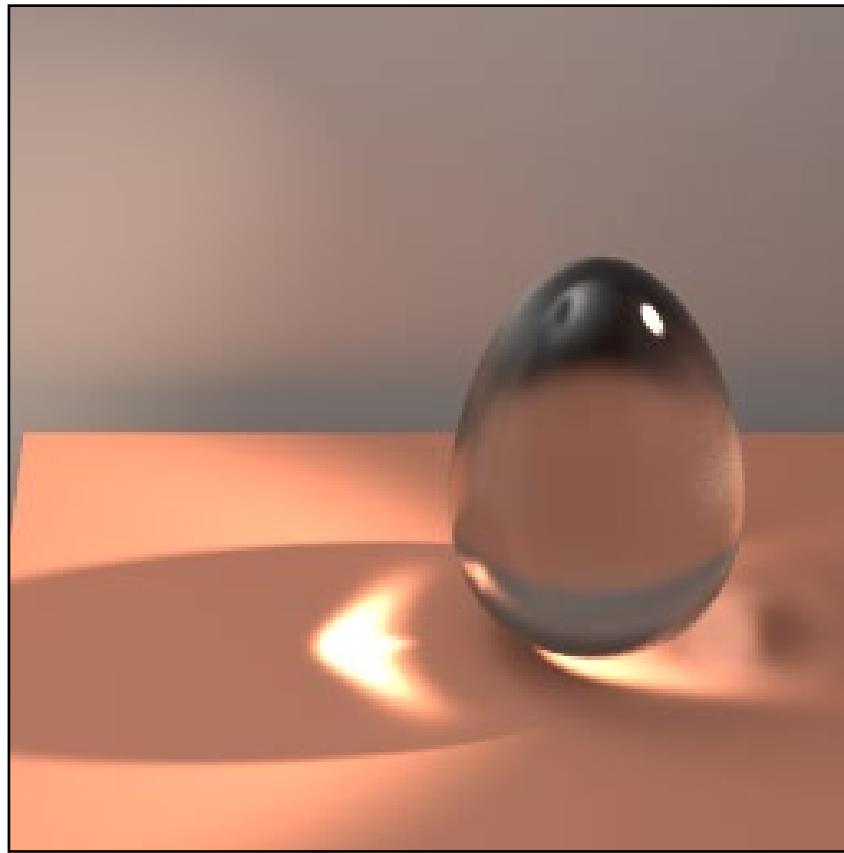


Inverse and differentiable rendering



15-468, 15-668, 15-868
Physics-based Rendering
Spring 2022, Lecture 16

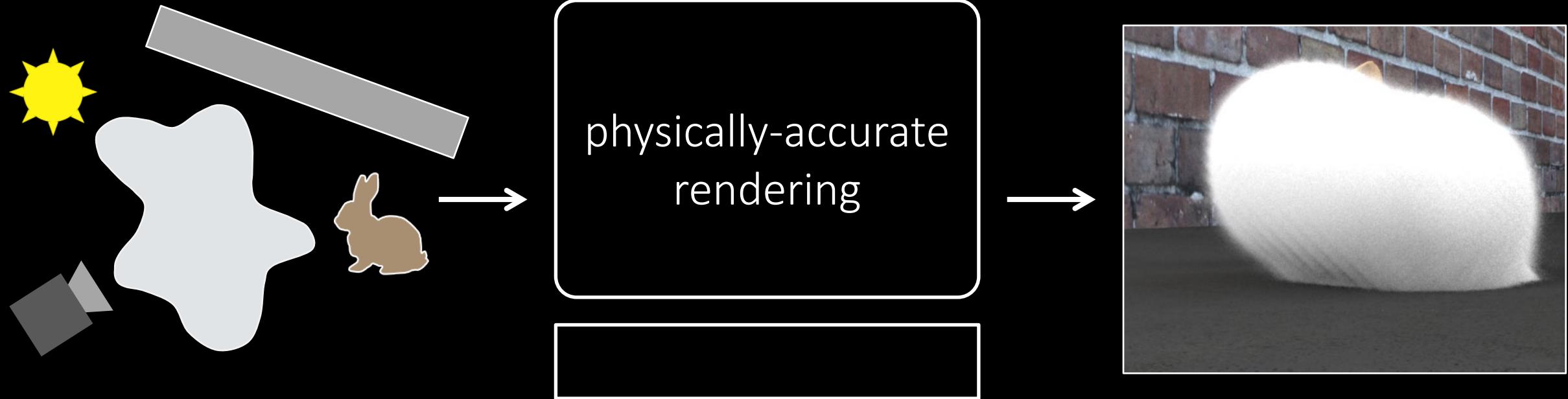
Course announcements

- Programming assignment 4 posted, due Friday 4/1 at 23:59.
 - How many of you have looked at/started/finished it?
 - Any questions?
- Take-home quiz 9 will be posted tonight.
- Talk by Alexandr Kuznetsov in graphics lab meeting.

Overview of today's lecture

- Inverse rendering.
- Differentiable rendering.
- Differentiating local parameters.
- Differentiating global parameters.
- Path-space differentiable rendering.
- Reparameterizations.

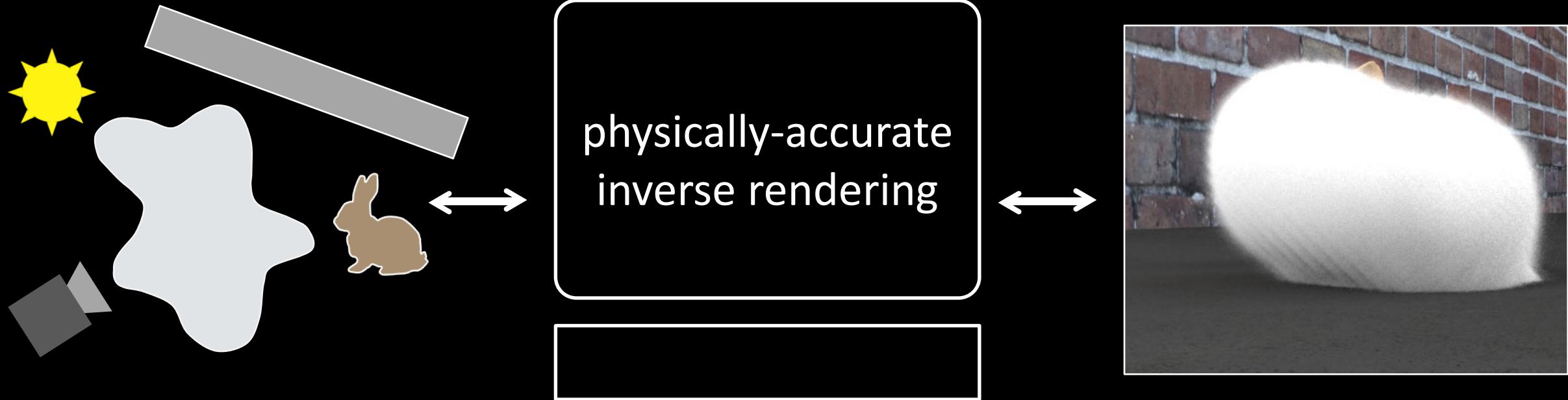
Forward rendering



digital scene specification
(geometry, materials,
optics, light sources)

photorealistic
simulated image

Inverse rendering



digital scene specification
(geometry, materials,
camera, light sources)

photographic
synthetic image

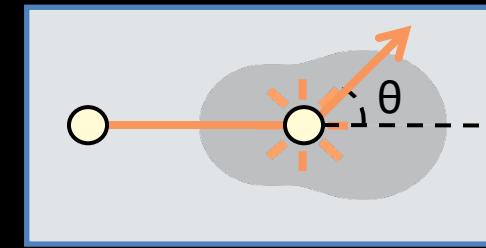
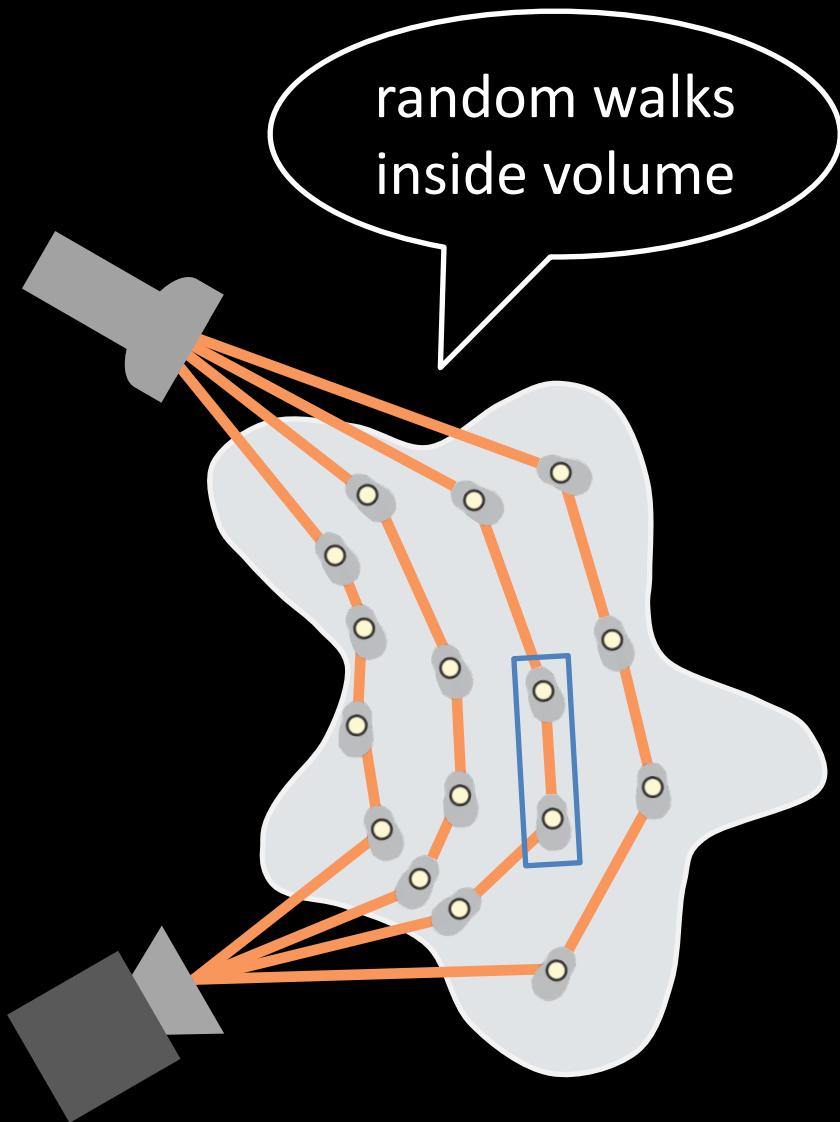
What I was doing in 2013



I wanted to make images such as this one

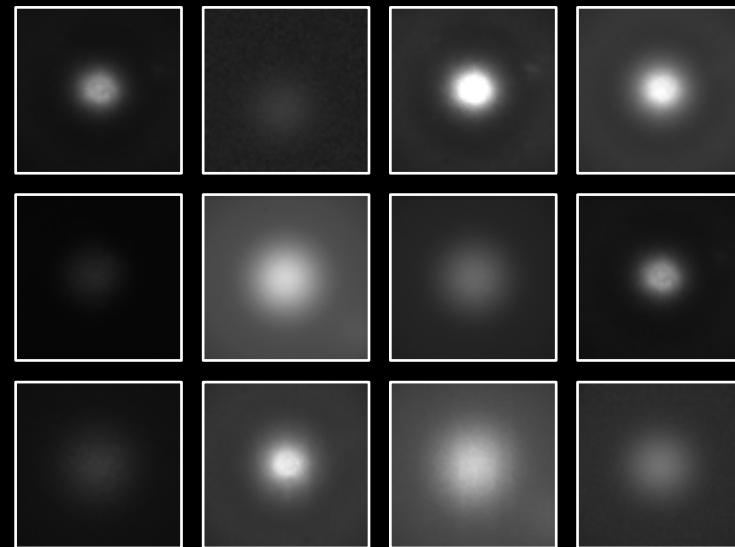
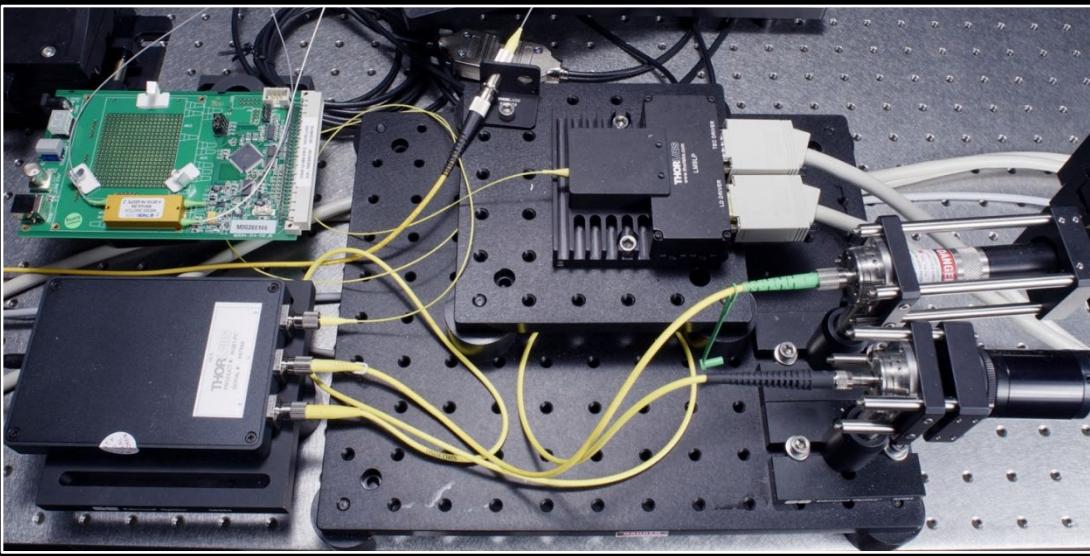
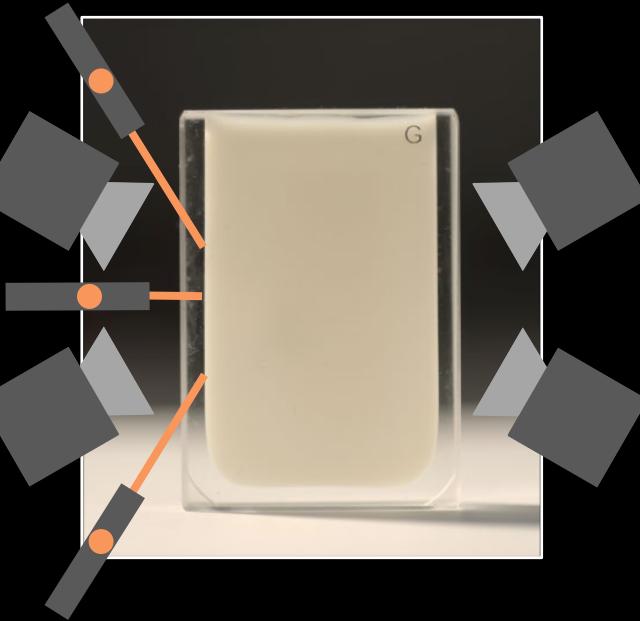
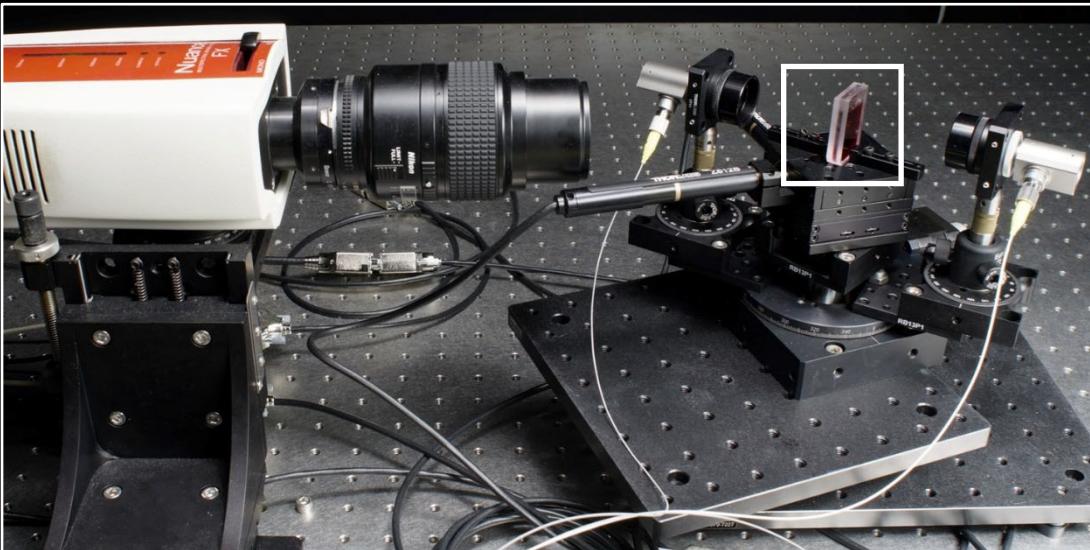


Scattering: extremely multi-path transport

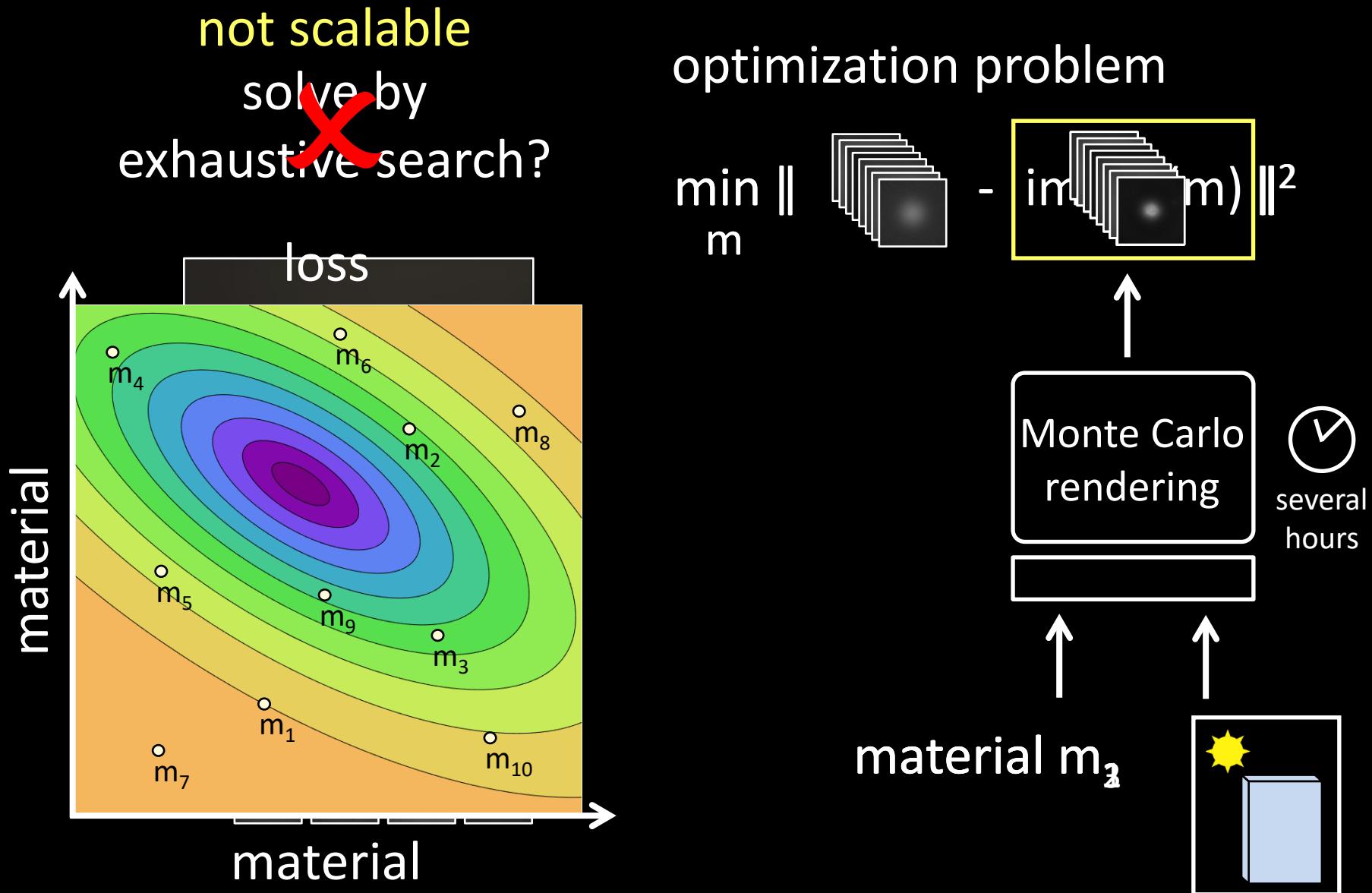


volumetric density (σ_t)
scattering albedo (a)
phase function (f_r)

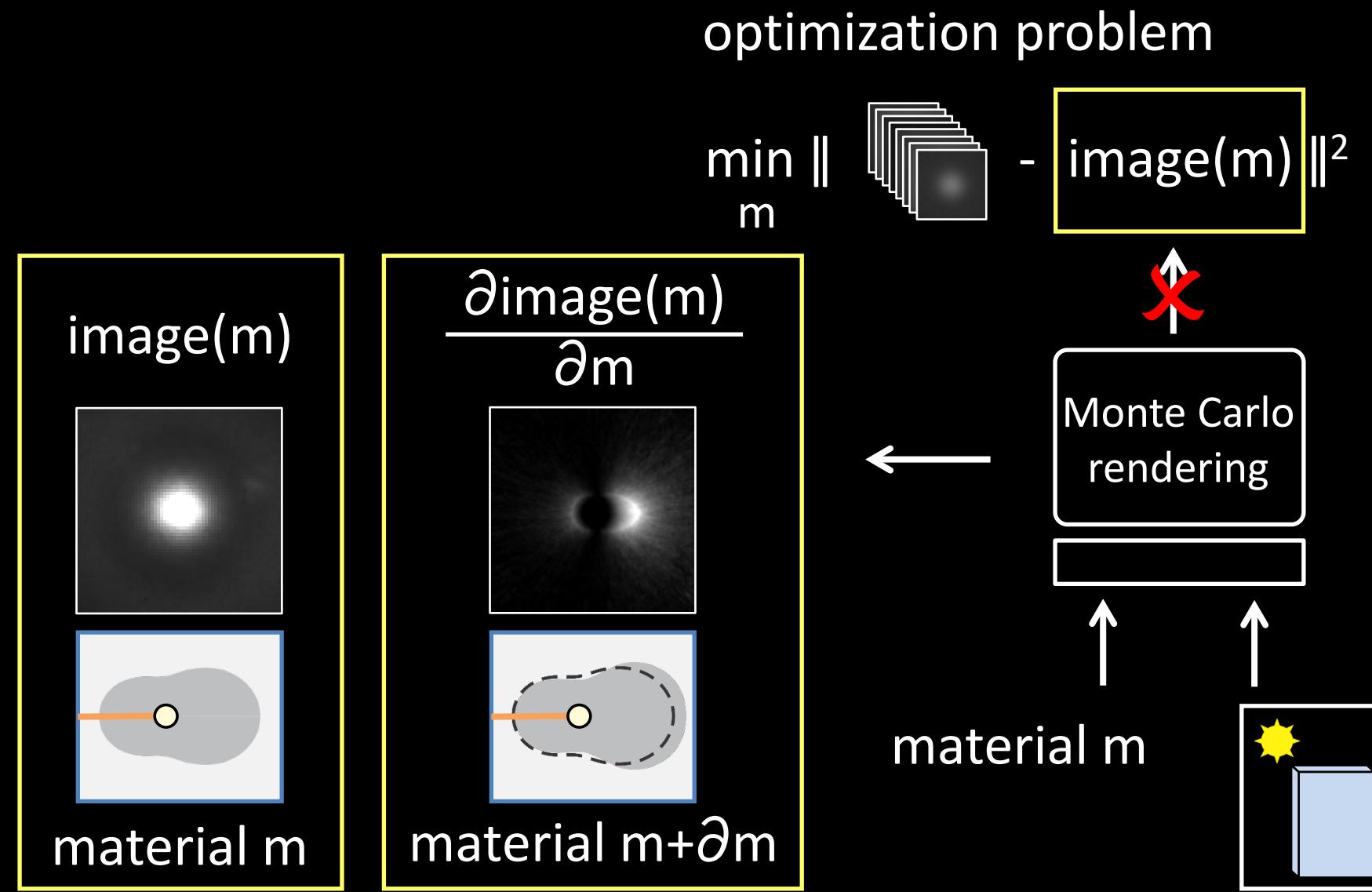
Acquisition setup



Analysis by synthesis (a.k.a. inverse rendering)



Analysis by synthesis (a.k.a. inverse rendering)



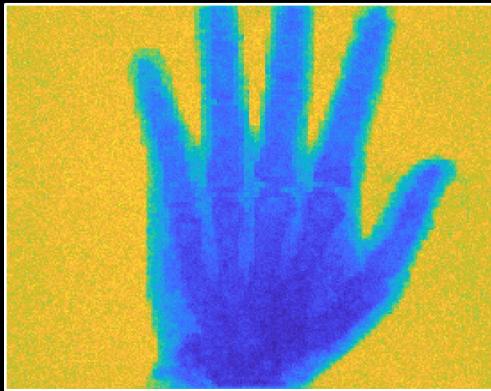
Other scattering materials



everyday materials
[Gkioulekas et al. 2013]



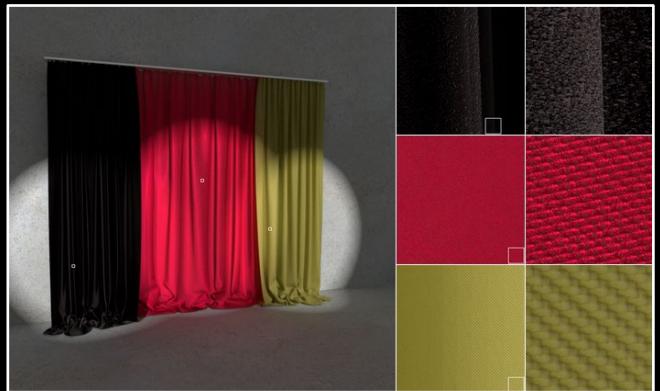
industrial dispersions
[Gkioulekas et al. 2013]



computed tomography
[Geva et al. 2018]



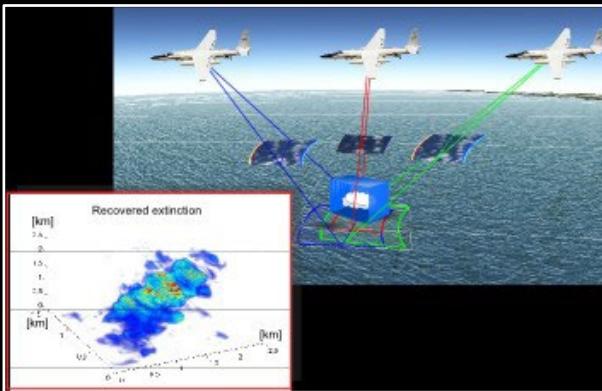
optical
tomography
[Gkioulekas et al.
2016]



woven fabrics
[Khungurn et al. 2015,
Zhao et al. 2016]

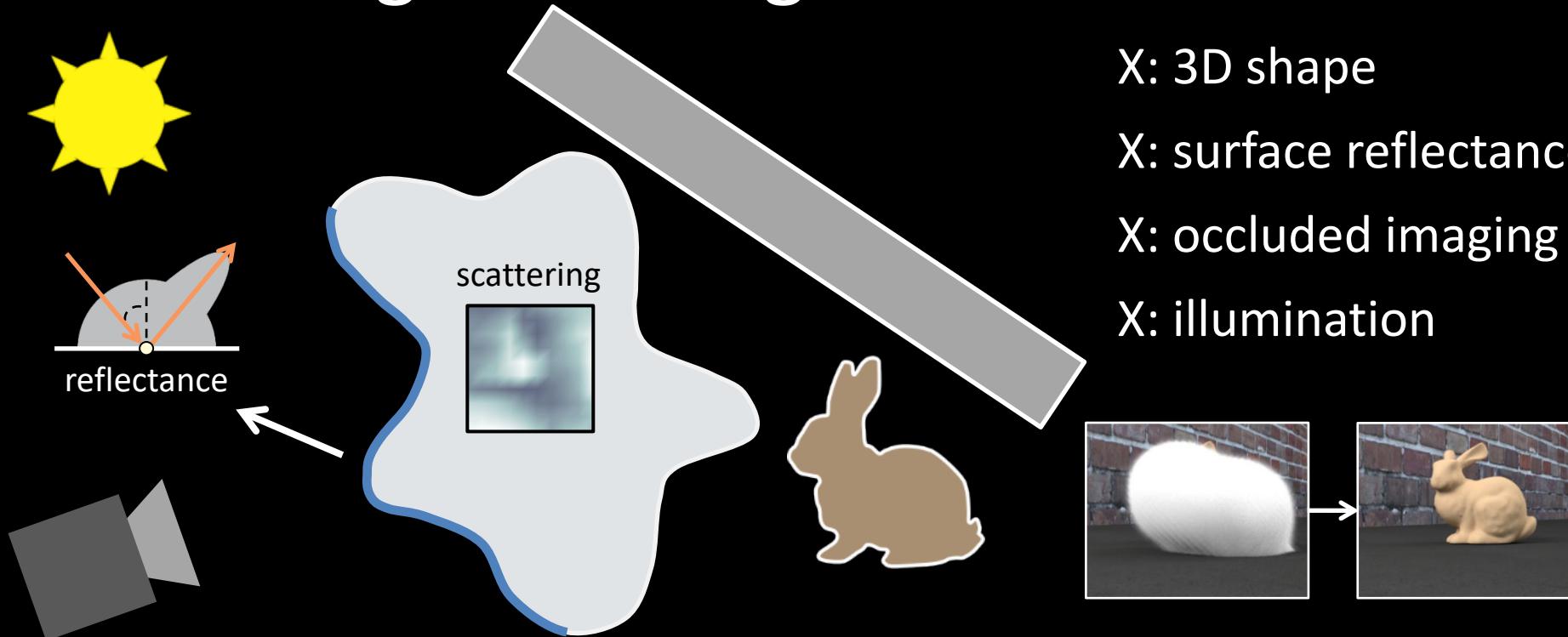


3D printing
[Elek et al. 2017, 2019]



clouds
[Levis et al. 2015, 2017]

Making sense of global illumination



X: 3D shape
X: surface reflectance
X: occluded imaging
X: illumination

analysis by synthesis

$$\min_X \| \text{image}(X) - \text{target image} \|^2$$

stochastic gradient descent

while (not converged)
update X with

$$\frac{\partial \text{loss}(X)}{\partial X}$$

Monte-Carlo rendering

differentiable rendering: image gradients with respect to arbitrary X

Differentiable rendering

Not related to:

Gradient-Domain Path Tracing

[Markus Kettunen¹](#) [Marco Manzi²](#) [Miika Aittala¹](#) [Jaakko Lehtinen^{1,3}](#) [Frédo Durand⁴](#) [Matthias Zwicker²](#)

¹[Aalto University](#)

²[University of Bern](#)

³[NVIDIA](#)

⁴[MIT CSAIL](#)

[ACM Transactions on Graphics 34\(4\) \(Proc. SIGGRAPH 2015\).](#)



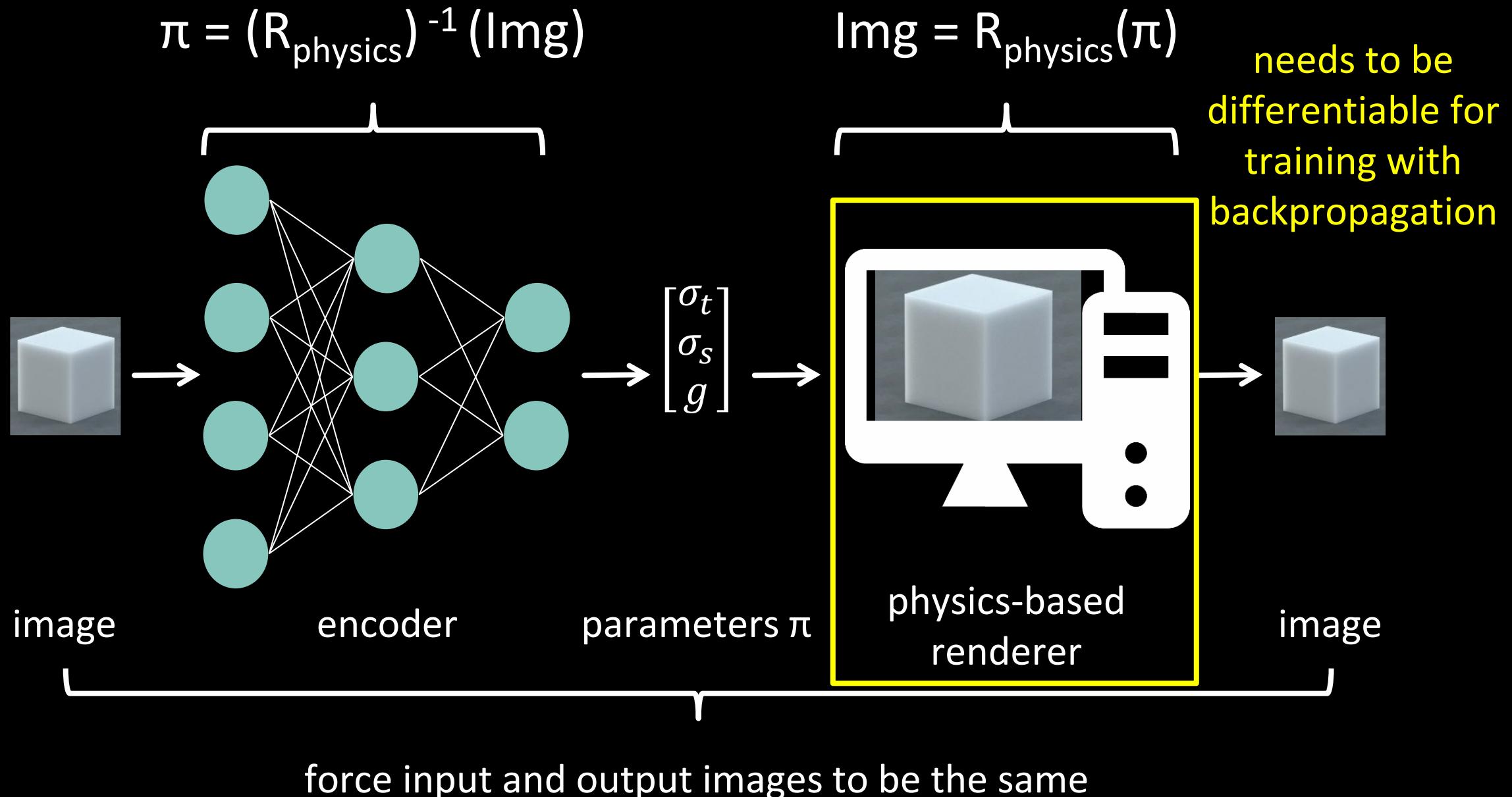
SIGGRAPH Asia 2018 Courses

Light Transport Simulation in the Gradient Domain



“Gradient” in their case refers to image edges.

Differentiable rendering and deep learning



Quick reminder from calculus

Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = ?$$

Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = \int_a^b \frac{\partial}{\partial \pi} f(x; \pi) dx$$

what is this rule called?

Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = \int_a^b \frac{\partial}{\partial \pi} f(x; \pi) dx \quad \text{differentiation under the integral sign}$$

$$\frac{\partial}{\partial \pi} \int_{a(\pi)}^{b(\pi)} f(x; \pi) dx = ?$$

Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = \int_a^b \frac{\partial}{\partial \pi} f(x; \pi) dx \quad \text{differentiation under the integral sign}$$

$$\begin{aligned} \frac{\partial}{\partial \pi} \int_{a(\pi)}^{b(\pi)} f(x; \pi) dx &= \int_{a(\pi)}^{b(\pi)} \frac{\partial}{\partial \pi} f(x; \pi) dx \\ &\quad + f(b(\pi); \pi) \frac{\partial b(\pi)}{\partial \pi} - f(a(\pi); \pi) \frac{\partial a(\pi)}{\partial \pi} \end{aligned} \quad \text{what is this rule called?}$$

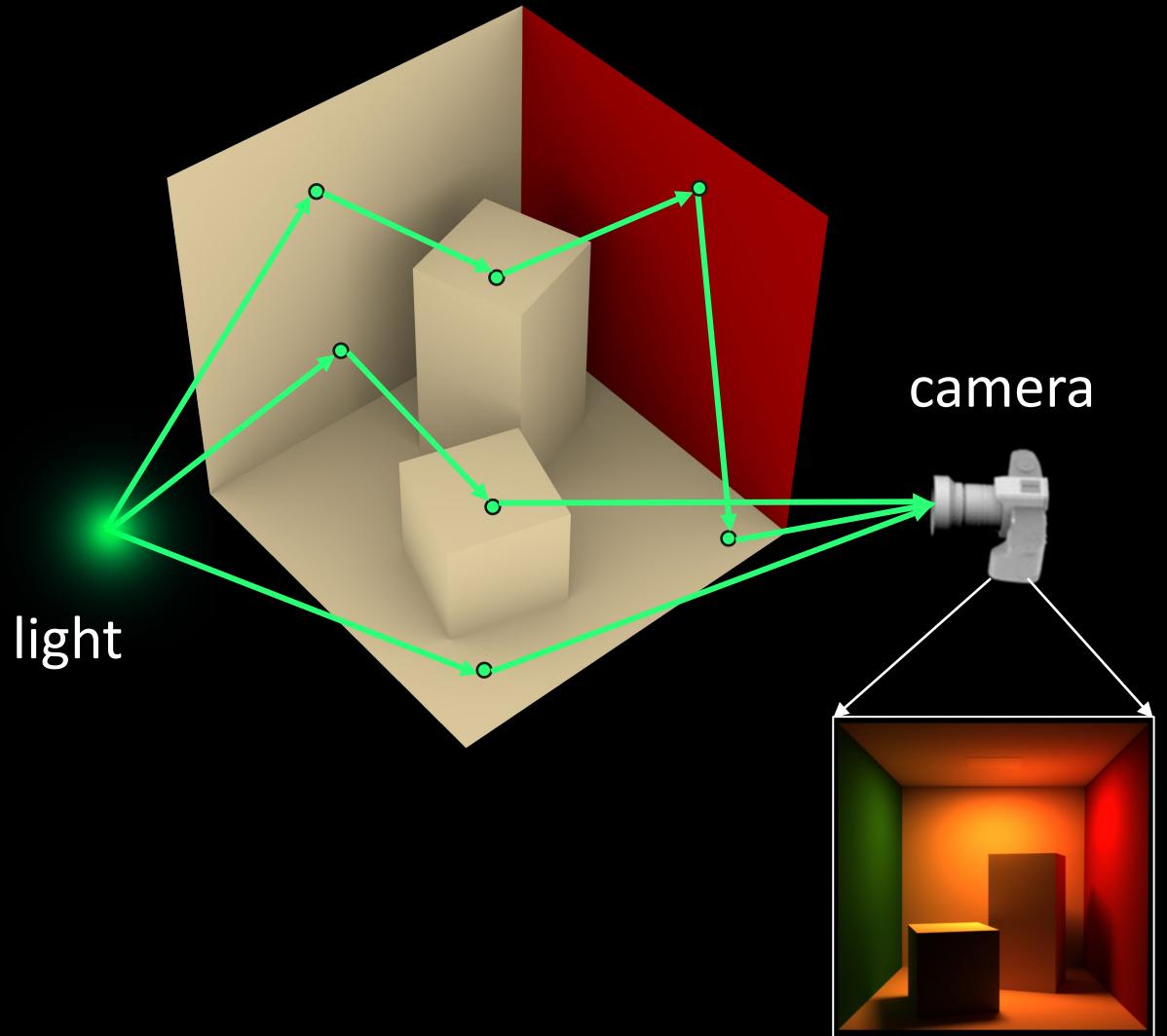
Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = \int_a^b \frac{\partial}{\partial \pi} f(x; \pi) dx \quad \text{differentiation under the integral sign}$$

$$\begin{aligned} \frac{\partial}{\partial \pi} \int_{a(\pi)}^{b(\pi)} f(x; \pi) dx &= \int_{a(\pi)}^{b(\pi)} \frac{\partial}{\partial \pi} f(x; \pi) dx && \text{Leibniz integral rule} \\ &+ f(b(\pi); \pi) \frac{\partial b(\pi)}{\partial \pi} - f(a(\pi); \pi) \frac{\partial a(\pi)}{\partial \pi} \end{aligned}$$

Trivial differentiable rendering

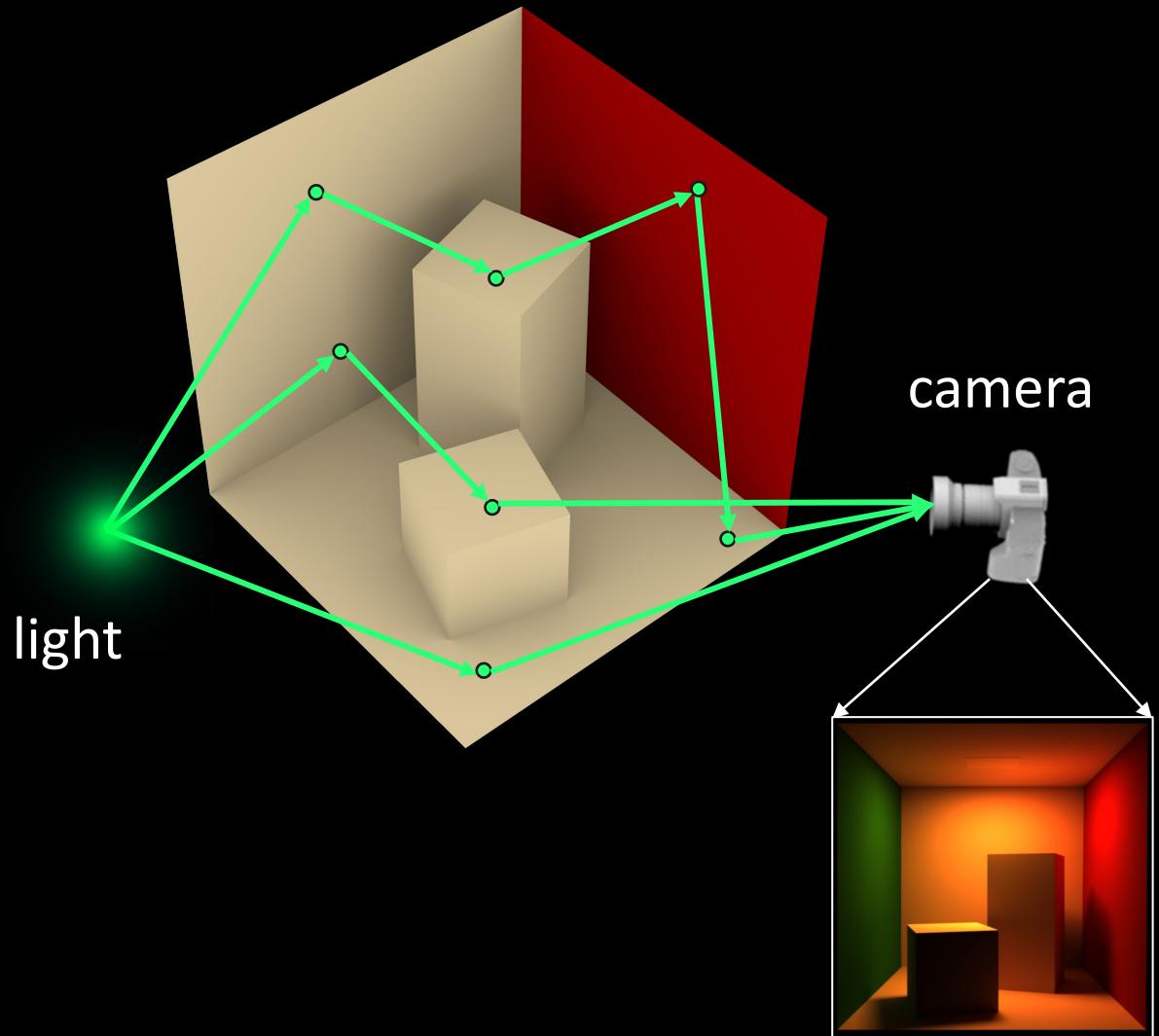
Images as path integrals



$$I(\pi) = \int_{\mathbb{P}} f(\bar{\mathbf{x}}; \pi) d\bar{\mathbf{x}}$$

- $\bar{\mathbf{x}}$ → Light path, set of ordered vertices on surfaces
- \mathbb{P} → Space of valid paths
- $f(\bar{\mathbf{x}})$ → Path contribution,
includes geometric terms (visibility, fall-off) &
local terms (BRDF, foreshortening, emission)

Monte Carlo rendering: approximating path integrals



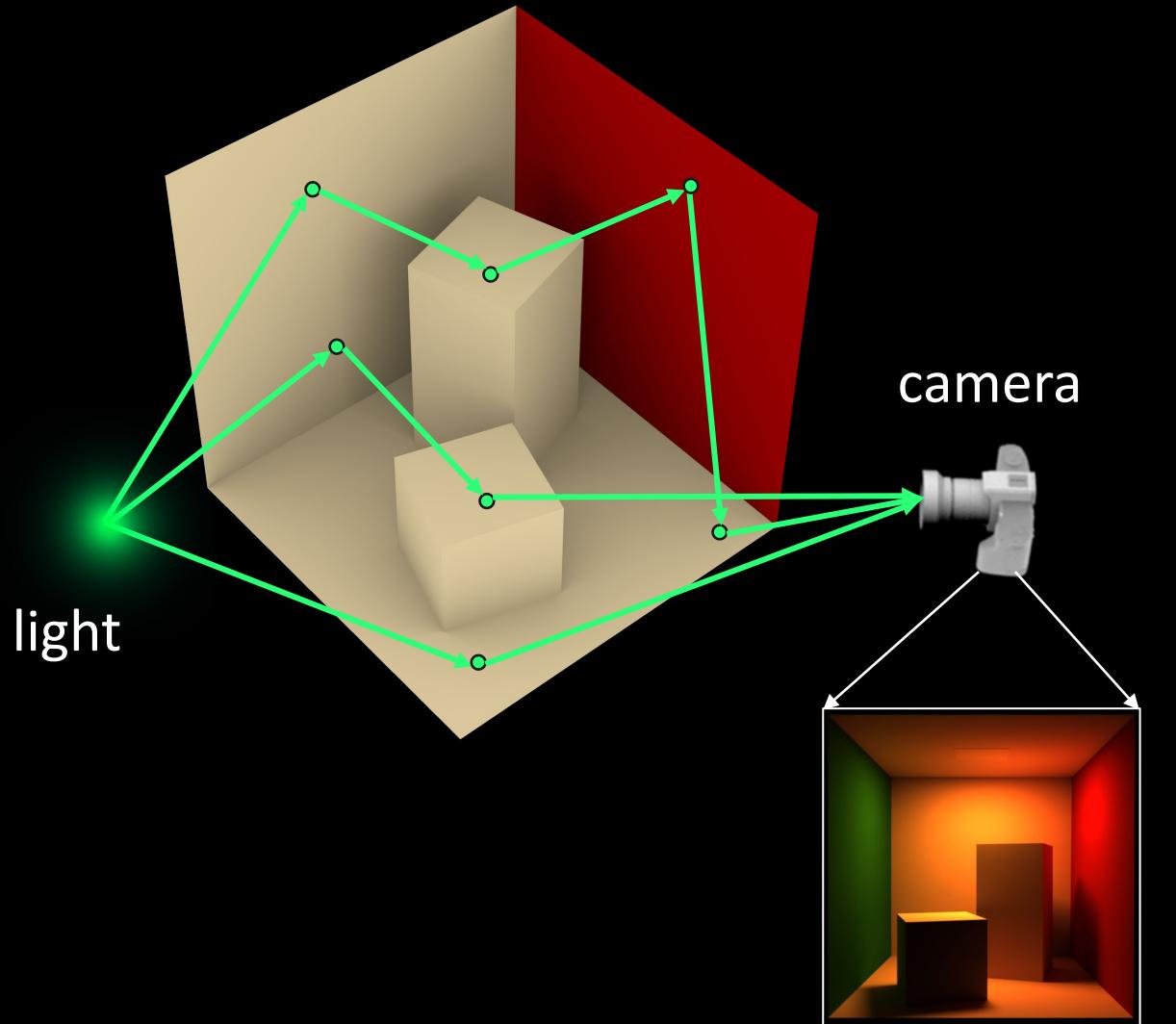
$$I(\pi) \approx \underbrace{\sum_{i=1}^N \frac{f(\bar{\mathbf{x}}_i; \pi)}{p(\bar{\mathbf{x}}_i)}}_{MC(\pi)}$$

$\bar{\mathbf{x}}_i$ → Randomly sampled light paths

$p(\bar{\mathbf{x}}_i)$ → Probability of sampling a path

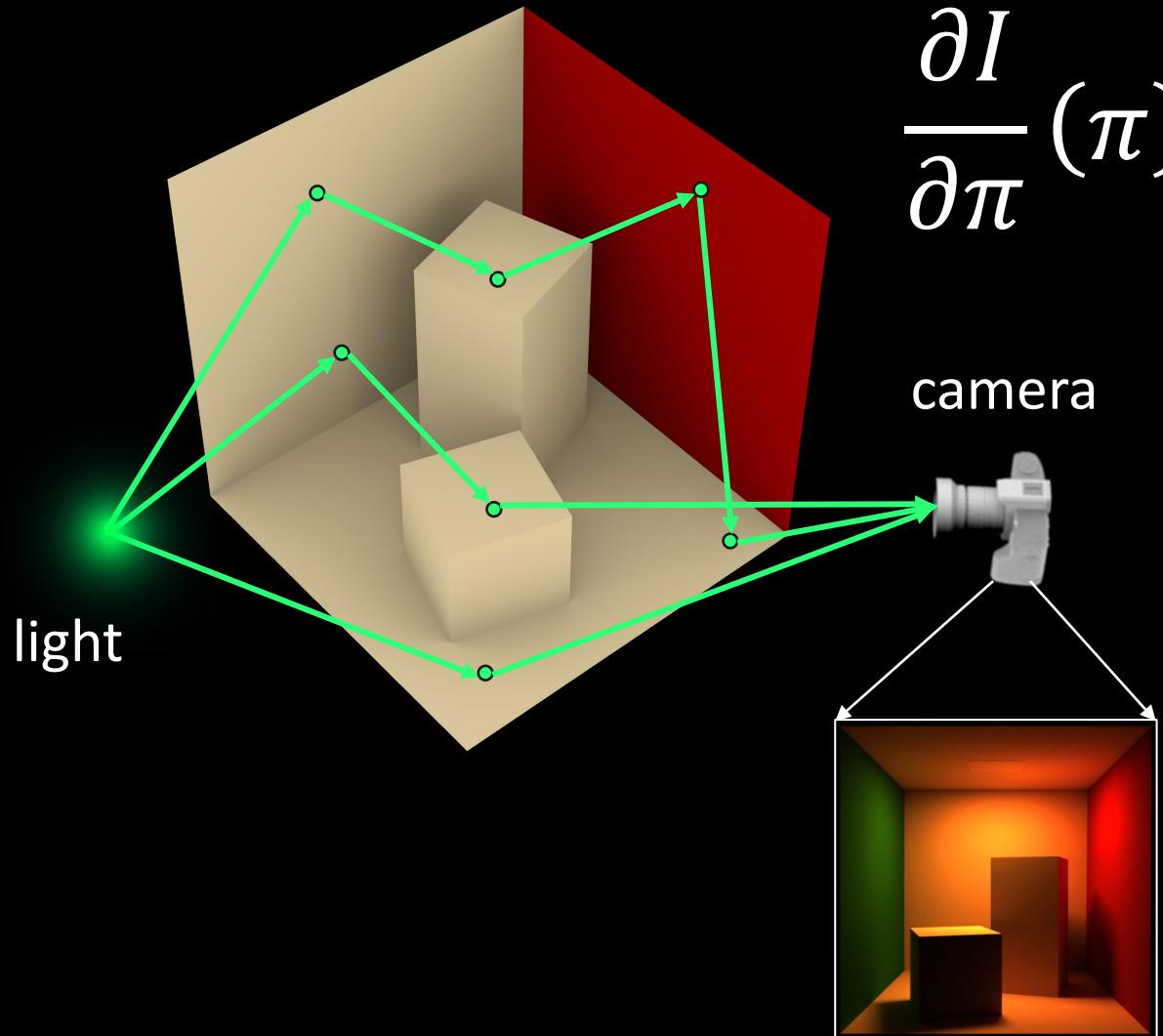
Algorithms such as path tracing, bidirectional path tracing, etc. sample paths.

How can we approximate the derivative of the image?



$$\frac{\partial I}{\partial \pi}(\pi) \approx ?$$

Easy approach 1: finite differences

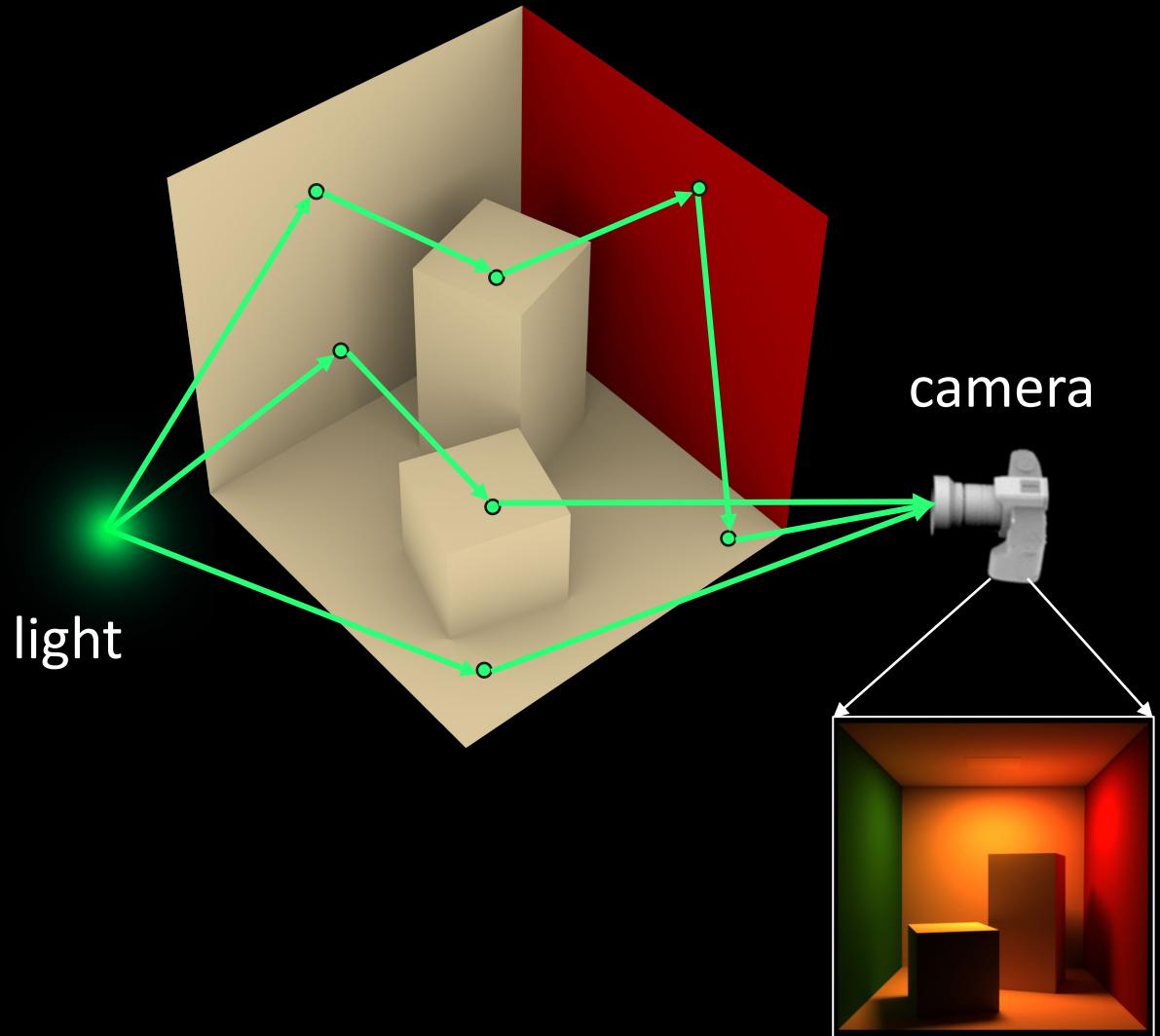


$$\frac{\partial I}{\partial \pi}(\pi) \approx \frac{MC(\pi + \varepsilon) - MC(\pi - \varepsilon)}{2\varepsilon}$$

Any issues with this?

- Incredibly noisy for small ε
- Very inaccurate for large ε
- Techniques for noise reduction exist, but generally impractical approach

Easy approach 2: automatic differentiation



$$\frac{\partial I}{\partial \pi}(\pi) \approx \text{autodiff}(MC(\pi))$$

Any issues with this?

- Many path sampling techniques are not differentiable
- High variance (consider $f(x;\pi) = \text{constant}$)
- Rendering produces enormous, non-local computational graphs.

OpenDR: An Approximate Differentiable Renderer

[Loper and Black 2015]

- Only direct illumination.
- Only shading parameters (normals, reflectance).

Abstract. Inverse graphics attempts to take sensor data and infer 3D geometry, illumination, materials, and motions such that a graphics renderer could realistically reproduce the observed scene. Renderers, however, are designed to solve the forward process of image synthesis. To go in the other direction, we propose an approximate *differentiable renderer (DR)* that explicitly models the relationship between changes in model parameters and image observations. We describe a publicly available *OpenDR* framework that makes it easy to express a forward graphics model and then automatically obtain derivatives with respect to the model parameters and to optimize over them. Built on a new auto-differentiation package and OpenGL, OpenDR provides a local optimization method that can be incorporated into probabilistic programming frameworks. We demonstrate the power and simplicity of programming with OpenDR by using it to solve the problem of estimating human body shape from Kinect depth and RGB data.

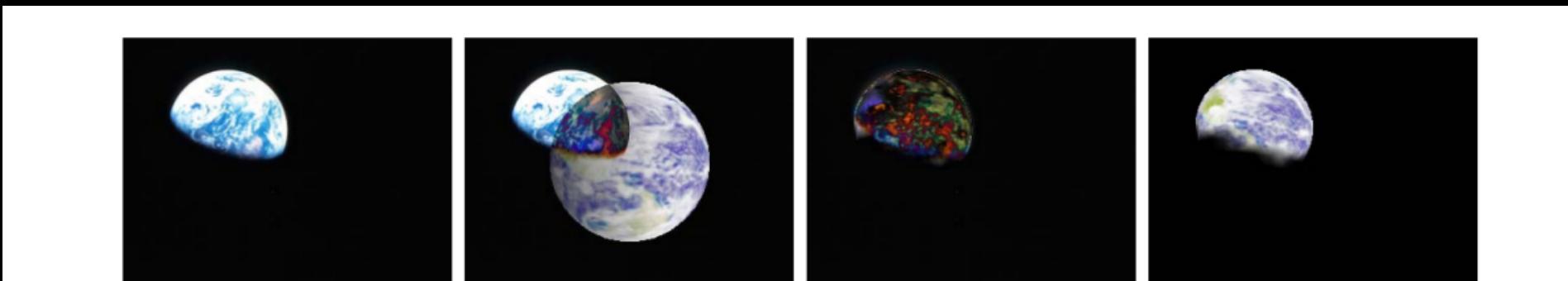
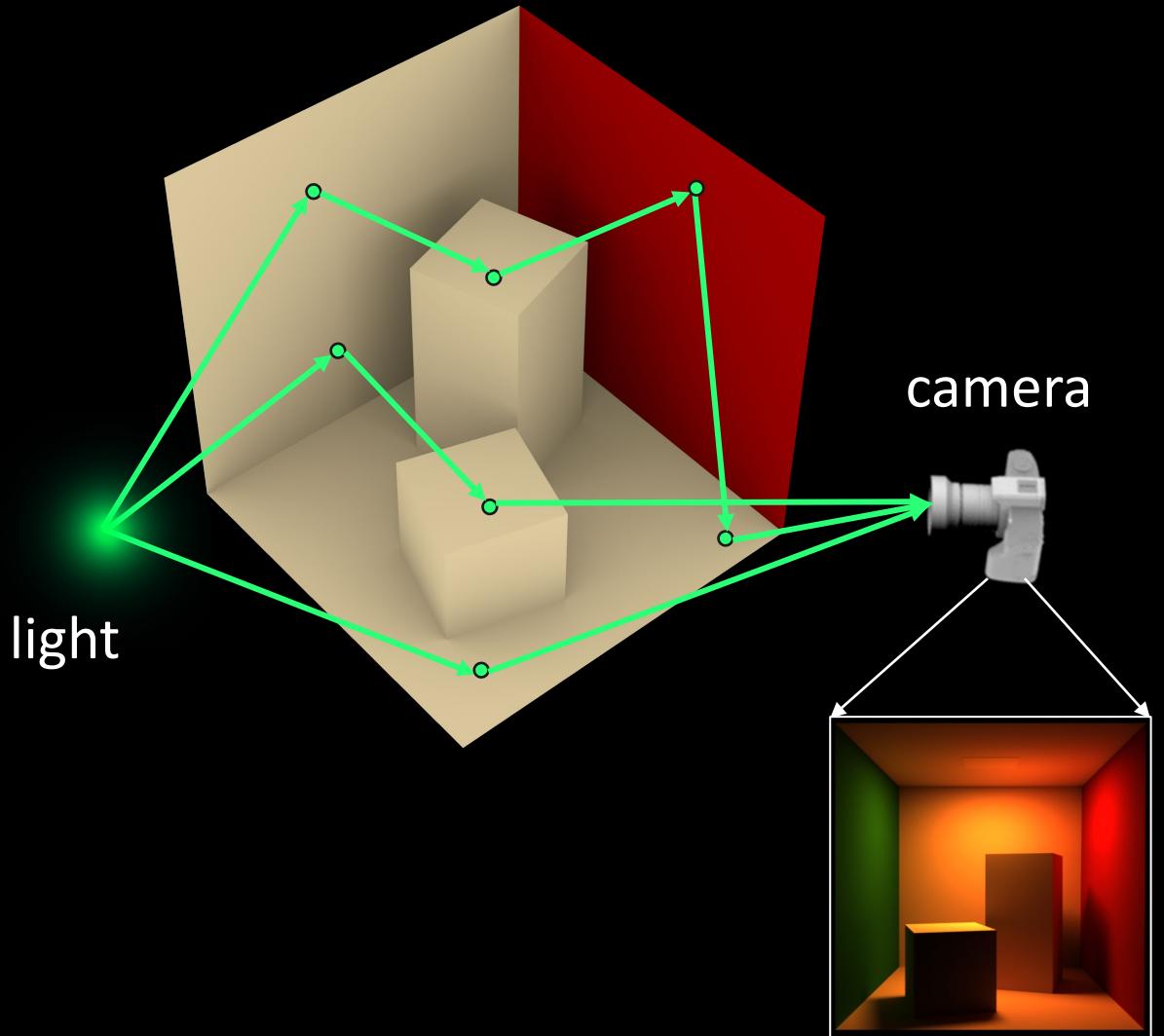


Fig. 4. Illustration of optimization in Figure 3. In order: observed image of earth, initial absolute difference between the rendered and observed image intensities, final difference, final result.

Differentiable rendering for local parameters

Images as path integrals

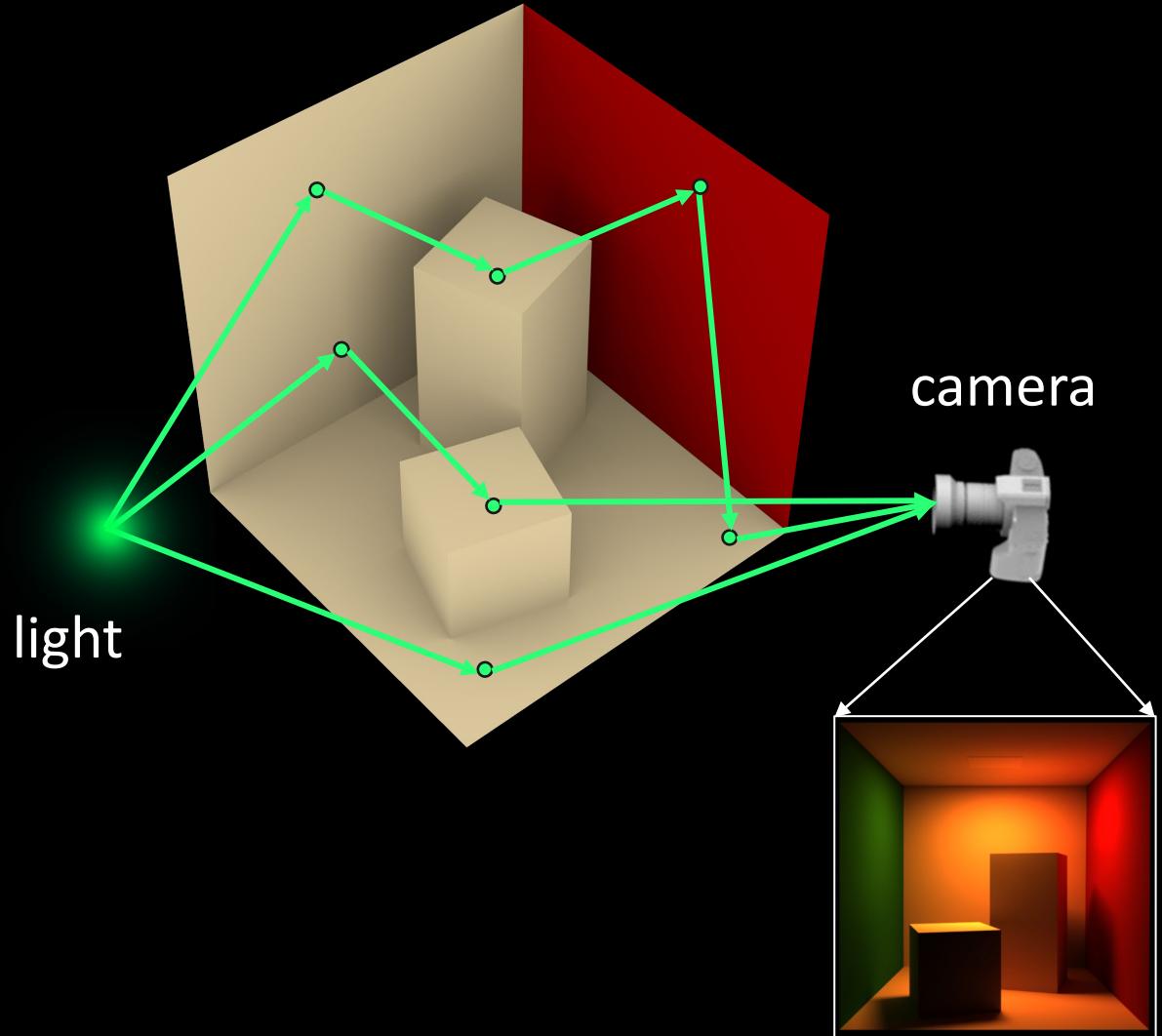


$$I(\pi) = \int_{\mathbb{P}} f(\bar{\mathbf{x}}; \pi) d\bar{\mathbf{x}}$$

- $\bar{\mathbf{x}}$ → Light path, set of ordered vertices on surfaces
- \mathbb{P} → Space of valid paths
- $f(\bar{\mathbf{x}})$ → Path contribution,
includes geometric terms (visibility, fall-off) &
local terms (BRDF, foreshortening, emission)

Assume \mathbb{P} is independent of π

Derivatives of images as path integrals

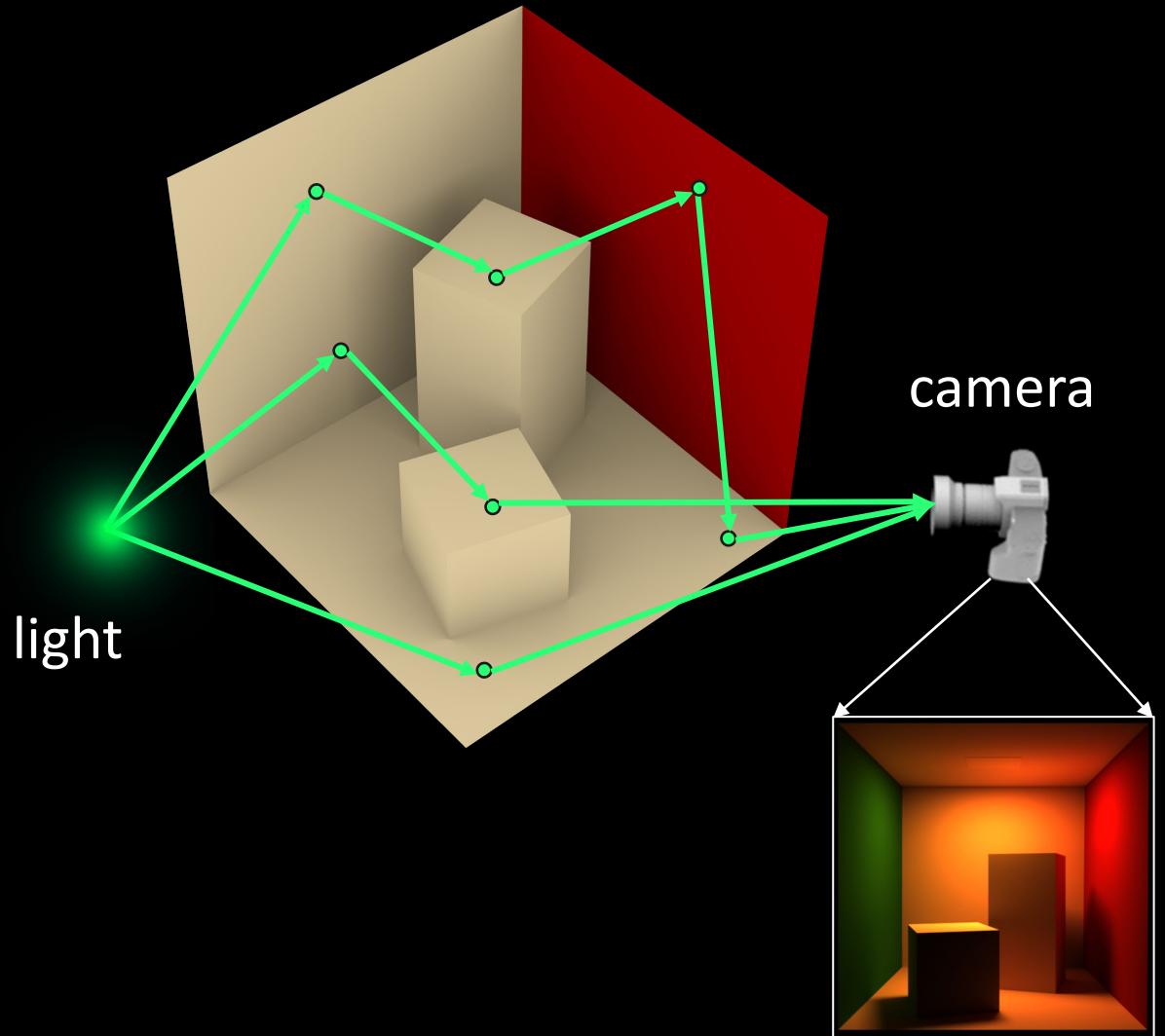


$$\frac{\partial I}{\partial \pi}(\pi) = ?$$

- \bar{x} → Light path, set of ordered vertices on surfaces
- \mathbb{P} → Space of valid paths
- $f(\bar{x})$ → Path contribution,
includes geometric terms (visibility, fall-off) &
local terms (BRDF, foreshortening, emission)

Assume \mathbb{P} is independent of π

Derivatives of images as path integrals



$$\frac{\partial I}{\partial \pi}(\pi) = \int_{\mathbb{P}} \frac{\partial f}{\partial \pi}(\bar{x}; \pi) d\bar{x}$$

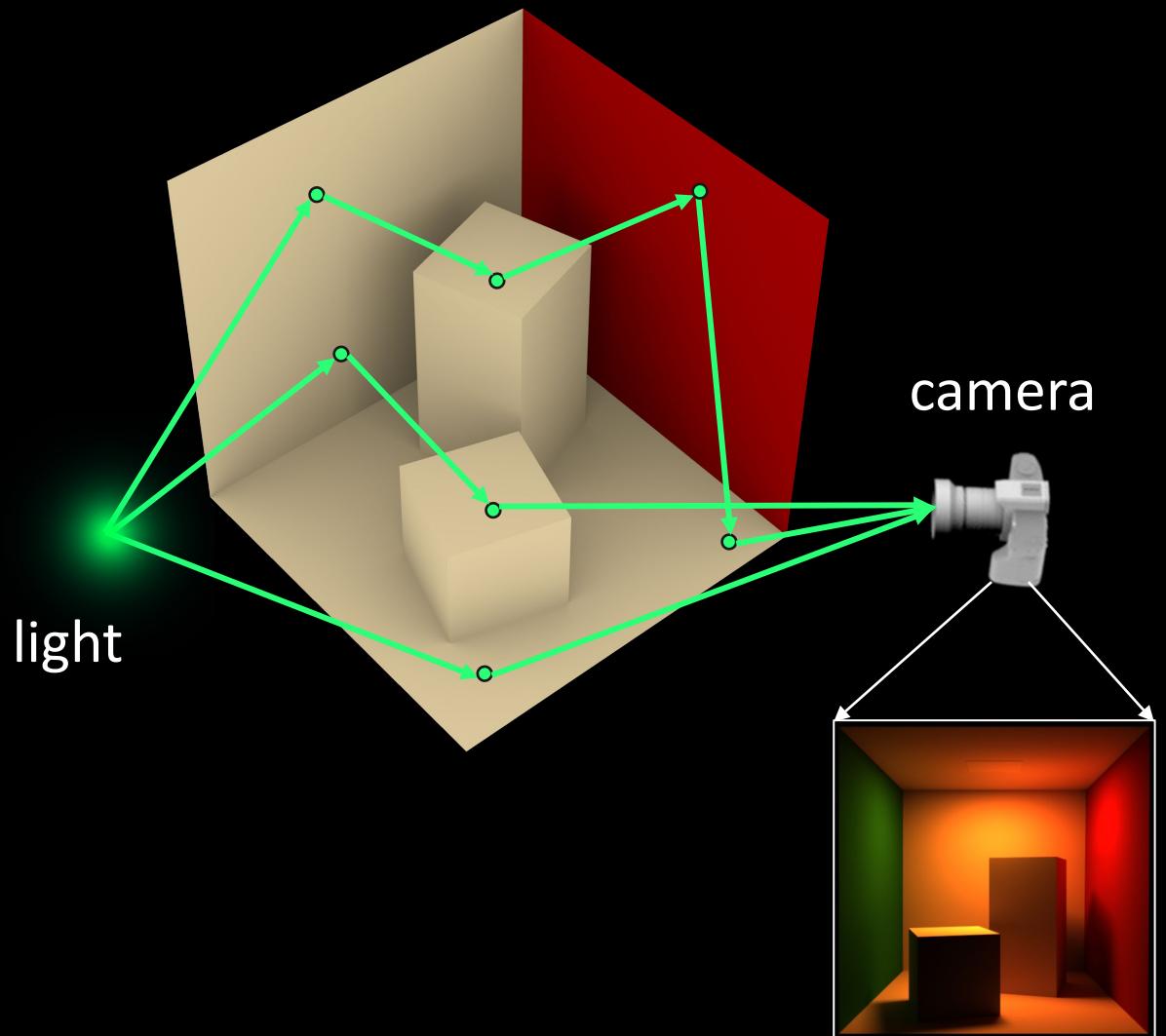
differentiation under the integral sign

- \bar{x} → Light path, set of ordered vertices on surfaces
- \mathbb{P} → Space of valid paths
- $f(\bar{x})$ → Path contribution,
includes geometric terms (visibility, fall-off) &
local terms (BRDF, foreshortening, emission)

Assume \mathbb{P} is independent of π

Monte Carlo differentiable rendering (for local parameters)

This term is generally easy to compute during path tracing



$$\frac{\partial I}{\partial \pi}(\pi) \approx \sum_{i=1}^N \frac{\frac{\partial f}{\partial \pi}(\bar{x}_i; \pi)}{p(\bar{x}_i)}$$

\bar{x}_i → Randomly sampled light paths

$p(\bar{x}_i)$ → Probability of sampling a path

Sample paths using path tracing etc.

Score estimator

$$f(\bar{\mathbf{x}}; \pi) = \prod_{b=1}^B f_s(x_{b-1} \rightarrow x_b \rightarrow x_{b+1}; \pi) \frac{V(x_{b-1} \leftrightarrow x_b)}{\|x_{b-1} - x_b\|^2}$$

Foreshortening terms are included in the BRDF

$$\frac{\partial f}{\partial \pi}(\bar{\mathbf{x}}; \pi) = \prod_{b=1}^B f_s(x_{b-1} \rightarrow x_b \rightarrow x_{b+1}; \pi) \frac{V(x_{b-1} \leftrightarrow x_b)}{\|x_{b-1} - x_b\|^2}$$

$$\sum_{b=1}^B \frac{\frac{\partial f_s}{\partial \pi}(x_{b-1} \rightarrow x_b \rightarrow x_{b+1}; \pi)}{f_s(x_{b-1} \rightarrow x_b \rightarrow x_{b+1}; \pi)}$$

At each path vertex:

- Update product throughput using f_s
- Update score sum using gradient of f_s

Multiply the two at end of path

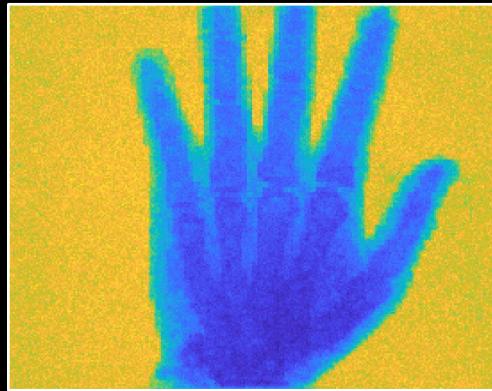
This is what all these papers do



everyday materials



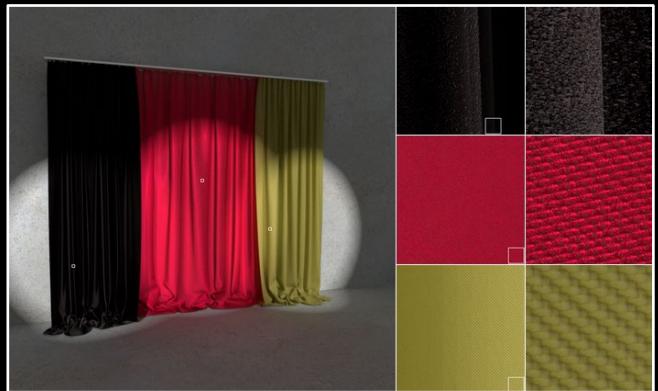
industrial
nanodispersions



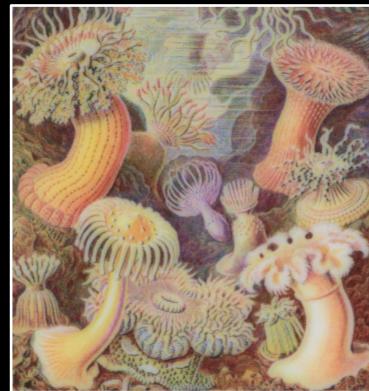
computed tomography
[Geva et al. 2018]



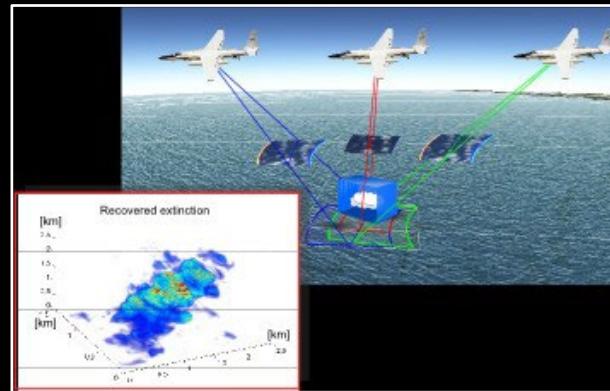
optical
tomography
[Gkioulekas et al.
2016]



woven fabrics
[Khungurn et al. 2015,
Zhao et al. 2016]

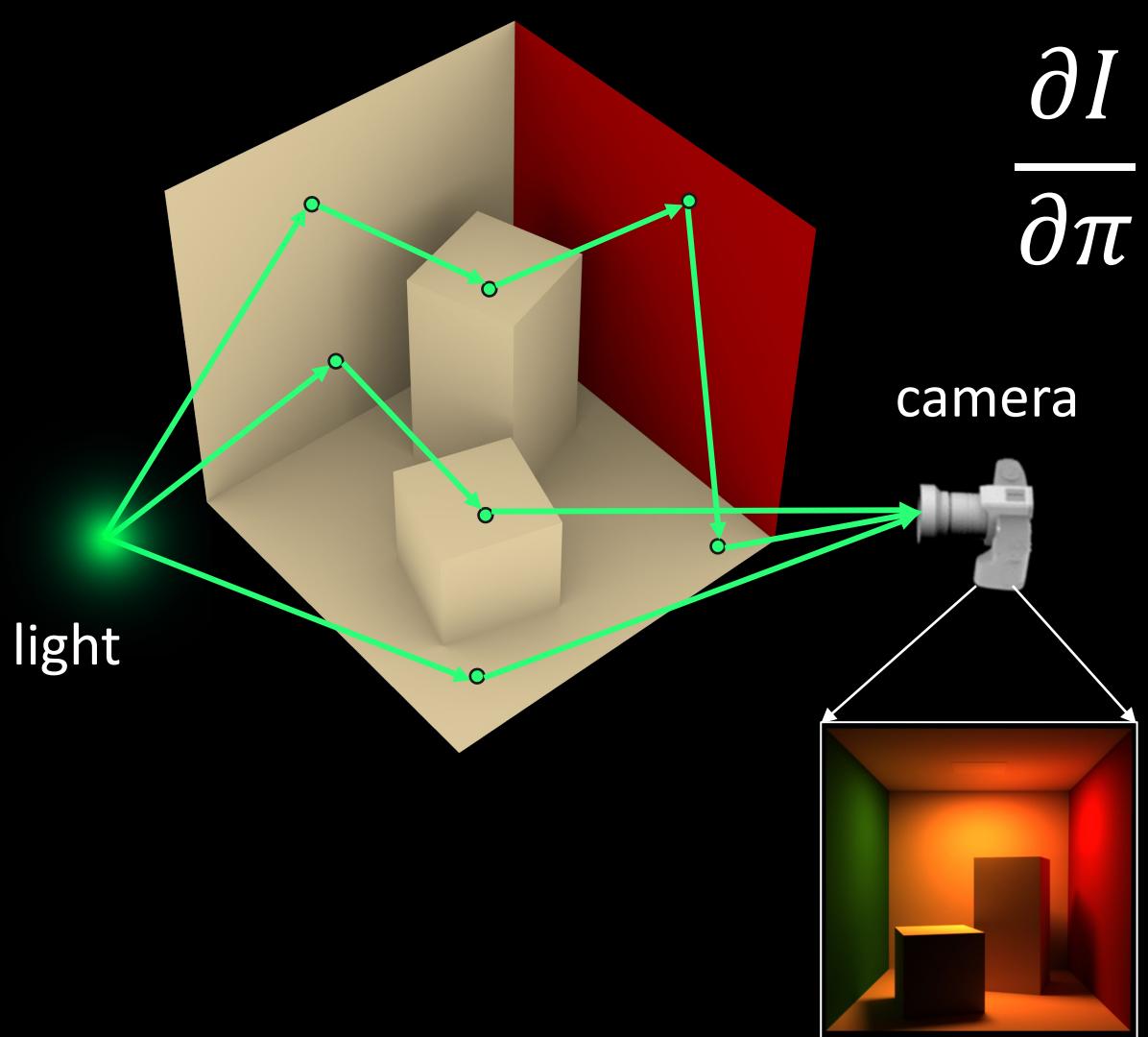


3D printing
[Elek et al. 2017, 2019]



clouds
[Levis et al. 2015, 2017]

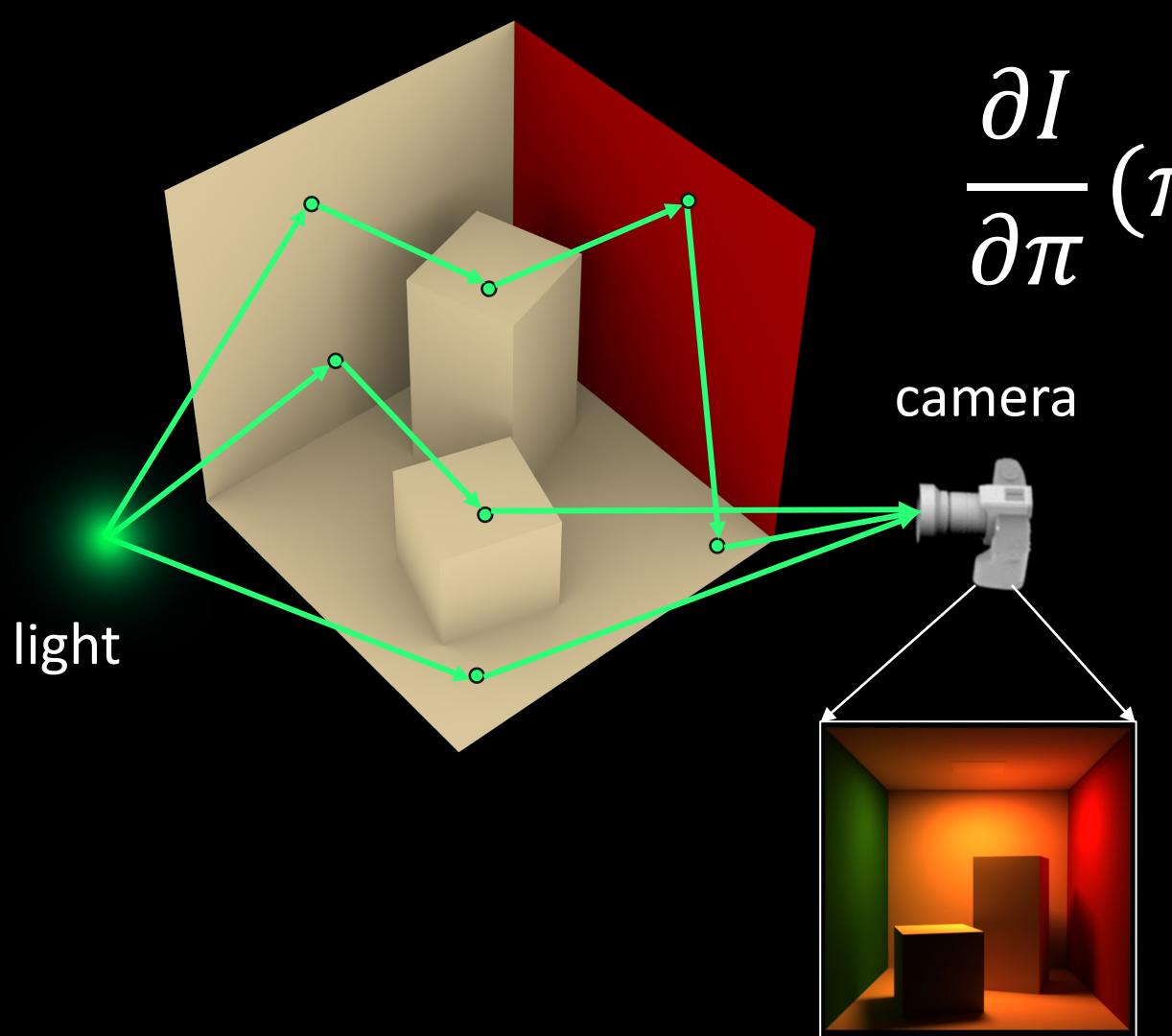
Even simpler: use autodiff



$$\frac{\partial I}{\partial \pi}(\pi) \approx \sum_{i=1}^N \frac{\text{autodiff}(f(\bar{\mathbf{x}}_i; \pi))}{p(\bar{\mathbf{x}}_i)}$$

$\bar{\mathbf{x}}_i$ → Randomly sampled light paths
 $p(\bar{\mathbf{x}}_i)$ → Probability of sampling a path

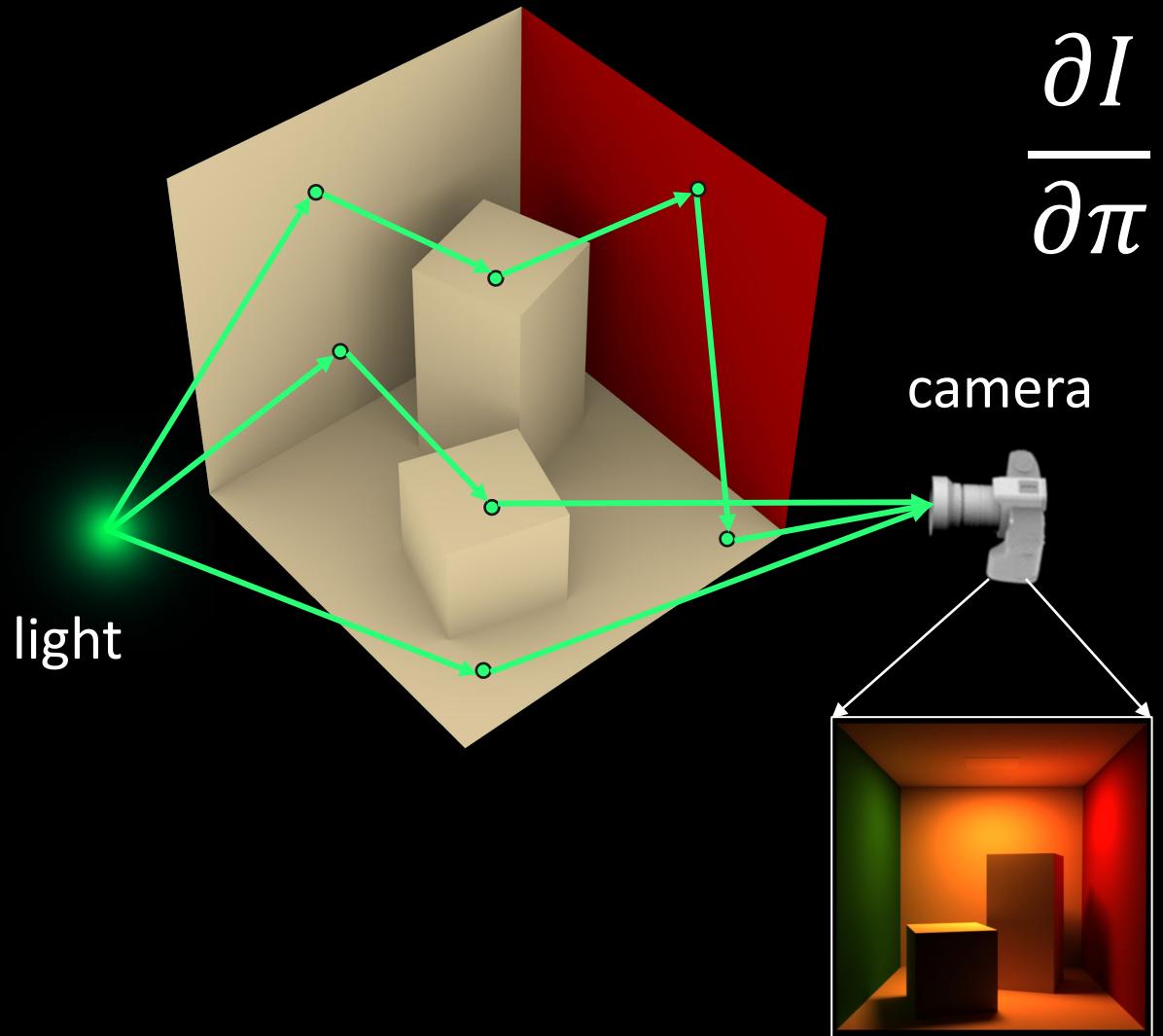
Compare with...



$$\frac{\partial I}{\partial \pi}(\pi) \approx \text{autodiff} \left(\sum_{i=1}^N \frac{f(\bar{\mathbf{x}}_i; \pi)}{p(\bar{\mathbf{x}}_i)} \right)$$

$\bar{\mathbf{x}}_i$ → Randomly sampled light paths
 $p(\bar{\mathbf{x}}_i)$ → Probability of sampling a path

Even simpler: use autodiff



$$\frac{\partial I}{\partial \pi}(\pi) \approx \sum_{i=1}^N \frac{\text{autodiff}(f(\bar{\mathbf{x}}_i; \pi))}{p(\bar{\mathbf{x}}_i)}$$

- Generally lower variance.
- Remember: *Compute an estimate of the derivative, not a derivative of the estimator.*

Compute an estimate of the derivative



derivative wrt volumetric density

Inverse Transport Networks

Chengqian Che
Carnegie Mellon University

Fujun Luan
Cornell University

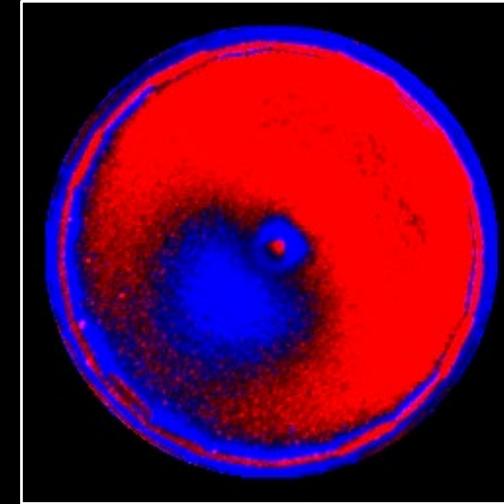
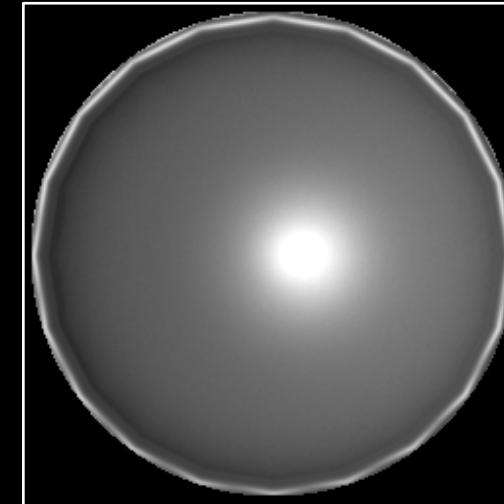
Shuang Zhao
University of California, Irvine

Kavita Bala
Cornell University

Ioannis Gkioulekas
Carnegie Mellon University



derivative wrt BRDF



derivative wrt normal

Comparison with finite differences



Forward



rendered



finite
differences

Note: Finite differences are great for testing the correctness of your gradient code.

Compute a derivative of the estimate



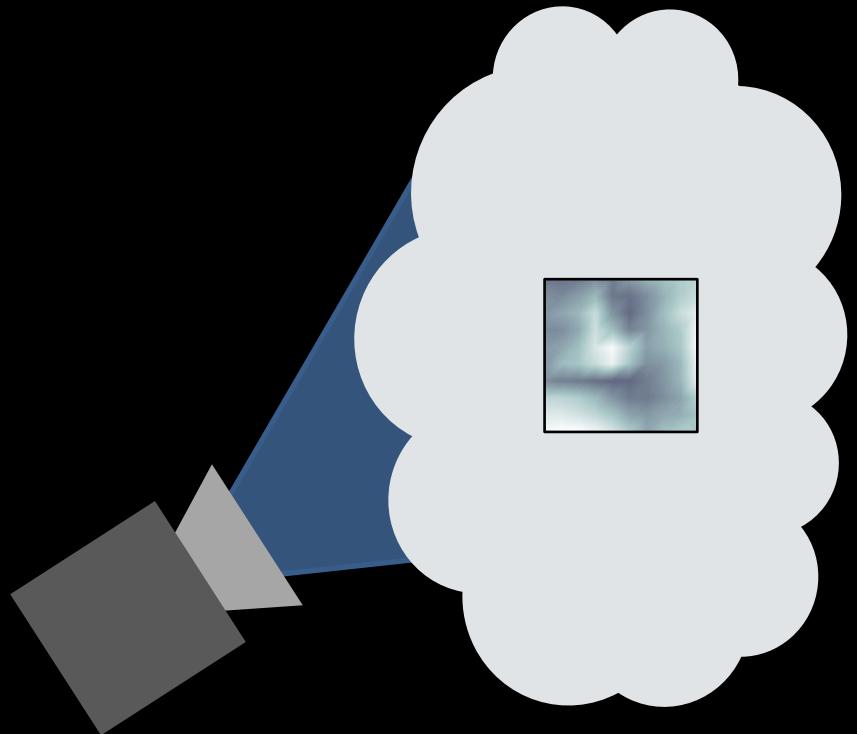
derivative wrt volumetric density

Mitsuba 2: A Retargetable Forward and Inverse Renderer

MERLIN NIMIER-DAVID*, École Polytechnique Fédérale de Lausanne
DELIO VICINI*, École Polytechnique Fédérale de Lausanne
TIZIAN ZELTNER, École Polytechnique Fédérale de Lausanne
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne

- A lot more general.
- GPU implementation.

Looking inside scattering objects

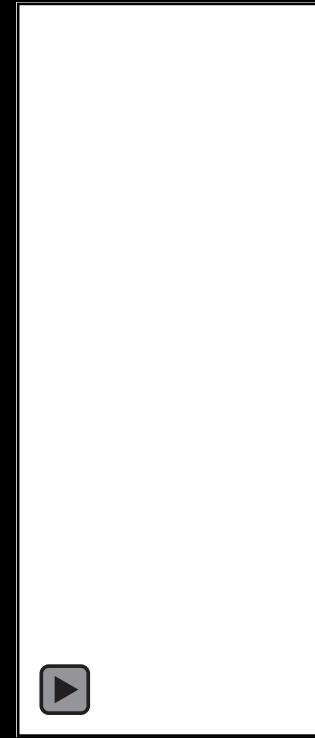


camera

thick smoke cloud



simulated camera
measurements

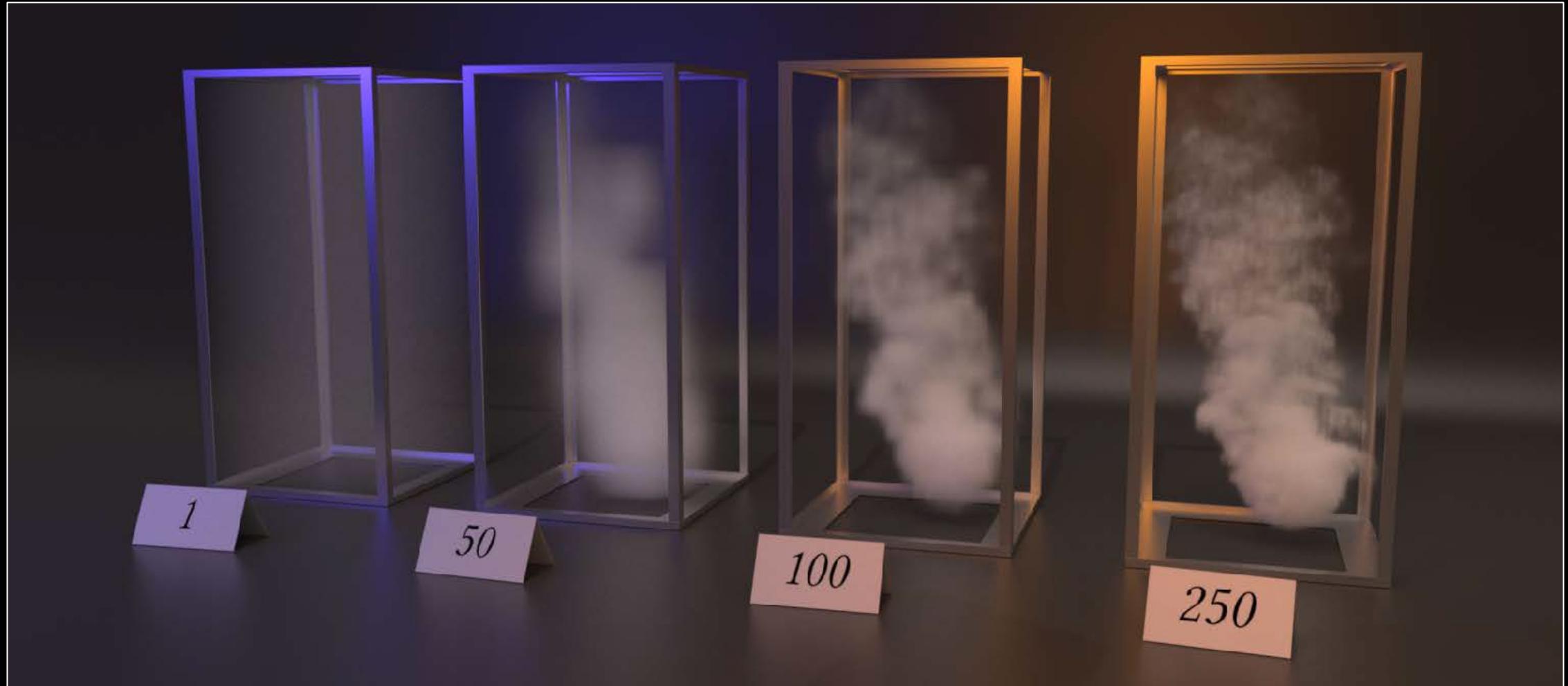


reconstructed cloud
volume

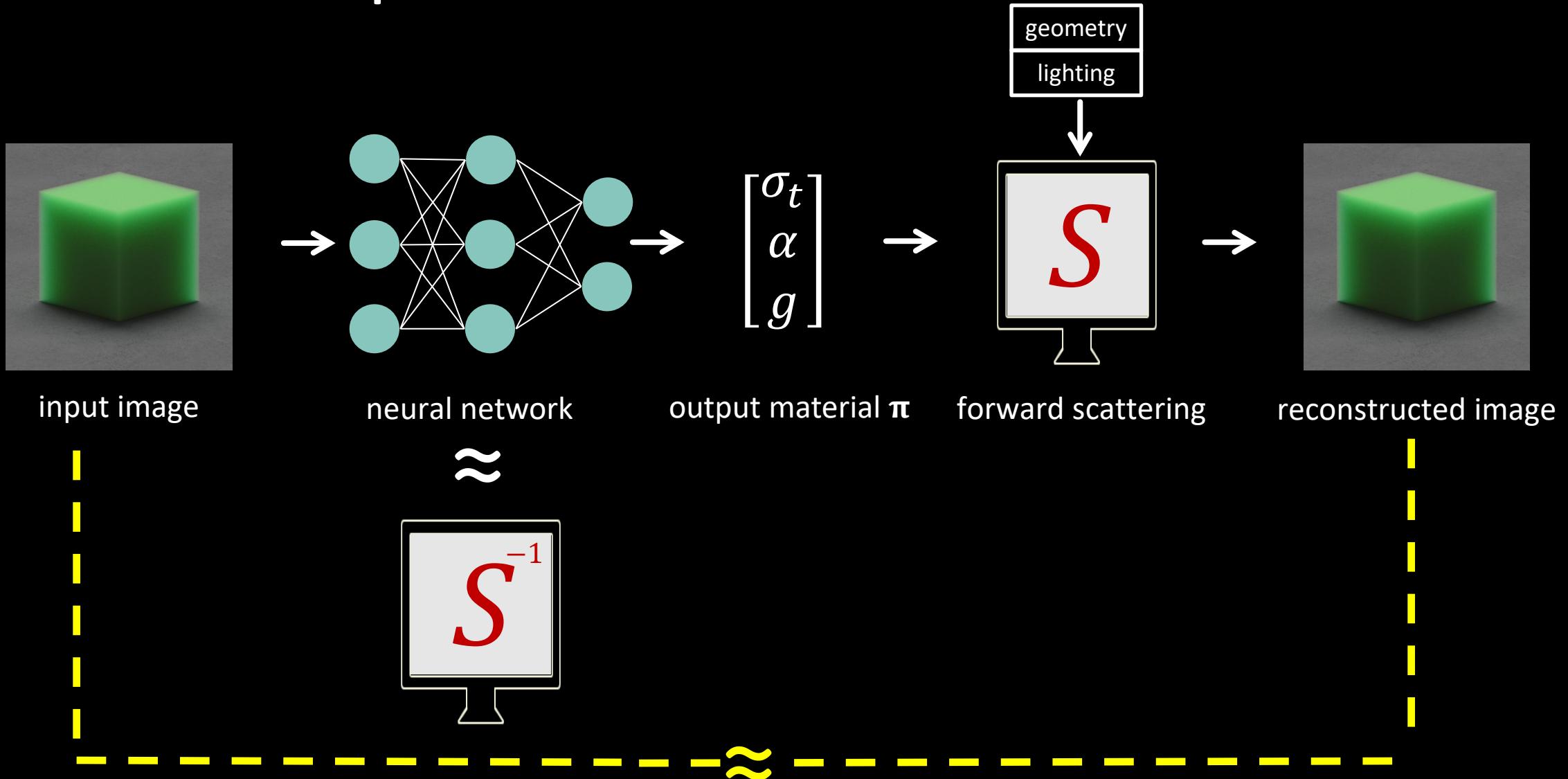


slice through
the cloud

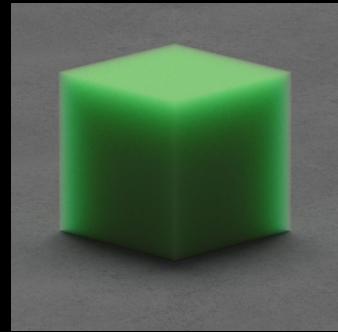
Looking inside scattering objects



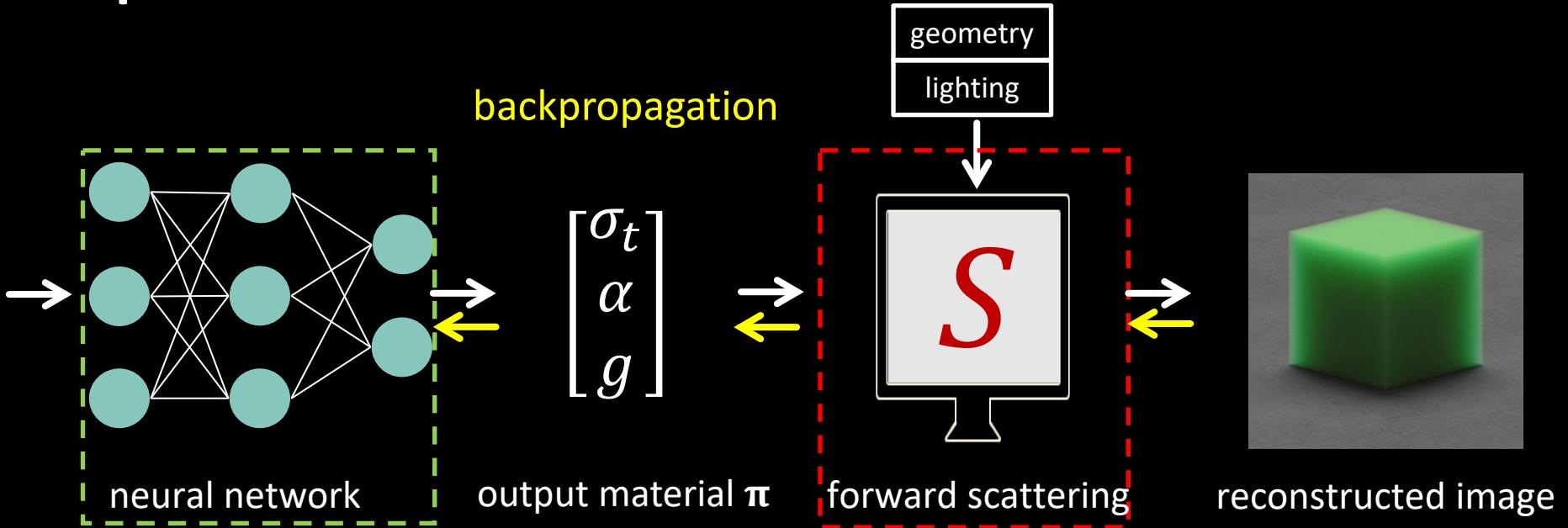
Inverse transport network



Inverse transport network



input image



derivatives

$$\frac{\partial \text{neuralNet}}{\partial \text{weights}}$$

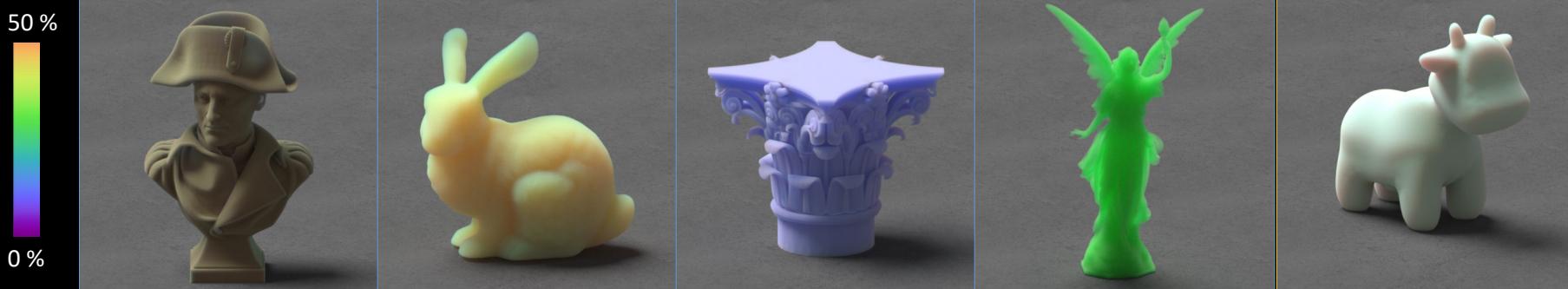
auto-diff (Torch, TensorFlow etc.)
parameter loss

$$\frac{\partial \text{forwardScattering}(\pi)}{\partial \pi}$$

volumetric differentiable renderer

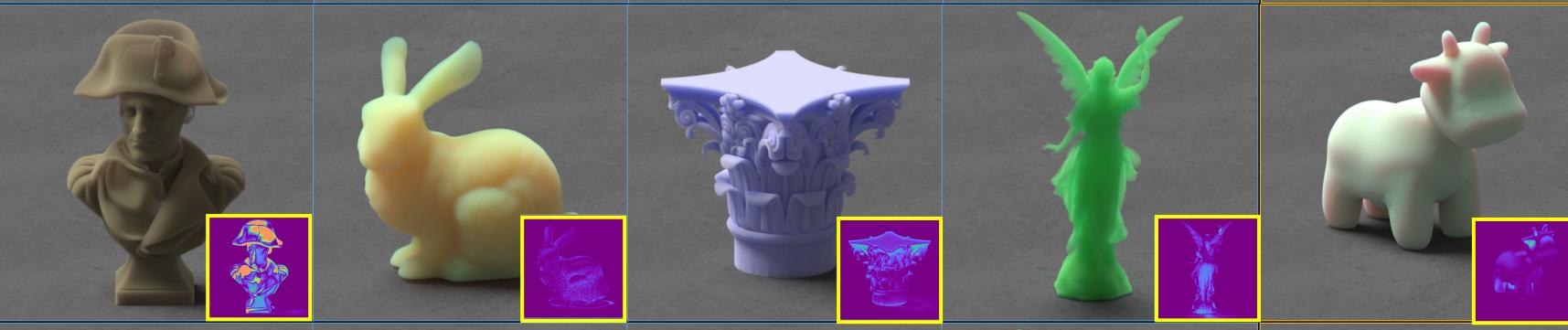
Examples

groundtruth



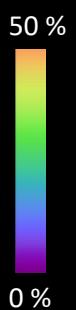
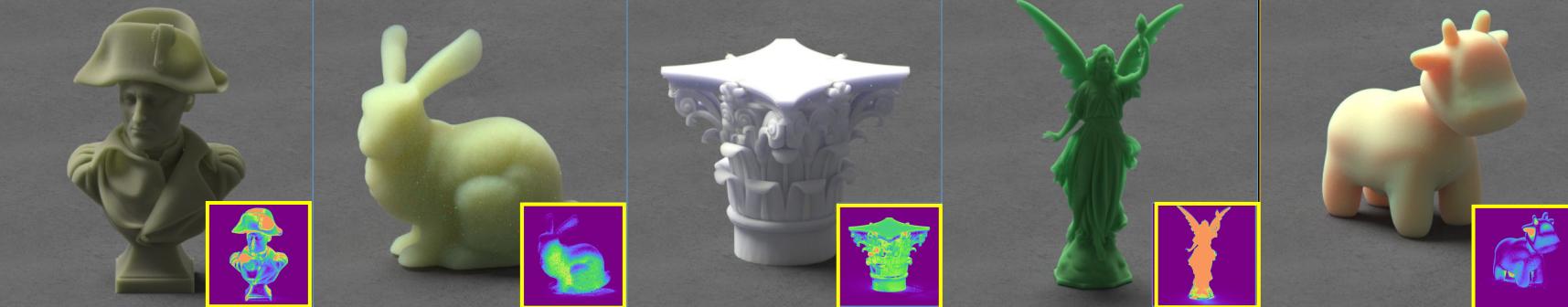
supervised and unsupervised

parameter loss: 0.60X
appearance loss: 0.40X
novel appearance loss: 0.42X

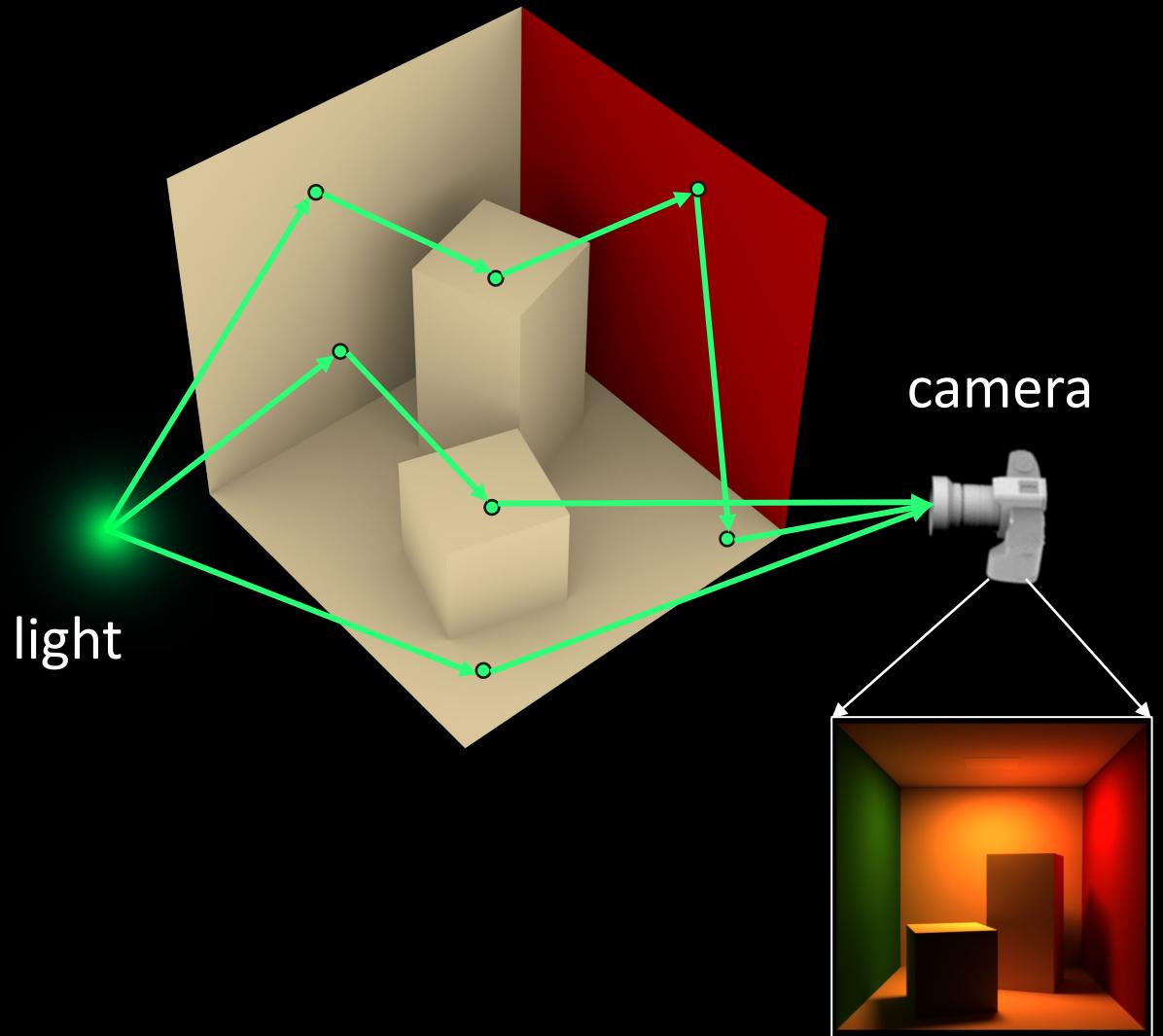


supervised only

parameter loss: 1X
appearance loss: 1X
novel appearance loss: 1X



Derivatives of images as path integrals



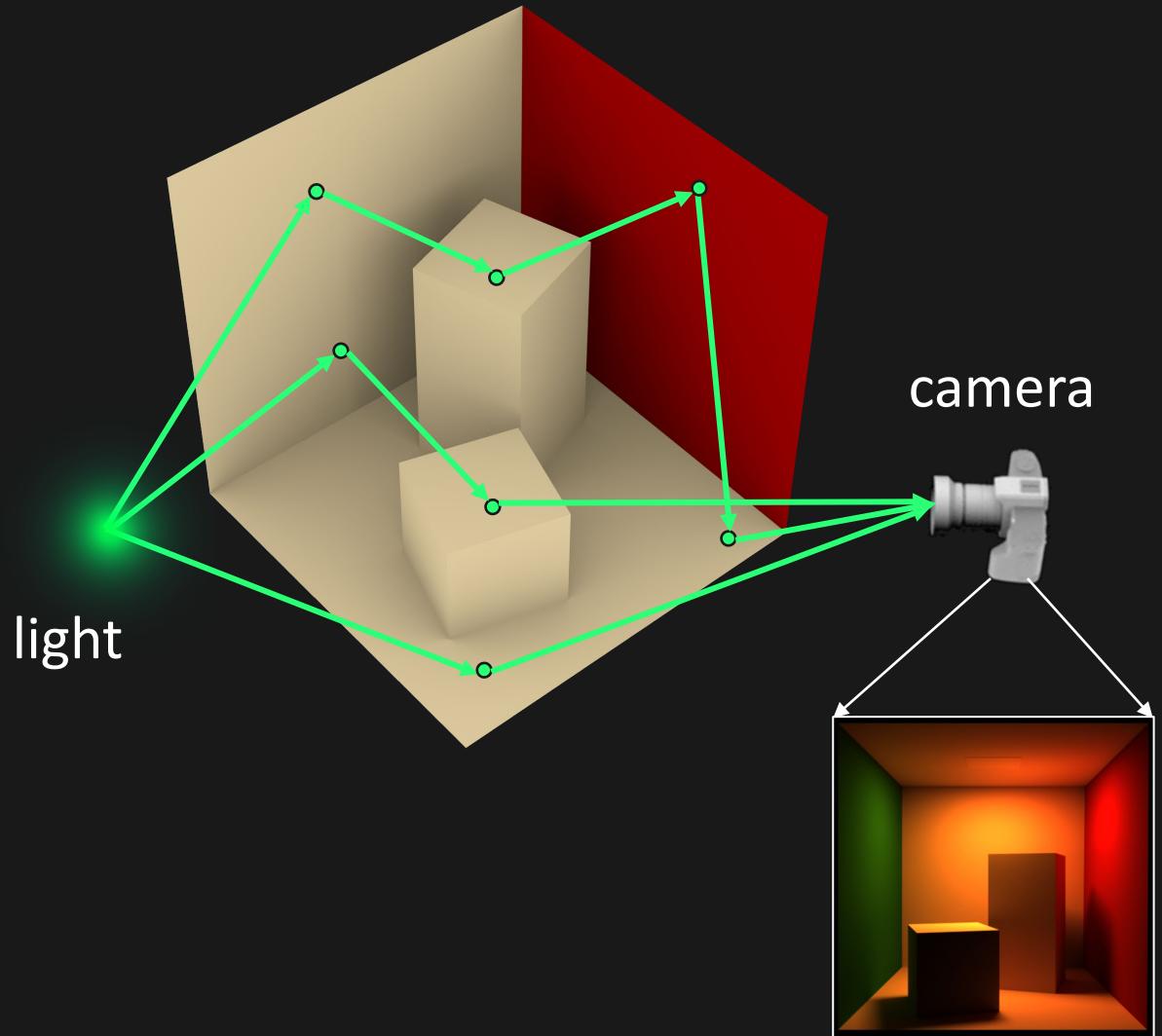
$$\frac{\partial I}{\partial \pi}(\pi) = \int_{\mathbb{P}} \frac{\partial f}{\partial \pi}(\bar{x}; \pi) d\bar{x}$$

differentiation under the integral sign

- \bar{x} → Light path, set of ordered vertices on surfaces
- \mathbb{P} → Space of valid paths
- $f(\bar{x})$ → Path contribution,
includes geometric terms (visibility, fall-off) &
local terms (BRDF, foreshortening, emission)

Assume \mathbb{P} is independent of π

Derivatives of images as path integrals



$$\frac{\partial I}{\partial \pi}(\pi) = \int_{\mathbb{P}} \frac{\partial f}{\partial \pi}(\bar{\mathbf{x}}; \pi) d\bar{\mathbf{x}}$$

differentiation under the integral sign

What about parameters π that change \mathbb{P} ?

- Location, pose, and shape of light, camera, and scene objects.

Differentiable rendering for global geometry

We'll work with the rendering equation

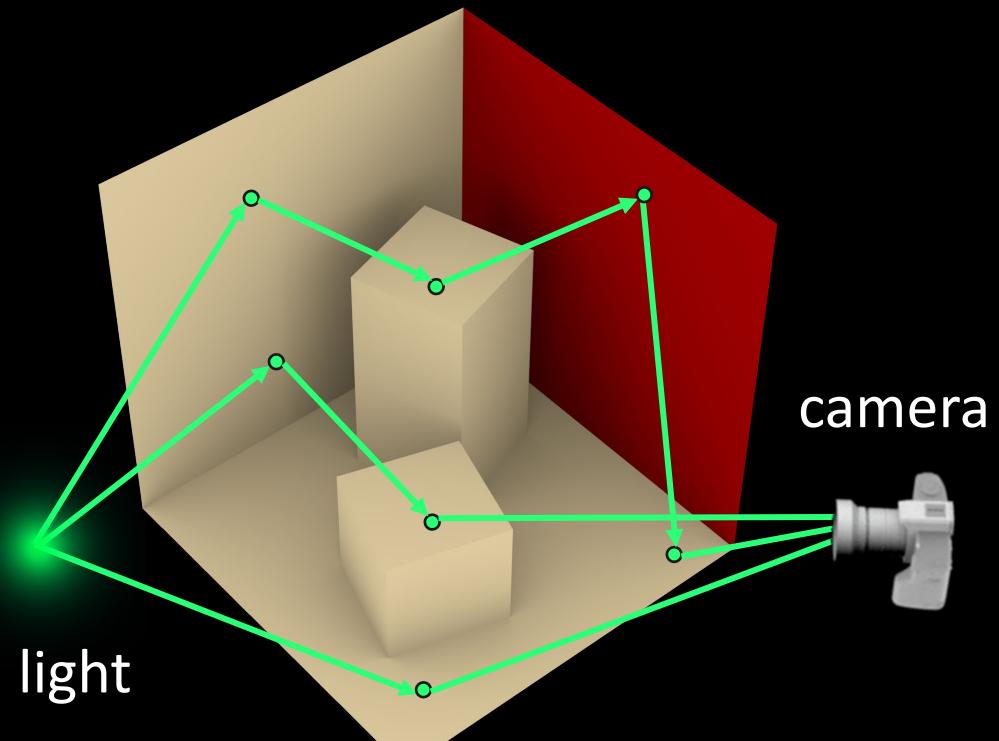
$$L(x, \omega; \pi) = \int_{G(\pi)} L(x' \rightarrow x; \pi) f(x' \rightarrow x, \omega; \pi) V(x' \leftrightarrow x; \pi) dA(x')$$

L → Radiance at a point and direction

G → All surfaces in the scene

f → Reflection, foreshortening, and fall-off

V → Visibility



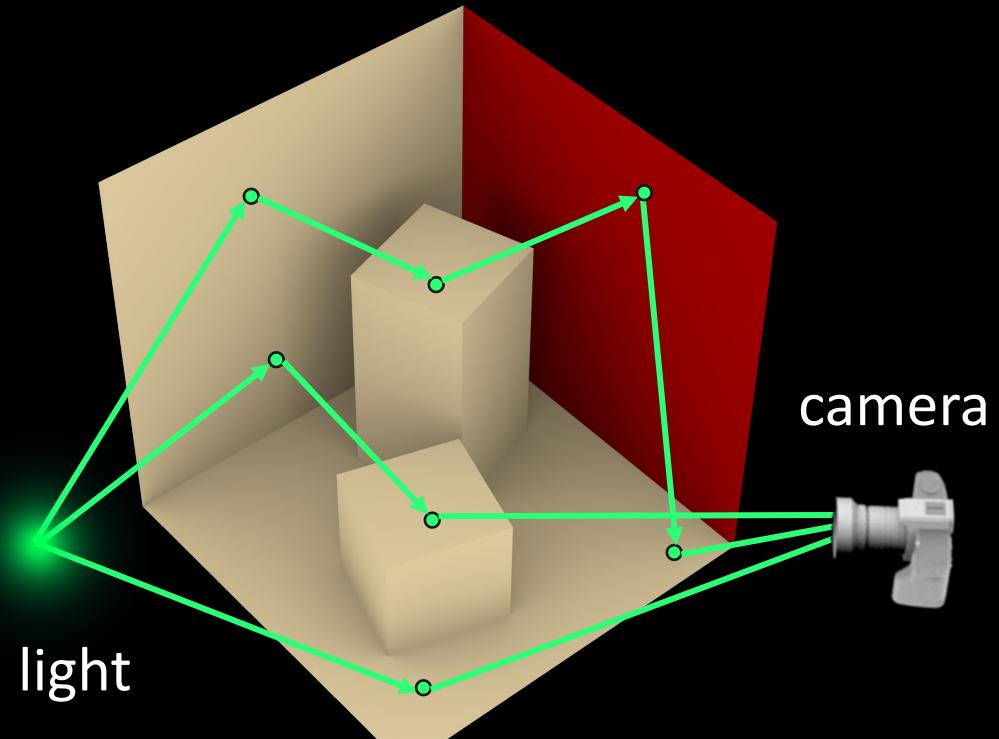
Let's slightly rewrite the rendering equation

$$L(x, \omega; \pi) = \int_{V(x, \pi)} L(x' \rightarrow x; \pi) f(x' \rightarrow x, \omega; \pi) dA(x')$$

L → Radiance at a point and direction

V → All visible surfaces in the scene

f → Reflection, foreshortening, and fall-off



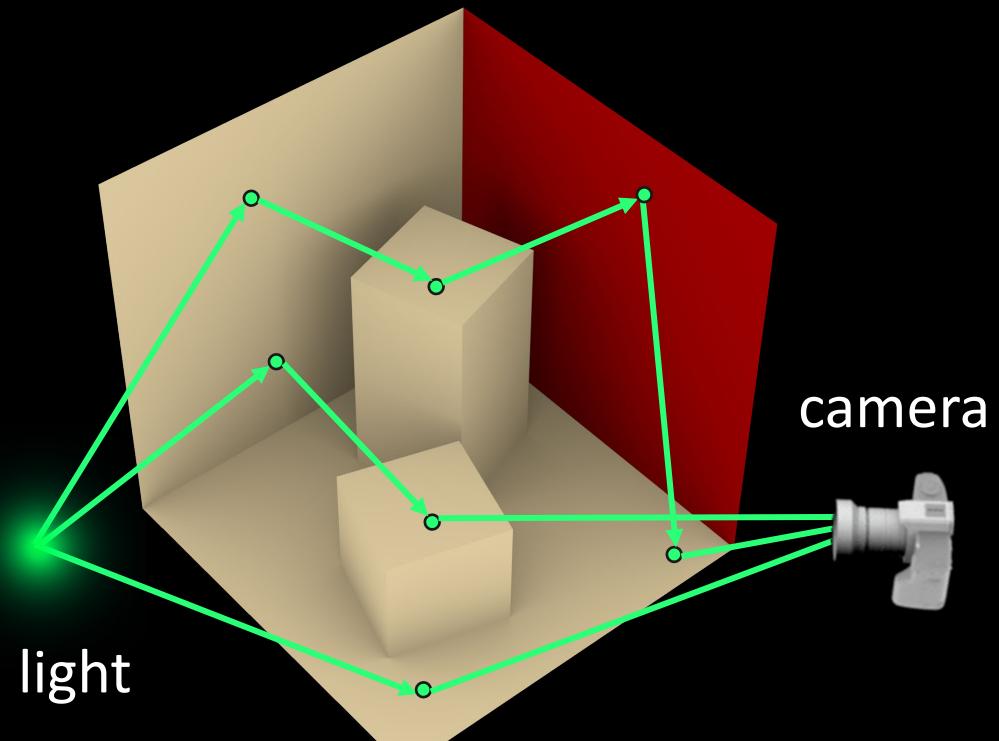
Let's differentiate it

$$\frac{\partial}{\partial \pi} L(x, \omega; \pi) = \frac{\partial}{\partial \pi} \int_{V(x, \pi)} L(x' \rightarrow x; \pi) f(x' \rightarrow x, \omega; \pi) dA(x')$$

$L \rightarrow$ Radiance at a point and direction

$V \rightarrow$ All visible surfaces in the scene

$f \rightarrow$ Reflection, foreshortening, and fall-off



Can we just move the integral inside?

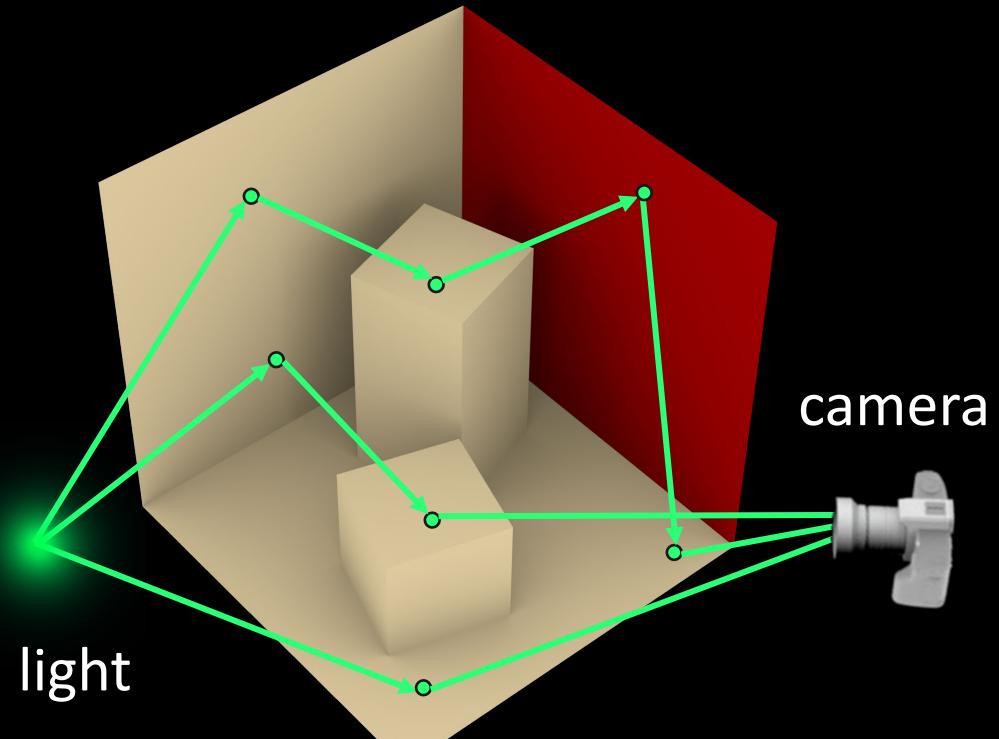
Let's differentiate it

$$\frac{\partial}{\partial \pi} L(x, \omega; \pi) = \frac{\partial}{\partial \pi} \int_{V(x, \pi)} L(x' \rightarrow x; \pi) f(x' \rightarrow x, \omega; \pi) dA(x')$$

$L \rightarrow$ Radiance at a point and direction

$V \rightarrow$ All visible surfaces in the scene

$f \rightarrow$ Reflection, foreshortening, and fall-off



Can we just move the integral inside?

- No. What can we do?

Basic differentiation rules

$$\frac{\partial}{\partial \pi} \int_a^b f(x; \pi) dx = \int_a^b \frac{\partial}{\partial \pi} f(x; \pi) dx \quad \text{differentiation under the integral sign}$$

$$\begin{aligned} \frac{\partial}{\partial \pi} \int_{a(\pi)}^{b(\pi)} f(x; \pi) dx &= \int_{a(\pi)}^{b(\pi)} \frac{\partial}{\partial \pi} f(x; \pi) dx && \text{Leibniz integral rule} \\ &+ f(b(\pi); \pi) \frac{\partial b(\pi)}{\partial \pi} - f(a(\pi); \pi) \frac{\partial a(\pi)}{\partial \pi} \end{aligned}$$

We need a version of this for surface integrals

Reynolds transport theorem for surfaces

$$\frac{\partial}{\partial \pi} \int_{S(\pi)} f(x; \pi) dA(x) = \int_{S(\pi)} \dot{f} dA(x) + \int_{\partial S(\pi)} f \left(t, \frac{\partial x}{\partial \pi} \right) dl(x)$$



surface
integral

line integral on *boundary*
and *discontinuities*

REYNOLDS TRANSPORT THEOREM

$$\frac{d}{d\pi} \int_{\Omega} f dA = \int_{\Omega} \frac{df}{d\pi} dA + \int_{\partial\Omega} g dl$$

Reynolds transport theorem

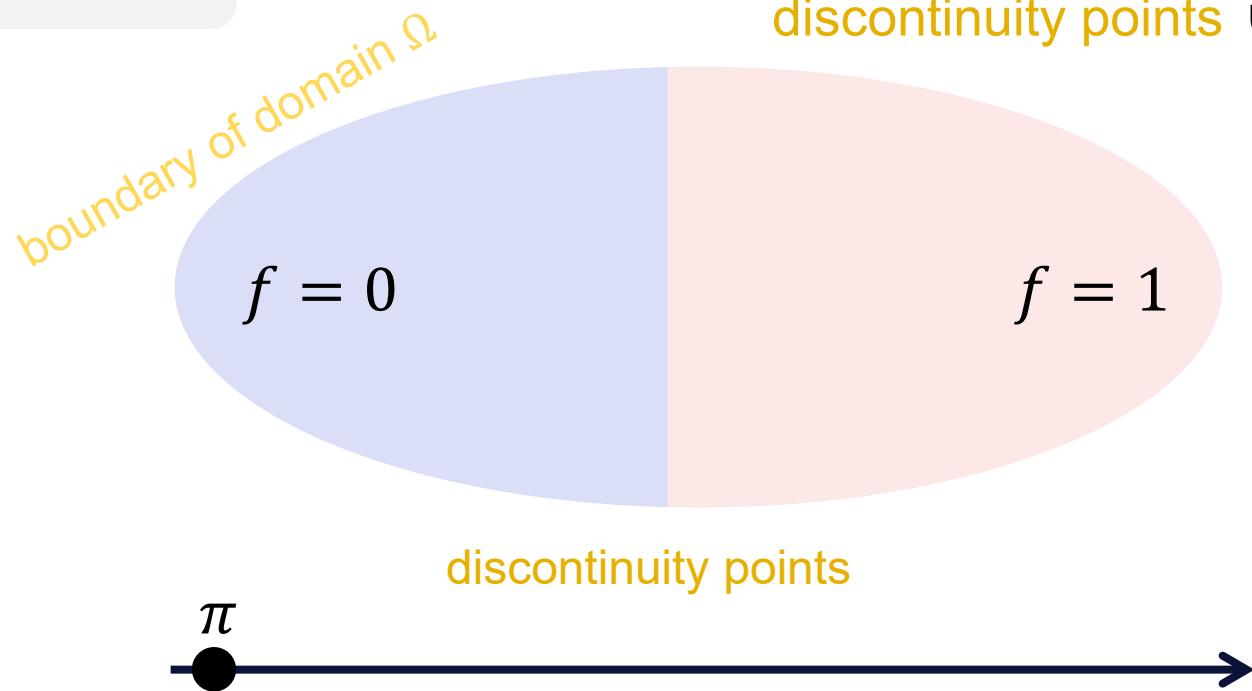
Generalization of Leibniz's rule

Interior integral

Boundary domain

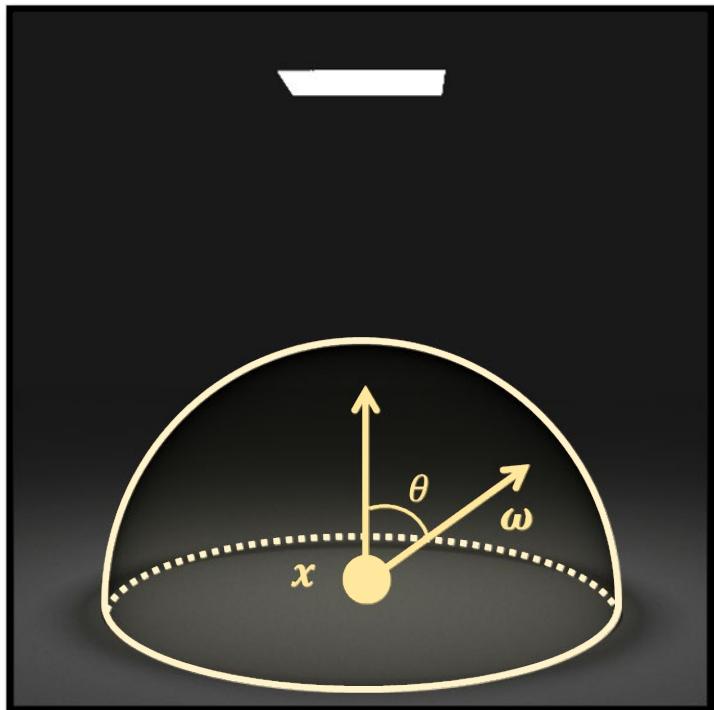
boundary integral

discontinuity points \cup boundary of domain Ω



REYNOLDS TRANSPORT THEOREM

π : size of the emitter



Irradiance at x

$$E = \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

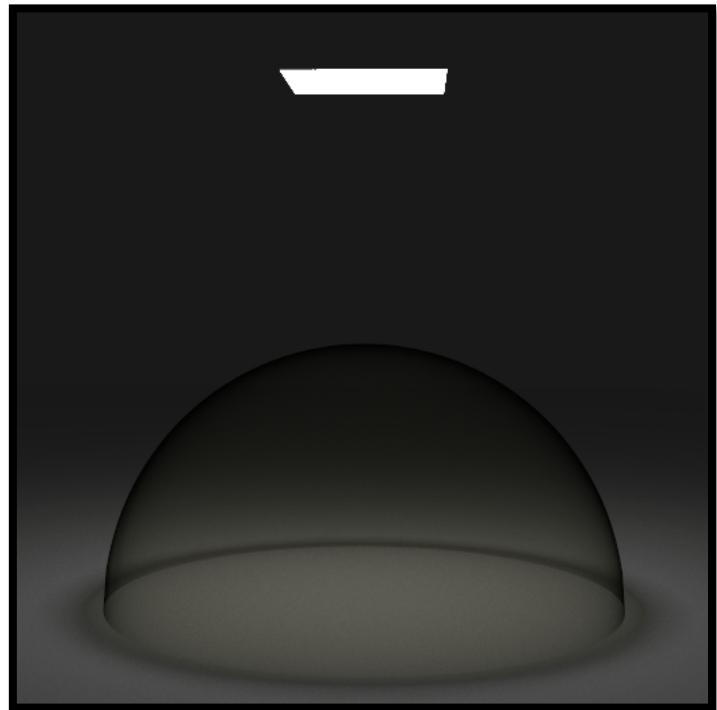
Unit hemisphere

Differential irradiance at x

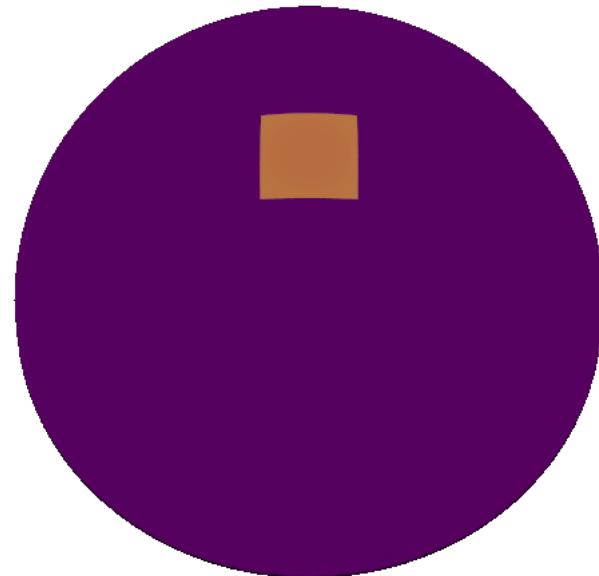
$$\frac{dE}{d\pi} = \frac{d}{d\pi} \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

REYNOLDS TRANSPORT THEOREM

π : size of the emitter



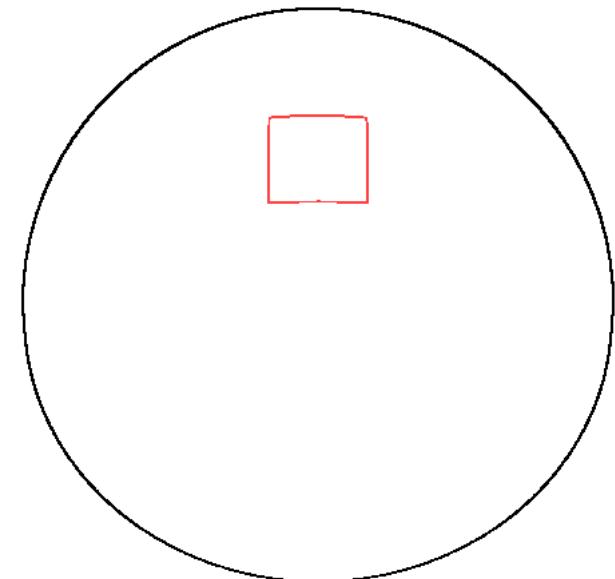
Low High



The integrand

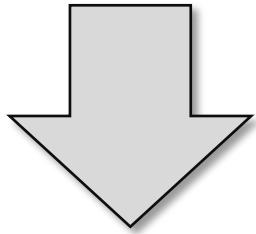
$$E = \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

Discontinuous points
(π -dependent)



REYNOLDS TRANSPORT THEOREM

$$E = \int_{\mathbb{H}^2} \overbrace{L_i(\omega) \cos\theta}^f d\sigma(\omega)$$

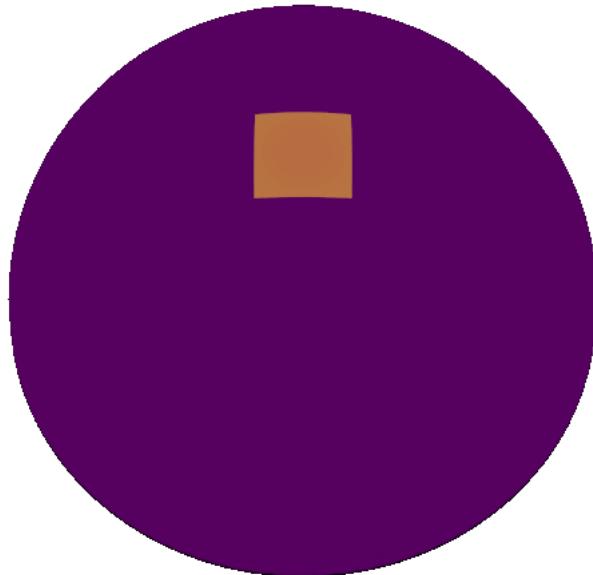


$$\frac{dE}{d\pi} = \int_{\mathbb{H}^2} \frac{df}{d\pi} d\sigma + \int_{\partial \mathbb{H}^2} g dl$$

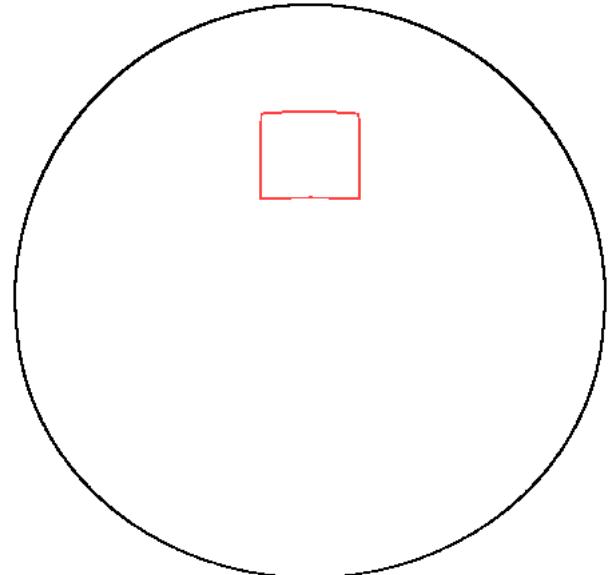
= 0

$\neq 0$

Low High



The integrand



Discontinuous points
(π -dependent)

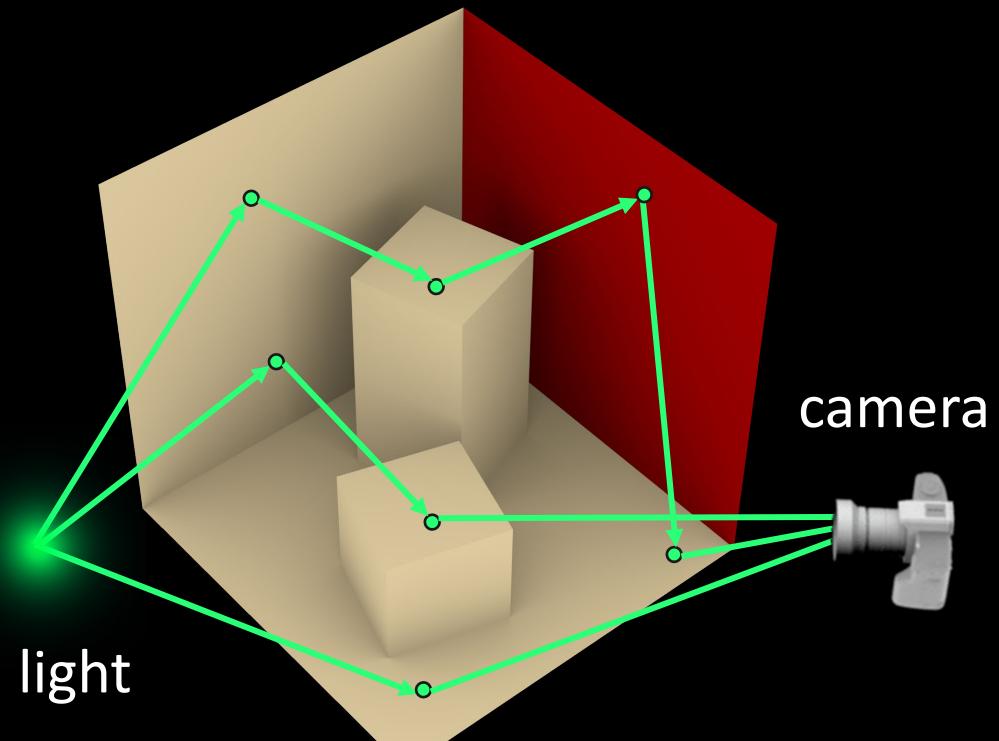
Let's differentiate the rendering equation

$$\frac{\partial}{\partial \pi} L(x, \omega; \pi) = \frac{\partial}{\partial \pi} \int_{V(x, \pi)} L(x' \rightarrow x; \pi) f(x' \rightarrow x, \omega; \pi) dA(x')$$

$L \rightarrow$ Radiance at a point and direction

$V \rightarrow$ All visible surfaces in the scene

$f \rightarrow$ Reflection, foreshortening, and fall-off

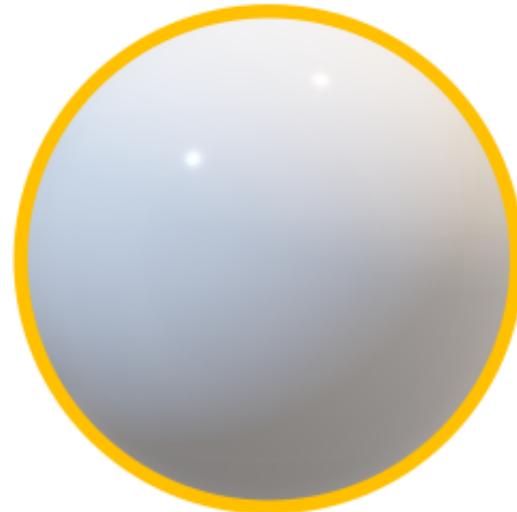


What are the “boundary” and discontinuities of V ?

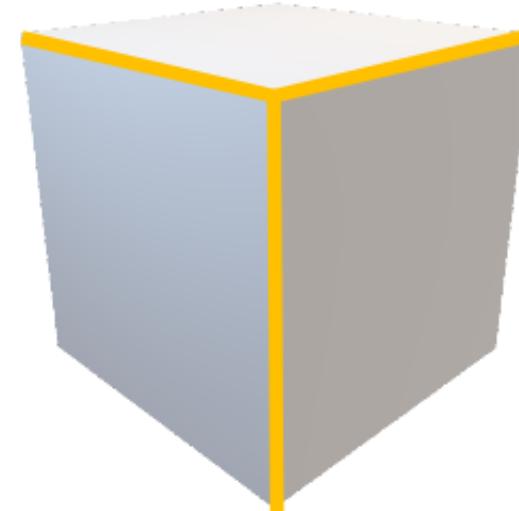
Boundaries



(a) **Boundary** edges



(b) **Silhouette** edges

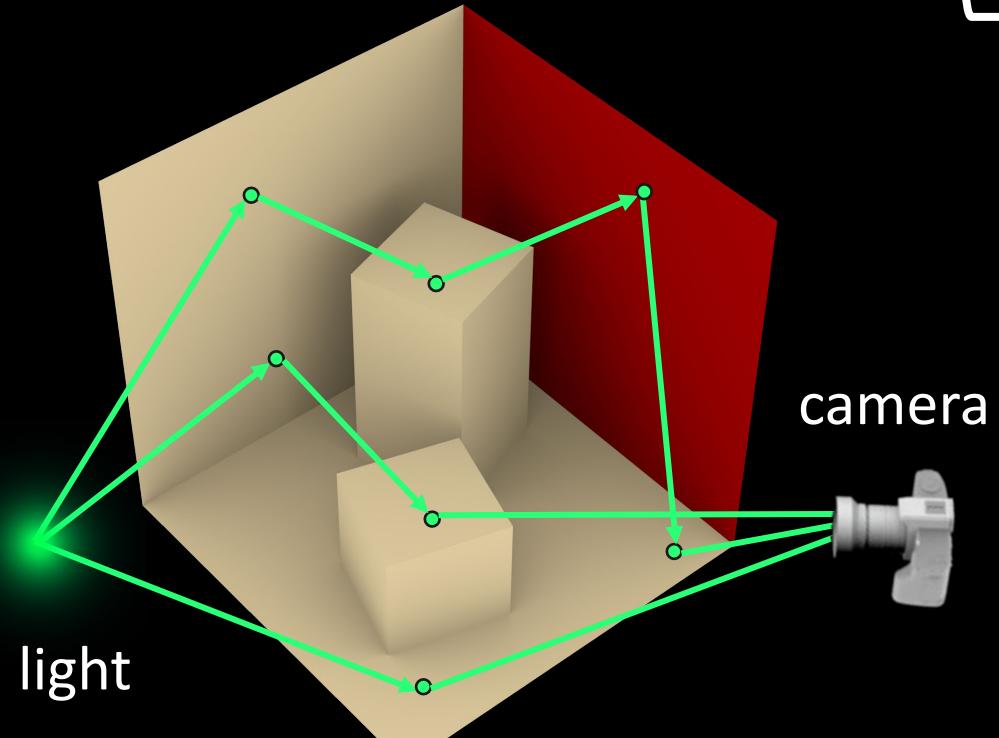


(c) **Sharp** edges

Fig. 5. Three types of edges (drawn in yellow) that can cause geometric discontinuities: (a) boundary, (b) silhouette, and (c) sharp.

Let's differentiate it

$$\frac{\partial}{\partial \pi} L(x, \omega; \pi) = \underbrace{\int_{V(x, \pi)} \frac{\partial}{\partial \pi} L dA(x)} + \underbrace{\int_{\partial V(x, \pi)} H(L) d\sigma(x)}$$



recursively estimate
derivative of L at
some visible point

recursively estimate
radiance L at some
boundary point

Not terribly good, as we ray trace, we need to:

- recompute silhouette at each vertex
- branch twice

Global geometry differentiation

Differentiable Monte Carlo Ray Tracing through Edge Sampling

TZU-MAO LI, MIT CSAIL

MIIKA AITTALA, MIT CSAIL

FRÉDO DURAND, MIT CSAIL

JAAKKO LEHTINEN, Aalto University & NVIDIA

Beyond Volumetric Albedo

— A Surface Optimization Framework for Non-Line-of-Sight Imaging

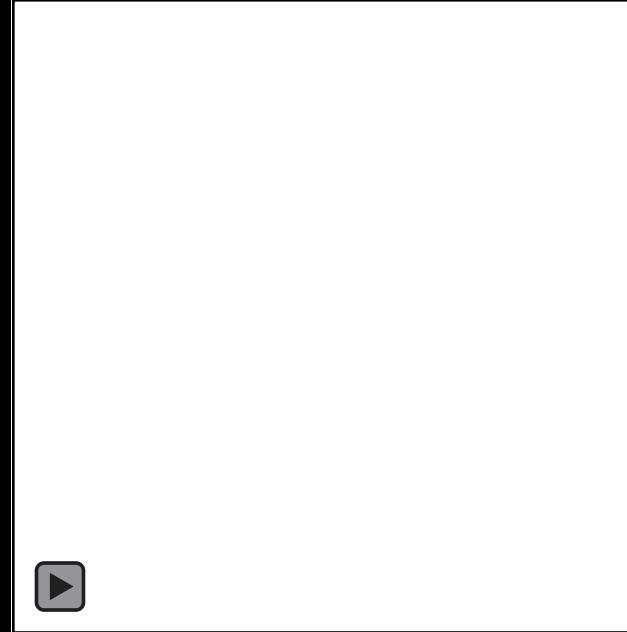
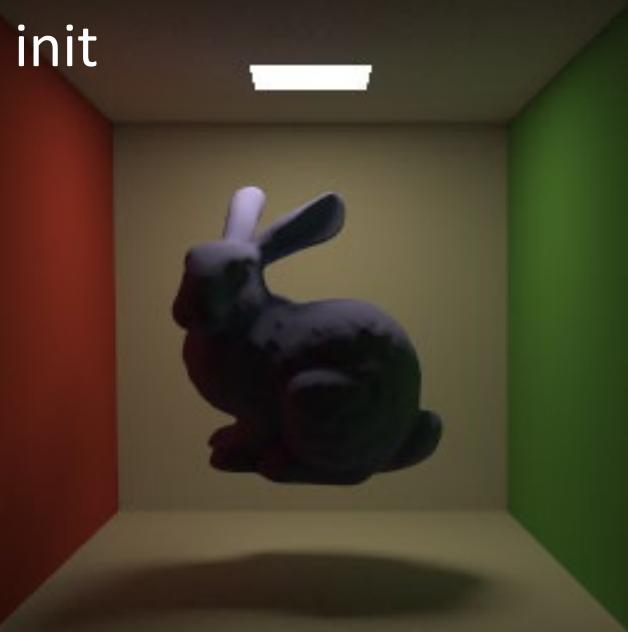
Chia-Yin Tsai, Aswin C. Sankaranarayanan, and Ioannis Gkioulekas
Carnegie Mellon University

Global geometry differentiation

target

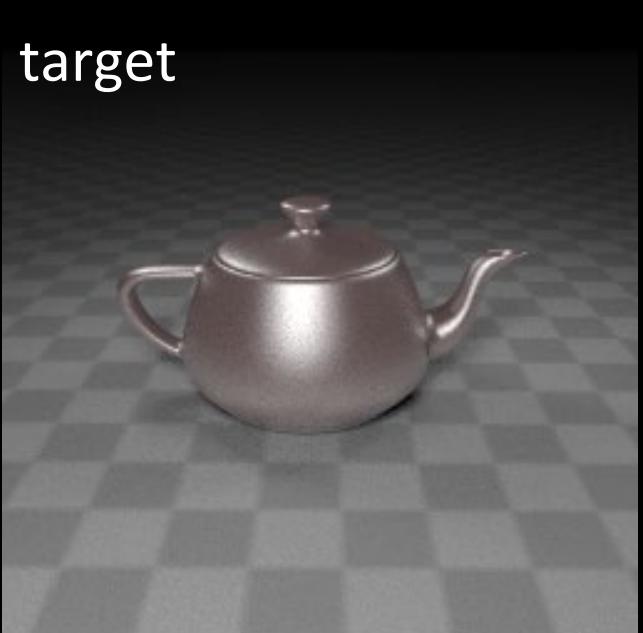


init

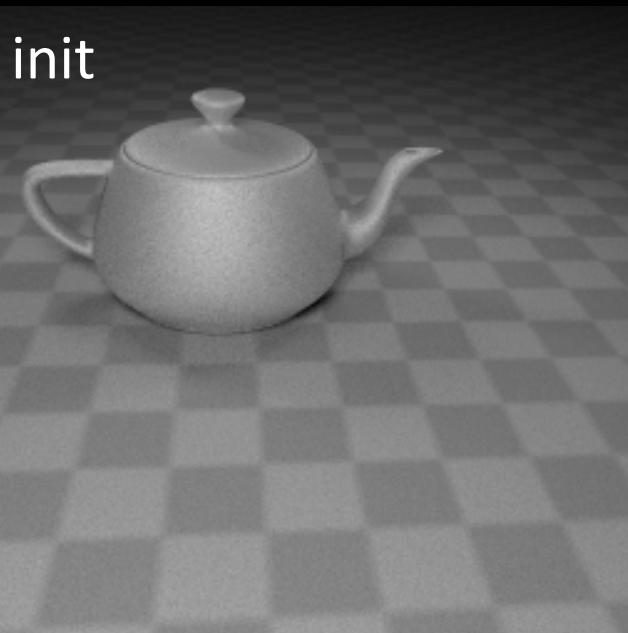


optimize
bunny
pose

target



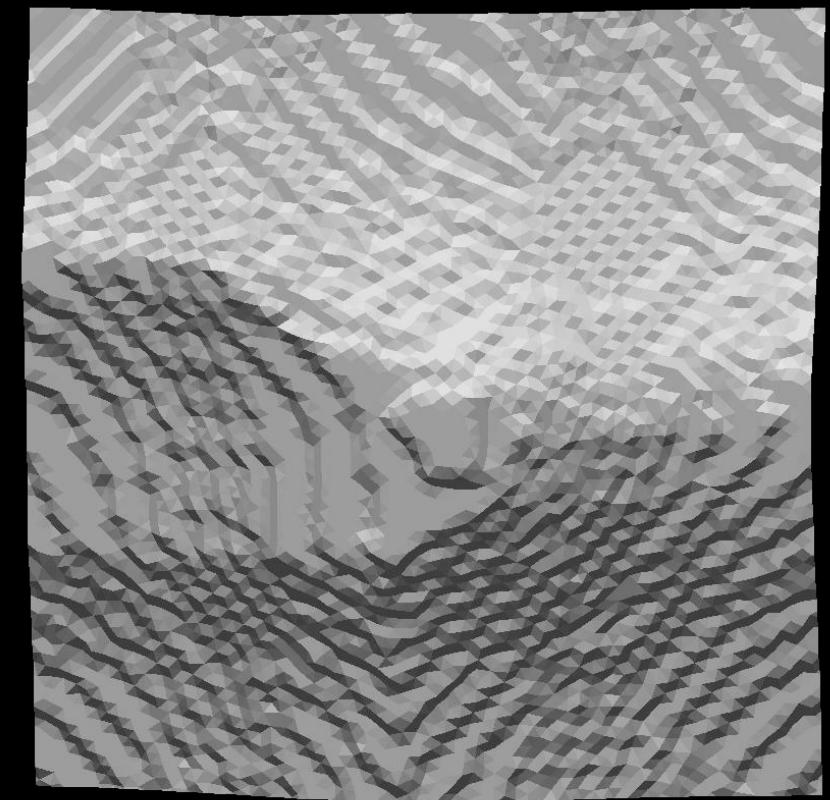
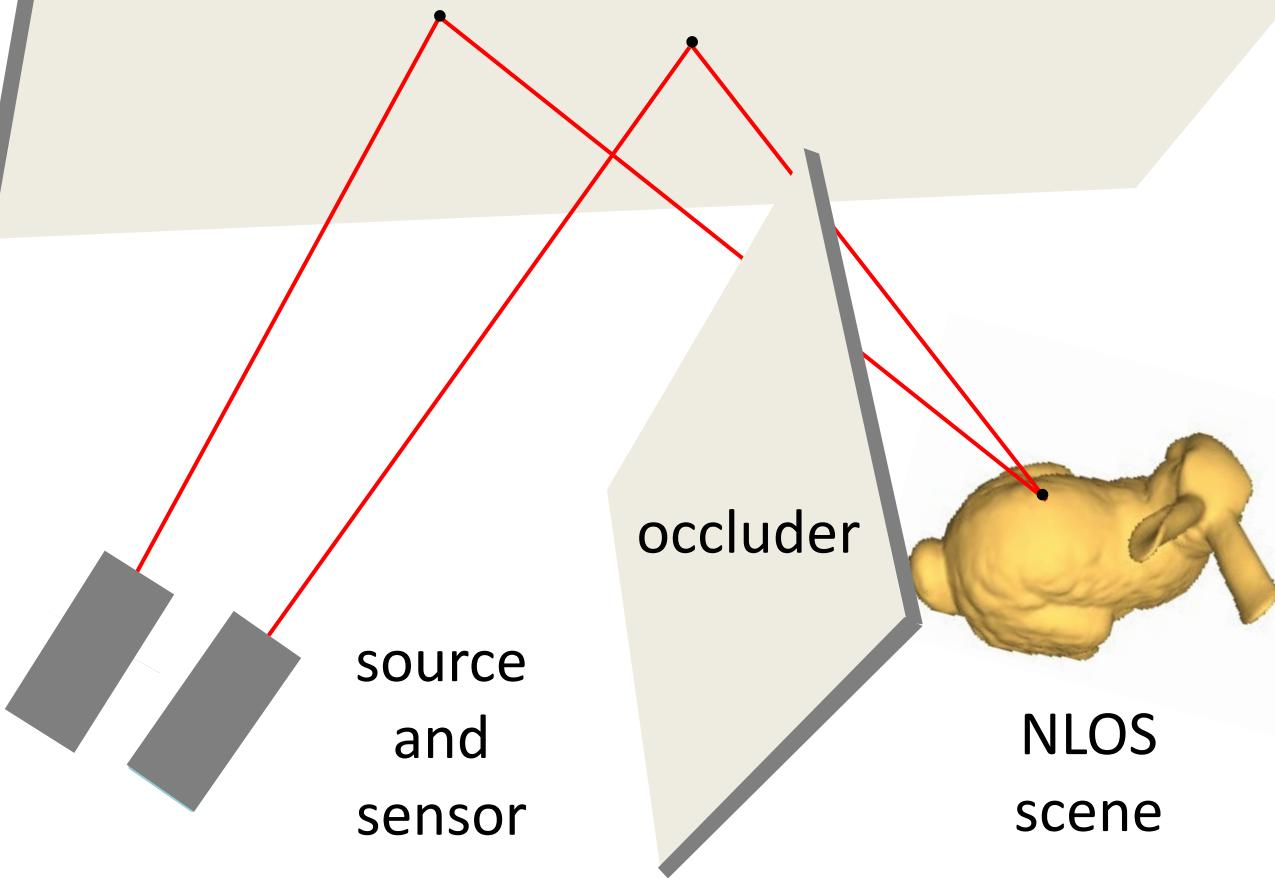
init



optimize
reflectance
and camera
pose

Optimize shape

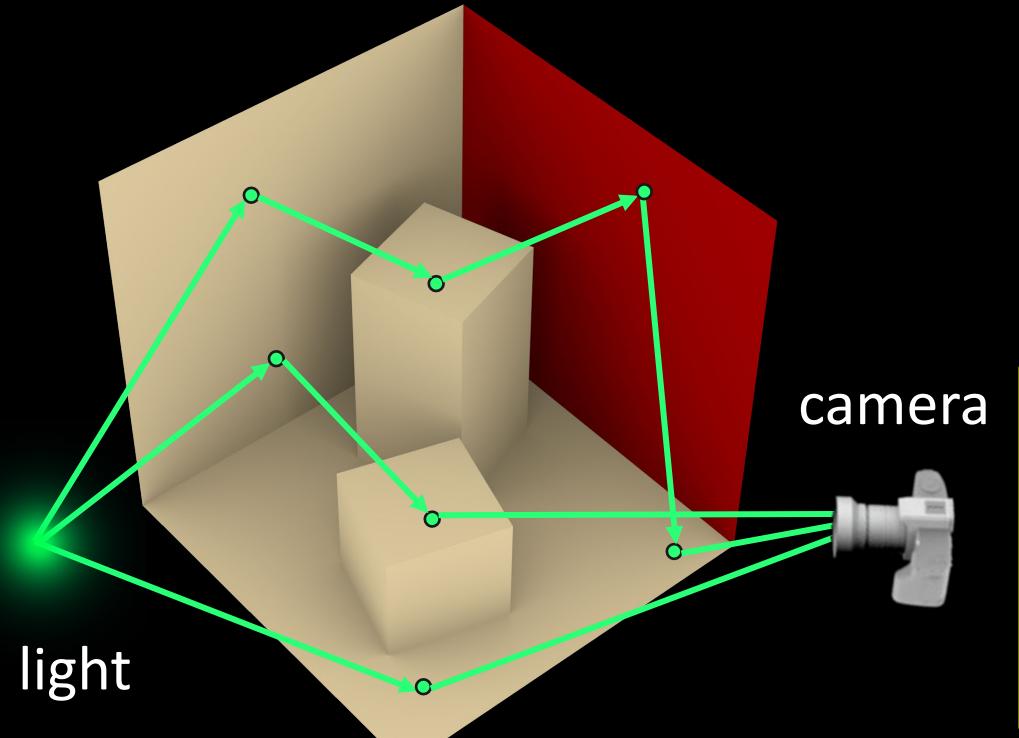
visible surface



reconstruction evolution

Let's differentiate it

$$\begin{aligned}\frac{\partial}{\partial \pi} L(x, \omega; \pi) \\ = \int_{V(x, \pi)} F \left(\underbrace{\frac{\partial}{\partial \pi} L}_{\text{render derivative}} \right) dA(x) + \int_{\partial V(x, \pi)} H(L) d\sigma(x)\end{aligned}$$



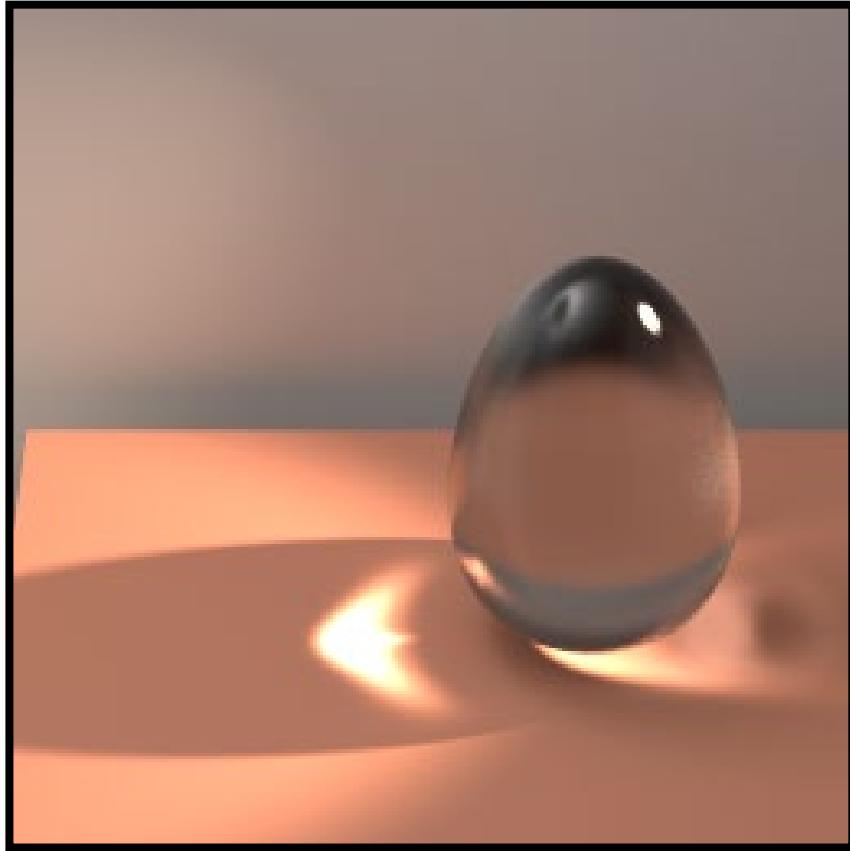
render derivative
of L at some
visible point

render L at some
boundary
(silhouette) point

Not terribly good:

- As we ray trace, we need to recompute silhouette
- Branching of two at each recursion

CHALLENGES



Complex light transport effects



Complex geometry

PATH-INTEGRAL FOR DIFFERENTIABLE RENDERING

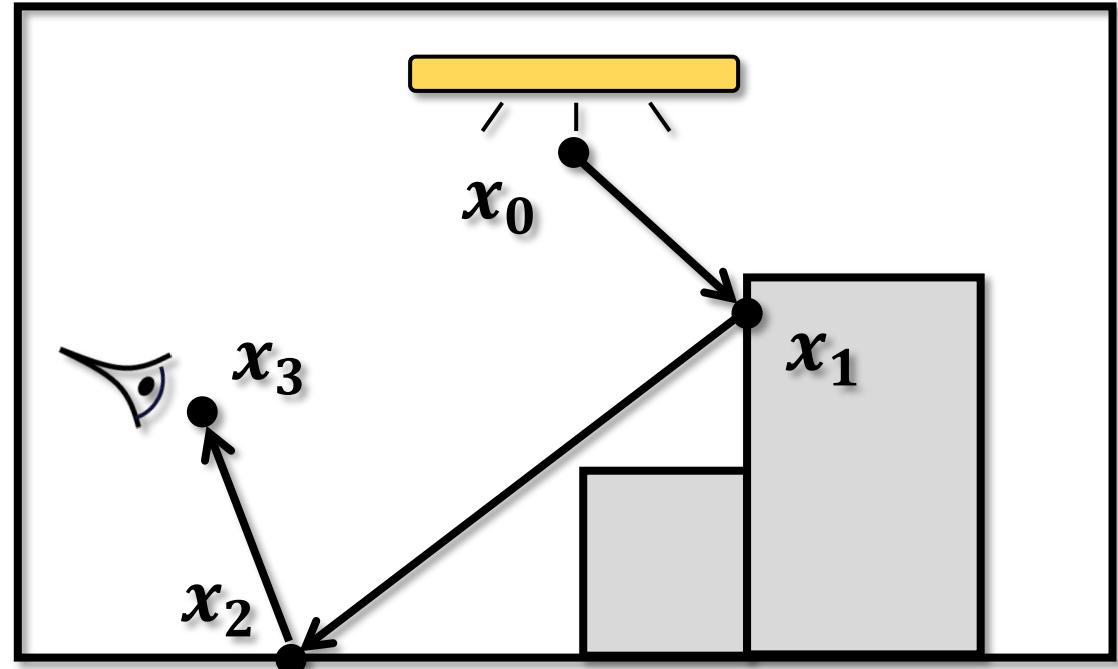
FORWARD PATH INTEGRAL

$$I = \int_{\Omega} f(\bar{x}) \, d\mu(\bar{x})$$

Measurement contribution function

Path space

Area-product measure



Light path $\bar{x} = (x_0, x_1, x_2, x_3)$

DIFFERENTIAL PATH INTEGRAL

Path Integral

$$I = \int_{\Omega} f(\bar{x}) d\mu(\bar{x}) \quad \longrightarrow \quad \frac{dI}{d\pi} = ?$$

A generalization of
Reynolds theorem

We now derive $\partial I_N / \partial \pi$ in Eq. (25) using the recursive relations provided by Eqs. (21) and (24). Let

$$h_n^{(0)} := [\prod_{n'=n+1}^N g(x_{n'}; x_{n'-2}, x_{n'-1})] W_e(x_N \rightarrow x_{N-1}), \quad (52)$$

$$h_n^{(1)} := \sum_{n'=n+1}^N \kappa(x_{n'}) V(x_{n'}), \quad (53)$$

$$\Delta h_{n,n'}^{(0)} := h_n^{(0)} \Delta g(x_{n'}; x_{n'-2}, x_{n'-1}) / g(x_{n'}; x_{n'-2}, x_{n'-1}), \quad (54)$$

for $0 \leq n < n' \leq N$. We omit the dependencies of $h_n^{(0)}$, $h_n^{(1)}$, and $\Delta h_{n,n'}^{(0)}$ on x_{n+1}, \dots, x_N for notational convenience.

We now show that, for all $0 \leq n < N$, it holds that

$$h_n(x_n; x_{n-1}) = \int_{\mathcal{M}^{N-n}} h_n^{(0)} \prod_{n'=n+1}^N dA(x_{n'}), \quad (55)$$

and

$$\begin{aligned} \dot{h}_n(x_n; x_{n-1}) &= \int_{\mathcal{M}^{N-n}} \left[\left(h_n^{(0)} \right)' - h_n^{(0)} h_n^{(1)} \right] \prod_{n'=n+1}^N dA(x_{n'}) \\ &\quad + \sum_{n'=n+1}^N \int \Delta h_{n,n'}^{(0)} V_{\overline{\partial \mathcal{M}_{n'}}}(x_{n'}) d\ell(x_{n'}) \prod_{\substack{n \leq i \leq N \\ i \neq n'}} dA(x_i), \end{aligned} \quad (56)$$

where the integral domain of the second term on the right-hand side, which is omitted for notational clarity, is $\mathcal{M}(\pi)$ for each x_i with $i \neq n'$ and $\overline{\partial \mathcal{M}_{n'}}(\pi)$, which depends on $x_{n'-1}$, for $x_{n'}$.

It is easy to verify that Eqs. (55) and (56) hold for $n = N - 1$. We now show that, if they hold for some $0 < n < N$, then it is also the case for $n - 1$. Let $g_{n-1} := g(x_n; x_{n-2}, x_{n-1})$ for all $0 < n \leq N$. Then,

$$\begin{aligned} h_{n-1}(x_{n-1}; x_{n-2}) &= \int_{\mathcal{M}} g_{n-1} \int_{\mathcal{M}^{N-n}} h_n^{(0)} \prod_{n'=n+1}^N dA(x_{n'}) dA(x_n) \\ &= \int_{\mathcal{M}^{N-n+1}} h_{n-1}^{(0)} \prod_{n'=n}^N dA(x_{n'}), \end{aligned} \quad (57)$$

and

$$\begin{aligned} \dot{h}_{n-1}(x_{n-1}; x_{n-2}) &= \int_{\mathcal{M}} \left[\dot{g}_{n-1} h_n^{(0)} + g_{n-1} \left(\dot{h}_n^{(0)} - h_n^{(0)} h_n^{(1)} \right) V(x_n) \right] dA(x_n) \\ &\quad + \int_{\overline{\partial \mathcal{M}_n}} \Delta g_{n-1} h_n V_{\overline{\partial \mathcal{M}_n}} d\ell(x_n) \\ &= \int_{\mathcal{M}^{N-n+1}} \left\{ \dot{g}_{n-1} h_n^{(0)} + g_{n-1} \left[\left(h_n^{(0)} \right)' - h_n^{(0)} h_{n-1}^{(1)} \right] \right\} \prod_{n'=n}^N dA(x_{n'}) \\ &\quad + \sum_{n'=n+1}^N \int g_{n-1} \Delta h_{n,n'}^{(0)} V_{\overline{\partial \mathcal{M}_{n'}}}(x_{n'}) d\ell(x_{n'}) \prod_{\substack{n \leq i \leq N \\ i \neq n'}} dA(x_i) \\ &\quad + \int \Delta g_{n-1} h_n^{(0)} V_{\overline{\partial \mathcal{M}_n}} d\ell(x_n) \prod_{n'=n+1}^N dA(x_{n'}) \\ &= \int_{\mathcal{M}^{N-n+1}} \left[\left(h_{n-1}^{(0)} \right)' - h_{n-1}^{(0)} h_{n-1}^{(1)} \right] \prod_{n'=n}^N dA(x_{n'}) \\ &\quad + \sum_{n'=n}^N \int \Delta h_{n-1,n'}^{(0)} V_{\overline{\partial \mathcal{M}_{n'}}}(x_{n'}) d\ell(x_{n'}) \prod_{\substack{n \leq i \leq N \\ i \neq n'}} dA(x_i). \end{aligned} \quad (58)$$

Thus, using mathematical induction, we know that Eqs. (55) and (56) hold for all $0 \leq n < N$.

Notice that $h_0^{(0)} = f$ and $\Delta h_{0,n'}^{(0)} = \Delta f_{n'}$, where $\Delta f_{n'}$ follows the definition in Eq. (28). Letting $n = 0$ in Eq. (56) yields

$$\begin{aligned} \dot{h}_0(x_0) &= \int_{\mathcal{M}^N} \left[\dot{f}(\bar{x}) - f(\bar{x}) \sum_{n'=1}^N \kappa(x_{n'}) V(x_{n'}) \right] \prod_{n'=1}^N dA(x_{n'}) \\ &\quad + \sum_{n'=1}^N \int \Delta f_{n'}(\bar{x}) V_{\overline{\partial \mathcal{M}_{n'}}} d\ell(x_{n'}) \prod_{\substack{0 \leq i \leq N \\ i \neq n'}} dA(x_i). \end{aligned} \quad (59)$$

Lastly, based on the assumption that h_0 is continuous in x_0 , Eq. (25) can be obtained by differentiating Eq. (23):

$$\begin{aligned} \frac{\partial I_N}{\partial \pi} &= \frac{\partial}{\partial \pi} \int_{\mathcal{M}} h_0(x_0) dA(x_0) \\ &= \int_{\mathcal{M}} [h_0(x_0) - h_0(x_0) \kappa(x_0) V(x_0)] dA(x_0) \\ &\quad + \int_{\overline{\partial \mathcal{M}_0}} h_0(x_0) V_{\overline{\partial \mathcal{M}_0}}(x_0) d\ell(x_0) \\ &= \int_{\Omega_N} [\dot{f}(\bar{x}) - f(\bar{x}) \sum_{K=0}^N \kappa(x_K) V(x_K)] d\mu(\bar{x}) \\ &\quad + \sum_{K=0}^N \int_{\Omega_{N,K}} \Delta f_K(\bar{x}) V_{\overline{\partial \mathcal{M}_K}} d\mu'_{N,K}(\bar{x}). \end{aligned} \quad (60)$$

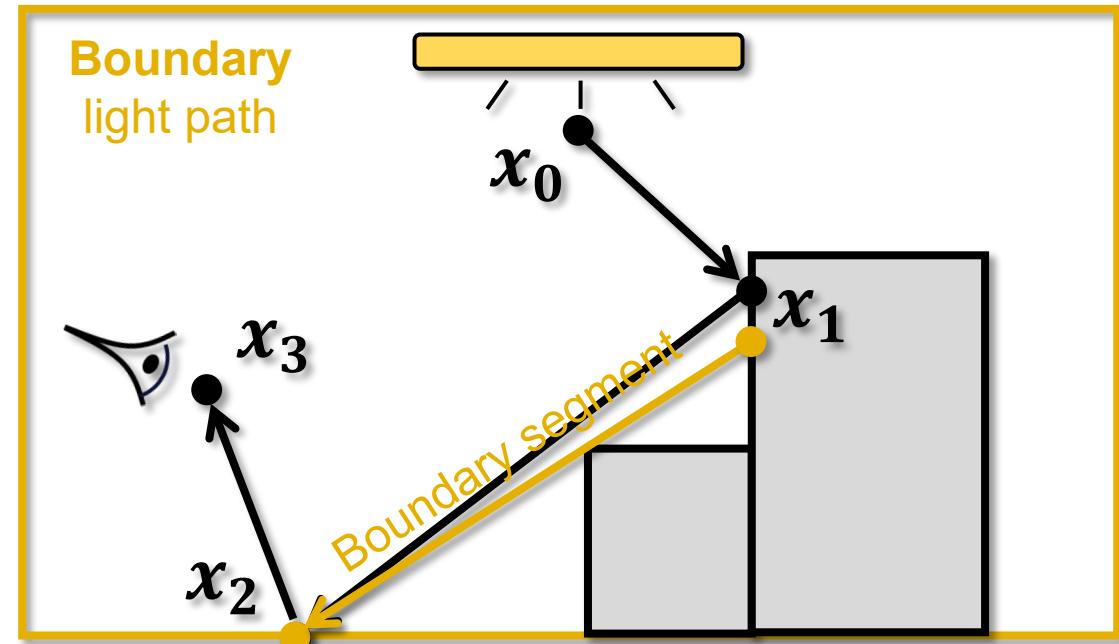
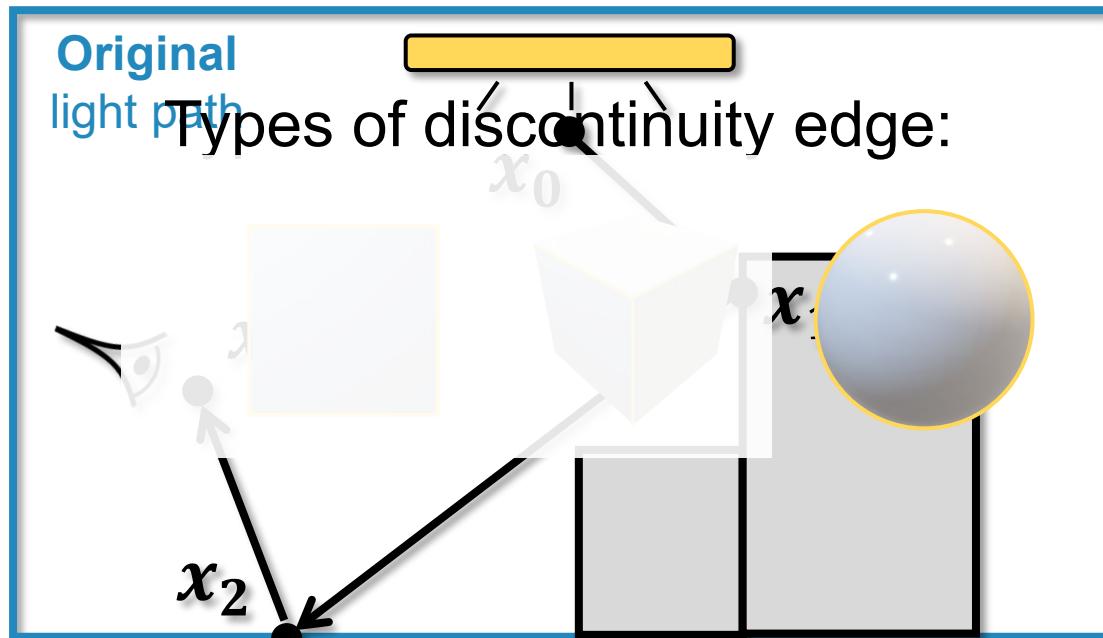
Full derivation in the paper

DIFFERENTIAL PATH INTEGRAL

Path Integral A generalization of Reynolds theorem Differential Path Integral

$$I = \int_{\Omega} f(\bar{x}) d\mu(\bar{x}) \quad \xrightarrow{\hspace{1cm}} \quad \frac{dI}{d\pi} = \int_{\Omega} \frac{d}{d\pi} f(\bar{x}) d\mu(\bar{x}) + \int_{\partial\Omega} g(\bar{x}) d\mu'(\bar{x})$$

path space Interboundary path space Boundary integral

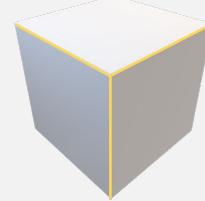


SOURCE OF DISCONTINUITIES

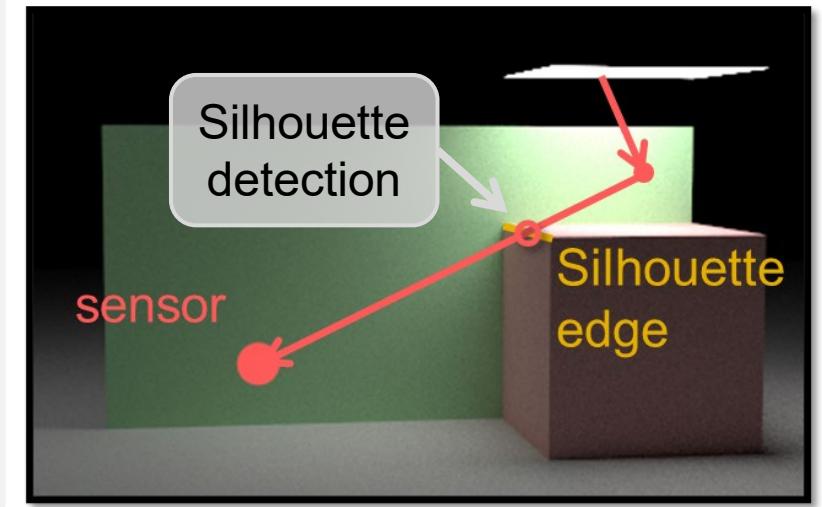
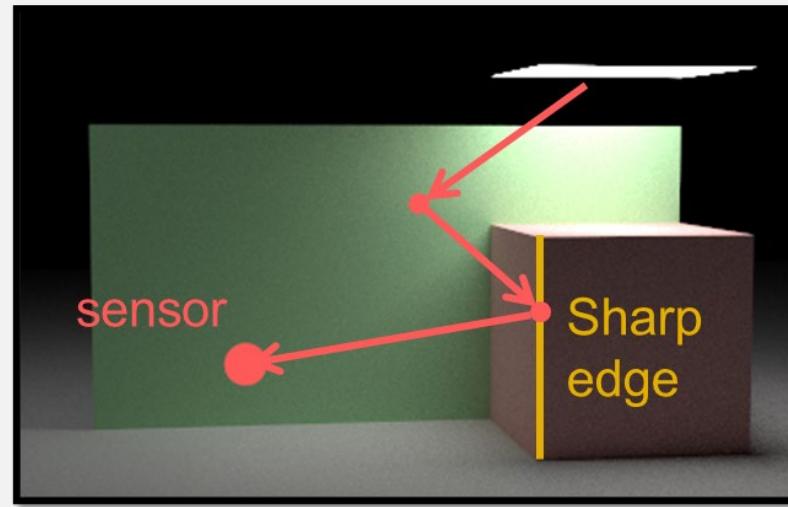
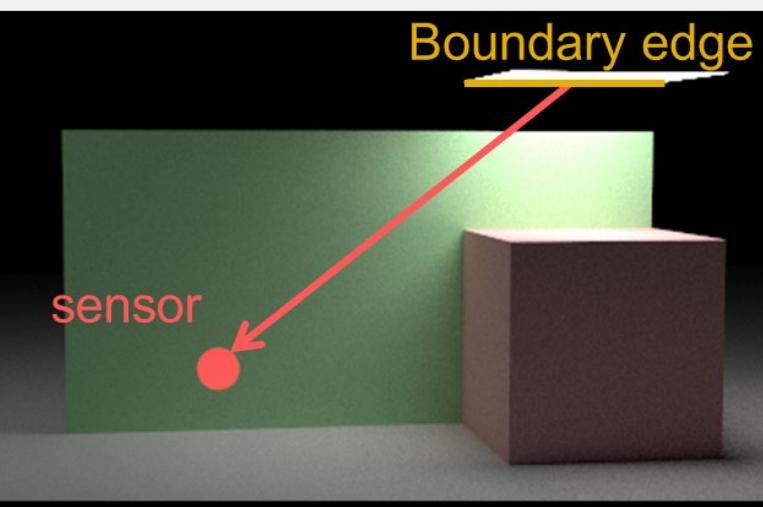
Boundary edge



Sharp edge



Silhouette edge



Topology-driven

Visibility-driven

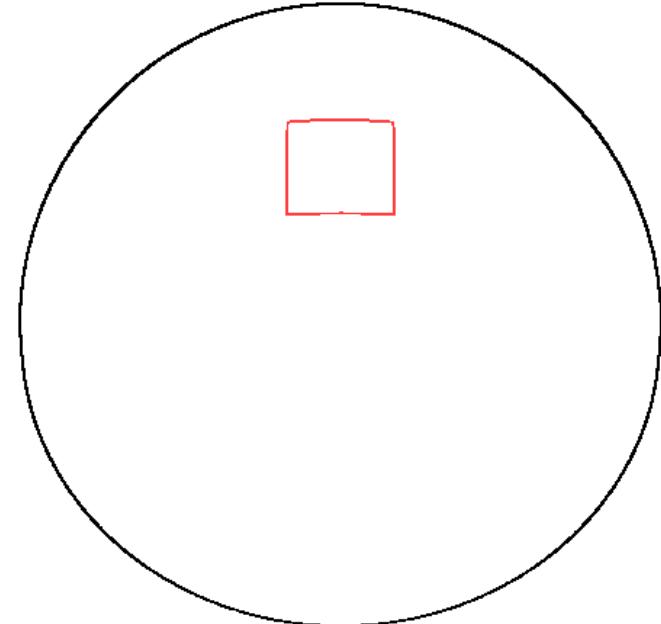
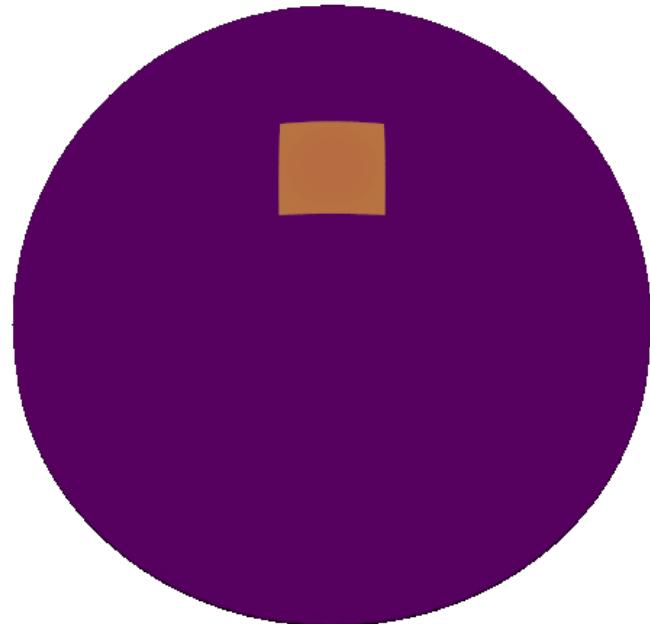
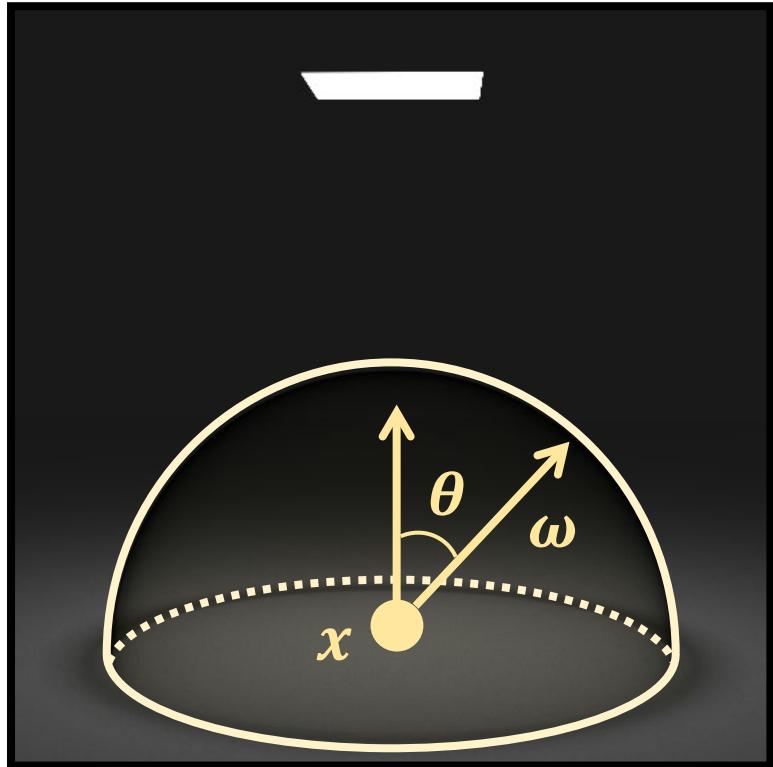
REPARAMETERIZATIONS FOR SIMPLIFYING THE BOUNDARY TERM

REVISIT - DIFFERENTIAL IRRADIANCE

π : size of the emitter

Low High

Discontinuities of f



$$E = \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

f

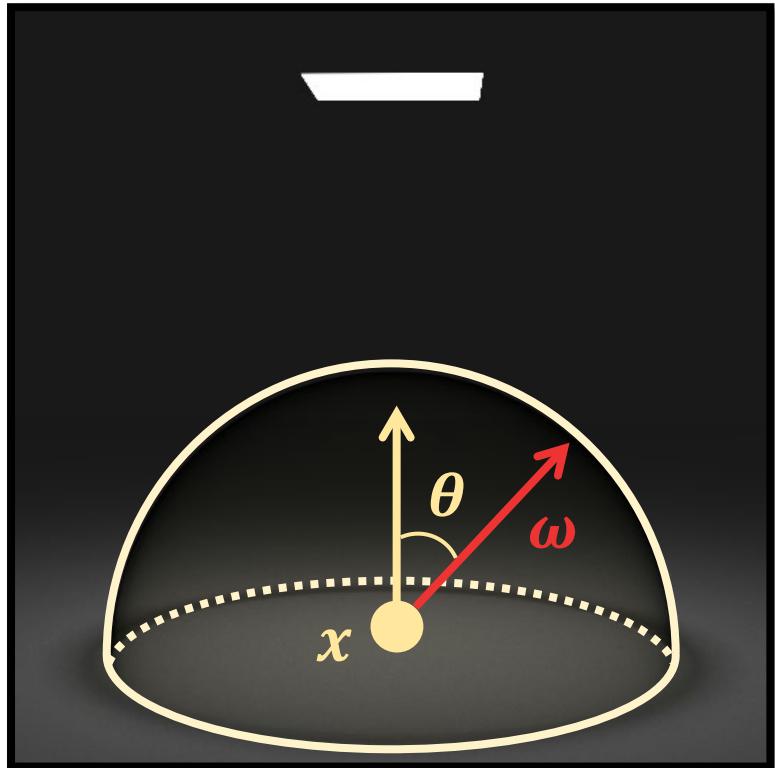
Differentiation

$$\frac{dE}{d\pi} = \int_{\mathbb{H}^2} \frac{df}{d\pi} d\sigma + \int_{\partial\mathbb{H}^2} g \, dl$$

$= 0$ $\neq 0$

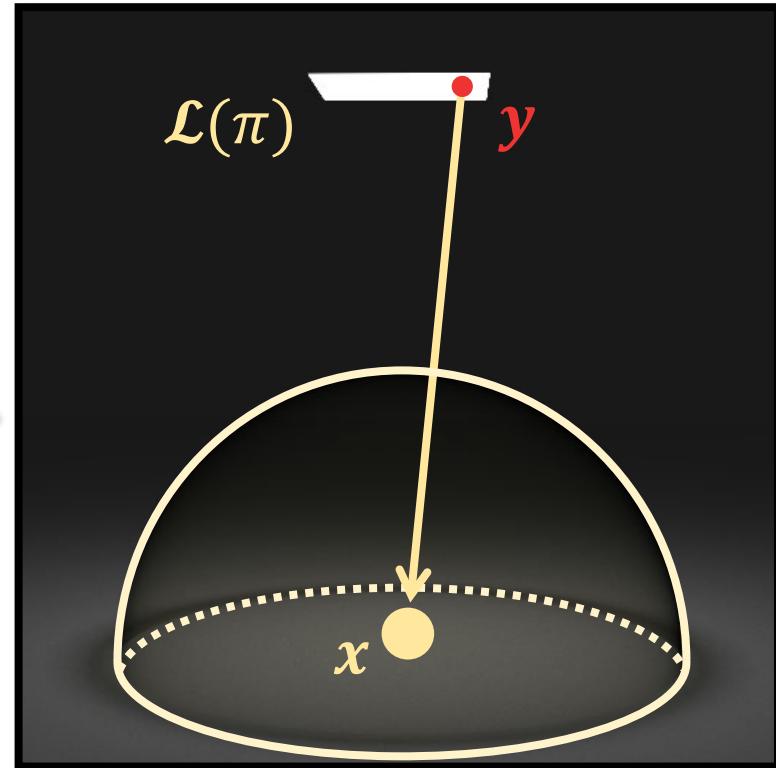
DIFFERENTIAL IRRADIANCE

Spherical integral



Change of variable

Surface integral

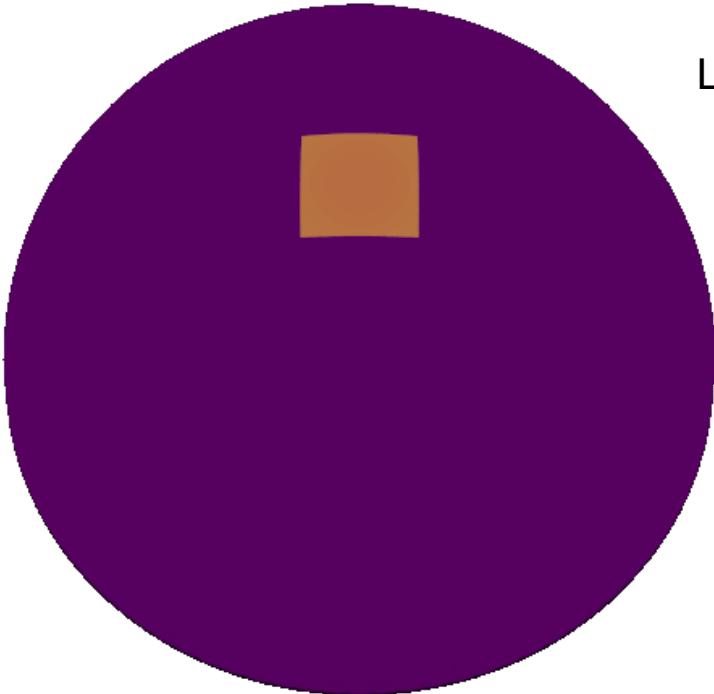


$$E = \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

$$E = \int_{\mathcal{L}(\pi)} L_e(y \rightarrow x) G(x, y) \, dA(y)$$

DIFFERENTIAL DIRECT ILLUMINATION

Spherical integral



Low High

Surface integral



Change of variable

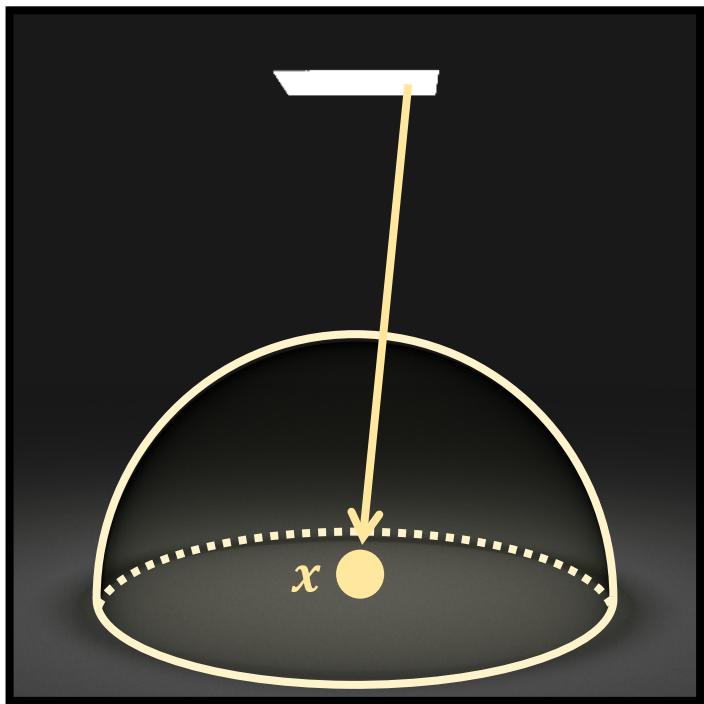
$$E = \int_{\mathbb{H}^2} L_i(\omega) \cos\theta \, d\sigma(\omega)$$

discontinuous
constant domain

$$E = \int_{\mathcal{L}(\pi)} L_e(y \rightarrow x) G(x, y) \, dA(y)$$

continuous
evolving domain

DIFFERENTIAL IRRADIANCE



Low High

Boundary of $\mathcal{L}(\pi)$

$$E = \int_{\mathcal{L}(\pi)} \overbrace{L_e(y \rightarrow x) G(x, y)}^f dA(y)$$



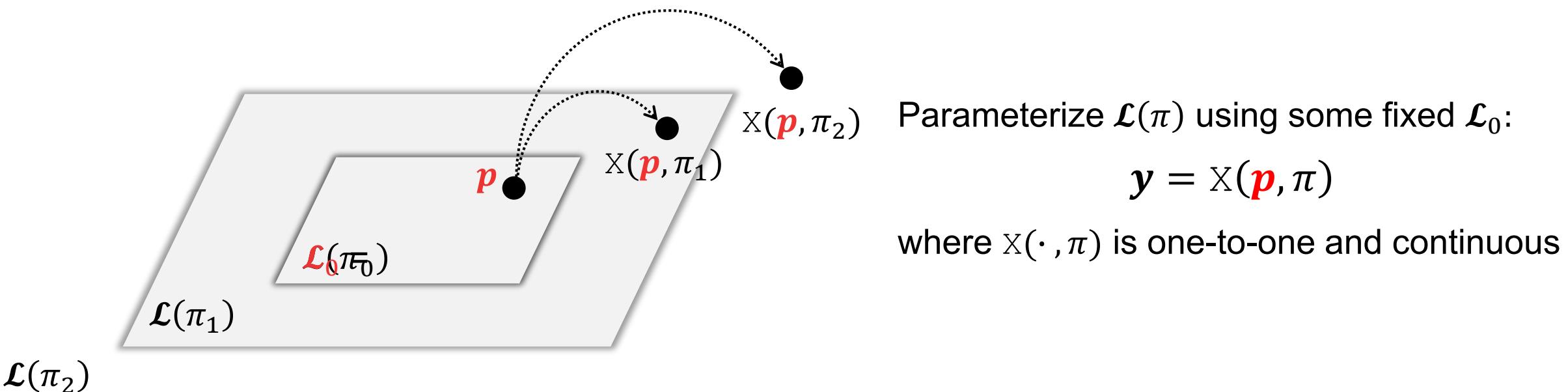
Differentiation
Reynolds theorem

$$\frac{dE}{d\pi} = \int_{\mathcal{L}(\pi)} \frac{df}{d\pi} dA + \int_{\partial \mathcal{L}(\pi)} g dl$$

$\neq 0$

REPARAMETERIZATION

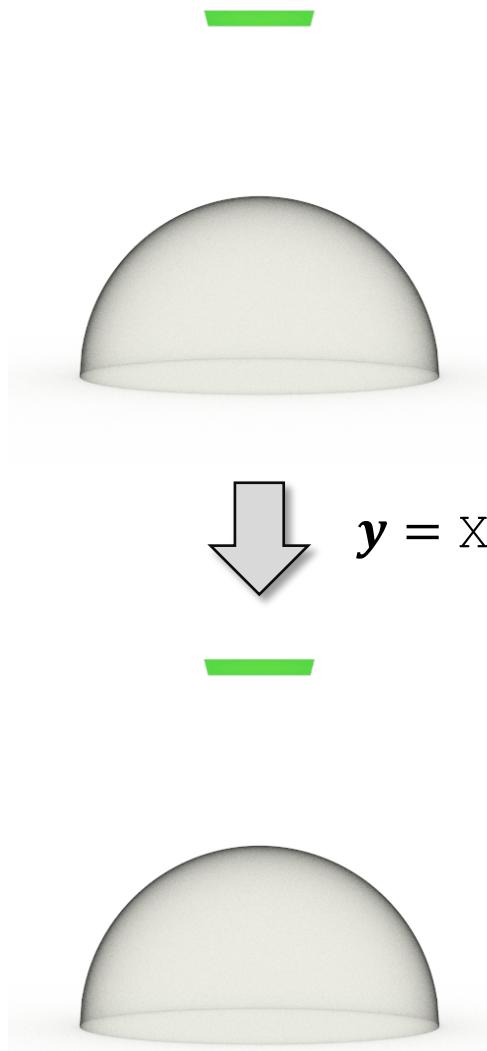
$$E = \int_{\mathcal{L}(\pi)} L_e(y \rightarrow x) G(x, y) dA(y)$$



Reparameterization with $y = x(\mathbf{p}, \pi)$:

$$E = \int_{\mathcal{L}_0} L_e(y \rightarrow x) G(x, y) \left| \frac{dA(y)}{dA(\mathbf{p})} \right| dA(\mathbf{p})$$

REPARAMETERIZATION



$$E = \int_{\mathcal{L}(\pi)} \overbrace{L_e(\mathbf{y} \rightarrow \mathbf{x}) G(\mathbf{x}, \mathbf{y})}^f \mathrm{d}A(\mathbf{y})$$

$$\frac{\mathrm{d}E}{\mathrm{d}\pi} = \int_{\mathcal{L}(\pi)} \frac{\mathrm{d}f}{\mathrm{d}\pi} \mathrm{d}A + \int_{\partial\mathcal{L}(\pi)} g \mathrm{d}l$$

$$= 0 \qquad \qquad \neq 0$$

$$E = \int_{\mathcal{L}_0} L_e(\mathbf{y} \rightarrow \mathbf{x}) G(\mathbf{x}, \mathbf{y}) \underbrace{\left| \frac{\mathrm{d}A(\mathbf{y})}{\mathrm{d}A(\mathbf{p})} \right|}_{f_0} \mathrm{d}A(\mathbf{p})$$

$$\frac{\mathrm{d}E}{\mathrm{d}\pi} = \int_{\mathcal{L}_0} \frac{\mathrm{d}f_0}{\mathrm{d}\pi} \mathrm{d}A + \int_{\partial\mathcal{L}_0} g_0 \mathrm{d}l$$

$$\neq 0 \qquad \qquad = 0$$

REPARAMETERIZATION

Reparameterization for irradiance

$$E = \int_{\mathcal{L}(\pi)} L_e(\mathbf{y} \rightarrow \mathbf{x}) G(\mathbf{x}, \mathbf{y}) dA(\mathbf{y})$$

$$\mathbf{y} = \mathbf{x}(\mathbf{p}, \boldsymbol{\pi})$$



$$E = \int_{\mathcal{L}_0} L_e(\mathbf{y} \rightarrow \mathbf{x}) G(\mathbf{x}, \mathbf{y}) \left| \frac{dA(\mathbf{y})}{dA(\mathbf{p})} \right| dA(\mathbf{p})$$

↑
Fixed surface

Reparameterization for path integral

$$I = \int_{\Omega(\pi)} f(\bar{\mathbf{x}}) d\mu(\bar{\mathbf{x}})$$

$$\bar{\mathbf{x}} = \mathbf{x}(\bar{\mathbf{p}}, \boldsymbol{\pi})$$



$$I = \int_{\Omega_0} f(\bar{\mathbf{x}}) \left| \frac{d\mu(\bar{\mathbf{x}})}{d\mu(\bar{\mathbf{p}})} \right| d\mu(\bar{\mathbf{p}})$$

↑
Fixed path space

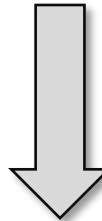
$$\prod_i \left| \frac{dA(\mathbf{x}_i)}{dA(\mathbf{p}_i)} \right|$$

||

DIFFERENTIAL PATH INTEGRAL

Original

$$I = \int_{\Omega(\pi)} f(\bar{x}) d\mu(\bar{x})$$



$$\bar{x} = x(\bar{p}, \pi)$$

Reparameterized

$$I = \int_{\Omega_0} f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| d\mu(\bar{p})$$

Original

$$\frac{dI}{d\pi} = \int_{\Omega(\pi)} \frac{df(\bar{x})}{d\pi} d\mu(\bar{x}) + \int_{\partial\Omega(\pi)} g(\bar{x}) d\mu'(\bar{x})$$

Pro: No global parametrization required

Con: More types of discontinuities

Reparameterized

$$\frac{dI}{d\pi} = \int_{\Omega_0} \frac{d}{d\pi} \left(f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| \right) d\mu(\bar{p}) + \int_{\partial\Omega_0} g(\bar{p}) d\mu'(\bar{p})$$

Con: Requires global parametrization x

Pro: Fewer types of discontinuities

DIFFERENTIAL PATH INTEGRAL

Differential path integral

$$\frac{dI}{d\pi} = \int_{\Omega(\pi)} \frac{df(\bar{x})}{d\pi} d\mu(\bar{x}) + \int_{\partial\Omega(\pi)} g(\bar{x}) d\mu'(\bar{x})$$

$$\frac{dI}{d\pi} = \int_{\Omega_0} \frac{d}{d\pi} \left(f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| \right) d\mu(\bar{p}) + \int_{\partial\Omega_0} g(\bar{p}) d\mu'(\bar{p})$$

Topology-driven

Boundary edge

sensor

sensor

Sharp edge

Visibility-driven

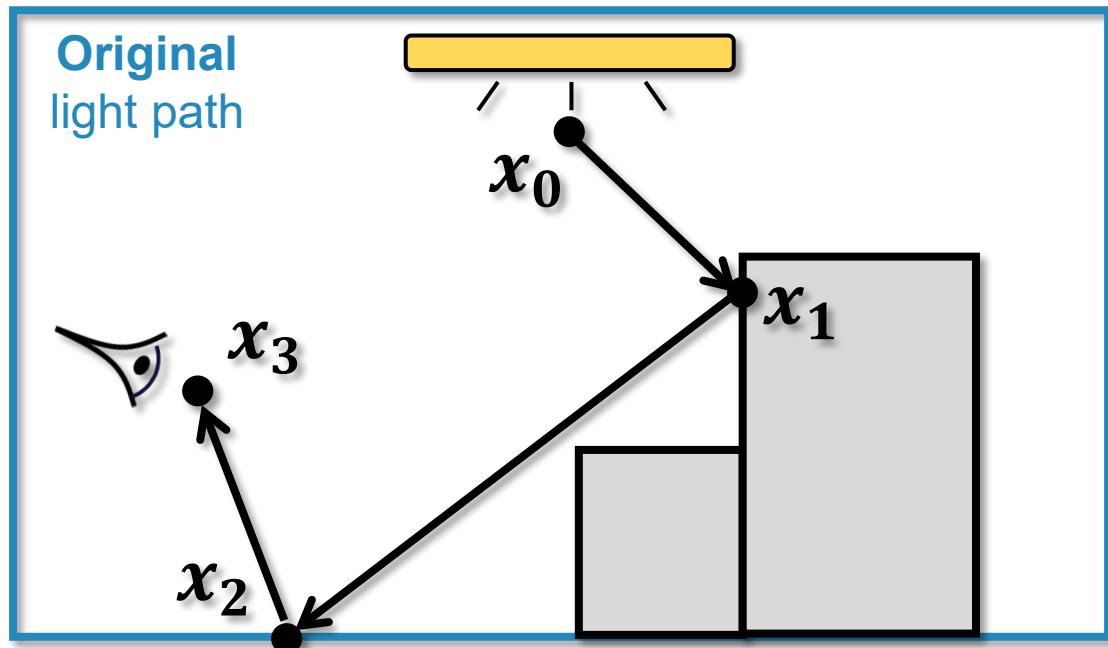
Silhouette edge

MONTE CARLO ESTIMATORS

ESTIMATING INTERIOR INTEGRAL

(Reparameterized)
Differential path Integral

$$\frac{\partial I}{\partial \pi} = \int_{\Omega_0} \frac{\partial}{\partial \pi} \left(f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| \right) d\mu(\bar{p}) + \int_{\partial\Omega_0} g(\bar{p}) d\mu'(\bar{p})$$



Interior integral

Boundary integral

- Can be estimated using identical path sampling strategies as forward rendering
 - Unidirectional path tracing
 - Bidirectional path tracing
 - ...

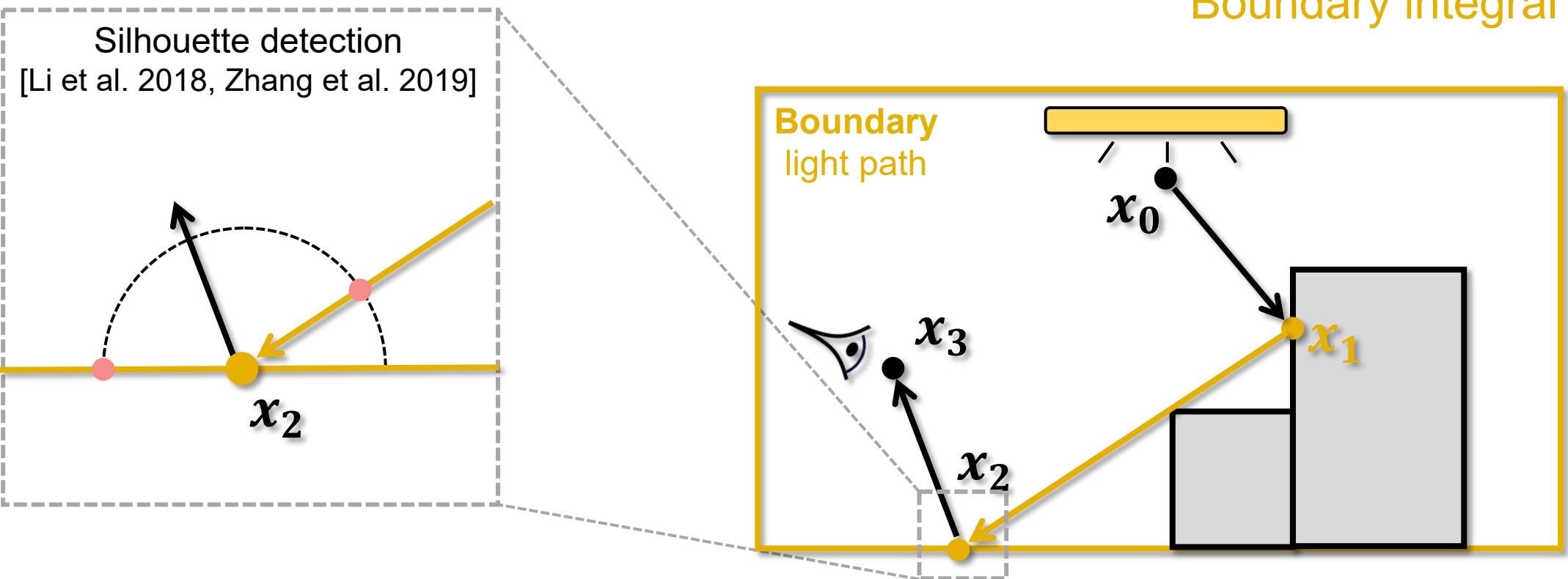
Different MC estimators

ESTIMATING BOUNDARY INTEGRAL

(Reparameterized)
Differential path Integral

$$\frac{\partial I}{\partial \pi} = \int_{\Omega_0} \frac{\partial}{\partial \pi} \left(f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| \right) d\mu(\bar{p}) + \int_{\partial\Omega_0} g(\bar{p}) d\mu'(\bar{p})$$

Boundary integral



ESTIMATING BOUNDARY INTEGRAL

(Reparameterized)
Differential path Integral

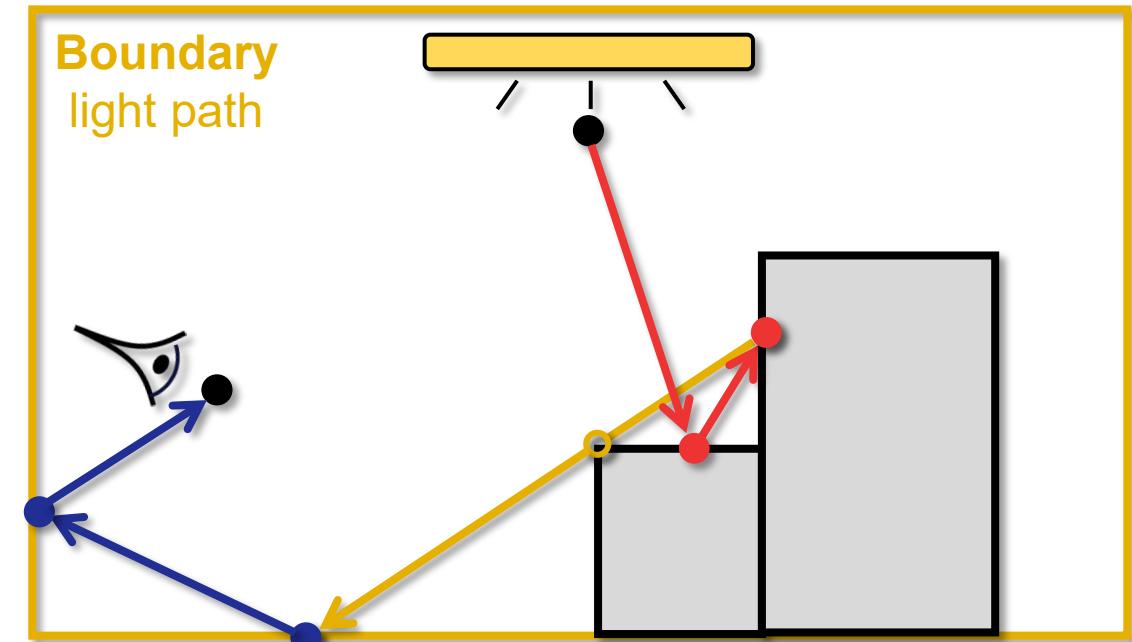
$$\frac{\partial I}{\partial \pi} = \int_{\Omega_0} \frac{\partial}{\partial \pi} \left(f(\bar{x}) \left| \frac{d\mu(\bar{x})}{d\mu(\bar{p})} \right| \right) d\mu(\bar{p}) + \int_{\partial\Omega_0} g(\bar{p}) d\mu'(\bar{p})$$

where $\bar{x} = x(\bar{p}, \pi)$

Boundary integral

- Construct boundary segment
- Construct source and sensor subpaths

- To improve efficiency
 - Next-event estimation
 - Importance sampling of boundary segments

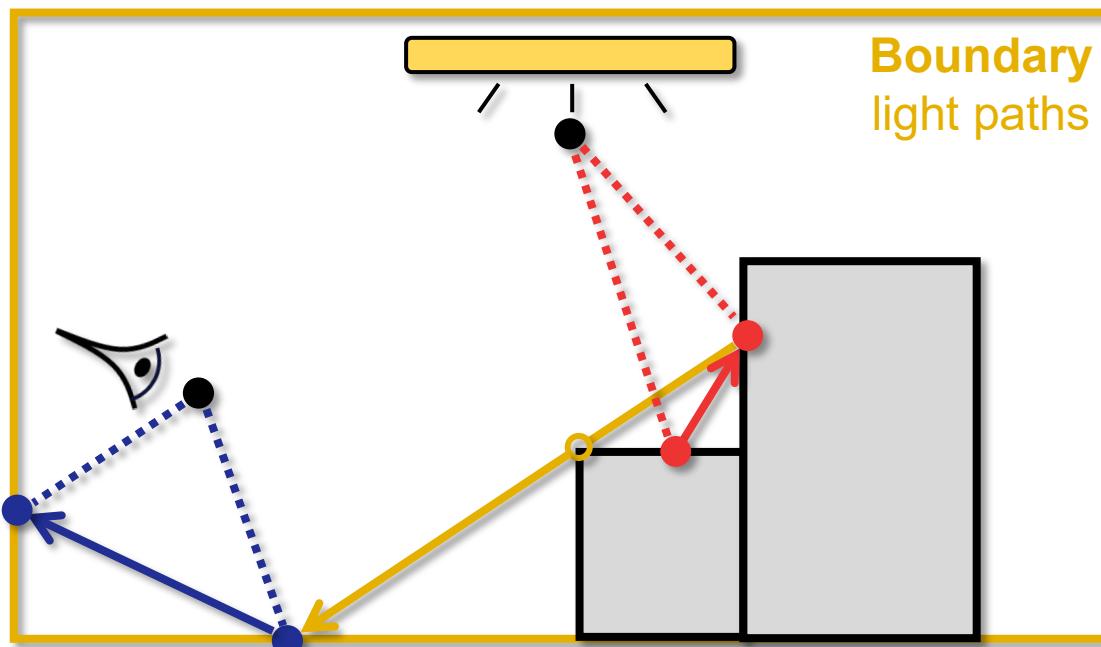


OUR ESTIMATORS

Unidirectional estimator

Interior: unidirectional path tracing

Boundary: unidirectional sampling of subpaths

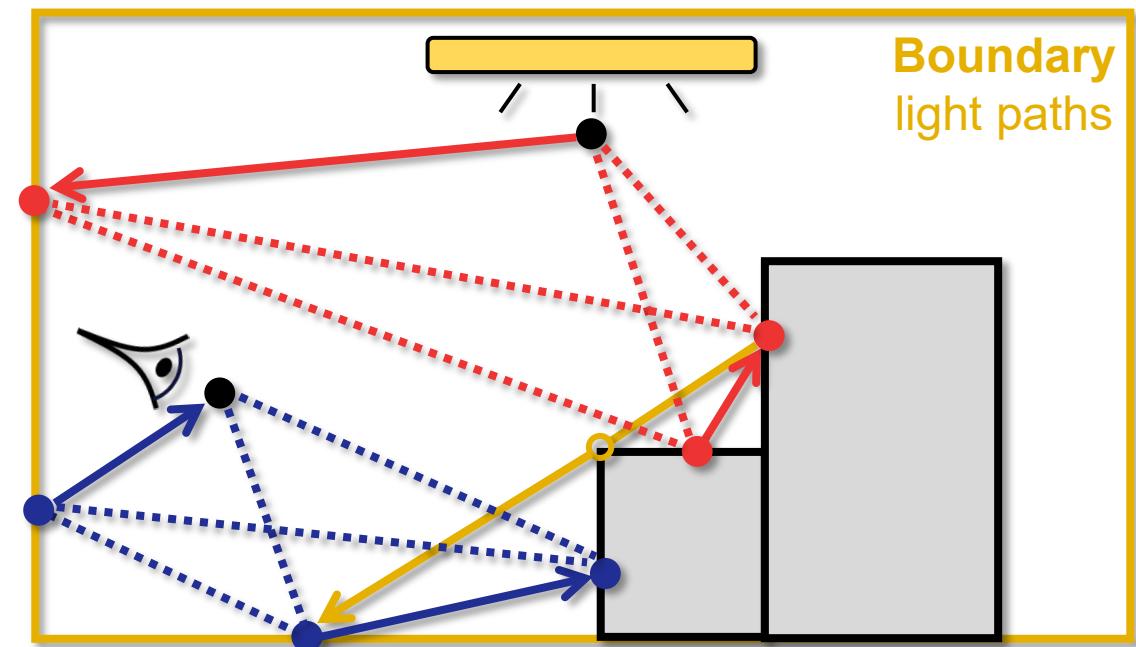


Unidirectional path tracing + NEE

Bidirectional estimator

Interior: bidirectional path tracing

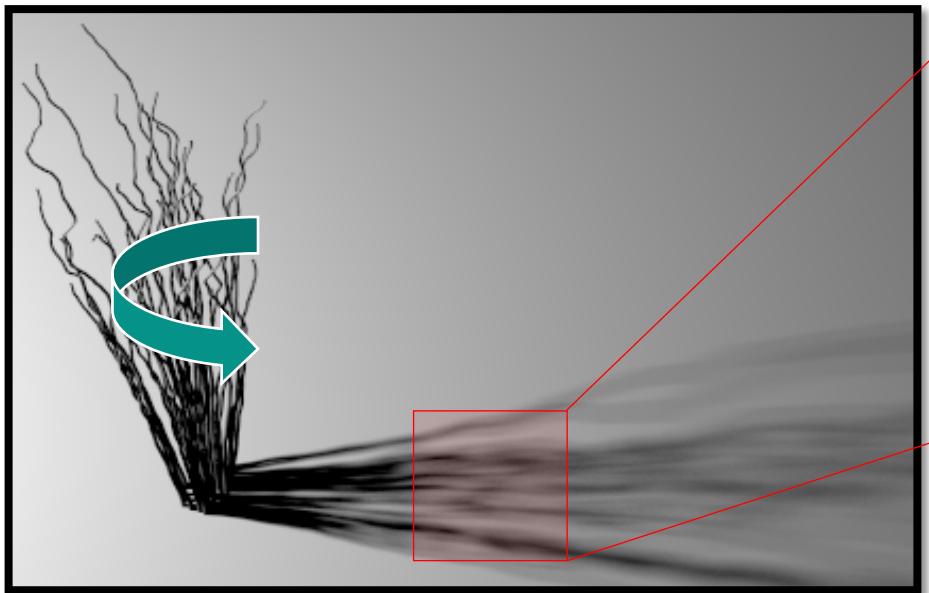
Boundary: bidirectional sampling of subpaths



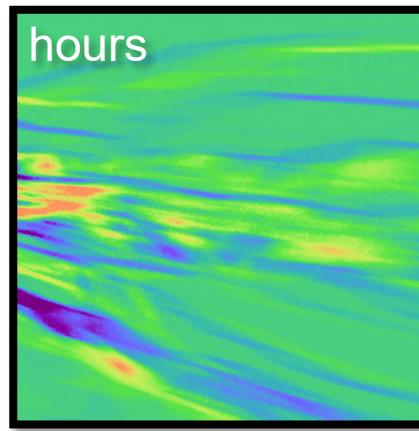
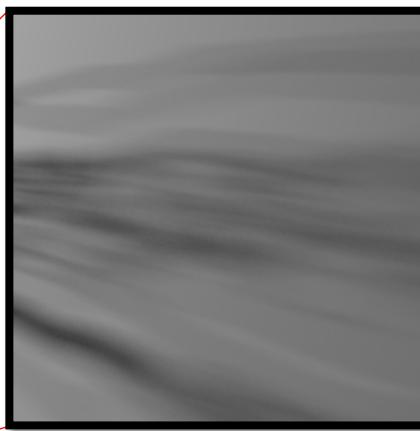
Bidirectional path tracing

SOME RESULTS

HANDLING COMPLEX GEOMETRY



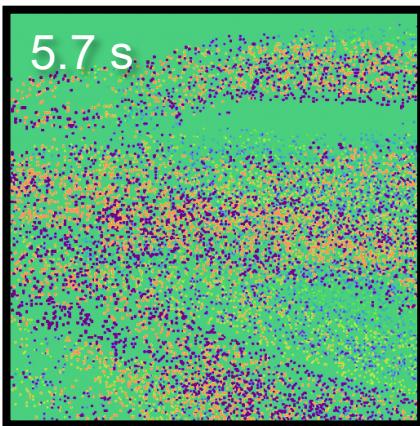
Complex geometry



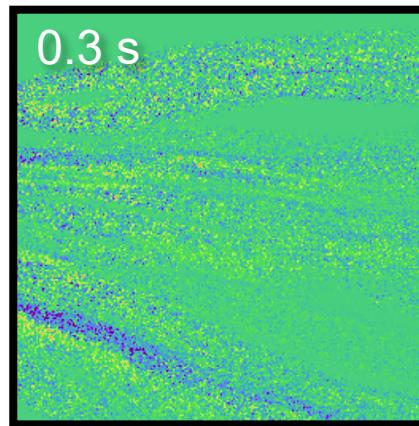
Reference

Negative Zero Positive

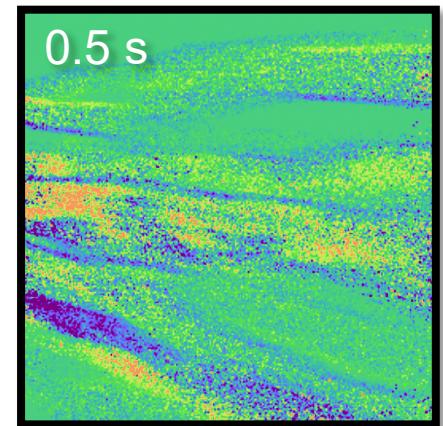
Equal-sample
comparison



[Zhang et al. 2019]



[Loubet et al. 2019]



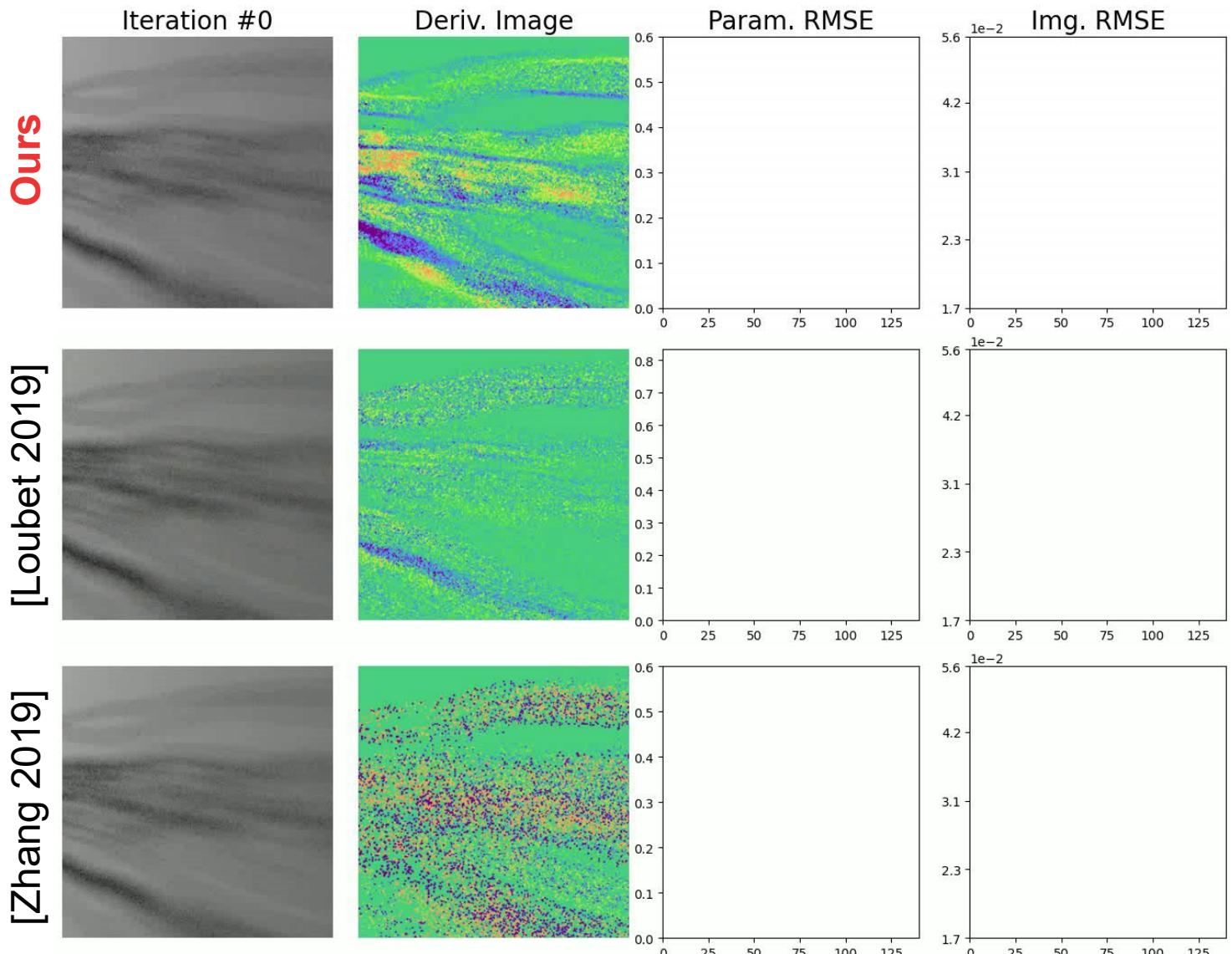
Ours

HANDLING COMPLEX GEOMETRY

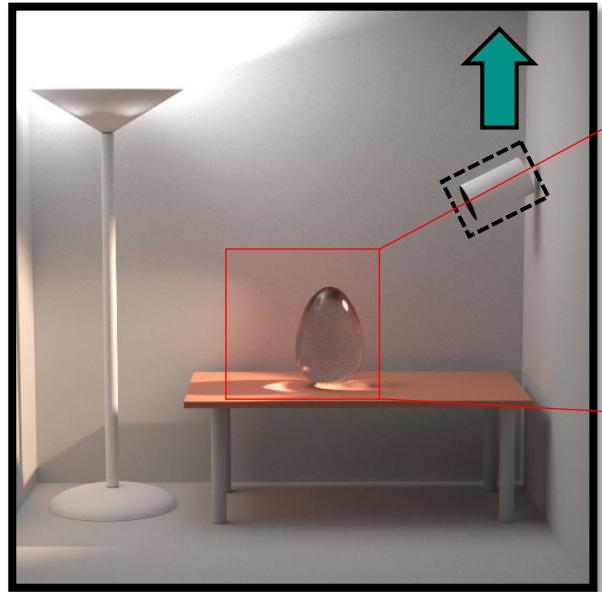
Target image



- Optimizing *rotation angle*
- **Equal-sample** per iteration
- **Identical** optimization setting
 - Learning rate (Adam)
 - Initializations

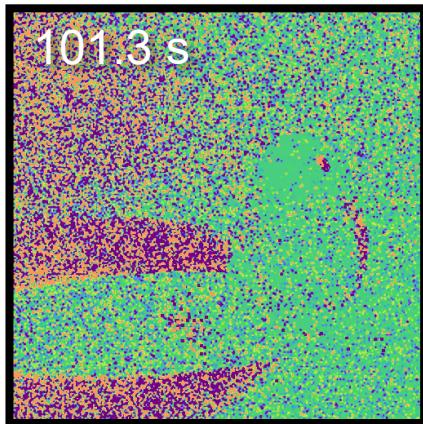
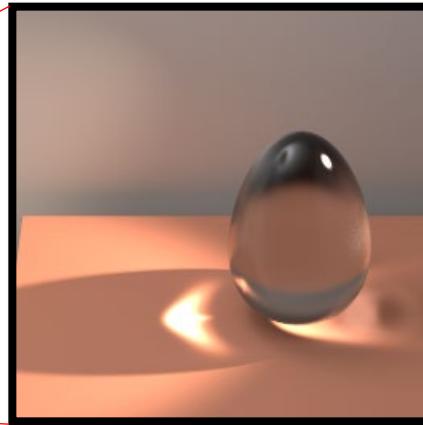


HANDLING CAUSTICS

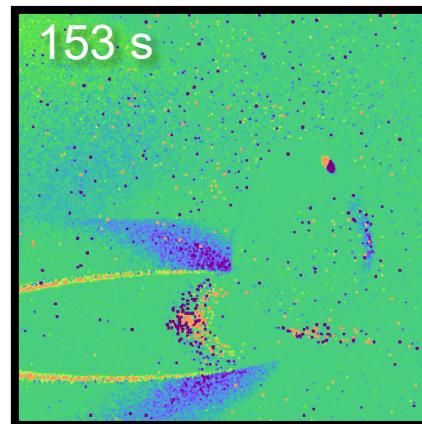


Complex light transport effects

Equal-sample
comparison



[Zhang et al. 2019]



[Loubet et al. 2019]

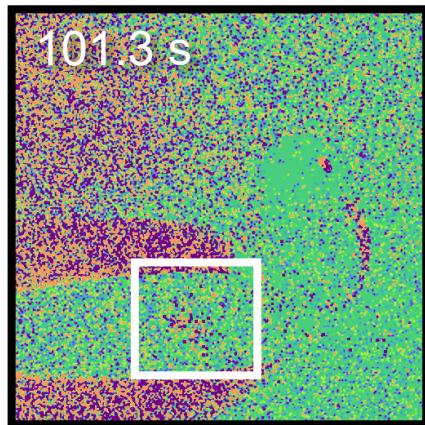
Reference

Negative Zero Positive

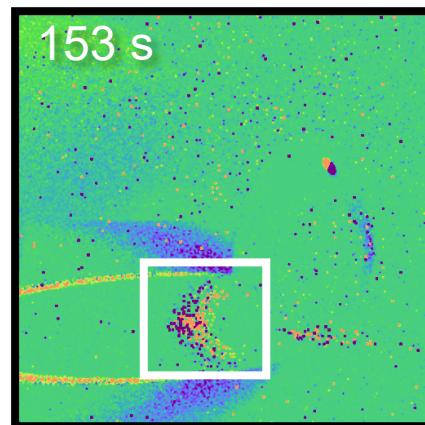
Ours

HANDLING CAUSTICS

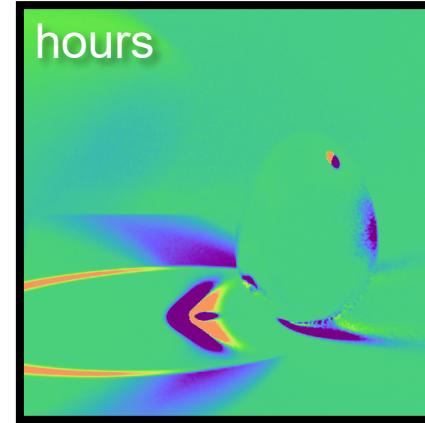
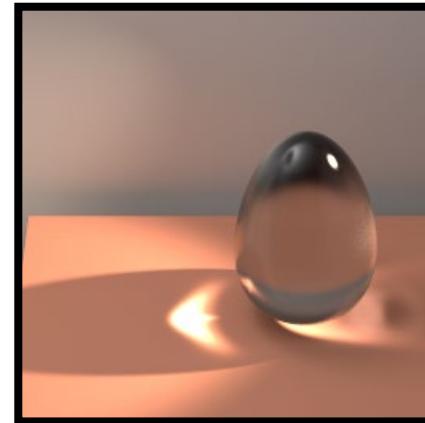
Equal-sample comparison



[Zhang et al. 2019]

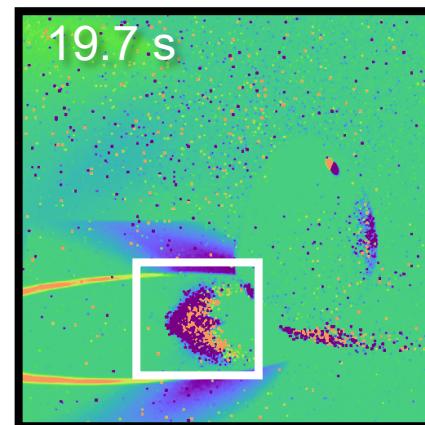


[Loubet et al. 2019]

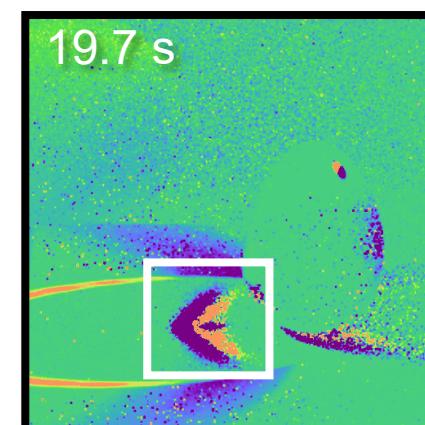


Reference

Negative Positive



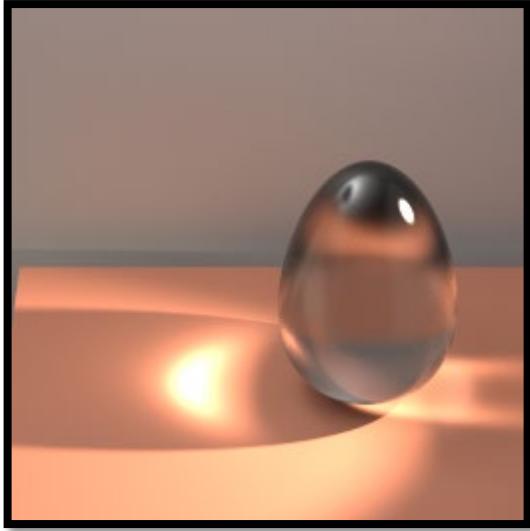
Ours (unidirectional)



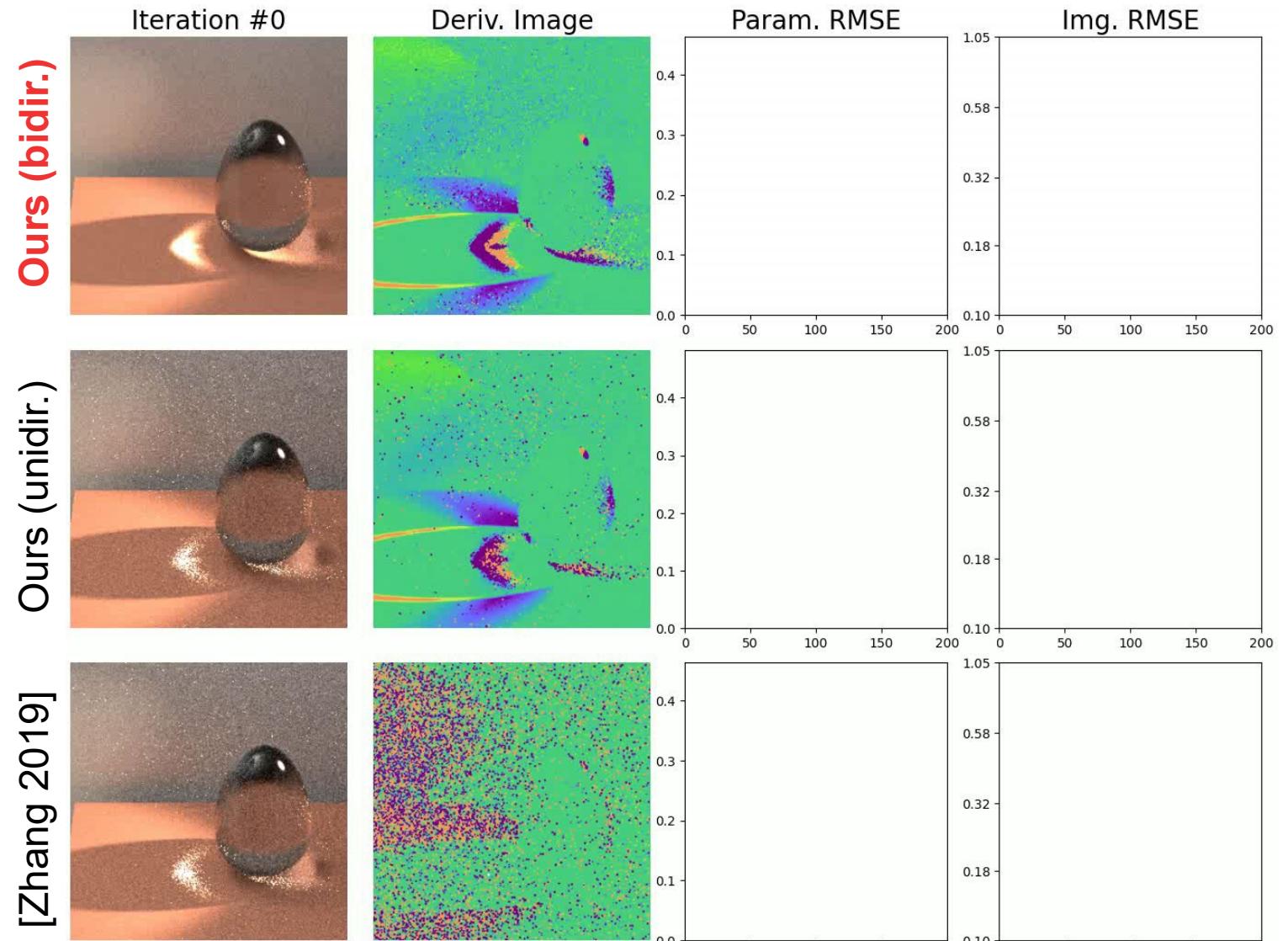
Ours (bidirectional)

HANDLING CAUSTICS

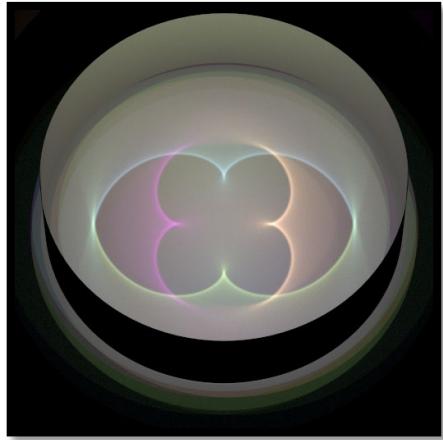
Target image



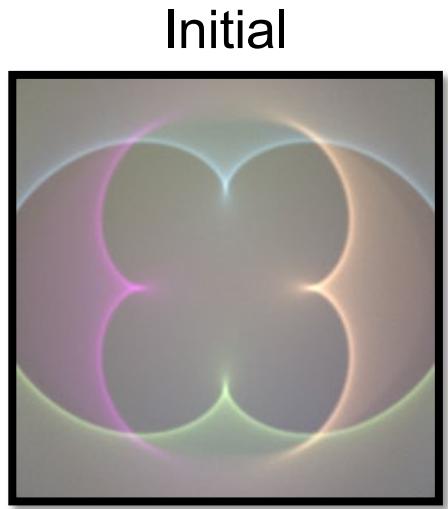
- Optimizing
 - Glass IOR
 - Spotlight position
- **Equal-time** per iteration
- **Identical** optimization setting



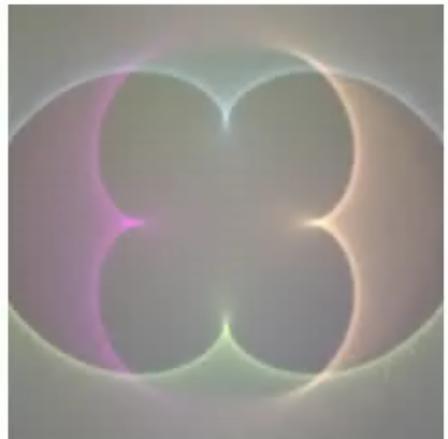
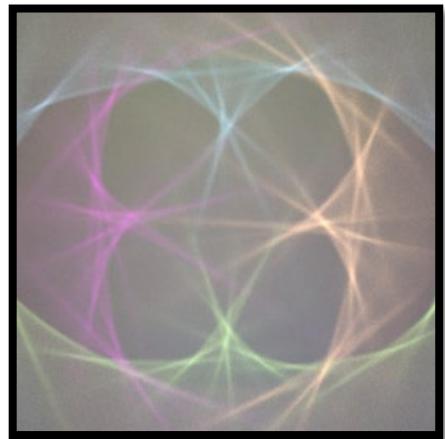
SHAPE OPTIMIZATION



Target image

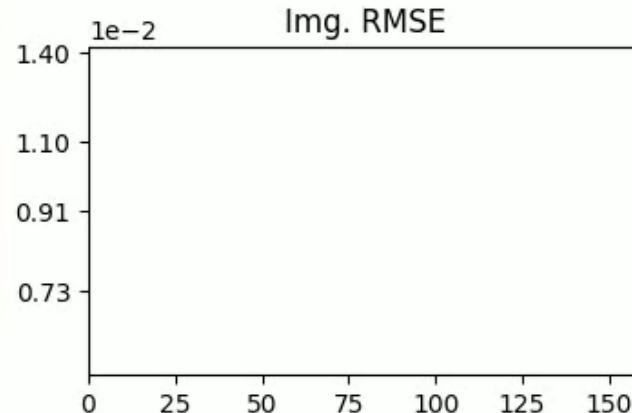


Initial

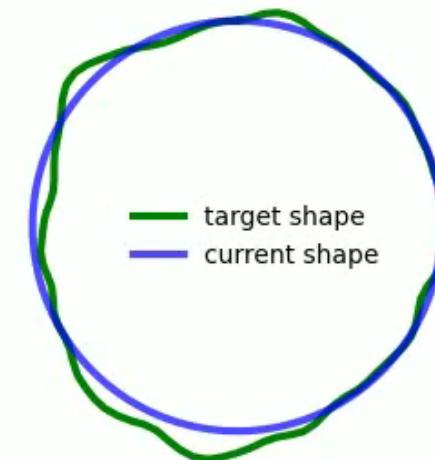


Iter #0

Optimizing **cross-sectional** shape (100 variables)

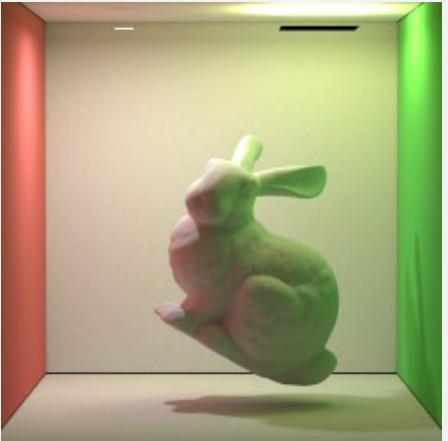


Cross-sectional shape
(displacement x 20)

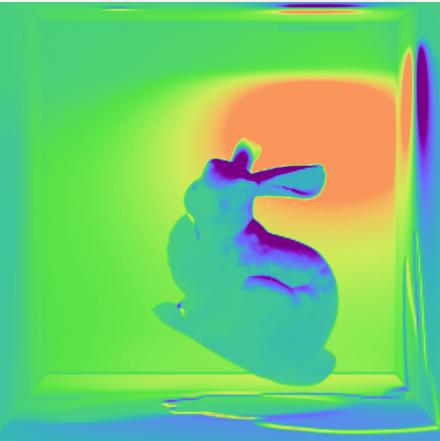


RESULTS

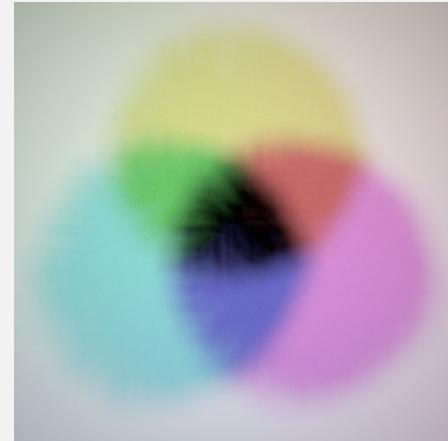
Original image



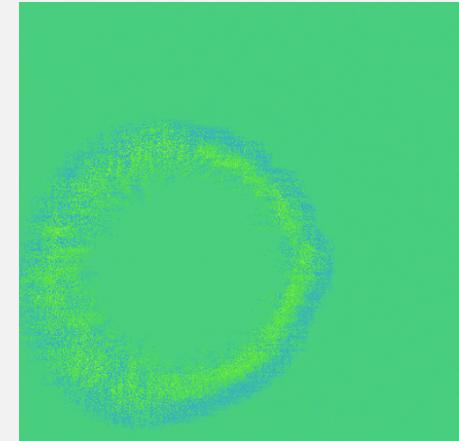
Derivative image



Original image



Derivative image



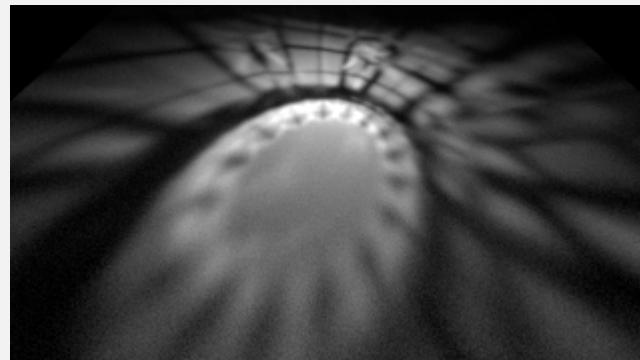
Config.



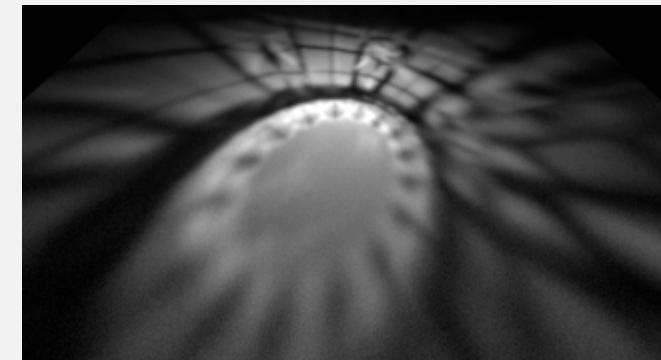
Optimize (initial)



Optimize (final)



Target



Stuff we are missing

We need path sampling algorithms tailored to differentiable rendering:

- Some simple versions exist for local differentiation (Gkioulekas et al. 2013, 2016).
- We need to take into account diff. geometric quantities in global case.
- We need to take into account loss function.

We need theory that can handle very low-dimensional path manifolds:

- We can't easily incorporate specular and refractive effects into arbitrary pipelines.
- Doable in isolation (Chen and Arvo 2000, Jakob and Marschner 2013, Xin et al. 2019).

Some more general thoughts

Initialization is super important:

- Approximate reconstruction assuming direct lighting is usually good enough.
- Coarse-to-fine schemes work well.

Parameterizations are super important:

- Loss functions very non-linear and change shape easily.
- Working with meshes is a pain (topology is awful and not (easily?) differentiable).

Parameterization matters

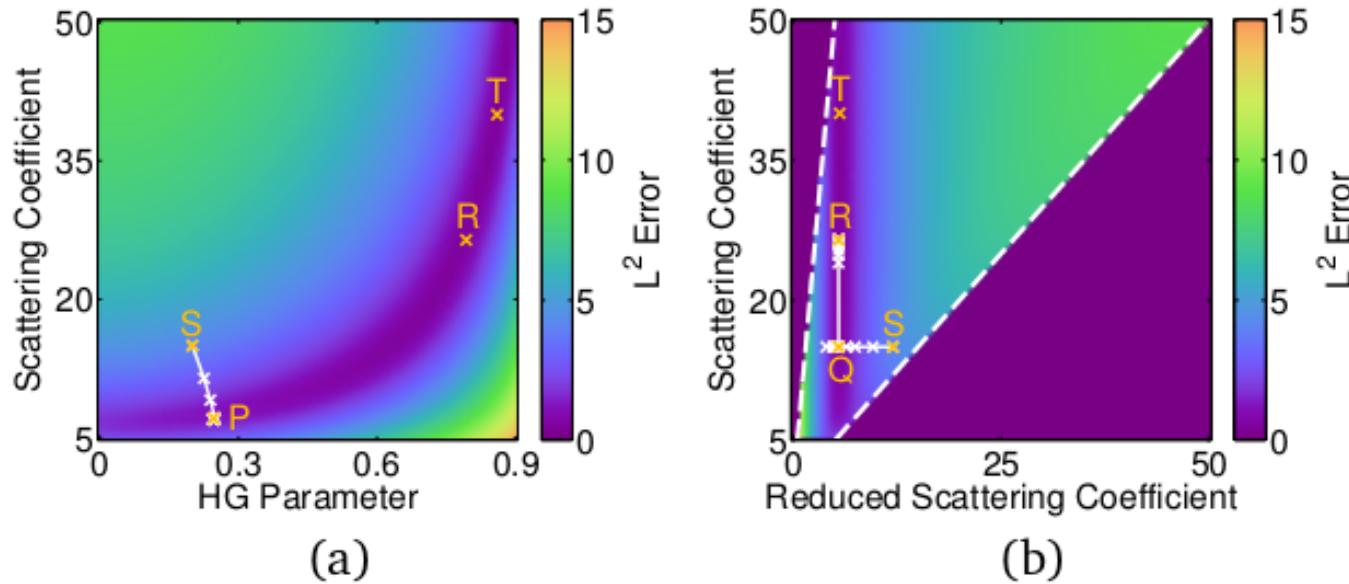


Figure 4: Search spaces for an inverse rendering problem: (a) the original space; (b) the reparameterized space. The plotted region in (a) maps to the area enclosed by the dashed lines in (b). Using the original space, the stochastic gradient descent (SGD) algorithm starting from point S is trapped at point P , which is far from the real solution T . Using the reparameterized space, the algorithm is able to find point R that is much closer to the real solution.

volumetric density σ_t
scattering albedo a
phase function f_r

Some more general thoughts

Initialization is super important:

- Approximate reconstruction assuming direct lighting is usually good enough.
- Coarse-to-fine schemes work well.

Parameterizations are super important:

- Loss functions very non-linear and change shape easily.
- Working with meshes is a pain (topology is awful and not (easily?) differentiable).

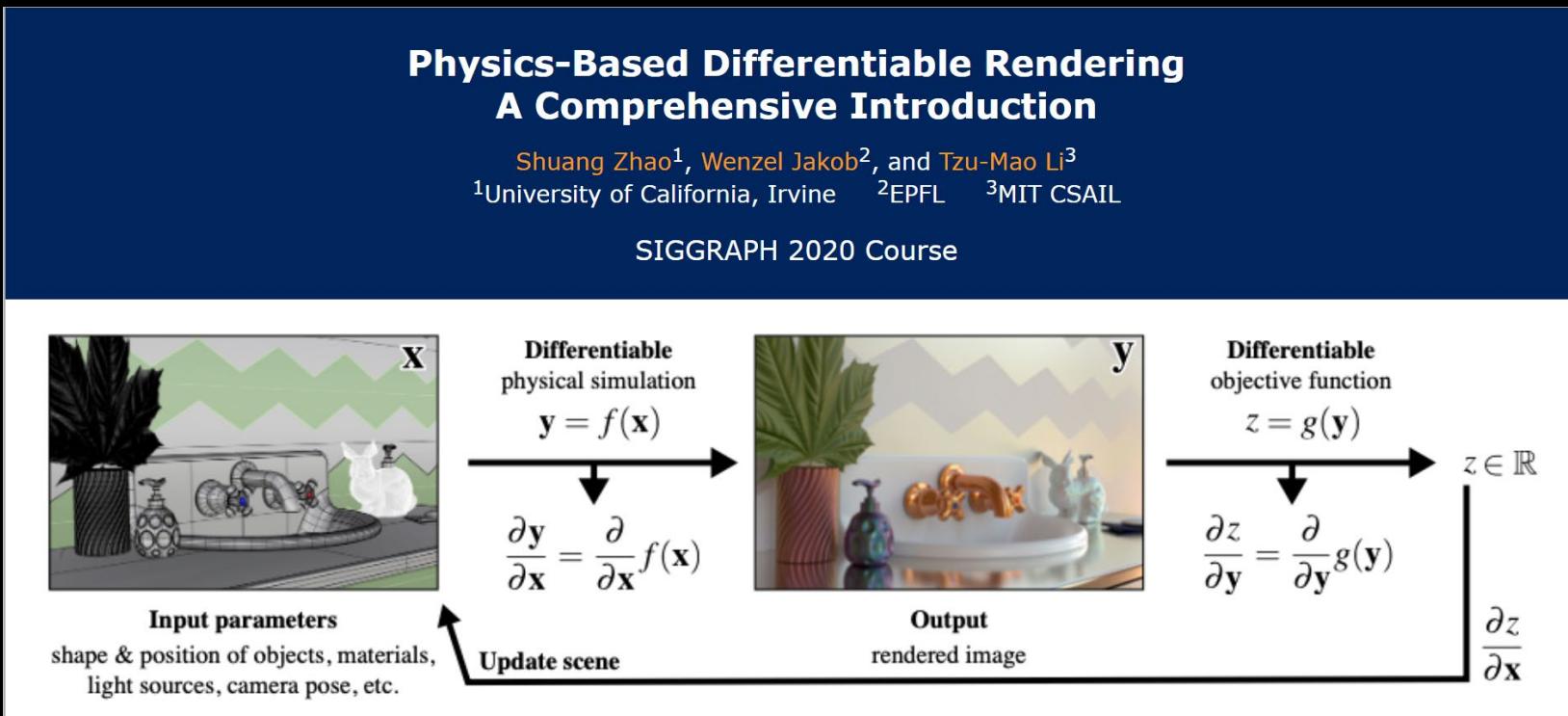
You don't always need Monte Carlo differentiable rendering:

- If you don't have strong global illumination, just use direct lighting.
- A lot of research in computer vision on differentiable rasterizers.

Remember that you are doing optimization:

- Unbiased and consistent gradients are very expensive to compute.
- Biased and/or inconsistent gradients can be very cheap to compute.
- Often, biased and/or inconsistent gradients are enough for convergence.
- Stochastic gradient descent matters a lot.

Reference material



CVPR 2021 Tutorial Proposal

Title: Tutorial on Physics-Based Differentiable Rendering

Proposers' Names, Titles, Affiliations, and Primary Contact Emails:

Shuang Zhao
Assistant Professor, CS
University of California, Irvine
shz@ics.uci.edu

Ioannis Gkioulekas
Assistant Professor, RI
Carnegie Mellon University
igkioule@cs.cmu.edu