Path Integral Methods for Light Transport Simulation: Theory & Practice

EUROGRAPHICS 2014 Tutorial

Tutorial materials are available from
http://cgg.mff.cuni.cz/~jaroslav/papers/2014-ltstutorial/index.htm

Organizer

Jaroslav Křivánek
Charles University in Prague

Lecturers

Iliyan Georgiev
Solid Angle, Saarland University

Anton S. Kaplanyan
Karlsruhe Institute of Technology

Juan Cañada
Next Limit Technologies

Synopsis. The tutorial presents a survey of the recent advances in robust light transport simulation methods. Based on a clear exposition of the path integral framework we discuss a wide range of algorithms, and discuss issues that arise when applying these advanced methods in practice.

Keywords. Global illumination, Light transport simuilation, Monte Carlo methods, Path integral.

Abstract

We are witnessing a renewed research interest in robust and efficient light transport simulation based on statistical methods. This research effort is propelled by the desire to accurately render general environments with complex materials and light sources, which is often difficult with the currently employed solutions. In addition, it has been recognized that advanced methods, which are able to render many effects in one pass without excessive tweaking, increase artists' productivity and allow them to focus on their creative work. For this reason, the movie industry is shifting away from approximate rendering solutions toward physically-based rendering methods, which poses new challenges in terms of strict requirements on image quality and algorithm robustness.

Many of the recent advances in light transport simulation, such as the robust combination of bidirectional path tracing with photon mapping (Vertex Connection and Merging / Unified Path Space), or the new Markov chain Monte Carlo methods are made possible by interpreting light transport as an integral in the space of light paths. However, there is a great deal of confusion among practitioners and researchers alike regarding these path space methods.

The goal of this tutorial is twofold. First, we present a coherent review of the path integral formulation of light transport and its applications, including the most recent ones. We show that rendering algorithms that may seem complex at first sight, are in fact naturally derived from this general framework. A significant part of the tutorial is devoted to the application of Markov chain Monte Carlo methods for light transport simulation, such as Metropolis Light Transport and its variants. We include an extensive empirical comparison of these MCMC methods. The second part of the tutorial discusses practical aspects of applying advanced light transport simulation methods in practical architectural visualization and VFX tasks.

Intended audience

Industry professionals and researchers interested in recent advances in robust light transport simulation for realistic rendering with global illumination.

Prerequisites

Familiarity with rendering and with concepts of global illumination computation is expected.

Level of difficulty

Intermediate

Tutorial length

Half-day (2×90 minutes)

Syllabus

1. Introduction & Welcome (*Křivánek*)
   (10 min)

2. Path Integral Formulation of Light Transport (*Křivánek*)
   (25 min)
   - Light transport as an integral over light paths
   - Monte Carlo integration primer
   - Path sampling methods and path probability density
   - Unidirectional path sampling: Path tracing, Light tracing, imitations

3. Bidirectional Path Sampling Techniques (*Křivánek*)
   (25 min)
   - Virtual point light rendering as a path sampling technique, limitations
   - Combining different path sampling techniques
   - Bidirectional path tracing
   - Limitations of local path sampling, SDS paths

4. Combining Photon Mapping and Bidirectional Path Tracing (*Georgiev*)
   (30 min)
   - (Progressive) photon mapping
   - Combining photon mapping with bidirectional path tracing
   - Consistency and convergence rate
   - Discussion: advantages & limitations

   Break

5. Markov Chain and Sequential Monte Carlo Methods (*Kaplanyan*)
   (25 min)
   - Markov chains
   - Metropolis-Hastings algorithm
   - Metropolis light transport
   - Normalization, start-up bias and stratification
   - Different mutation strategies and their properties
   - Light transport with sequential Monte Carlo

6. Comparison of Advanced Light Transport Methods (*Kaplanyan*)
   (30 min)
   - Ordinary Monte Carlo methods
   - Metropolis light transport with different mutation strategies
   - Energy redistribution path tracing
   - Markov chain progressive photon mapping
   - Population Monte Carlo light transport

7. Advanced Light Transport in the VFX/Archiviz industry (*Cañada*)
   (30 min)

- Stage of the Industry - the reasons for accurate light transport in practice.
- Current problems, solutions, and workarounds.
- What's next?

8. Conclusion (*all*)

Tutorial presenter information

Jaroslav Křivánek *Charles University, Prague*
jaroslav.krivanek@mff.cuni.cz, *http://cgg.mff.cuni.cz/~jaroslav/*

Jaroslav is an associate professor at Charles University in Prague. Jaroslav received his Ph.D. from IRISA/INRIA Rennes and the Czech Technical University (joint degree) in 2005. In 2003-2004 he was a research associate at the University of Central Florida. In 2008-2010 he was a Marie Curie research fellow at the Cornell University. His primary research interest is realistic rendering and light transport simulation.

Iliyan Georgiev *Saarland University*
georgiev@cs.uni-saarland.de, *http://iliyan.com/*

Iliyan is a graphics researcher at Saarland University, Germany, pursuing a Ph.D. degree. He received a B.Sc. degree in computer science from Sofia University, Bulgaria, and a M.Sc. degree in computer science from Saarland University. His primary research topics are high performance ray tracing and Monte Carlo methods for physically-based light transport simulation. His aspiration for practical rendering solutions has given him the opportunity to work with leading companies like Chaos Group (V-Ray), Disney, and Weta Digital.

Anton S. Kaplanyan *Karlsruhe Institute of Technology*
anton.kaplanyan@kit.edu, *http://cg.ibds.kit.edu/kaplanyan/*

Anton S. Kaplanyan is a graphics researcher at Karlsruhe Institute of Technology (KIT), Germany. Additionally he is pursuing his Ph.D. His primary research and recent publications are about advanced light transport methods in global illumination. Prior to joining academia Anton had been working for Crytek at various positions from senior R&D graphics engineer to lead researcher. He received his M.Sc. in Applied Mathematics at National Research University of Electronic Technology, Moscow in 2007.

Juan Cañada *Next Limit Technologies*
juan.canada@nextlimit.com

Juan joined Next Limit in 2001 to work in the Realflow development team and later he moved to the newborn Maxwell research team. Since then Juan held several positions in the team, leading it since 2007. He holds a bachelor's degree in Mechanical Engineering and a degree in Environmental Sciences.

Previous appearance of the presented material

This tutorial is derived from the SIGGRAPH 2013 course “Recent Advances in Light Transport Simulation: Theory & Practice”. Based on the positive feedback we have received on the course we have been invited to bring the material to the EUROGRAPHICS 2014 audience.

Presentation materials and tutorial notes

Up-to-date materials are available form http://cgg.mff.cuni.cz/~jaroslav/papers/2014-ltstutorial/index.htm.

Jaroslav Křivánek:

Introduction & Welcome

PATH INTEGRAL METHODS FOR LIGHT TRANSPORT SIMULATION

THEORY & PRACTICE

Jaroslav Křivánek
Charles University Prague

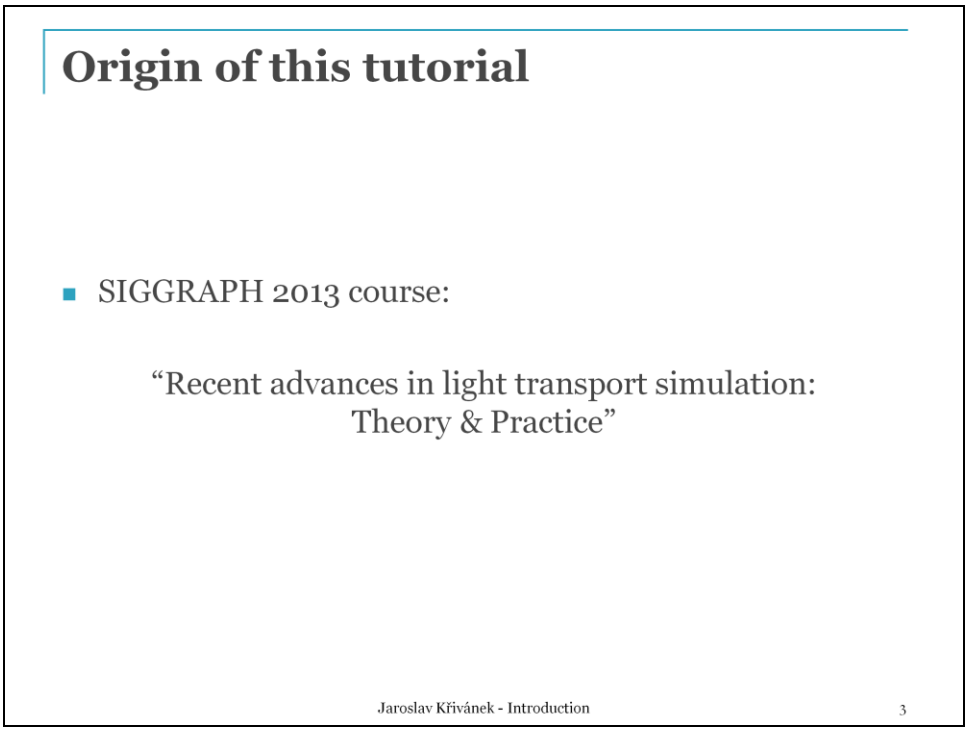
Iliyan Georgiev
Saarland University

Anton Kaplanyan
KIT

Juan Cañada
Next Limit Technologies

INTRODUCTION

Jaroslav Křivánek
Charles University in Prague



• This tutorial is derived from the SIGGRAPH 2013 course with the title "Recent advances in light transport simulation: Theory & Practice"

• When I first though about proposing that course, I wanted to give it a title that read „Path integral formulation of light transport and applications".

• The co-presenters advised me that this would be a bit too technical and could scare people away.

• So we changed the title for the SIGGRAPH submission, but in fact, the course content hasn't changed much at all: it still put a lot of emphasis on the *path integral formulation* of light transport and all the great algorithms it allows us to develop.

• For tutorial we felt a bit bolder and put the "path integral" right in the title. After all, that is what we will be talking about most of the time.

Light transport – Global illumination

Archviz

Movies

Image courtesy of Columbia Pictures.
© 2006 Columbia Pictures Industries, Inc.

Jaroslav Křivánek - Introduction 4

• Light transport simulation is an essential component of rendering realistic images with global illumination.

• It's been a standard tool in architectural and product visualization for many years now, with maxwell renderer being one of the first commercially available solutions based on unbiased Monte Carlo simulation.

• Its use in movies picked up later due to technical difficulties (large scenes, tighter rendering time budget).

Light transport – Global illumination

Movies

- **2002, Shrek 2**
  (PDI/Dreamworks)
  - 1 bounce indirect

- **2006, Monster House**
  (Sony Imageworks)
  - **Full light transport**
    (path traced)
  - Arnold renderer

Image courtesy of Columbia Pictures.
© 2006 Columbia Pictures Industries, Inc.

Jaroslav Křivánek - Introduction 5

• The use of global illumination in feature film production started with PDI/Dreamwork's Shrek 2. They used irradiance caching to compute a single bounce of diffuse indirect illumination – so not really the full light transport.

• The Monster House in 2006 was probably the time that ray-based, accurate Monte Carlo light transport simulation was used in movie production.

• It was rendered with the Arnold renderer – a brute force path tracer developed by Marcos Fajardo in collaboration with Sony Imageworks.

Light transport – Global illumination

Movies

- **2006, Monster House** (Sony Imageworks)
  - **Full light transport** (path traced)
  - Arnold renderer

Image courtesy of Columbia Pictures.
© 2006 Columbia Pictures Industries, Inc.

- **Full light transport simulation**
  - Accuracy
  - Ease of use
  - **Visual consistency**

Jaroslav Křivánek - Introduction 6

• Arnold has in fact started a quiet revolution, where most VFX and animation studios are nowadays shifting toward rendering solutions based on physically plausible light transport simulation.

• Advantages of this approach are indisputable

  • improved accuracy

  • easier rendering set up – no need for specialized solutions for different illumination effects

  • guaranteed visual consistency – the most important thing in movies!

• The shift in movie production toward physically based light transport underlines the importance of research and development in this area. It is also one of the important motivations behind this course.

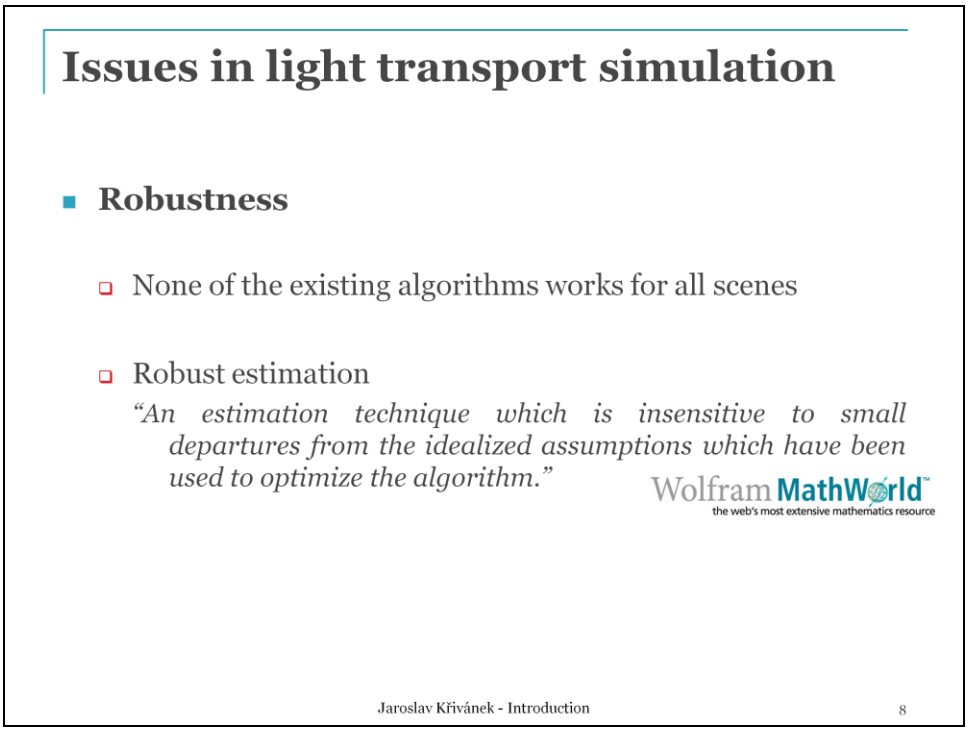
Light transport – Global illumination

- **More information**
  - "The State of Rendering"

- **Full light transport simulation**
  - Accuracy
  - Ease of use
  - **Visual consistency**

• A fairly detailed account on the state of rendering in the VFX community is given in a recent fxguide article "The State of Rendering".

• They also mention that the Vertex Connection and Merging algorithm will be used in PRMan 19 (http://cgg.mff.cuni.cz/~jaroslav/papers/2012-vcm/index.htm).



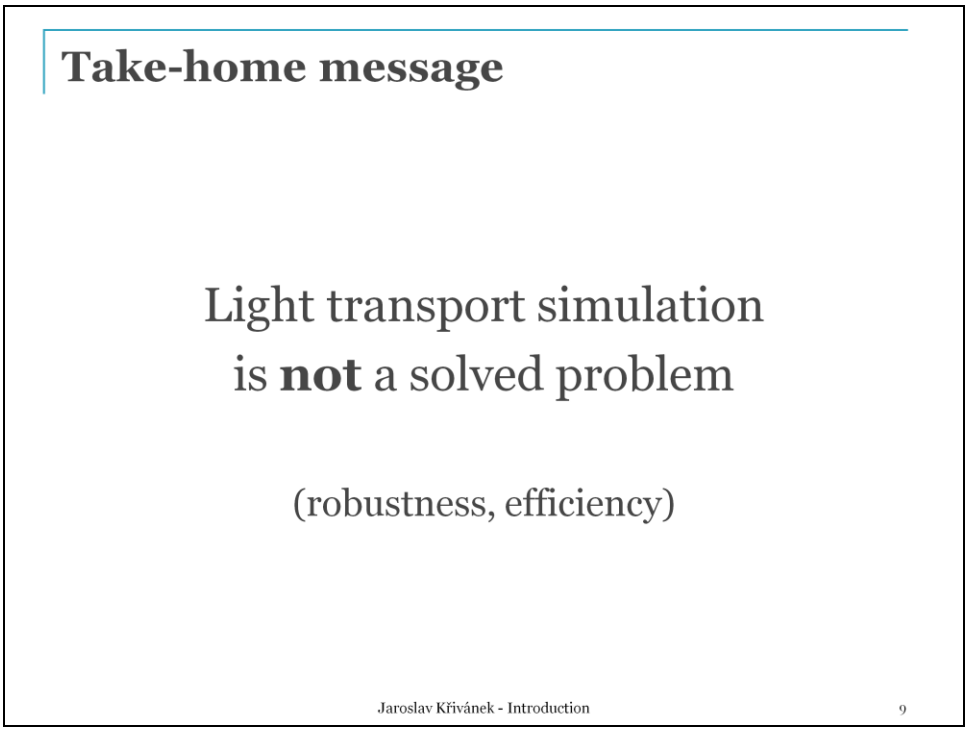
• A number of light transport algorithms exist, such as path tracing (PT), bidirectional path tracing (BPT), photon mapping (PM), or Metropolis light transport (MLT); and there are many variants of these algorithms.

• However, the single most pressing issue with all of these solutions is that none of them really works for all practical scenes.

• In other words, these solutions are not **robust** enough.

• Robustness of a statistical estimator (such as our rendering process) is defined by Wolframs MathWorld as follows: "*An estimation technique which is insensitive to small departures from the idealized assumptions which have been used to optimize the algorithm.*"

• In rendering, it means that a robust algorithm should be reasonably efficient for **any** input scene. The current light transport algorithms unfortunately do not exhibit this desirable feature.



• So in spite of the amazing results that we are able to produce, light transport simulation (and, more generally, rendering) is definitely not a solved problem (despite what we can hear here and there).

• We need a robust solution that will minimize all the manual work and parameter tweaking that currently has to go into preparing a scene for rendering.

• We also need a general improvement in efficiency such that light transport simulation can be used in interactive application.

• I'm not saying this to start on a negative note. On the contrary, I'm saying this to motivate and encourage the present researchers to contribute to the interesting and exciting research area.

Recent advances

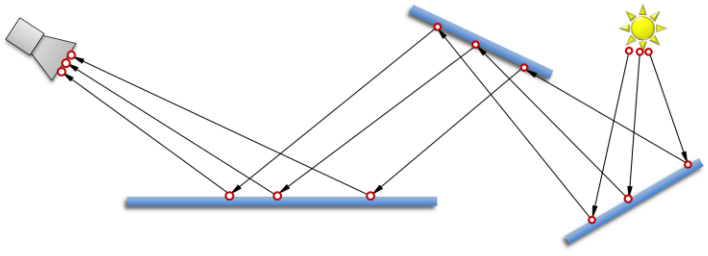
- Consistent density estimation – progressive photon mapping [Hachisuka et al. 08, 09, 10], [Knaus and Zwicker 11]
- Vertex Connection and Merging (VCM) = BPT + PPM [Georgiev et al. 12], [Hachisuka et al. 12]
- Improvements on Metropolis Light Transport [Jakob and Marchner 12], [Lehtinen et al. 13]

• Recently, there have been some significant advances in improving the robustness of light transport simulation that we will review in this course.

• These include for example progressive photon mapping, its robust combination with bidirectional path tracing (dubbed “vertex connection and merging”), as well as advances on Markov Chain Monte Carlo methods (Metropolis Light Transport).

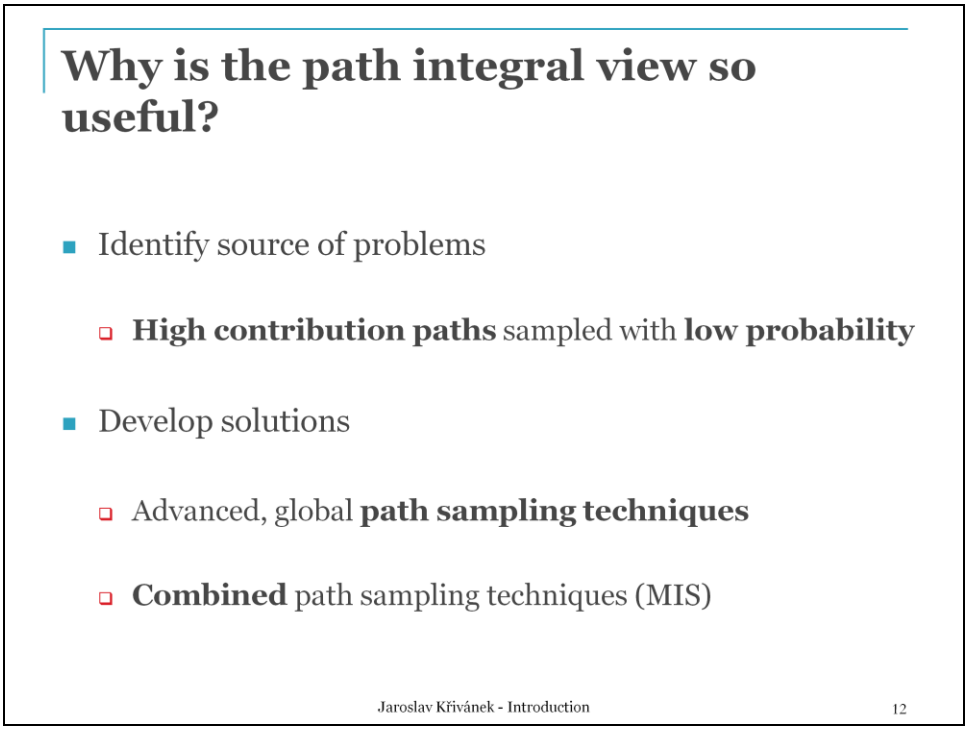
Common denominator

- **Path integral formulation** of light transport
  [Veach and Guibas 1995], [Veach 1997]



• The common to most of these techniques is the view of light transport as an integral over a space of paths.

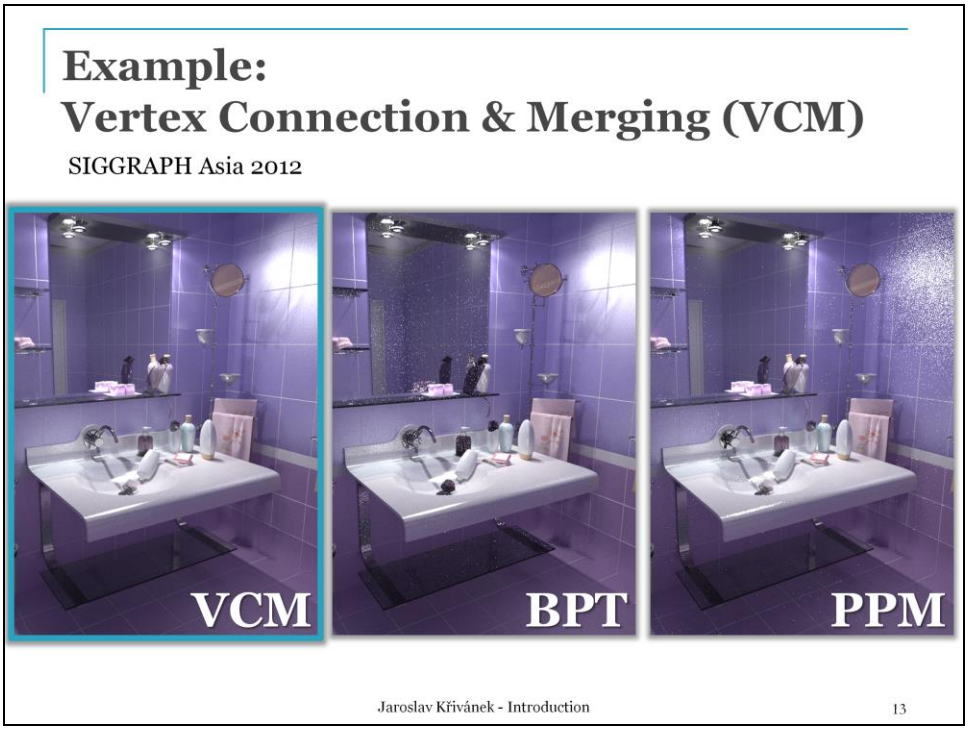
• This is why we will put a significant emphasis on this view of light transport in the course.



• So why is the path integral framework so useful?

• First, it allows us to identify the weaknesses of existing algorithms. With a little bit of simplification, we could say that all problems of current light transport solutions boil down to poor path sampling. Specifically, to the fact that some light transport paths that bring significant amount of energy from the light sources to the camera are not sampled with appropriately high probability. This means high estimator variance that produces noise & fireflies in the renderings.

• Second, the path integral framework allows us to develop new light transport algorithms based on advanced, global path sampling techniques, such as Metropolis Light Transport. It also provides us with a means of combining different path sampling techniques in a provably good way using Multiple importance sampling.



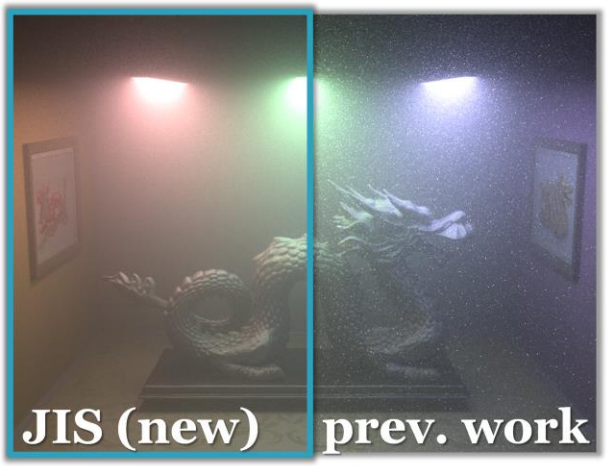
• As an example of the robust combination of path sampling techniques, in the recent Vertex Connection and Merging algorithm, that I co-developed, it was only through re-formulating photon mapping in the path integral framework that we were ably to robustly combine it with bidirectional path tracing and obtain the nice results that you can see on the slide.

• So the VCM algorithm is a very tangible example of the strength of the path integral framework.

• For more details, see http://cgg.mff.cuni.cz/~jaroslav/papers/2012-vcm/index.htm or the 3rd part of the course.

Example:
Joint Importance Sampling (JIS)

SIGGRAPH Asia 2013



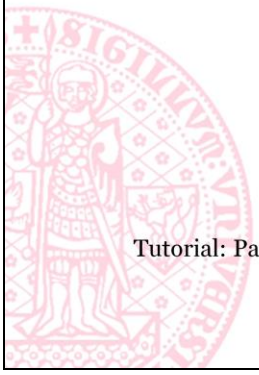
• Another example that I co-developed, and that I'm particularly excited about because it just got accepted to SIGGRAPH Asia, is our work on Joint Importance Sampling for low-order volumetric scattering. Again, it is through a treatment in the path integral framework that we were able to develop new path sampling techniques for low-order scattering in media that achieve orders of magnitude variance reduction in some cases.

Course outline

- **2:10 pm ... Path Integral Formulation of Light Transport** (*Jaroslav Křivánek)*
- **2:35 pm ... Bidirectional Path Sampling Techniques** (*Jaroslav Křivánek)*
- **2:55 pm ... Vertex Connection and Merging** (*Iliyan Georgiev)*
- **3:30 pm ... Break** (15 minutes)

Course outline

- **3:30 pm ... Break** (15 minutes)
- **3:45 pm ... Markov Chain and Sequential Monte Carlo Methods** (*Anton Kaplanyan)*
- **4:10 pm ... Comparison of Advanced Light Transport Methods** (*Anton Kaplanyan)*
- **4:40 pm ... Advanced Light Transport in the VFX/Archiviz industry** (Juan Cañada)



Comments?
Questions?

Tutorial: Path Integral Methods for Light Transport Simulation

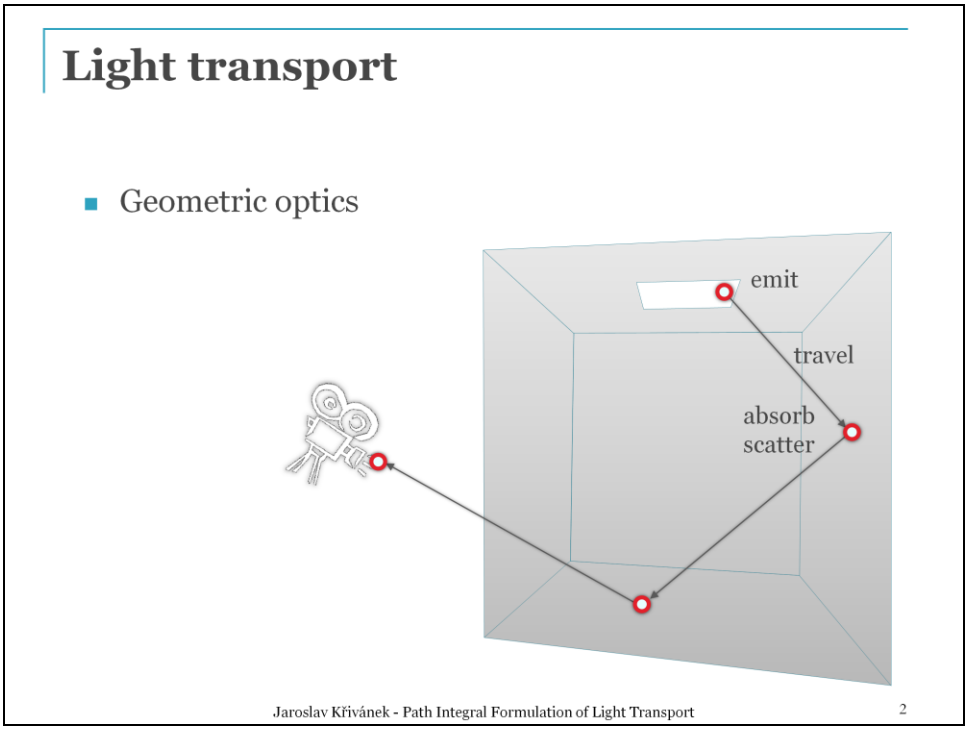
Jaroslav Křivánek – Introduction

Jaroslav Křivánek:

Path Integral Formulation of Light Transport

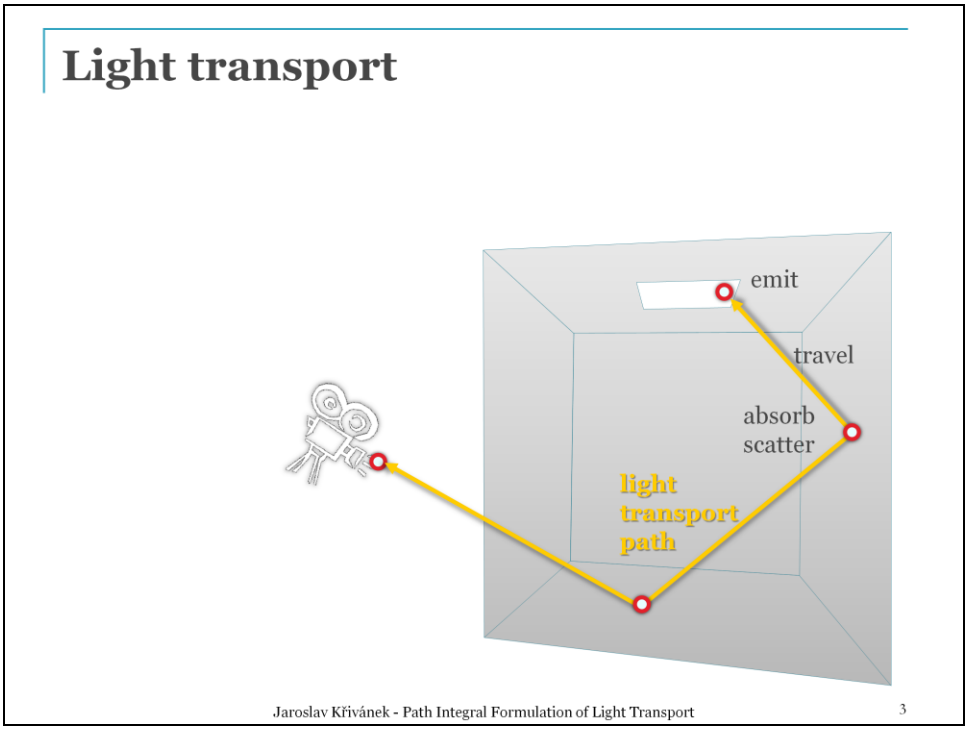
PATH INTEGRAL FORMULATION OF LIGHT TRANSPORT

Jaroslav Křivánek
Charles University in Prague
http://cgg.mff.cuni.cz/~jaroslav/



• In the real world, light sources emit light particles, which travel in space, scatter at objects (potentially multiple times) until they are absorbed.

• On their way, they might hit the sensor of the camera which will record the light contribution.

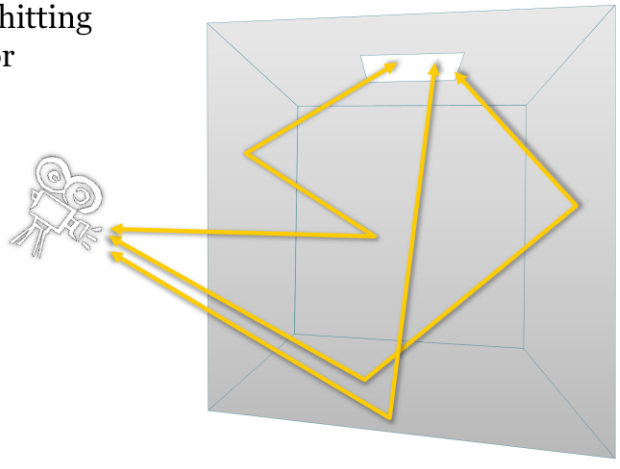


• The light particles travel along trajectories that we call “light transport paths”.

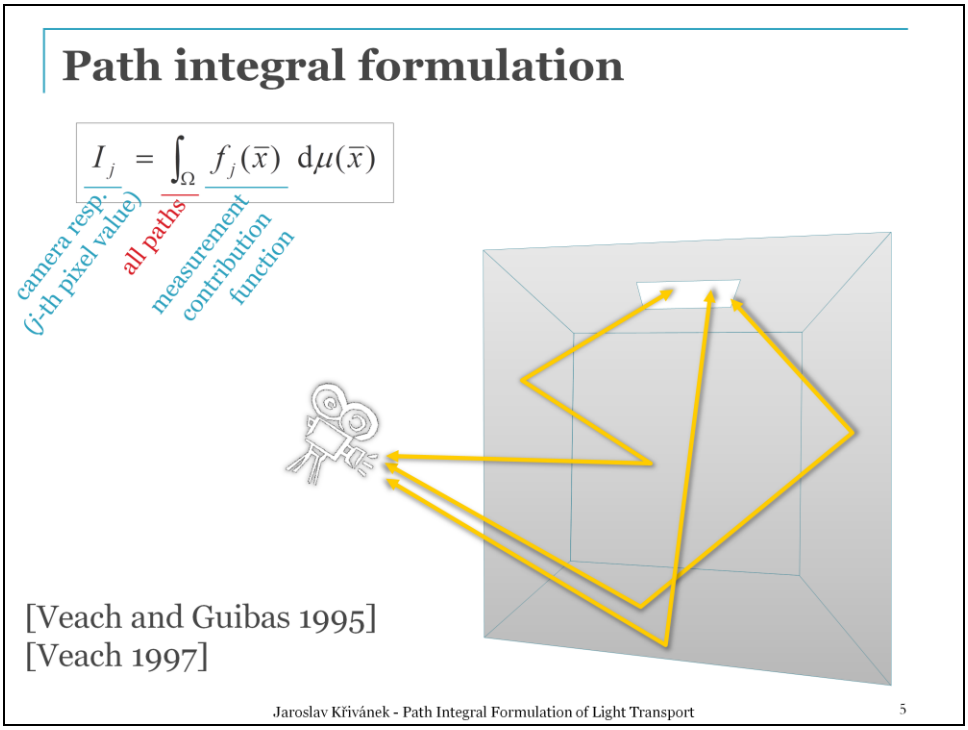
• In an environment consisting of opaque surfaces in vacuum, these paths are polylines whose vertices correspond to scattering (reflection) at surfaces, and the straight edges correspond to light travelling the in free space.

Light transport

- **Camera response**
  - all paths hitting the sensor



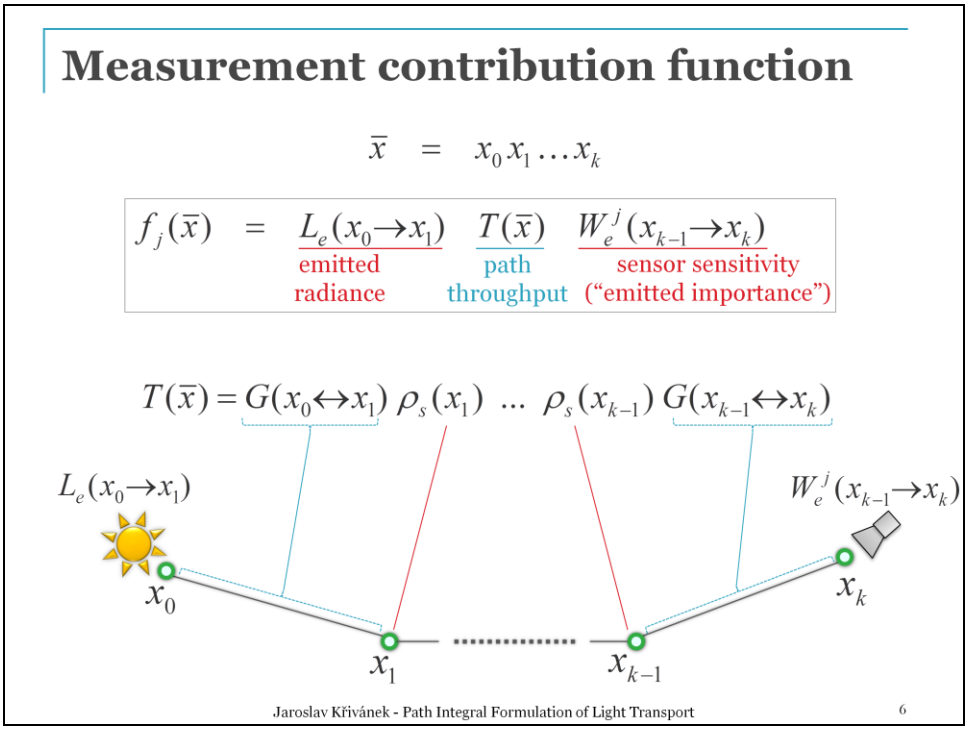
• The final response of the camera is due to all the light particles – travelling over all possible paths – that hit the camera sensor during the shutter open period.



• The path integral formulation of light transport formalizes this idea by writing the camera response (which, in image synthesis will be the value of a pixel) as an integral over all light transport paths of all lengths in the scene.

• The integrand of this integral is the so called “measurement contribution function”.

• The measurement contribution function of a given path encompasses the “amount” of light emitted along the path, the light carrying capacity of the path, and the sensitivity of the sensor to light brought along the path.



• Let us now look at a more formal definition of the measurement contribution for a light path.

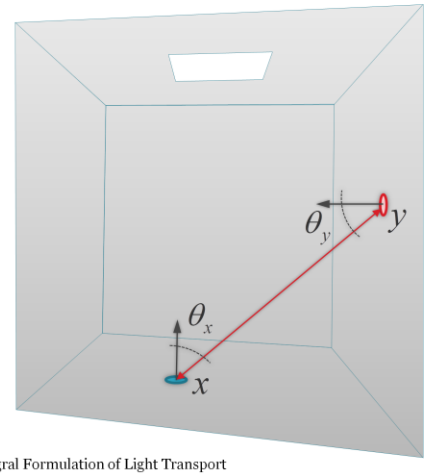
• As I already mentioned, a light transport path is a polyline with vertices corresponding to light scattering on surfaces.

• We write the path simply as a sequence of vertices, in the order of the light flow. So the first vertex corresponds to light emission at the light source and the last vertex to light measurement at the sensor.

• The measurement contribution function $f(x)$ for a path x of length k is defined as the emitted radiance L_e at the first vertex, times the sensitivity (or "emitted importance") at the last vertex, times the path throughput $T(x)$. The throughput is defined as the product of the geometric G and scattering terms associated with the path edges and interior vertices, respectively.

Geometry term

$$G(x \leftrightarrow y) = \frac{|\cos\theta_x||\cos\theta_y|}{\|x-y\|^2} V(x \leftrightarrow y)$$



Jaroslav Křivánek - Path Integral Formulation of Light Transport 7

• The geometry term, or point-to-point form factor, of a path edge expresses the throughput of the edge due to geometry configuration of the scene, and is given by the product of the inverse-squared-distance, cosine factor at the vertices and the visibility term.

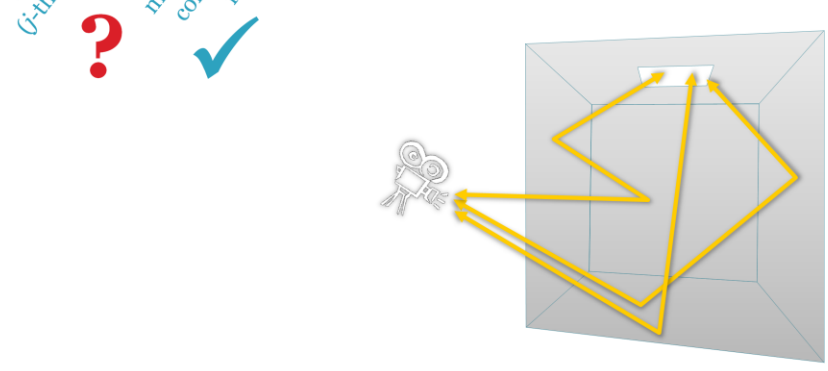
Path integral formulation

$$I_j = \int_\Omega f_j(\bar{x})\ \mathrm{d}\mu(\bar{x})$$

camera resp. (j-th pixel value)

all paths

measurement contribution function



- Back to the path integral…
- We now know the meaning of the integrand – the path contribution function – but the integration domain “all path” needs more clarification.

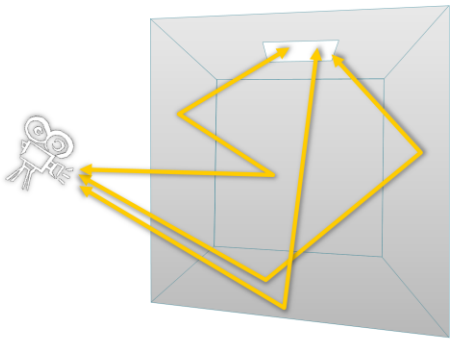
Path integral formulation

$$I_j = \int_\Omega f_j(\bar{x})\ \mathrm{d}\mu(\bar{x})$$

$$= \sum_{k=1}^{\infty} \int_{M^{k+1}} f_j(x_0 \ldots x_k)\ \mathrm{d}A(x_0) \ldots \mathrm{d}A(x_k)$$

all path lengths

all possible vertex positions



Jaroslav Křivánek - Path Integral Formulation of Light Transport 9

- The path integral actually hides an infinite sum over all possible path lengths.
- Each summand of this sum is a multiple integral, where we integrate the path contribution over all possible positions of the path vertices.
- So each of the integrals is taken over the Cartesian product of the surface of the scene with itself, taken k+1 times (=number of vertices for a length-k path.)
- The integration measure is the area-product measure, i.e. the natural surface area, raised to the power of k+1.

Path integral

$$I_j = \int_\Omega f_j(\bar{x})\, \mathrm{d}\mu(\bar{x})$$

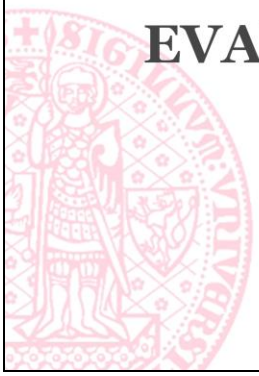
pixel value — all paths — contribution function

Jaroslav Křivánek - Path Integral Formulation of Light Transport 10

- We now have a formula for pixel values that has a form of a simple (though infinite-dimensional) integral.
- To render images, we need to numerically evaluate this integral for all image pixels.

RENDERING :

EVALUATING THE PATH INTEGRAL



• So the next section of this presentation well be devoted to numerical evaluation of the path integral.

Path integral

$$I_j = \int_\Omega f_j(\bar{x})\, \mathrm{d}\mu(\bar{x})$$

pixel value — all paths — contribution function

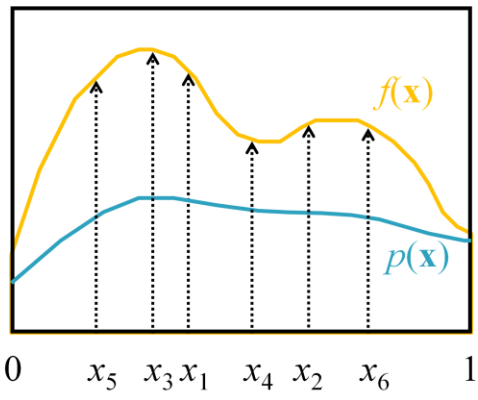
- **Monte Carlo integration**

Jaroslav Křivánek - Path Integral Formulation of Light Transport 12

- We will use Monte Carlo integration for this purpose.

Monte Carlo integration

- General approach to numerical evaluation of integrals



Integral:

$$I = \int f(x)\mathrm{d}x$$

Monte Carlo estimate of I:

$$\langle I \rangle = \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{p(x_i)}; \quad x_i \propto p(x)$$

Correct „on average“:

$$E[\langle I \rangle] = I$$

Jaroslav Křivánek - Path Integral Formulation of Light Transport 13

• Let me briefly review the basic elements of MC integration.

• Suppose we are given a real function $f(x)$ and we want to compute the integral Int $f(x)$ dx over some domain (in this example we use the interval [0,1] for simplicity, but the domain can be almost arbitrary.)

• The Monte Carlo integration procedure consists in generating a ‘sample’, that is, a random x-value from the integration domain, drawn from some probability distribution with probability density $p(x)$. In the case of path integral, the x-value is an entire light transport path.

• For this sample x_i, we evaluate the integrand $f(x_i)$, and the probability density $p(x_i)$.

• The ratio $f(x_i) / p(x_i)$ is an estimator of the integrand. To make the estimator more accurate (i.e. to reduce its variance) we repeat the procedure for a number of random samples $x_1, x_2, \ldots, x_N$, and average the result as shown in the formula on the slide.

• This procedure provides an unbiased estimator of the integrand, which means that “on average”, it produces the correct result i.e. the integral that we want to compute.

MC evaluation of the path integral

Path integral

$$I_j = \int_\Omega f_j(\bar{x})\, \mathrm{d}\mu(\bar{x})$$

MC estimator

$$\left\langle I_j \right\rangle = \frac{f_j(\bar{x})}{p(\bar{x})}$$

- Sample path $\bar{x}$ from some distribution with PDF $p(\bar{x})$?
- Evaluate the probability density $p(\bar{x})$?
- Evaluate the integrand $f_j(\bar{x})$ ✓

Jaroslav Křivánek - Path Integral Formulation of Light Transport 14

• Thanks to the formal simplicity of the path integral formulation, applying Monte Carlo integration is really a more-or-less mechanical process.

• For each pixel, we need to repeatedly evaluate the estimator shown at the top right of the slide and average the estimates.

• This involves the following three steps:

  • First, we need to draw (or sample, or generate – all are synonyms) a random light transport path *x* in the scene (connecting a light source to the camera).

  • Then we need to evaluate the probability density of this path, and the contribution function.

  • Finally, we simply evaluate the formula at the top of the slide.

• Evaluating the path contribution function is simple – we have an analytic formula for this that takes a path and returns a number – the path contribution.

• However, we have not discussed so far how paths can be sampled and how the PDF of the resulting path can be evaluated.

Path sampling

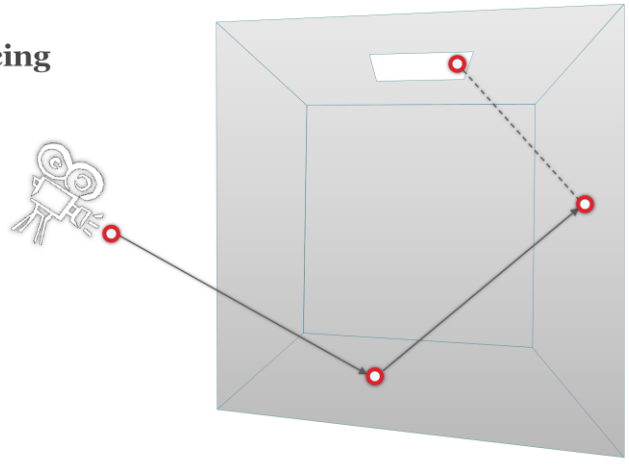
- Algorithms = different path sampling techniques

• Path sampling techniques and the induced path PDF are an essential aspect of the path integral framework.

• In fact, from the point of view of the path integral formulation, the only difference between many light transport simulation algorithms are the employed path sampling techniques and their PDFs.

Path sampling

- Algorithms = different path sampling techniques
  - **Path tracing**

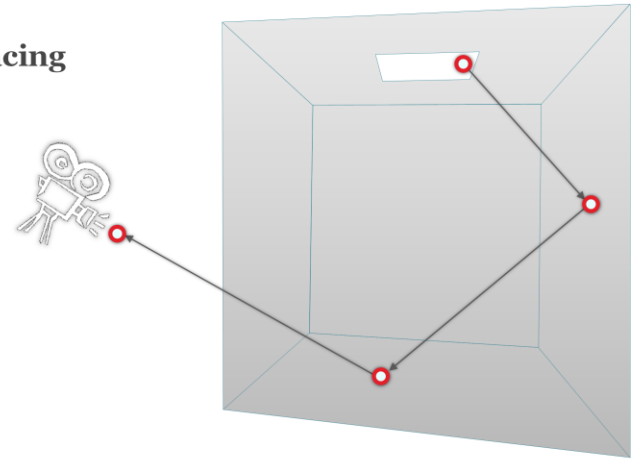


Jaroslav Křivánek - Path Integral Formulation of Light Transport 16

• For example, path tracing samples paths by starting at the camera sensor, and extending the path by BRDF importance sampling, and possibly explicitly connecting a to a vertex sampled on the light source.

Path sampling

- Algorithms = different path sampling techniques
  - **Light tracing**



• Light tracing, on the other hand, starts paths at the light sources.

Path sampling

- Algorithms = different path sampling techniques
- **Same** general form of **estimator**

$$\langle I_j \rangle = \frac{f_j(\bar{x})}{p(\bar{x})}$$

- **No importance transport, no adjoint equations!!!**

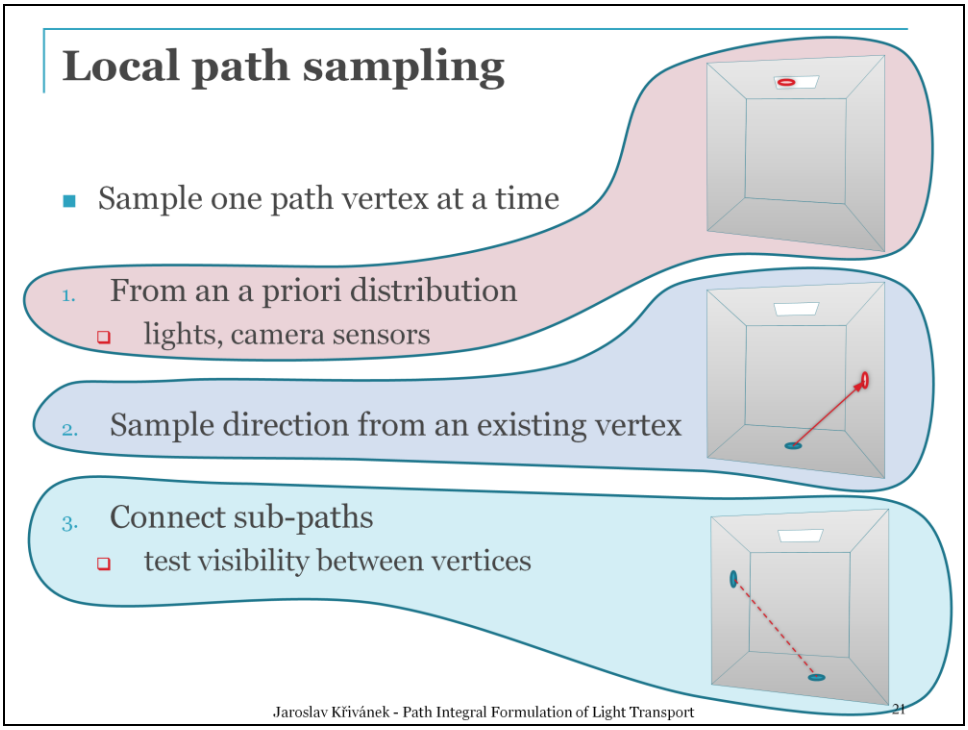
• Though path tracing and light tracing may seem like very different algorithms, from the path integral point of view they are essentially the same.

• The only difference is the path sampling procedure, and the associated path PDF.

• But the general Monte Carlo estimator is exactly the same – only the PDF formula changes.

• Without the path integral framework, we would need equations of importance transport to formulate light tracing, which can get messy.

• So how exactly do we sample the paths and how do we compute the path PDF?



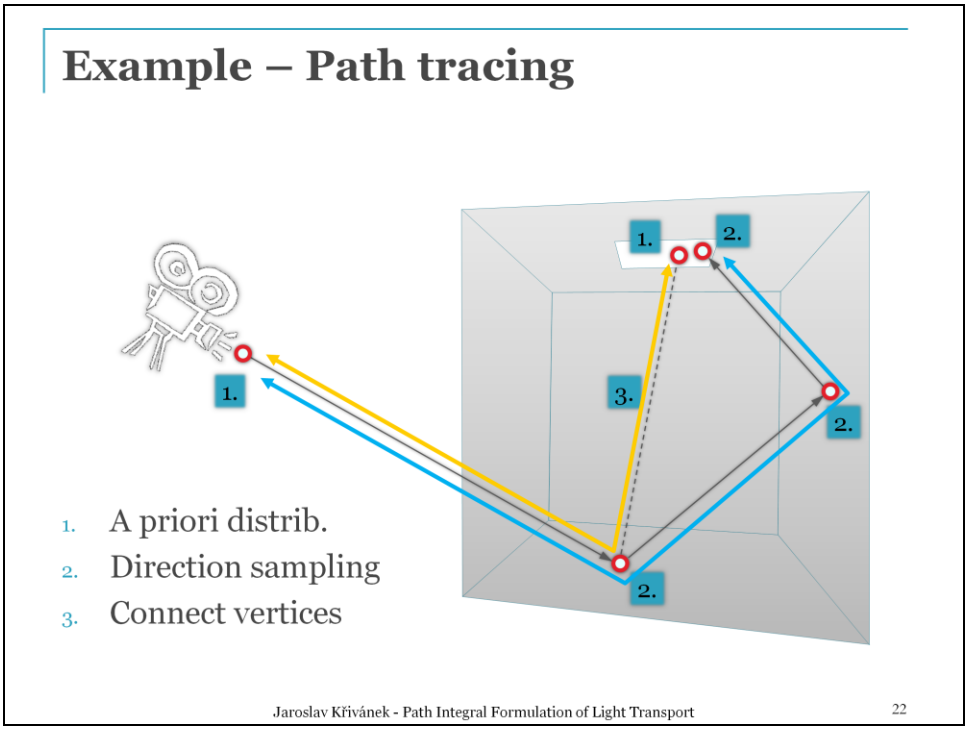
• Most of the practical algorithms rely on local path sampling, where paths are build by adding one vertex at a time until a complete path is built.

• There are three common basic operations.

  • First, we can sample a path vertex from an a priori given distribution over scene surfaces. We usually employ this technique to start a path either on a light source or on the camera sensor.

  • Second, given a sub-path that we've already sampled with a vertex at its end, we may sample a direction from that vertex, and shoot a ray in this direction to obtain the next path vertex.

  • Finally, given two sub-paths, we may connect their end-vertices to form a full light transport path. This technique actually does not add any vertex to the path. It is more or less a simple visibility check to see if the contribution function of the path is non-zero.



• Let us see how these three basic operations are used in a simple path tracer.

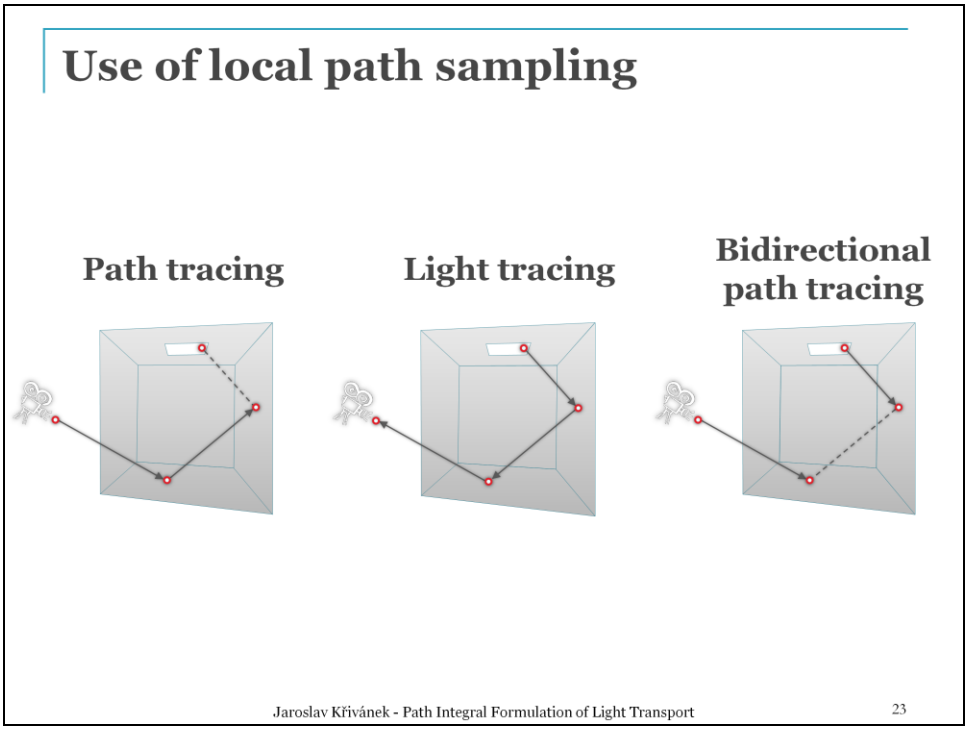
• First, we generate a vertex on the camera lens, usually from a uniform distribution over the lens surface – so this corresponds to operation 1.

• Second, we pick a random direction form this point such that it passes through the image plane, and shoot a ray to extend the path. This is operation 2.

• Third, we may generate an independent point on the light source (operation 1) and test visibility (operation 3) to form a complete light transport path.

• We could also continue the path by sampling a random direction and shooting a ray (operation 2), and eventually hit the light source to complete the path.

• An important thing to notice is that one single primary ray from the camera actually creates a full family of light transport paths. These path are correlated, because they share some of the path vertices, but they are distinct entities in the path space.



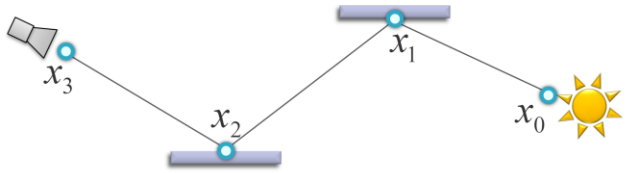
• These same basic operations are used to construct paths in light tracing and bidirectional path tracing.

Probability density function (PDF)

path PDF

$$p(\bar{x}) = p(x_0, \ldots, x_k)$$

joint PDF of path vertices



Jaroslav Křivánek - Path Integral Formulation of Light Transport 24

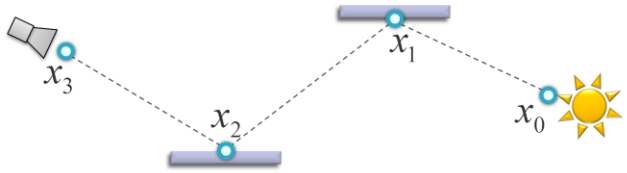
- Not that we know how to construct a path, we need to evaluate its PDF so that we can plug it into the MC estimator.
- In general the PDF of a light path is simply the **joint** PDF of the path vertices.
- That is to say, the PDF that the first vertex is where it is *and* the second vertex is where it is, etc.

Probability density function (PDF)

path PDF

$$p(\bar{x}) = p(x_0, \ldots, x_k)$$

joint PDF of path vertices



Probability density function (PDF)

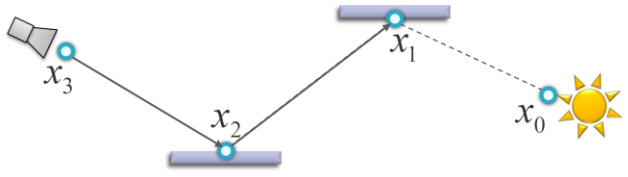
path PDF

$$p(\bar{x}) = p(x_0, \ldots, x_k) = p(x_3)\, p(x_2 \mid x_3)\, p(x_1 \mid x_2)\, p(x_0)$$

joint PDF of path vertices

product of (conditional) vertex PDFs

Path tracing example:



Jaroslav Křivánek - Path Integral Formulation of Light Transport 26

- The joint path PDF is given by the product of the conditional vertex PDF.
- To see what this means, let us again take the example of path tracing, where we build a path starting from the camera.
- Vertex x_3 comes from an a priori distribution $p(x_3)$ over the camera lens (usually uniform; or the delta distribution for a pinhole camera).
- Vertex x_2 is sampled by generating a random direction from x_3 and shooting a ray. This induces a PDF for x_2, $p(x_2 \mid x_3)$, which is in fact conditional on vertex x_3.
- The same thing holds for vertex x_1, which is sampled by shooting a ray in a random direction from x_2.
- Finally, vertex x_0 on the light source might be sampled from an uniform distribution over the light source area with pdf $p(x_0)$, independently of the other path vertices.
- The full joint PDF is given by the product of all these individual terms.

Probability density function (PDF)

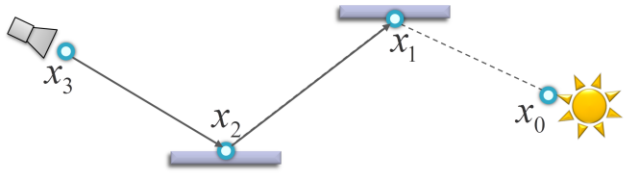
path PDF

$$p(\bar{x}) = p(x_0, \ldots, x_k) = p(x_3)\,p(x_2)\,p(x_1)\,p(x_0)$$

$p(\bar{x})$: **joint PDF** of path vertices

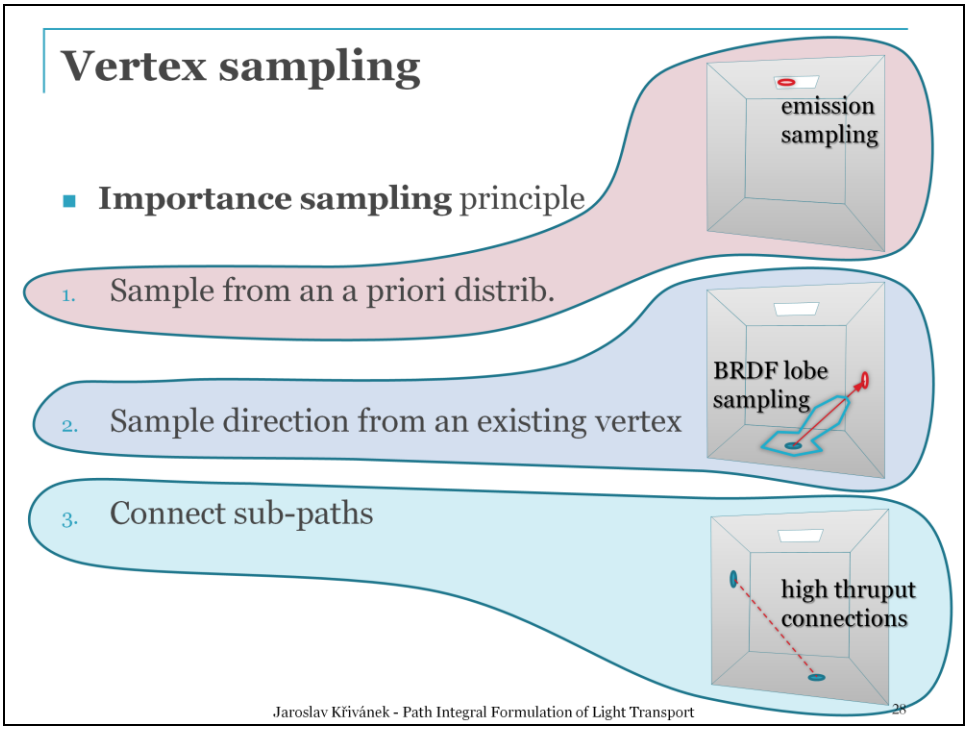
$p(x_3)\,p(x_2)\,p(x_1)\,p(x_0)$: **product** of (conditional) vertex PDFs

Path tracing example:



Jaroslav Křivánek - Path Integral Formulation of Light Transport 27

• It is customary to simplify this somewhat pedantic notation and leave out the conditional signs. Nonetheless, it is important to keep in mind that the path vertex PDFs for vertices that are not sampled independently are indeed conditional PDFs.



• In accordance with the principle of importance sampling, we want to generate full paths paths from a distribution with probability density proportional to the measurement contribution function. That is, high-contribution paths should have proportionally high probability of being sampled.

• Local path sampling takes an approximate approach, where each local sampling operation tries to importance sample the terms of the contribution function associated with the vertex being sampled.

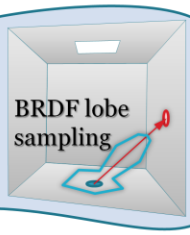
• For example, when starting a path on the light source, we usually sample the initial vertex from a distribution proportional to the emitted power.

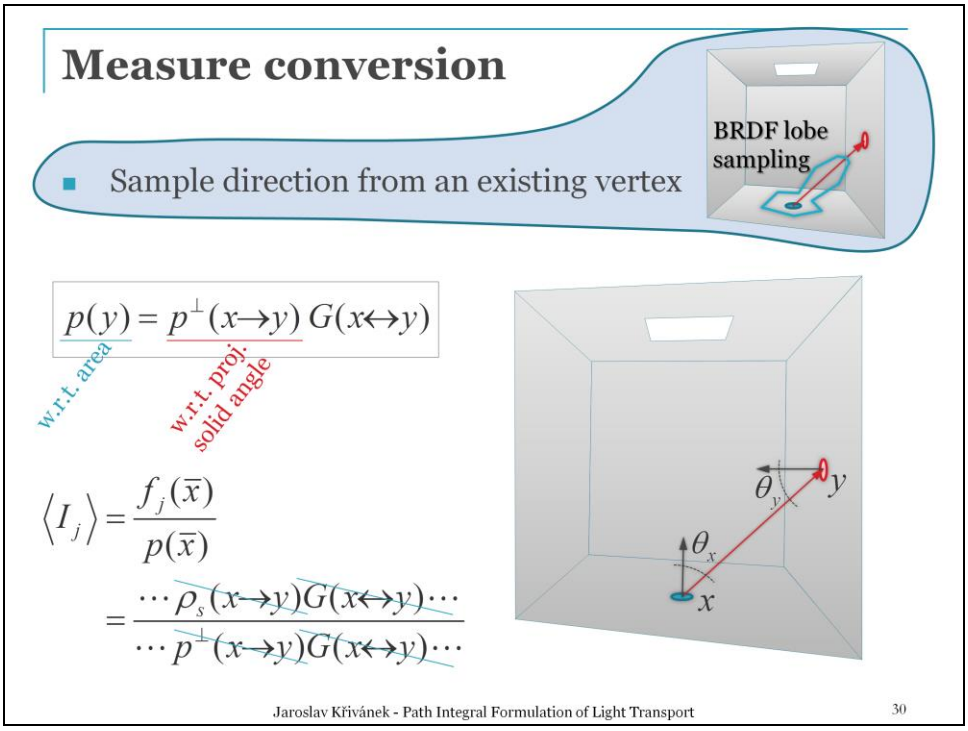
• When extending the path from an existing vertex, we usually sample the random direction proportionally to the BRDF at the vertex.

• Similarly, when connecting two sub-paths with an edge, we may want to prefer connections with high throughput (though this is rarely done in practice).

Vertex sampling

- Sample direction from an existing vertex



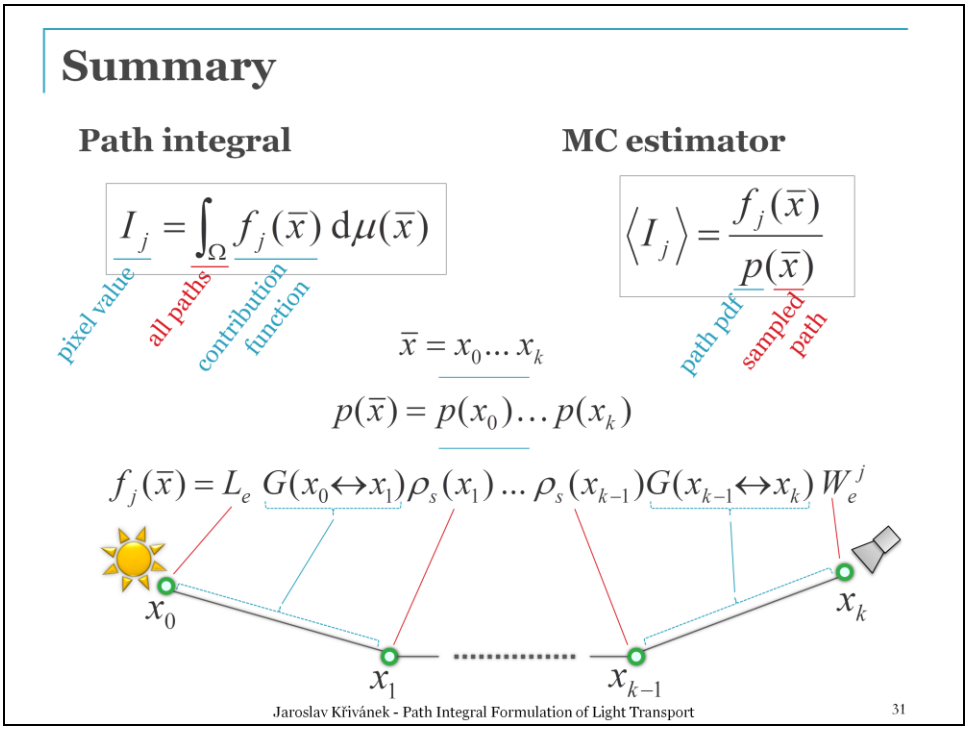


• There is one important technical detail associated with computing the path PDF for vertices created by direction sampling.

• The path integral is expressed with respect to the surface area measure – we are integrating over the surface of the scene – but the direction sampling usually gives the PDF with respect to the (projected) solid angle.

• The conversion factor from the projected solid angle measure to the area measure is the geometry factor.

• This means that any vertex generated by first picking a direction and then shooting a ray has the geometry factor of the generated edge importance sampled – the only geometry factor that are not importance sampled actually correspond to the connecting edges (operation 3 in local path sampling).



• At this point, we should summarize the mathematical development so far.

• The path integral formulation gives the pixel value as an integral over all light transport paths of all lengths.

• A light transport path is the trajectory of a light carrying particle. It is usually a polyline specified by its vertices.

• The integrand is the measurement contribution function, which is the product of emitted radiance, sensor sensitivity, and path throughput.

• Path throughput is the product of BRDFs at the inner path vertices and geometry factors at the path edges.

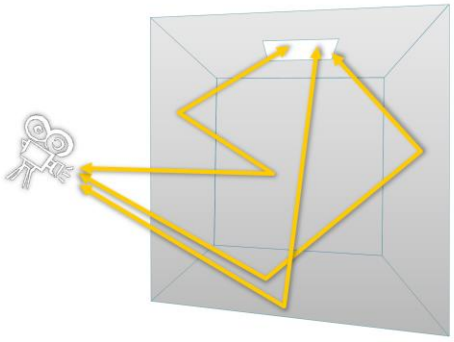
• To evaluate the integral, we use Monte Carlo estimator in the form shown at the top right of the slide.

• To evaluate the estimator, we need to sample a random path from a suitable distribution with PDF $p(x)$.

• We usually use local path sampling to construct the paths, where we add one vertex at a time. In this case, the full path PDF is given by the product of the (conditional) PDFs for sampling the individual path vertices.

Summary

- **Algorithms**
  - different path sampling techniques
  - different path PDF



Jaroslav Křivánek - Path Integral Formulation of Light Transport 32

• From the point of view of the path integral framework, different light transport algorithms (based on Monte Carlo sampling) only differ by the path sampling techniques employed.

• The different sampling techniques imply different path PDF and therefore different relative efficiency for specific lighting effects. This will be detailed in the next part of the course.

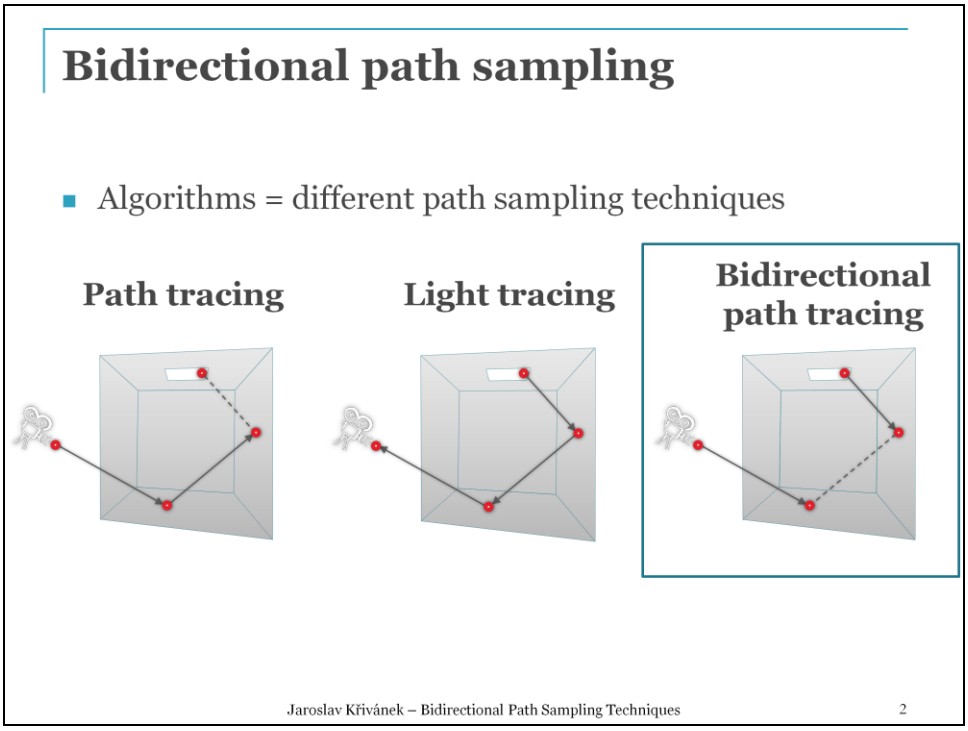
Time for questions…

Jaroslav Křivánek:

Bidirectional Path Sampling Techniques

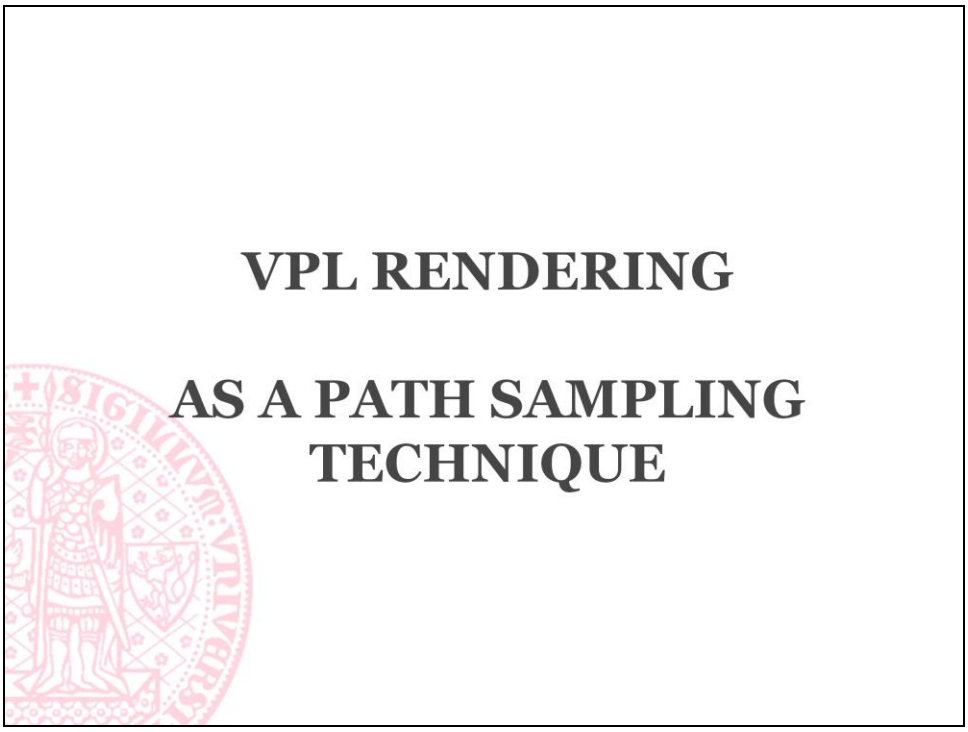
BIDIRECTIONAL PATH SAMPLING TECHNIQUES

Jaroslav Křivánek
Charles University in Prague
http://cgg.mff.cuni.cz/~jaroslav/



• In the previous part of the course, I said that from the point of view of the path integral framework, the major difference between different LTS algorithms is the employed path sampling technique.

• In this part of the course, we will focus on bidirectional path sampling techniques, that build a sub-path from the camera and from the light source and connect the two sub-paths to generate an entire path. Such sampling techniques are employed in the bidirectional path tracing algorithm.

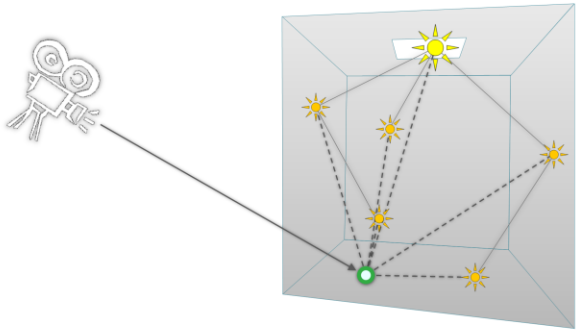


• But before I get to the description of bidirectional path tracing, I would like to start with Virtual Point Light Rendering (a.k.a. Instant radiosity) and show the advantages of looking at this algorithm through the prism of the path integral framework.

Instant radiosity – VPL rendering

[Keller 1997]

1. **Distribute VPLs**
2. **Accumulate VPL contributions**



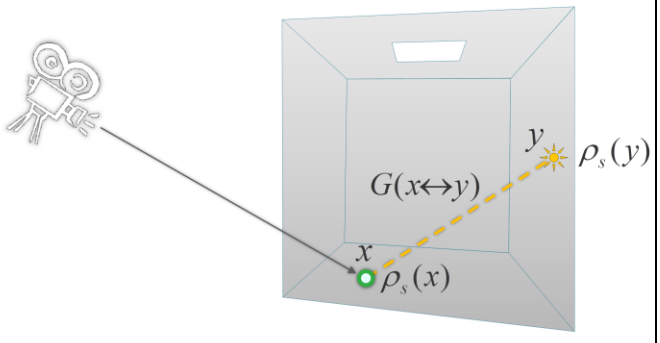
•The instant radiosity algorithm proceeds in two stages.

• In the first stage, we trace sub-paths starting from the light sources, depositing “virtual point lights” at every surface intersection.

• In the second stage, we render the image by computing the contribution from all the virtual point lights to the scene points seen through camera pixels.

VPL contribution

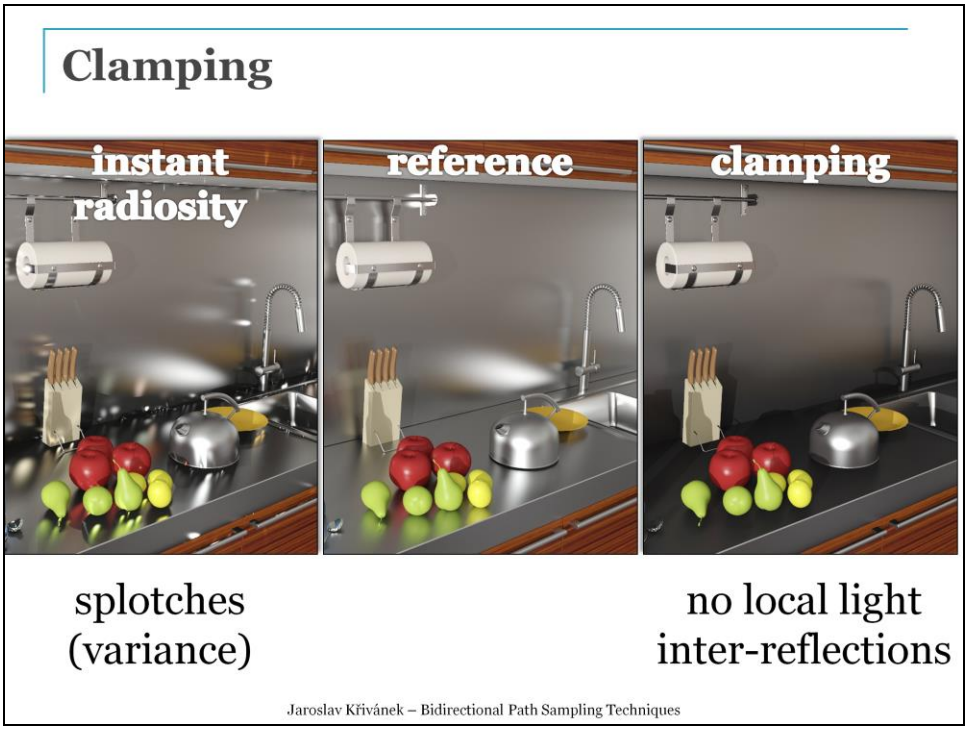
- High values for spiky BRDFs
- Diverges as $\| x - y \| \rightarrow 0$



Jaroslav Křivánek – Bidirectional Path Sampling Techniques 5

• The contribution of the VPL at *y* to a surface point at *x* is given by the product of the scattering term (BSDF) at the two vertices, the geometry term of the connecting edge, and the VPL energy.

• This expression can take one extremely large values for spiky BSDFs and when the points *x* and *y* approach each other (in fact, the expression diverges – goes to infinity – in the latter case).

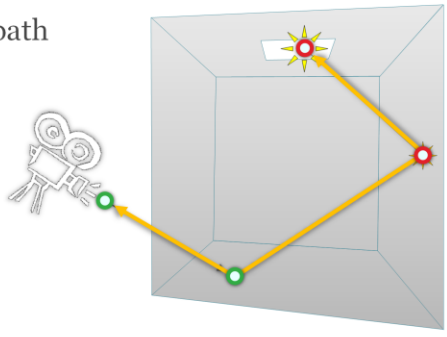


• The usual approach to deal with the singularities in VPL contribution is the so called *clamping*, where we simply limit the contribution of the VPL by some maximum allowed value.

• However, this is far from being an ideal solution because it removes a lot of energy from the scene, yielding darkening of surface and change of material appearance.

VPL rendering as a bidirectional path sampling technique

1. **Distribute VPLs**
   - = sample light sub-paths
2. **Camera ray**
   - = sample camera sub-path
3. **VPL contribution**
   - = sub-path connection



• Let's now look at the exact same process in the path integral framework.

• In the first step, we distribute the VPLs. Of course, that is nothing else than sampling sub-paths starting from a light source.

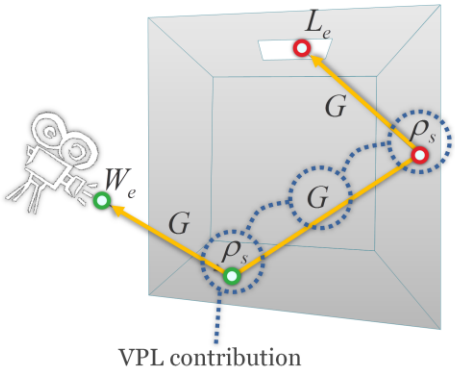
• Finding out a point visible through a pixel from the camera involves building a length-1 sub-path from the camera.

• And finally, evaluating the VPL contribution completes a full light transport path by connecting two sub-paths together.

VPL rendering as a bidirectional path sampling technique

- The usual path integral **estimator**

$$\langle I_j \rangle = \frac{f_j(\bar{x})}{p(\bar{x})}$$



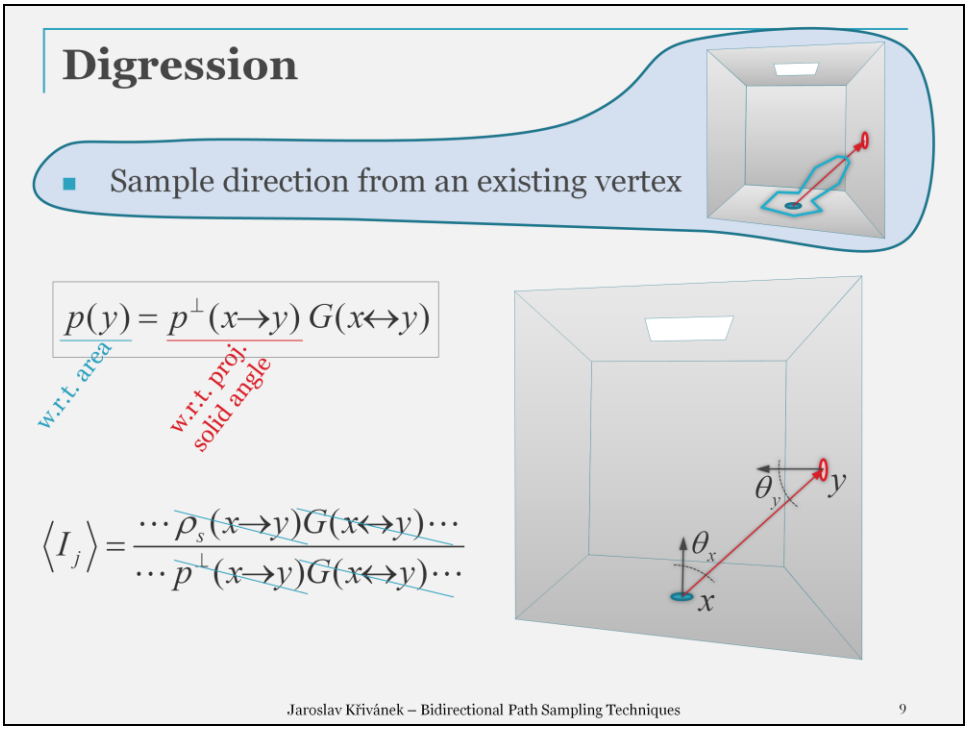
Jaroslav Křivánek – Bidirectional Path Sampling Techniques 8

• Once again, we use the exact same general form the of path integral estimator, that is the value of the measurement contribution function divided by the path PDF.

• Again, the measurement contribution function is given by the product of the emission, sensor sensitivity, BSDFs at the path vertices, and the geometry terms for the path segments.

• And notice that the factor of the contribution function associated with the VPL connection edge are exactly the terms that need to be evaluated when computing the contribution of a VPL.

• So how does the path PDF looks like?



• To see that, I will allow myself a little digression back to a slide I showed a couple of minutes ago.

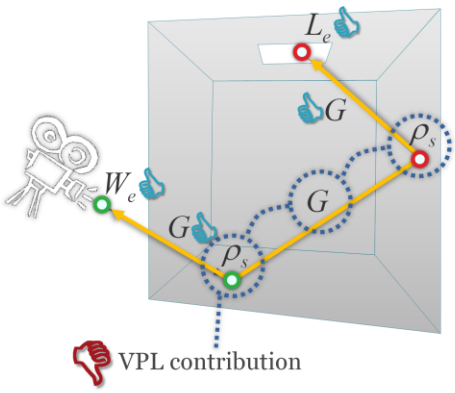
• On this slide I was showing that every time we sample a vertex, *y* in this example, by picking a random direction from another vertex, *x* here, and shooting a ray, we automatically importance sample the geometry term along the edge.

• However, the geometry term for edges constructed by connection are not importance sampled.

VPL rendering as a bidirectional path sampling technique

- **The usual estimator**

$$\langle I_j \rangle = \frac{f_j(\bar{x})}{p(\bar{x})}$$



Jaroslav Křivánek – Bidirectional Path Sampling Techniques 10

• So we see here that the geometry terms for the segments on the light and camera sub-path are importance sampled.

• Also, the radiance emission and sensor sensitivity are importance sampled because we usually pick the initial path vertex and initial direction proportional to these quantities.

• But notice that none of the quantities associated to the connecting edge are importance sample – Indeed, we just blindly connect two vertices. And that's exactly where all the problems with VPLs are coming from.

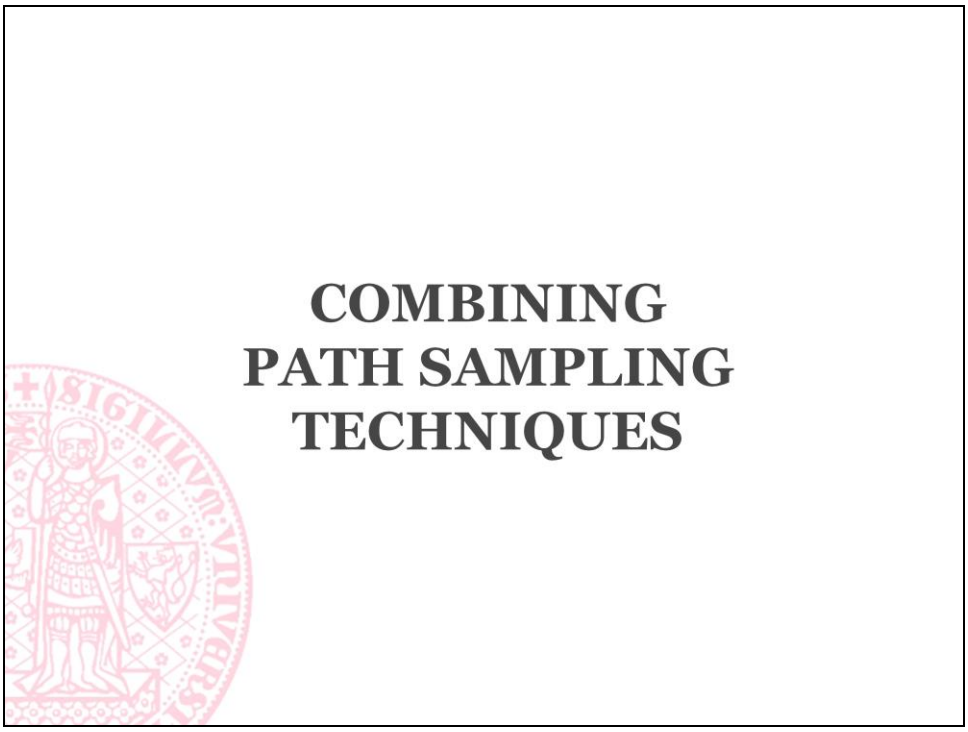
VPL rendering summary

- **VPL rendering** corresponds to a **bidirectional path sampling technique**
- Splotches = noise = variance
  - Due to **bad path sampling**
  - Correlation

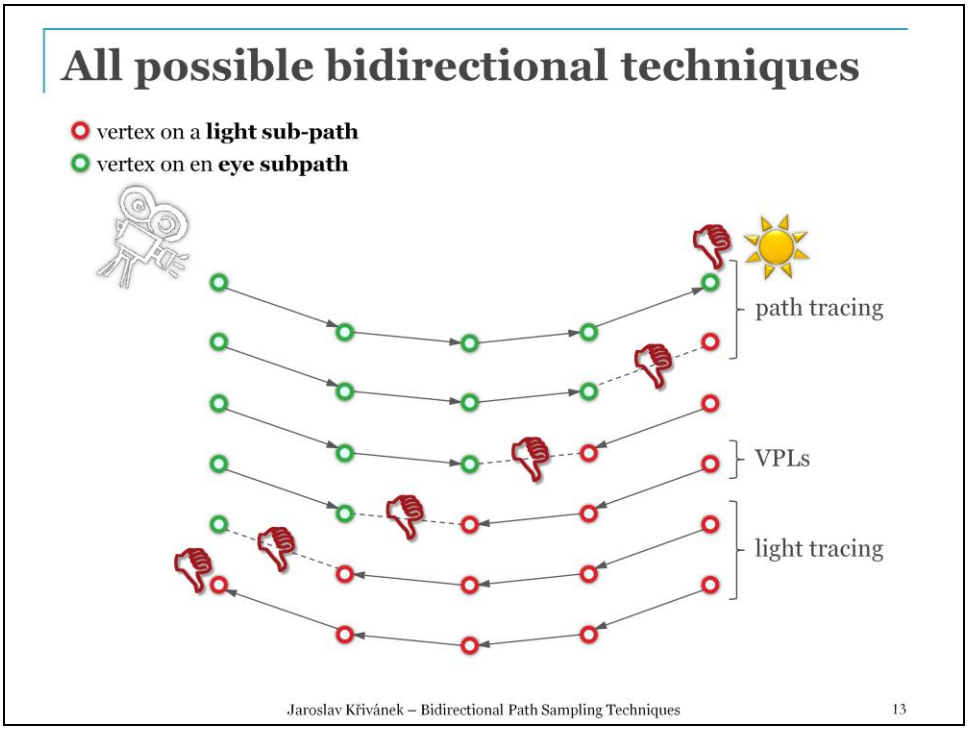
• To summarize, VPL rendering is easily interpreted in the path integral framework as a bidirectional path sampling technique.

• This view allows us to clearly identify that the splotches typical for VPL rendering are in fact just a demonstration of the variance caused by bad path sampling.

• Also notice that the splotches are in fact conceptually the exact same thing as noise in path tracing: both are just a visual manifestation of the variance of the underlying estimators. The reason we obtain splotches in VPL rendering is the inter-pixel correlation due to the VPL reuse across different pixels.



• Let’s now use the experience form the VPL rendering example to motivate the bidirectional path tracing algorithm.



• This slide schematically shows all the possible bidirectional techniques that we can obtain by starting a path either on light source or on the camera and applying only the basic three operations of local path sampling for an example path of length 4.

• The first two cases correspond to what a regular path tracer usually does (randomly hitting the light sources and explicit light source connections.)

• The fourth correspond to VPL sampling.

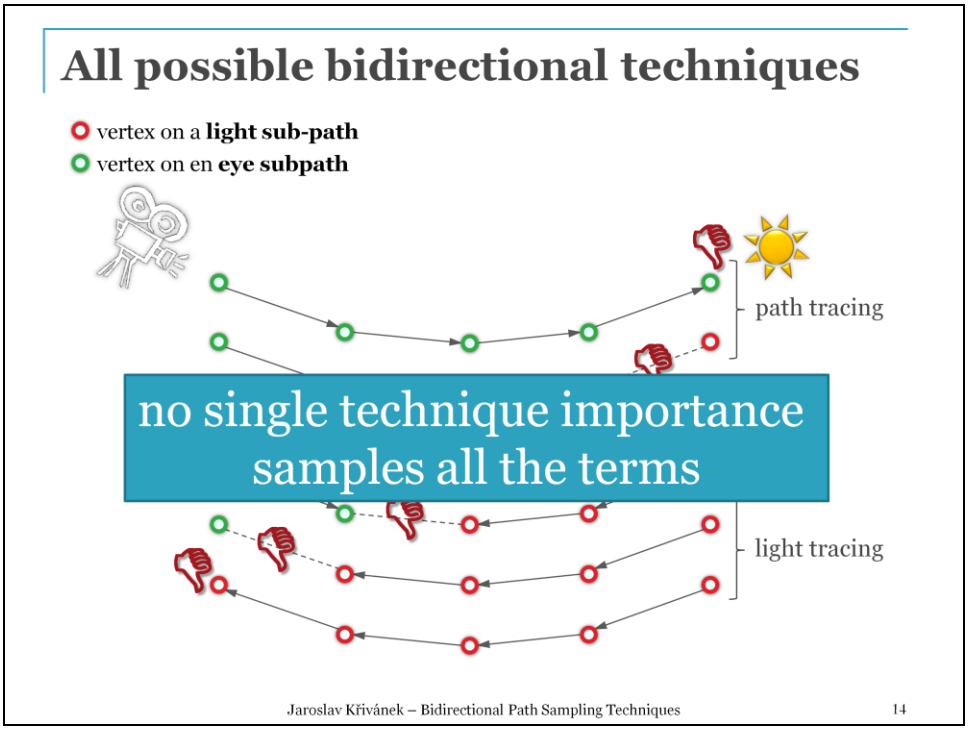
• And the last two are complementary to the first two and therefore correspond to light tracing.

• Each sampling technique importance samples a different subset of terms of the measurement contribution function.

• However, in each of these techniques, there are some terms of the measurement contribution function that are not importance sampled.

• The purely unidirectional techniques (top and bottom) do not importance sample the light emission and sensor sensitivity, respectively. Indeed, for example the technique at the top relies on randomly hitting a light source, without incorporating any information about the location of light sources in the scene.

• All the bidirectional techniques, that is, those that involve connection of two sub-paths, are unable to importance sample the terms associated with the connection edge, exactly as in the case of VPLs.



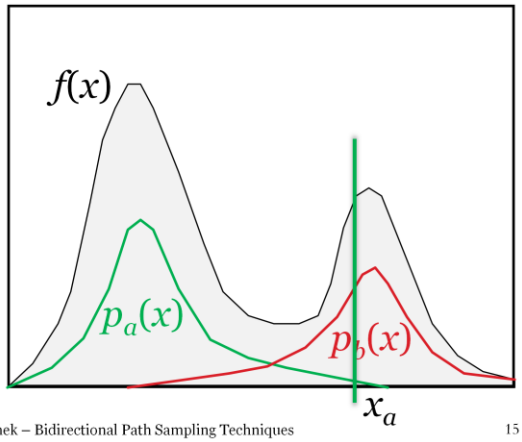
• But none of the techniques shown here is able to importance sample all of the terms of the measurement contribution function.

Multiple Importance Sampling (MIS)

[Veach & Guibas, 95]

Combined estimator:

$$\langle I \rangle = \frac{f(x)}{[p_a(x) + p_b(x)]/2}$$



Jaroslav Křivánek – Bidirectional Path Sampling Techniques 15

• This slide shows this situation in a simpler setting. We have a complex multimodal integrand $f(x)$ and we have no single PDF that is proportional to the integrand in the entire domain.

• But we do have two distributions with PDFs p_a and p_b each of which is a good match for the integrand under different conditions.

• We can use Multiple Importance Sampling (MIS) to combine the sampling techniques corresponding to the two PDFs into a single, robust, combined technique.

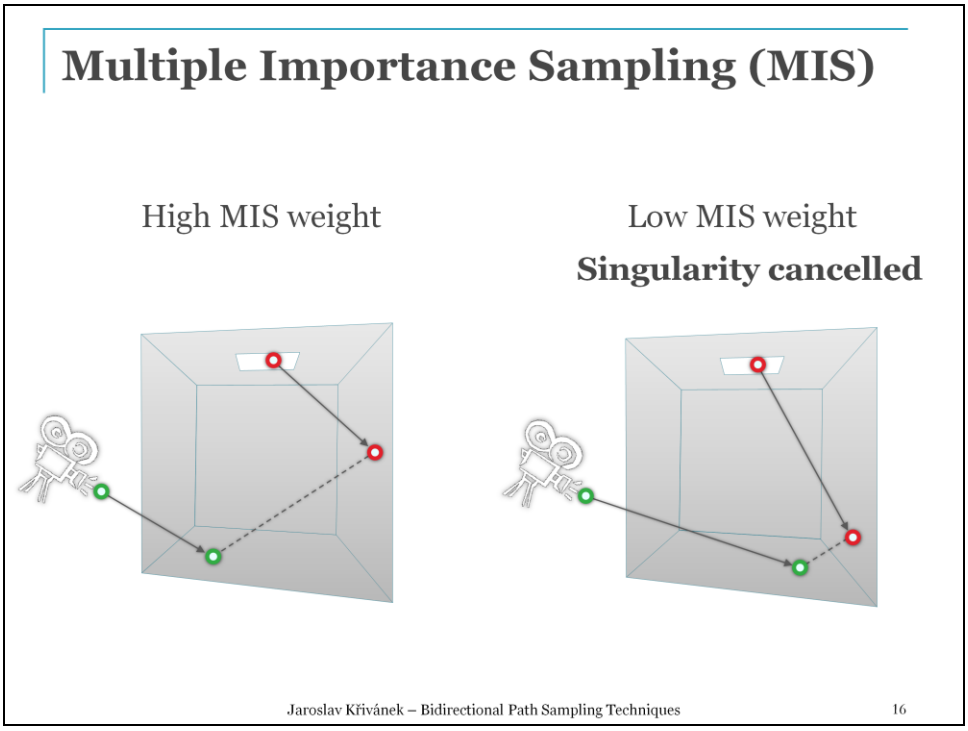
• MIS proceeds by first picking one distribution to sample from (p_a or p_b , say with fifty-fifty chance) and then taking the sample from the selected distribution.

• This essentially corresponds to sampling from a weighted average of the two distributions, which is reflected in the form of the estimator, shown on the slide.

• This estimator is really powerful at suppressing outlier samples such as those that you would obtain by picking x_from the tail of p_a, where $f(x)$ might still be large.

• Without having p_b at our disposal, we would be dividing the large $f(x)$ by the small $p_a(x)$, producing an outlier.

• However, the combined technique has a much higher chance of producing this particular x (because it can sample it also from p_b), so the combined estimator divides $f(x)$ by $[p_a(x) + p_b(x)] / 2$, which yields a much more reasonable sample value.

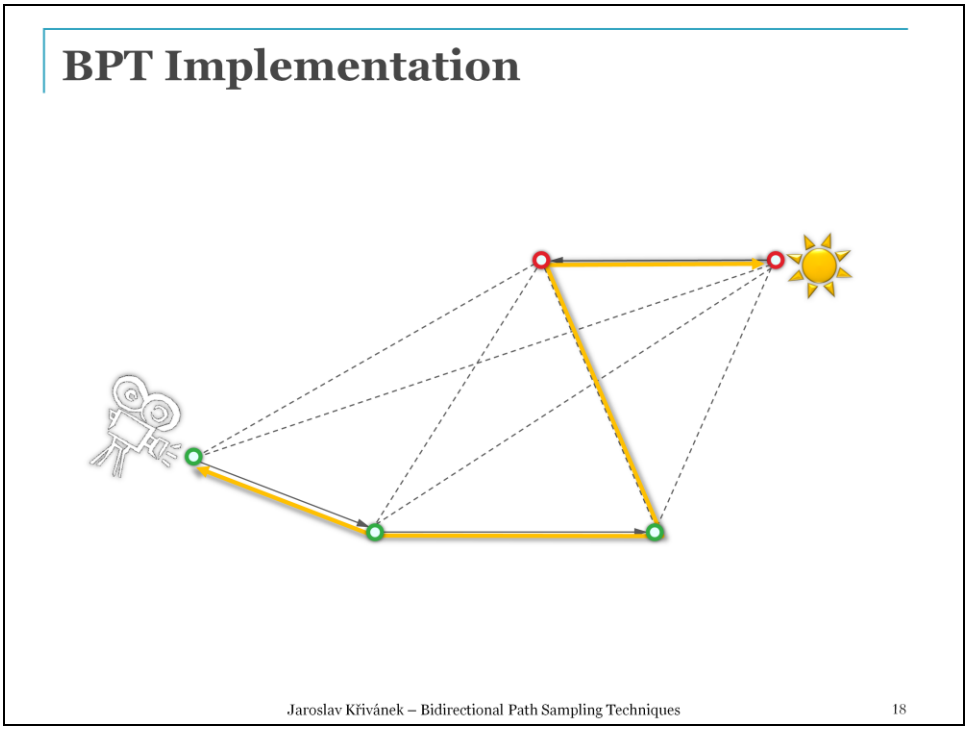


• Applying this to path sampling, this technique will weight down a lot the contribution of paths with short connecting edges, suppressing the high variance observed in the VPL methods.

Bidirectional path tracing

- Use **all** of the above sampling techniques
- Combine using **Multiple Importance Sampling**

• The main idea of bidirectional path tracing is to use all of the sampling techniques above and combine them using multiple importance sampling.



• The usual description of bidirectional path tracing, where two independent sub-paths are generated first, and the each vertex from the first is connected to each vertex of the second, is really just an implementation detail. It does improve the efficiency thanks to reuse of the sub-paths, but does not contribute to the robustness of BPT in any way.

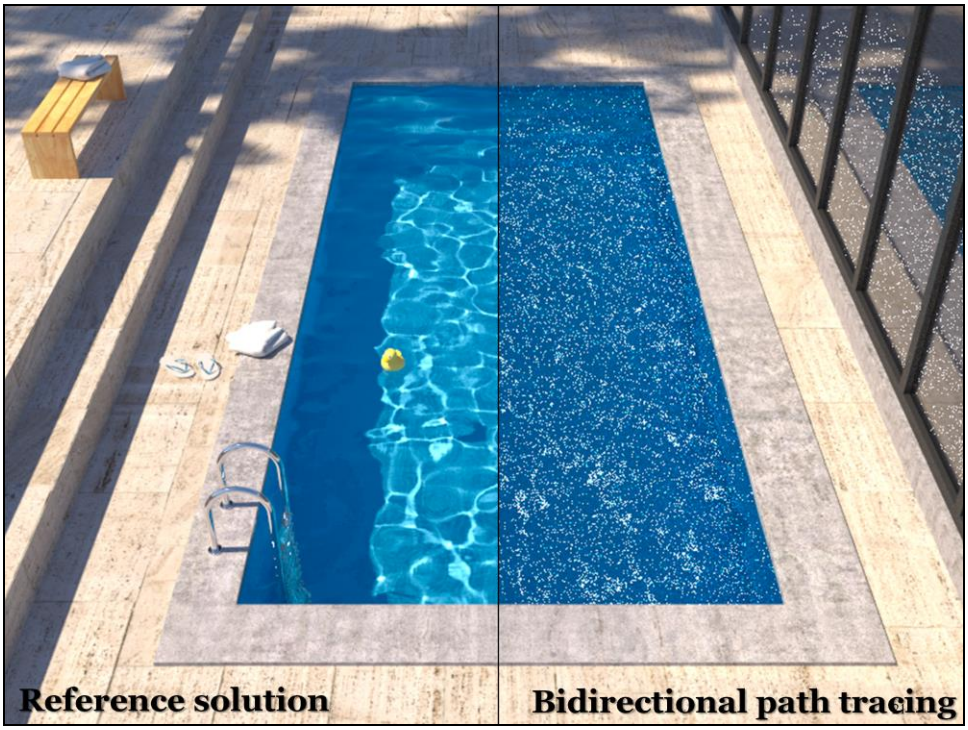
Results

BPT, 25 samples per pixel

PT, 56 samples per pixel

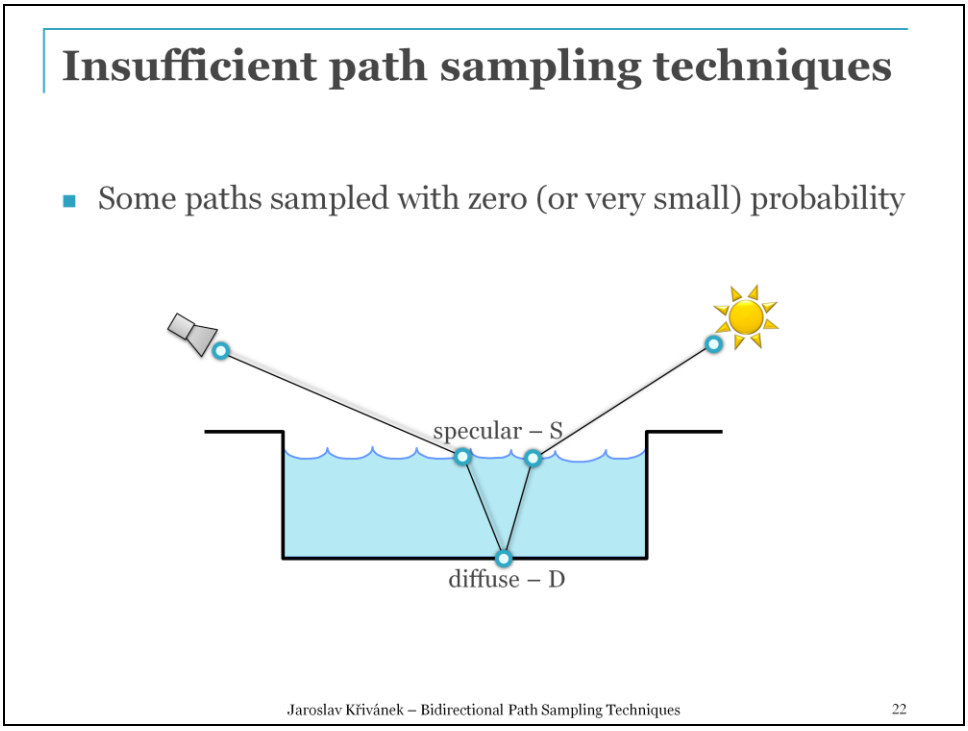
Images: Eric Veach

LIMITATIONS OF LOCAL PATH SAMPLING



• Bidirectional path tracing is much more robust than path tracing, light tracing, or VPL rendering.

• However, it still struggles with some lighting effects, the most common of which are the specular-diffuse-specular (SDS) paths corresponding to reflected caustics – in this example the caustics at the pool bottom.



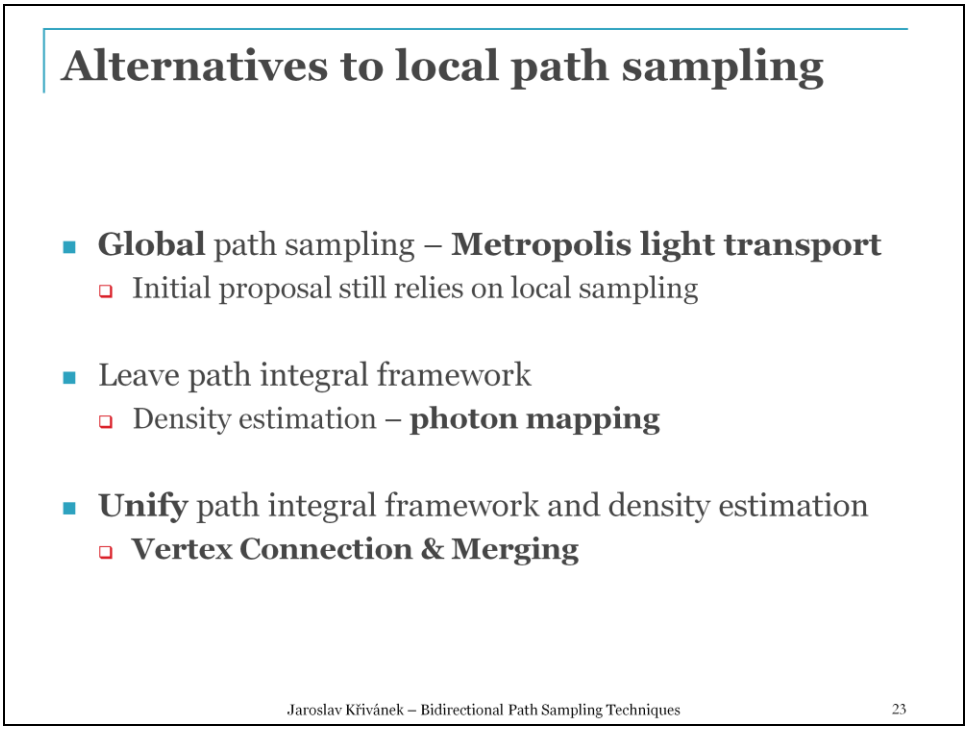
• As usual, the culprit is inappropriate path sampling. The problem is that none of the path sampling techniques used in bidirectional path tracing is efficient at sampling the SDS effects, so their combination cannot sample those effects either.

• To see this, consider the example on the slide. We have a pool (diffuse – D) filler with water (specular – S), a pinhole camera, and a small light source.

• No path connections are possible because of the two specular vertices. Unidirectional sampling from the light source is not possible either because of the pinhole camera.

• So we are left with one single (unidirectional) path sampling technique that starts from the camera, and hopes to randomly strike the light source. It is not hard to see that the smaller the source, the lower the probability of hitting the source and the higher the estimator variance.

• In the limit, for point sources and pinhole camera, the SDS effects cannot be sampled by local path sampling at all.



• To sample SDS effects efficiently, we need to look for alternatives to local path sampling.

• One such alternative are global path sampling techniques that sample a path as a whole, as opposed to incrementally vertex-by-vertex. This is for example the case of Metropolis light transport and other Markov Chain Monte Carlo techniques. The issue is that the Markov chain of path mutations needs to be initialized with some path, and this path has to be generated somehow – and we're back to local path sampling.

• We could also look for an entirely different solution of light transport, outside the classic path integral formulation. A popular example of this approach is photon mapping, which relies on density estimation.

• However, photon mapping is fairly inefficient at some lighting effects (such as diffuse inter-reflections) which are very efficiently handled by the path sampling techniques in BPT.

• So the vertex connection and merging algorithm reformulates photon mapping in the path integral framework, which enables a robust combination of photon mapping and BPT using Multiple Importance Sampling.

NEARLY THERE...

"Path integral" – A historical remark

- This course [Veach and Guibas 1995], [Veach 1997]
  - Easily derived form the rendering equation [Veach 1997]
- Feynman path integral formulation of quantum mechanics [Feynman and Hibbs 65]
- Homogeneous materials [Tessendorf 89, 91, 92]
- Rendering [Premože et al. 03, 04]

• The term "path integral formulation" has a different meaning for different people.

• What I've presented in this course has been derived by Veach and Guibas by fairly straightforward manipulation of the Neumann series solution of the rendering equation.

• More widely known is Feynman's path integral formulation, used to solve problems in quantum mechanics. In this formulation the transport paths are general curves, so the path space that we considered here (polylines) is just a small sub-space (of measure zero) of the path space in Feynman's formulation.

•Tessendorf used Feynman's path integral formulation to derive the solution of light transport in strongly forward scattering media.

•This solution has been used for rendering by Premože and colleagues.

Summary

- **VPL rendering**
  - One bidirectional path sampling technique
  - Not robust
- **Bidirectional path tracing**
  - Combines many bidirectional techniques
  - More robust
  - Bad at reflected caustics

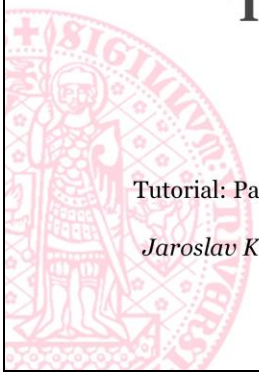
Acknowledgements

- **Czech Science Foundation**
  - grant no. P202-13-26189S

- Images
  - **Ondra "Keymaster" Karlík**
  - **Eric Tabellion**
  - **Marcos Fajardo**

THANK YOU!

Time for questions…

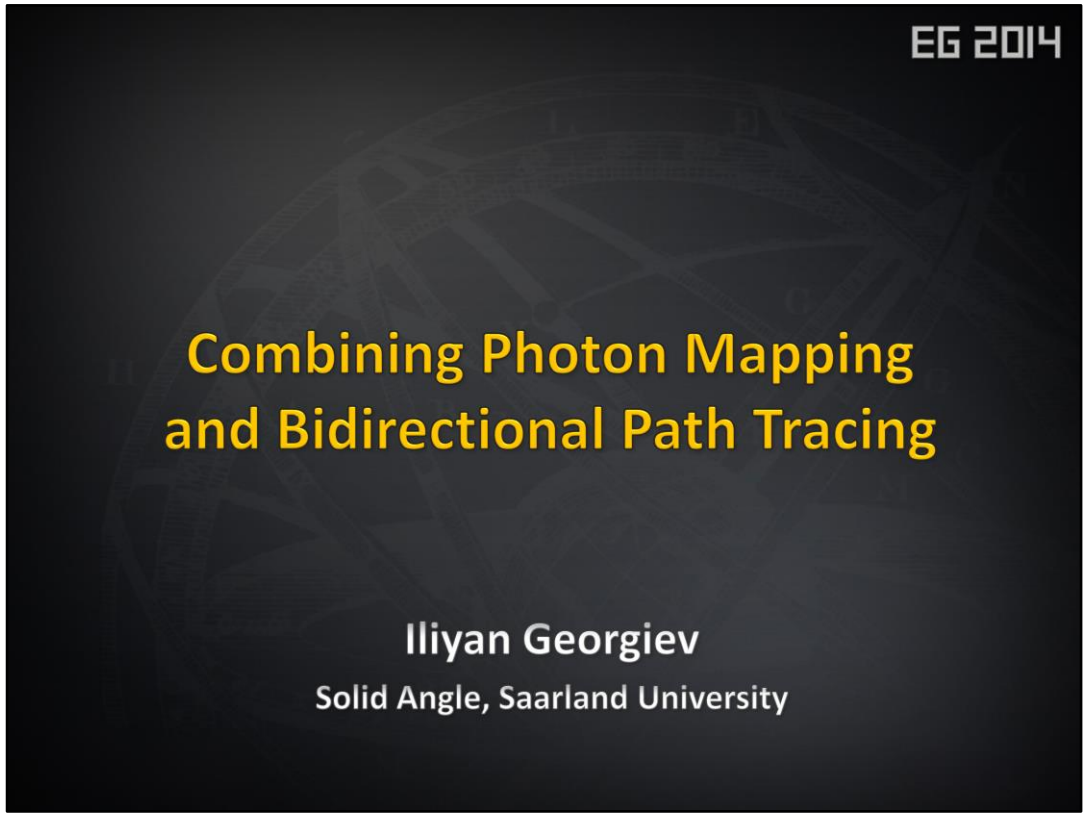


Tutorial: Path Integral Methods for Light Transport Simulation

Jaroslav Křivánek – Bidirectional Path Sampling Techniques

Iliyan Georgiev:

Combining Photon Mapping and Bidirectional Path Tracing

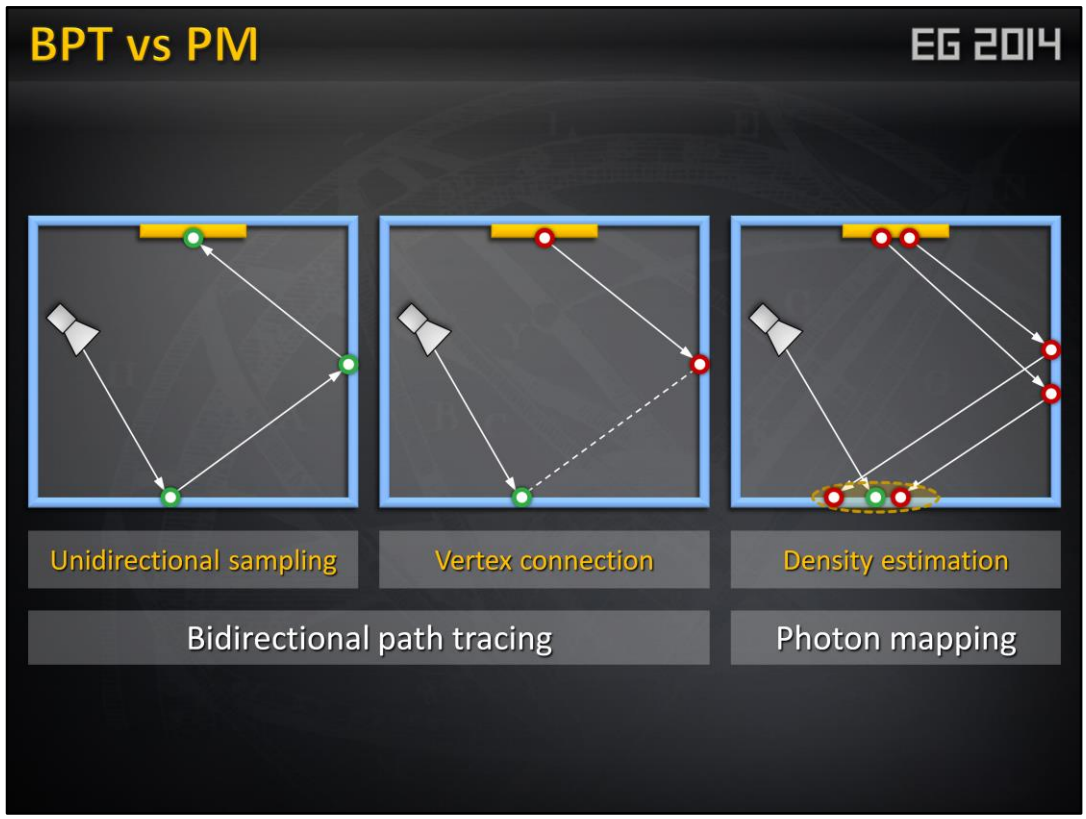


In the previous talk, Jaroslav discussed the path integral formulation of light transport and demonstrated its conceptual simplicity and flexibility. I will now show how we can leverage this framework to seamlessly combine bidirectional path tracing and photon mapping via multiple importance sampling.

Bidirectional path tracing is one of the most versatile light transport simulation algorithms available today. It can robustly handle a wide range of illumination and scene configurations, but is notoriously inefficient for specular-diffuse-specular light interactions, which occur e.g. when a caustic is seen through a reflection/refraction.

On the other hand, photon mapping (PM) is well known for its efficient handling of caustics. Recently, Hachisuka and Jensen [2009] showed a progressive variant of PM that converges to the correct result with a fixed memory footprint. Their stochastic progressive photon mapping (PPM) algorithm captures the reflected caustics in our scene quite well. However, it has hard time handling the strong distant indirect illumination coming from the part of the scene behind the camera.

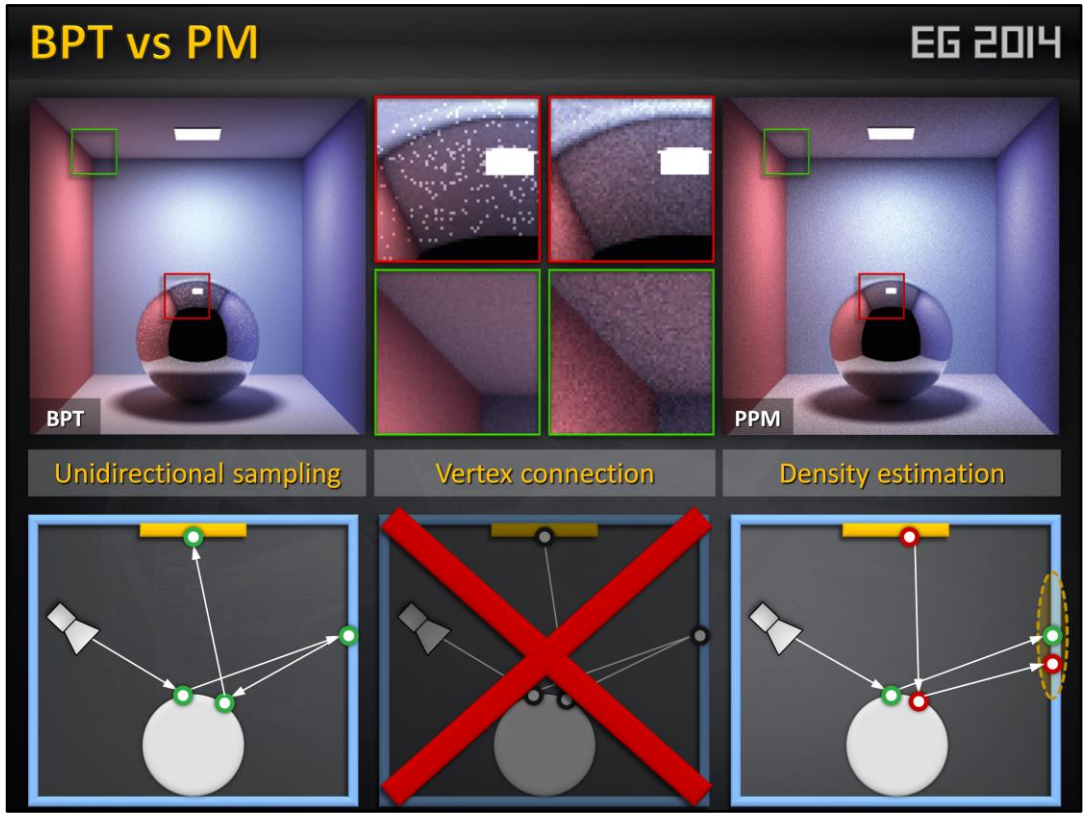
By using multiple importance sampling to combine estimators from bidirectional path tracing and photon mapping, the algorithm I will talk about today automatically finds a good mixture of techniques for each individual light transport path, and produces a clean image in the same amount of time.



Let us start by reviewing how bidirectional path tracing (BPT) and photon mapping (PM) sample light transport paths that connect the light sources to the camera:

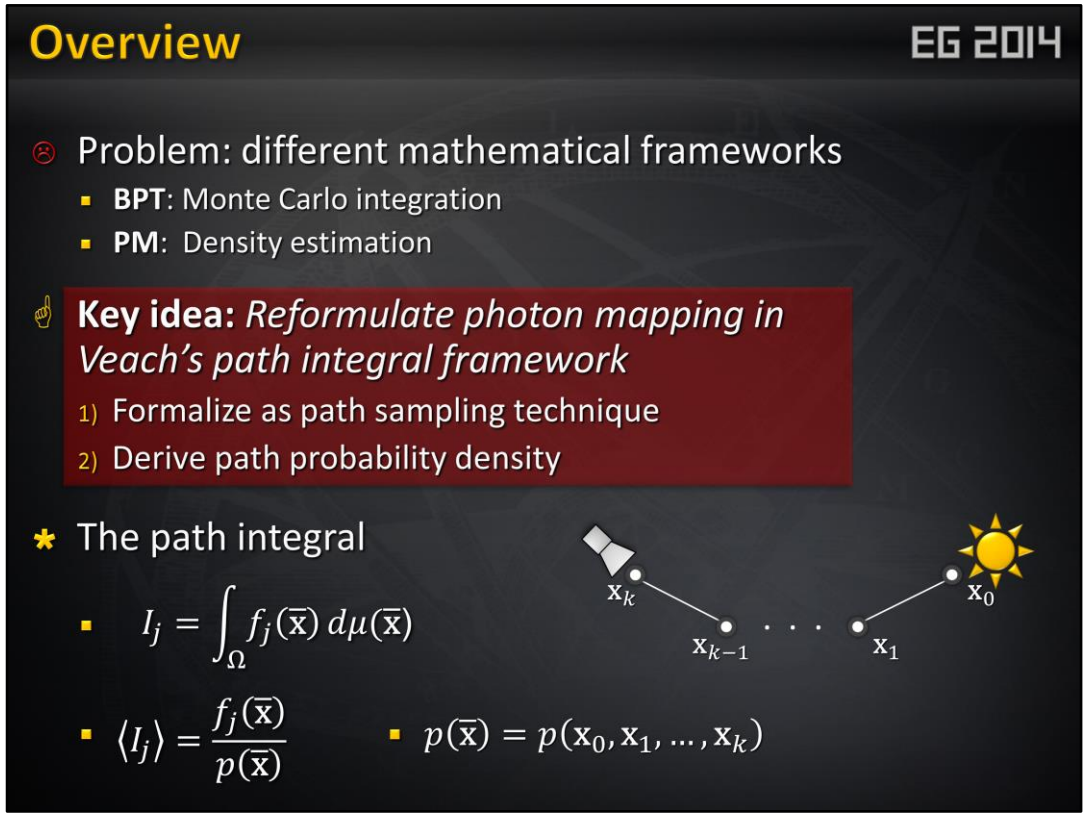
The techniques BPT employs can be roughly categorized to *unidirectional sampling* (US) and *vertex connection* (VC). US constructs a path by starting either from a light source or the camera and tracing a random walk in the scene until termination. On the other hand, VC traces one subpath from a light source and another one from the camera, and then completes a full path by connects their endpoints.

In contrast, PM first traces a number of light subpaths and stores their vertices, a.k.a. photons. It then traces subpaths from the camera and computes the outgoing radiance at the hit points using density estimation by looking up nearby photons.



BPT can efficiently capture directly visible caustics, as it can connect light subpath vertices to the camera. However, for sampling specular-diffuse-specular paths, BPT can only rely on unidirectional sampling, as VC cannot perform connections with specular vertices. Since the probability of randomly hitting the light source is often very low, the resulting images suffer from excessive noise.

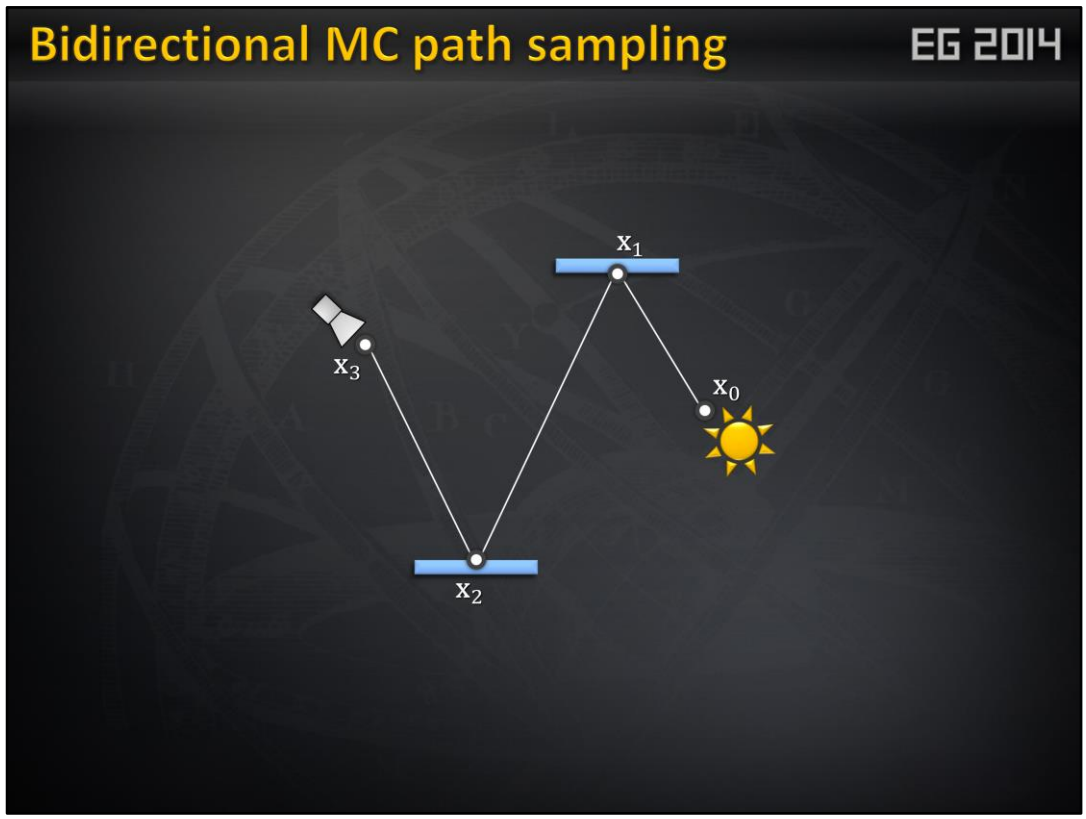
On the other hand, photon mapping handles both direct and reflected caustics pretty much the same way – by loosely connecting nearby eye and light sub-path vertices. But as can be seen in the images above, it is less efficient than BPT for diffuse illumination.



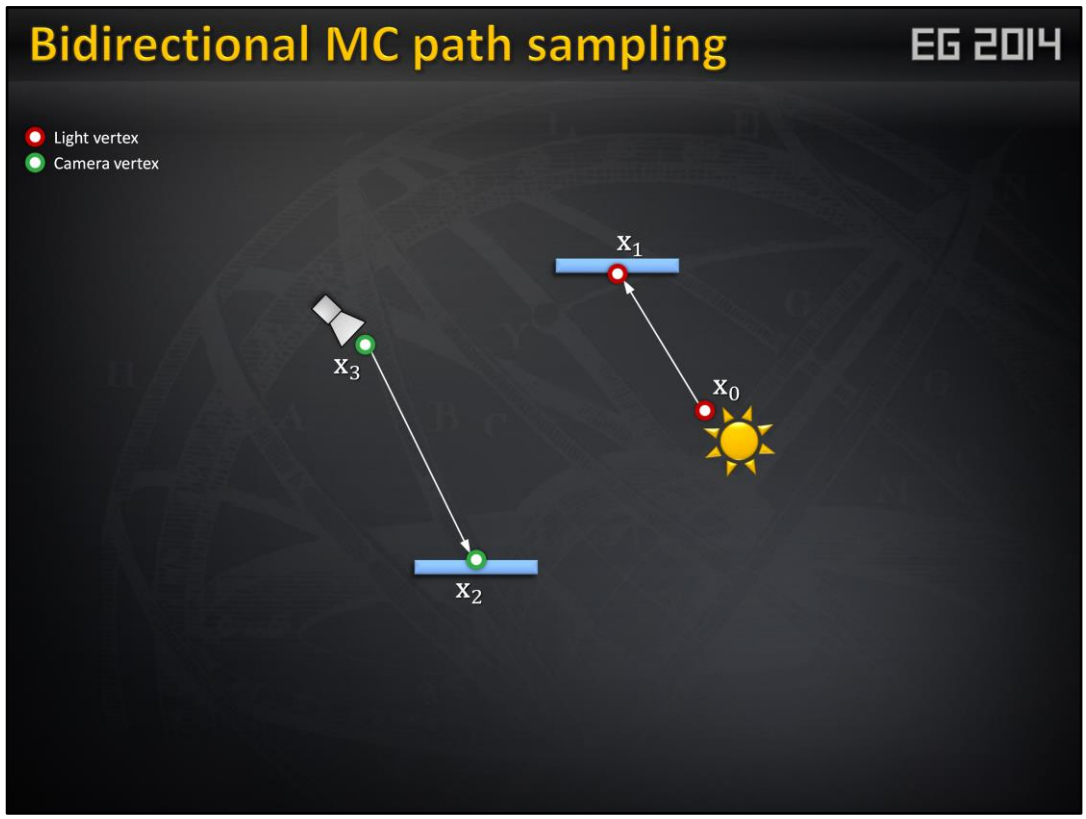
It has been long recognized that bidirectional path tracing (BPT) and photon mapping (PM) complement each other in terms of the light transport effects they can efficiently handle. However, even though both methods have been published more than 15 years ago, neither a rigorous analysis of their relative performance nor an efficient combination had been shown until very recently. The reason for this is that BPT and PM have originally been defined in different theoretical frameworks – BPT as a standard Monte Carlo estimator to the path integral, and PM as an outgoing radiance estimator based on photon density estimation.

The first step toward combining these two methods is to put them in the same mathematical framework. We choose Veach's path integral formulation of light transport, as it has a number of good properties (which Jaroslav discussed) and also because BPT is already naturally defined in this framework.

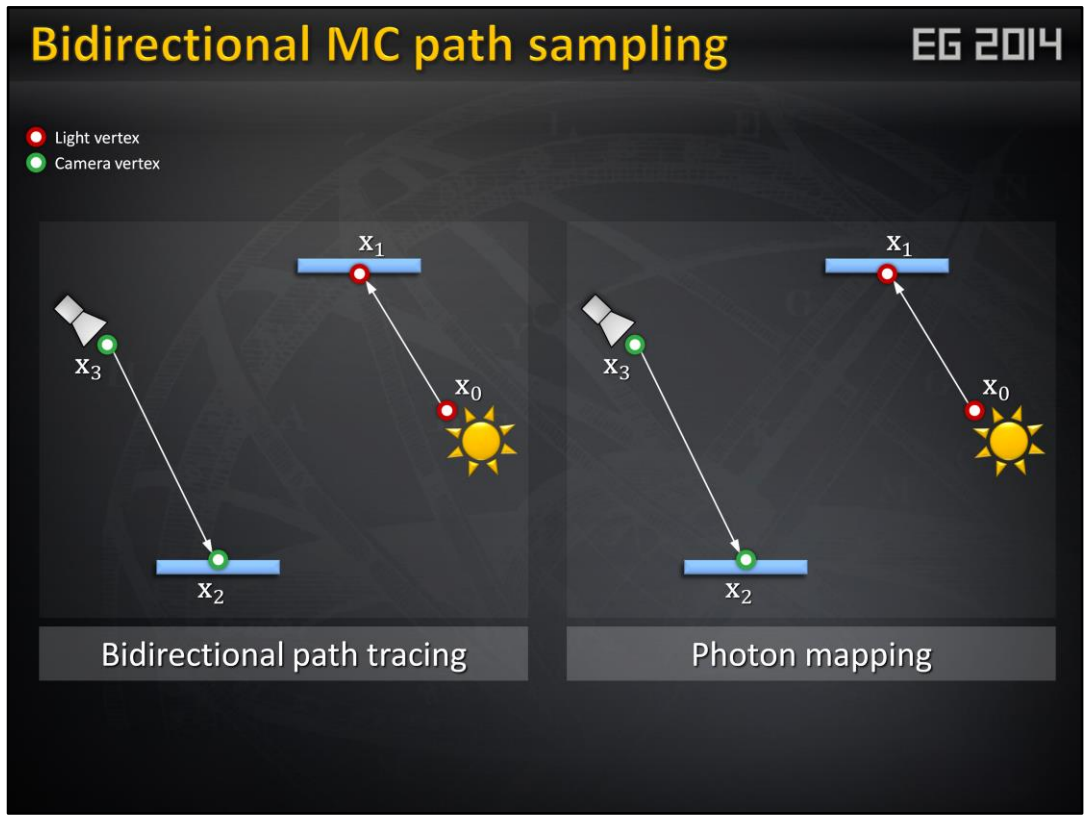
We need two key ingredients: (1) express PM as a sampling technique that constructs light transport paths that connect the light sources to the camera, and (2) derive the probability densities for paths sampled with this technique. This will give us a basis for reasoning about the relative efficiency of BPT and PM. And more importantly, it will lay the ground for combining their corresponding estimators via multiple importance sampling.



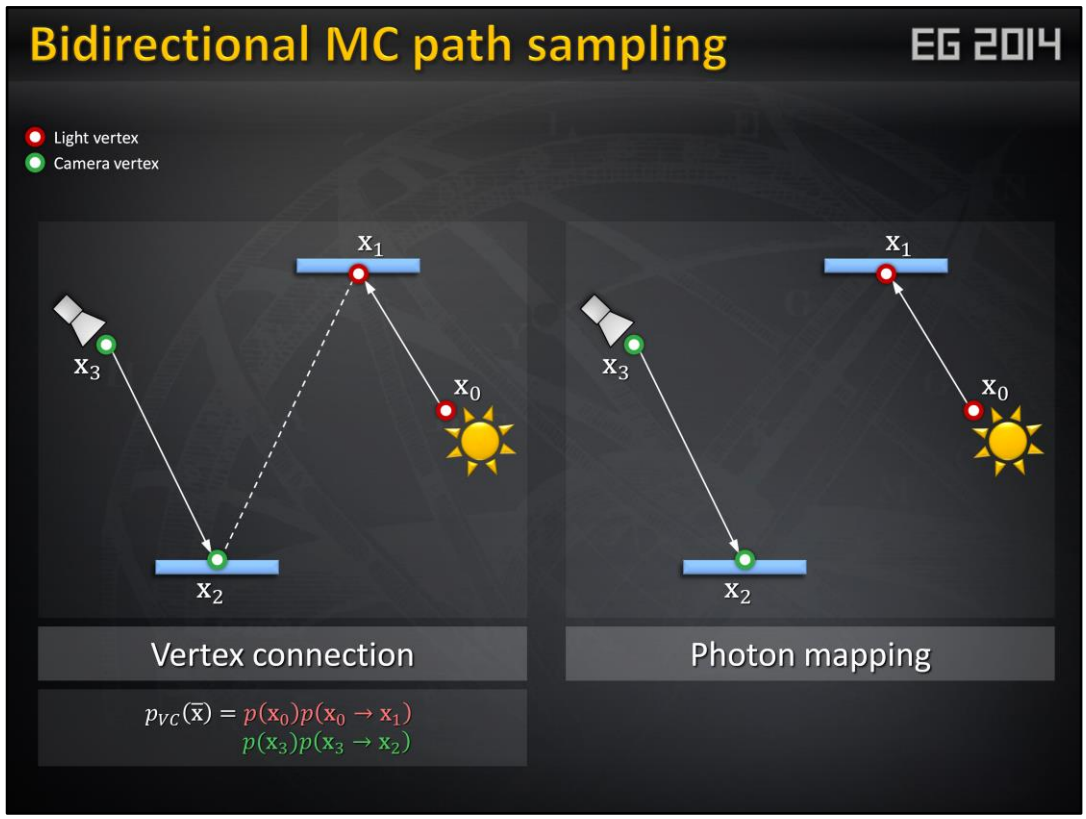
Let us start by taking a simple length-3 path and see how it can be constructed bidirectionally.



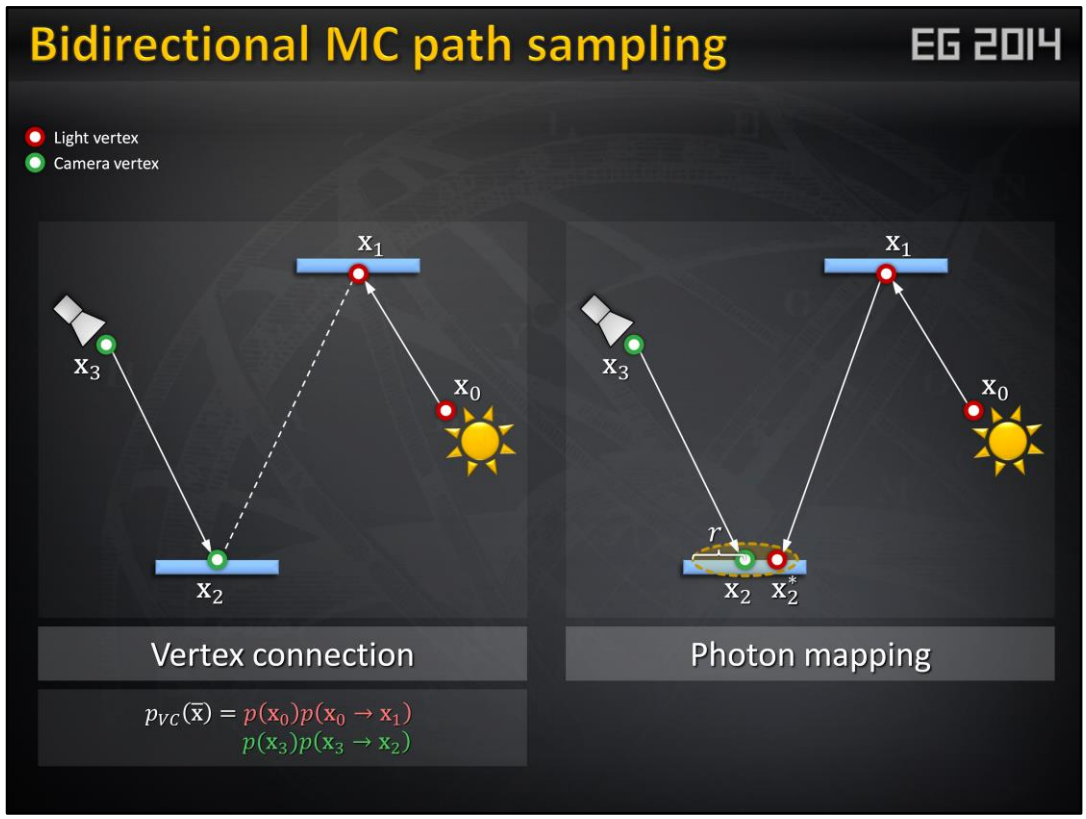
We first trace one subpath from the camera and another one from a light source.



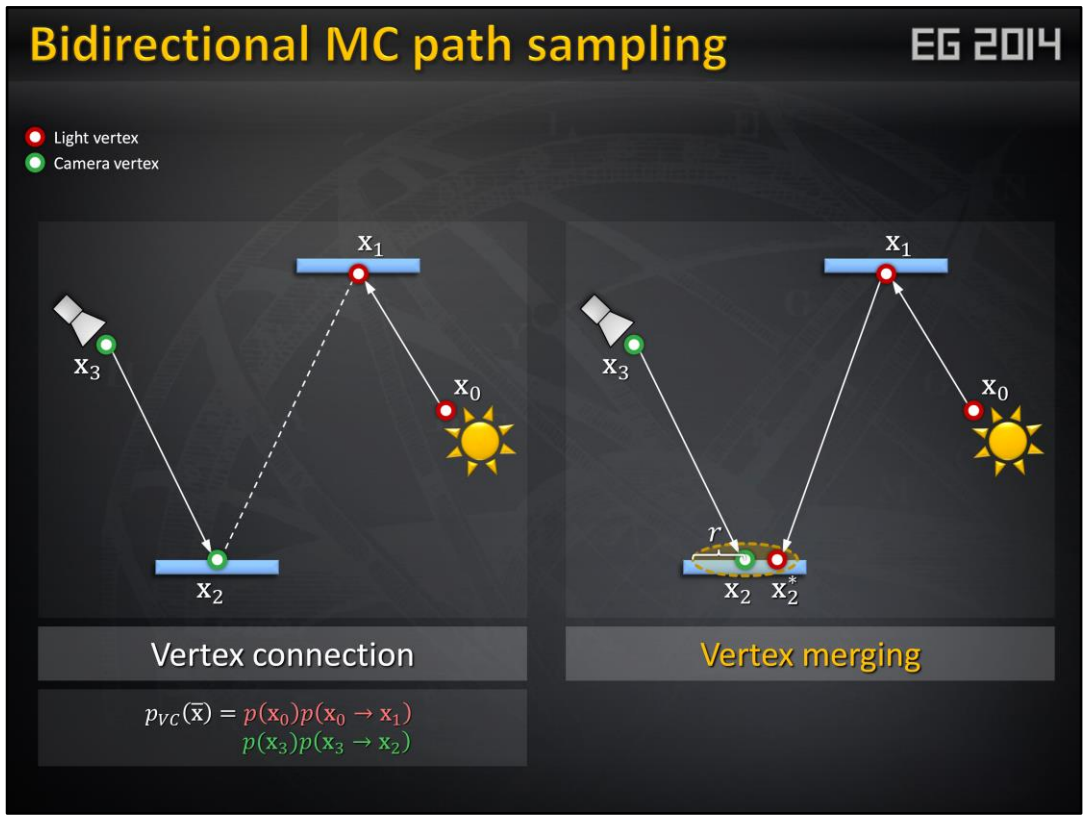
Now let's see how we complete a full path in BPT and PM.



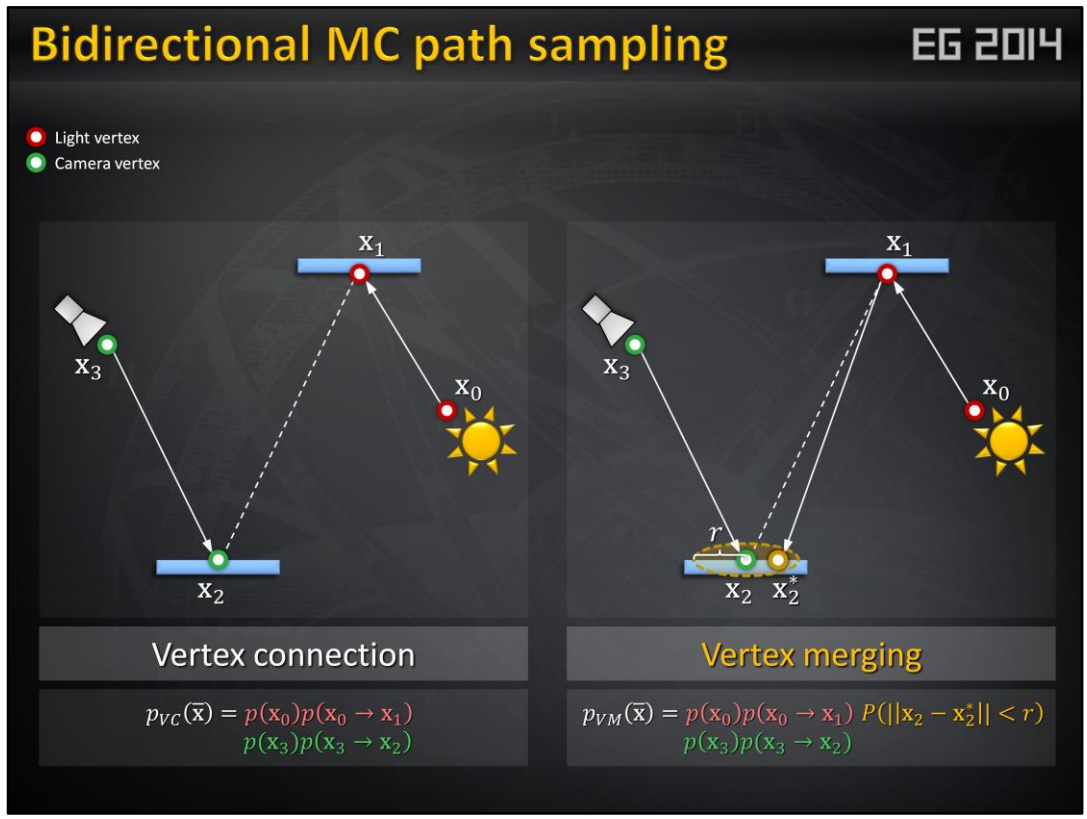
Bidirectional path tracing connects the subpath endpoints deterministically. We call this technique *vertex connection*. The PDF of the resulting full path is well known, and is simply the product of the PDFs of two subpaths, which have been sampled independently.



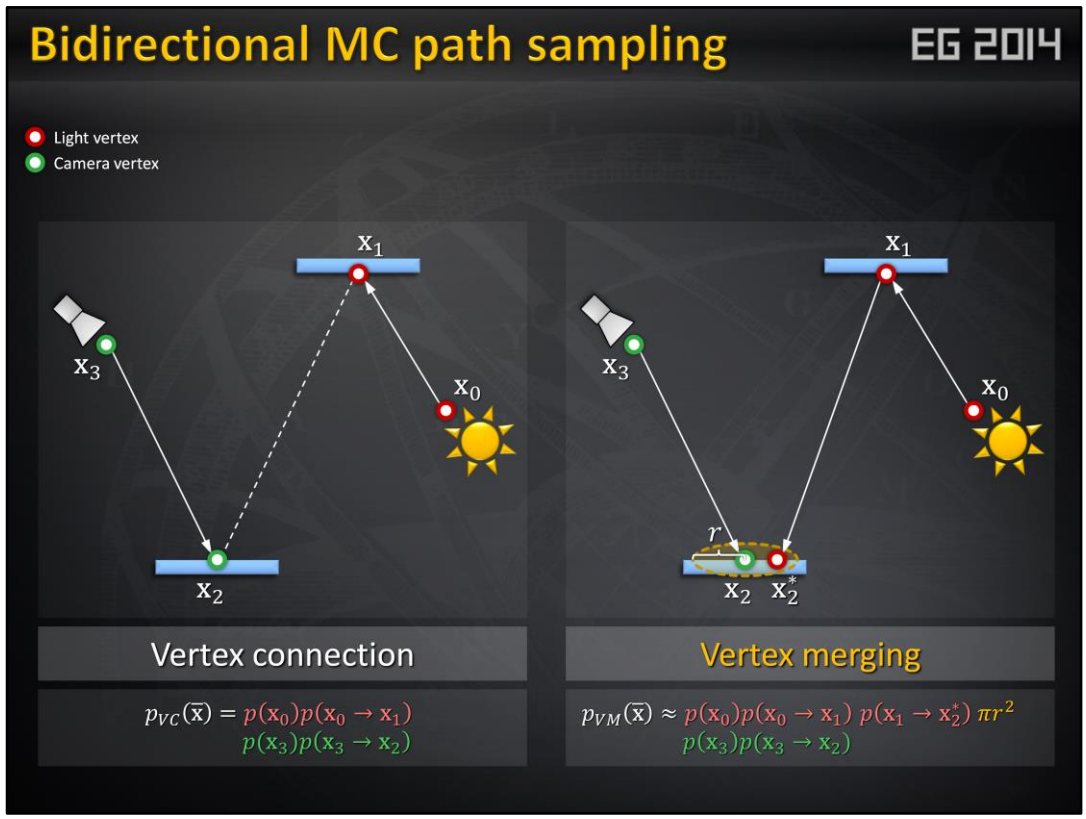
Photon mapping, on the other hand, will extend the light subpath by sampling one more vertex from $\mathbf{x}_1$, and will concatenate the two subpaths only if the "photon" hit-point $\mathbf{x}_2^*$ lies within a distance r from $\mathbf{x}_2$.



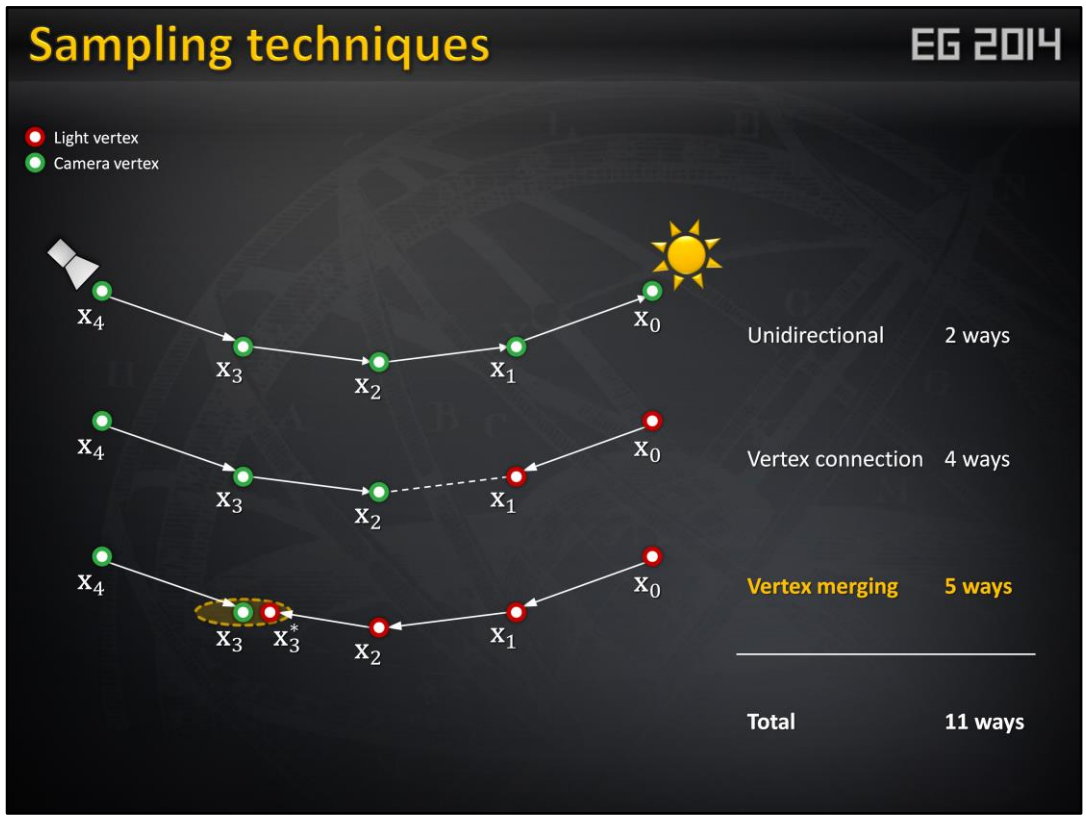
We label this technique *vertex merging*, as it can be intuitively thought to weld the endpoints of the two subpaths if they lie close to each other.



What remains is to derive the PDF of the resulting full path. To do this, we can interpret the last step as establishing a regular vertex connection between $\mathbf{x}_1$ and $\mathbf{x}_2$, but conditioning its acceptance on the random event that a vertex $\mathbf{x}_2^*$ sampled from $\mathbf{x}_1$ lands within a distance r to $\mathbf{x}_2$. This probabilistic acceptance is nothing more than a Russian roulette decision. The full path PDF is then again the product of the subpath PDFs, but in addition multiplied by the probability of sampling the point $\mathbf{x}_2^*$ within a distance r of $\mathbf{x}_2$. This acceptance probability is equal to the integral of the PDF of $\mathbf{x}_2^*$ over the r-neighborhood of $\mathbf{x}_1$.

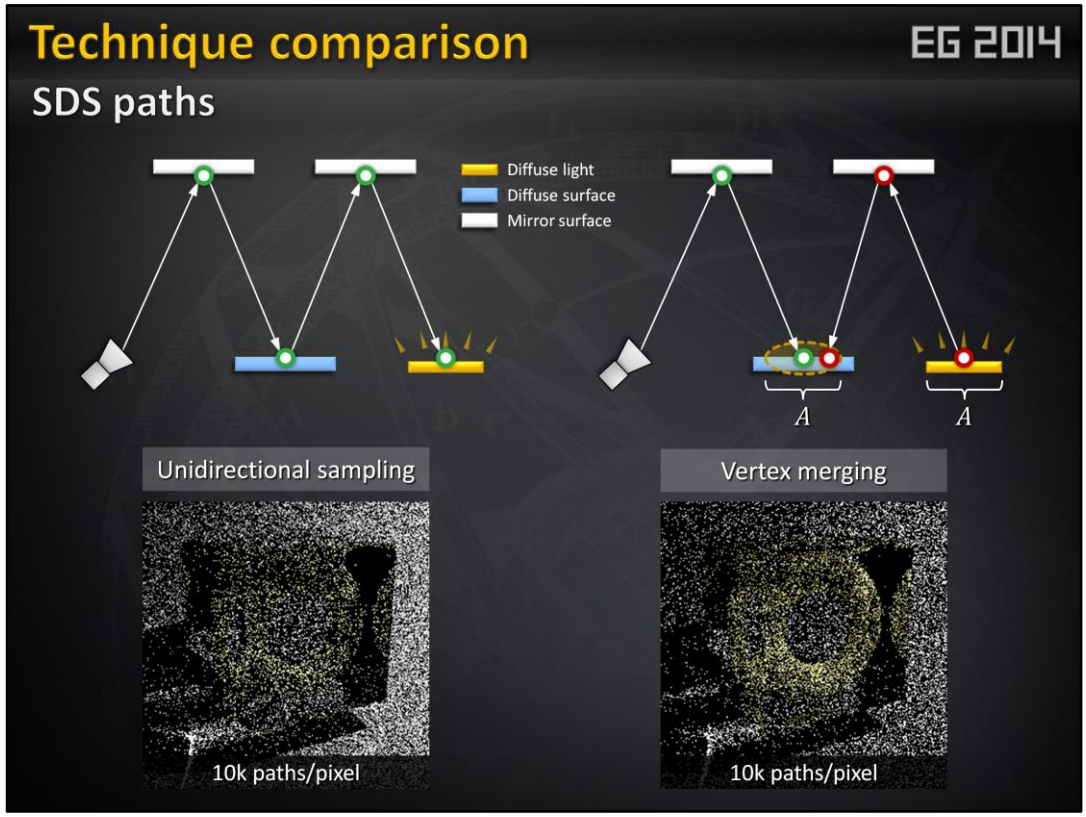


Under the reasonable assumptions that the surface around $\mathbf{x}_1$ is locally flat, i.e. that this neighborhood is a disk, and that the density of $\mathbf{x}_2^*$ is constant inside this disc, the integral can be well approximated by the PDF of the actual point $\mathbf{x}_2^*$ we have sampled, multiplied by the disc area πr^2.



Now that we have formulated the vertex merging path sampling technique, we can put it side by side with the already available techniques in BPT. There are two ways to sample a length-4 path unidirectionally, and four ways to sample it via vertex connection. Vertex merging adds five new ways to sample the path, corresponding to merging at the five individual path vertices. In practice, we can avoid merging at the light source and the camera, as directly evaluating emission and sensitivity is usually cheap.

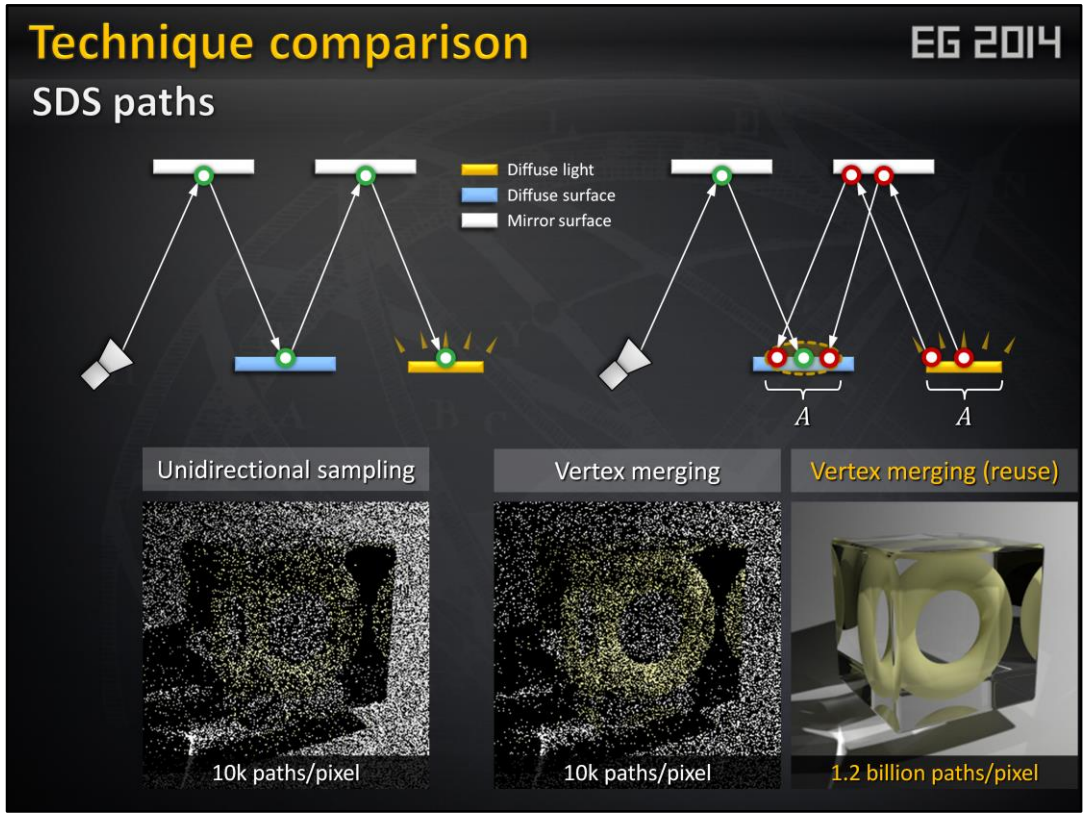
But with so many ways to sample the same light transport path, a question naturally arises in the mind of the curious: which technique is the most efficient for what types of paths?



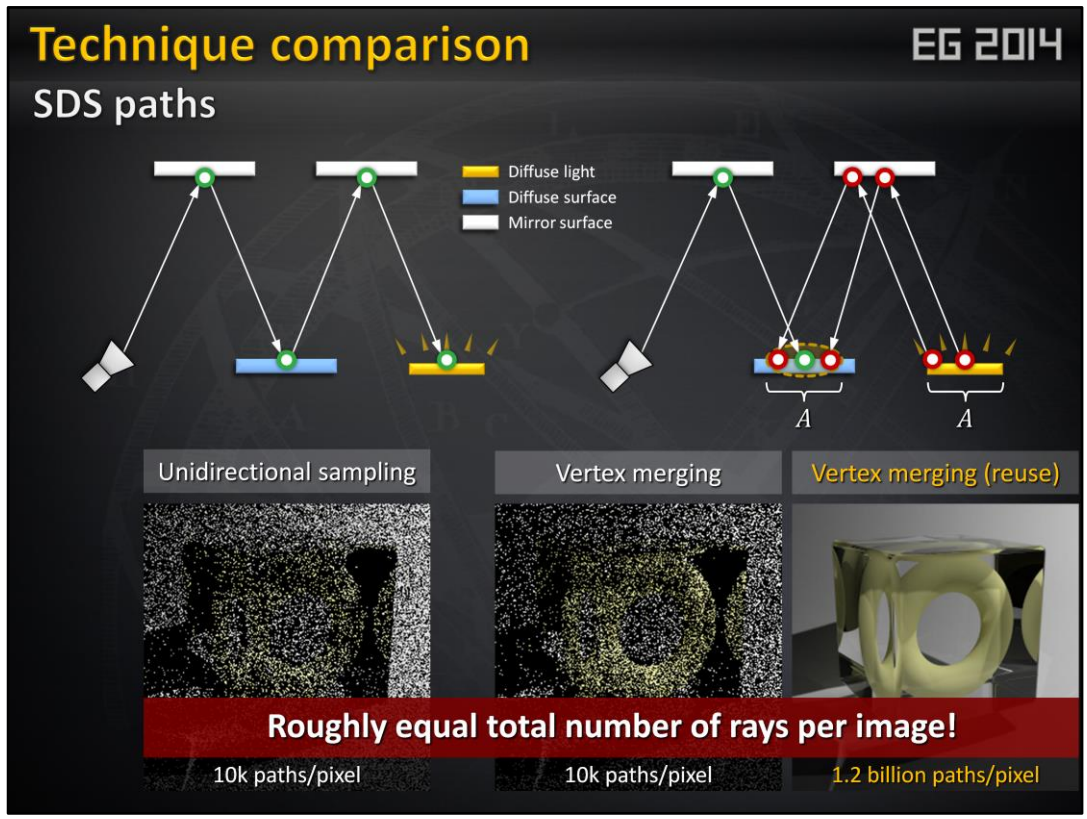
To answer this question, let us first take a look at specular-diffuse-specular (SDS) paths. Here, bidirectional path tracing can only rely on unidirectional sampling: it traces a path from the camera hoping to randomly hit the light source. With vertex merging, we can trace one light and one camera subpath, and merge their endpoints on the diffuse surface.

It can be shown that if the light source and the merging disk have the same area A, then unidirectional sampling and vertex merging sample paths with roughly the same probability density. This means that we should expect the two techniques to perform similarly in terms of rendering quality.

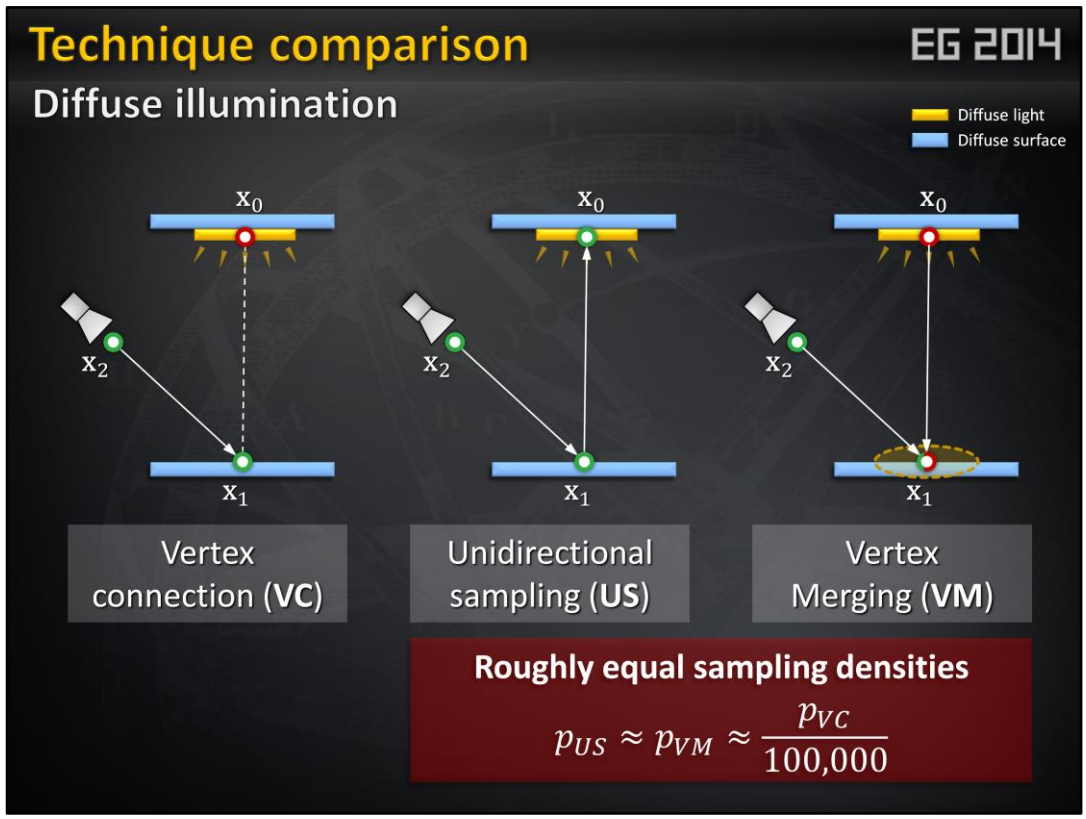
We render these two images progressively, sampling one full path per pixel per iteration. For the left image we trace paths from the camera until they hit the light. For image on the right, we trace subpaths from both ends, and merge their endpoints if they lie within a distance $r = \sqrt{A/\pi}$ from each other. Both images look equally noisy, even after sampling 10,000 paths per pixel. This confirms that vertex merging, and thus photon mapping, is *not* an intrinsically more robust sampling technique for SDS paths than unidirectional sampling.



However, the strength of vertex merging is computational efficiency – we can very efficiently reuse the light subpaths traced for *all* pixels at the cost of a single range search query. This allows us to quickly construct orders of magnitude more light transport estimators from the same sampling data, with a minimal computational overhead, resulting in a substantial quality improvement.



For all these three images we have traced roughly the same number of rays, and the only difference between the one in the center and the one on the right is that the for right image we have enabled path reuse, by storing, and looking up, the light subpath vertices in a photon map at every rendering iteration.

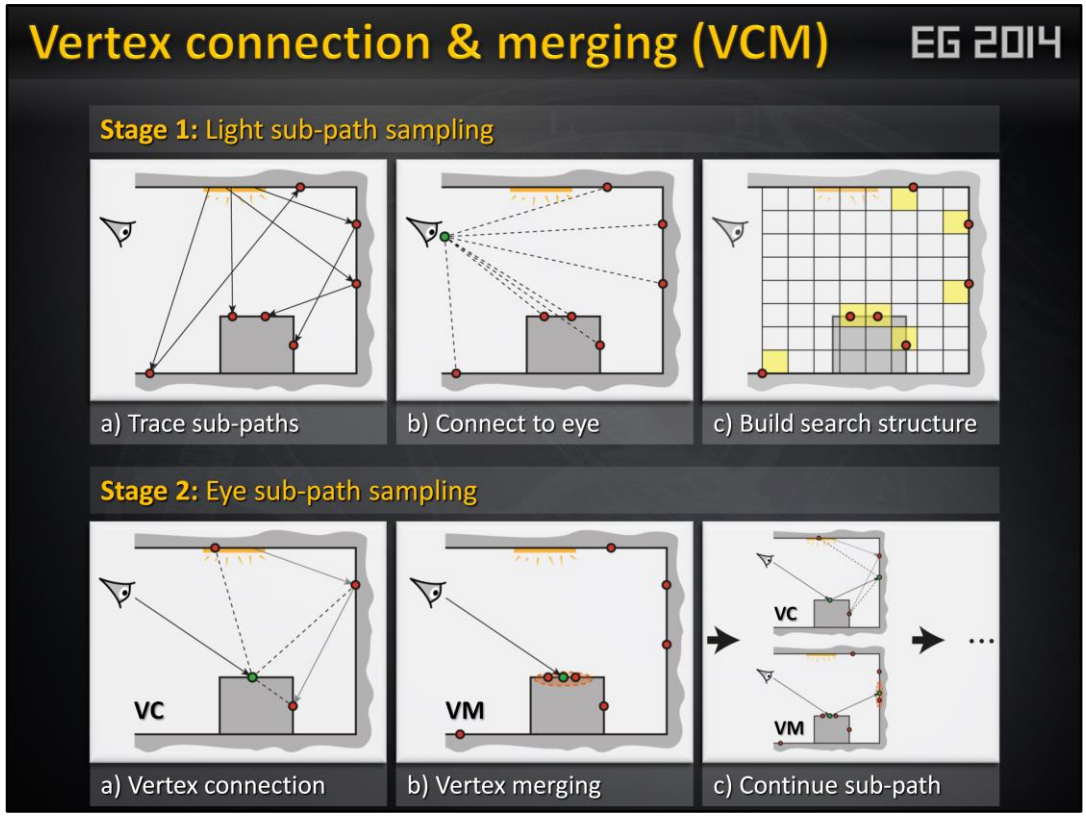


Now let's look at another extreme example – diffuse illumination. Note that vertex connection (VC) constructs the edge between $\mathbf{x}_1$ and $\mathbf{x}_2$ deterministically, while unidirectional sampling (US) and vertex merging (VM) both rely on random sampling.

Once again, it can be shown that if the light source and the merging disk have the same area, then US and VM sample this path with roughly the same probability density.

For the specific case shown on this slide, this density is about 100,000 lower than that of VC. This demonstrates that VM is not an intrinsically more robust sampling technique than VC either. This is not surprising – if we recall the expression for the VM path PDF, we see that it can only be lower than that of the corresponding VC technique, as their only difference is the probability factor in the VM PDF, which is necessarily in the range $[0; 1]$. Still, by reusing paths across pixels, vertex merging, and thus photon mapping, gains a lot of efficiency over unidirectional sampling.

All these useful insights emerge from the reformulation of photon mapping as a path sampling technique.



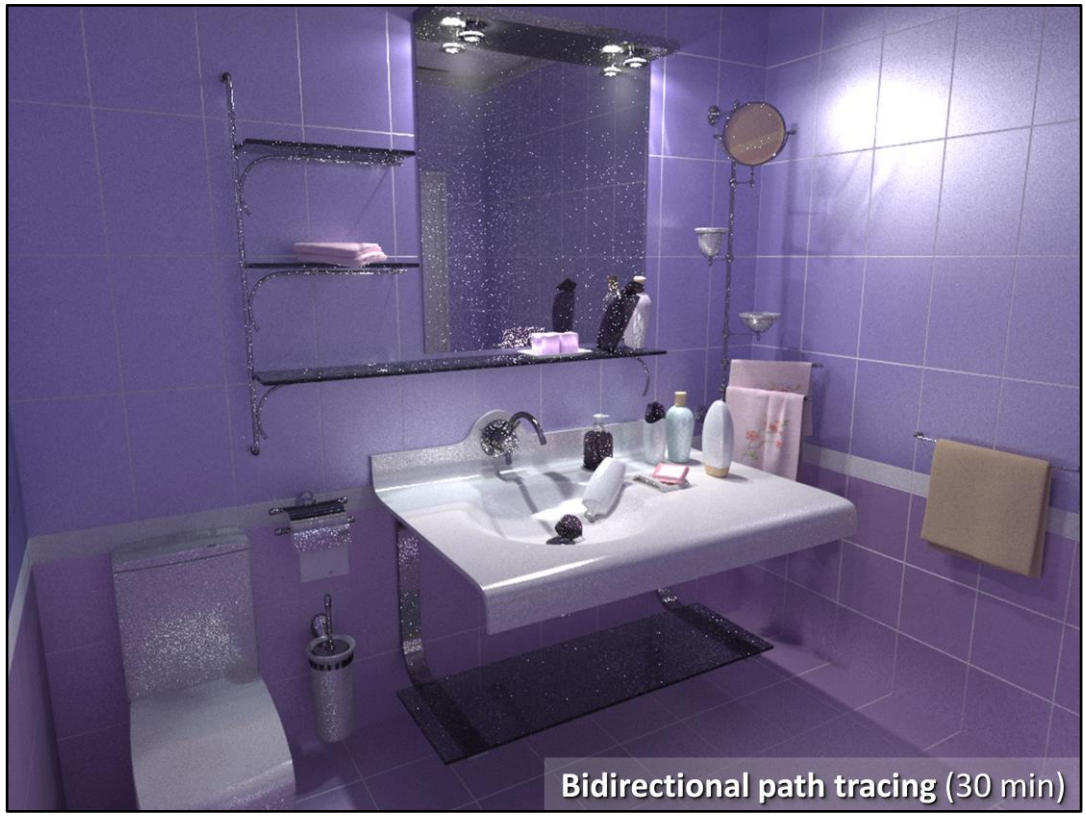
Even more usefully, we now have the necessary ingredients for combining photon mapping and bidirectional path tracing into one unified algorithm. The vertex merging path PDFs tell us how to weight all sampling techniques in multiple importance sampling, and the insights from the previous two slides command to strive for path reuse.

The combined algorithm, which we call *vertex connection and merging* (VCM), operates in two stages.

1. In the first stage, we
    a) trace the light subpaths for all pixels,
    b) connect them to the camera, and
    c) store them in a range search acceleration data structure (e.g. a kd-tree or a hashed grid).

2. In the second stage, we trace a camera subpath for every pixel.
    a) Each sampled vertex on this path is connected to a light source (a.k.a. next event estimation), connected to the vertices of the light subpath corresponding to that pixel, and
    b) merged with the vertices of *all* light subpaths.
    c) We then sample the next vertex and do the same.

In a progressive rendering setup, we perform these steps at each rendering iteration, progressively reducing the vertex merging radius . For details on this, please refer to the cited papers below for details.



Let us now see how this combined algorithm stacks up against bidirectional path tracing and stochastic progressive photon mapping on a number of scenes with complex illumination.


Stochastic progressive photon mapping (30 min)


Vertex connection and merging (30 min)

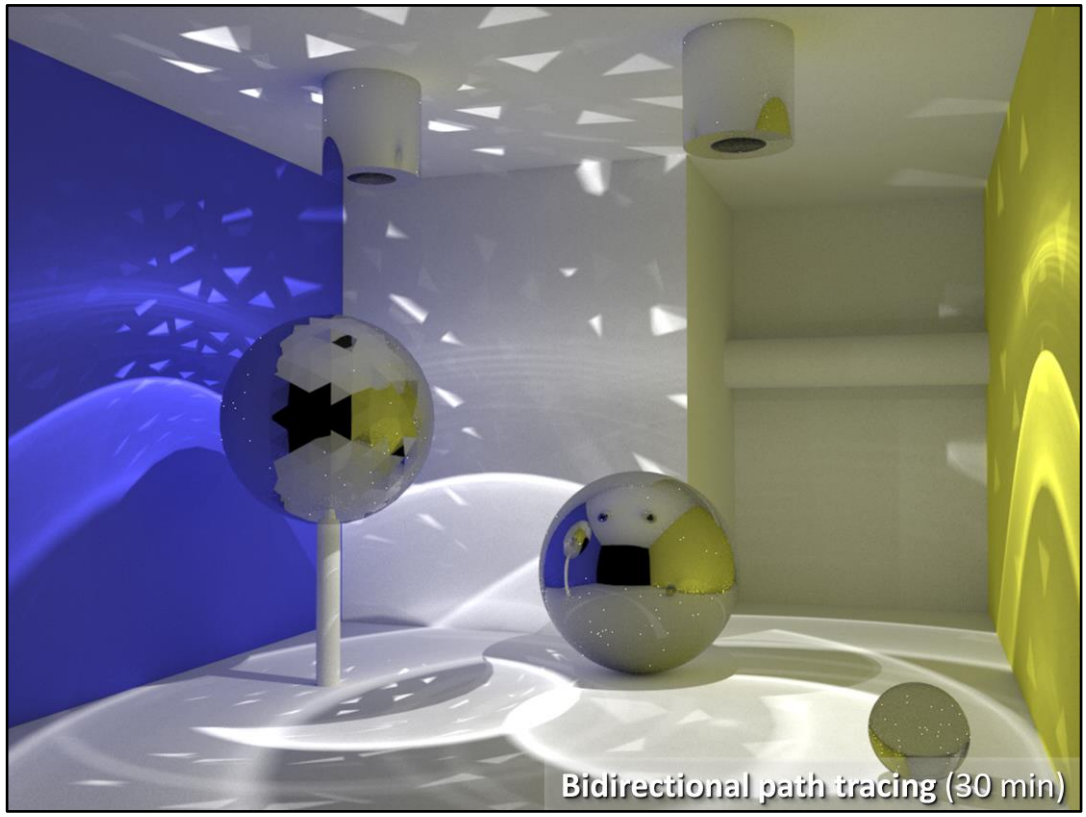
Here, we visualize the relative contributions of VM and VC techniques to the VCM image from the previous slide. This directly corresponds to the weights that VCM assigned to these techniques.

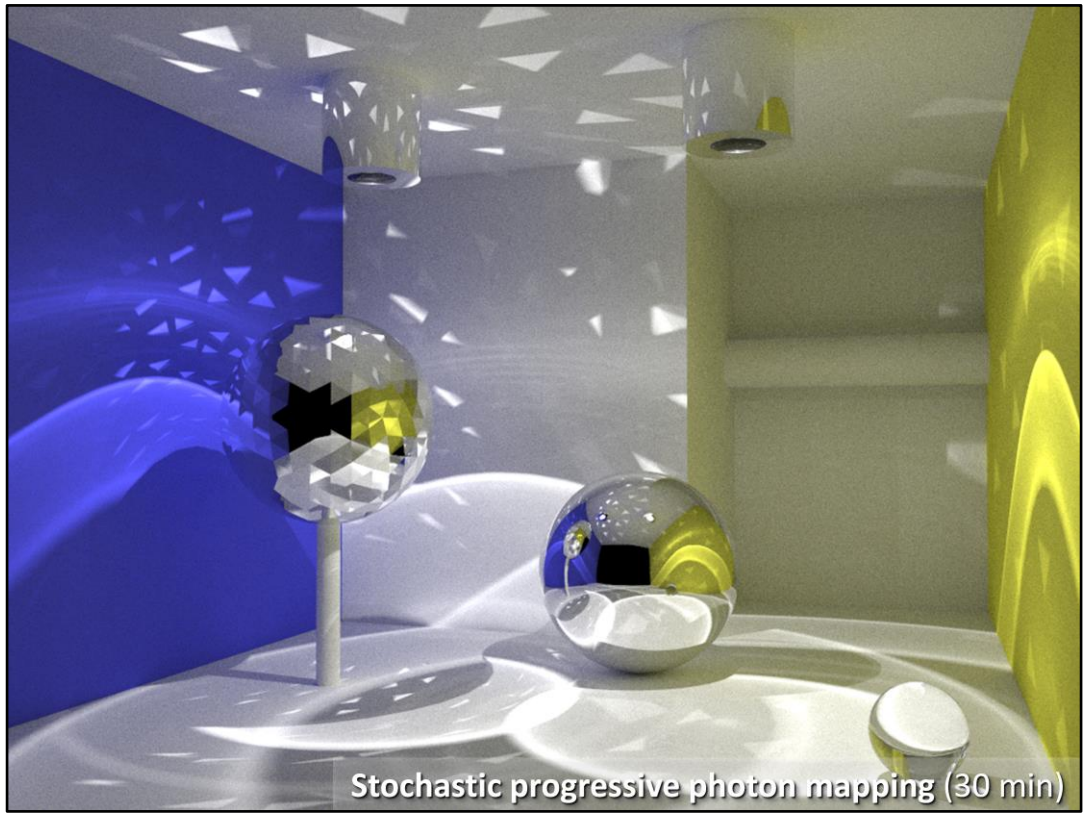

AA-340
Bidirectional path tracing (30 min)

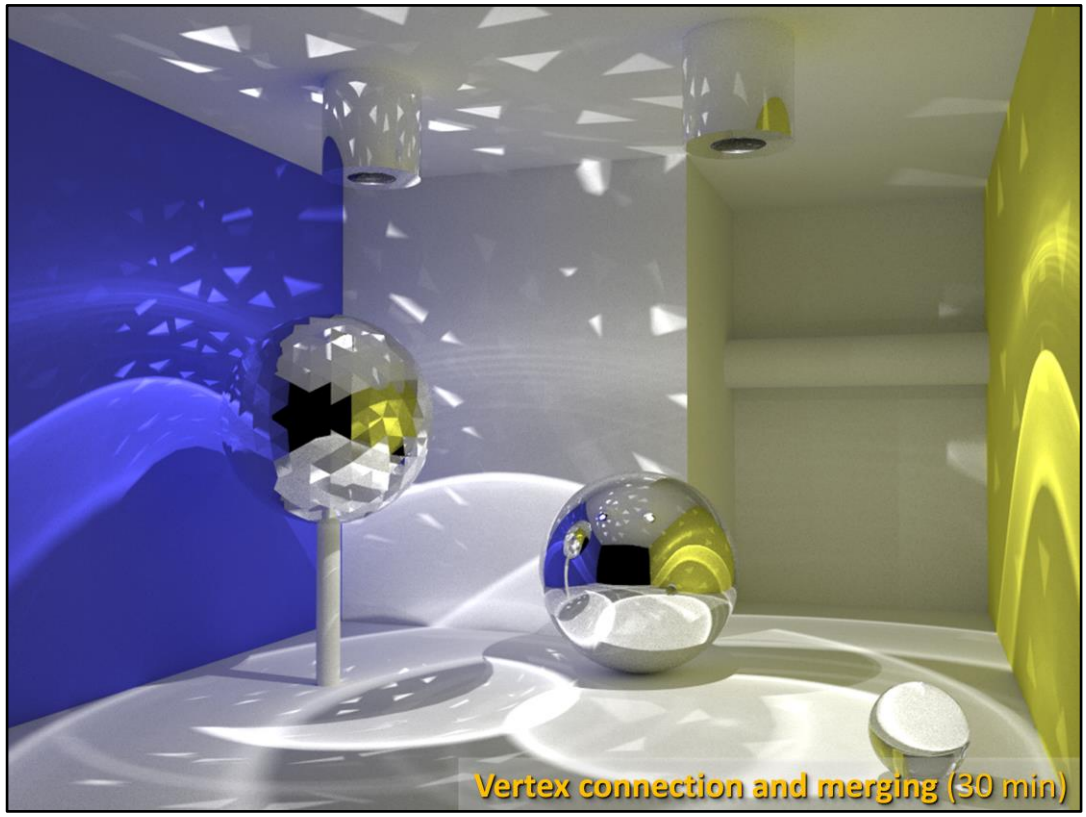

AA-340
Stochastic progressive photon mapping (30 min)

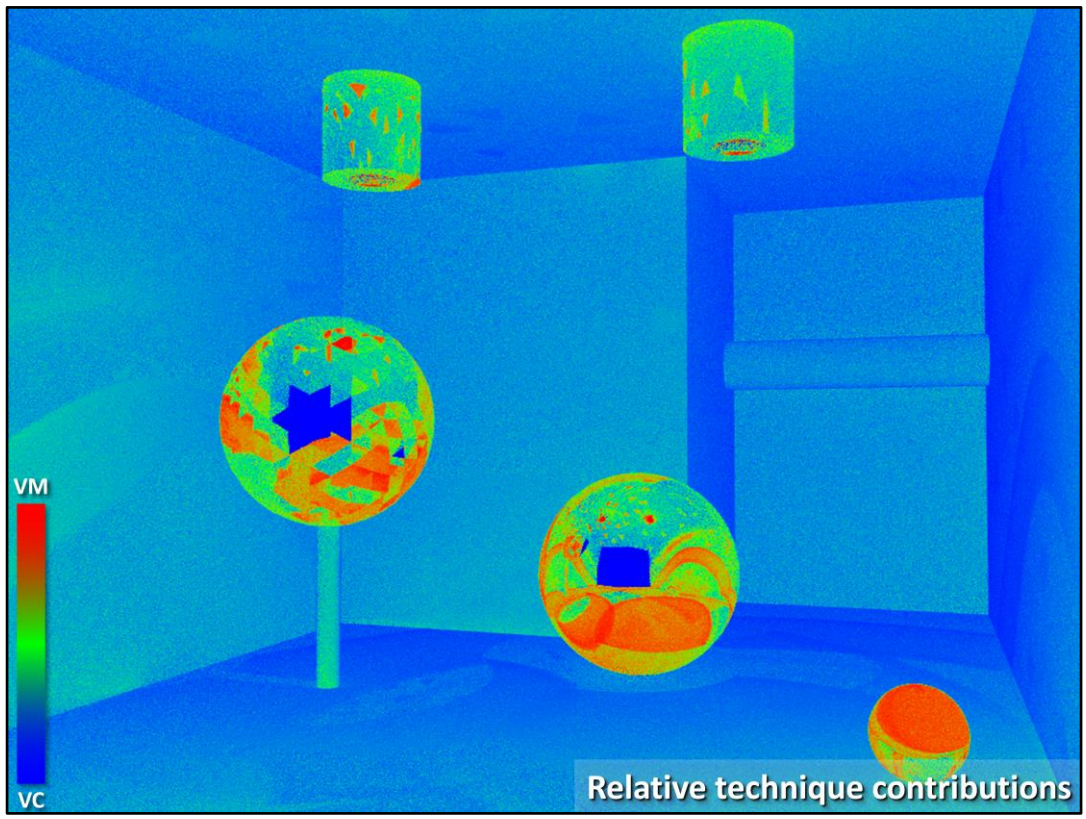

AA-340
Vertex connection and merging (30 min)

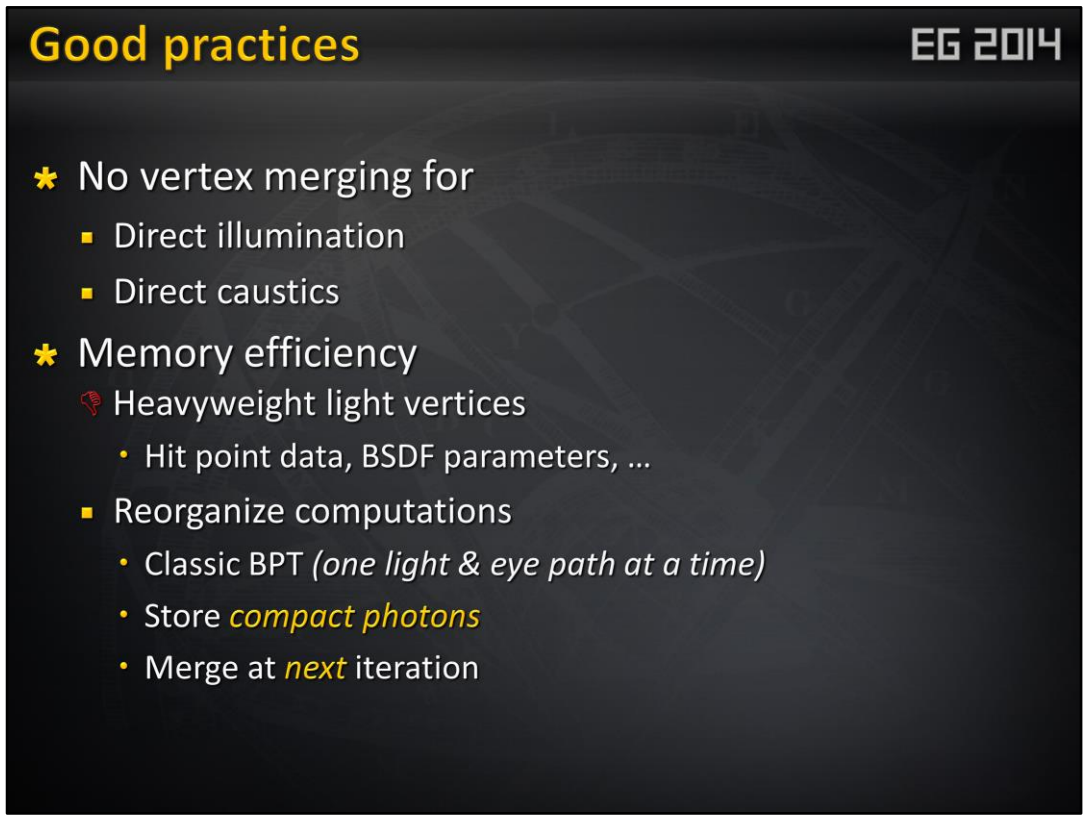

VM
VC
AA-340
Relative technique contributions


Bidirectional path tracing (30 min)


Stochastic progressive photon mapping (30 min)


Vertex connection and merging (30 min)


VM
VC
Relative technique contributions

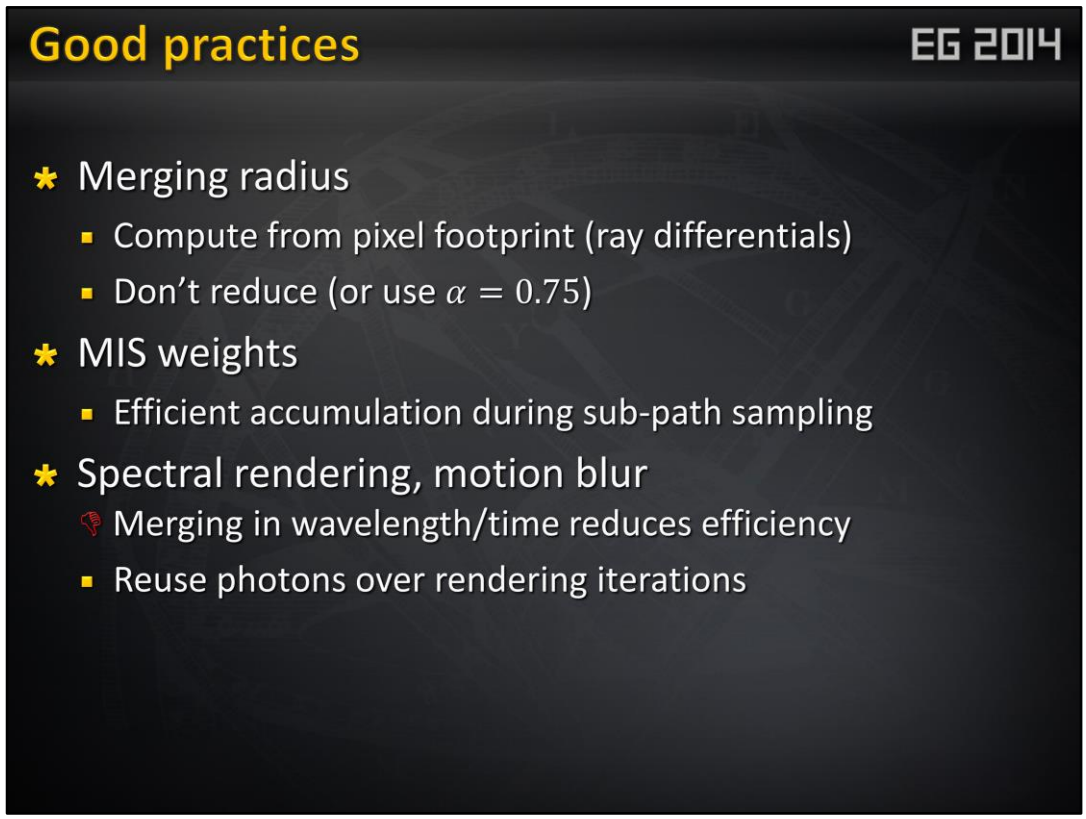


I will now discuss some good practices for the practical implementation of VCM in a production renderer.

We can almost always skip vertex merging (VM) for direct illumination and directly visible caustics, as vertex connection usually performs better. Doing this also avoids the correlated noise and bias inherent to VM.

The combined VCM algorithm I just described has the practical issue that its memory footprint can be much larger than that of (progressive) photon mapping. The problem comes from using the stored light vertices not only for merging, but for connection as well. We need to store hit point data with these vertices, which includes coordinate frame, BSDF structure, and possibly other data, making the vertex footprint as large as 1KB in some cases. This results in 1GB of storage for 1 million vertices.

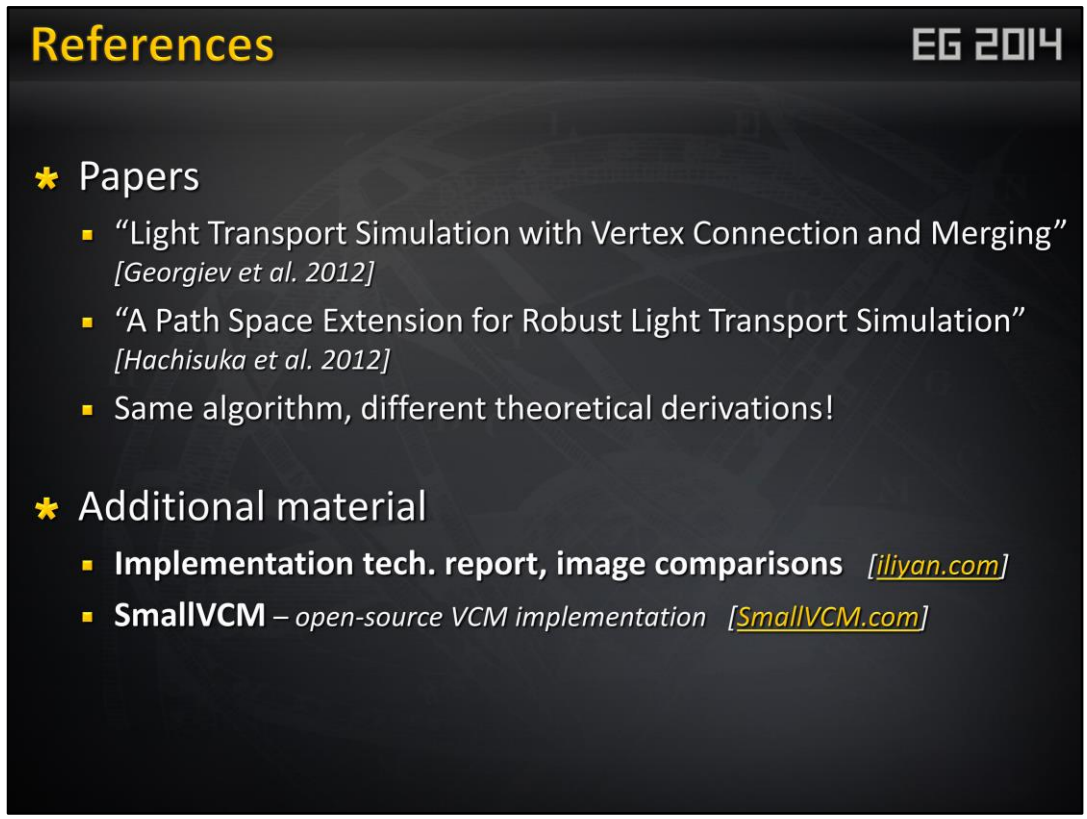
For the progressive variant of VCM, we can dramatically reduce this footprint by rearranging its computations. We follow the classical BPT implementation, where for each pixel we trace one light subpath and one camera subpath. After performing the vertex connections, we store the light subpath vertices in the acceleration structure. These vertices will then be used for merging at the *next* rendering iteration, while the camera subpath vertices for the current iteration are merged with the light vertices from the *previous* iteration. This allows us to safely strip the hit point data off the light vertices before storing them in the structure, as they will be only used for merging. And just like in photon mapping, for merging we only need to keep around a small amount of vertex data, i.e. position, direction, and throughput.



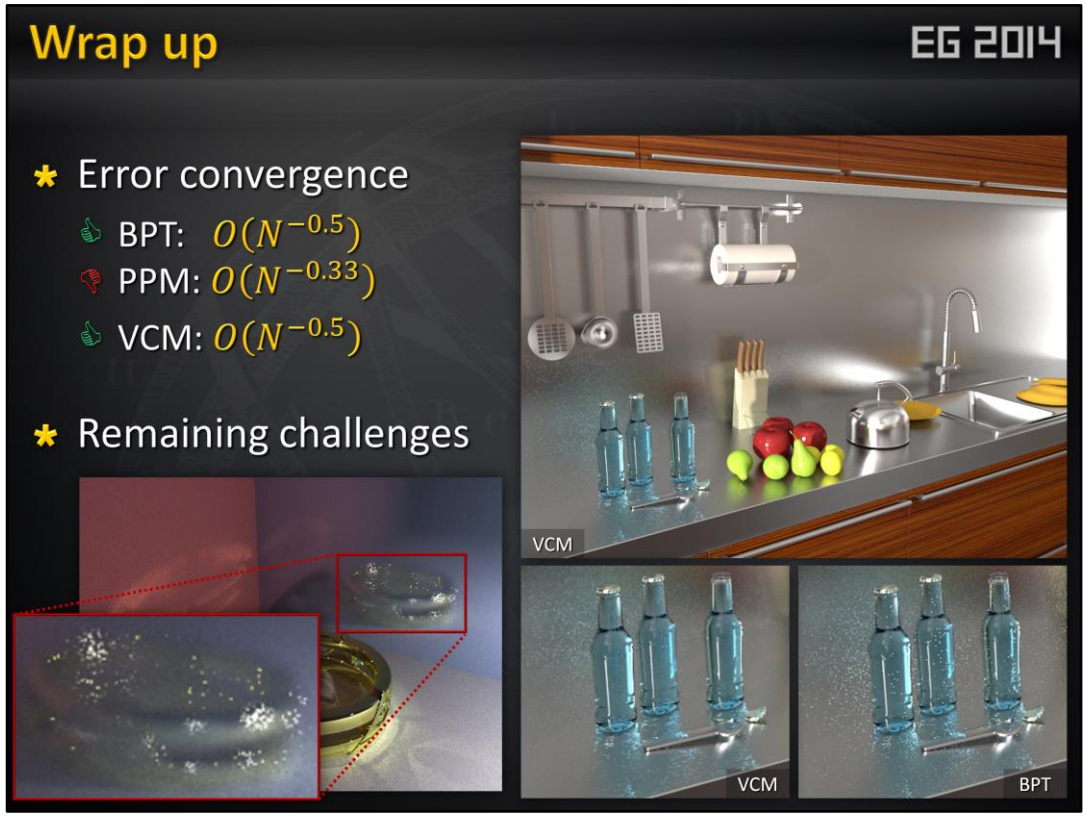
The progressive VCM algorithm has two parameters: the initial merging radius r and its reduction rate α (see "Progressive Photon Mapping" [Hachisuka et al. 2008]). We can often afford to use a much smaller radius than in PPM, as VCM mixes a large number of path sampling techniques, and VM is mostly used for caustics. An automatic and robust way to compute a good merging radius for each camera path is to derive it from the pixel footprint, which is given by the ray differentials that most renderers already implement and use for texture filtering. As for the reduction parameter α, I recommend using $\alpha = 0.75$, which is a provably good value (see VCM paper [Georgiev et al. 2012]), Alternatively, we can also opt to not reduce the radius altogether (i.e. setting $\alpha = 1$), especially when using ray differentials which give a small enough radius to mostly avoid noticeable correlated noise and bias.

Efficient MIS weight computation is another important practical aspect of VCM, and fortunately there exists a scheme for cumulative computation of weight data during the subpath random walks. This scheme is discussed in length in the technical report cited on the next slide.

And finally, a note on spectral rendering and motion blur. When a (sub)path can only carry a single randomly sampled wavelength and/or time instant, in order to merge two subpath endpoints, they not only need to be close to each other in Euclidean space, but also in wavelength and/or time. This can significantly reduce the efficiency of vertex merging, since the VM PDFs are then additionally multiplied by a corresponding wavelength/time acceptance probability, and can thus be much lower. A simple solution to ameliorate this problem is to accumulate and reuse light vertices (photons) over N iterations. This number N depends on the wavelength/time merging radius – the larger the radius, the smaller N can be. Note that efficient light vertex storage in this case becomes even more important.

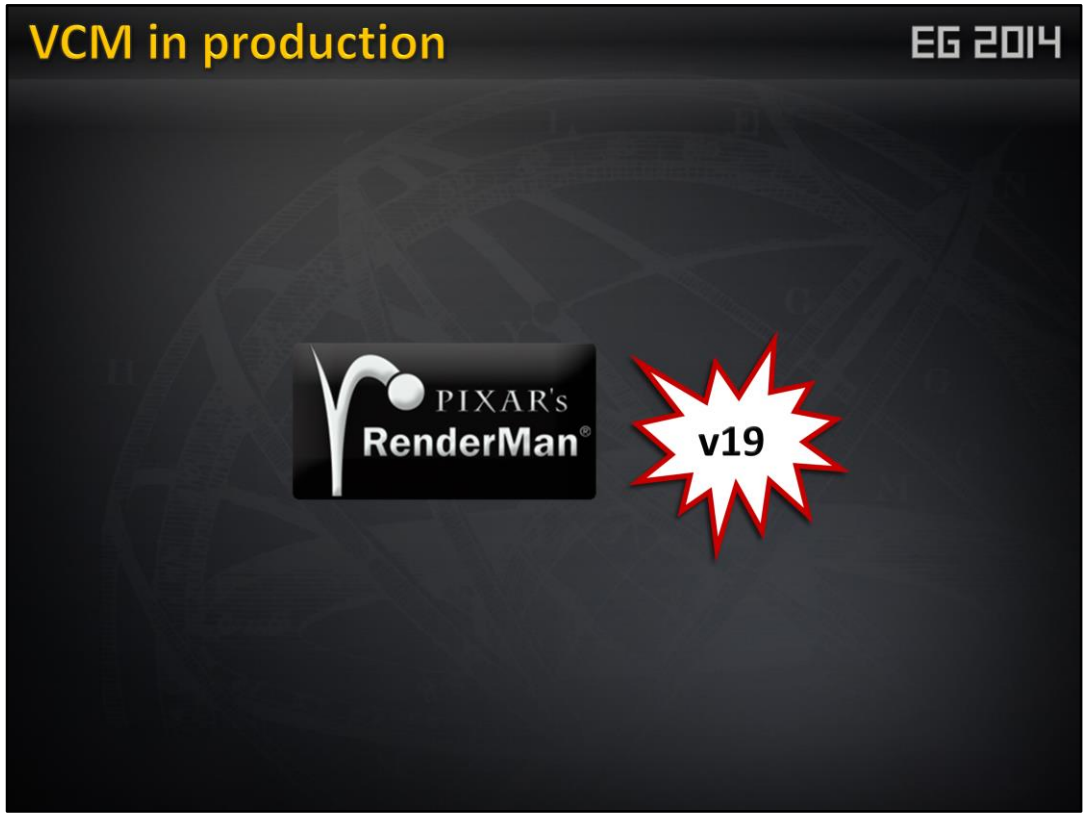


In this talk, I could only give a high-level description of VCM. For more details, please refer to these two papers, which derive the same practical algorithm using two different theoretical formulations. They were simultaneously published by two independent research groups at SIGGRAPH Asia 2012, which we believe only reaffirms the correctness of the approach. Lots of additional material can be found on my web site, and we have also released a reference open source implementation in the SmallVCM renderer.



In summary, vertex connection and merging tries to combine the best of bidirectional path tracing (BPT) and (progressive) photon mapping. An important property of the algorithm is that it retains the higher order of convergence of BPT, meaning that it approaches the correct solution faster than PPM as we spend more computational effort (i.e. sample more paths). The asymptotic analysis can be found in the VCM paper.

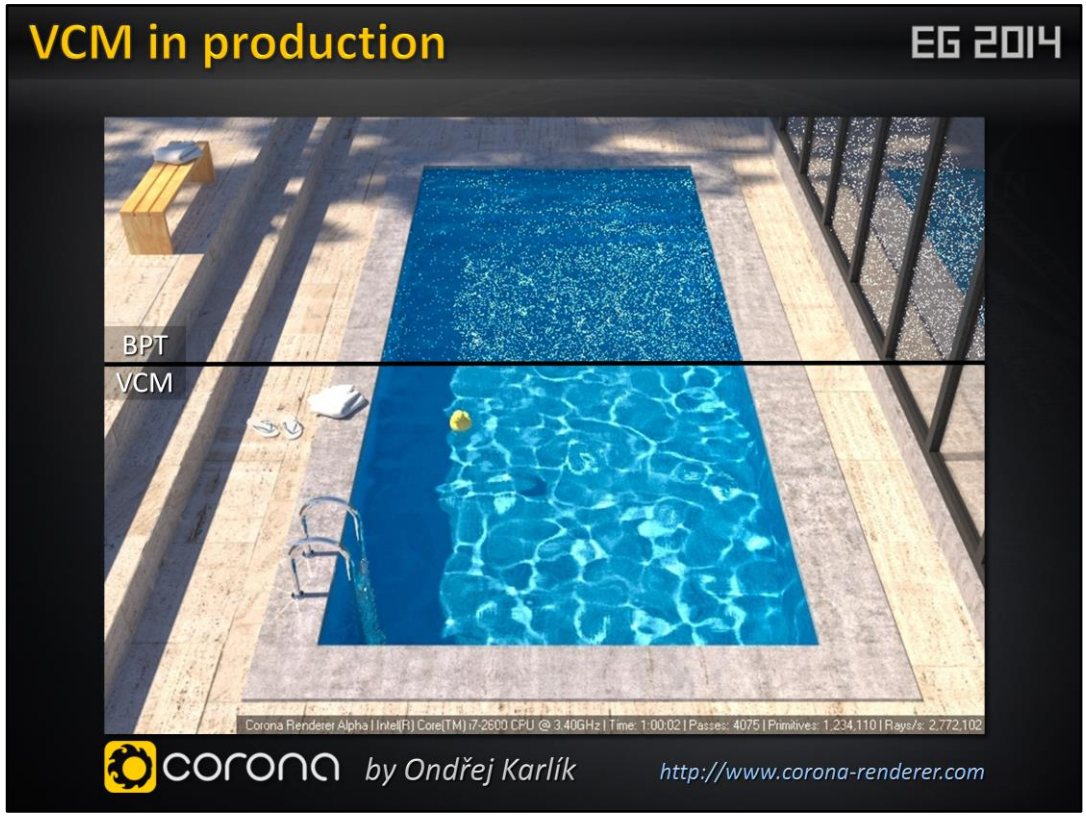
Even though VCM is a step forward in Monte Carlo rendering and has proven very useful in practice, it doesn't come without limitations. Specifically, it cannot handle more efficiently those types of light transport paths that are difficult for both BPT and PM to sample. A prominent example are caustics falling on a glossy surface. On the kitchen scene, even though VCM brings practical improvements over BPT, there is still a lot to be desired from the caustics on the glossy surface.



A number of companies have announced VCM integration in the upcoming releases of their commercial renderers.

For example, VCM is coming in Pixar's Photorealistic RenderMan v19.

Chaos Groups have also shown first images from V-Ray 3.0 with VCM support.



VCM is also implemented in the public Alpha release of Corona renderer, which I encourage you to download and play around with.

Anton S. Kaplanyan:

Markov Chain and Sequential Monte Carlo Methods

Recent Advances in Light Transport Simulation: Theory & Practice

Introduction to Markov Chain and Sequential Monte Carlo

Markov Chains

Markov Chain: Introduction

- Imagine a molecule moving randomly in space
- Current molecule position x_n - *current state*
- Time is discrete ($n = 0,1,2,..$), x_0 - *initial state*
- Set of all possible positions is a *state space*
- It takes a new position x_{n+1} with some probability $T(x_{n+1}|x_n)$ based on position x_n

3

One intuitive way of thinking about a Markov chain would be to imagine it as a molecule performing random movements in a gas or fluid. We imply an important assumption that the molecule has no memory, i.e. its next move depends solely on its current position and does not depend on any of its previous positions. In this case, such memory-less process is called a *Markov chain*. Current position x_n of the molecule at time *n* is called a *current state* of the Markov chain. If time *n* is continuous, the process is called time-continuous Markov Process. Hereafter we will consider only Markov chains with discrete time. The position x_0 of the molecule at time *n*=0 is called an *initial state* of the Markov chain. The space of all possible positions of the molecule is called a *state space*, on which a Markov chain is defined. The molecule moves probabilistically in this medium, such that for each current position of the molecule (current state) x_n and any desired position x_{n+1} (proposal state) for the next move there is defined a conditional transition probability $T(x_{n+1}|x_n)$ of performing such move.

Now we will move away of the molecule example. That means that the state x of the Markov chain can belong to any abstract state space Ω. For example, as we will see later a state x can correspond to a complete light path from a light source to a camera sensor.

Markov Chain

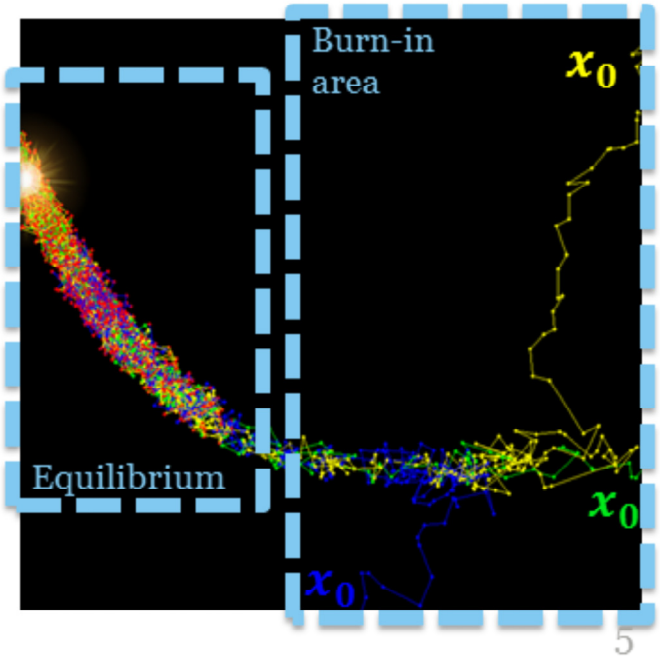
- Random walk implies a *transition probability* $T(x_{n+1} = j | x_n = i) \equiv T_{i \to j}$ for each move
- At each move the chain forms a *posterior distribution* over state space
  - A histogram of all visited states up to move n
- *Detailed balance* defined as $T_{i \to j} = T_{j \to i}$

4

The random walk implicitly defines the conditional transition probability T for every move. For every current state i and a proposal state j at move n a transition probability $T(x_{n+1} = j | x_n = i)$ is defined. Hereafter we will denote this probability simply as $T_{i \to j}$ for brevity. After n moves we can build a histogram of all visited states up to the move n. This histogram forms a *posterior* probability distribution which evolves with every move n. This posterior distribution is induced by the constructed Markov chain.

One important property of the transition probability is called *detailed balance*. It implies symmetry of the transition probability, i.e. the transition from state i to state j is equally probable as the transition back from j to i. Intuitively that means that the random walk can be reversed. A Markov chain running with such symmetric transition probability is called a *reversible* Markov chain. Physically speaking, detailed balance implies that, the random walk from state *i* to state *j* is always compensated by the reverse random walk, keeping the system in equilibrium.

Markov Chain



- Posterior converges to the *target* distribution *if* the detailed balance obeyed and all states are reachable (*ergodicity*)
- With "bad" initial state x_0 the *start-up bias* (burn-in phase) can be significant

5

If the detailed balance is obeyed and all states of the path space can be reached by the proposed transition rules the proposal distribution then a Markov chain converges to a unique stationary distribution as number of moves $n \to \infty$.

Such stationary distribution is called an *equilibrium* or *target distribution* of a constructed Markov chain and the chain is called *ergodic*.

The trajectory to the target distribution can take many steps and depends on the initial state x_0 of the Markov chain. Illustration on the right show three different Markov chains (yellow, green and blue) converging to the same equilibrium distribution from different initial states. The highlighted equilibrium zone roughly denote the high-probability region of the unimodal banana-shaped target distribution and receives the most samples.

The zone on the right half of the illustration shows the so-called "burn-in zone" – the phase when a Markov chain is located in the regions of low probability of the target distribution. If such phase it too long, it can shift the resulting posterior distribution from the target distribution, because a lot of visited states can lie in the low-probability regions of the target distribution. Thus this effect caused by a not properly chosen initial state is called *start-up bias*.

So practically it is very important to seed a Markov chain with an initial state that lies in high probability regions of the target distribution. That leads to faster convergence of the posterior distribution to the target distribution.

Metropolis-Hastings Algorithm

Metropolis-Hastings (MH) Algorithm

- Goal: Random walk according to a desired function f
- Define conditional rejection sampling probability

$$a_{i\to j} = \frac{f(x_j)}{f(x_i)} = \frac{f_j}{f_i}$$

  - $a_{i\to j}$ is *acceptance probability* at state i for proposal state j
- Detailed balance is affected as $a_{i\to j}T_{i\to j} = a_{j\to i}T_{j\to i}$
- Posterior distribution is then proportional to f
  - Accurate to a scaling factor = normalization constant

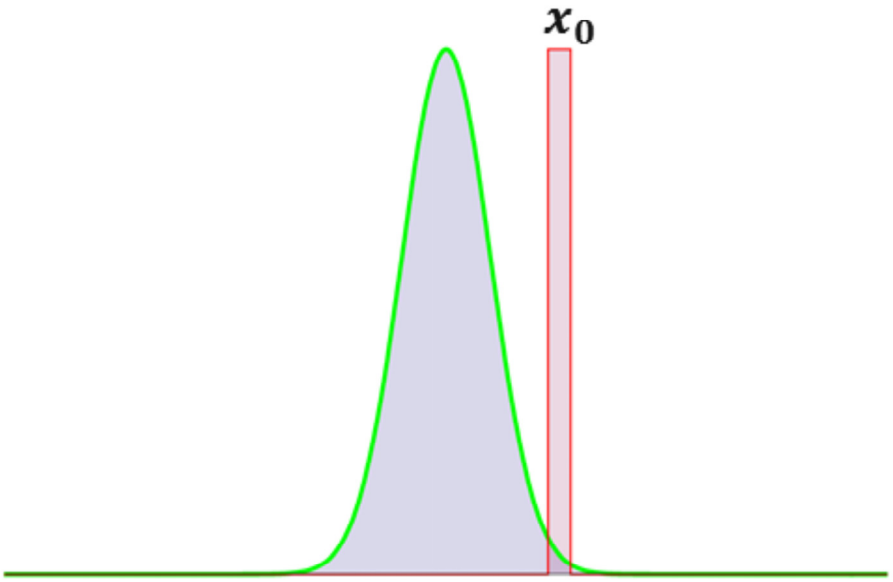
7

Assume we have some function f that can be evaluated point-wise. We cannot directly sample from f. However we still want to sample proportionally to f.

We can construct a Markov chain, whose posterior distribution converges to the function of interest f accurate to the normalization factor (since f is generally not normalized). This method was first proposed by Metropolis in 1953 and then generalized by Hastings in 1970 for arbitrary target distributions. The idea is to alter the transition distribution by affecting it by a conditional rejection sampling probability based on the desired target distribution. This probability is similar to the ordinary rejection sampling probability, however the key difference is that it is conditional on the current state, that is, $a_{i\to j} = \frac{f_j}{f_i}$ means that it's a probability of conditionally accepting the new proposal state j given current state i. It is called an acceptance probability because at each move n of the Markov chain we either accept the proposal state j with probability $a_{i\to j}$ or otherwise reject it and keep the current state i (with probability $1 - a_{i\to j}$ accordingly).

Given that the transition probability is selected such that the detailed balance is obeyed, the posterior distribution of the constructed Markov chain will converge to the desired target function f. Note that the detailed balance equation is affected by this acceptance probability. Important to note that the Metropolis-Hastings algorithm always constructs a normalized pdf, while the original target function is not necessarily (and is usually not) normalized. Thus this poses another important problem of finding the normalization constant of f, i.e. the integral of f over the whole state space. This can be as hard as the original problem. However we will see that for particular needs of light transport it can be easily estimated using one of alternative methods.

Metropolis-Hastings: Example



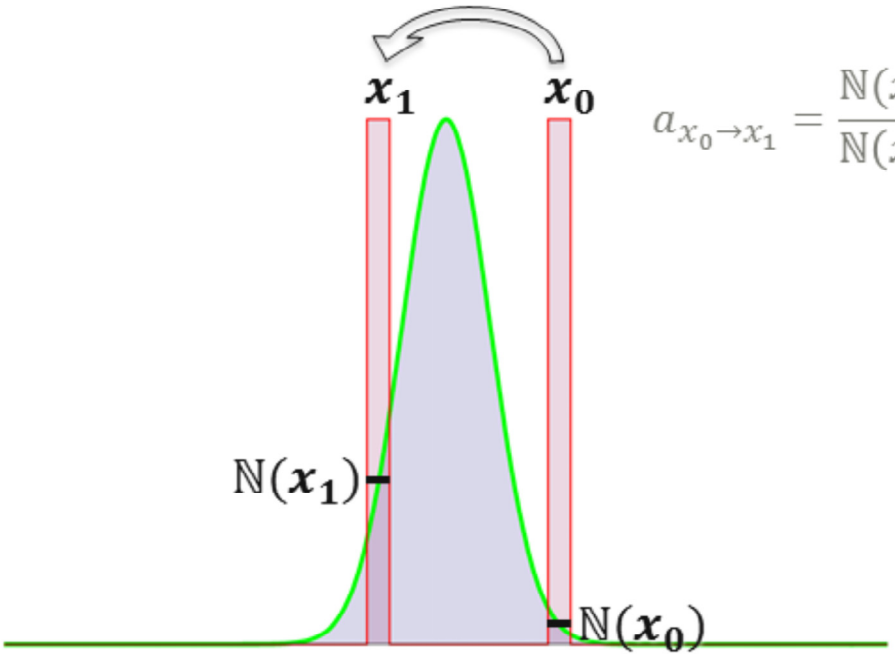
8

Here we show a simple step-by-step example of Metropolis-Hastings algorithm in action. We consider a simple 1D case of unimodal normal distribution $\mathbb{N}$ depicted in green. We will run Markov chain with uniform random walk and the acceptance probability being the ratio of values of proposal to current state, as was described.

We start with some initial state x_0 chosen to be close to the high-density region of the target function.

This already forms the posterior distribution with one bar of the histogram, depicted in red under the position of the initial state.

Metropolis-Hastings: Example

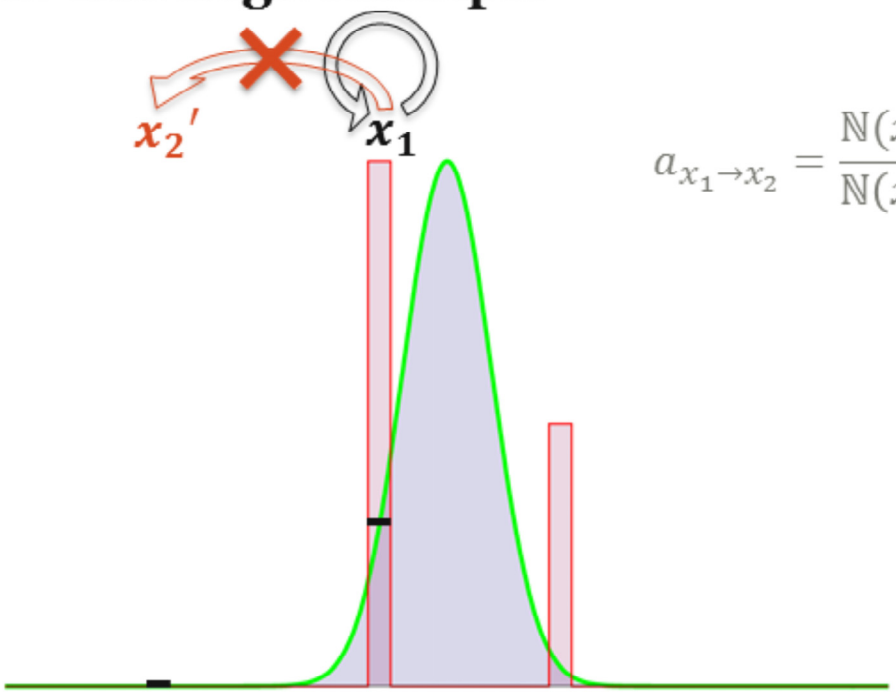


$$a_{x_0 \to x_1} = \frac{\mathbb{N}(x_1)}{\mathbb{N}(x_0)} > 1$$

9

In order to make the first move we generate a proposal state x_1 with uniform random walk and compute the acceptance probability as $a_{x_0 \to x_1} = \frac{\mathbb{N}(x_1)}{\mathbb{N}(x_0)} > 1$. That implies an unconditional acceptance of the new proposal state.

Metropolis-Hastings: Example



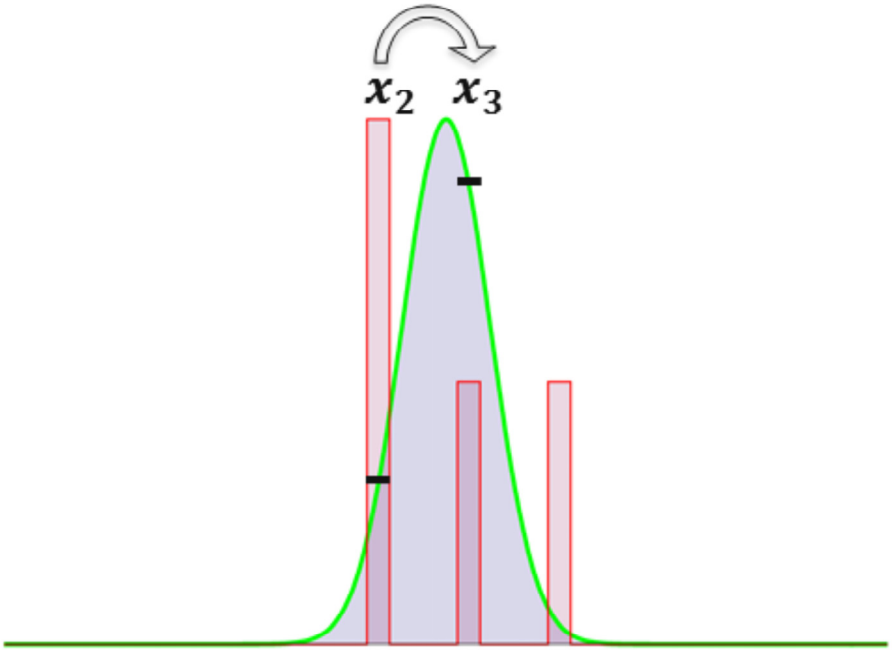
$$a_{x_1 \to x_2} = \frac{\mathbb{N}(x_2)}{\mathbb{N}(x_1)} \ll 1$$

10

At the second step again we generate a new proposal, this time from a low-density region. Thus the acceptance probability $a_{x_1 \to x_2} = \frac{\mathbb{N}(x_2)}{\mathbb{N}(x_1)}$ is very low. Thus if we make a move, most probably such a proposal will be rejected.

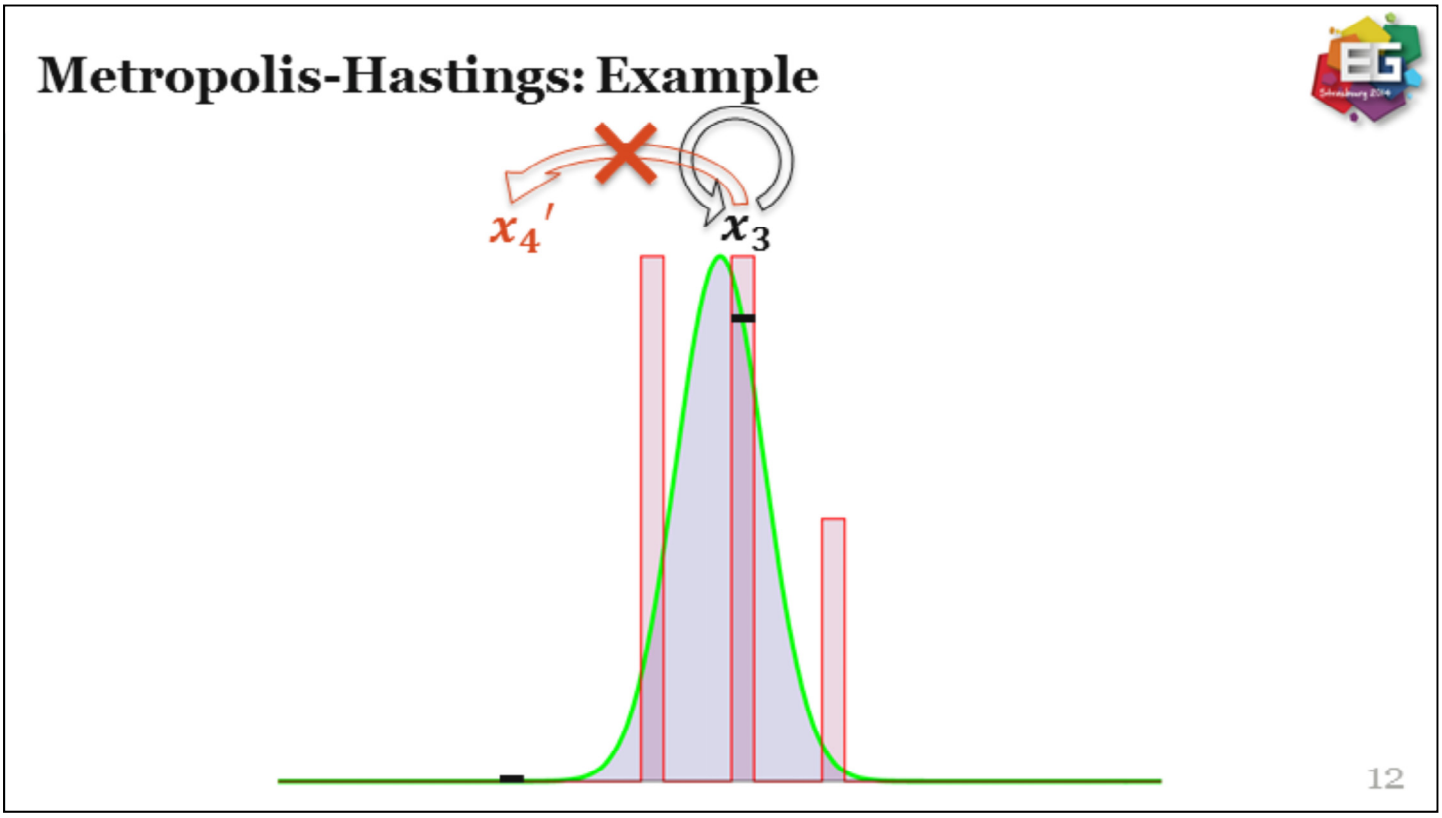
It is important to note at this point that the current state of the Markov chain does not change (x_2 becomes x_1), however every move affects the histogram. Thus the peak at the current state become more prominent.

Metropolis-Hastings: Example

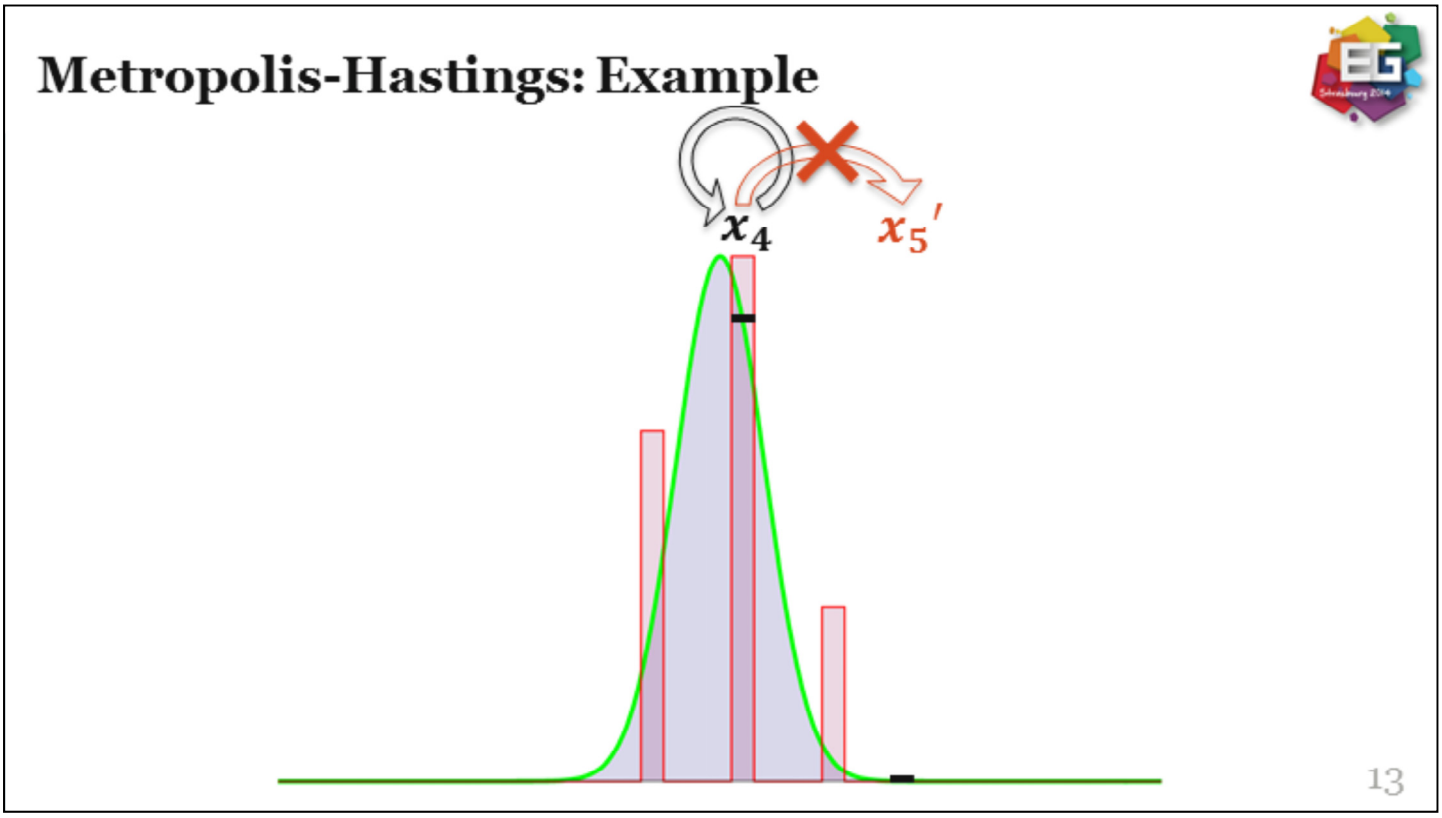


11

Now we generate a proposal in a high-density region. That leads to unconditional acceptance of the new proposal x_3.

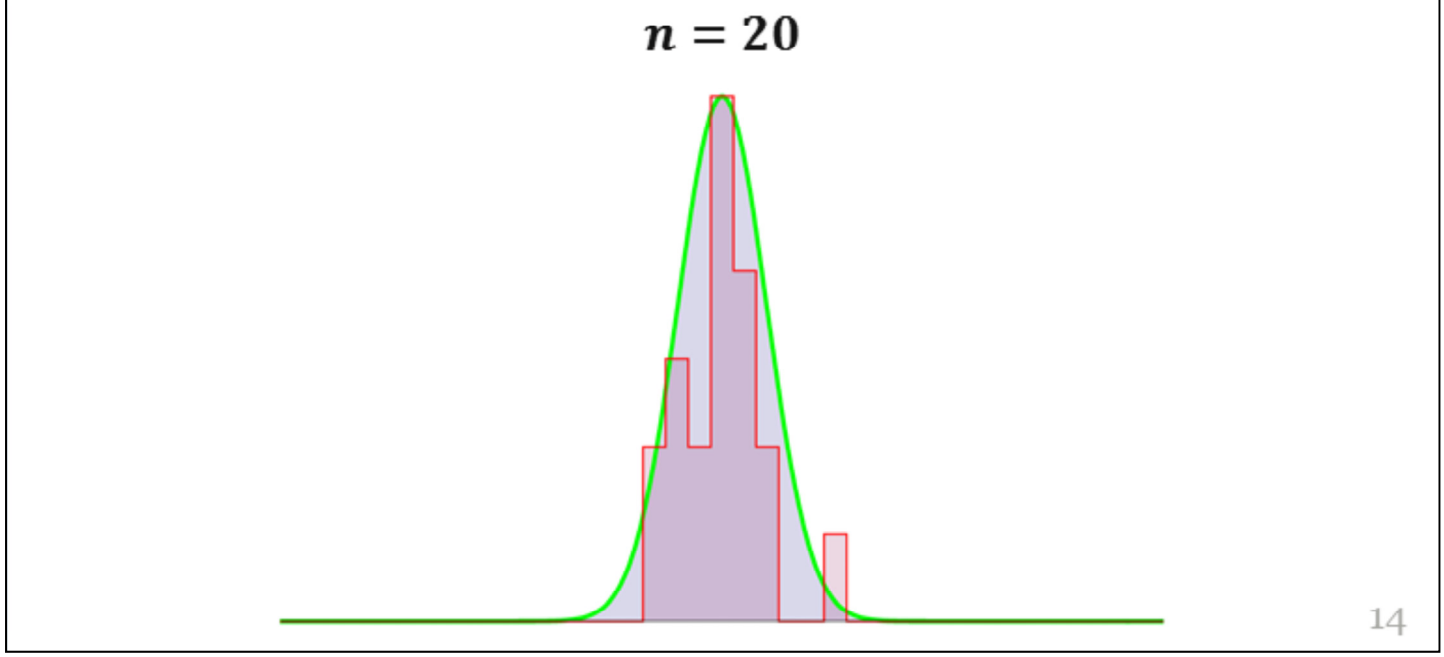


The next proposal is also rejected due to the low value of the target function. Again that leads to one more sample added to the histogram at the current state.



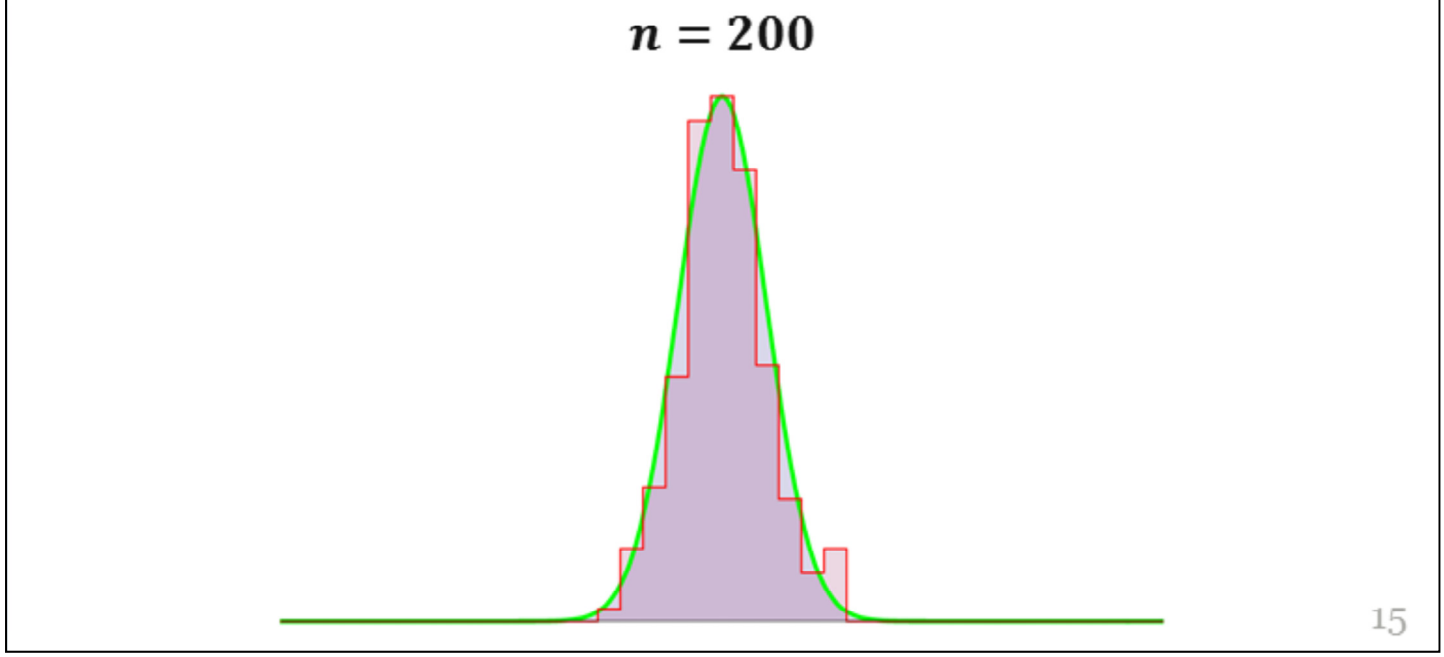
One more time the proposal is rejected. This is the natural behavior if the chain is stuck in a very high-density region of the target function. Note that many sequential rejections might also increase the correlation of samples, thus slowing down the convergence.

Metropolis-Hastings: Example



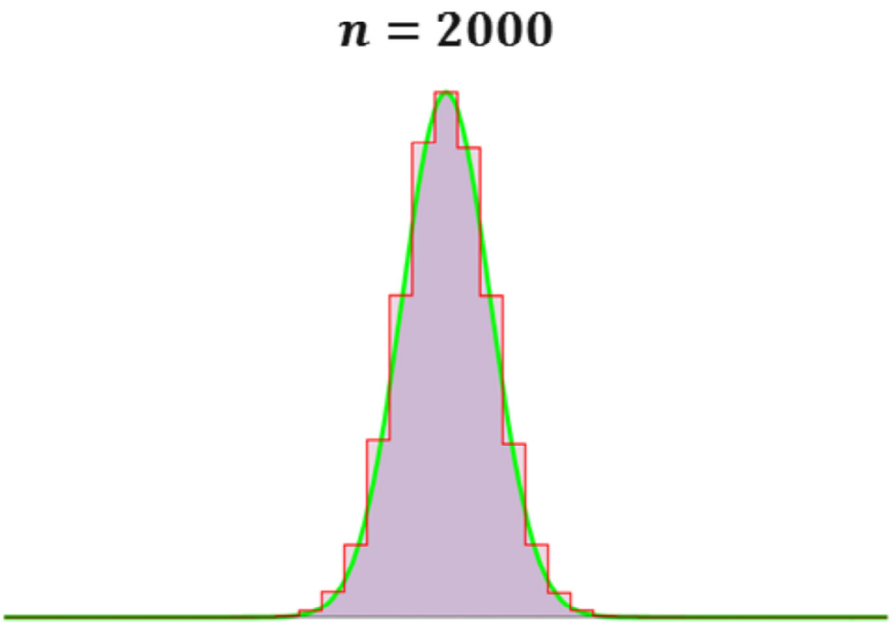
This is the posterior distribution produced by the Markov chain after 20 moves.

Metropolis-Hastings: Example



This is the posterior distribution produced by the Markov chain after 200 moves.

Metropolis-Hastings: Example



16

This is the posterior distribution produced by the Markov chain after 2000 moves. As we can see the posterior distribution closely converges to the target function.

Importance Sampling

- Cannot fetch proposals directly from f
- Generate a proposal j from some pdf T_j
  - Similar to importance sampling in Monte Carlo
  - T can depend on the current state i: $T_{i \to j}$
- Acceptance probability is then

$$a_{i \to j} = \left(\frac{f_j}{T_{i \to j}} \right) / \left(\frac{f_i}{T_{j \to i}} \right)$$

17

In practical situations, doing just a uniform random walk around the state space might lead to a poor exploration of state space (important features might be missing or under-sampled).

In this case, one can generate new proposals according to some importance function *T*, which is somewhat similar to *f*.

This is almost equivalent to the importance sampling technique used in Monte Carlo.

The key difference is that such importance function in MCMC can also depend and rely on the current state! This function is called transition probability function.

And the acceptance rate should account for this transition probability similarly to Monte Carlo methods (by dividing the value of *f* by the probability of sampling it).

Correspondence Table

Ordinary Monte Carlo	Markov chain Monte Carlo
Convergence rate, usually $O(\frac{1}{\sqrt{N}})$	Mixing rate, depends on multiple factors, can be geometric $O(\gamma^N), \gamma \in (0; 1)$
Convergence to an expected value	Convergence of the posterior to the target distribution (e.g., in total variation)
Importance sampling distribution $p(x)$	Transition probability $T_{i \to j}$
Variance of the estimate	Acceptance rate, correlation of samples
Number of samples	Number of moves (mutations)

18

This table shows the correspondence between the major terms and properties used in ordinary Monte Carlo (MC) to the equivalent terms used in Markov Chain Monte Carlo (MCMC).

Note that the theoretical convergence of MCMC methods can be fundamentally different comparing to ordinary MC methods.

The convergence in context of Markov chains is usually referring to the convergence of the posterior distribution to the target distribution (in some norm, for example, in total variation).

We have transition probability instead of importance function. Note that we have much more freedom in constructing this transition probability function, because we can also rely on the current state of a Markov chain.

The error of MH is very hard to compute, since the samples are inherently correlated. So, we cannot use just variance anymore. Acceptance rate can be a good initial indicator of the MCMC sampler performance.

And, instead of number of samples, we have number of moves made by Markov chain.

Metropolis Light Transport

Now, I'll try to explain how we can apply MH algorithm in the context of light transport.

Image Generation

- Reduce per-pixel integrals to a single integral
  - Each pixel has an individual filter function then
- Compute the distribution over the image plane
  - Bin this distribution into corresponding pixels
- Walk over the image plane

20

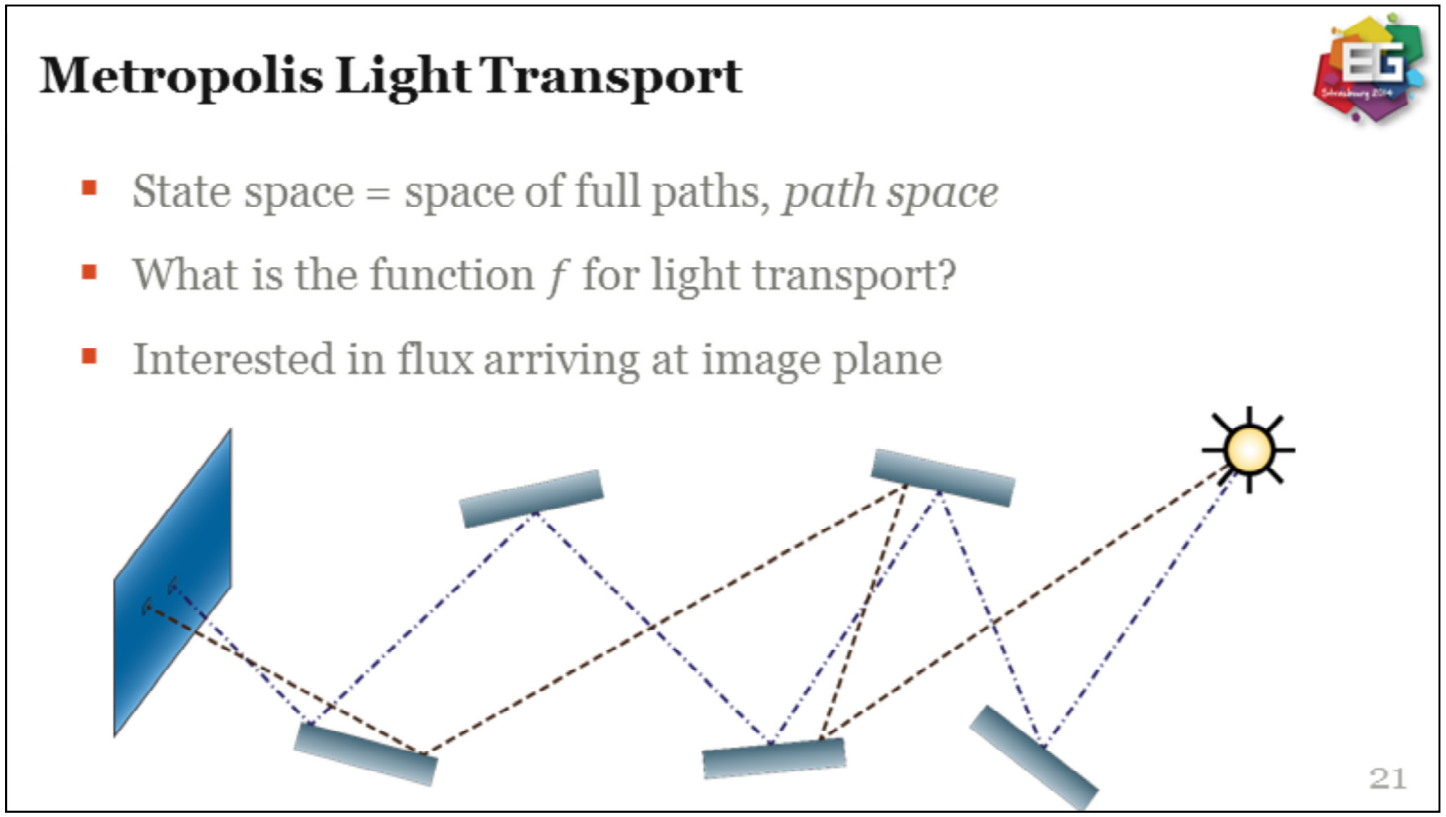
In order to achieve more efficient exploration of the path space and utilize the potential correlation between the separate integrals for each pixel, we reduce the task to a single integral. Each pixel integral can be then deduced by applying a pixel filter.

This single integral computes the distribution of flux on the image plane.

Then we can obtain the image by just distributing the corresponding samples from the posterior distribution into the corresponding bins of image pixels.

This way the MH algorithm is able to freely walk over the complete image plane while exploring important parts of the image adaptively.

Metropolis Light Transport



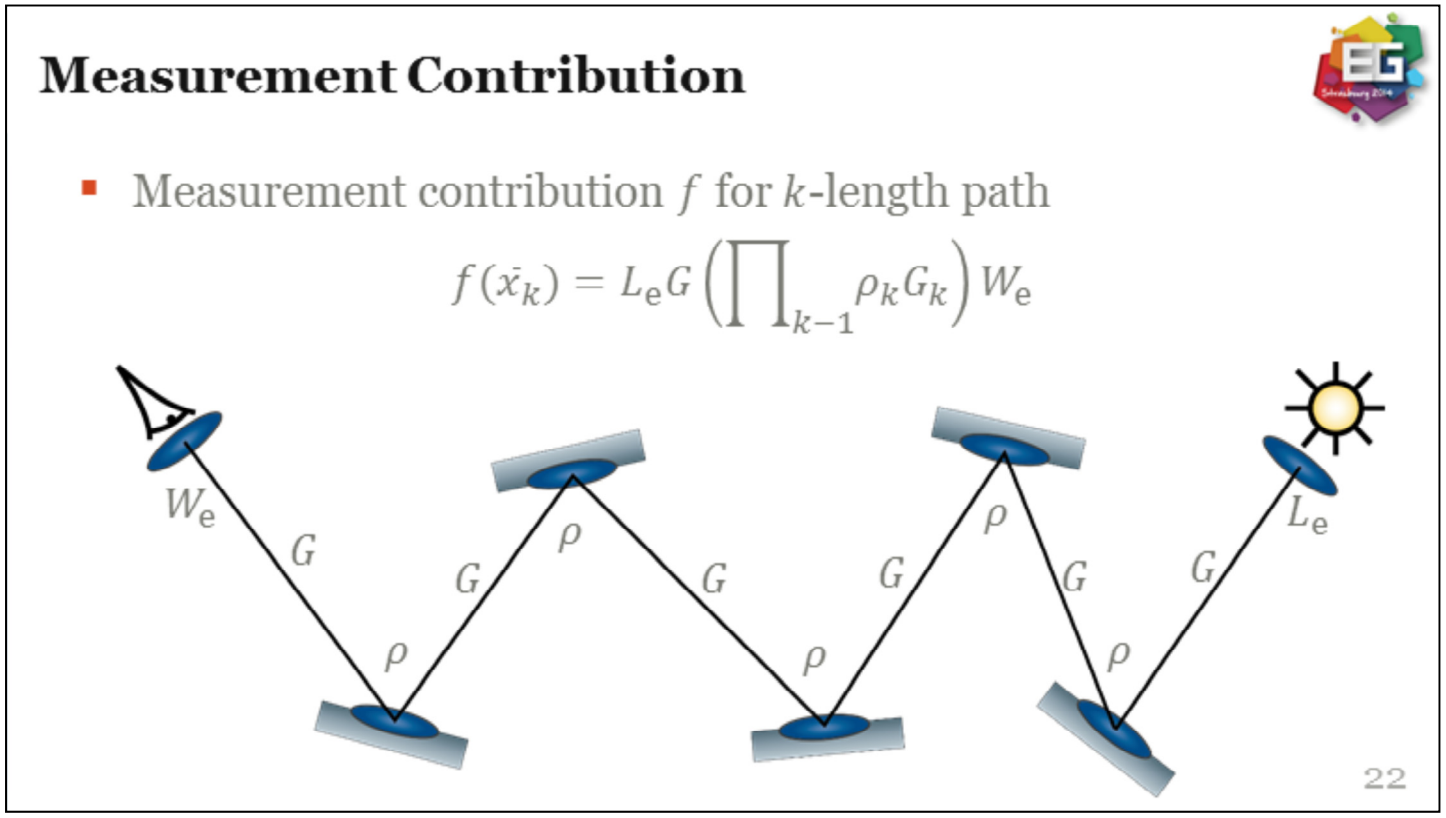
So, ideally, we are interested in all possible trajectories (paths) from light source to camera.

This naturally forms the state space for a Markov chain: now each state is a full path from light source to the camera. This state space is called a *path space* in light transport.

How would one define a target function for Metropolis-Hastings algorithm in context of light transport?

Ideally, we're interested in the equilibrium distribution of flux incident to the image plane.

Measurement Contribution



Thus, as was explained before by Jaroslav, we can introduce the measurement contribution function for a path x_k.

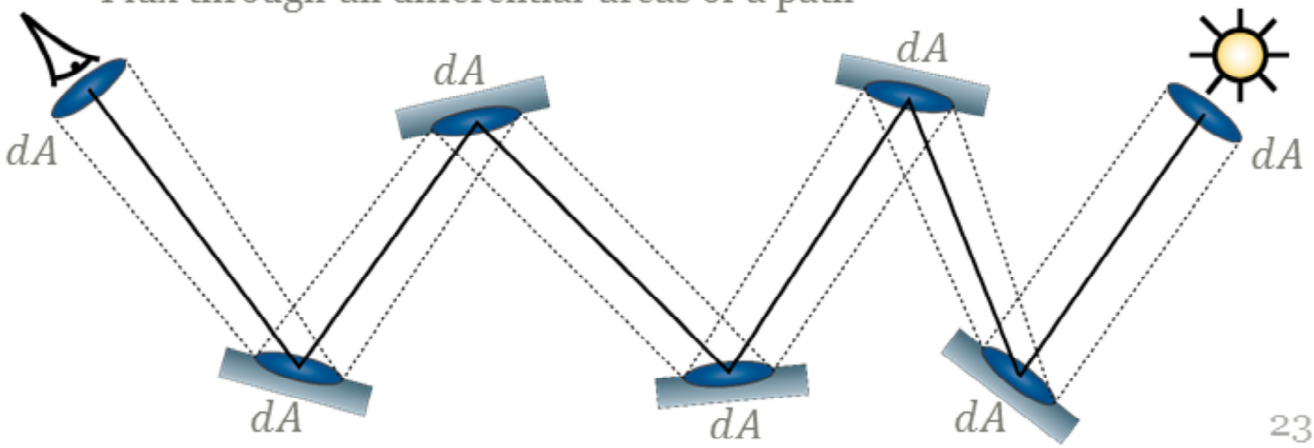
It consists of subsequently interleaved events: emission(L_e)->propagation(G)->scattering(f_r)->propagation(G)-> …. -> absorption by sensor (W_e)

and provides the contribution carried by the path.

Measurement Contribution

$$f(\bar{x}_k) = \prod_k \frac{dQ}{dA_k} = \frac{dQ}{d\mu_k} \qquad [W/(m^2)^k]$$

– Flux through all differential areas of a path



Let's take a closer look at the physical meaning of the measurement contribution f.

Eric Veach showed in his PhD thesis that we can define the measurement contribution as chain derivative of energy with respect to surface areas at each interaction. By folding the product, we get that the measurement contribution f is a derivative of energy function Q with respect to the area product measure μ_k.

The physical quantity of the measurement contribution is Watts per square meter to the kth power.

Intuitively, it defines an energy flow through a differential beam around the path.

In other words, we count the number of photons going through the infinitesimal beam around the path.

This definition reveals the underlying physical justification behind Metropolis light transport.

Comparing Paths

- MH needs to compare two states (paths)
  - Use flux through the infinitesimal path beam
- Directly comparable for equal-length paths
  - Compare flows of energy through each path
- For different lengths the measure is different
  - Always compare fluxes going through each path

24

In context of MH, we always need to be able to compare two different paths.

As we know, the measurement contribution *f* provides the amount of flux going through the infinitesimal beam around the path.

This makes the paths of equal length directly comparable to each other in terms of the carried energy.

The only remaining question is how to compare paths of different lengths?

Interestingly, if we define our integration measure as a product area measure, which adjusts based on the path length, then we can directly compare the amount of energy (e.g. number of photons) going through the path.

Path Integral

- For path of length k: $I_k = \int_{\Omega_k} f(\vec{x}) d\mu_k(\vec{x})$
- Combine all path lengths into a single integral
  - Use unified measure for all paths

$$d\mu(D) = \sum_{k=1}^{\infty} d\mu_k(D \cap \Omega_k)$$

  - Compare paths of different length
  - Compare groups of paths

25

So, we can construct a MH integration process for all paths of the same length *k*.

However, we need to construct a single generalized integral.

This can be done by just treating this family of integrals as a single generalized path integral.

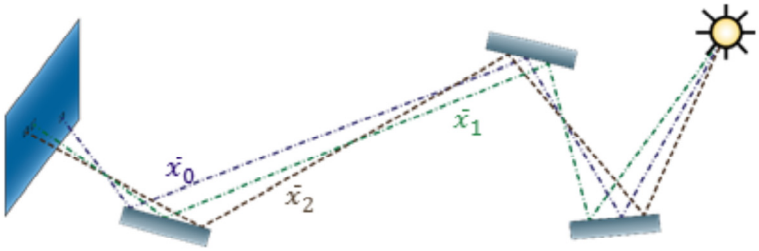
In this case, we can introduce a generalized product area measure $d\mu$ as a sum measure.

This enables us to use a single integral for paths of all lengths for MH.

Moreover, this way we can compare all paths and even groups of paths with each other in the context of carried energy (flux).

Metropolis Light Transport

1. Generate initial path $\bar{x}_0$ using PT/BDPT
2. Mutate with some transition probability $T_{\bar{x}_i \to \bar{x}_j}$
3. Accept new path $\bar{x}_j$ with probability $a_{\bar{x}_i \to \bar{x}_j}$
4. Accumulate contribution to the image plane
5. Go to step 2



26

We showed how light transport can be reformulated for MH integration.

Now I'll outline the actual steps of the MLT algorithm:

1. We generate an initial state (full path) of a Markov chain using one of the existing sampling methods (e.g. PT or BDPT).
2. Then we start the actual mutation process. Mutate the current path using one of the available mutation strategies, compute the transition probability.
3. Compute the acceptance probability, accept the new proposed path according to this probability.
4. Accumulate the contribution of the current path to the image plane, apply pixel filter to bin the path into the corresponding pixel in-place.
5. Proceed to step 2 to cycle the random walk.

Advantages

- More robust to complex light paths
  - Remembers successful paths
- Utilizes coherence of image pixels
  - Explores features faster
- Cheaper samples
  - Correlated
- Flexible path generators (mutations)

27

I'd like to emphasize that we have already quite good methods for image rendering, like BDPT.

So, why do we need yet another, way more complicated rendering method?

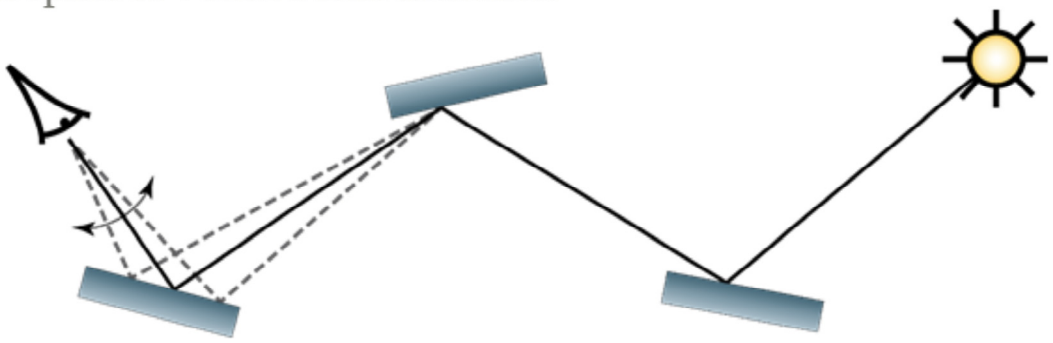
First of all, MLT is much more robust to the complex light paths, meaning that it tries to "remember" the successful paths. That is, the current state of a Markov chain is always a correct full path from light source to the camera.

As another advantage, Markov chain can easily explore the similar surrounding paths by perturbing the current path slightly, thus exploring the whole illumination features at a low cost. On the other hand, this can also cause some unwanted correlation of samples, slowing down the rendering convergence.

And last, but not the least, MLT framework provides us the great freedom of constructing path generators for almost any special situation.

Variations and Improvements

- Energy redistribution path tracing [Cline05]
  - Run many short Markov chains for each seed
  - Adaptive number of chains according to path energy
  - In spirit of Veach's lens mutation



28

ERPT utilizes the fact that independent samplers, like BDPT, already provide a very good distribution.

The idea is to try to redistribute the amount of energy carried by each initial path.

In order to do that, multiple Markov chains are started with the same seed path, the number of chains is computed adaptively based on the path energy.

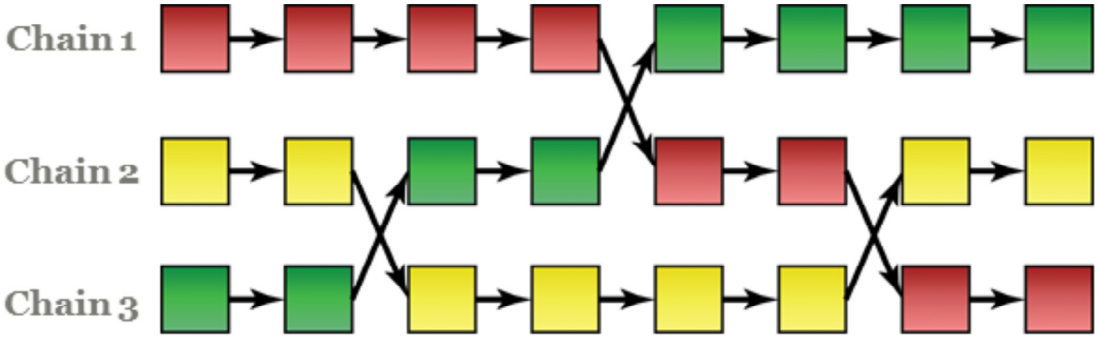
This scheme is very similar to the lens mutation proposed by Eric Veach with the number of mutations between reseedings of the chain being very low.

Efficiency of ERPT depends a lot on the seeding sampler – how good it is. E.g. BDPT w/o MIS provides very unbalanced sampling, leading to ERPT being stuck for a long time in some regions due to high redistribution workload.

Moreover, the distribution region is manually set, making it non-trivial to tweak the parameters to achieve the best redistribution vs. stratification trade-off.

Variations and Improvements

- Replica exchange [Kitaoka09]
  - Run separate Markov chain(s) for specific features
  - Exchange the discovered paths between chains



29

Replica exchange has been known for a while in statistical MCMC field and was recently introduced in context of light transport.

The idea is relatively simple. Imagine that one mutation strategy can easily discover important features and another mutation strategy can easily explore such features, but has difficulties discovering them.

In this case, we can run two separate chains with these two strategies, and once the “discovering” chain has found a feature, it passes this feature to another chain to explore it. This way the “discovering” chain is responsible for finding important features, while these features can be efficiently explored by the second chain.

Normalization and Start-up Bias in MLT

Differences to MCMC

- We *do* have a good alternative sampler
  - Path tracer / bidirectional path tracer
  - Easy to compute normalization constant
- No start-up bias, start within the equilibrium
  - Start many chains stratified over path space
  - Scales well with massively parallel MLT

31

The key difference of MLT compared to the usual MCMC situation is that we already have methods that can generate an image well in most situations.

So, many regular MCMC problems like normalization constant estimation and start-up bias are solved easier.

For example, in order to compute the normalization constant, which is just an average flux received by the image plane, it is practically sufficient to sample a few hundred thousand paths with BDPT, which is a negligible cost comparing to the actual image rendering.

As for the start-up bias, we can seed Markov chains directly within the high-probability regions of the target distribution.

The usual practice is to collect many samples from BDPT and then seed Markov chain with one importance-sampled w.r.t. the path contribution. In case of many chains, their initial states can be also stratified with respect to the path contribution to have a good initial coverage of the path space.

And it also scales naturally well with tens of thousands of Markov chains for massively parallel devices like GPUs.

Mutation Strategies and Their Properties

Let’s try to understand how to mutate the paths.

Good Mutation Criteria

- Lightweight mutation: change a few vertices
- Low correlation of samples
  - Large steps in path space
- Good stratification over the image plane
  - Hard to control, usually done by re-seeding
- It's OK to have many specialized mutations

33

First of all, it is important to understand what criteria should an ideal mutation strategy fulfill.

So, the mutation should be as lightweight as possible, that is, it should try to introduce minimal changes to the path, triggering as few vertex updates as possible.

Then it should also produce a sequence of samples with low correlation, doing large steps in path space.

Also, specific to the image rendering process, the mutation should try to sample the image plane as uniform as possible. That is usually hard to control in the context of MCMC, thus the best practice is to reseed the chain with paths stratified over the image plane.

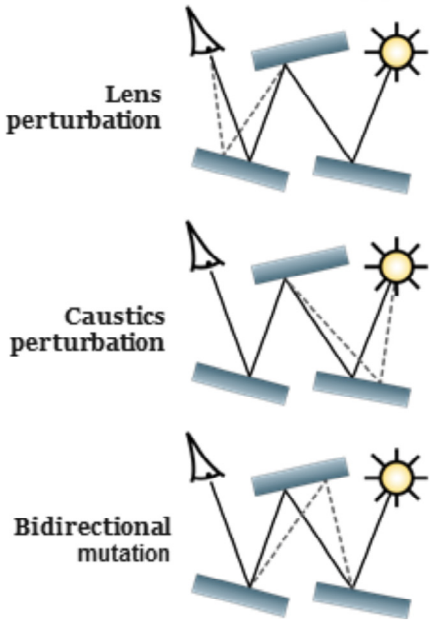
And finally, it is completely fine if the mutation can efficiently explore only some certain subset of path space, for example only caustics, leaving other features to other specialized mutations. In the end, that is one of the advantages of MLT.

Existing Mutation Strategies

Now I'll do an overview of the existing mutation strategies.

Veach Mutations

- Minimal changes to the path
  - Lens, caustics, multi-chain perturbations
- Large changes to the path
  - Bidirectional mutation
    - BDPT-like large step
  - Lens mutation
    - stratified seeding on the image plane



35

Eric Veach has first introduced MLT and proposed the original set of mutation strategies.

We can roughly classify them into two groups.

The first group perturbs the current path slightly, thus such mutations are called perturbations.

They are mostly crafted to efficiently explore the image plane and such difficult effects as caustics and chains of them.

Another group of mutations tries to do large changes to the path.

Namely, bidirectional mutation works similarly to BDPT, with the only difference that it completely resamples not the full path, but a randomly selected subpath of the current path.

Lens mutation reseeds the chain with a path from the pool of paths stratified over the image plane.

Kelemen Mutation

- Mutate a “random” vector that maps to a path
- Symmetric perturbation of “random” numbers
- Use the “random” vector for importance pdfs
  - Primary space: importance function domain
  - Assume the importance sampling is good

36

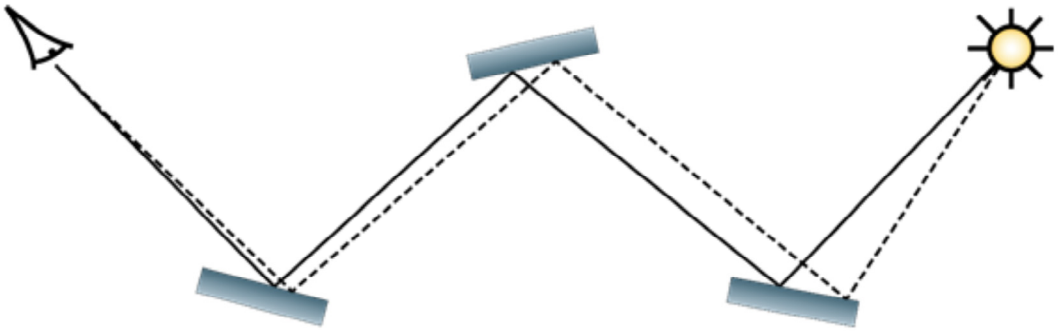
A popular mutation proposed by Kelemen is to mutate the paths in the so-called primary sample space, that is, the original space of the importance functions used for constructing the path in BDPT and PT.

Usually it is represented as a vector of random numbers in the unit hypercube, which is perturbed using some symmetric probability, like a multidimensional Gaussian distribution.

The major assumption is that the importance sampling functions already make the integrand flat enough that we can walk it using some uniform random walk in this primary space.

Kelemen Mutation, Part II

- Acceptance probability $a_{i \to j} = (f_j / p_j) / (f_i / p_i)$
  - Easy to compute: just take values from PT/BDPT
- Large step: pure PT / BDPT step
  - Generate primary sample (random vector) anew



37

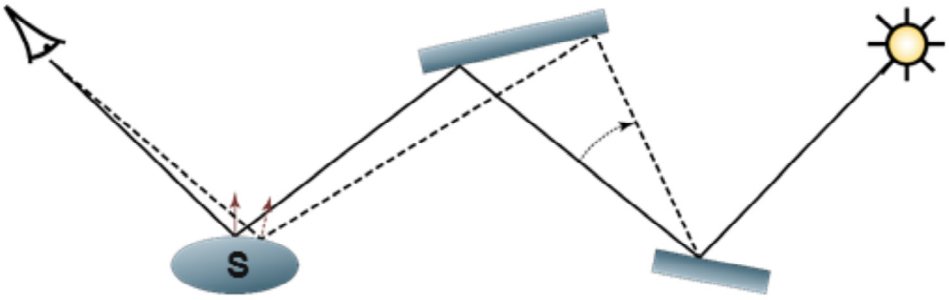
The good thing about this strategy is that a lot of terms in the ratio of measurement contribution to the transition probability (that is required to compute the acceptance probability) just cancel out.

Thus, since the perturbation probability is also symmetric, the final acceptance probability is computed as a ratio of the simple path throughputs computed by PT or BDPT. *[This makes it very simple to implement such a mutation strategy: just take an existing PT or BDPT, replace the random number generator by a replayeable sampler with symmetric perturbation and use the ratio of throughputs as an acceptance probability.]*

In order to discover new features quicker, we also need to do some large steps. For this reason a large step mutation was proposed. The idea here is also simple: just regenerate the complete random vector from scratch and try to construct a path. That is equivalent to just generating a random path with PT / BDPT.

Manifold Exploration Mutation

- Works in the local parameterization of current path
- Can connect through a specular chain
- Freezes integration dimensions
  - Tries to keep f constant by obeying constraints



38

Yet another recent mutation strategy that was introduced by Wenzel Jakob is called manifold exploration.

This is a supplementary mutation strategy, which is meant to replace the set of Veach's perturbations.

The idea here is that the path is perturbed from some vertex and then in order to construct the new subpath, first we construct a local on-surface parameterization of the current path and try to iteratively construct the new path in the space of this local tangent frame parameterization.

The idea comes from the differential geometry. This mutation tries to preserve hard constraints, like specular reflections, by utilizing the local knowledge about the geometry around the current path.

This way, manifold exploration can, for example, construct a connection from one point to another through a chain of specular or highly-glossy interactions.

In fact, this strategy tries to "lock in"/eliminate some of the integration dimensions (with specular/glossy interactions), while sampling others.

As a consequence, it tries to keep the measurement contribution function as constant as possible by locking or just slightly changing the terms of the measurement contribution function corresponding to the locked dimensions.

This strategy is similar to Gibbs sampling known from statistical MCMC.

Combinations

- Manifold exploration can be combined
  - With Veach mutation strategies in MLT
  - With energy redistribution path tracing
- Combine Kelemen's and Veach's mutations?
  - Possible, yet unexplored option

39

Some strategies and methods can be combined with each other.

The original set of mutations can be augmented by manifold exploration.

Also the same can be done in the context of ERPT.

Moreover, another yet unexplored option is to combine the original set of mutations with Kelemen mutation.

Population Monte Carlo Light Transport

Population methods can be used on top of MLT.

Population Monte Carlo Framework

- Use a *population* of Markov chains
  - Can operate on top of Metropolis-Hastings
- Rebalance the workload
  - Weakest chains are eliminated
  - Strongest chains are forked into multiple
- Use mixture of mutations, adapt to the data
  - Select optimal mutation on the fly

41

Population Monte Carlo framework stems from genetic algorithms.

Its idea is to keep a population of Markov chains (in our case it can be paths).

This method is a high-level superstructure, which can sit, for example, on top of an existing Metropolis-Hastings sampler.

Firstly, we keep only relevant samples in the population. This is done by the elimination and regeneration: the chains with a small contribution (under some threshold) are eliminated and being reseeded from the chains with very high contribution. It essentially dynamically rebalances the sampling efforts to the important places of the state space.

Moreover, we can adopt the mutation parameters (like a step size) based on the past samples and the state of the whole population on the fly.

Population Monte Carlo ERPT [Lai07]

- Spawn a population of chains with paths
  - Do elimination and reseeding based on path energy
- Use many mutations with different parameters
  - Reweight them on-the-fly based on the efficiency
  - Lens and caustics perturbations in the original paper
- We will show PMC with manifold exploration

42

Population Monte Carlo framework was applied to light transport by Lai et al. in the context of ERPT.

The process is similar to ERPT, yet it keeps the constant population of chains by reseeding the chains with low contribution from a pool of stratified paths.

The core idea is to use a set of existing mutation strategies, where each strategy can be present multiple times with different user-defined parameters, like step size. For example, the set might contain three caustics perturbation with different perturbation sizes and so on. The selection weights are then adjusted for these mutations on the fly based on the performance of each mutation. This process quickly emphasizes mutations with good performance, making the transition probability adapting to the data.

In the original paper, the authors propose to use caustics and lens perturbations.

However, in the second part of the course, we will demonstrate this method with multiple manifold exploration mutations with different perturbation parameters.

Thank You for Your attention.

Part one questions?

Anton S. Kaplanyan:

Comparison of Advanced Light Transport Methods

Recent Advances in Light Transport Simulation: Theory & Practice

Comparison of Advanced Light Transport Methods

Equal Time Comparison

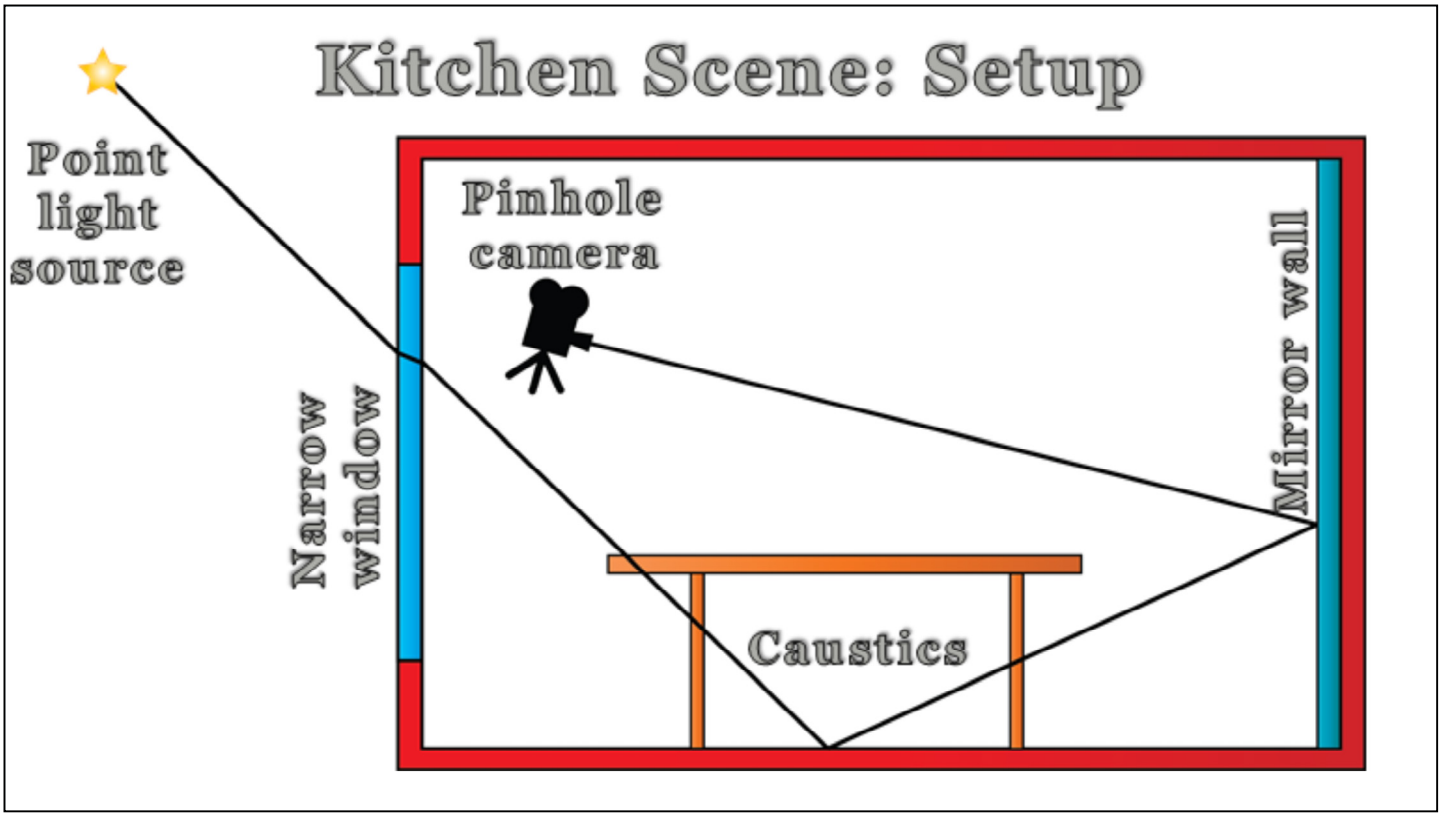
- Rendering time: 1 hour on GeForce 580
  - Some are rendered with Mitsuba [Jakob10]
- *Kitchen* – a hard-to-render scene
  - Light occluded by glass, caustics
  - Many mirrors, reflected caustics
  - Glossy materials
  - Curved glass with many specular interactions

In this part of the course I would like to provide a qualitative comparison of different existing methods.

All renderings are done in equal time (1 hour) on the Nvidia GeForce 580 GTX GPU.

Some images are rendered on CPU with Mitsuba renderer with a performance equivalent of 1 hour on GPU.

I have set up a scene with multiple difficult lighting features in order to evaluate state-of-the-art light transport methods.

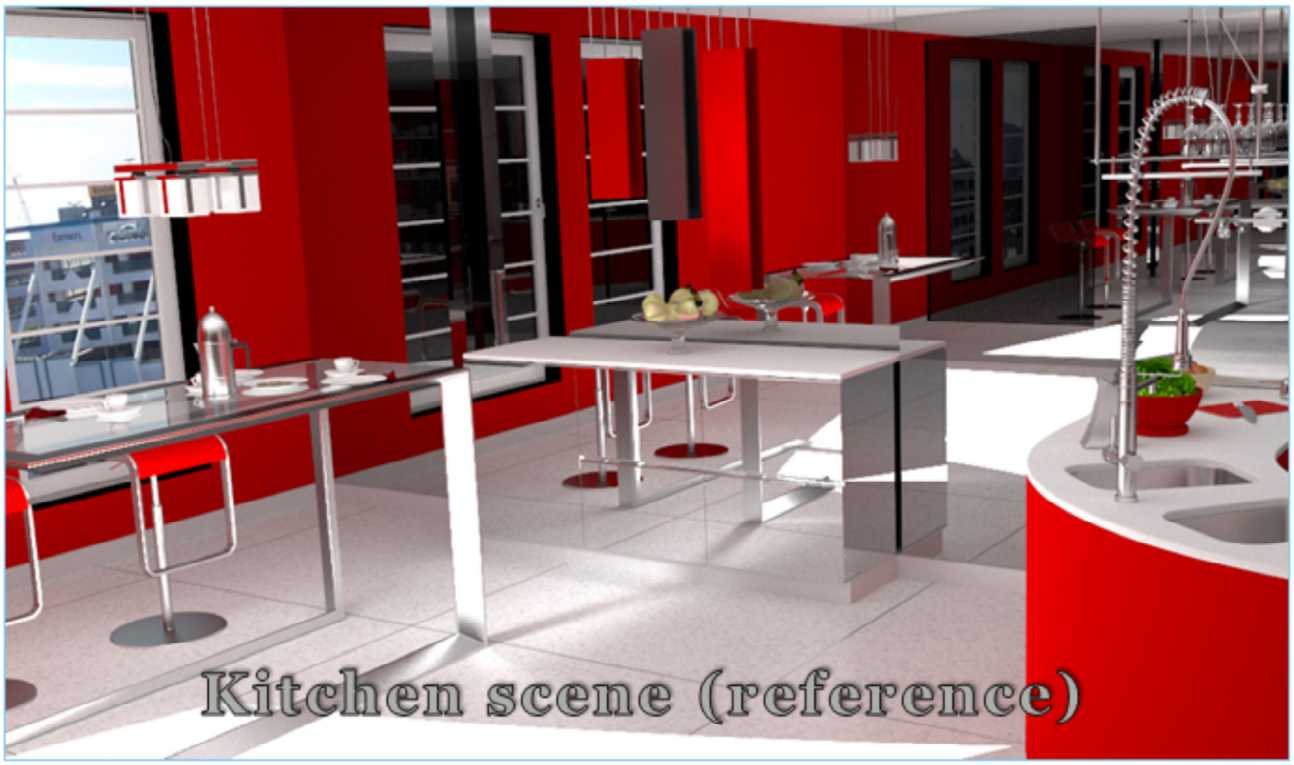


Here is the scene configuration.

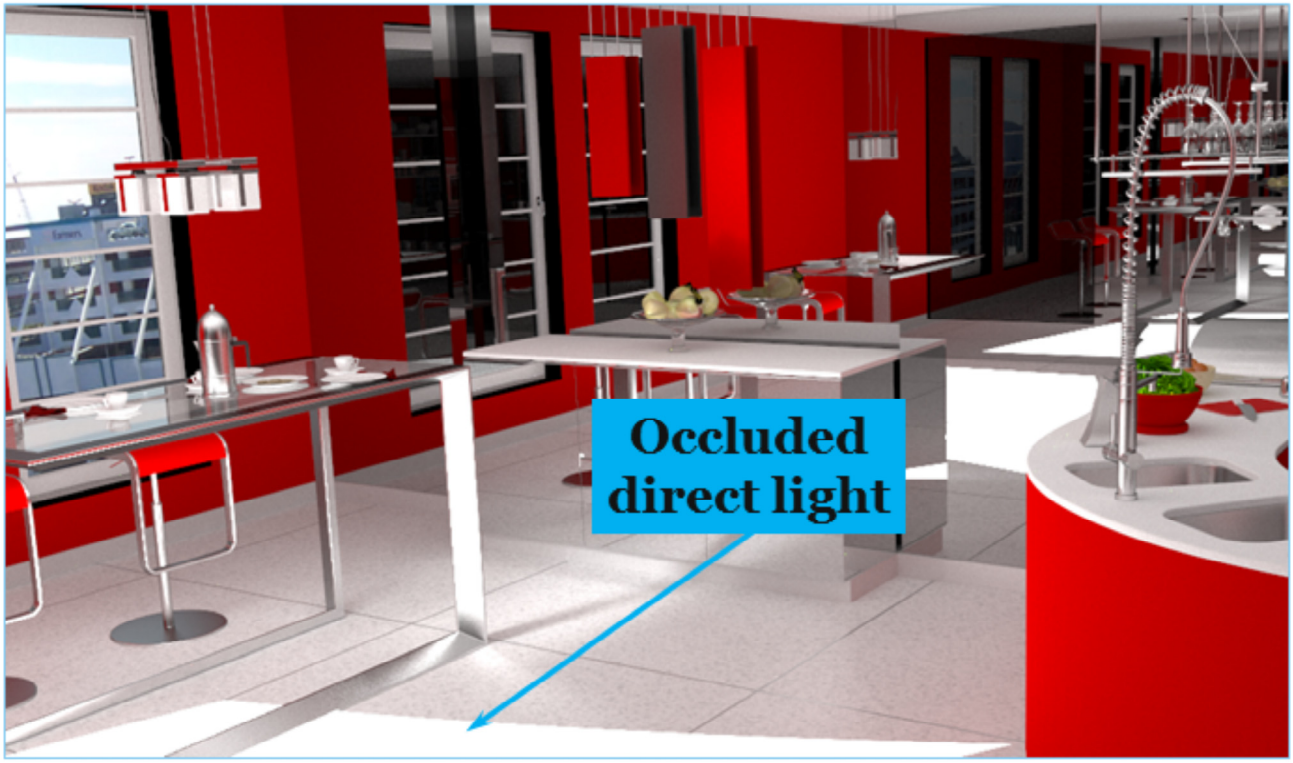
Point light source is placed outside, illuminating the room through a glass window, causing a caustic on the floor.

The scene has also a mirror wall causing a lot of reflections.

This is the “Kitchen” scene rendered with diffuse materials to show the geometric topology.



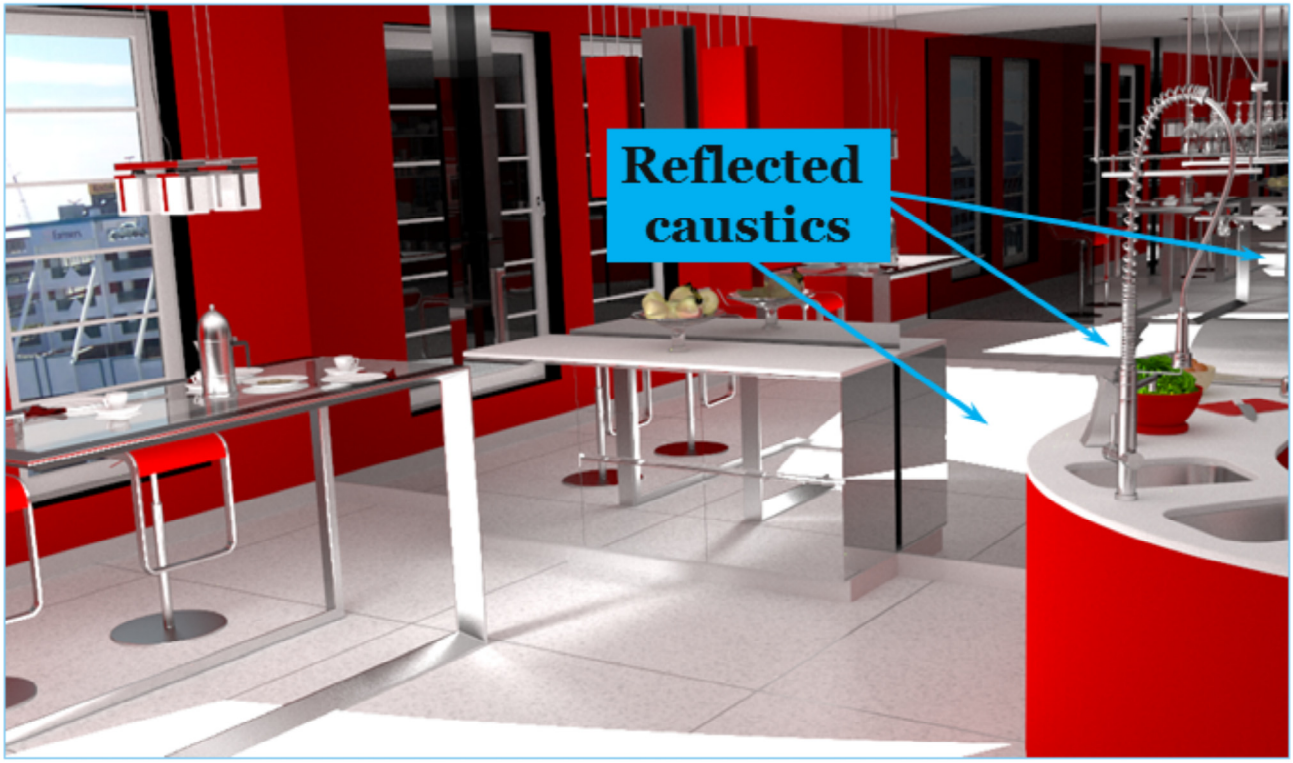
This is the reference image of the “Kitchen” scene with final materials. Rendered with Vertex Connection and Merging technique for 24 hours.



Here are some difficult features.

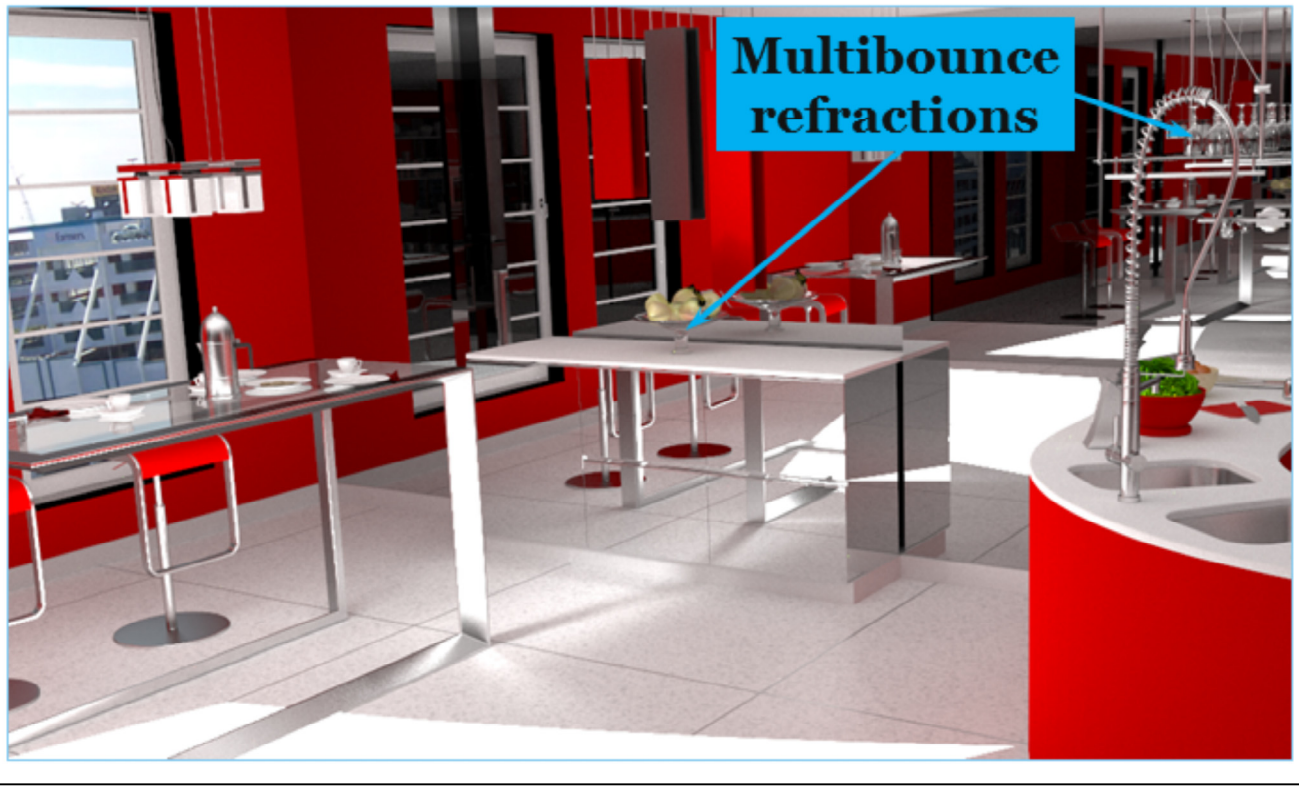
First of all, the direct lighting is highly occluded, since the window visible from the light source takes a small part of the emissive sphere of directions.

This situation is similar to the classical complex light transport scenarios, like an ajar door [Veach97].

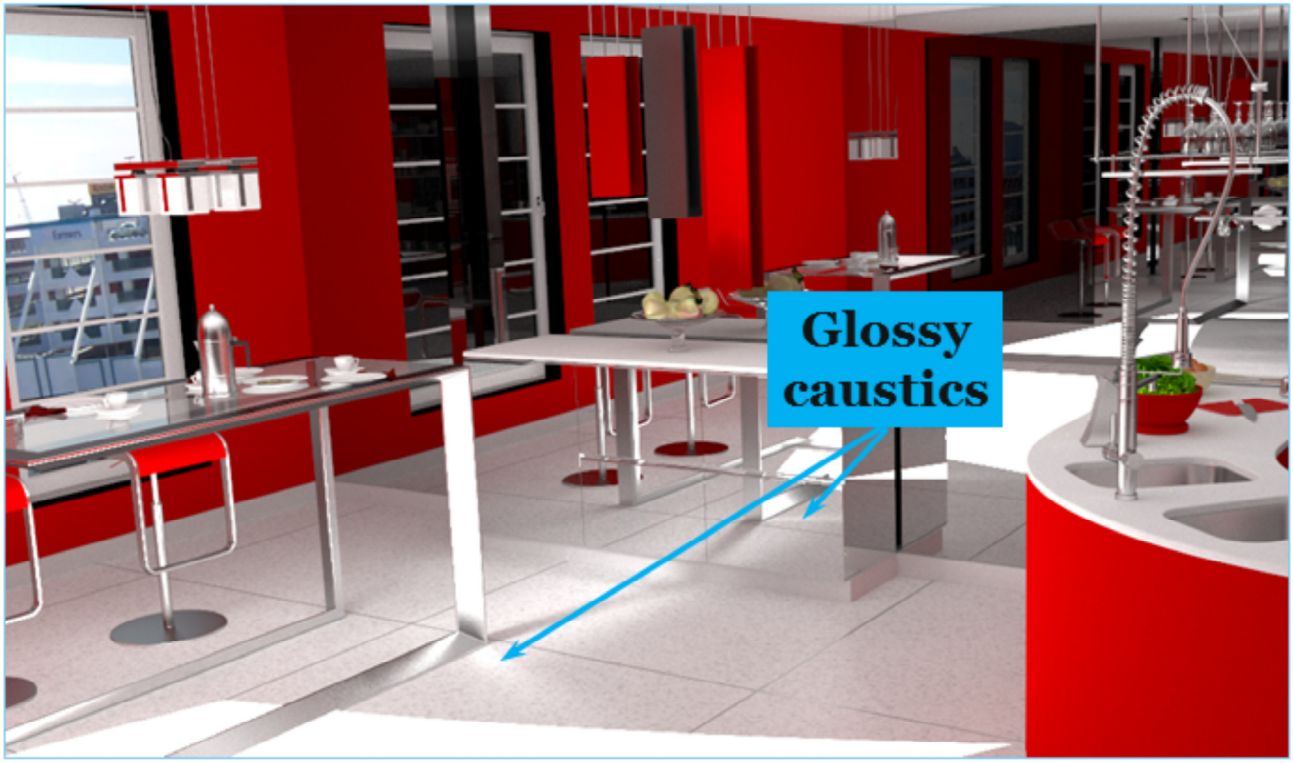


Then, the reflected caustics caused by the point light source are impossible to find with all unbiased methods.

Also caustics, experiencing multiple bounces (window->table->floor) pose a difficult sampling problem for some Monte Carlo methods.



Refractions with multiple bounces (specular chains) has been also known to be a hard sampling problem [Jakob12].



Finally, caustics from highly glossy objects are hard to find in path space (they have a small “mass”), thus might be difficult to discover even by advanced MCMC methods.

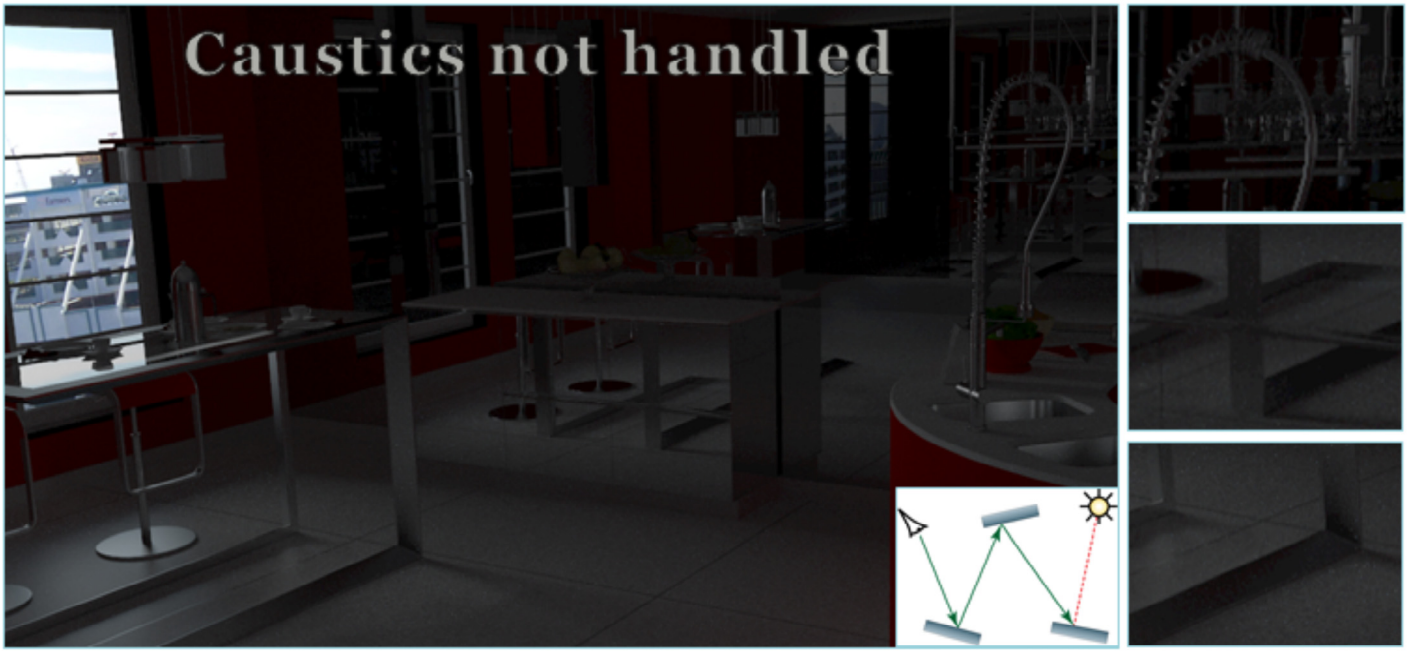
Ordinary
Monte Carlo Methods

Ordinary Monte Carlo Methods

- Path tracing [Kajiya86]
- Light tracing [Arvo86, Dutre93]
- Bidirectional path tracing [LaFortune93, Veach94]
- Vertex connection and merging [Hachisuka12, Georgiev12]

We start with ordinary Monte Carlo methods. They can usually handle majority of scenes.
We will show the following Monte Carlo methods.

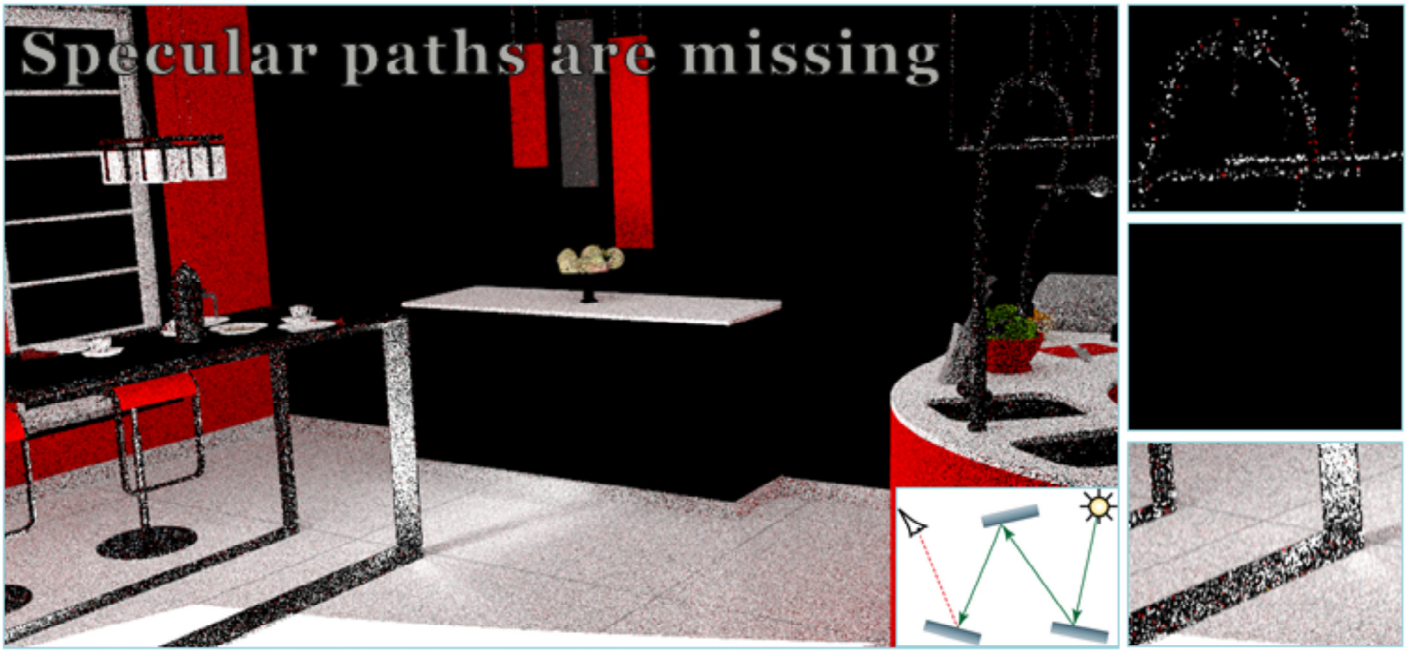
Path Tracing



The first method: path tracing. It can handle only the small fraction of light coming from the environment.

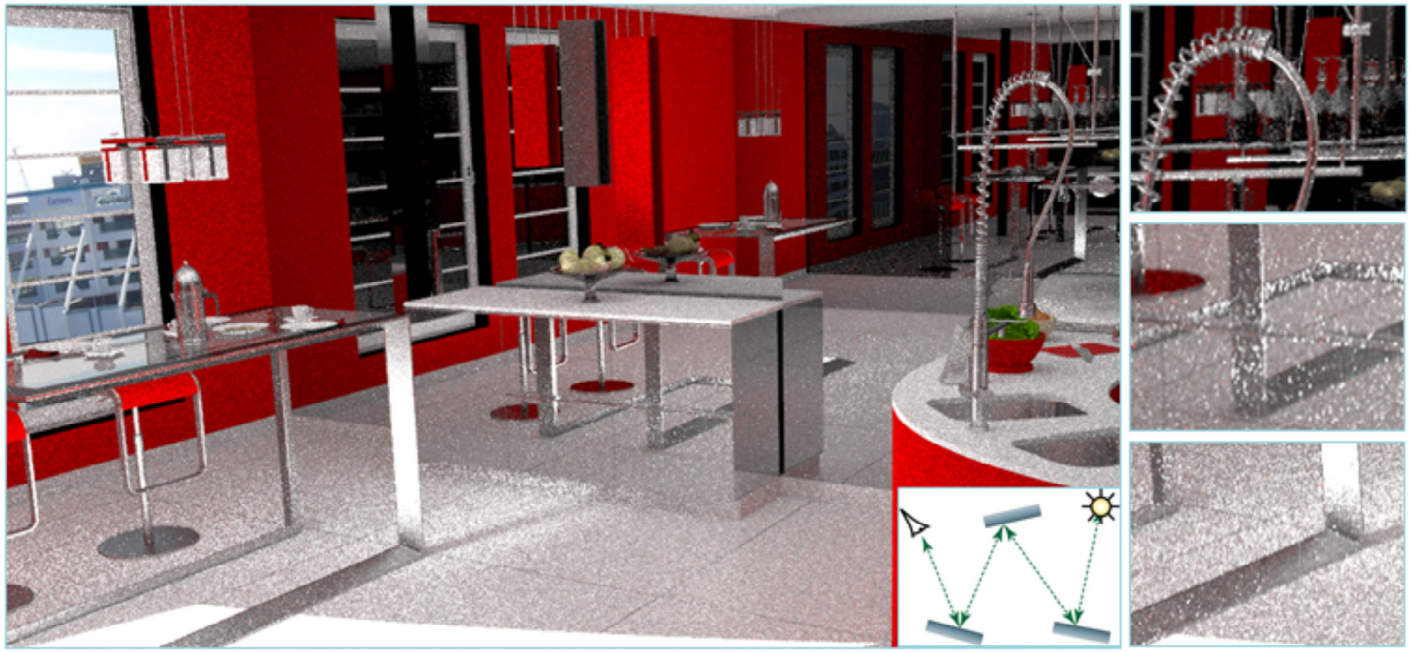
This is due to the fact that al illumination coming from the point light source first comes through the glass. This makes it impossible to directly connect to the light source.

Light Tracing



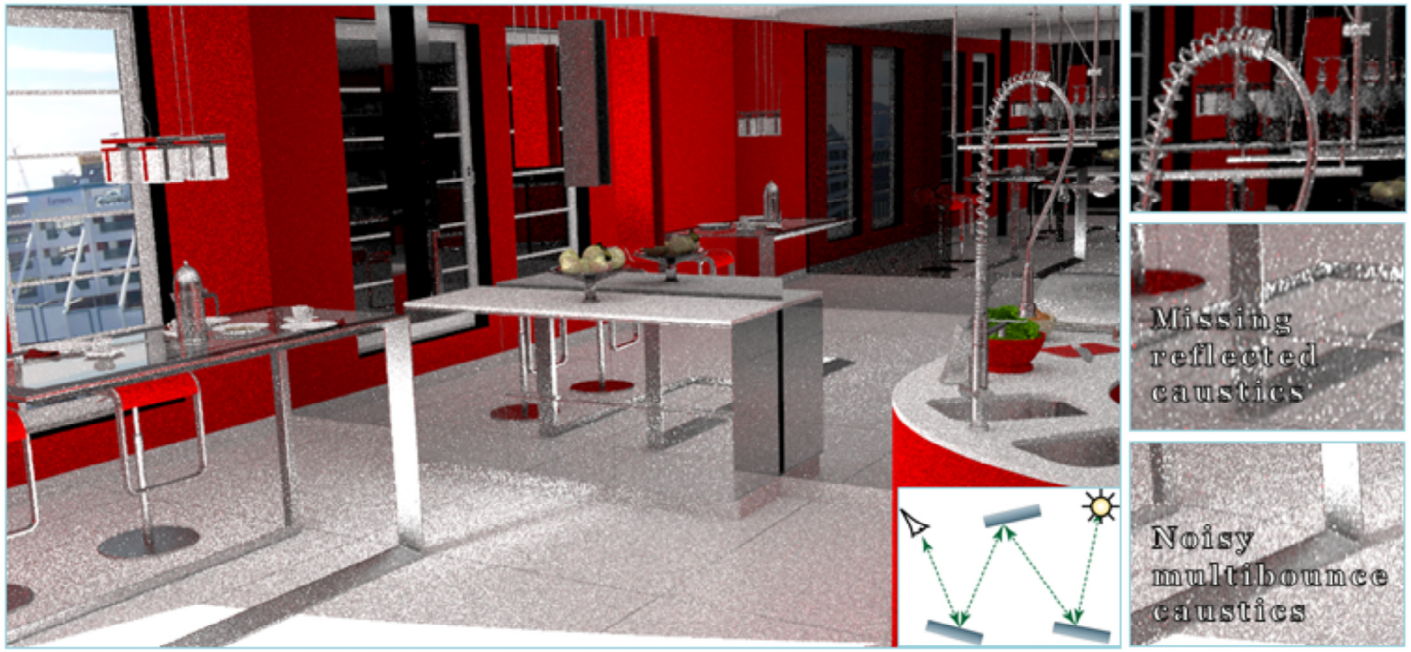
In opposite, the light tracing technique handles caustics coming through the window quite well, yet failing all specular reflections and refractions, because they cannot be constructed using connection to camera through specular surfaces.

Bidirectional Path Tracing



Bidirectional path tracing handles almost all possible paths, however some difficult paths are noisy or missing.

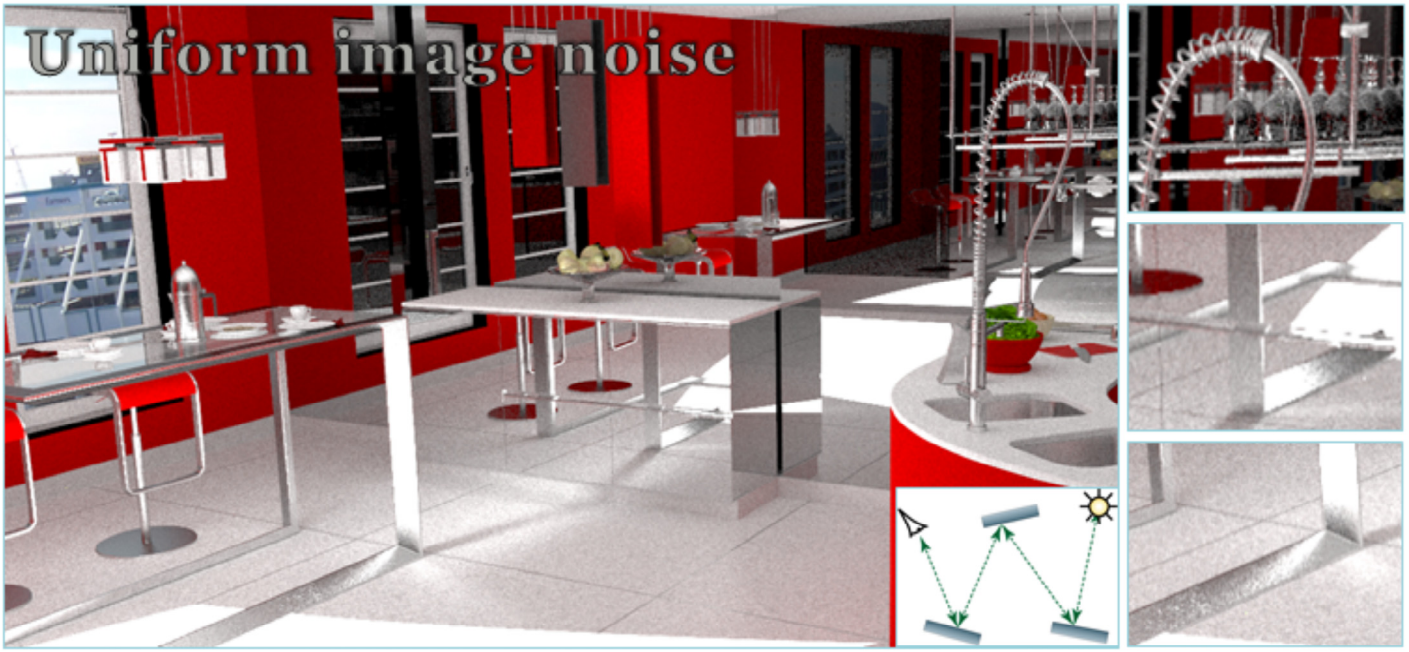
Bidirectional Path Tracing



BDPT has difficulties sampling caustics that go through multiple bounces. The reason is that the mass of such caustics in the path space is small, making it difficult to find them stochastically.

And the reflected caustics from point light cannot be sampled due to missing connection opportunity (that is, a path edge without singular materials).

Vertex Connection and Merging



Recent vertex connection and merging technique can handle caustics much better.

Moreover it can even sample reflected caustics. However, the method has uniform image noise due to the highly occluded light source (which is being sampled stochastically).

Why Monte Carlo methods work well in most of the cases?

You have probably seen the recent femtosecond photography of real world light propagation.

(0) I decided to do the same, but with the light transport simulation.

(1) This video demonstrates that the light transport turns into a diffusion process very quickly.

(2) The same happens when I propagate the virtual wavefront of “importance” from the camera. Note that it is not a physically valid value, yet it shows how the adjoint quantity is propagated, e.g., in bidirectional path tracing.

Both quantities spread around the scene very quickly, making it easy to establish the connection for unidirectional and bidirectional methods.

(3) However let’s take a look at the glossy example: the diffusion is not that prominent anymore, making such configuration difficult to sample by ordinary Monte Carlo methods.

Markov Chain Monte Carlo Methods

Markov Chain Monte Carlo Methods

- Metropolis light transport [Veach97]
- Different mutations
  - Primary sample space mutation [Kelemen02]
  - Path space mutations [Veach97]
  - + Manifold exploration mutation [Jakob12]
- Energy redistribution path tracing [Cline05]

However as we will see, some difficult illumination features and visibility situations can be handled more efficiently with Markov chain Monte Carlo methods.

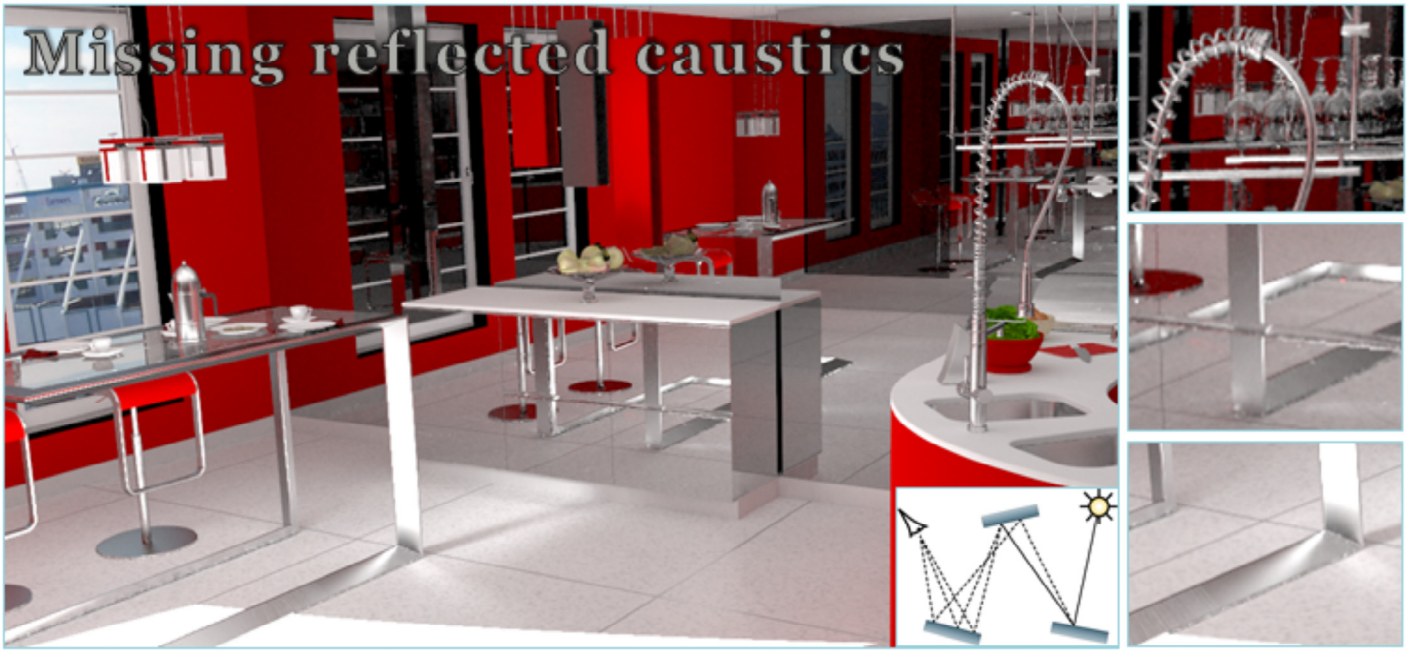
I'll present the following techniques

Markov Chain Monte Carlo Methods, part II

- PPM with MCMC photon tracing [Hachisuka11]
- Population Monte Carlo energy redistribution [Lai07]

I will also show some recent and advanced MCMC methods.

Metropolis Light Transport



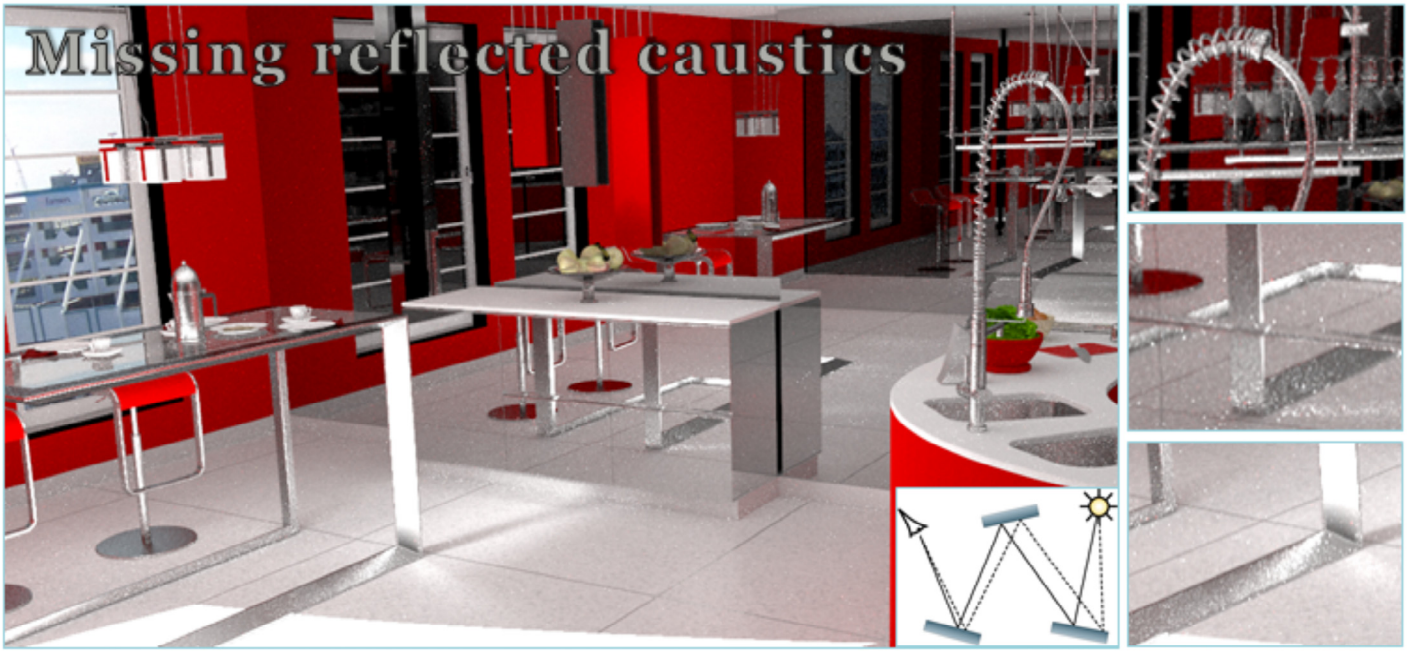
The first MCMC method (also chronologically) is the original Metropolis light transport algorithm [Veach97] with the original set of mutations.

The image is handled well, without noticeable noise. Yet reflected caustics are missing, as expected from an unbiased method.

Note some slight under-sampling at geometric boundaries (for example, the farther horizontal table leg on the floor has some bright splotches).

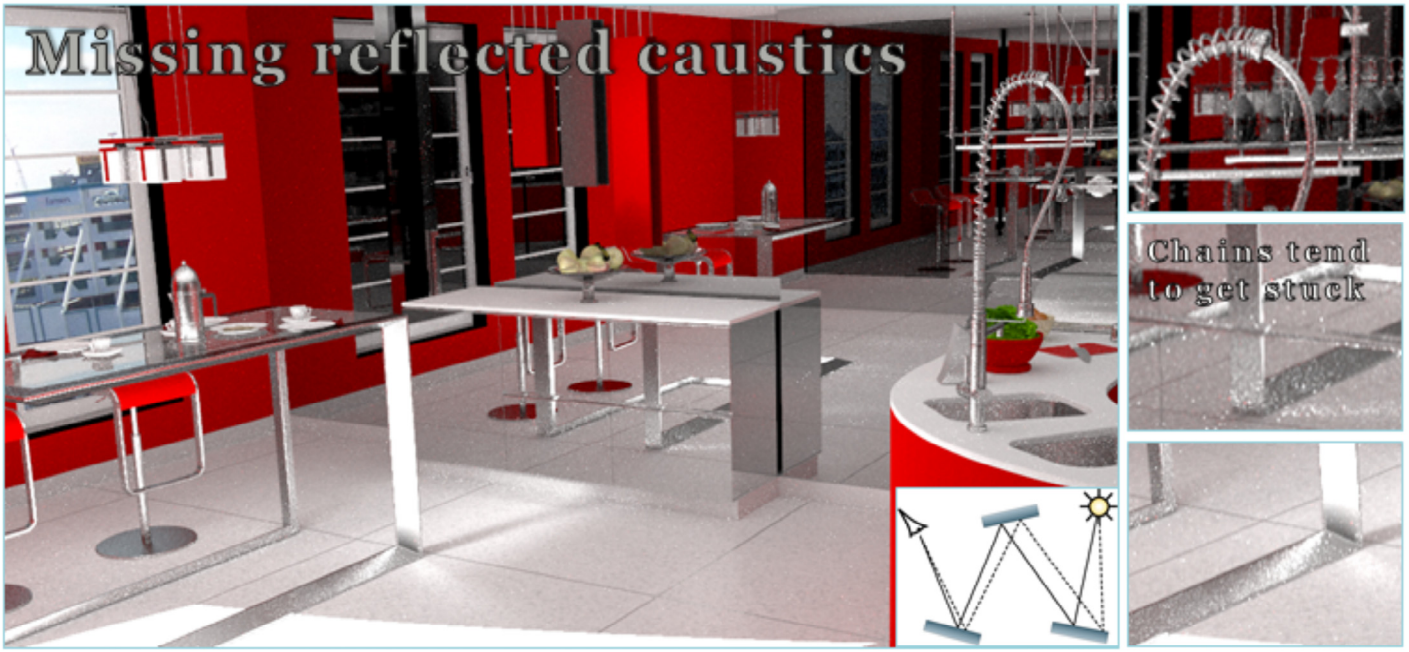
This is due to high rejection rate of perturbations closer to the change of surface.

MLT in Primary Sample Space



Here is Metropolis light transport with mutations in primary sample space [Kelemen02].

MLT in Primary Sample Space

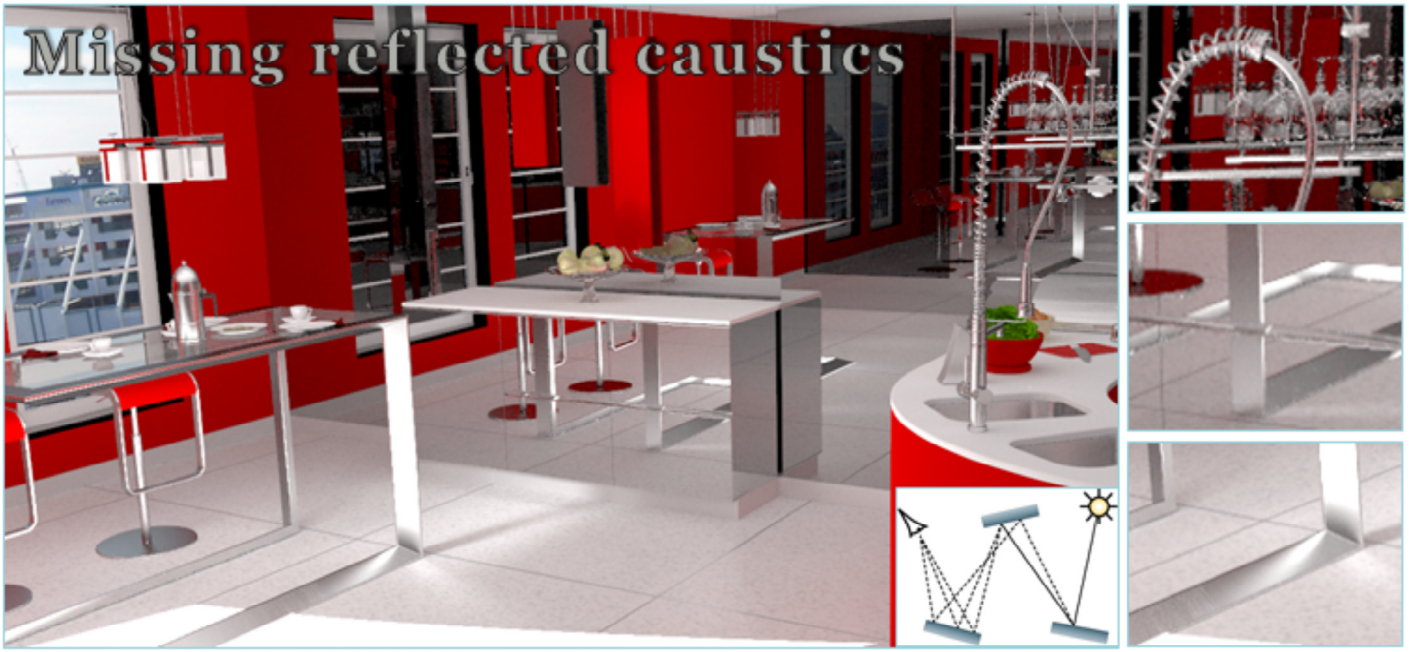


Note that some chains get stuck in complex caustics paths.

This happens because of many rejections since the whole path is usually regenerated, thus constantly jumping from the valid form.

Also the small step size is specified globally for random numbers and is not adapted for long or highly-occluded paths.

MLT with Manifold Exploration

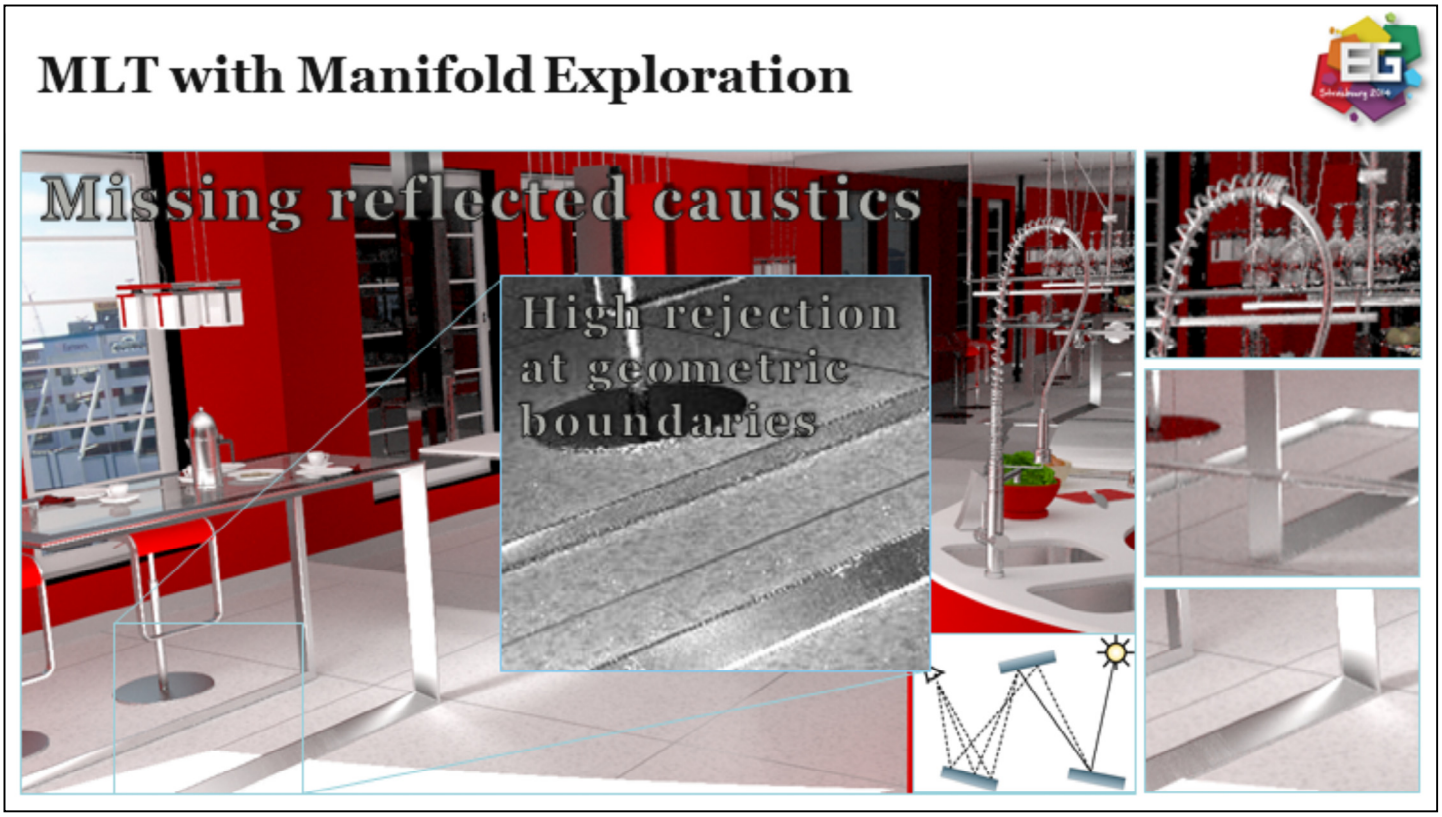


Here is the original MLT algorithm with the recently introduced manifold exploration mutation [Jakob12].

Note how much better the multibounce refractions are handled.

Also note that the reflection of the glossy caustics from the table leg are handled much better than with the original MLT.

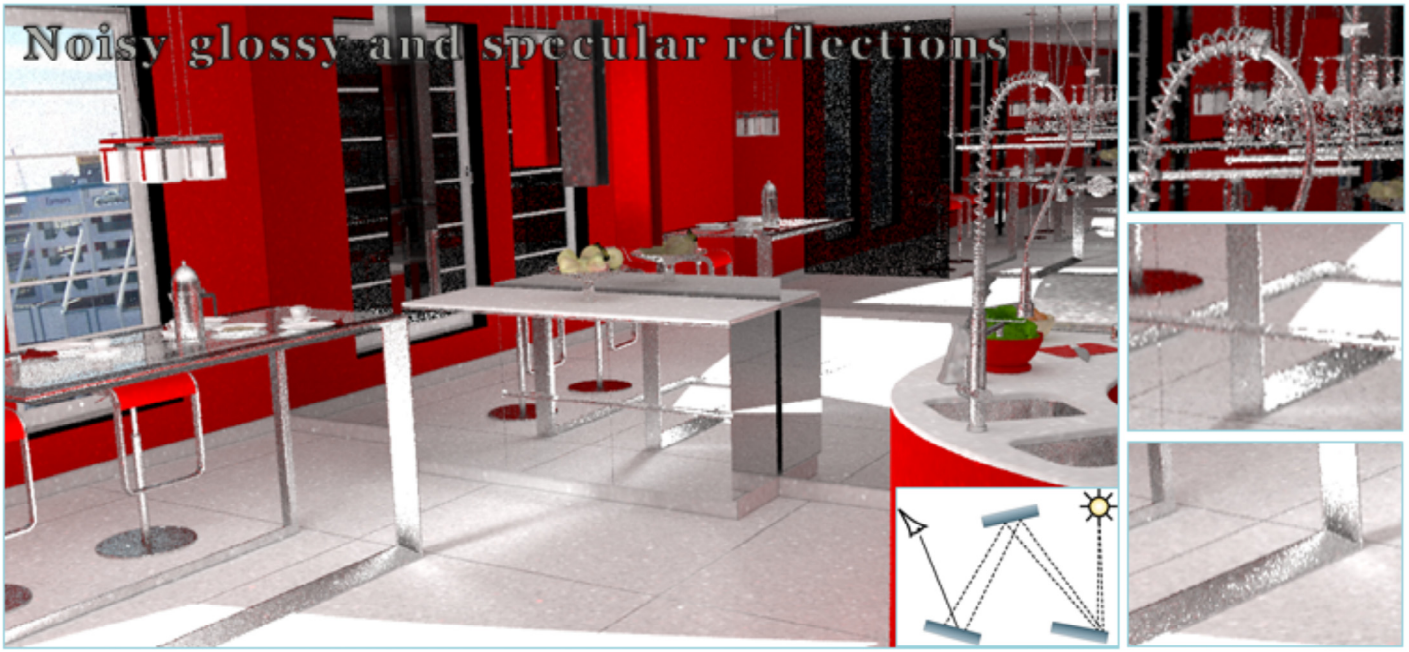
This is due to the fact that this mutation can skip through specular interactions, such as mirror reflections and refractions.



Also note that some subtle under-sampling is present along geometric edges, because of high rejection rate due to the invalidation of local path parameterization when jumping from one surface to another.

This issue can be addressed with the new gradient-domain Metropolis light transport method [Lehtinen13], which is presented by Jaakko Lehtinen on Wednesday.

Markov Chain PPM



Here is another recent method that is presented by Toshiya Hachisuka on Wednesday.

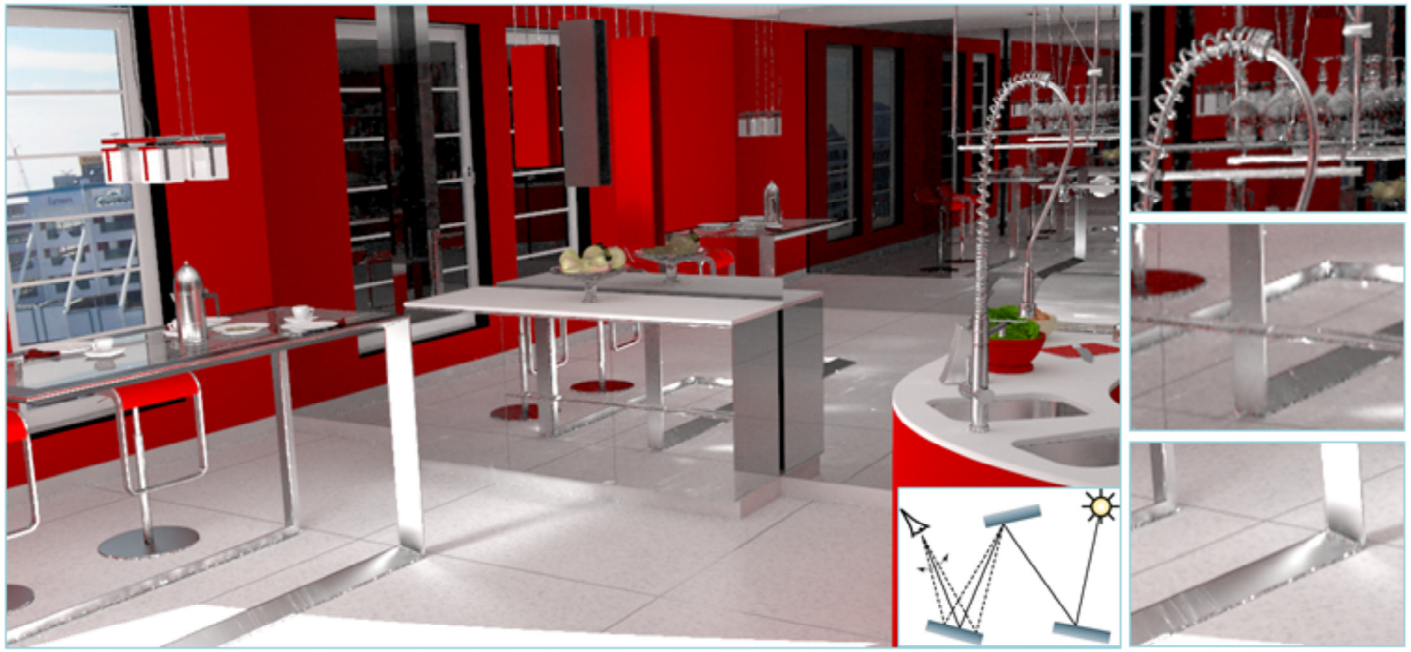
It is an extension to stochastic progressive photon mapping [Hachisuka08] with Markov chain photon shooting [Hachisuka11].

The original Kelemen mutation strategy with recommended parameters was used to mutate photon paths.

In order to keep the target distribution constant, the camera subpaths should be updated rarely.

This leads to under-sampling at glossy reflective and refractive surfaces.

Energy Redistribution Path Tracing (ERPT)



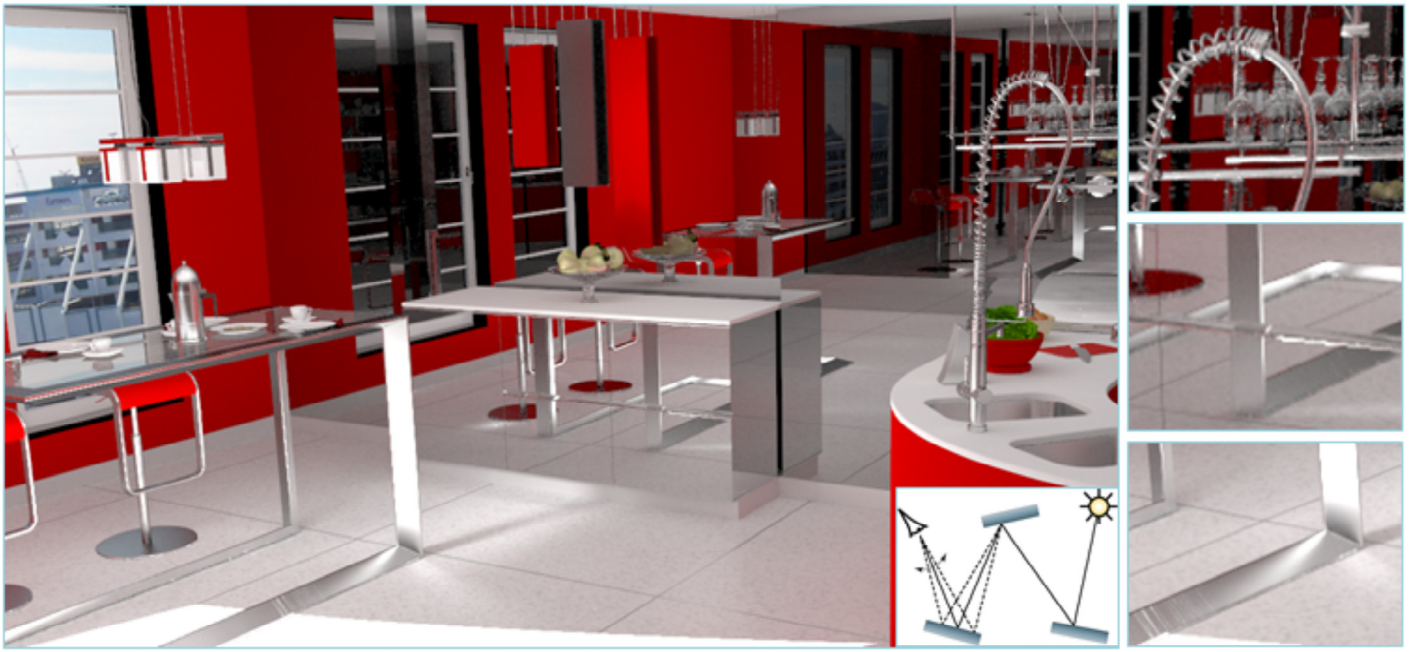
Here is the original energy redistribution path tracing method [Cline05] with the original set of mutations [Veach97], seeded with BDPT.

We used one chain per pixel (on average) and 100 mutations per chain (being the values recommended by the author).

As we can see, ERPT has difficulties efficiently redistributing energy from glossy paths, especially if they also experience one or more perfect reflections/refractions.

Moreover, some difficult parts of the image become undersampled in the same rendering time comparing to MLT. This happens because Markov chains are re-seeded with BDPT much more often, yet BDPT doesn't provide a good quality of seeding, as we have seen from the BDPT rendering (due to the highly occluded light source and complex light transport).

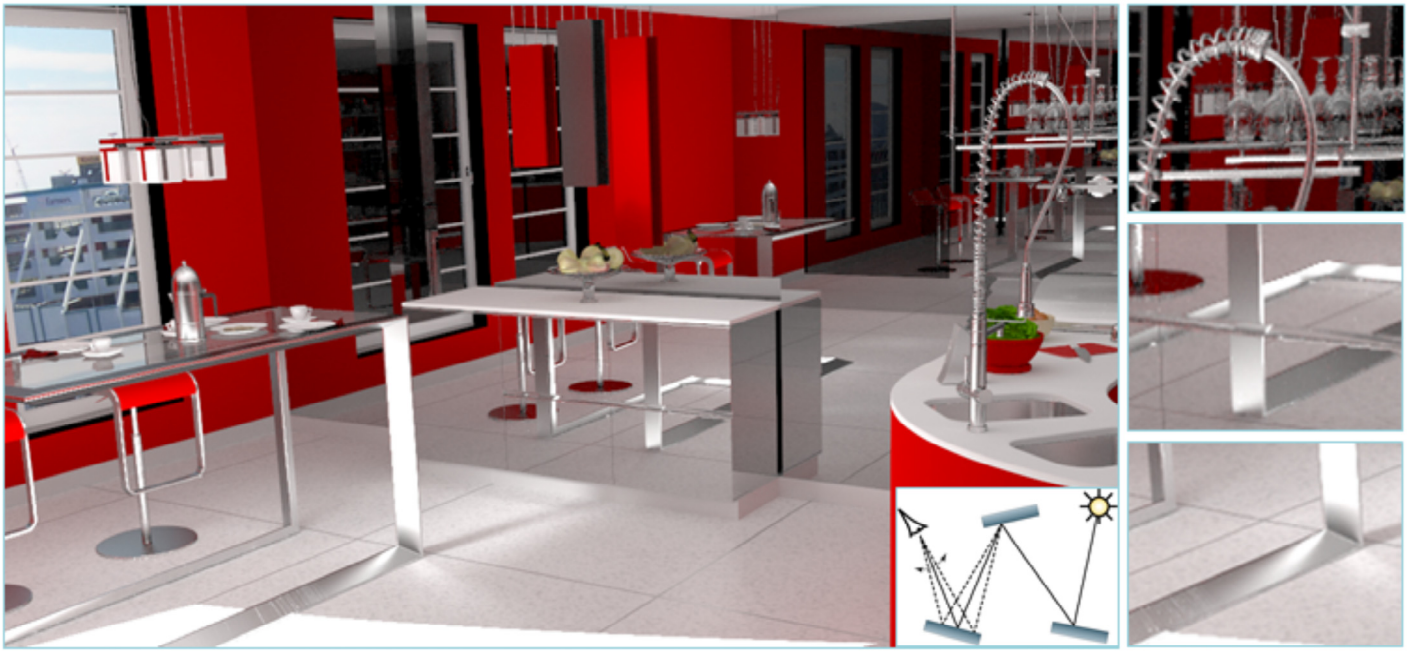
ERPT + Manifold Exploration



Another combination is ERPT with manifold exploration.

As we can see, manifold exploration allows to redistribute the energy of complex specular and glossy paths much better, leaving just a few splotches on the image.

ERPT, Manifold Exploration Only

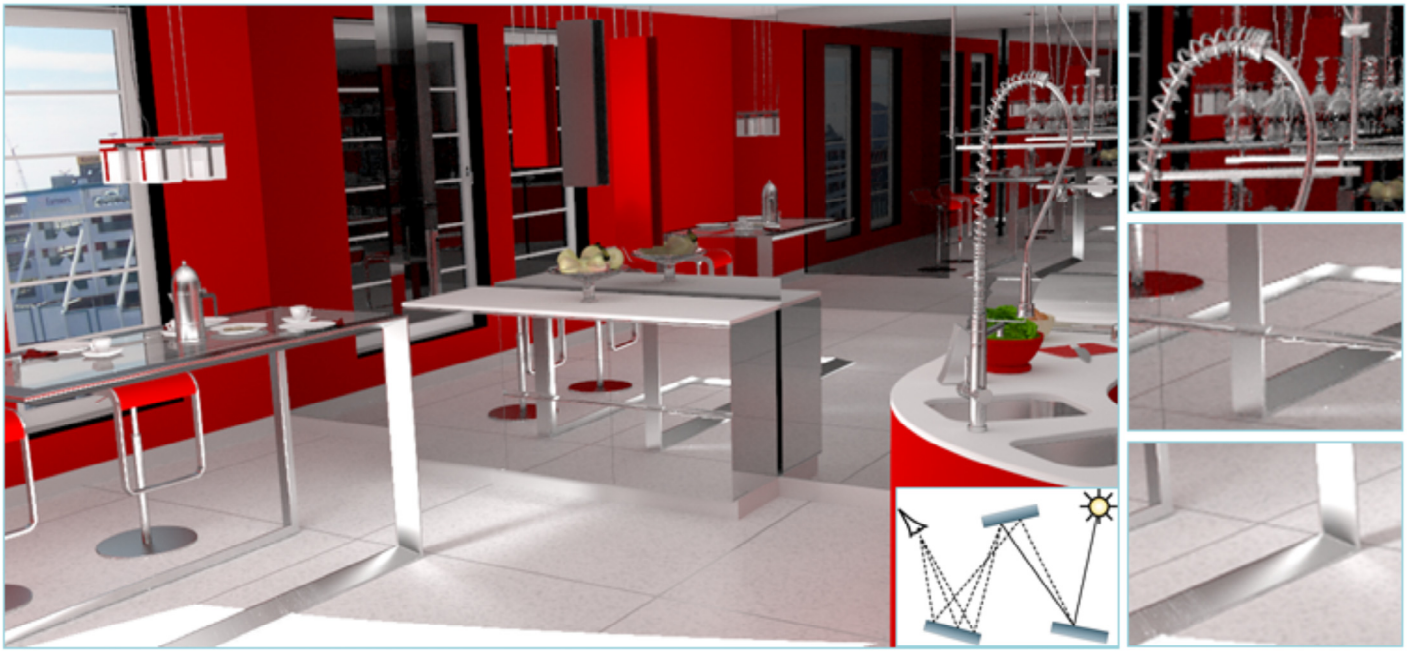


Another interesting experiment is to provide only manifold exploration as an available redistribution mutation, while relying on BDPT for the stratification and search of new features.

As we can see it also works relatively well, leading to the conclusion that this single mutation strategy is enough for achieving good results with ERPT.

However the equilibrium is not achieved with any of ERPT variations in a given time budget.

Population Monte Carlo ER + ME



In this rendering we use the Population Monte Carlo framework applied on top of ERPT [Lai et al. 2007] with multiple manifold exploration mutations.

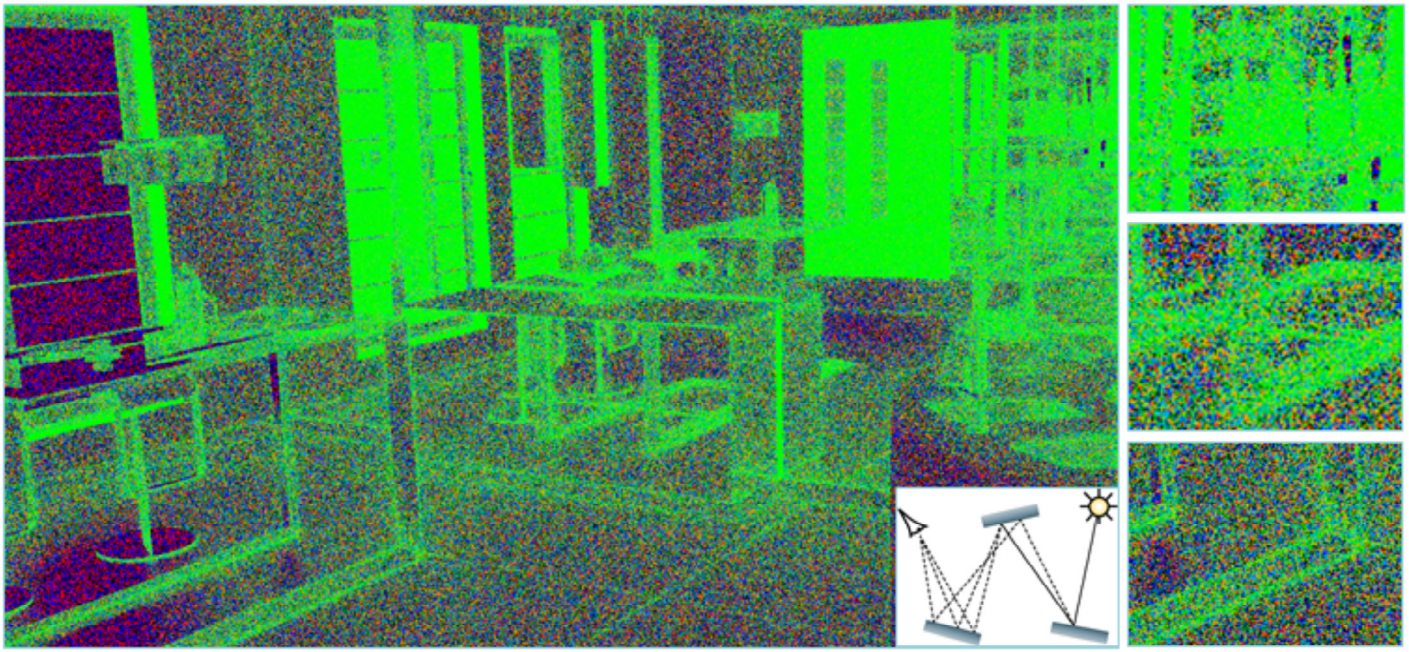
We use a population of 1024 chains and various values from 20 to 200 for manifold exploration mutation step parameter lambda.

Note that the geometric edges are sampled much better due to adaptive selection of the optimal mutation step size.

For example, edging of the glossy table is sampled much better, which is even more noticeable in the reflection of the table legs.

This is due to the fact that the population adapts and selects the mutation strategy with smaller perturbation step.

Population Monte Carlo ER + ME



This is a false-colored overlay of weights for different selected strategies.

Greener color corresponds to higher weight of a mutation with smaller step size.

These weights are reset every time the chain is re-seeded from BDPT, thus the weights are that noisy.

Note how the method prefers mutations with smaller step along geometric discontinuities.

Conclusion

- MCMC is more robust to complex lighting
  - \+ Better survives the *curse of dimensionality*
  - \+ Rule of thumb: for ≥~7-10 bounces → MCMC
  - \+ Helps with highly occluded and glossy scenes
  - – Non-uniform convergence: bad for previews
- PMC adapts mutation parameters

I'd like to mention that modern MCMC method survive the course of dimensionality better than ordinary MC methods (like PT and BDPT).

That means that whenever the majority of light transport in the image is high-dimensional (that is, roughly more than 7-10 bounces), then MCMC behaves much better in such cases.

It also helps exploring narrow islands and peaks in path space caused by highly occluded lighting configuration and/or highly glossy materials.

However, MCMC is famous for its non-uniform convergence, that is, it might take a while before Markov chain finds some important feature on the image, like some distant reflected caustic.

In such case, this feature just suddenly appears on the image rendered with progressive rendering.

This is unwanted in quick preview scenarios, when artists want to briefly assess the lighting.

Additionally, population Monte Carlo framework provides an easy and unbiased adoption of parameters for mutation strategies and also provides slightly better redistribution.

Mutation parameters' adaption can play a significant role in scenarios with high rejection rate, for example, caused by high-frequency geometry of tree leaves when rendering forest-like scene.

Thank You for Your attention.

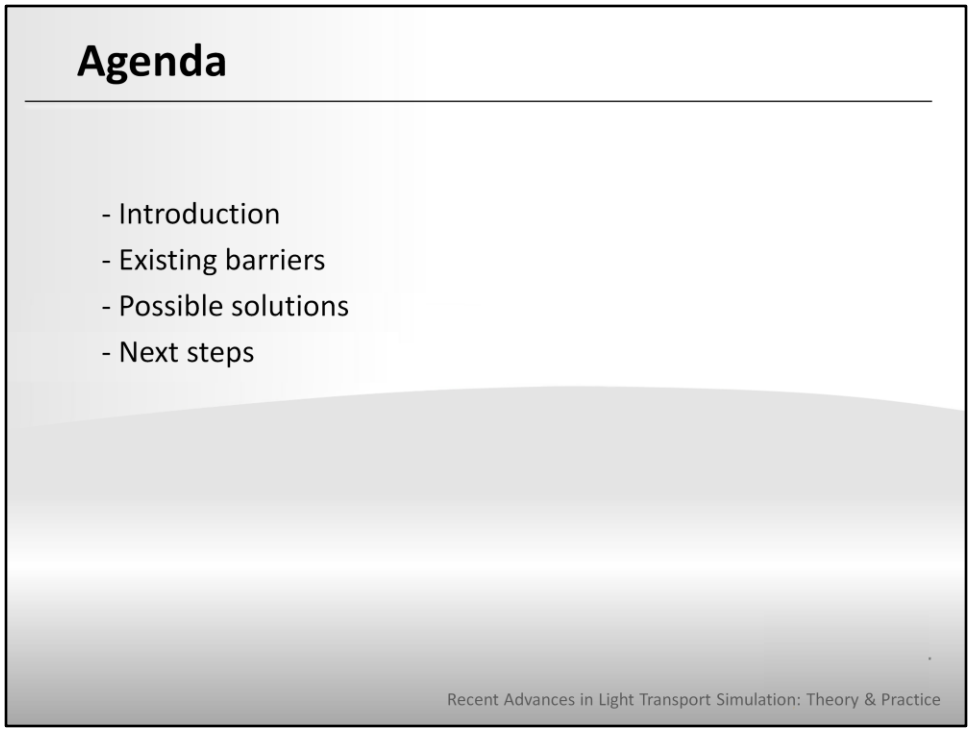
Part two questions?

Juan Cañada:

Advanced Light Transport in the VFX/Archiviz industry

Advanced Light Transport in the VFX/Archiviz industry

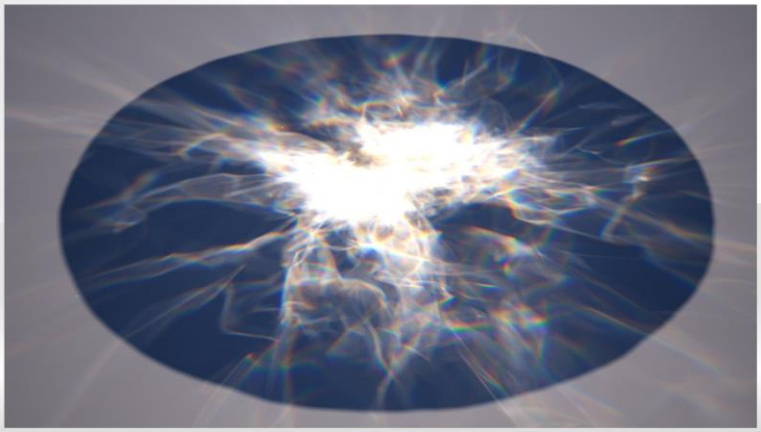
Juan Cañada – Head of Maxwell Render
Next Limit Technologies



Jaroslav, Iliyan and Anton have talked extensively about the state-of-the art light transport algorithms. I will talk about how the industry applies this knowhow. And specifically I will talk about which parts are not used yet, and why. I will focus on existing barriers, possible solutions and next steps that will probably be taken.

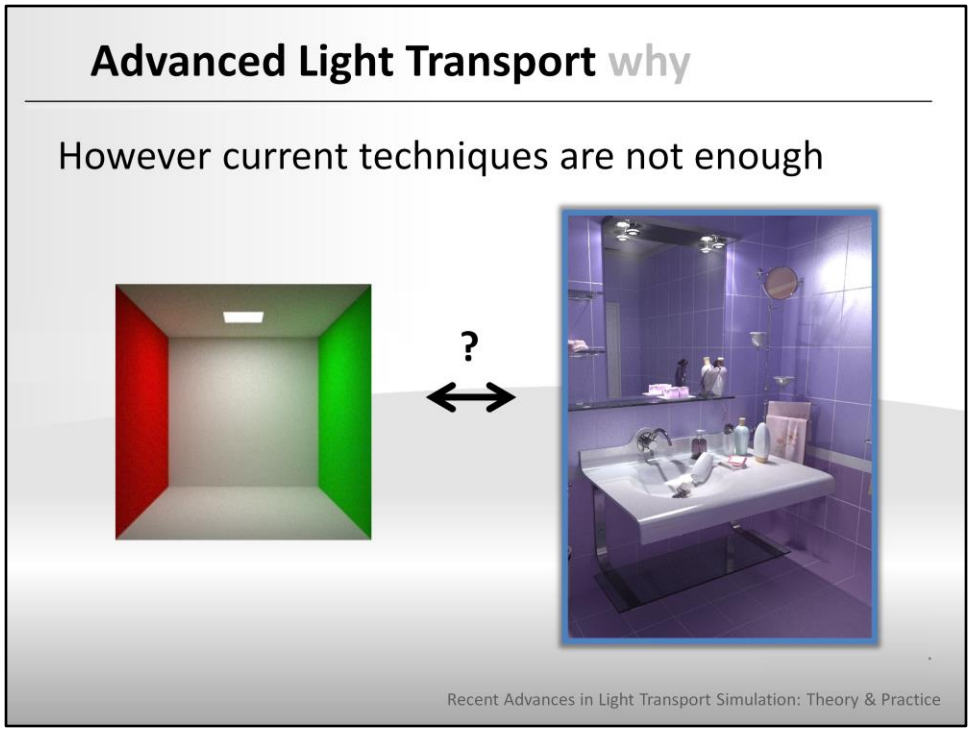
Advanced Light Transport why

Research on light transport allows us to simulate complex scenarios

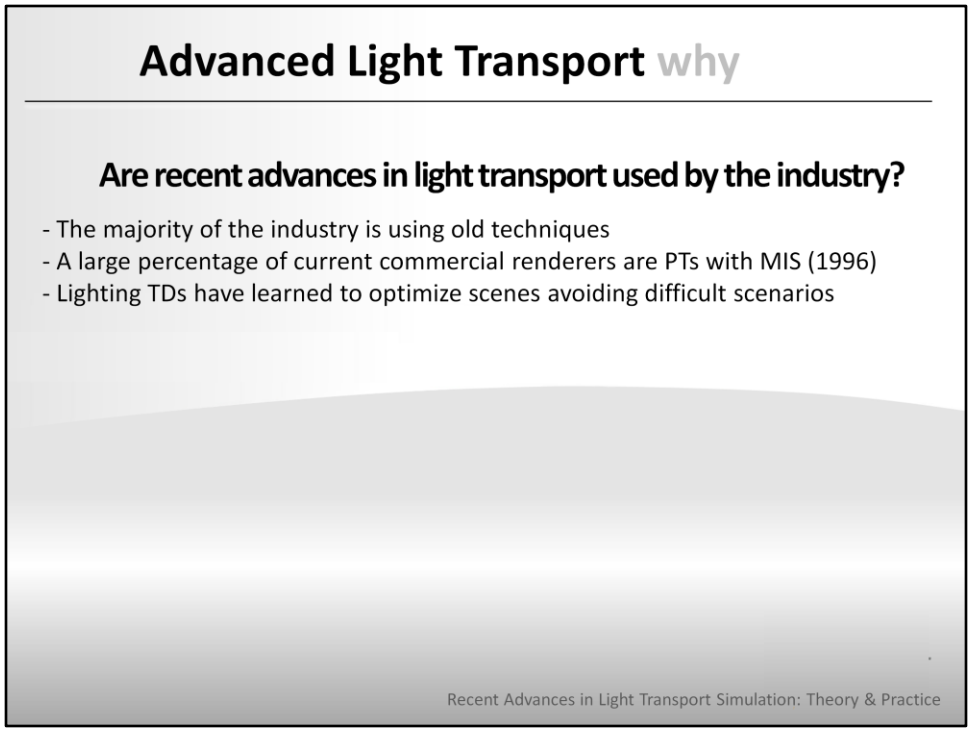


Recent Advances in Light Transport Simulation: Theory & Practice

Nowadays techniques allow us to render very complex scenes. This animation (mw_dispersion_caustics.mp4) shows refractive caustics with dispersion, and it was rendered with no difficulties on old machines.



However, despite the fact that current techniques are very good, they are not enough for rendering every kind of scenario. Many scenes require a lot of time to render, or the use of counterintuitive parameters to change the convergence of the rendering algorithm (i.e. bounces, light samples, etc). A lot of work still needs to be done in the search for a good balance between simple scenes and corner cases; meanwhile it is better that a render engine provides robustness in the majority of situations, rather than good performance sporadically.

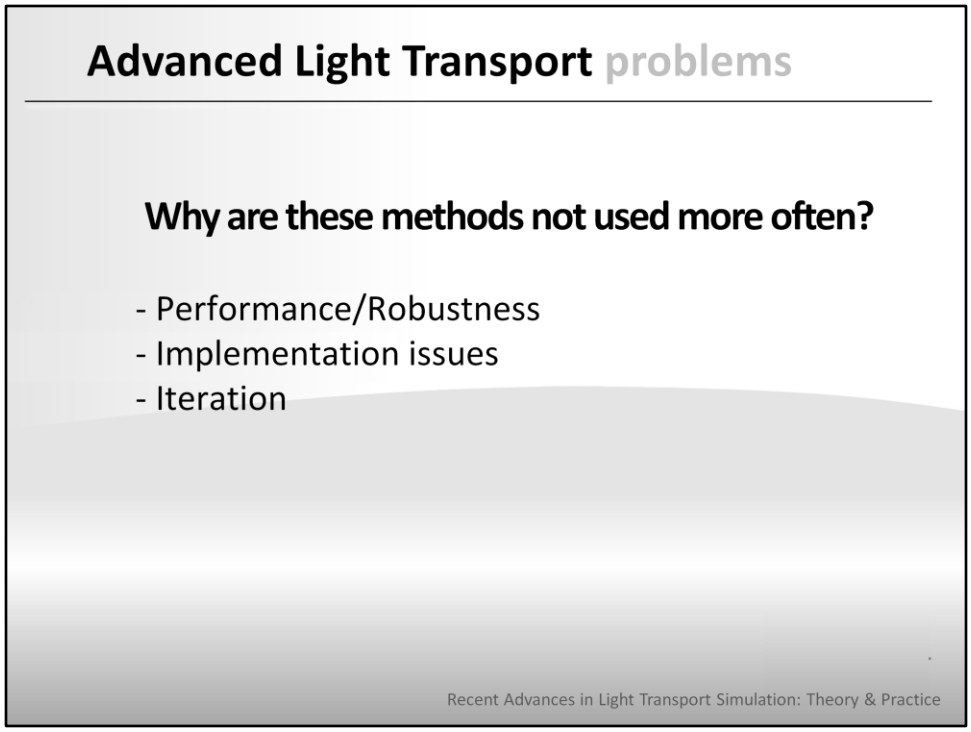


Are recent advances in light transport used by the industry?: No.

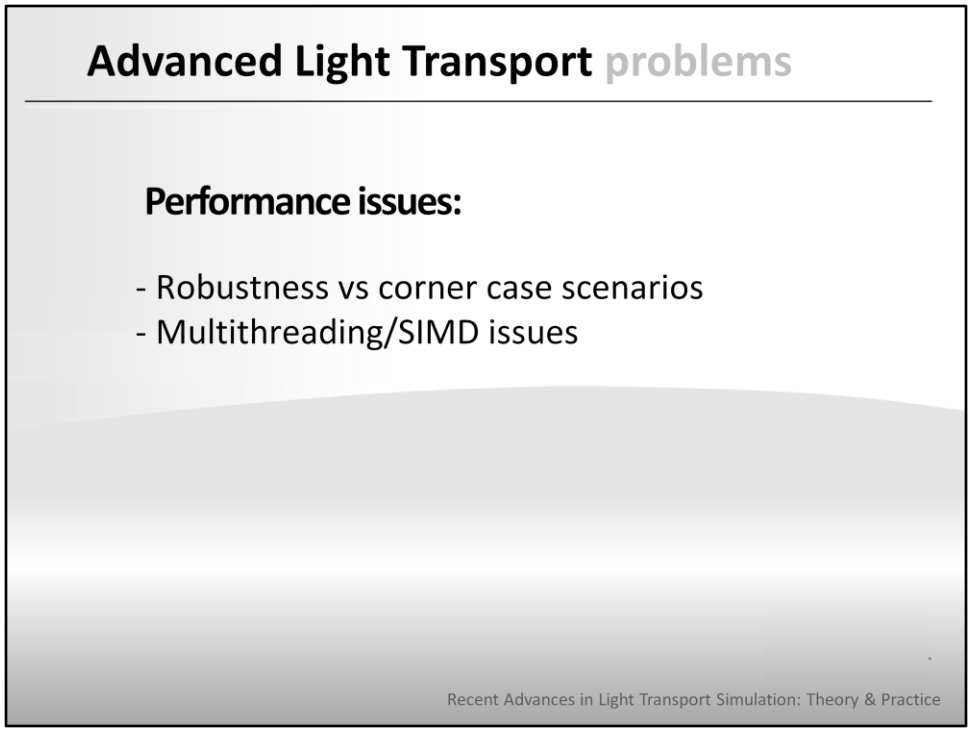
The majority of commercial renderers are really not getting the most of the state of the art know-how in light transport. In fact many of them are just path tracers with multiple importance sampling (MIS), a technique published in papers almost 20 years ago. Few renders go beyond and make use of bidirectional path tracer, metropolis light transport, vertex merging and other more modern techniques.

Lighting artists have learned to optimize scenes to be used with path tracers with MIS (i.e replacing complex lighting with IBL lighting, avoiding interiors, complex scenarios with too many reflective/refractive surfaces, etc).

It is also worth mentioning that in many cases, disruptive changes are not adopted quickly. Professionals have invested years in learning how to use specific rendering techniques and know how to control the parameters in order to optimize render times or reduce artifacts. Complex rendering pipelines have been built on top of these rendering methods. Therefore new techniques that require different approaches take time before they become standard (i.e. physically based rendering has been commercially available for a long time, but only recently has it become very popular).



When discussing which rendering algorithm is better, most of the time the discussion is focused solely on convergence and performance. However there are other aspects that are equally important, such the difficulty of implementation or the appearance of the image while it converges.



Performance issues:

Some of the rendering algorithms mentioned here (specifically the ones based in metropolis light transport - MLT), behave well in some scenarios but poorly in many others.

While these techniques have a lot of relevance from the researcher perspective it is often more desirable for a commercial renderer to sacrifice performance in corner case scenarios in order to be more robust and generic. Performance in simple scenes is critical, because users simplify scenes for rendering. Finding a way for the engine to behave well in complex scenes, while maintaining performance in simpler ones at the same level as basic path tracers, is one of the main open challenges.

Also, these implementations usually need the developer to adjust internal parameters within the render engine, which dramatically affect performance, such how many mutations are done, and how and when they expand to neighbor pixels.

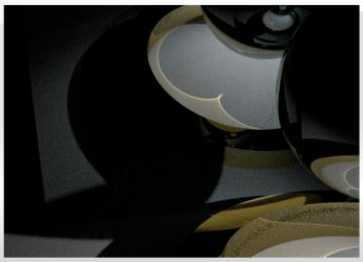
Also regarding performance, MLT algorithms are less SIMD/GPU friendly than simpler approaches. Depending how the mutation system works, it is very likely that calculations in some areas of the image are several orders of magnitude more expensive than in others. It is precisely because of the nature of the exploration/mutations, that the parallelization of the algorithms become more

complex and less efficient.

Advanced Light Transport problems

Implementation issues:

- MLT/ERT, etc implementations are convoluted
- Commercial renderers have to implement features on top
- Debugging is not fun



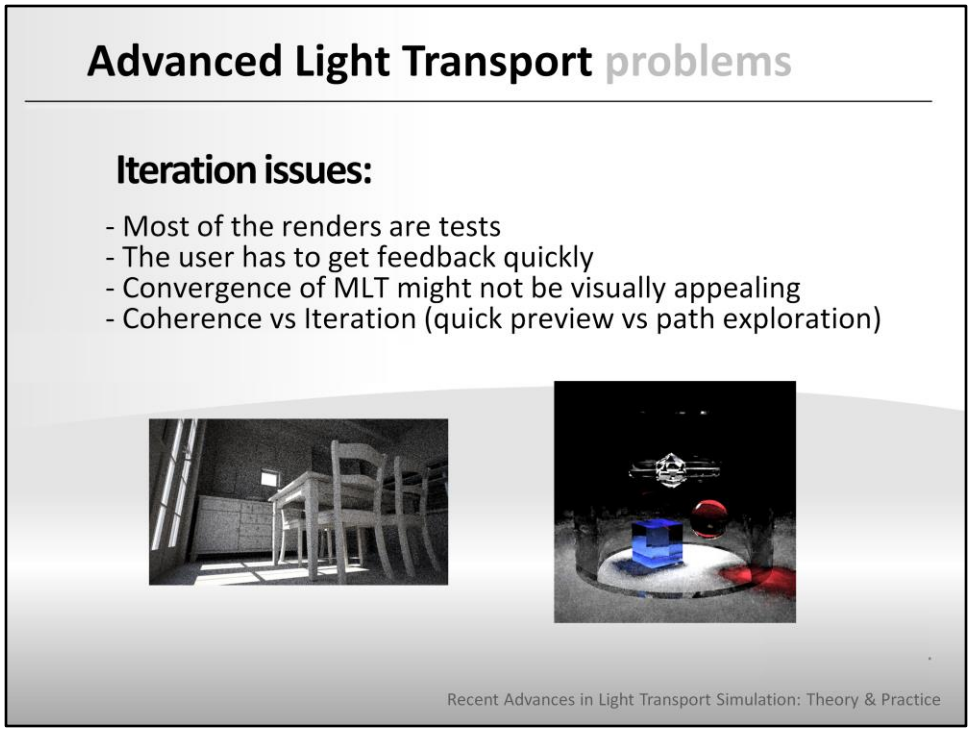
Recent Advances in Light Transport Simulation: Theory & Practice

Implementation issues:

The implementation of BPT/ERT/MLT based algorithms is significantly more convoluted than other approaches.

The problem gets worse because commercial renderers have to implement a huge amount of features that are not taken into account in non-commercial ones. Namely implementing things as matte objects, shadow passes with GI, additive (even non energy conserving) materials, etc. These all add a layer of complexity on top of something that is already difficult to debug and maintain.

Debugging can also be extremely tricky, totally leaving the geometry domain and entering a whole new fun universe of joy.

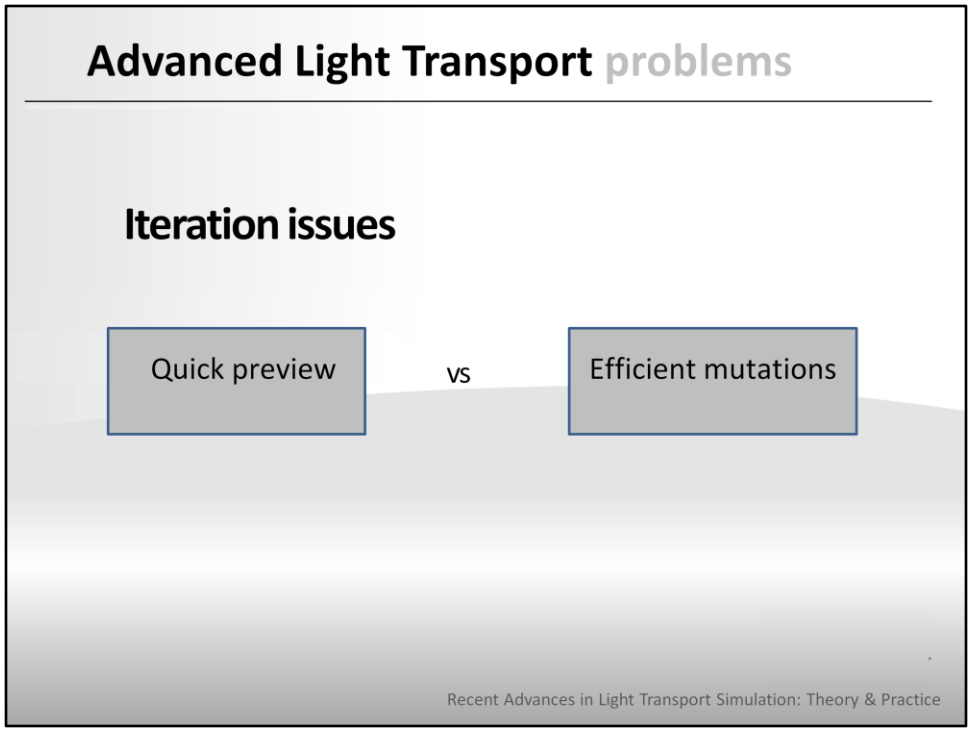


One aspect that is usually not mentioned is how these algorithms iterate (in other words, how the render progress is shown).

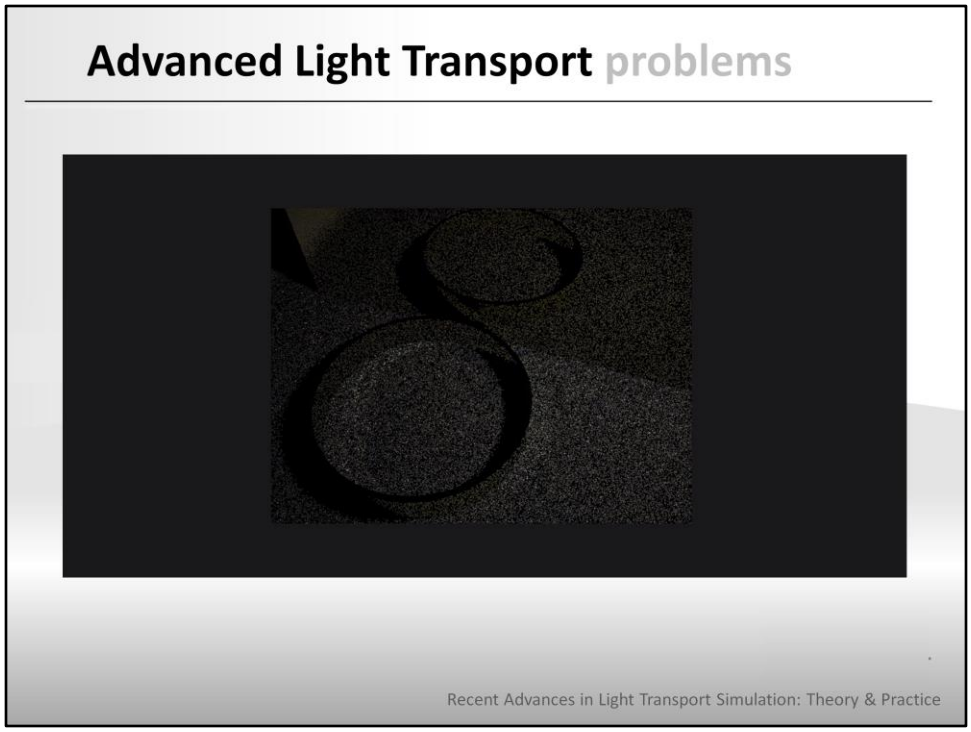
Almost all the renders done by an artist are trial & error, not final images. They need quick results. In these cases techniques such as path tracing or bidirectional path tracing may be less capable of solving the difficulties of the scene, but provide a more consistent look that might give more useful information so the artist can make decisions and iteratively adjusts parameters. MLT techniques, on the other hand, can be visually less appealing than pure path tracers, which show the render progress more or less equally in the whole image

Precisely the strength of mutation-based engines, which is exploring coherence efficiently, introduces a problem here. Most of the time, the image does not converge homogeneously but some areas clean much faster than others. This pollutes the image with a characteristic patches-based look that is less pleasing for users.

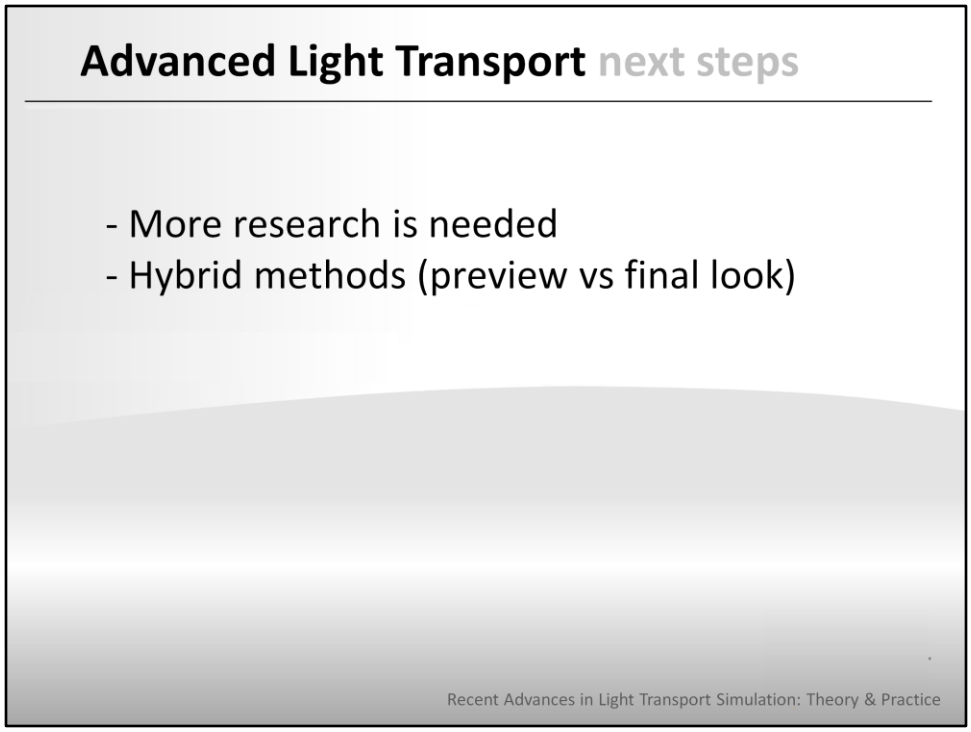
Coherence vs Iteration: It is not mentioned often but this is one of the main open problems to be solved. The faster we want the preview, the less each patch should explore.



See videos (convergence_1 & convergence_2) which show different ratios of time spent per pixel. While “convergence_1” produces an image with more noise, it quickly gives an overall idea of how the image will look. “convergence_2” spends more time on each region so they get cleaner faster, but the rest of the image remains black.



The video “convergence_3” shows the same scene as in the previous videos but using a bidirectional path tracing instead. There is less information in the scene but a better look from the beginning. It is easier to make decisions based on this image than on one with a few buckets clean surrounded by too many dark areas.



In real life there are still many areas where current rendering technologies are not powerful enough to provide generic solutions viable for commercial applications. More research is needed, this is an open problem. Biased solutions can solve some of these problems but restrict the scope of these techniques to visualization/artistic purposes, so the render engine cannot be used as a design tool or produce reliable photometric data.

Latest relevant contributions: Manifold exploration, Vertex connection and merging. More work is needed following the same philosophy: *”Parameter Control for Metropolis Light Transport “(Zsolnai, Laszlo)*.

Advanced Light Transport questions

Questions

Thanks!