Metropolis Sampling

Matt Pharr
cs348b

May 20, 2003

Introduction

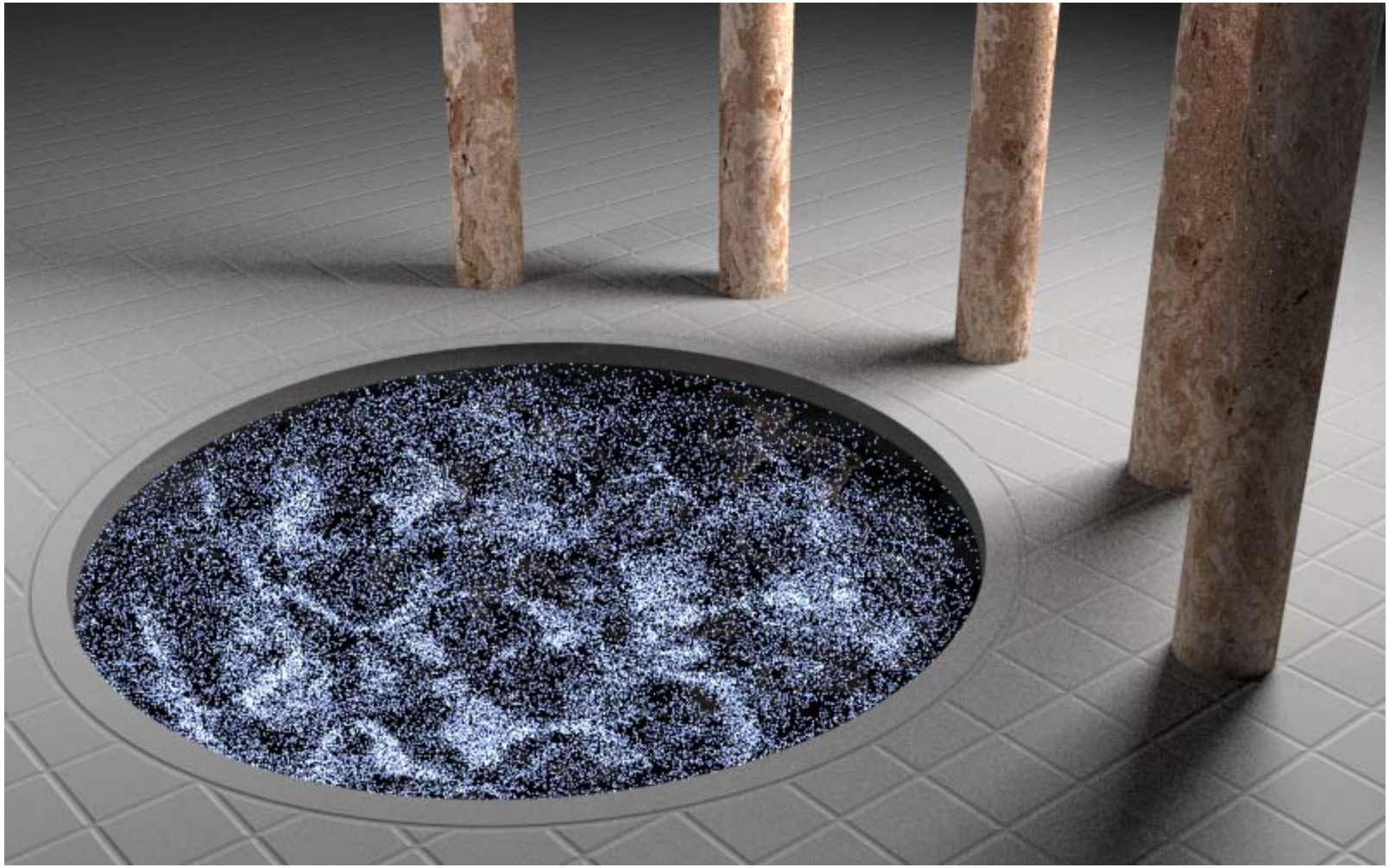
- Unbiased MC method for sampling from functions' distributions
- Robustness in the face of difficult problems
- Application to a wide variety of problems
- Flexibility in choosing how to sample
- Introduced to CG by Veach and Guibas

(a) Bidirectional path tracing with 40 samples per pixel.

Image credit: Eric Veach

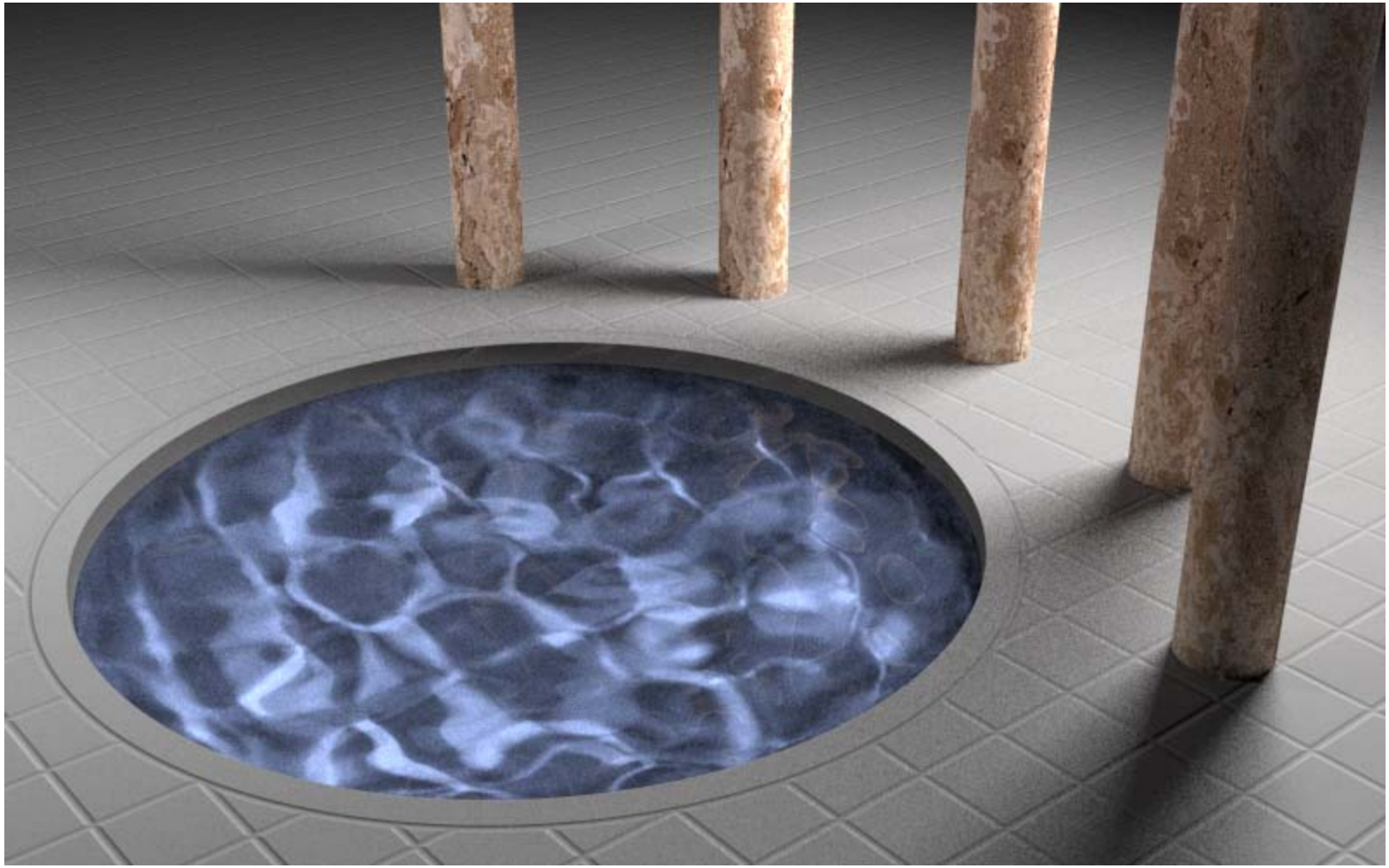
(b) Metropolis light transport with an average of 250 mutations per pixel [the same computation time as (a)].

Image credit: Eric Veach



(a) Path tracing with 210 samples per pixel.

Image credit: Eric Veach



(b) Metropolis light transport with an average of 100 mutations per pixel [the same computation time as (a)].

Image credit: Eric Veach

Overview

- For arbitrary $f(x) \rightarrow \mathbb{R}$, $x \in \Omega$

Overview

- For arbitrary $f(x) \rightarrow \mathbb{R}$, $x \in \Omega$
- Define $\mathbf{I}(f) = \int_\Omega f(x) \mathrm{d}\Omega$ so $f_{\mathrm{pdf}} = f/\mathbf{I}(f)$

Overview

- For arbitrary $f(x) \to \mathbb{R}$, $x \in \Omega$
- Define $\mathbf{I}(f) = \int_\Omega f(x) \mathrm{d}\Omega$ so $f_{\mathrm{pdf}} = f/\mathbf{I}(f)$
- Generates samples $X = \{x_i\}$, $x_i \sim f_{\mathrm{pdf}}$
- *Without needing to compute* f_{pdf} *or* $\mathbf{I}(f)$

Overview

- Introduction to Metropolis sampling
- Examples with 1D problems
- Extension to 3D, motion blur
- Overview of Metropolis Light Transport

Basic Algorithm

- Function $f(x)$ over state space Ω, $f : \Omega \rightarrow \mathbb{R}$.
- Markov Chain: new sample x_i using x_{i-1}

Basic Algorithm

- Function $f(x)$ over state space Ω, $f : \Omega \longrightarrow \mathbb{R}$.
- Markov Chain: new sample x_i using x_{i-1}
- New samples from *mutation* to $x_{i-1} \longrightarrow x'$
- Mutation accepted or rejected so $x_i \sim f_{\mathrm{pdf}}$
- If rejected, $x_i = x_{i-1}$
- Acceptance guarantees distribution of x_i is the *stationary distribution*

Pseudo-code

```
x = x0
for i = 1 to n
    x' = mutate(x)
    a = accept(x, x')
    if (random() < a)
        x = x'
    record(x)
```

Expected Values

- Metropolis avoids parts of Ω where $f(x)$ is small
- But e.g. dim parts of an image need samples
- Record samples at both x and x'
- Samples are weighted based on $a(x \rightarrow x')$
- Same result in the limit

Expected Values – Pseudo-code

```
x = x0
for i = 1 to n
    x' = mutate(x)
    a = accept(x, x')
    record(x, (1-a) * weight)
    record(x', a * weight)
    if (random() < a)
        x = x'
```

Mutations, Transitions, Acceptance

- Mutations propose x' given x_i
- $T(x \to x')$ is probability density of proposing x' from x
- $a(x \to x')$ probability of accepting the transition

Detailed Balance – The Key

- By defining $a(x \to x')$ carefully, can ensure $x_i \sim f(x)$

$$f(x)\, T(x \to x')\, a(x \to x') = f(x')\, T(x' \to x)\, a(x' \to x)$$

- Since f and T are given, gives conditions on acceptance probability
- (Will not show derivation here)

Acceptance Probability

- Efficient choice:

$$a(x \to x') = \min\left(1, \frac{f(x')\,T(x' \to x)}{f(x)\,T(x \to x')}\right)$$

Acceptance Probability – Goals

- Doesn't affect unbiasedness; just variance
- Maximize the acceptance probability $\rightarrow$
  - Explore state space better
  - Reduce correlation (image artifacts...)
- Want transitions that are likely to be accepted
  - i.e. transitions that head where $f(x)$ is large

Mutations: Metropolis

- $T(a \to b) = T(b \to a)$ for all a, b

$$a(x \to x') = \min\left(1, \frac{f(x')}{f(x)}\right)$$

- Random walk Metropolis

$$T(x \to x') = T(|x - x'|)$$

Mutations: Independence Sampler

- If we have some pdf p, can sample $x \sim p$,
- Straightforward transition function:

$$T(x \to x') = p(x')$$

- If $p(x) = f_{\text{pdf}}$, wouldn't need Metropolis
- But can use pdfs to approximate parts of f...

Mutation Strategies: General

- Adaptive methods: vary transition based on experience
- Flexibility: base on value of $f(x)$, etc. pretty freely
- Remember: just need to be able to compute transition densities for the mutation
- The more mutations, the merrier
- Relative frequency of them not so important

1D Example

- Consider the function

$$f^1(x) = \begin{cases} (x - .5)^2 & : \quad 0 \leq x \leq 1 \\ 0 & : \quad \text{otherwise} \end{cases}$$

- Want to generate samples from $f^1(x)$

1D Mutation #1

$$\begin{aligned} \text{mutate}_1(x) &\rightarrow \xi \\ T_1(x \rightarrow x') &= 1 \end{aligned}$$

- Simplest mutation possible
- Random walk Metropolis

1D Mutation #2

$$\begin{aligned}
\text{mutate}_2(x) &\rightarrow x + .1 * (\xi - .5) \\
T_2(x \rightarrow x') &= \begin{cases} \frac{1}{0.1} & : \quad |x - x'| \leq .05 \\ 0 & : \quad \text{otherwise} \end{cases}
\end{aligned}$$

- Also random walk Metropolis

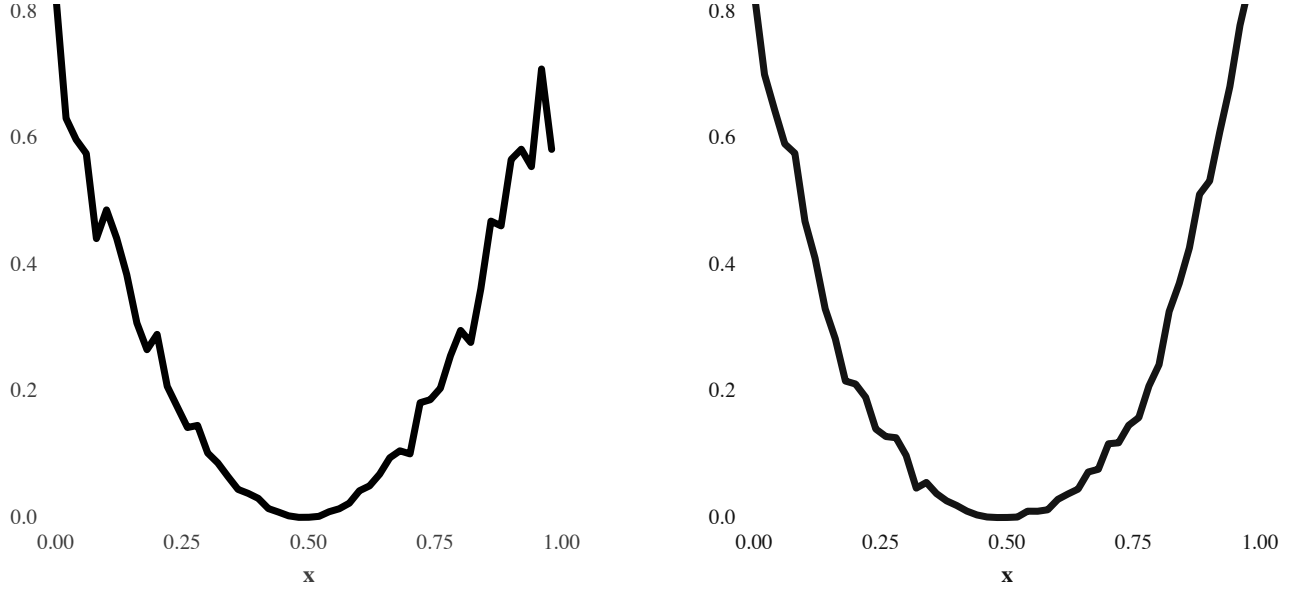
1D Mutation #2

- mutate_2 increases acceptance probability

$$a(x \to x') = \min\left(1, \frac{f(x')\,T(x' \to x)}{f(x)\,T(x \to x')}\right)$$

- When $f(x)$ is large, will avoid x' when $f(x') < f(x)$
- Should try to avoid proposing mutations to such x'

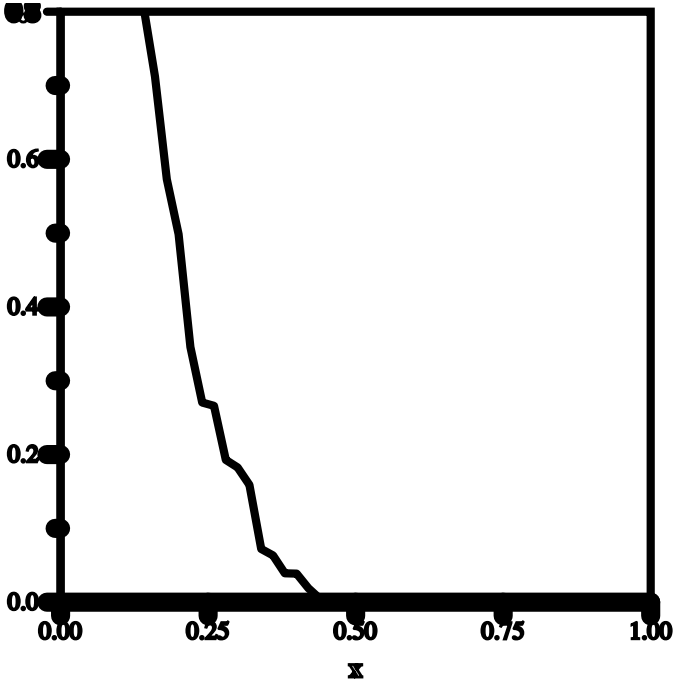
1D Results - pdf graphs



- Left: mutate_1 only
- Right: a mix of the two (10%/90%)
- 10,000 mutations total

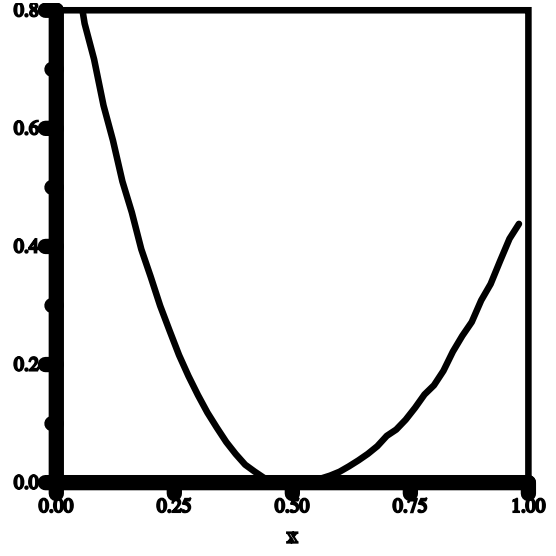
Why bother with mutate_1, then?

- If we just use the second mutation ($\pm .05$)...



Ergodicity

- Need finite prob. of sampling x, $f(x) > 0$
- This is true with mutate_2, but is inefficient:



- Still unbiased in the limit...

Ergodicity – Easy Solution

- Periodically pick an entirely new x
- e.g. sample uniformly over Ω...

Application to Integration

- Given integral, $\int f(x)g(x)\mathrm{d}\Omega$
- Standard Monte Carlo estimator:

$$\int_\Omega f(x)g(x)\,\mathrm{d}\Omega \approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)g(x_i)}{p(x_i)}$$

- where $x_i \sim p(x)$, an arbitrary pdf

Application to Integration

$$\int_\Omega f(x)g(x)\,\mathrm{d}\Omega \approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)g(x_i)}{p(x_i)}$$

Application to Integration

$$\int_\Omega f(x)g(x)\,\mathrm{d}\Omega \approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)g(x_i)}{p(x_i)}$$

- Metropolis gives $x_1, \ldots, x_N$, $x_i \sim f_{\mathrm{pdf}}(x)$

$$\int_\Omega f(x)g(x)\,\mathrm{d}\Omega \approx \left[\frac{1}{N}\sum_{i=1}^{N} g(x_i)\right] \cdot \mathbf{I}(f)$$

- (Recall $\mathbf{I}(f) = \int_\Omega f(x)\mathrm{d}\Omega$)

Start-Up Bias

- x_i converges to f_{pdf}, but never actually reaches it
- Especially in early iterations, the distribution of x_i can be quite different from f_{pdf}
- This problem is called the **Start-Up Bias**

Eliminating Start-Up Bias

- Start from any x_0, make a couple of "dummy" iterations without recording the results
  - How many "dummy" iterations?
  - x_i only proportional to f_{pdf} for *i* -> infty anyway
  - Not a good solution

Eliminating Start-Up Bias

- Generate x_0, proportional to any suitable pdf $p(x)$
- Weight all contributions by $w = f(x_0) / p(x_0)$

  - Unbiased on average (over many runs)
  - Each run may be completely "off"
    (even black image, if $f(x_0)$ was zero)

  - Not a good solution

Eliminating Start-Up Bias

- Generate *n* initial samples $x_{0,1}, \ldots, x_{0,n}$ proportional to any suitable pdf $p(x)$
- Compute weights $w_i = f(x_{0,i}) / p(x_{0,i})$
- Pick initial state proportional to w_i
- Weight all contributions by $w = \frac{1}{n} \sum_{i=1}^{n} w_i$
  - Note that

$$E[w] = E\left[\frac{1}{n} \sum_{i=1}^{n} \frac{f(x_{0,i})}{p(x_{0,i})} \right] = \int_{\Omega} f(x) dx$$

Motion Blur

- Onward to a 3D problem
- Scene radiance function $L(u, v, t)$ (e.g. evaluated with ray tracing)
- $L = 0$ outside the image boundary
- Ω is $(u, v, t) \in [0, \mathrm{u_{max}}] \times [0, \mathrm{v_{max}}] \times [0, 1]$

Image Contribution Function

- The key to applying Metro to image synthesis

$$I_j = \int_\Omega h_j(u,v)\, L(u,v,t)\, \mathrm{d}u\, \mathrm{d}v\, \mathrm{d}t$$

- I_j is value of j'th pixel
- h_j is pixel reconstruction filter

Image Contribution Function

- So if we sample $x_i \sim L_{\mathrm{pdf}}$

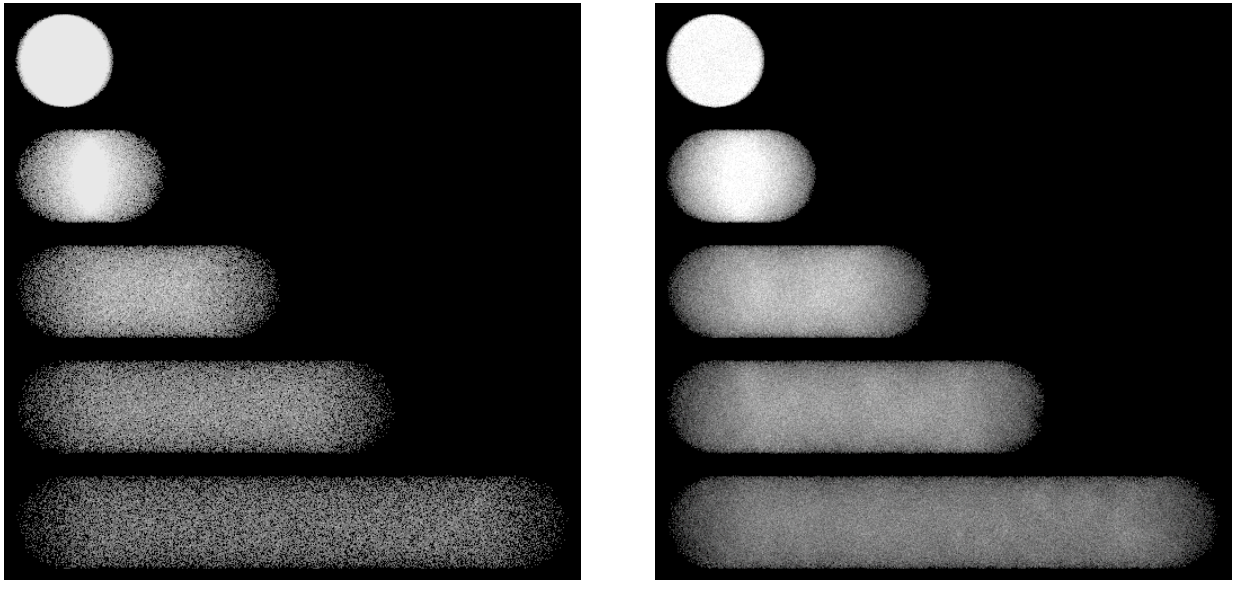
$$I_j \approx \frac{1}{N} \sum_{i=1}^{N} h_j(x_i) \cdot \left(\int_\Omega L(x) \, \mathrm{d}\Omega \right),$$

- *The distribution of x_i on the image plane forms the image*
- Estimate $\int_\Omega L(x) \, \mathrm{d}\Omega$ with standard MC

Two Basic Mutations

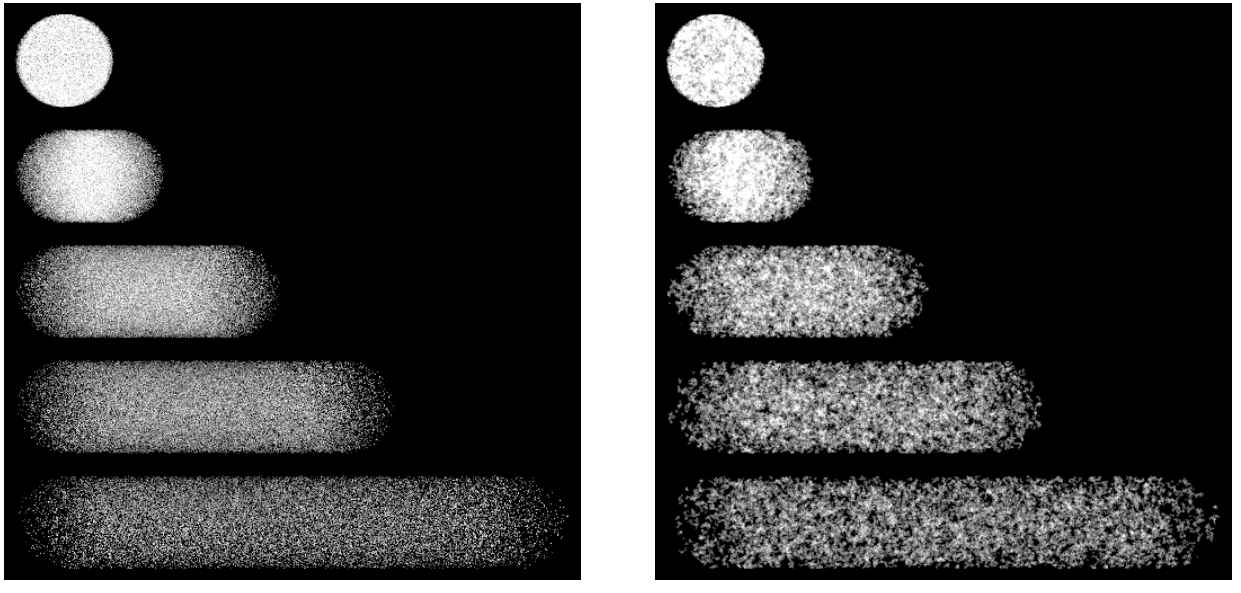
- Pick completely new (u, v, t) value
- Perturb u and v ± 8 pixels, time $\pm .01$.
- Both are symmetric, Random-walk Metropolis

Motion Blur – Result



- Left: Distribution RT, stratified sampling
- Right: Metropolis sampling
- Same total number of samples

Motion Blur – Parameter Studies



- Left: ± 80 pixels, $\pm .5$ time. Many rejections.
- Right: ± 0.5 pixels, $\pm .001$ time. Didn't explore Ω well.

Exponential Distribution

- Vary the scale of proposed mutations

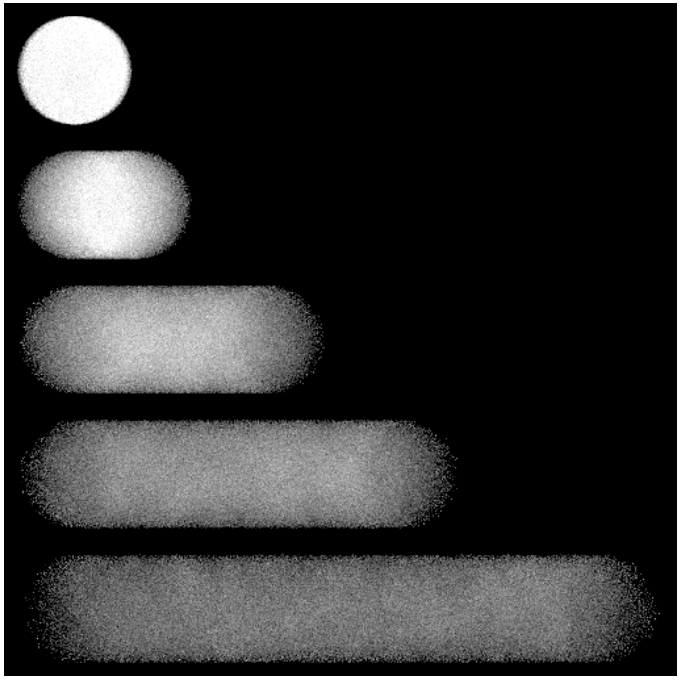
$$r = r_{\max} \, \mathrm{e}^{-\log(r_{\max}/r_{\min})\xi}, \quad \theta = 2\pi\xi$$

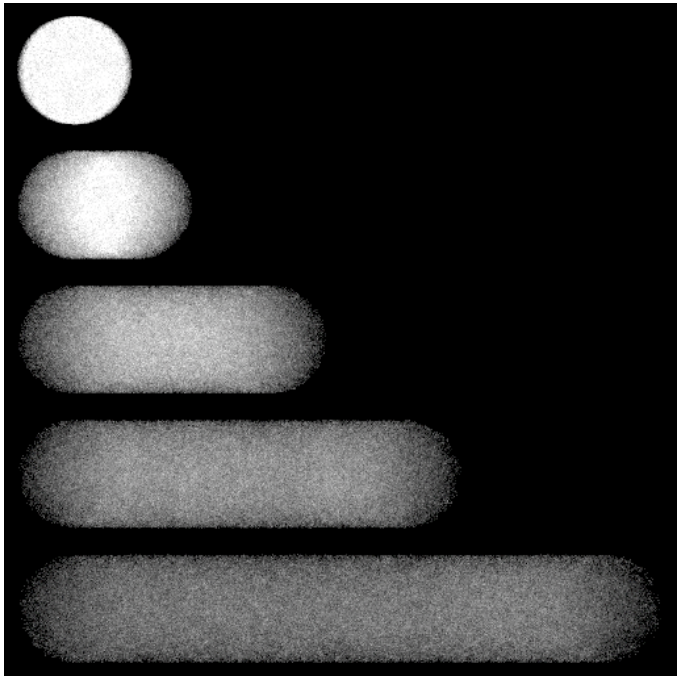
$$(du, dv) = (r \sin\theta, r \cos\theta)$$

$$dt = t_{\max} \, \mathrm{e}^{-\log(t_{\max}/t_{\min})\xi}$$

- Will reject when too big, still try wide variety

Exponential distribution results



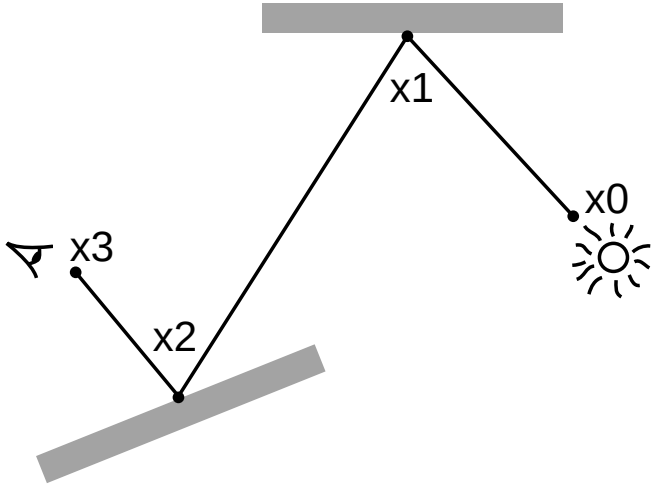


Light Transport

- Image contribution function was key
- $f(x)$ over infinite space of paths
- State-space is light-carrying paths through the scene–from light source to sensor
- Robustness is particularly nice–solve difficult transport problems efficiently
- Few specialized parameters to set

Light Transport – Setting

- Samples x from Ω are sequences $v_0 v_1 \ldots v_k$, $k \geq 1$, of vertices on scene surfaces



- $f(x)$ is the product of emitted light, BRDF values, cosines, etc.

Light Transport – Strategy

- Explore the infinite-dimensional path space
- Metropolis's natural focus on areas of high contribution makes it efficient
- New issues:
  - Stratifying over pixels
  - Perceptual issues
  - Spectral issues
  - Direct lighting

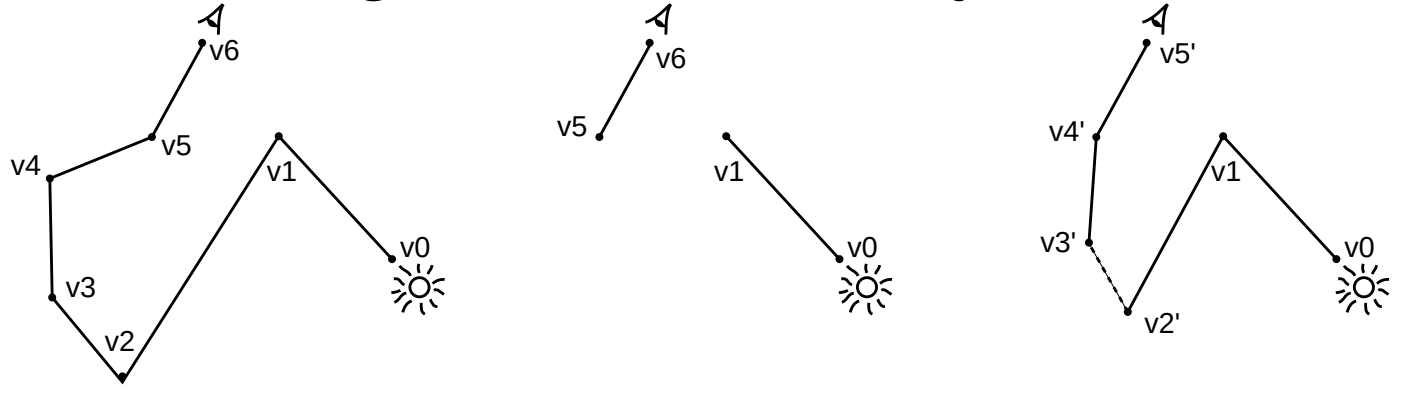
MLT Pseudocode

Legend:	
$\mathbf{z}$	sampling state (light path)
$I(\mathbf{z})$	"scalar contribution function" (i.e target function for Metropolis mutations, was $f(\mathbf{z})$ earlier)
$F(\mathbf{z})$	integrand (was $h(\mathbf{z})f(\mathbf{z})$ earlier)

Generate path seeds
Approximate $b = \int I(\mathbf{z})\, d\mathbf{z}$ from the seeds
Find $\mathbf{z}_1$ from the seeds using $I(\mathbf{z})$
for $i = 1$ **to** M **do**
    Based on $\mathbf{z}_i$, sample a tentative point $\mathbf{z}_t$ using $T(\mathbf{z}_i \rightarrow \mathbf{z}_t)$
    $a(\mathbf{z}_i \rightarrow \mathbf{z}_t) = \min\left\{\frac{I(\mathbf{z}_t)\cdot T(\mathbf{z}_t \rightarrow \mathbf{z}_i)}{I(\mathbf{z}_i)\cdot T(\mathbf{z}_i \rightarrow \mathbf{z}_t)}, 1\right\}$
    Select pixel j to which $\mathbf{z}_i$ contributes
    $\Phi_j \mathrel{+}= \frac{b}{M} \cdot \frac{F(\mathbf{z}_i)}{I(\mathbf{z}_i)} \cdot (1 - a(\mathbf{z}_i \rightarrow \mathbf{z}_t))$
    Select pixel k to which $\mathbf{z}_t$ contributes
    $\Phi_k \mathrel{+}= \frac{b}{M} \cdot \frac{F(\mathbf{z}_t)}{I(\mathbf{z}_t)} \cdot a(\mathbf{z}_i \rightarrow \mathbf{z}_t)$
                                    // accept with probability $a(\mathbf{z}_i \rightarrow \mathbf{z}_t)$
    Generate random number r in $[0, 1]$.
    if $r < a(\mathbf{z}_i \rightarrow \mathbf{z}_t)$ **then** $\mathbf{z}_{i+1} = \mathbf{z}_t$
    else $\mathbf{z}_{i+1} = \mathbf{z}_i$
endfor

Bidirectional Mutation

- Delete a subpath from the current path
- Generate a new one
- Connect things with shadow rays



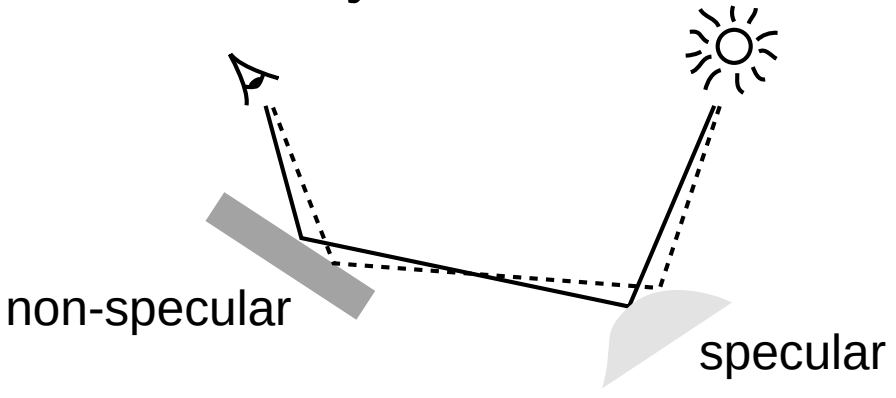
- If occluded, then just reject

Bidirectional Mutation

- Very flexible path re-use
- Ensures ergodicity–may discard the entire path
- Inefficient when a very small part of path space is important
- Transition densities are tricky: need to consider *all* possible ways of sampling the path

Caustic Perturbation

- Caustic path: one more more specular surface hits before diffuse, eye



- Slightly shift outgoing direction from light source, regenerate path

Lens Perturbation

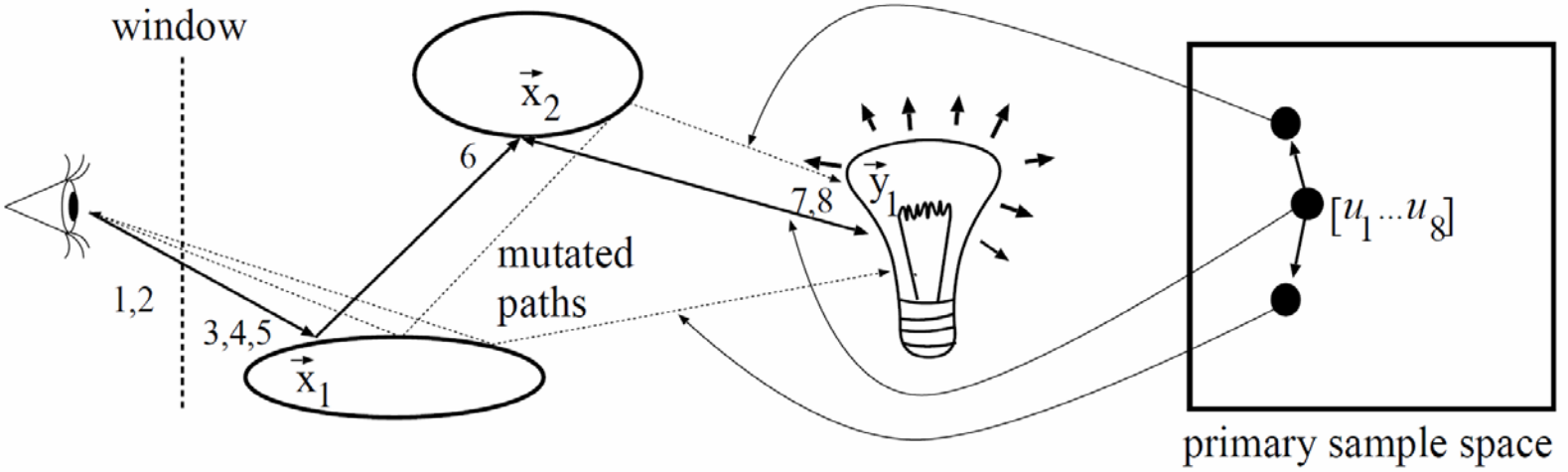
- Similarly perturb outgoing ray from camera
- Keeps image samples from clumping together

Why It Works Well

- Path Reuse
  - Efficiency–paths are built from pieces of old ones
  - (Could be used in stuff like path tracing...)
- Local Exploration
  - Given important path, incrementally sample close to it in Ω
  - When f is small over much of Ω, this is extra helpful

MLT in Primary Sample Space

- [Kelement et al. 2002]
- Light path is uniquely determined by the random numbers used to generate it.



MLT in Primary Sample Space

- Formulate MLT as an integration problem in the space of uniformly distributed random numbers (the “primary space”)
- New integrand:

$$F^*(\mathbf{u}) = F(S(\mathbf{u})) \cdot \left| \frac{dS(\mathbf{u})}{d\mathbf{u}} \right| = \frac{F(S(\mathbf{u}))}{p_S(\mathbf{u})}$$

- New scalar contrib. function:

$$I^*(\mathbf{u}) = \frac{I(S(\mathbf{u}))}{p_S(\mathbf{u})}$$

MLT in Primary Sample Space

- Mutations in primary space
- Small steps
  - Perturb all the u_i's using exponential distribution
- Large steps
  - Regenerate path from scratch

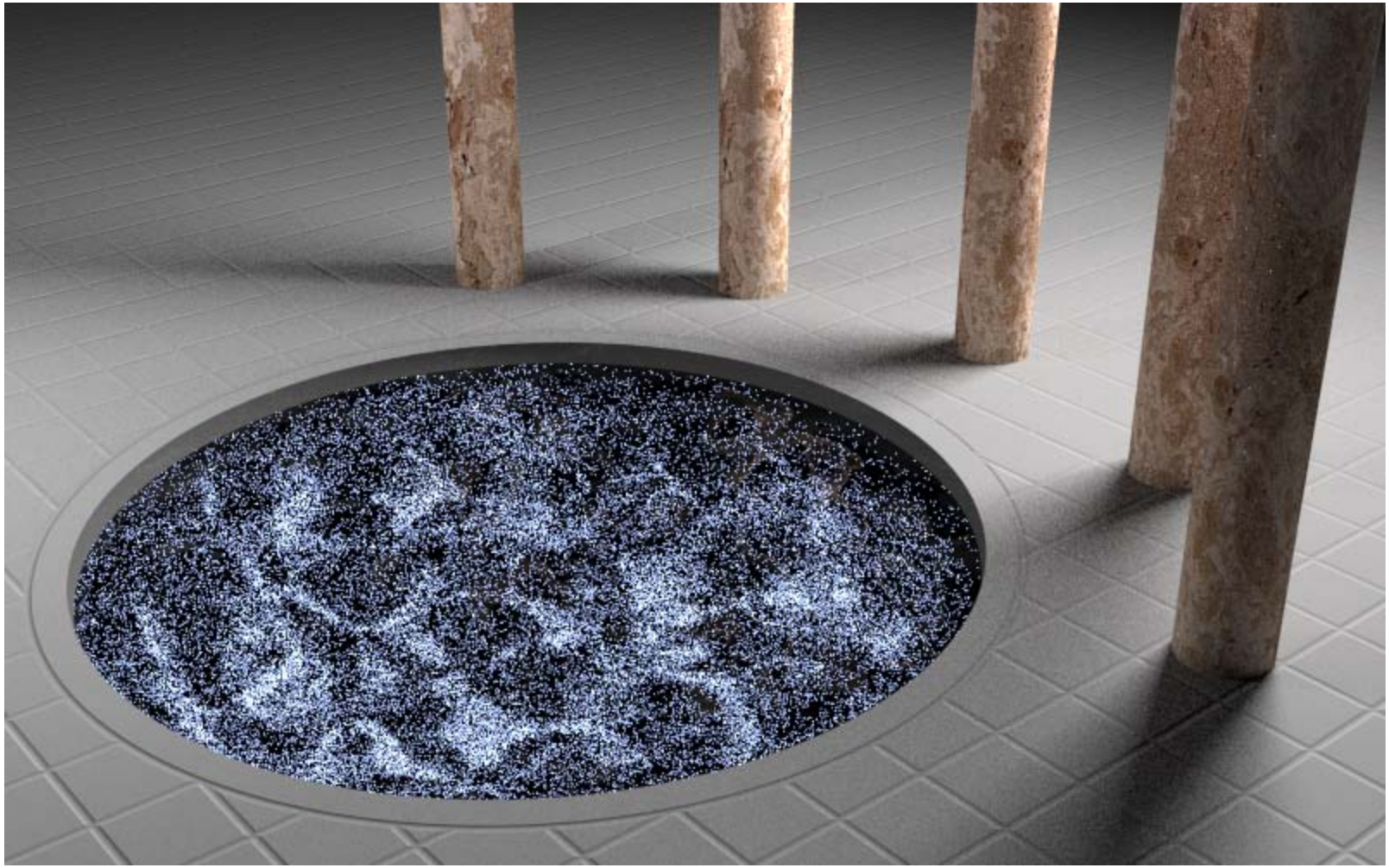
- Ergodicity, symmetry (no need to evaluate T)

(a) Bidirectional path tracing with 40 samples per pixel.

Image credit: Eric Veach

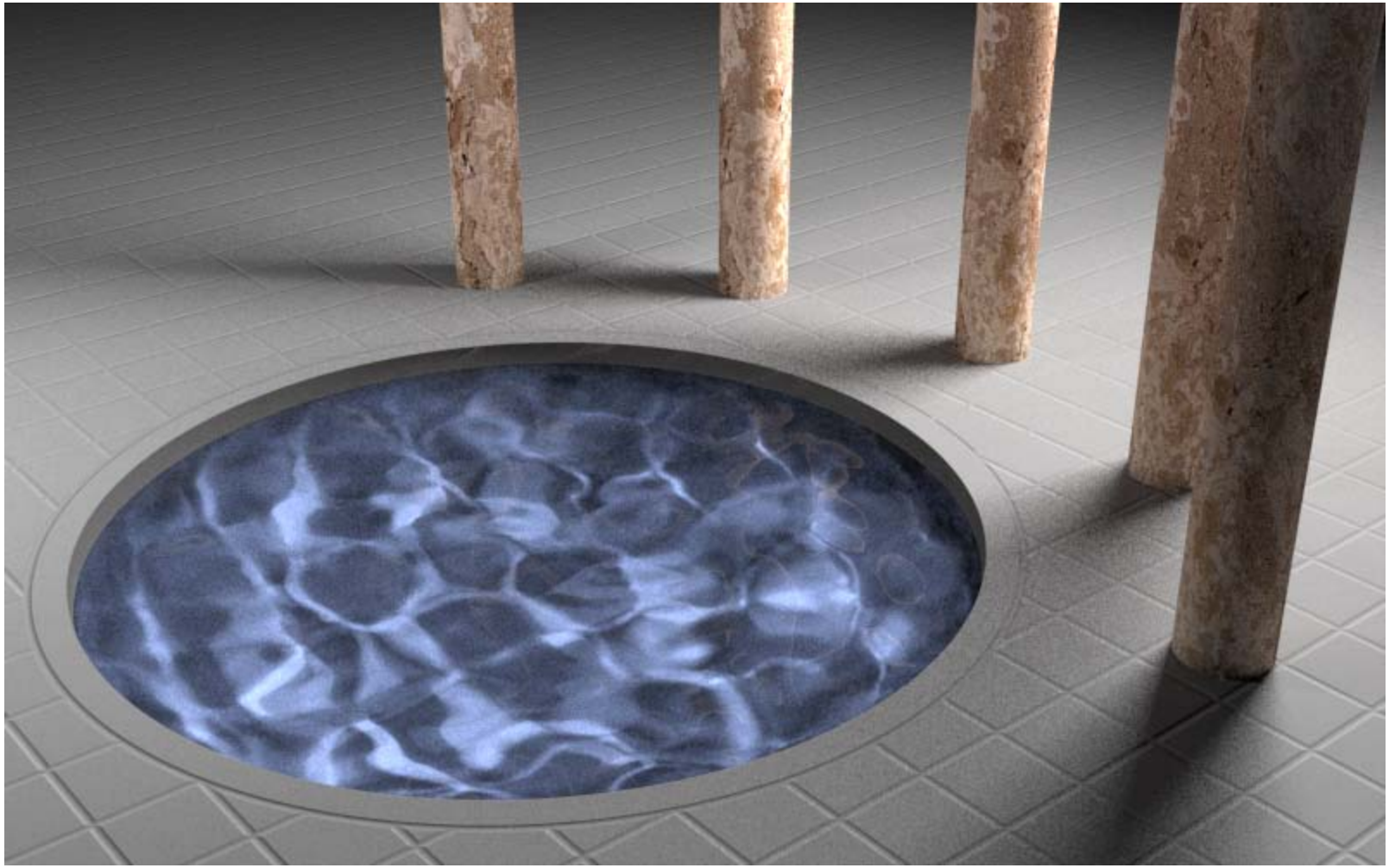
(b) Metropolis light transport with an average of 250 mutations per pixel [the same computation time as (a)].

Image credit: Eric Veach



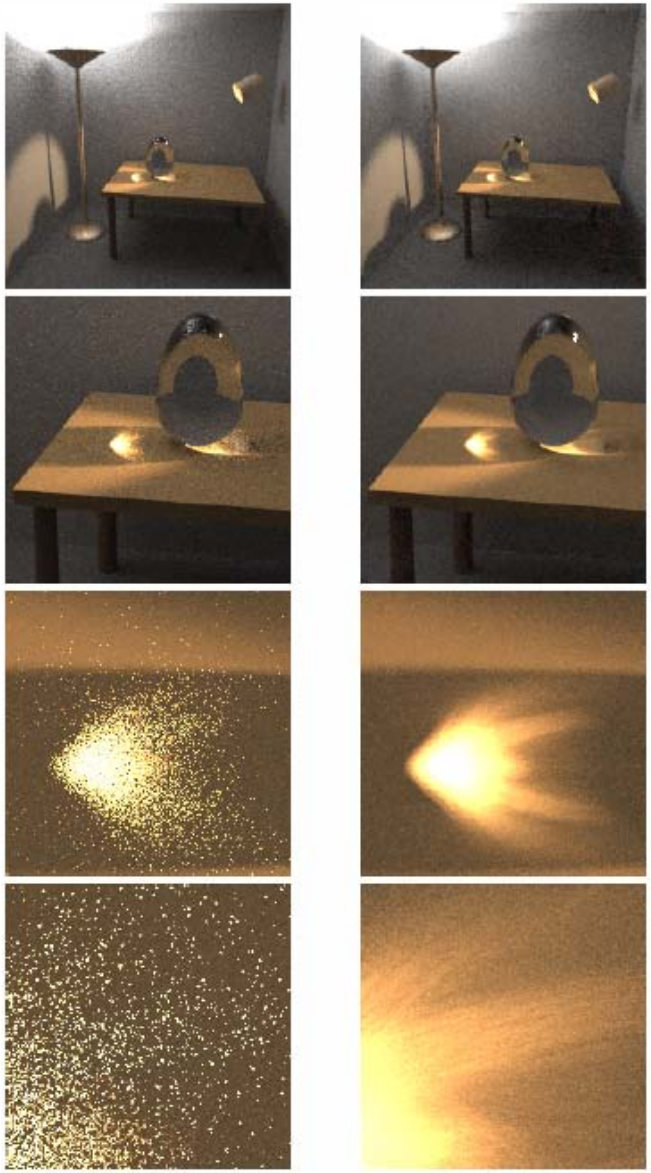
(a) Path tracing with 210 samples per pixel.

Image credit: Eric Veach



(b) Metropolis light transport with an average of 100 mutations per pixel [the same computation time as (a)].

Image credit: Eric Veach



(a) (b)

Figure 6: These images show caustics formed by a spotlight shining on a glass egg. Column (a) was computed with bidirectional path tracing using 25 samples per pixel, while (b) uses Metropolis light transport with the same number of ray queries (varying between 120 and 200 mutations per pixel). The solutions include all paths of up to length 7, and the images are 200 by 200 pixels.

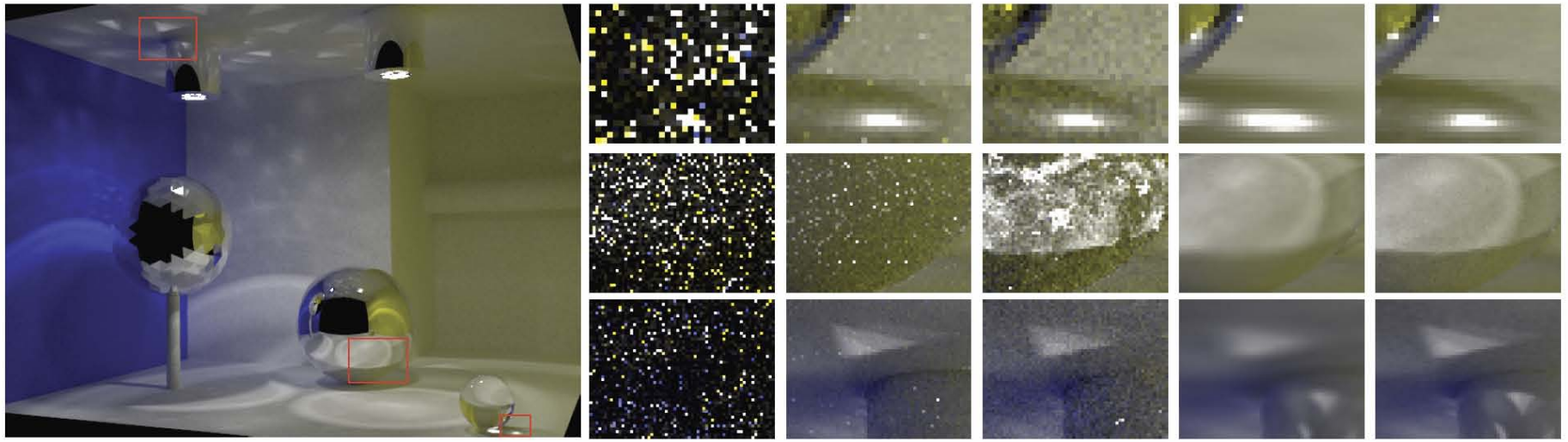
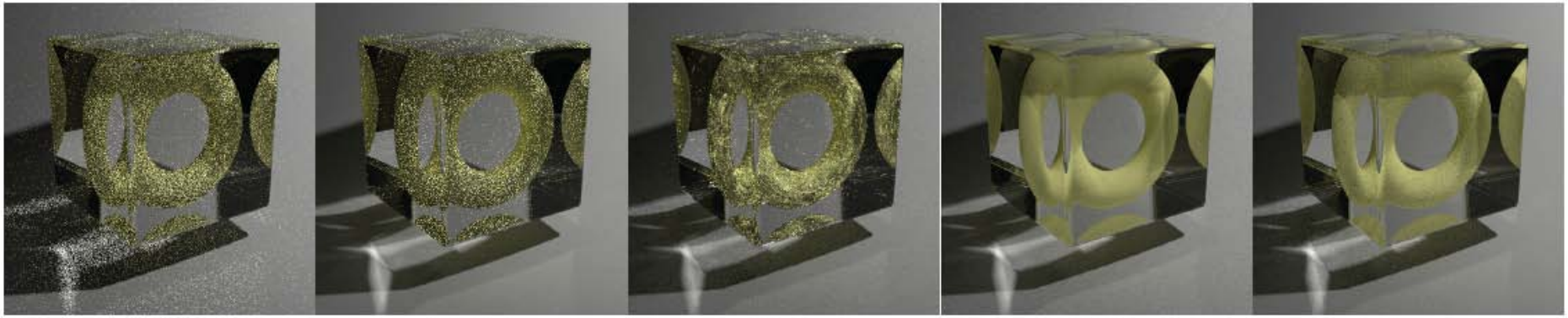


Figure 6: *A box scene illuminated by a lighting fixture. The lighting fixture is behind glass and the illumination in the scene is dominated by caustics. The specular reflections and refractions have significant noise even with Metropolis light transport. Standard photon mapping cannot resolve the sharp illumination details in the scene with the maximum 20 million photons in the photon map. With progressive photon mapping we could use 213 million photons, which resolves all the details in the scene and provides a noise free image in the same rendering time as the Monte Carlo ray tracing methods.*



PT BDPT MLT PPM Reference

Figure 7: *Torus embedded in a glass cube. The reference image on the far right have been rendered using path tracing with 51500 samples per pixel. The Monte Carlo ray tracing methods fail to capture the lighting within the glass cube, while progressive photon mapping provides a smooth result using the same rendering time.*

Image credit: Toshiya Hachisuka

Figure 8: *Lighting simulation in a bathroom. The scene is illuminated by a small lighting fixture consisting of a light source embedded in glass. The illumination in the mirror cannot be resolved using Monte Carlo ray tracing. Photon mapping with 20 million photons results in a noisy and blurry image, while progressive photon mapping is able to resolve the details in the mirror and in the illumination without noise.*

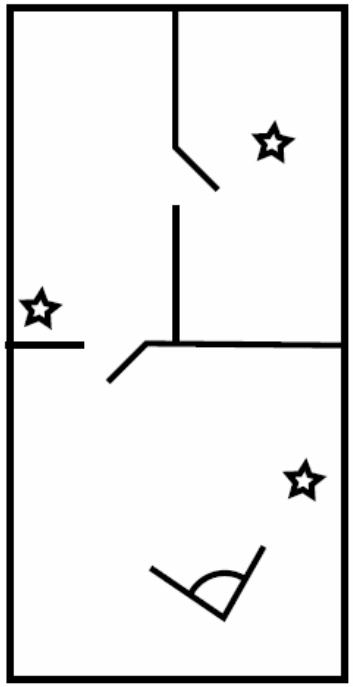
Image credit: Toshiya Hachisuka

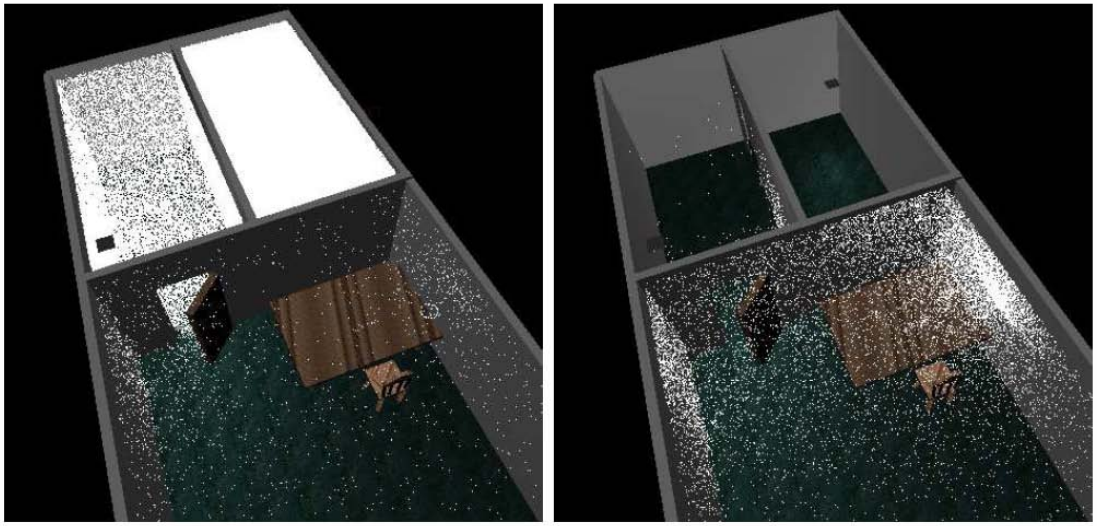
Other applications of Metropolis sampling in Rendering

- Segovia et al. 2007
  Metropolis Instant Radiosity
- Ghosh & Heidrich 2006
  Metropolis sampling of environment maps

- Metropolis photon sampling
  - Fan et al. 2005
  - Hachisuka and Jensen 2011

Fan et al. 2005





Hachisuka and Jensen 2011

- Simplifying trick:
  Target function is the binary photon visibility V(x)

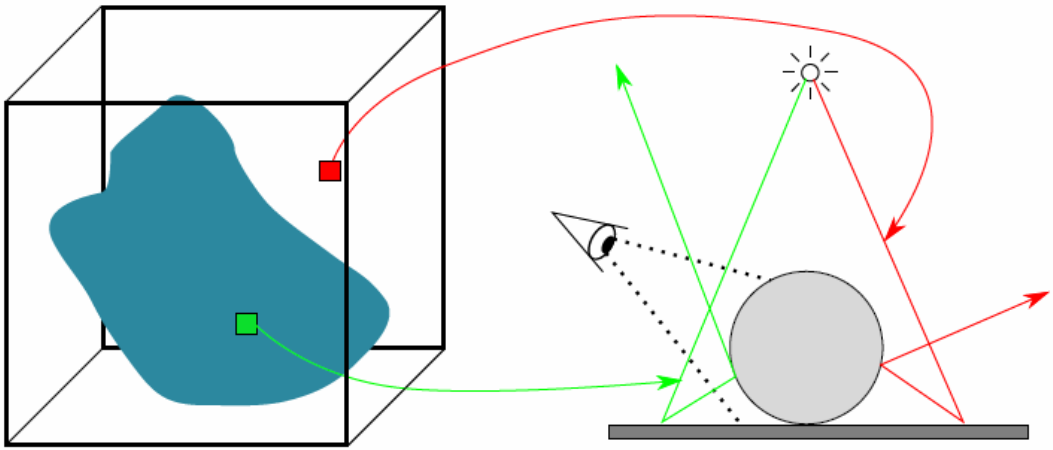


Fig. 2. Sampling space of our method. We define a function $V(x)$ in the hypercube of random numbers. The function returns 1 if a corresponding photon path contributes to the image (the green point in the shaded region) and 0 otherwise (the red point outside the shaded region).

Hachisuka and Jensen 2011

- Other important trick: Adaptive mutation size
  - Adapt mutation size to the number of accepted mutations

Fig. 7. The effect of adaptive mutation size. A pocket watch is illuminated by a hemispherical light source and a directional light source and is rendered with depth-of-field. Illumination on the dial-plate is due to caustics from the glass cover and the metal lid. The images shown are rendered by uniform random sampling (the leftmost image) and our photon tracing method (right three images) in the same rendering time. The second image uses mutation size that is too small ($d_i = 0.01$) and the fourth image uses mutation size that is too large ($d_i = 4.0$). The adaptive Markov chain Monte Carlo method used in our method (the third image) produces the least noisy result without any parameter tuning.

Hachisuka and Jensen 2011

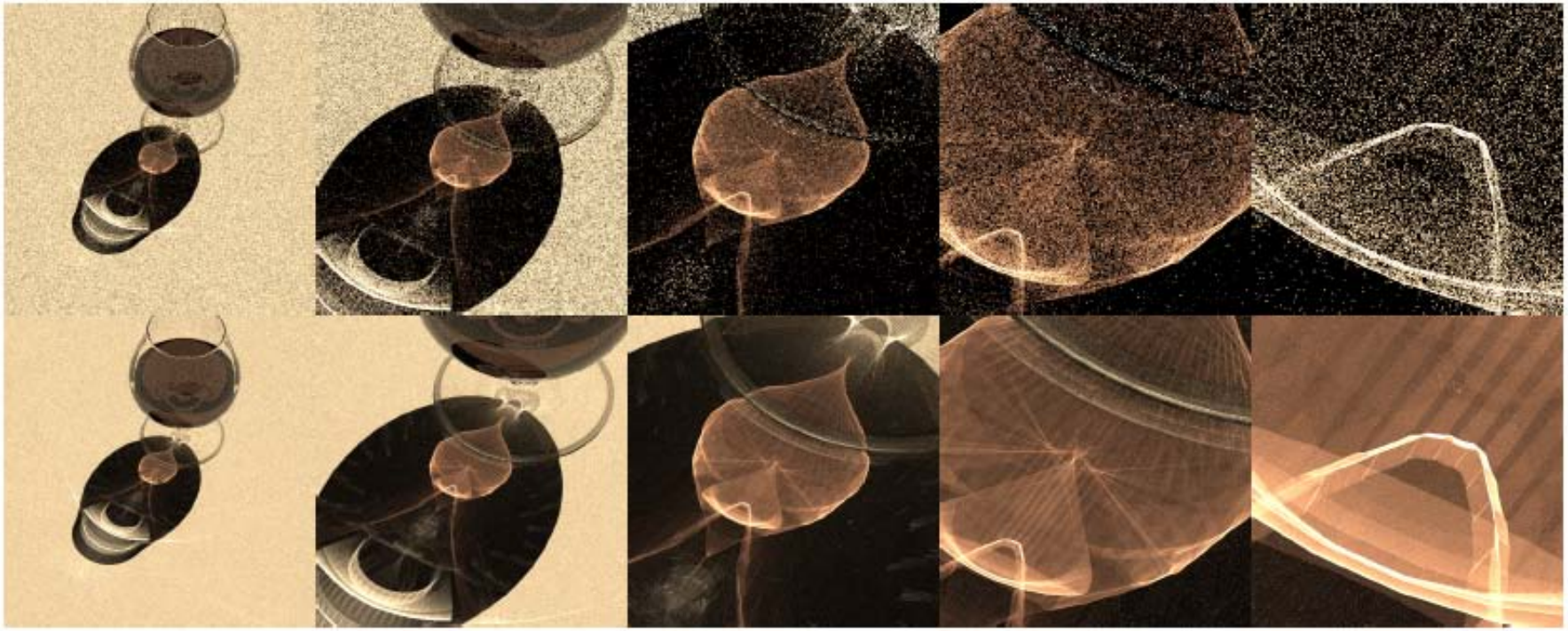


Fig. 1. Cognac glass illuminated by a directional light source. The figure compares rendered images using progressive photon mapping with the same rendering time (120 min), but with different photon tracing algorithms. The images on the top row are rendered using random sampling of photons, which become increasingly noisy as we zoom into the caustic. Using our photon tracing method (bottom row), we can focus tracing photon paths into the region that contributes to the image without any portal, and render the close-ups with less noise in the same rendering time. Note that no other existing global illumination methods can render illumination under the cognac glass accurately, since this illumination comes from specular-diffuse-specular paths from a light source with zero solid angle. The combination of progressive photon mapping and our photon tracing technique is the first method that works effectively and robustly in this kind of scene. The stripe patterns in the caustic are not artifacts of our method - they are caused by the tessellation of the cognac glass.

Hachisuka and Jensen 2011

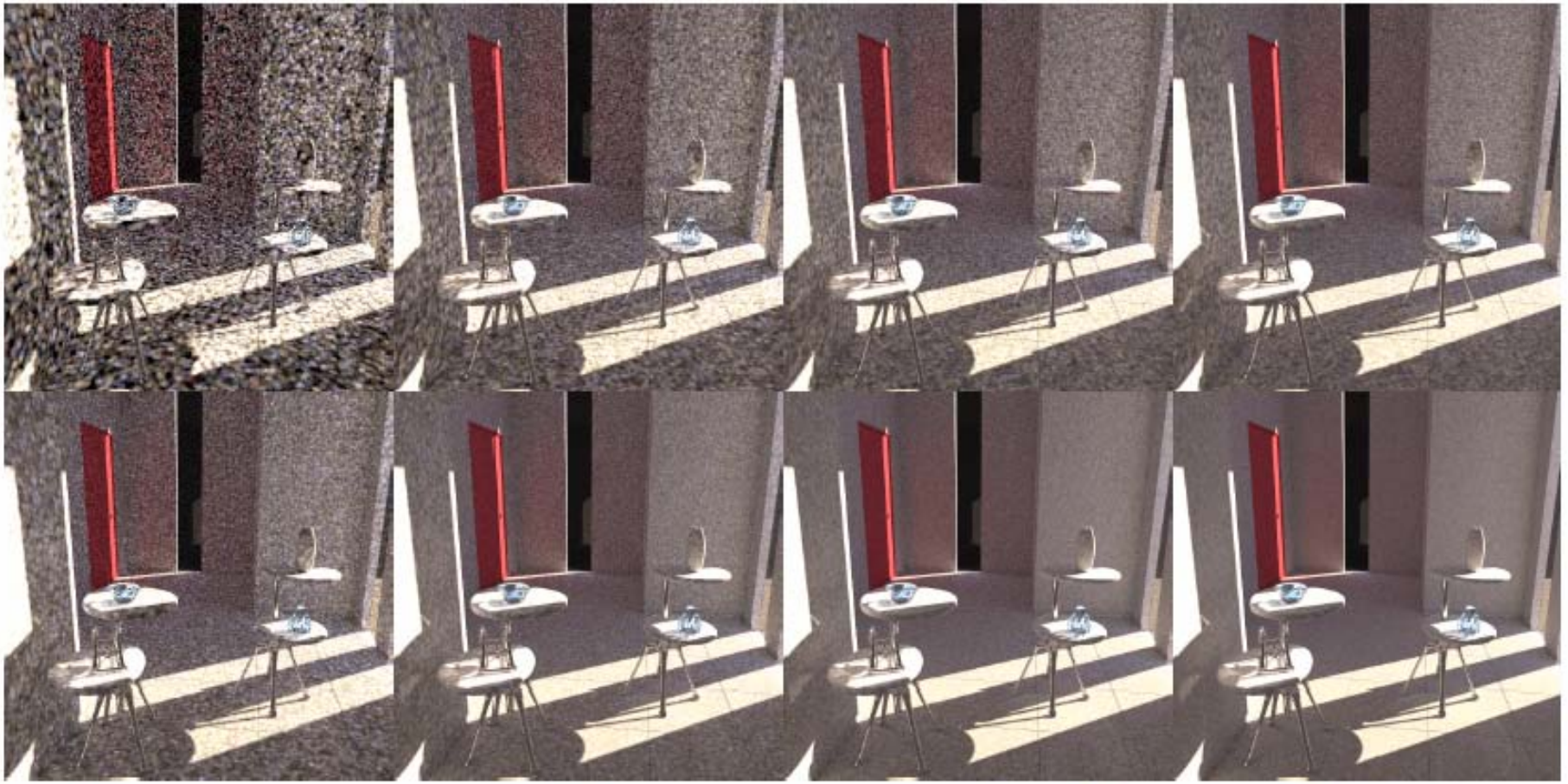


Fig. 5. Sequences of rendered images of a room (with permission of Youichi Kimura) illuminated by a directional light source . The top row shows the results with uniform sampling and the bottom row shows the results with our method using the same rendering time (1, 15, 30, and 60 min from left to right). Our photon tracing method robustly and automatically handles scenes that are considered difficult to render with existing photon tracing approaches. The illumination is coming through the glass window and only photon tracing approaches can handle such paths without ignoring specular reflections and refractions at the window.

Conclusion

- A very different way of thinking about integration
- Robustness is highly attractive
- Implementation can be tricky