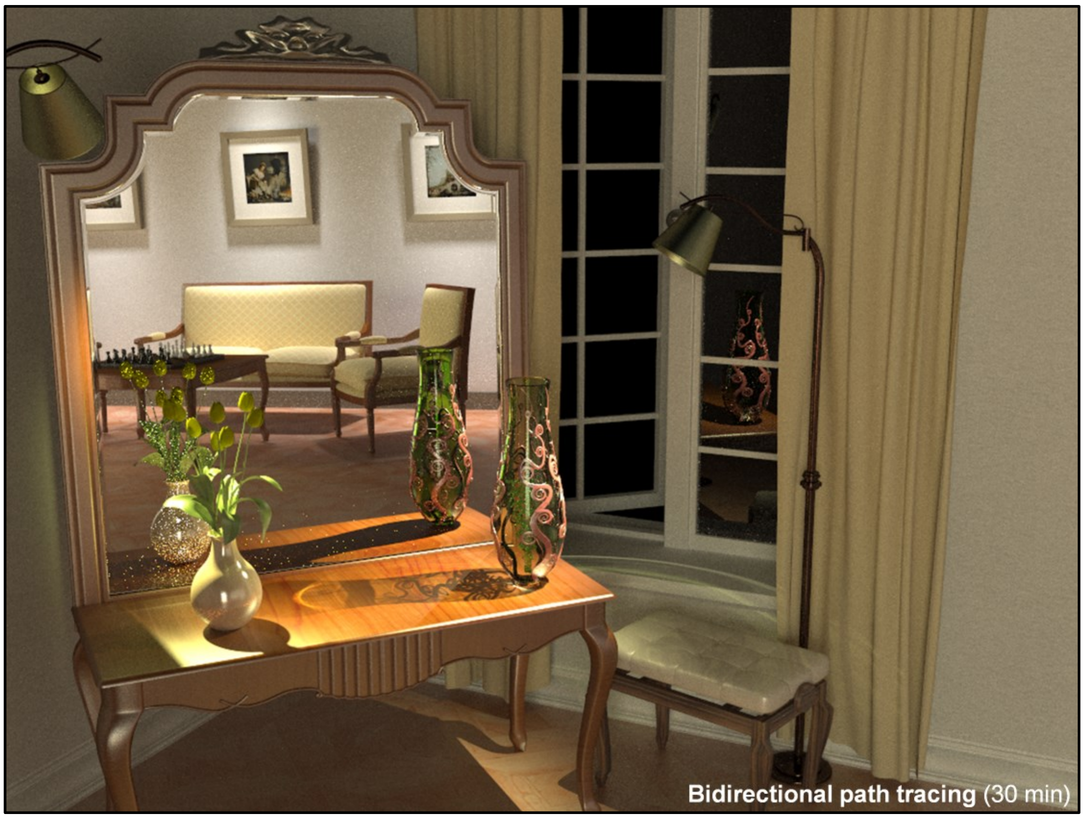
Combining Photon Mapping and Bidirectional Path Tracing

SIGGRAPH ASIA 2013

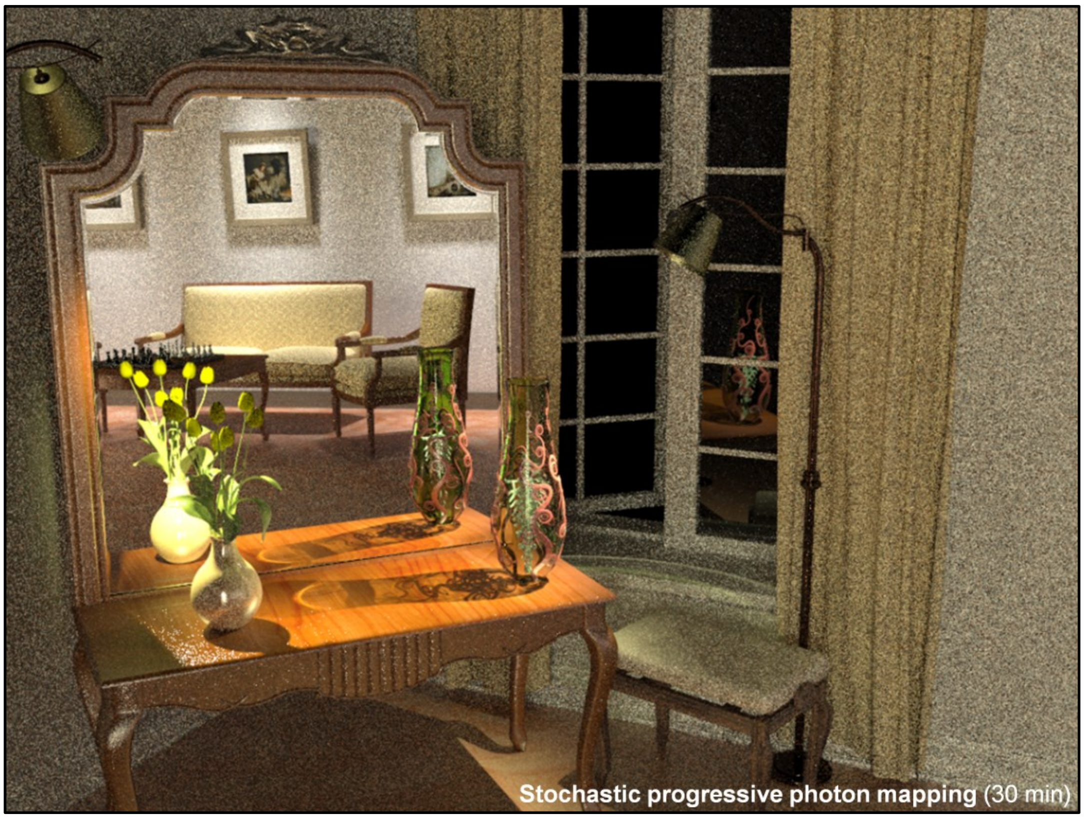
SA2013.SIGGRAPH.ORG

SPONSORED BY

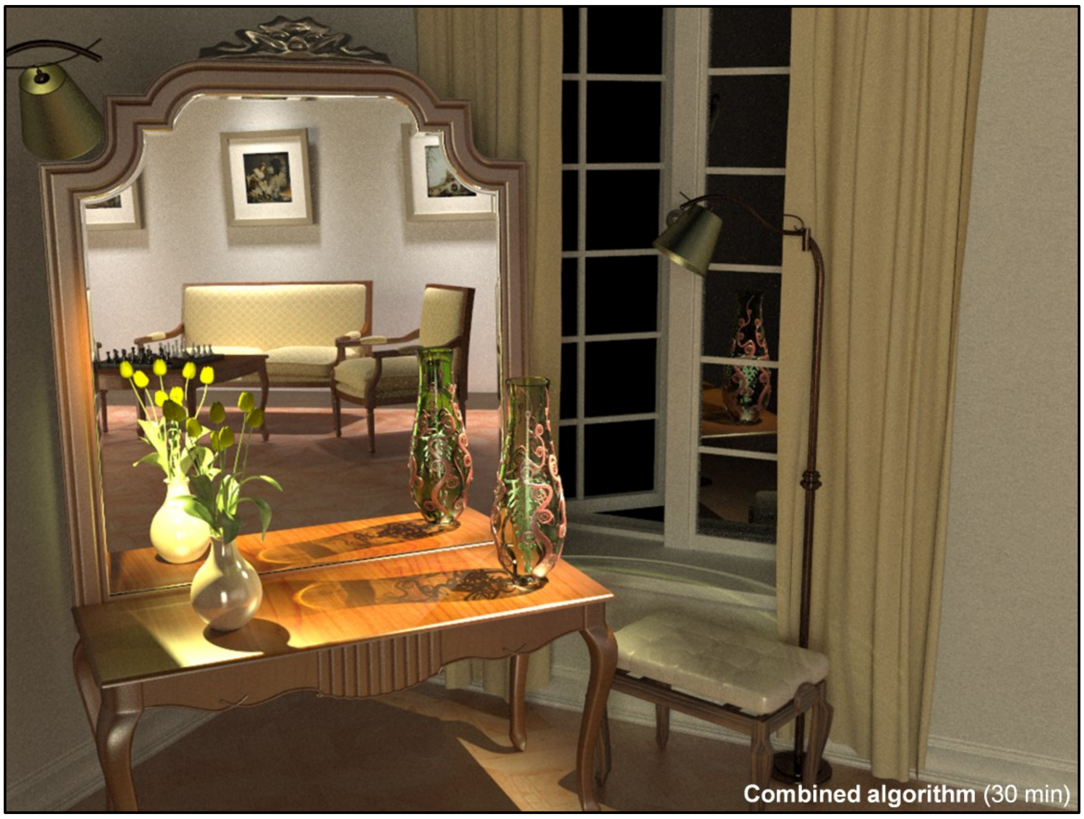
The previous presentations in this course demonstrated the robustness of photon mapping under various difficult lighting conditions. In this talk, I will show how photon mapping can be combined with bidirectional path tracing – another very robust rendering algorithm – to handle complex light transport even more efficiently.



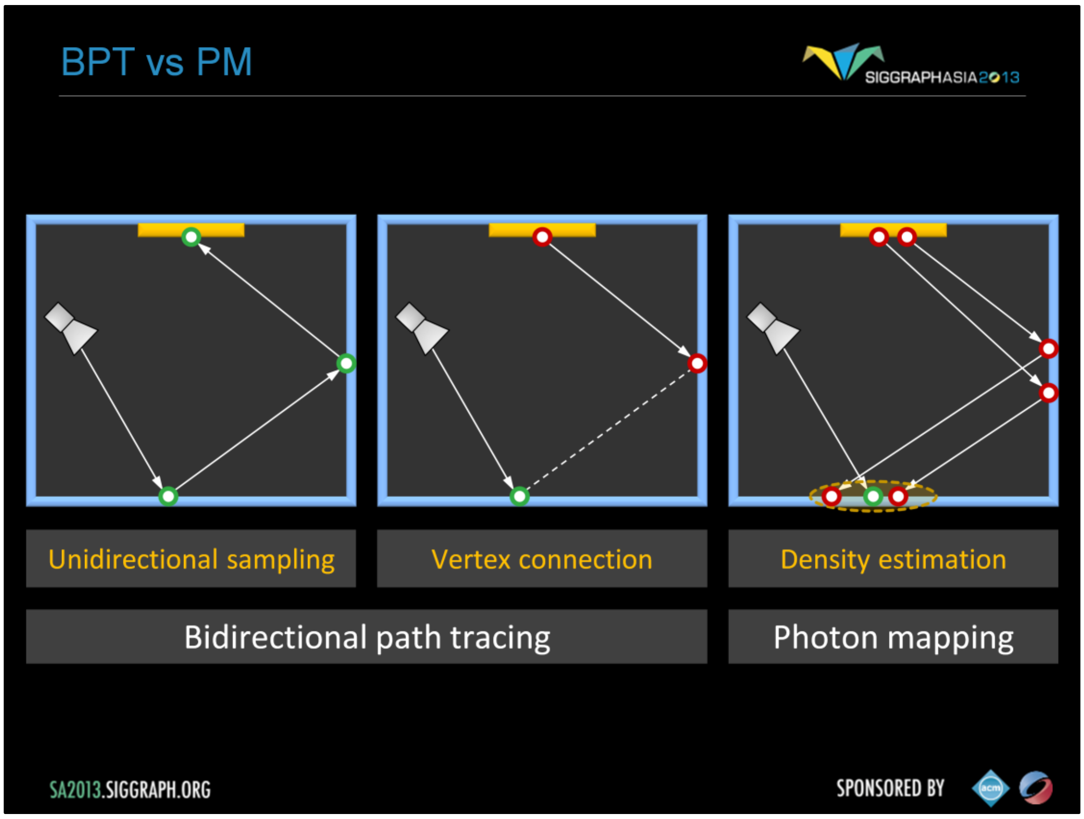
Bidirectional path tracing is one of the most versatile light transport simulation algorithms available today. It can robustly handle a wide range of illumination and scene configurations, but is notoriously inefficient for specular-diffuse-specular light interactions, which occur e.g. when a caustic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



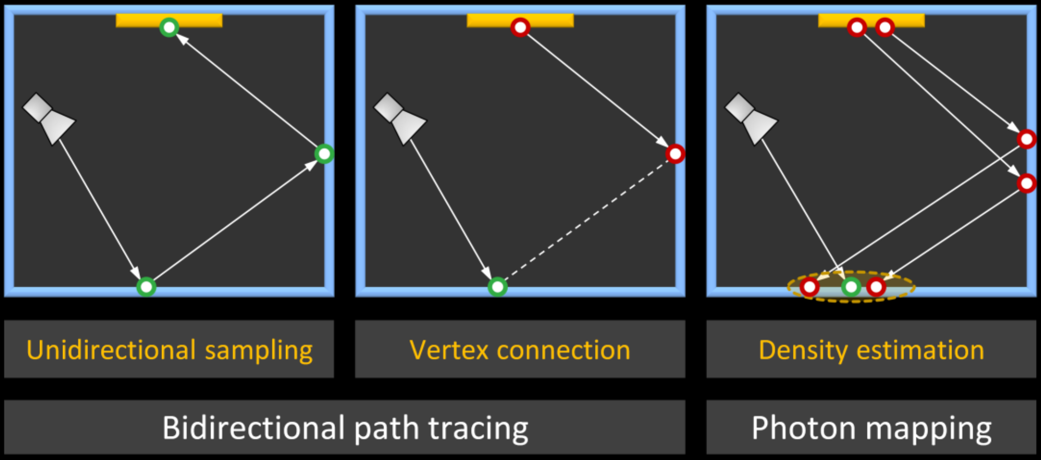
On the other hand, photon mapping (PM) is well known for its efficient handling of caustics. Recently, Hachisuka and Jensen [2009] showed a progressive variant of PM that converges to the correct solution with a fixed memory footprint. Their stochastic progressive photon mapping (PPM) algorithm captures 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.



BPT vs PM



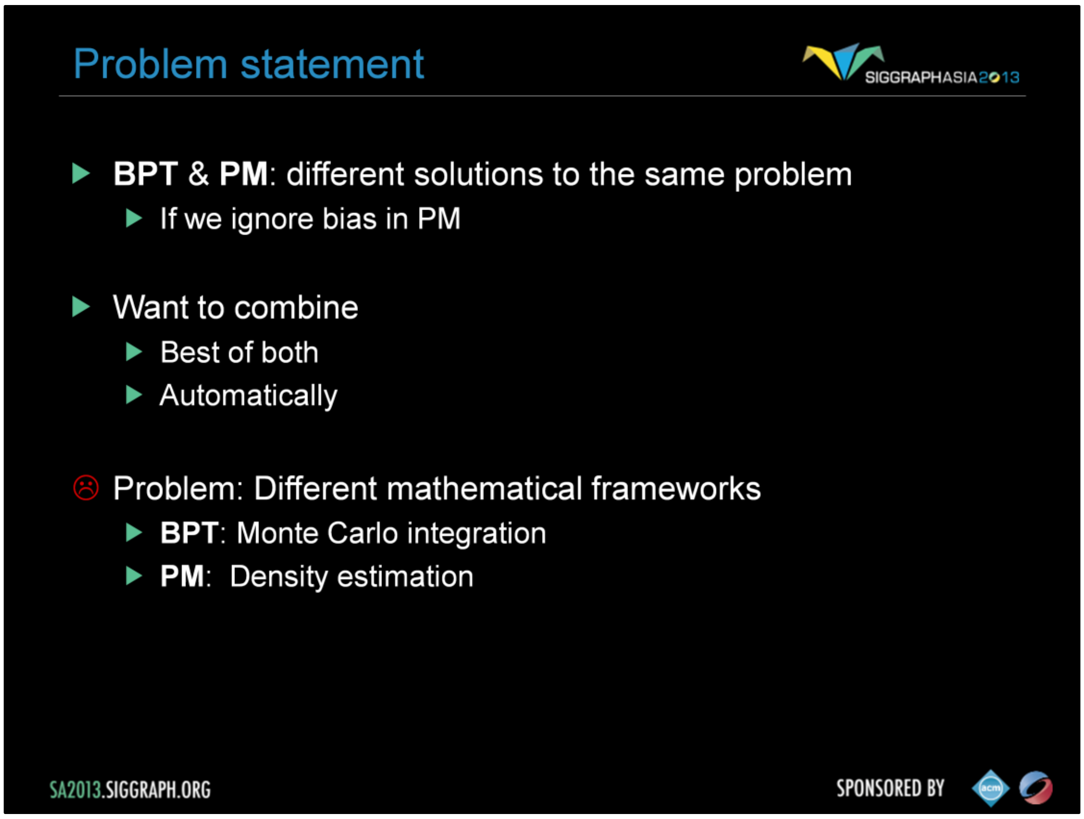
SA2013.SIGGRAPH.ORG

SPONSORED BY

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 connecting 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.



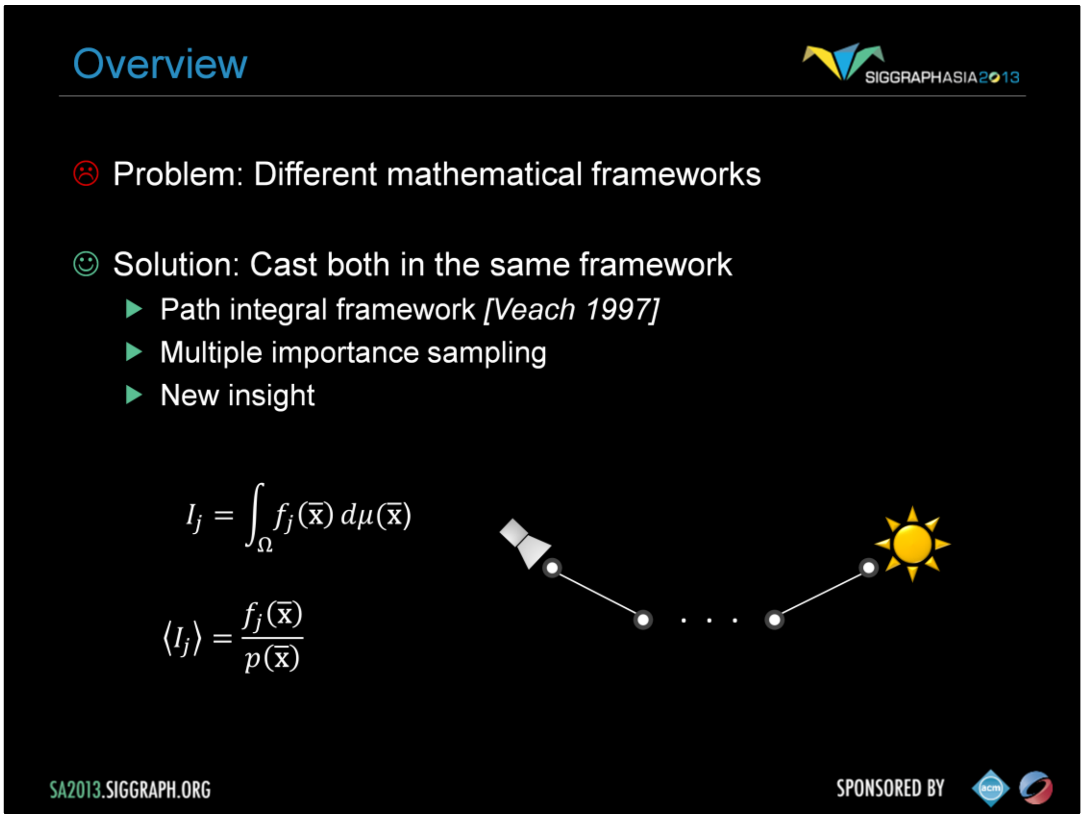
Problem statement

- **BPT & PM**: different solutions to the same problem
  - If we ignore bias in PM
- Want to combine
  - Best of both
  - Automatically
- ☹ Problem: Different mathematical frameworks
  - **BPT**: Monte Carlo integration
  - **PM**: Density estimation

SA2013.SIGGRAPH.ORG SPONSORED BY

We see that bidirectional path tracing (BPT) and photon mapping (PM) are different solutions to the same problem. That is, if we ignore the bias introduced by PM. Furthermore, the previous comparison shows that BPT and PM complement each other in terms of the light transport effects they can efficiently handle.

It is thus logical to want to combine these two methods. Ideally, we want an automatic combination that preserves the qualities of each. Interestingly, even though both methods have been published over 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.



Overview

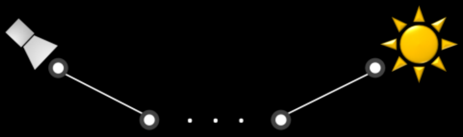
☹ Problem: Different mathematical frameworks

☺ Solution: Cast both in the same framework

- Path integral framework *[Veach 1997]*
- Multiple importance sampling
- New insight

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

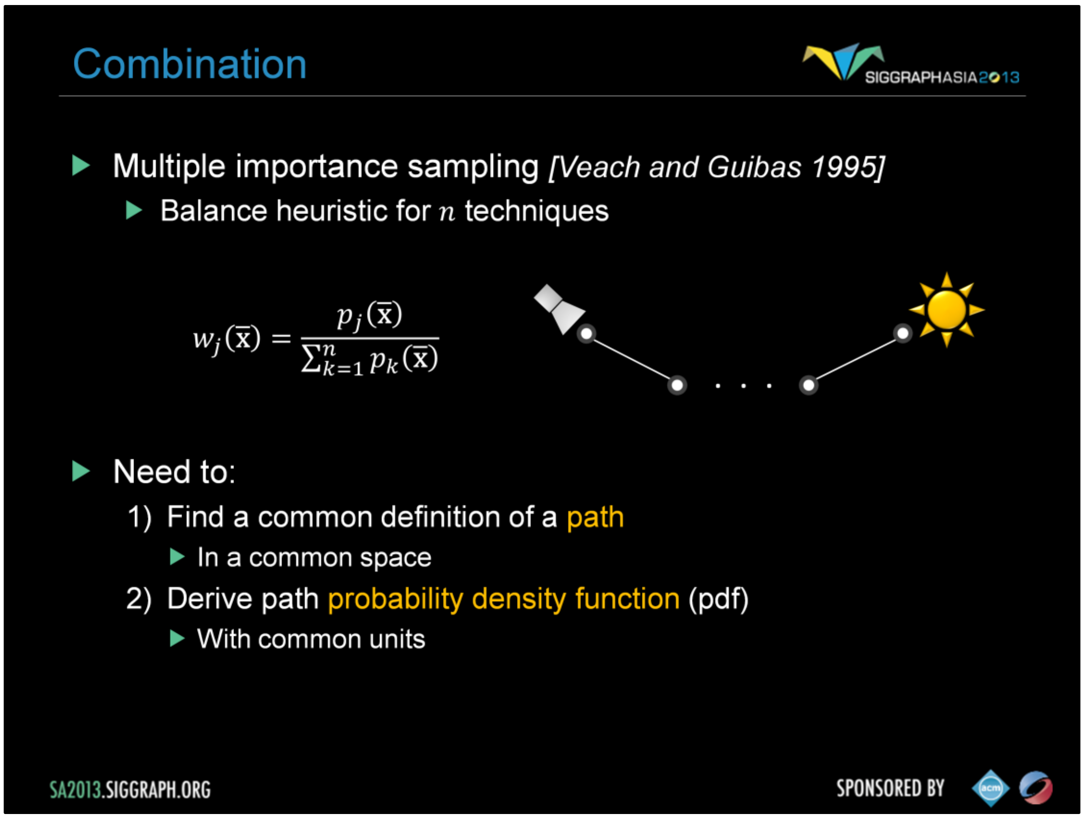
$$\langle I_j \rangle = \frac{f_j(\overline{\mathrm{x}})}{p(\overline{\mathrm{x}})}$$



SA2013.SIGGRAPH.ORG SPONSORED BY

The first step toward combining these two methods is to cast them in the same mathematical framework. We choose Veach's path integral framework where BPT is already naturally defined. This framework formulates the problem of computing the value of a pixel 'j' as an integral over the energy contribution of all light transport paths from the light sources to the camera. An estimator for this value is obtained by sampling one such random path and dividing its contribution by the path pdf. Different path sampling techniques result in different path pdfs, and BPT uses multiple importance sampling to efficiently combine the resulting estimators.

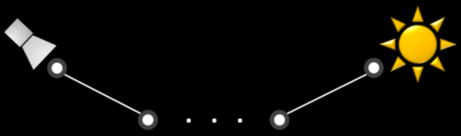
Later on, we will see that casting both methods in the same path integral framework will also provide a basis for reasoning about the relative efficiency of BPT and PM.



Combination

- Multiple importance sampling *[Veach and Guibas 1995]*
  - Balance heuristic for n techniques

$$w_j(\overline{\mathrm{x}}) = \frac{p_j(\overline{\mathrm{x}})}{\sum_{k=1}^{n} p_k(\overline{\mathrm{x}})}$$



- Need to:
  1) Find a common definition of a path
     - In a common space
  2) Derive path probability density function (pdf)
     - With common units

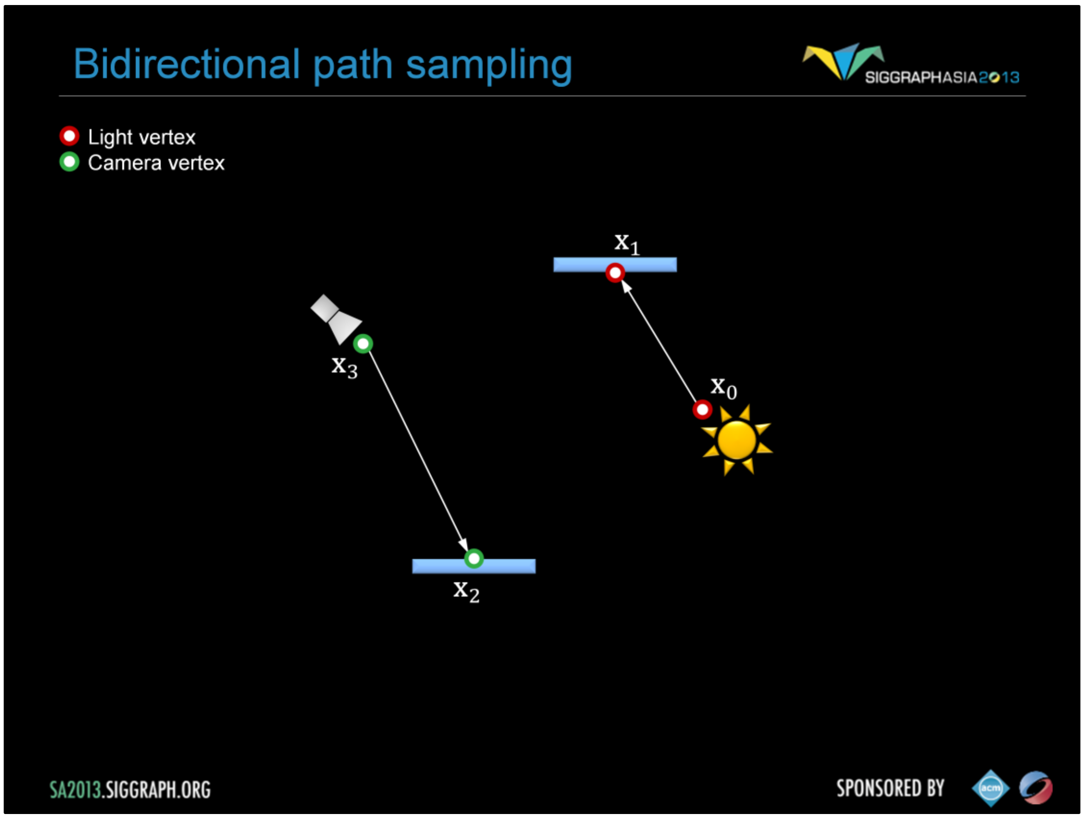
SA2013.SIGGRAPH.ORG

SPONSORED BY

Multiple importance sampling (MIS) combines the estimators corresponding to different path sampling techniques by assigning each estimator a weight that is a function of the actual sampled path. The balance heuristic makes this weight proportional to the path pdf.

In order to combine BPT and PM using MIS, we need two things. First, we have to find a common definition of the light transport paths sampled by both methods. Recall that photon mapping has been originally defined as an outgoing radiance estimator, without explicit distinction between individual full light transport paths constructed with each photon subpath.

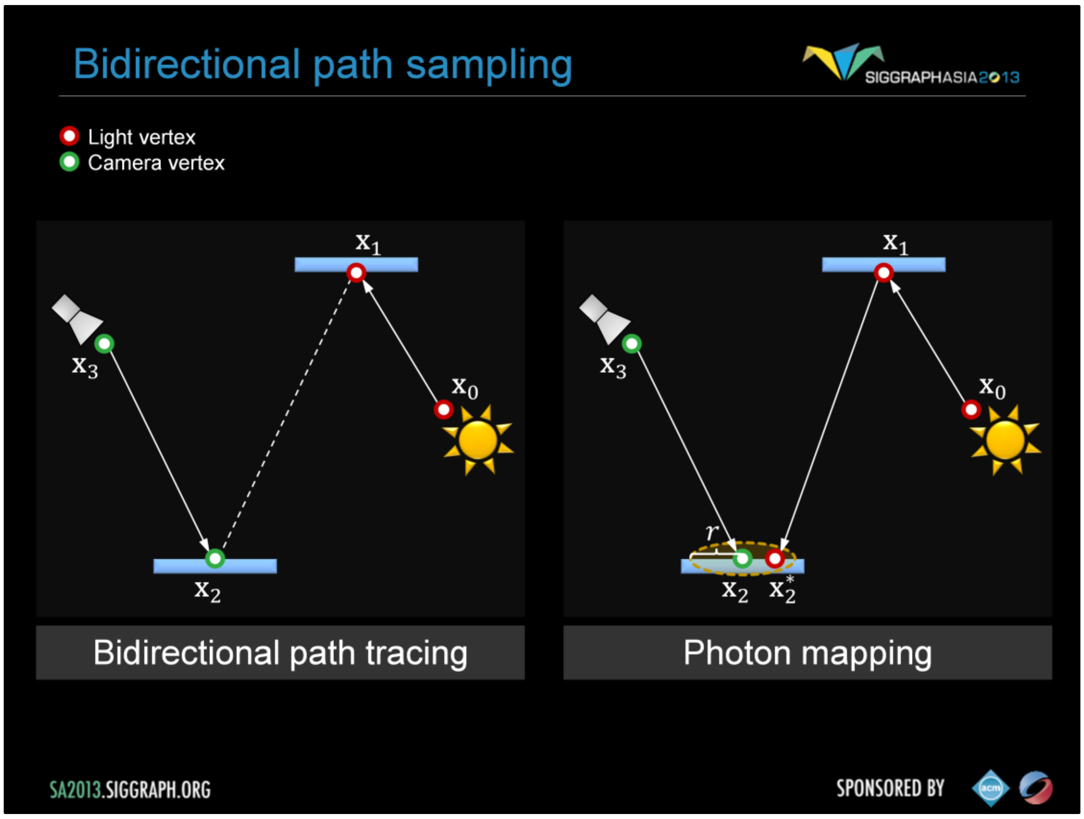
Second, we need to derive the probability density function (pdf) for sampling these paths. In order to apply the balance heuristic in a meaningful way, the pdfs corresponding to the path sampling techniques in both methods should have the same units. This means that the paths need to reside in the same space, i.e. have the same number of vertices.



Bidirectional path sampling

- 🔴 Light vertex
- 🟢 Camera vertex

To find a common definition of a path, let us follow the sampling procedures used in BPT and PM. We start by tracing two independent subpaths, one from a light source and another one from the camera.

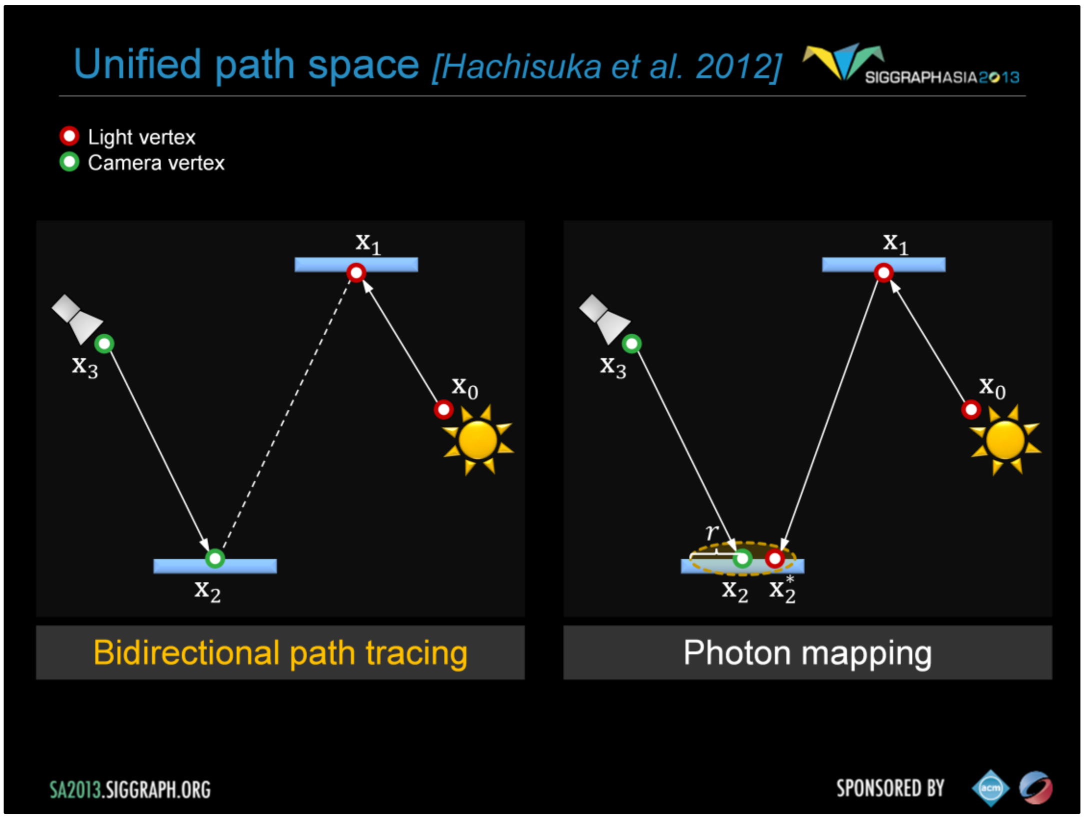


Bidirectional path sampling

Now let us see how BPT and PM complete a full path.

Given these two subpaths, BPT places a shadow connection between their endpoints. Photon mapping, on the other hand, extends the light subpath by sampling one more vertex from $\mathbf{x}_1$, and then concatenates the subpaths only if the photon hit-point $\mathbf{x}_2^*$ lies within a distance r from $\mathbf{x}_2$.

The resulting full path in PM has one more vertex than the corresponding BPT path. Thus, they reside in different path spaces and their pdfs also have different units. Two recent works on combining PM and BPT have approached this problem in different ways.



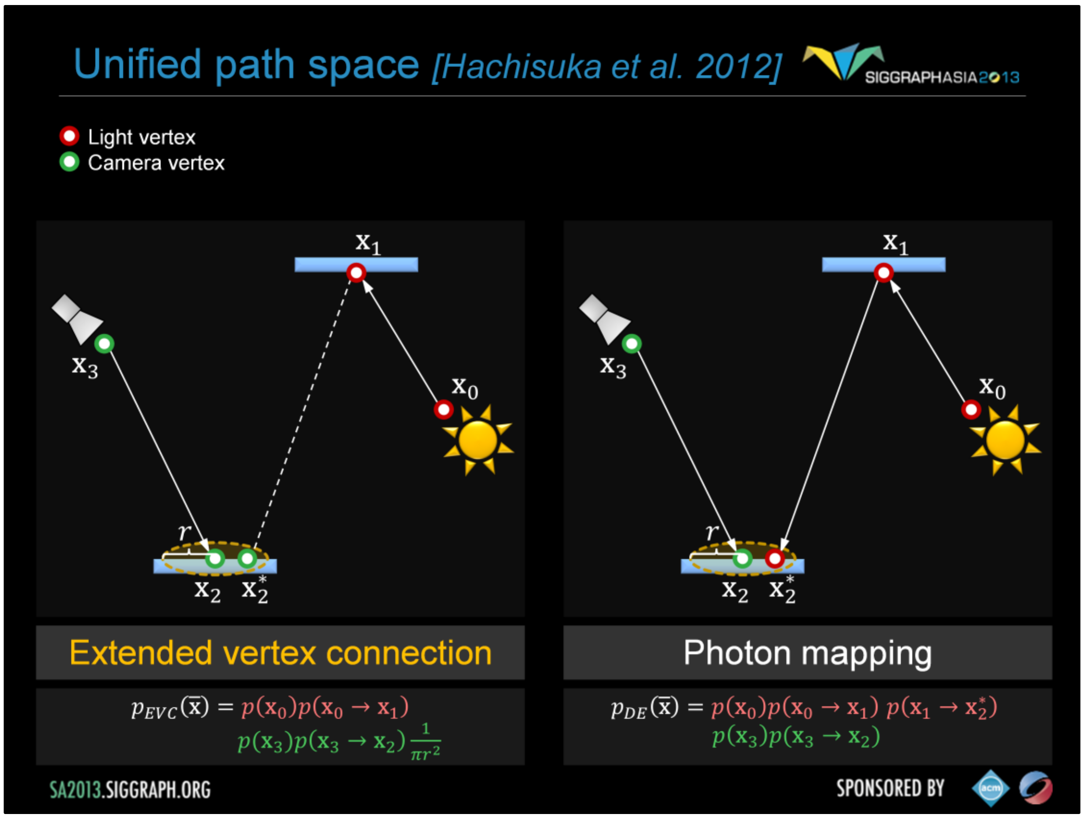
Unified path space *[Hachisuka et al. 2012]*

- 🟥 Light vertex
- 🟩 Camera vertex

SA2013.SIGGRAPH.ORG

SPONSORED BY

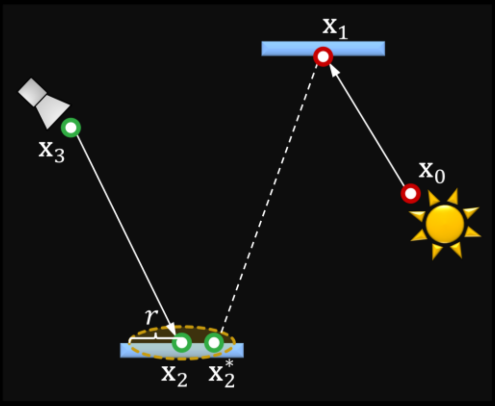
Hachisuka et al. [2012] consider an extension of BPT that samples paths in the higher-dimensional space of PM.

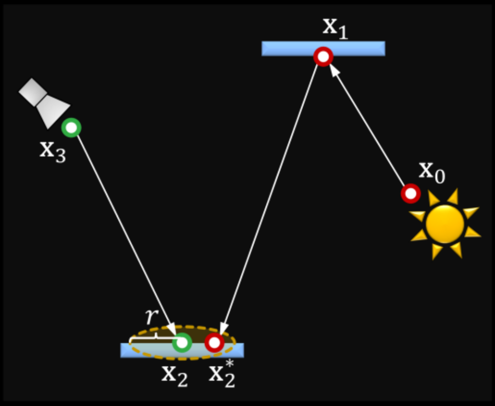


Unified path space *[Hachisuka et al. 2012]*

SIGGRAPHASIA2013

- ● Light vertex
- ● Camera vertex





Extended vertex connection

$$p_{EVC}(\overline{\mathrm{x}}) = p(\mathrm{x}_0)p(\mathrm{x}_0 \rightarrow \mathrm{x}_1) \; p(\mathrm{x}_3)p(\mathrm{x}_3 \rightarrow \mathrm{x}_2)\frac{1}{\pi r^2}$$

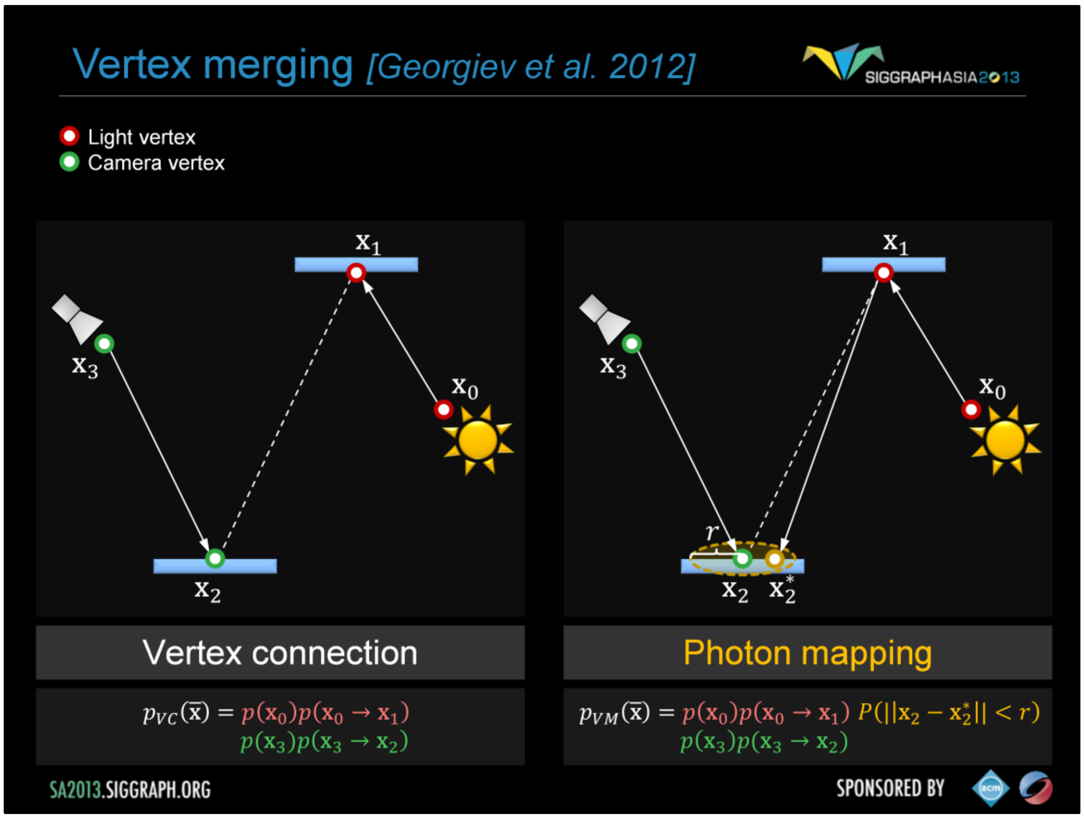
Photon mapping

$$p_{DE}(\overline{\mathrm{x}}) = p(\mathrm{x}_0)p(\mathrm{x}_0 \rightarrow \mathrm{x}_1)\, p(\mathrm{x}_1 \rightarrow \mathrm{x}_2^*) \; p(\mathrm{x}_3)p(\mathrm{x}_3 \rightarrow \mathrm{x}_2)$$

SA2013.SIGGRAPH.ORG SPONSORED BY

This is done by considering a random perturbation of vertex $\mathbf{x}_2$ in an r-neighborhood that yields a new vertex $\mathbf{x}_2^*$. This vertex is then connected to $\mathbf{x}_1$. With this modification, both algorithms sample paths in the same higher-dimensional space. In the case of unidirectional sampling, the camera subpath tracing continues from $\mathbf{x}_2^*$.

The path pdf of the extended vertex connection technique is the product of the pdfs of the individual vertices, where the pdf of $\mathbf{x}_2^*$ is $\frac{1}{\pi r^2}$, as it is sampled uniformly in a circle with a radius r. The pdf of the corresponding photon mapping technique is the product of the vertex pdfs, which are given by the random walk sampling procedure.



Vertex merging *[Georgiev et al. 2012]*

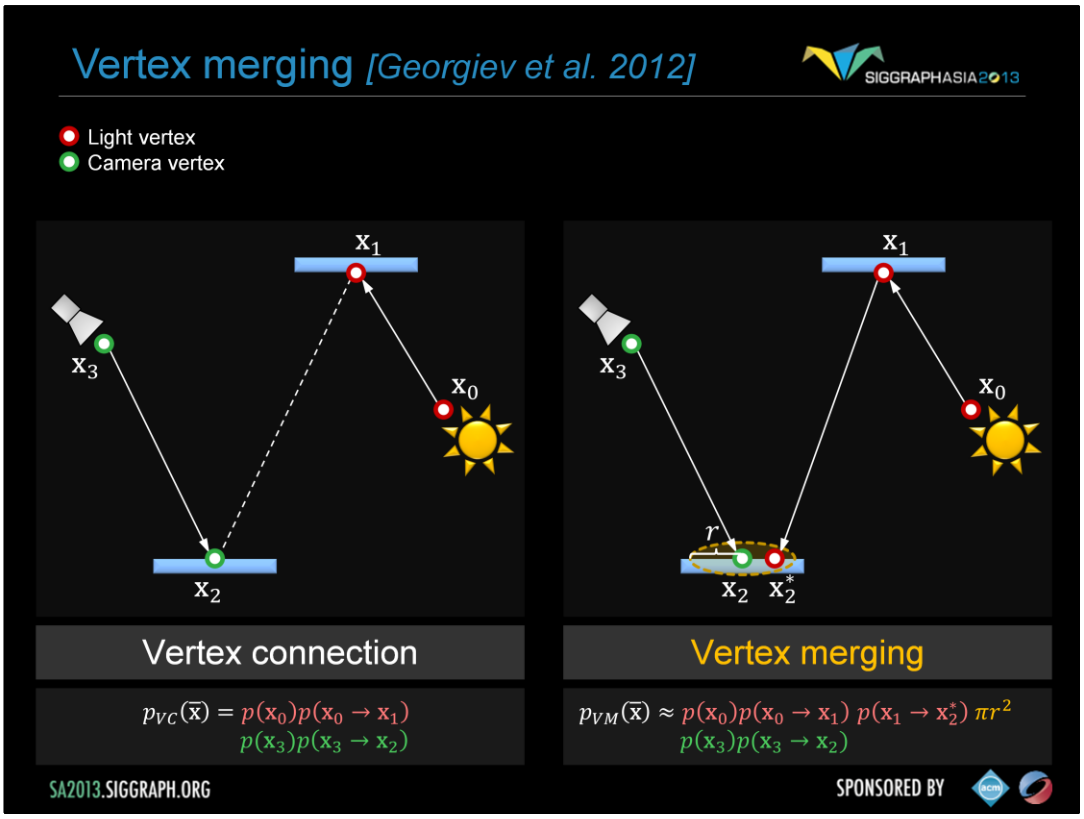
● Light vertex
● Camera vertex

$$p_{VC}(\bar{\mathrm{x}}) = p(\mathrm{x}_0)p(\mathrm{x}_0 \to \mathrm{x}_1)\, p(\mathrm{x}_3)p(\mathrm{x}_3 \to \mathrm{x}_2)$$

$$p_{VM}(\bar{\mathrm{x}}) = p(\mathrm{x}_0)p(\mathrm{x}_0 \to \mathrm{x}_1)\, P(||\mathrm{x}_2 - \mathrm{x}_2^*|| < r)\, p(\mathrm{x}_3)p(\mathrm{x}_3 \to \mathrm{x}_2)$$

SA2013.SIGGRAPH.ORG SPONSORED BY

In contrast, the vertex merging interpretation of Georgiev et al. [2012] considers the last step in photon mapping 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 simply a Russian roulette decision. The full path pdf is then the product of the subpath pdf 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 $\mathbf{x}_1$.



Vertex merging *[Georgiev et al. 2012]*

● Light vertex
● Camera vertex

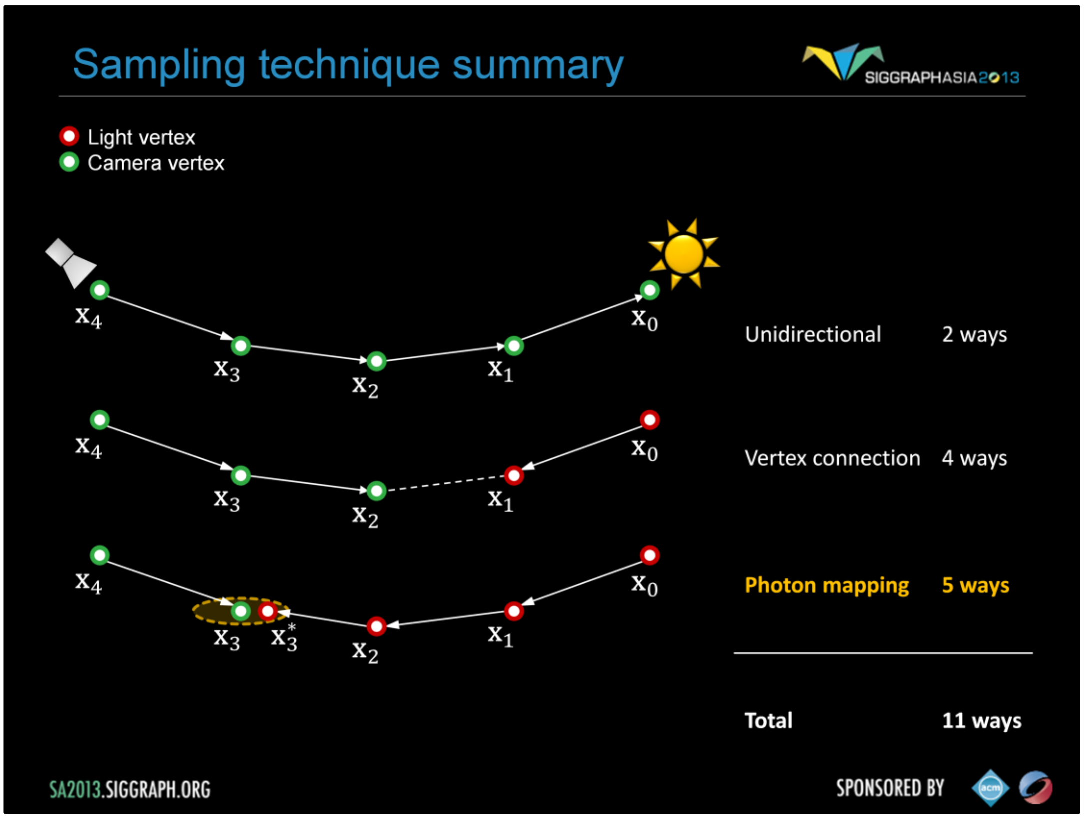
Vertex connection	Vertex merging
$p_{VC}(\bar{\mathrm{x}}) = p(\mathrm{x}_0)p(\mathrm{x}_0 \to \mathrm{x}_1)\, p(\mathrm{x}_3)p(\mathrm{x}_3 \to \mathrm{x}_2)$	$p_{VM}(\bar{\mathrm{x}}) \approx p(\mathrm{x}_0)p(\mathrm{x}_0 \to \mathrm{x}_1)\, p(\mathrm{x}_1 \to \mathrm{x}_2^*)\, \pi r^2\, p(\mathrm{x}_3)p(\mathrm{x}_3 \to \mathrm{x}_2)$

SA2013.SIGGRAPH.ORG

SPONSORED BY

Under the assumptions that the surface around $\mathbf{x}_1$ is locally flat, i.e. that the r-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. This technique is called *vertex merging*, as it can be intuitively thought to weld the endpoints of the two subpaths if they lie close to each other.

Now that we have common definitions of the paths and their corresponding pdfs in both BPT and PM, we can use the balance heuristic to compute path weights that account for all possible ways to sample the same path in both algorithms. Note that the only difference in the path pdfs given by the two different interpretations is the πr^2 factor in the vertex merging, which appears as a denominator in the extended vertex connection of Hachisuka et al. [2012]. Interestingly, both interpretations result in the same path weights when the path pdfs are plugged into the balance heuristic formula.

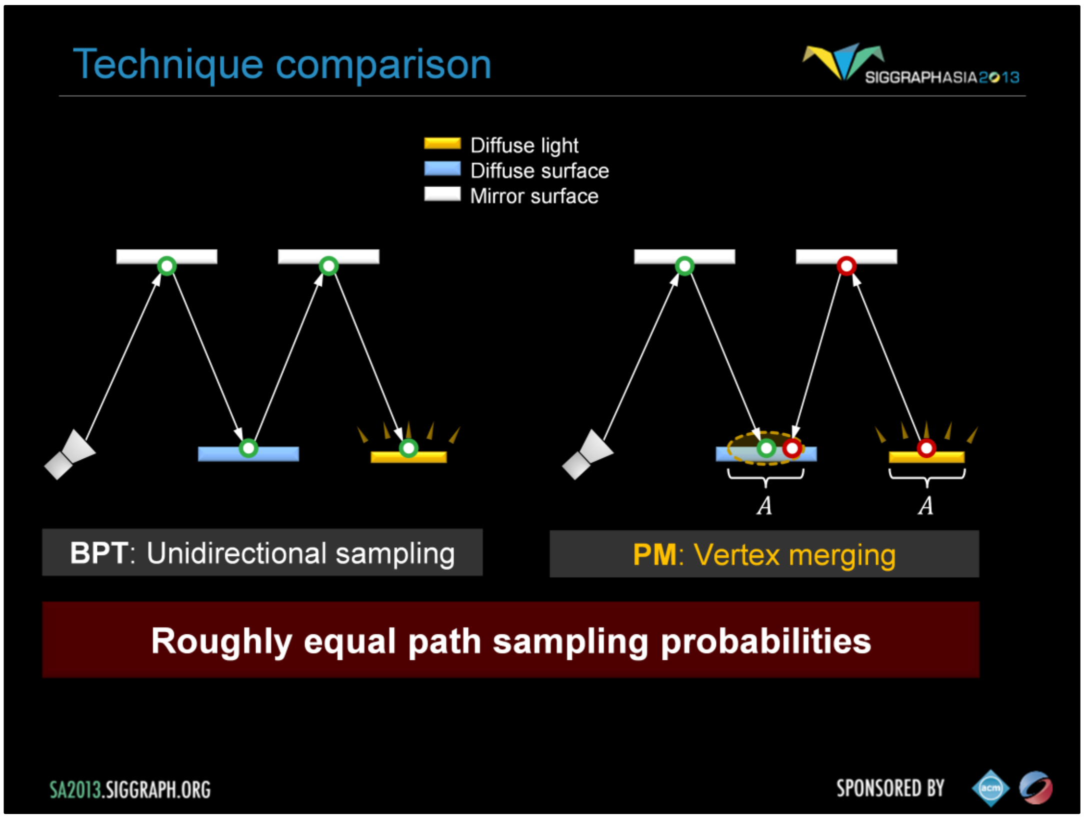


Sampling technique summary

Unidirectional	2 ways
Vertex connection	4 ways
Photon mapping	**5 ways**
Total	**11 ways**

Having formulated photon mapping as a path sampling technique, we can put it side by side with the techniques in BPT. There are two ways to sample a length-4 path unidirectionally, and four ways to sample it via vertex connection. Photon mapping 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.

With so many ways to sample the same light transport path, an interesting question arises: which technique is the most efficient for which types of paths? In the following discussion, we will use the vertex merging interpretation to argue about the relative efficiency of BPT and PM.



Technique comparison

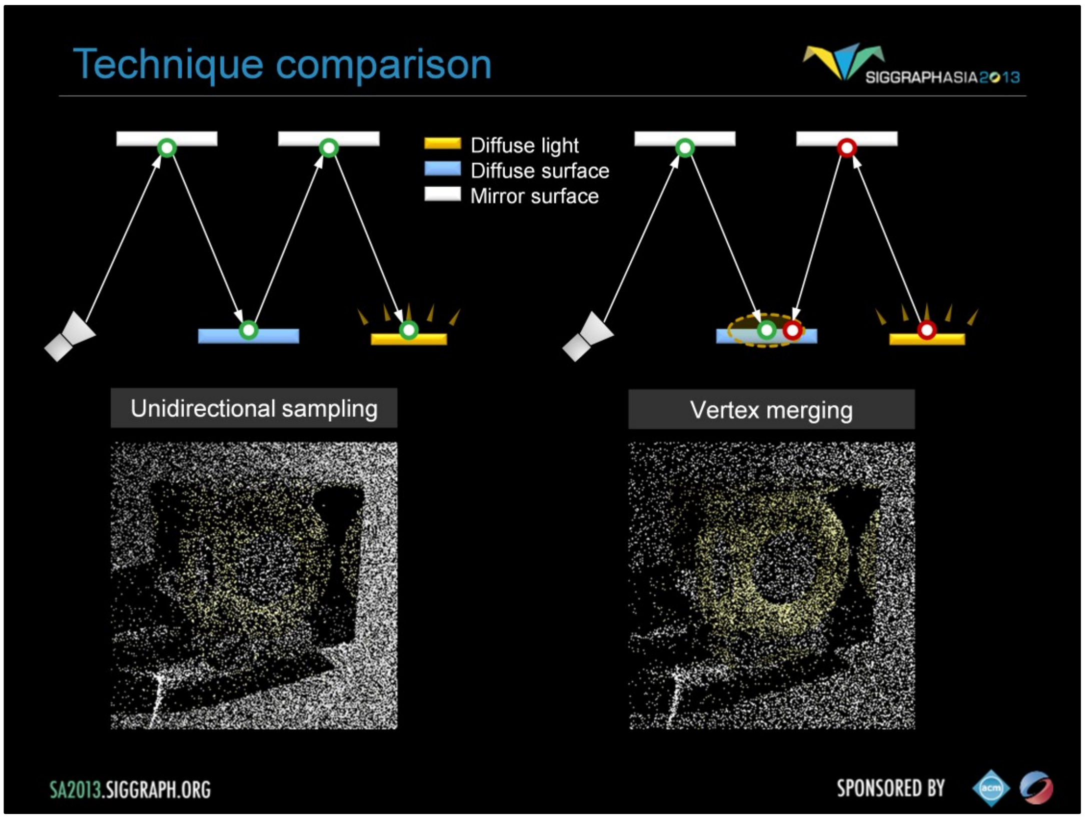
BPT: Unidirectional sampling

PM: Vertex merging

Roughly equal path sampling probabilities

Let us first take a look at specular-diffuse-specular (SDS) paths. Here, BPT 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.

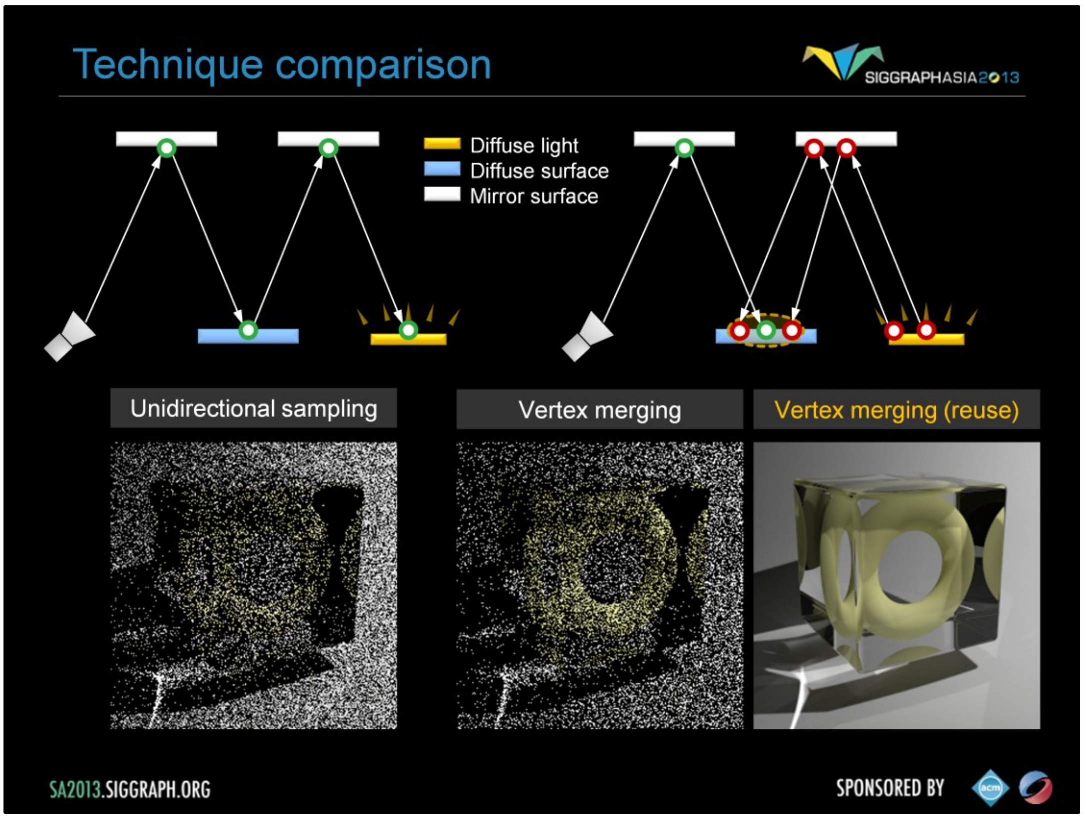


Technique comparison

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 termination or hitting the light. For image on the right, we trace subpaths from both ends, and concatenate them if their endpoints if they lie within a distance $r = \sqrt{A/\pi}$ from each other. Both images look equally noisy, even with 10,000 paths per pixel. This confirms that the path sampling technique used in photon mapping is *not* intrinsically more robust for SDS paths than unidirectional sampling.

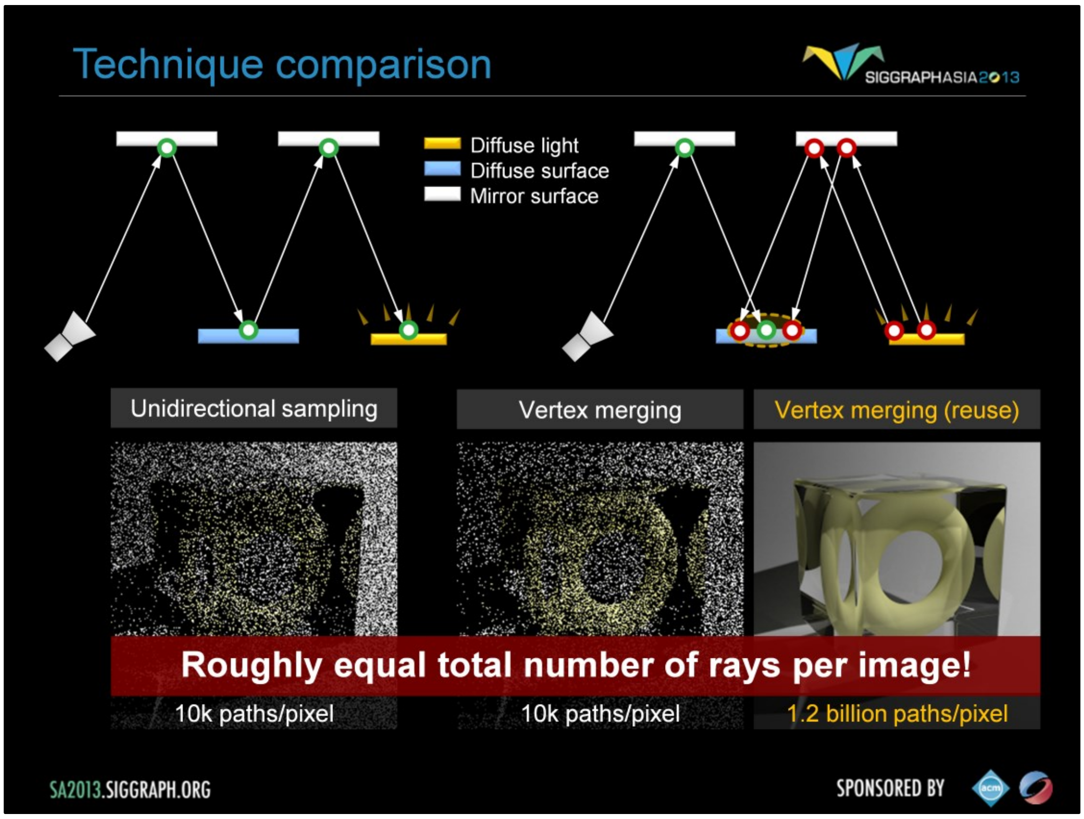
However, the strength of photon mapping is computational efficiency – we can very efficiently reuse the light subpaths traced for *all* pixels at the cost of a single range search query. This allows us to quickly construct orders of magnitude more light transport estimators from the same sampling data, 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.



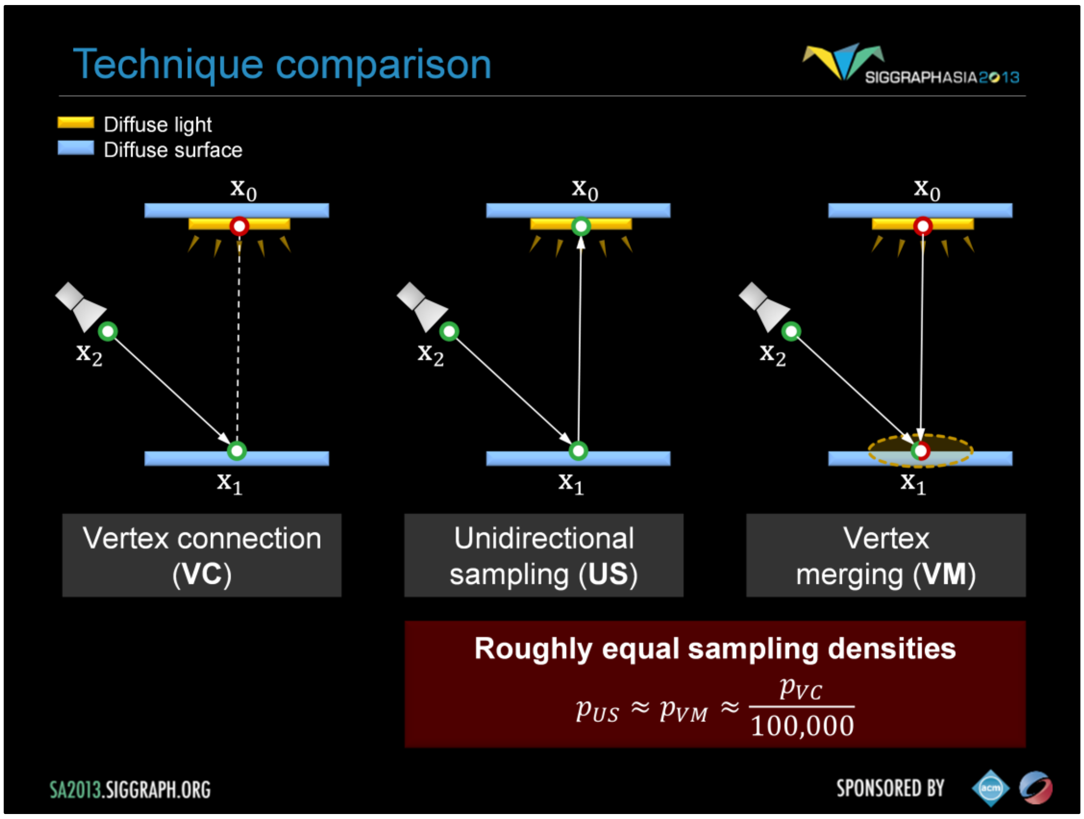
Technique comparison

However, the strength of photon mapping 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.



Technique comparison

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.



Technique comparison

Vertex connection (**VC**)

Unidirectional sampling (**US**)

Vertex merging (**VM**)

Roughly equal sampling densities

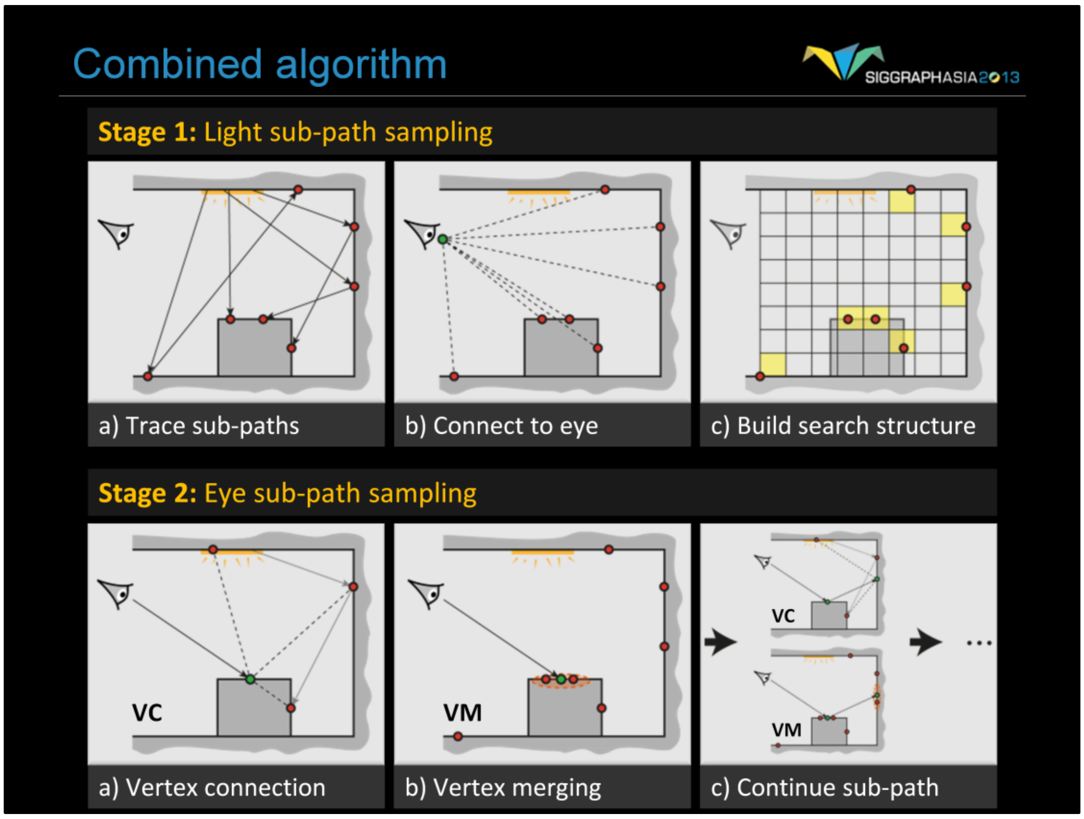
$$p_{US} \approx p_{VM} \approx \frac{p_{VC}}{100{,}000}$$

Now let us 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, photon mapping gains a lot of efficiency over unidirectional sampling.

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



Combined algorithm

SIGGRAPHASIA2013

Stage 1: Light sub-path sampling

a) Trace sub-paths

b) Connect to eye

c) Build search structure

Stage 2: Eye sub-path sampling

a) Vertex connection

b) Vertex merging

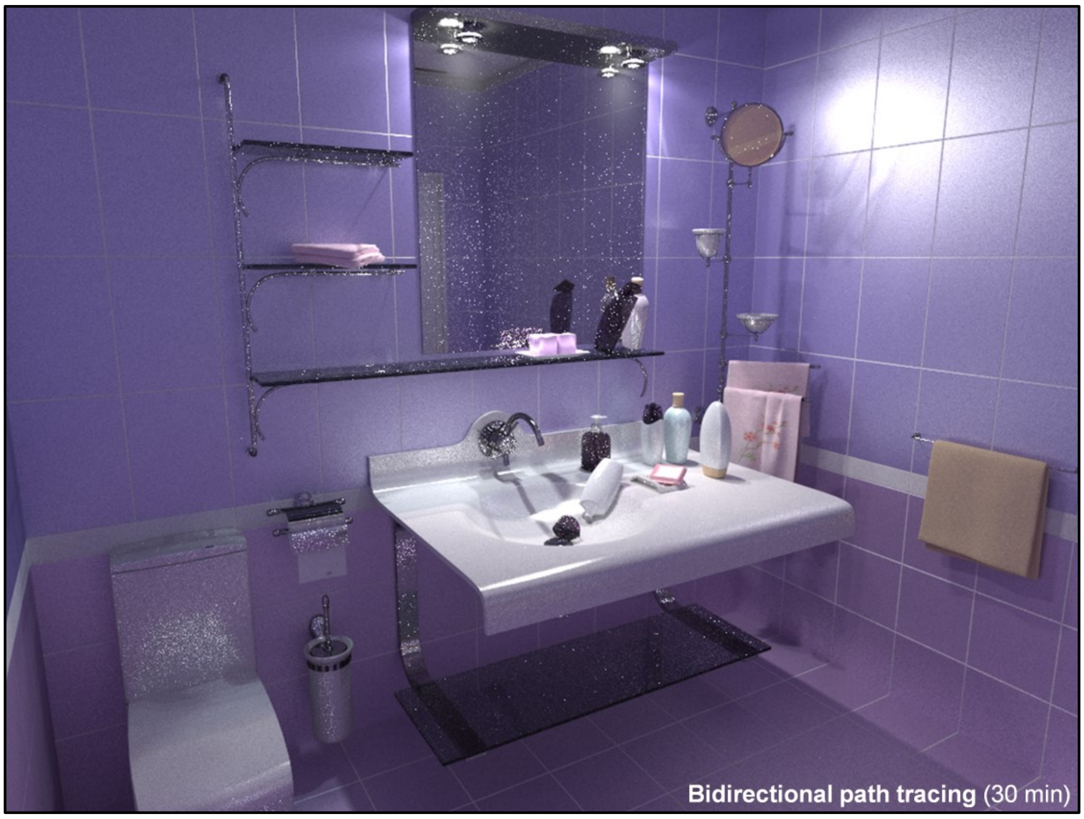
c) Continue sub-path

Even more importantly, 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 operates in two stages.

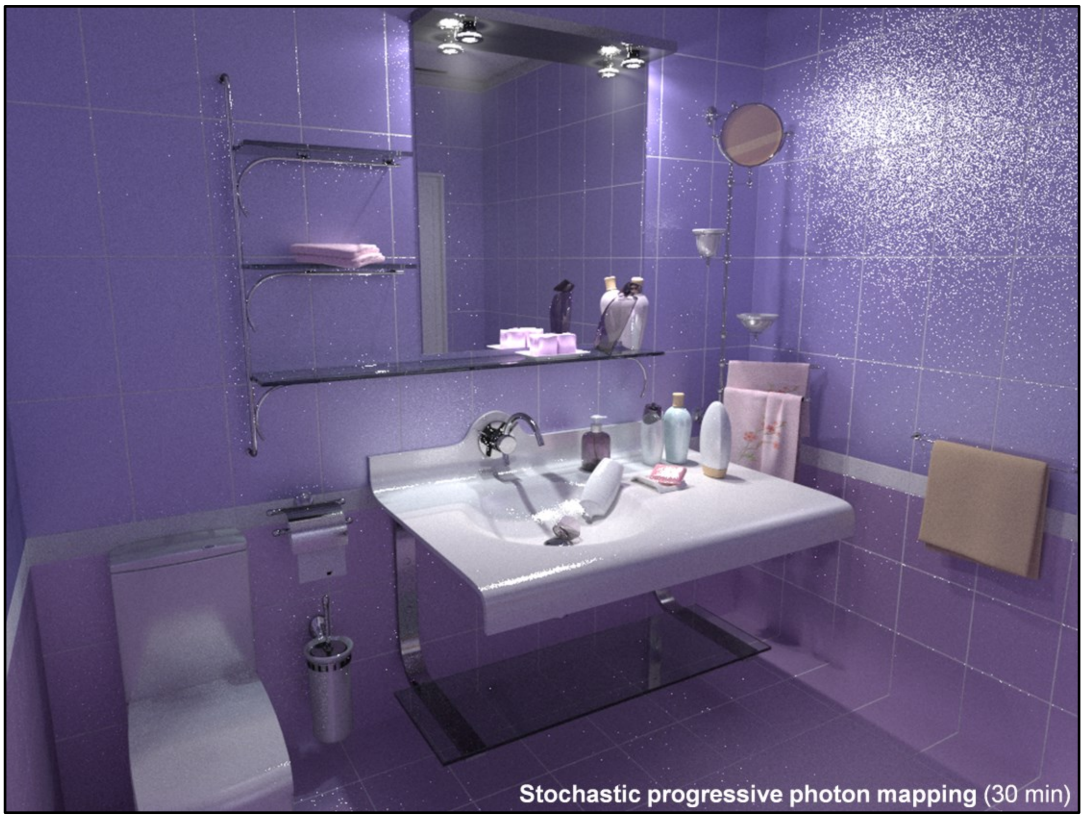
1. In the first stage, we
   a) trace the light subpaths for all pixels,
   b) connect them to the camera, and
   c) store them in a range search acceleration data structure (e.g. a kd-tree or a hashed grid).

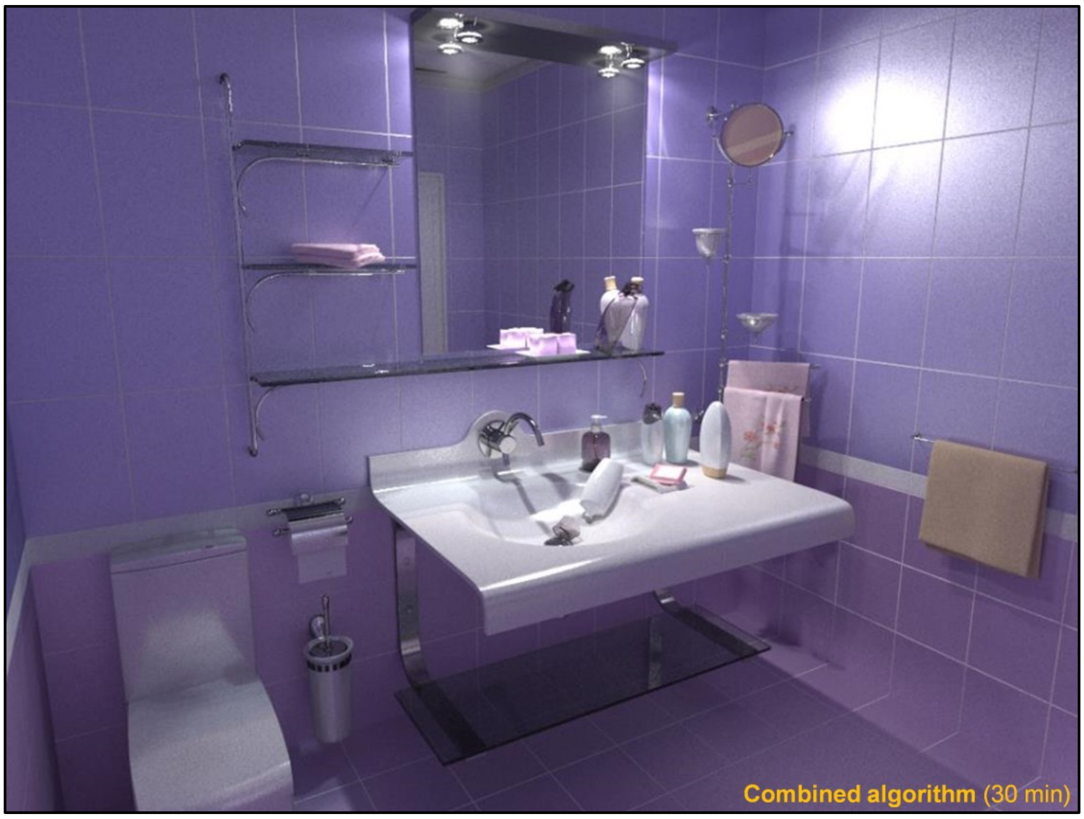
2. In the second stage, we trace a camera subpath for every pixel.
   a) Each sampled vertex on this path is connected to a light source (a.k.a. next event estimation), connected to the vertices of the light subpath corresponding to that pixel, and
   b) merged with the vertices of *all* light subpaths.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

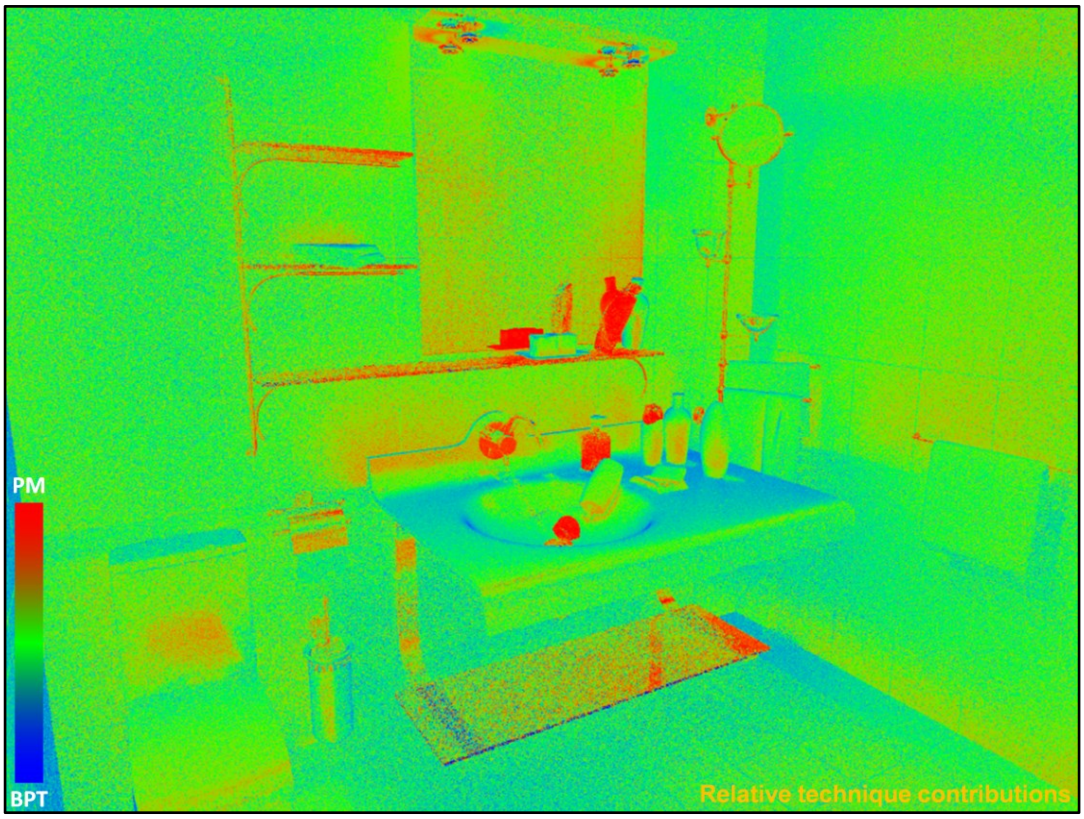
In a progressive rendering setup, we perform these steps at each rendering iteration, reducing the vertex merging radius.



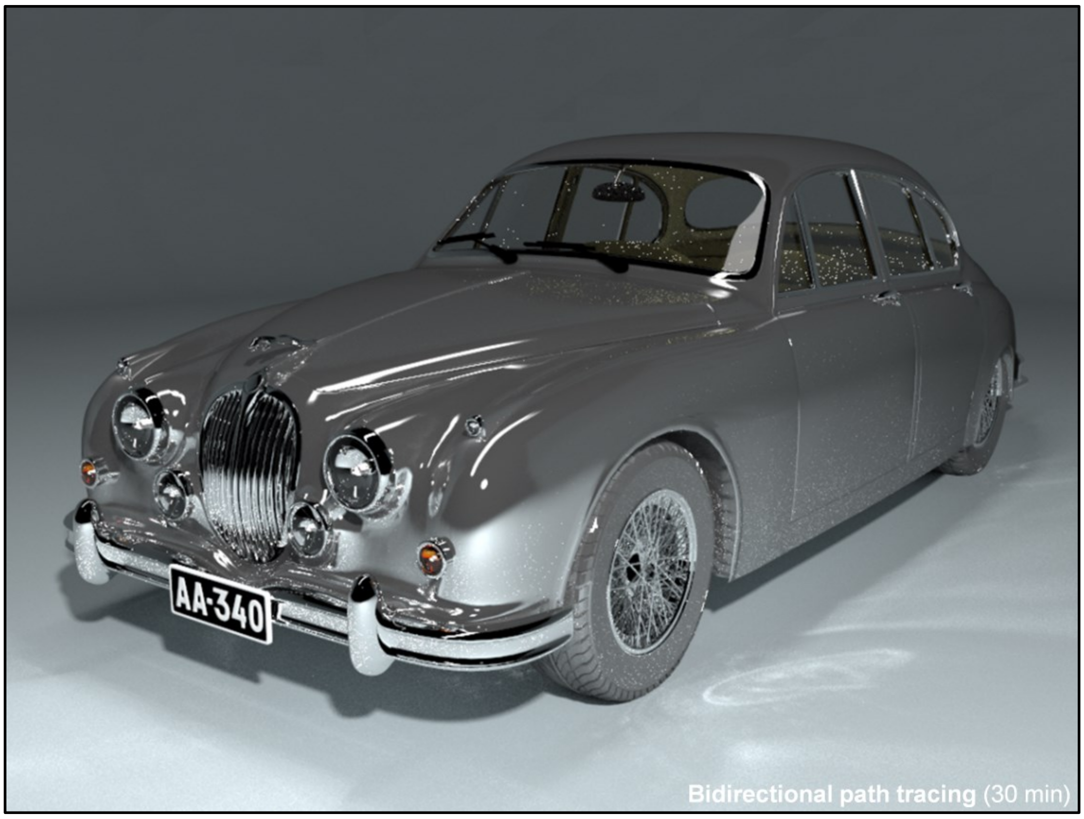
Let us now see how this combined algorithm stacks up against bidirectional path tracing and stochastic progressive photon mapping on a number of scenes with complex illumination.

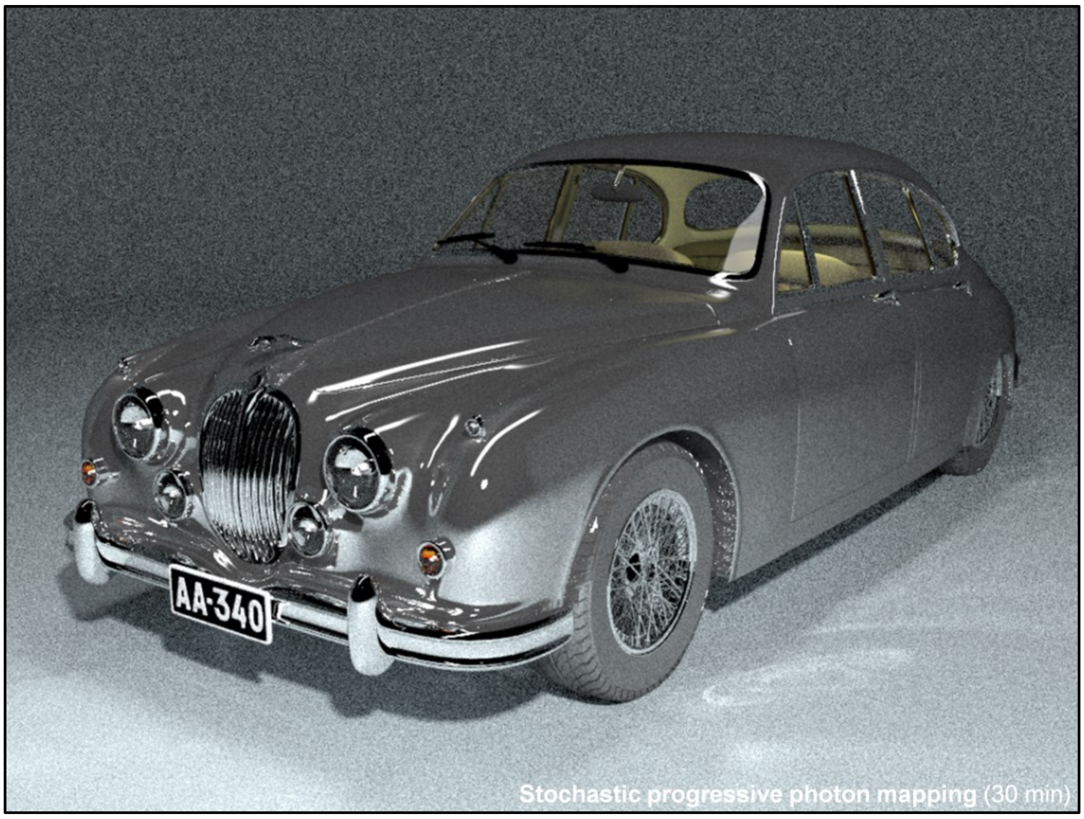

Stochastic progressive photon mapping (30 min)

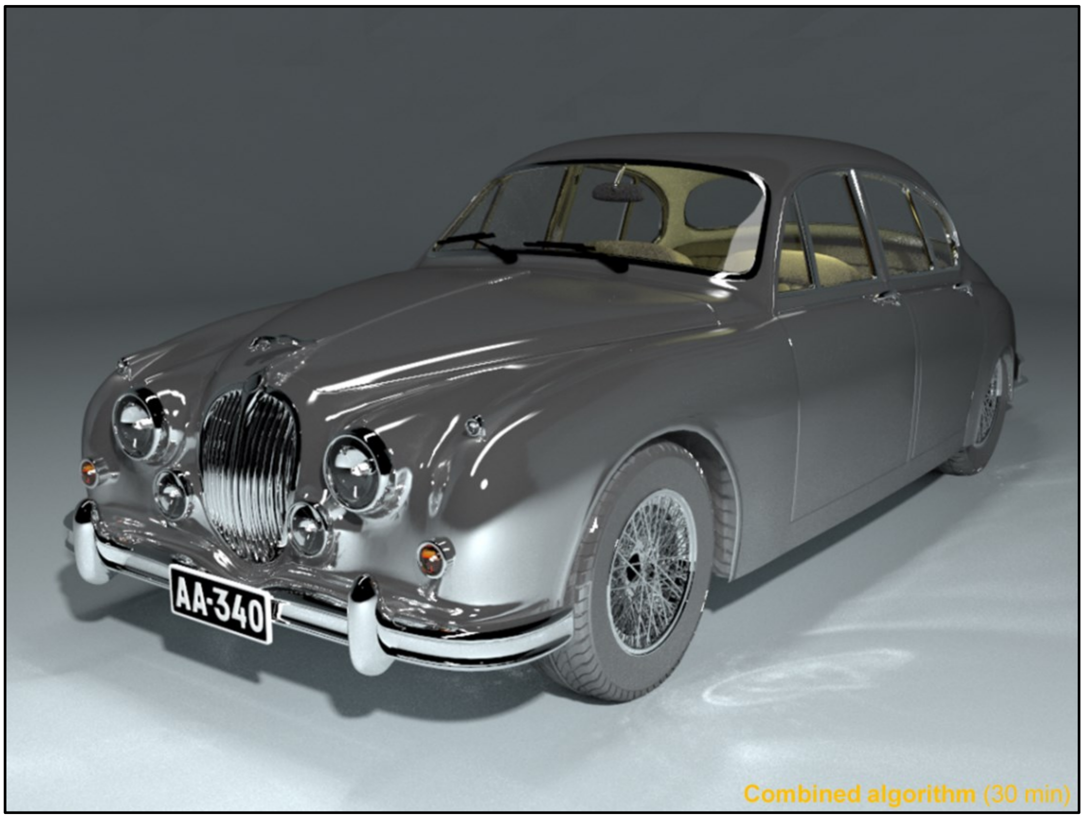

Combined algorithm (30 min)

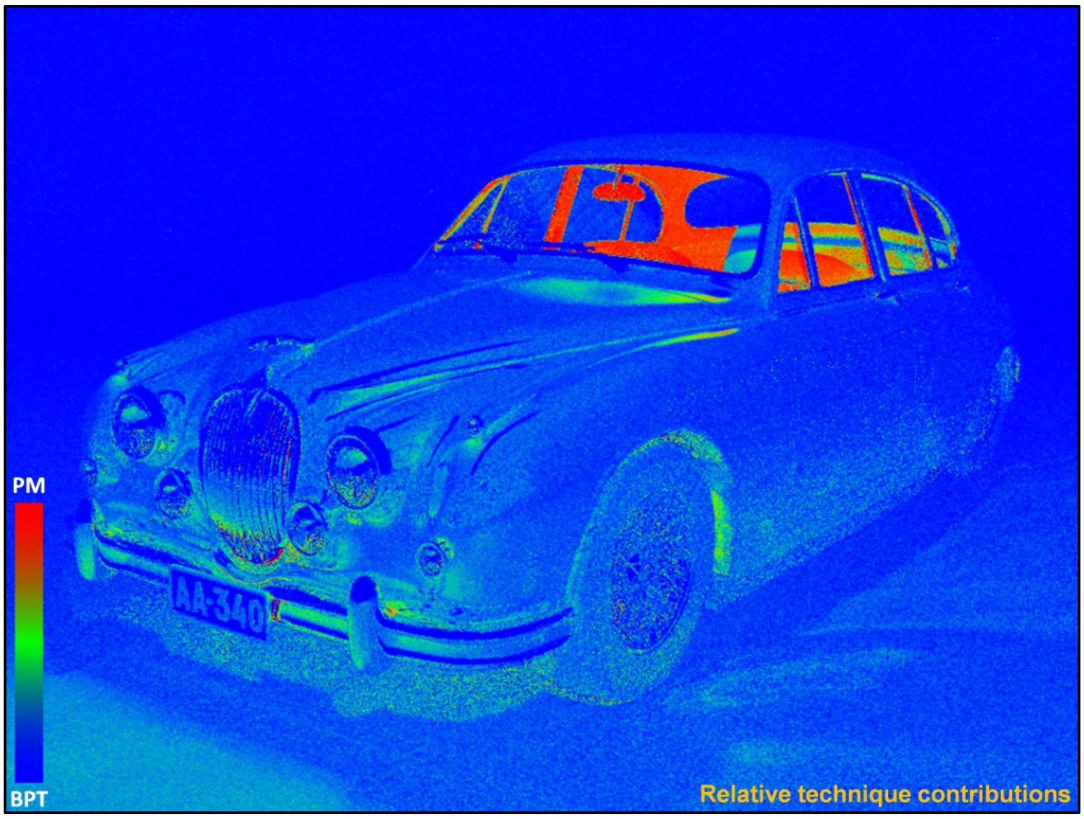


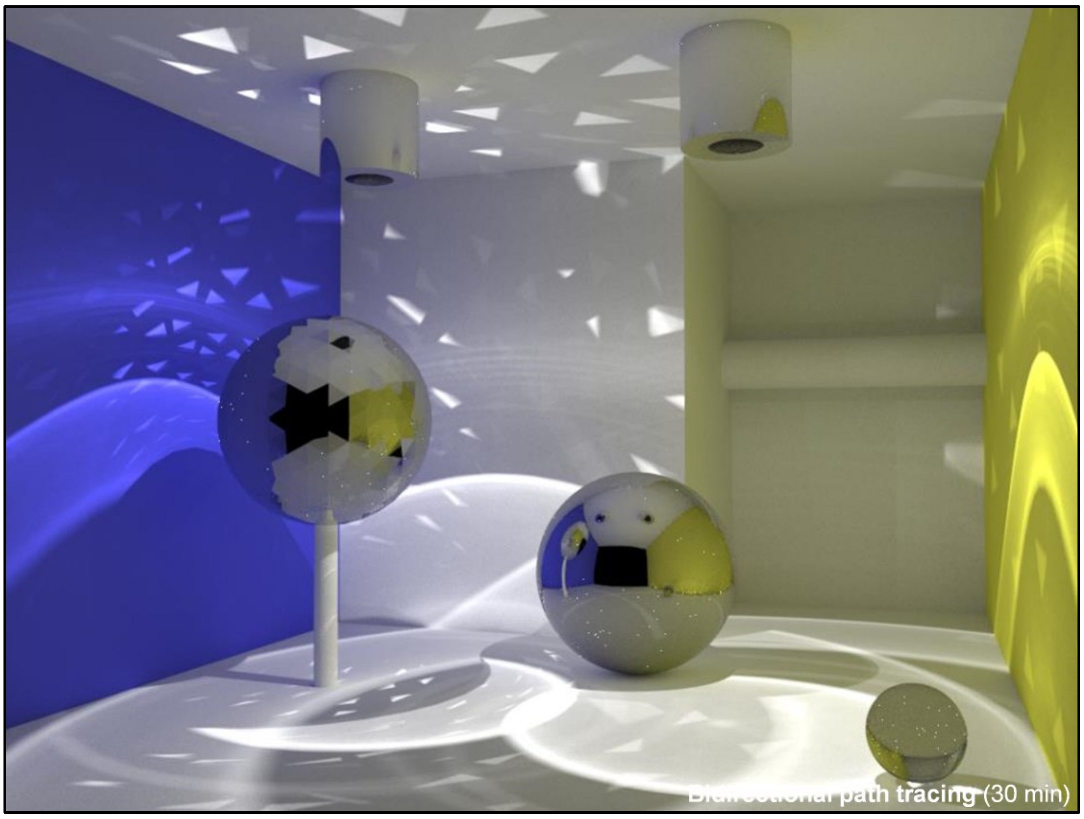
Here, we visualize the relative contributions of BPT and PM techniques to the combined image from the previous slide. This directly corresponds to the weights that the combined algorithm assigned to these techniques.

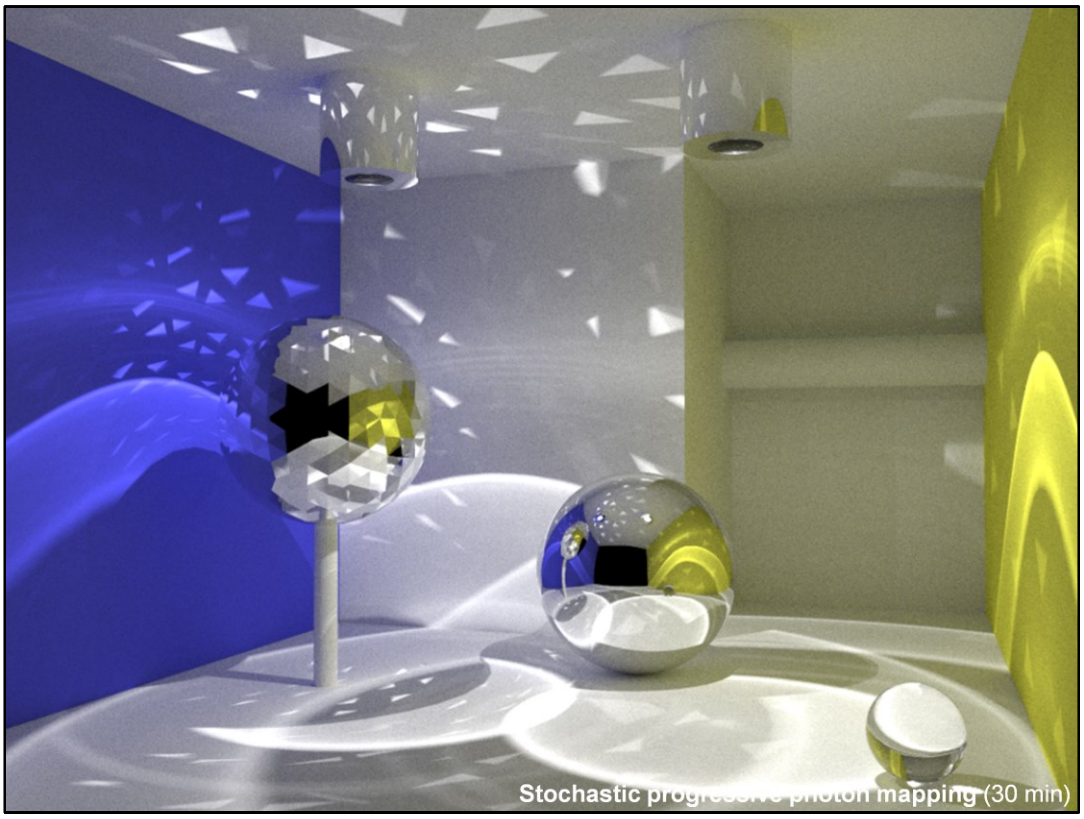

AA-340
Bidirectional path tracing (30 min)

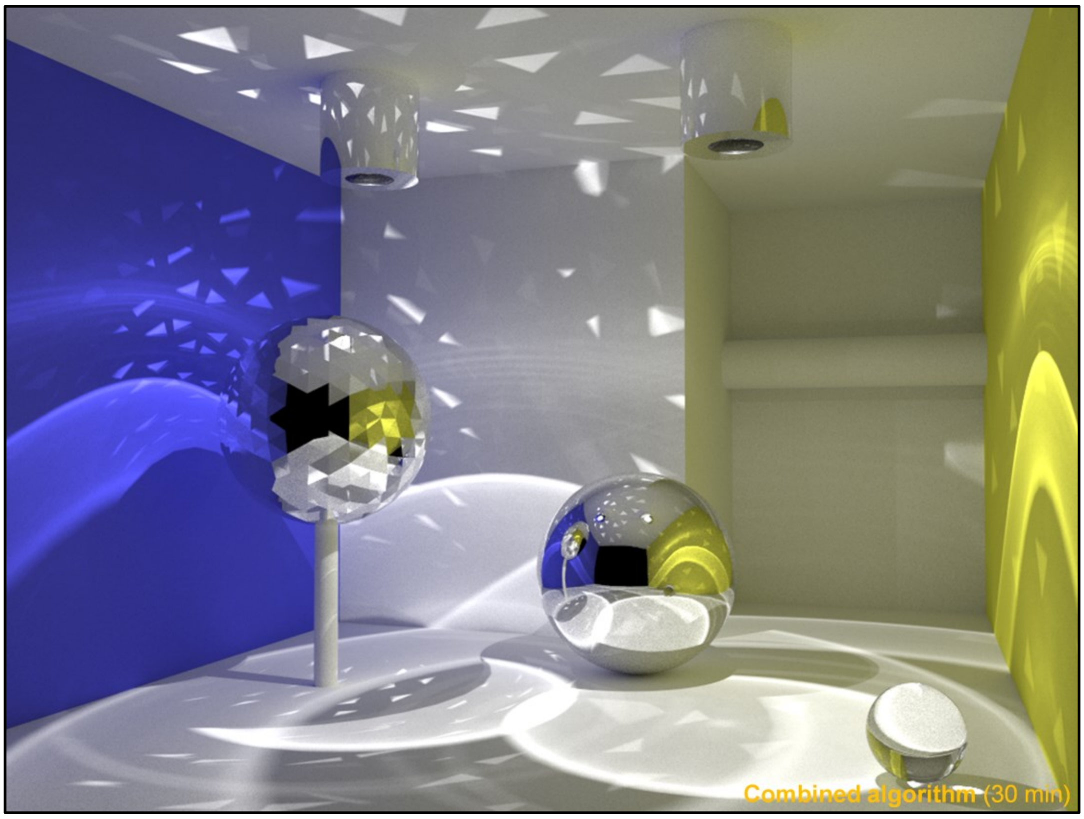

AA-340
Stochastic progressive photon mapping (30 min)

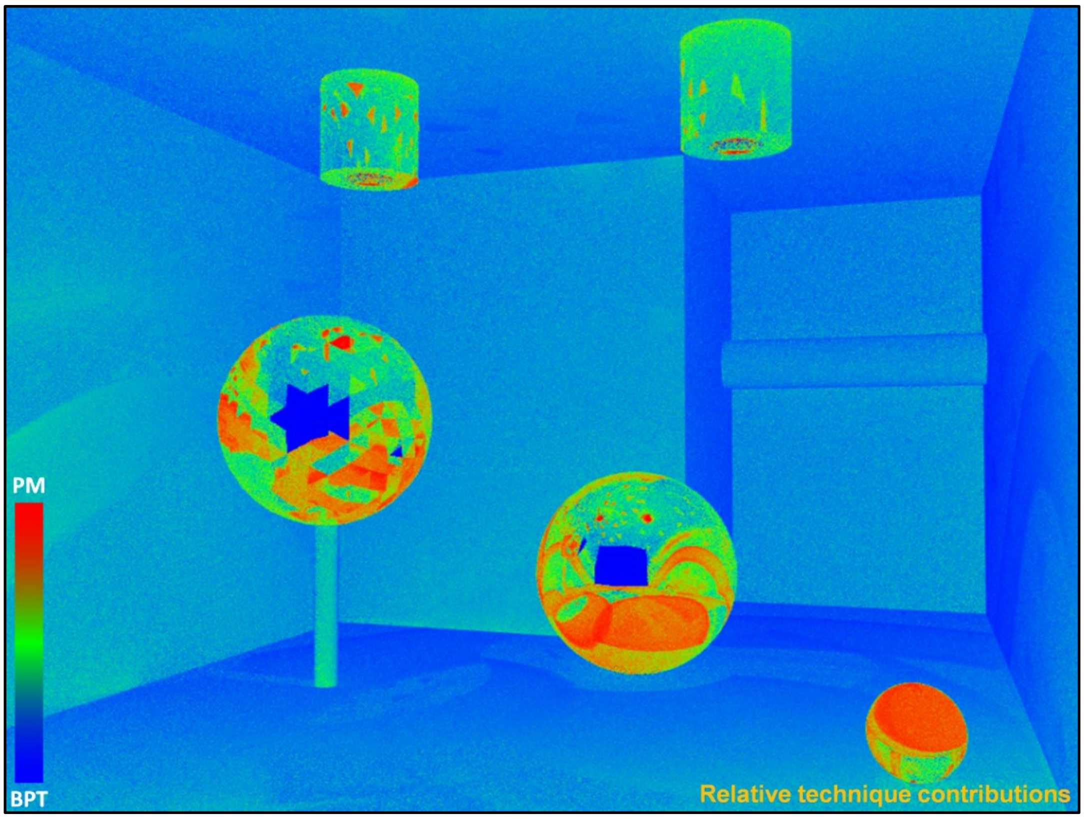

AA-340
Combined algorithm (30 min)

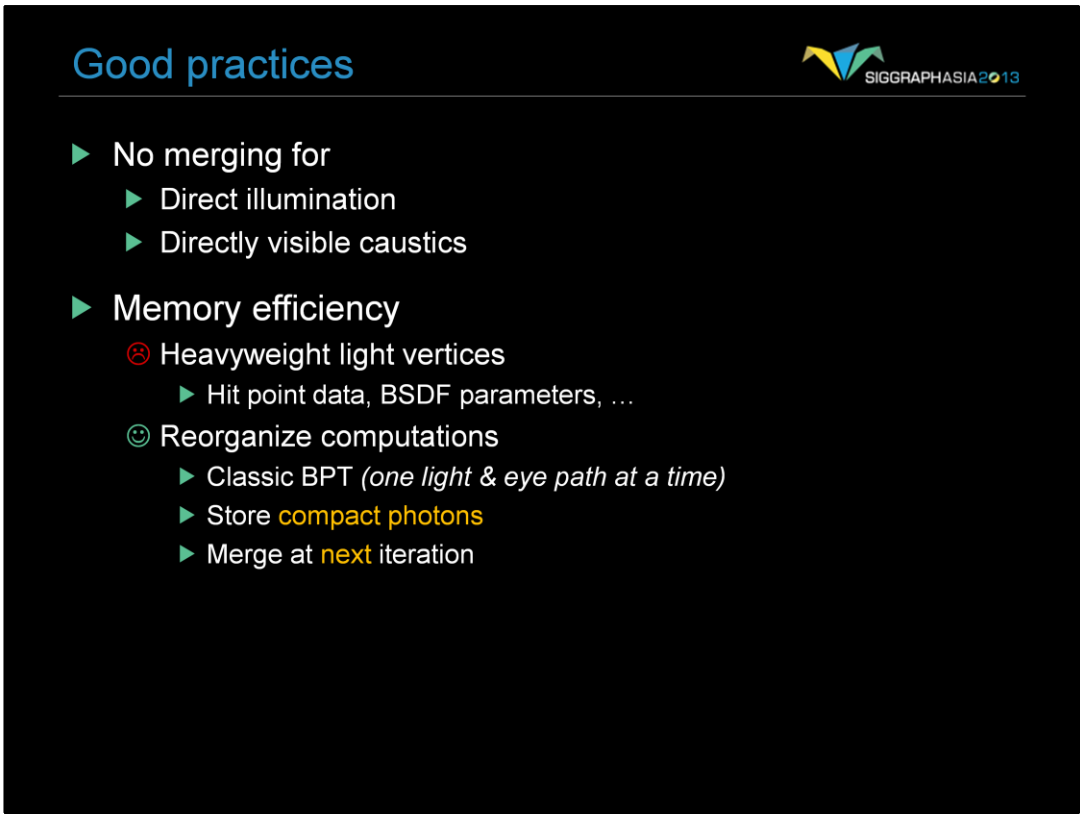

PM
BPT
AA-340
Relative technique contributions


Bidirectional path tracing (30 min)


Stochastic progressive photon mapping (30 min)


Combined algorithm (30 min)


PM
BPT
Relative technique contributions



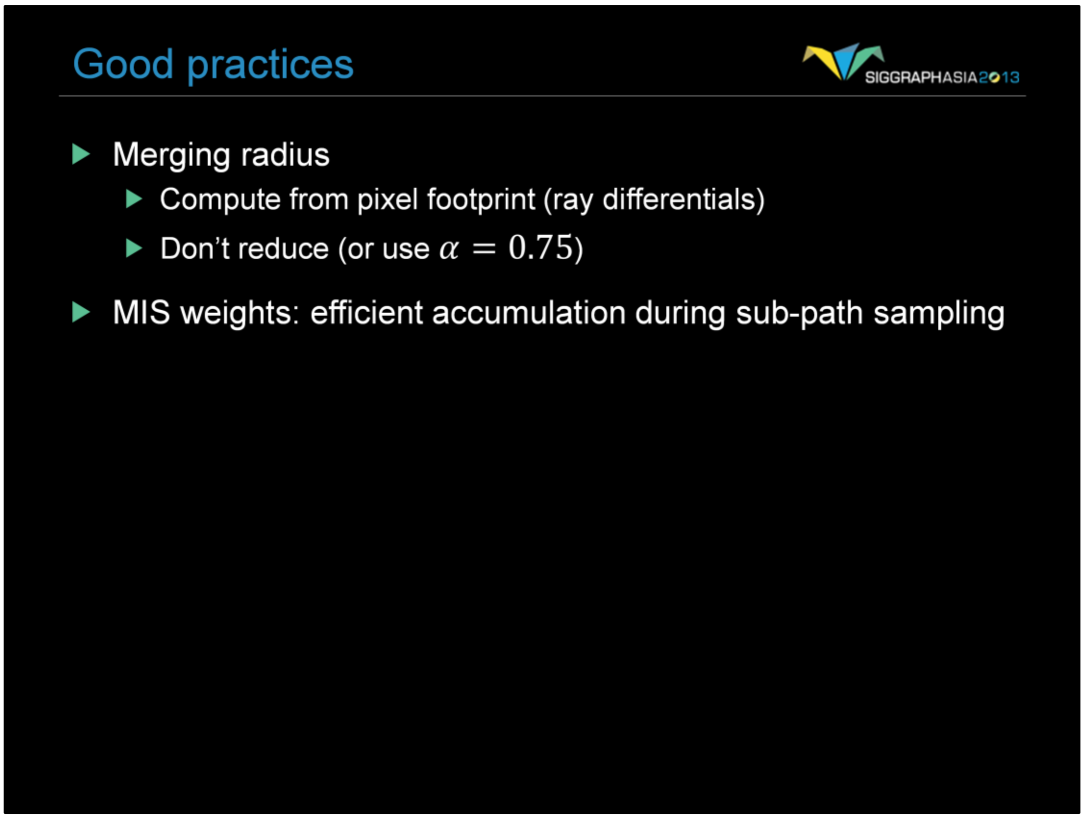
Good practices

- No merging for
  - Direct illumination
  - Directly visible caustics
- Memory efficiency
  - ☹ Heavyweight light vertices
    - Hit point data, BSDF parameters, …
  - ☺ Reorganize computations
    - Classic BPT *(one light & eye path at a time)*
    - Store compact photons
    - Merge at next iteration

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

We can practically 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 algorithm has the problem that its memory footprint can be much larger than that of (progressive) photon mapping. This 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 the combined algorithm, 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.



Good practices

SIGGRAPHASIA2013

- Merging radius
  - Compute from pixel footprint (ray differentials)
  - Don't reduce (or use $\alpha = 0.75$)
- MIS weights: efficient accumulation during sub-path sampling

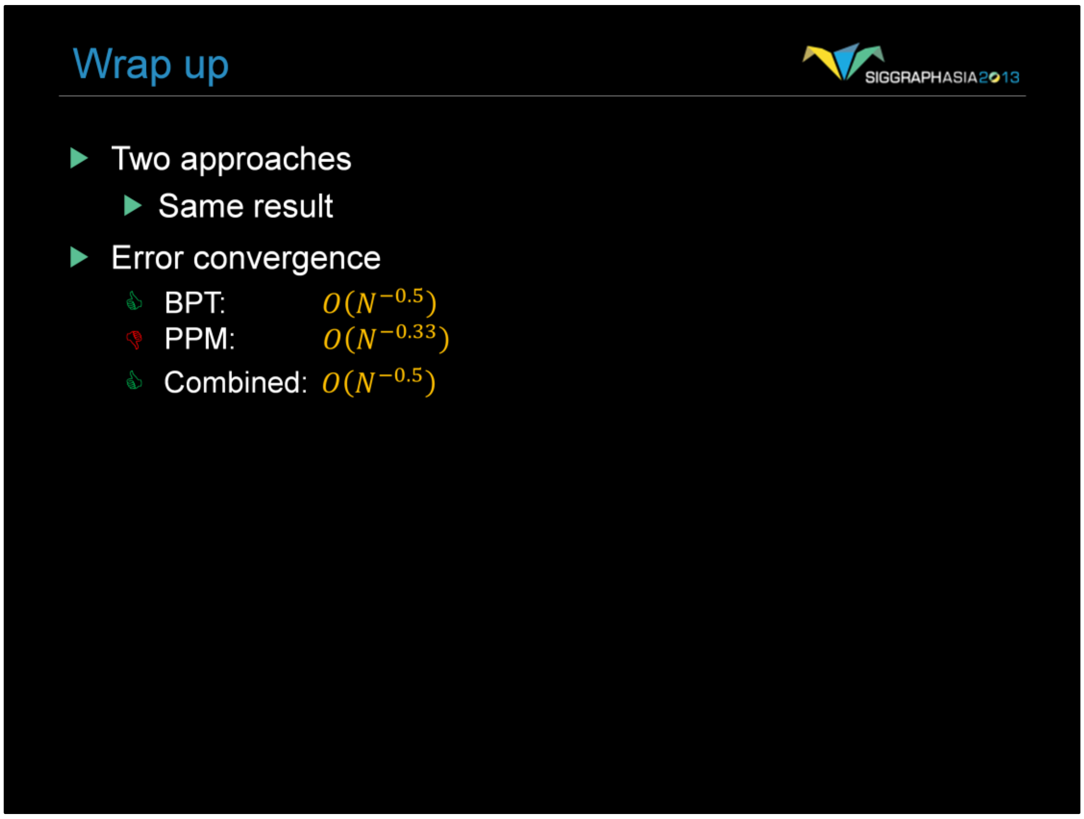
The progressive combined algorithm has two parameters: the initial merging radius r and its reduction rate α (see "Progressive Photon Mapping" [Hachisuka et al. 2008]). We can often afford to use a much smaller radius than in PPM, as VCM mixes a large number of path sampling techniques, and VM is mostly 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 the combined algorithms, 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.

Additional material

- **A path space extension for robust light transport simulation**
  [Hachisuka et al. 2012]
  - Paper, supplemental analysis *[http://cs.au.dk/~toshiya/]*

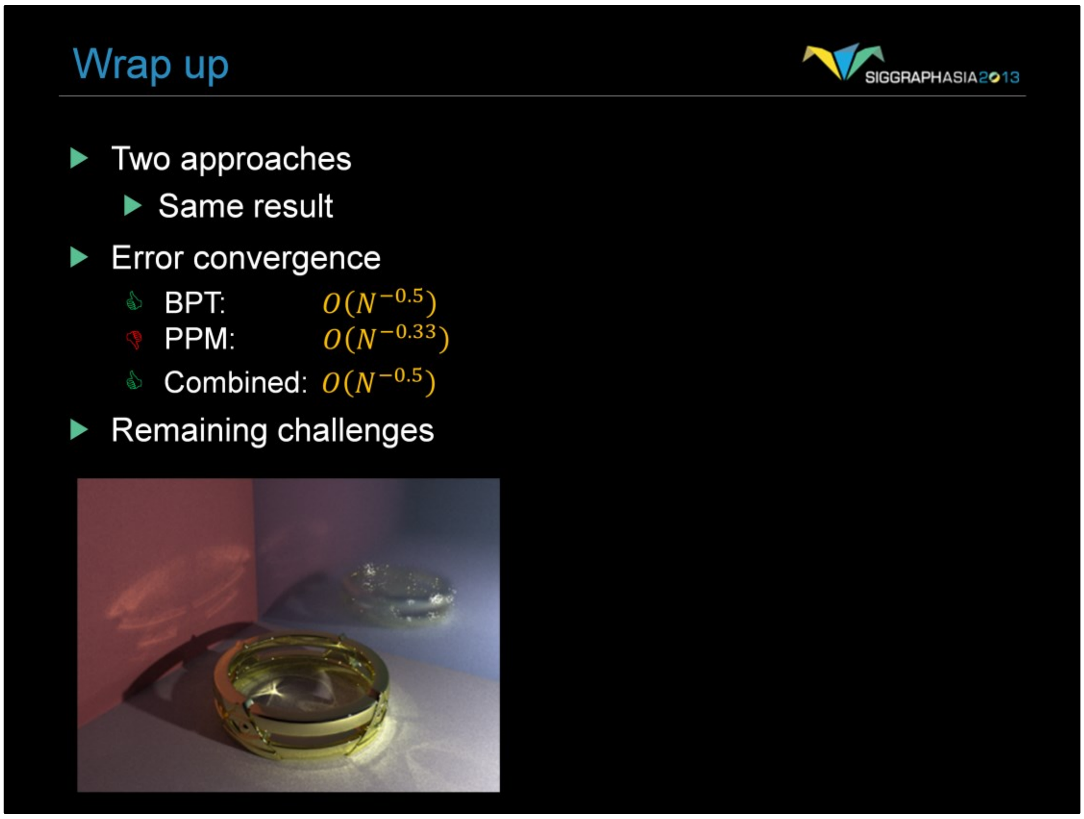
- **Light transport simulation with vertex connection & merging**
  [Georgiev et al. 2012]
  - Paper, tech. report, image comparisons *[http://www.iliyan.com]*



Wrap up

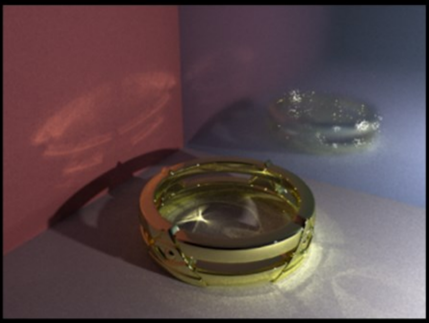
- Two approaches
  - Same result
- Error convergence
  - 👍 BPT: $O(N^{-0.5})$
  - 👎 PPM: $O(N^{-0.33})$
  - 👍 Combined: $O(N^{-0.5})$

Two independent groups of researchers have taken different approaches on efficiently combining BPT and PM. The result of both these efforts is the same algorithm, which importantly also inherits the higher asymptotic error convergence rate 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 [Georgiev et al. 2012].

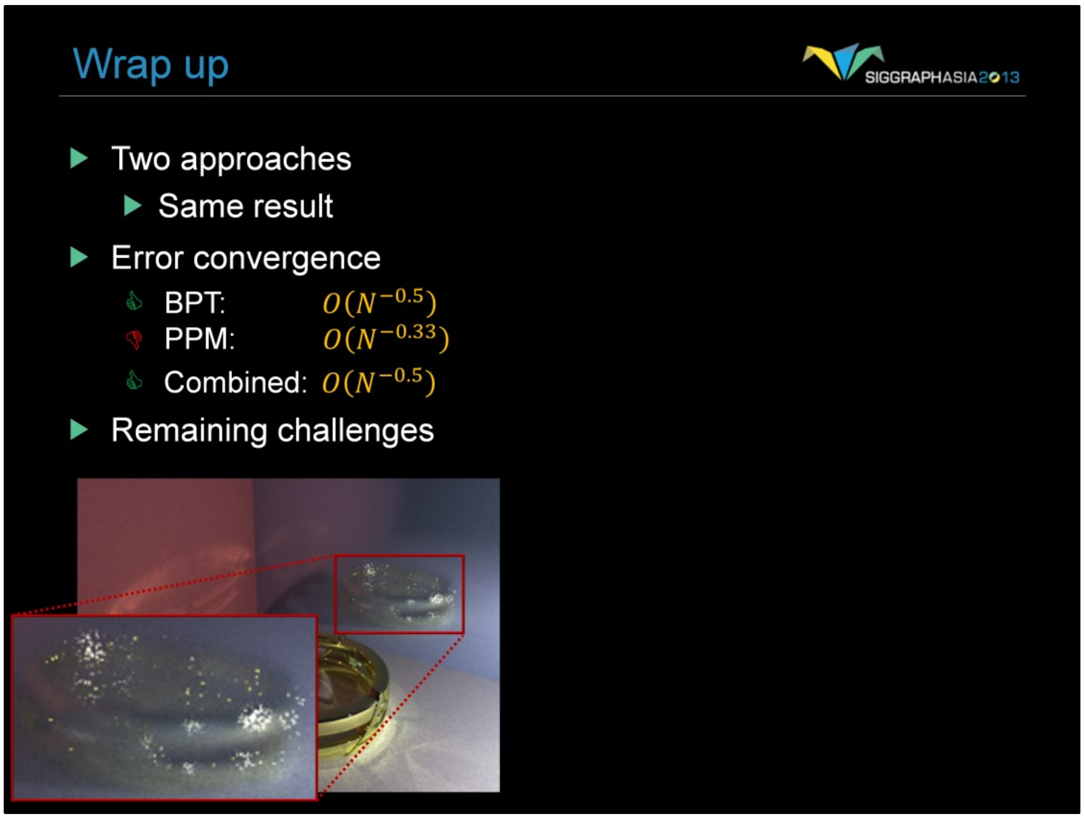


Wrap up

- Two approaches
  - Same result
- Error convergence
  - 👍 BPT: $O(N^{-0.5})$
  - 👎 PPM: $O(N^{-0.33})$
  - 👍 Combined: $O(N^{-0.5})$
- Remaining challenges



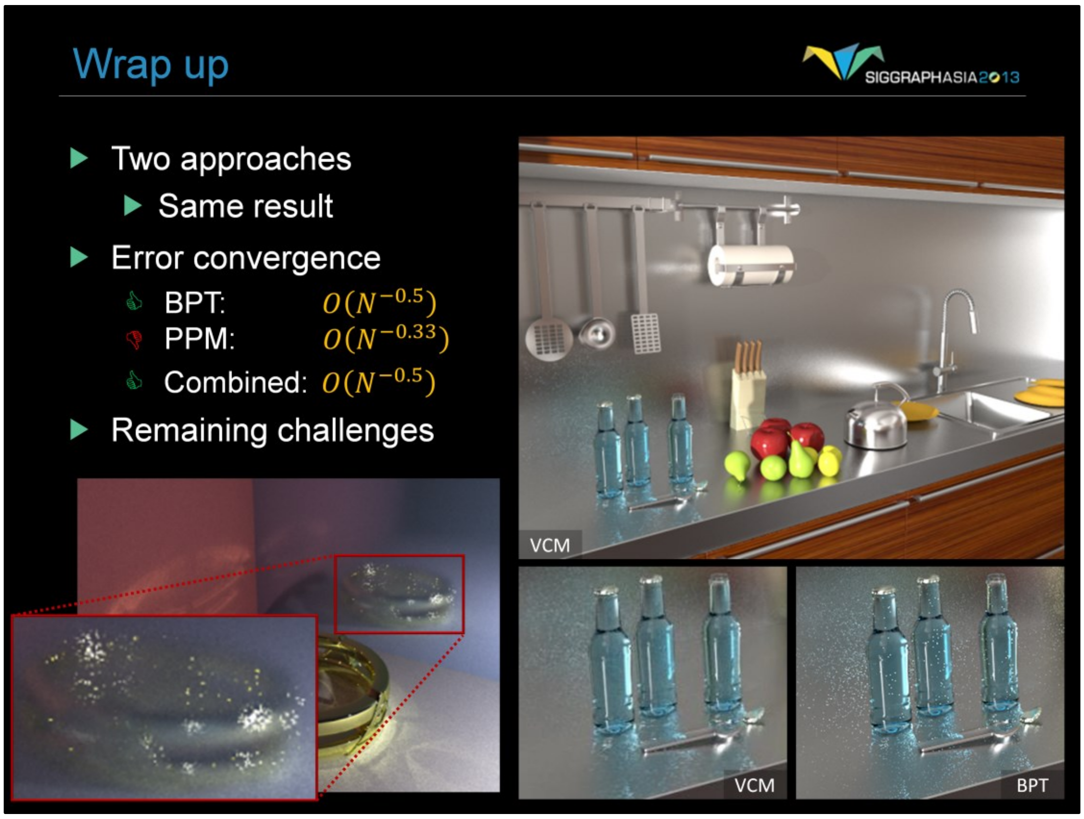
Even though VCM is a step forward in Monte Carlo rendering and has proven very useful in practice, it doesn't come without limitations. Specifically, it cannot handle more efficiently those types of light transport paths that are difficult for both BPT and PM to sample.



Wrap up

- Two approaches
  - Same result
- Error convergence
  - 👍 BPT: $O(N^{-0.5})$
  - 👎 PPM: $O(N^{-0.33})$
  - 👍 Combined: $O(N^{-0.5})$
- Remaining challenges

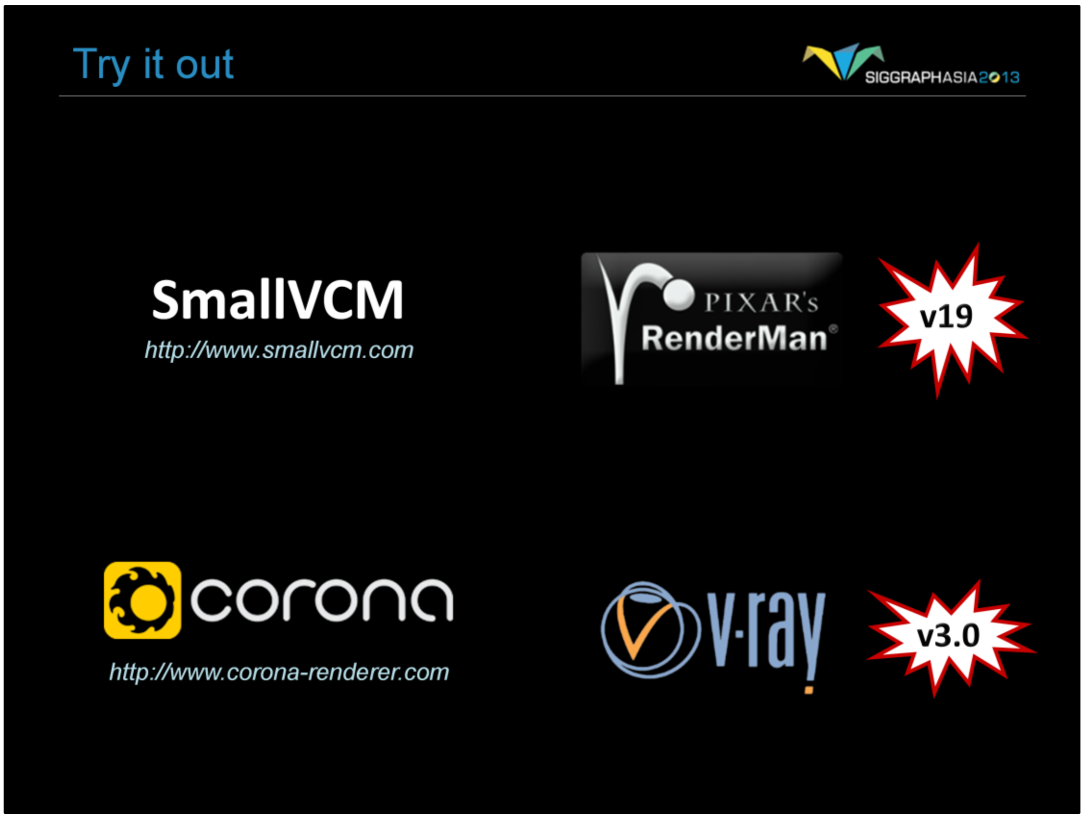
A prominent example are caustics falling on a glossy surface.



Wrap up

- Two approaches
  - Same result
- Error convergence
  - 👍 BPT: $O(N^{-0.5})$
  - 👎 PPM: $O(N^{-0.33})$
  - 👍 Combined: $O(N^{-0.5})$
- Remaining challenges

And on this kitchen scene, even though VCM brings practical improvement over BPT, there is still a lot to be desired from the caustics.



Try it out

SIGGRAPHASIA2013

SmallVCM
http://www.smallvcm.com

PIXAR's RenderMan® v19

corona
http://www.corona-renderer.com

V-Ray v3.0

If you want to try this algorithm yourself, we have released a reference open source implementation in the SmallVCM renderer.

A number of companies have also announced integration of the algorithm in the upcoming releases of their commercial renderers. One example is Pixar's Photorealistic RenderMan v19, and Chaos Group have already shown first images from V-Ray 3.0. The algorithm is also already implemented in the public Alpha release of Corona renderer.