Fusing State Spaces for Markov Chain Monte Carlo Rendering

HISANARI OTSU, The University of Tokyo
ANTON S. KAPLANYAN, NVIDIA
JOHANNES HANIKA, Karlsruhe Institute of Technology
CARSTEN DACHSBACHER, Karlsruhe Institute of Technology
TOSHIYA HACHISUKA, The University of Tokyo

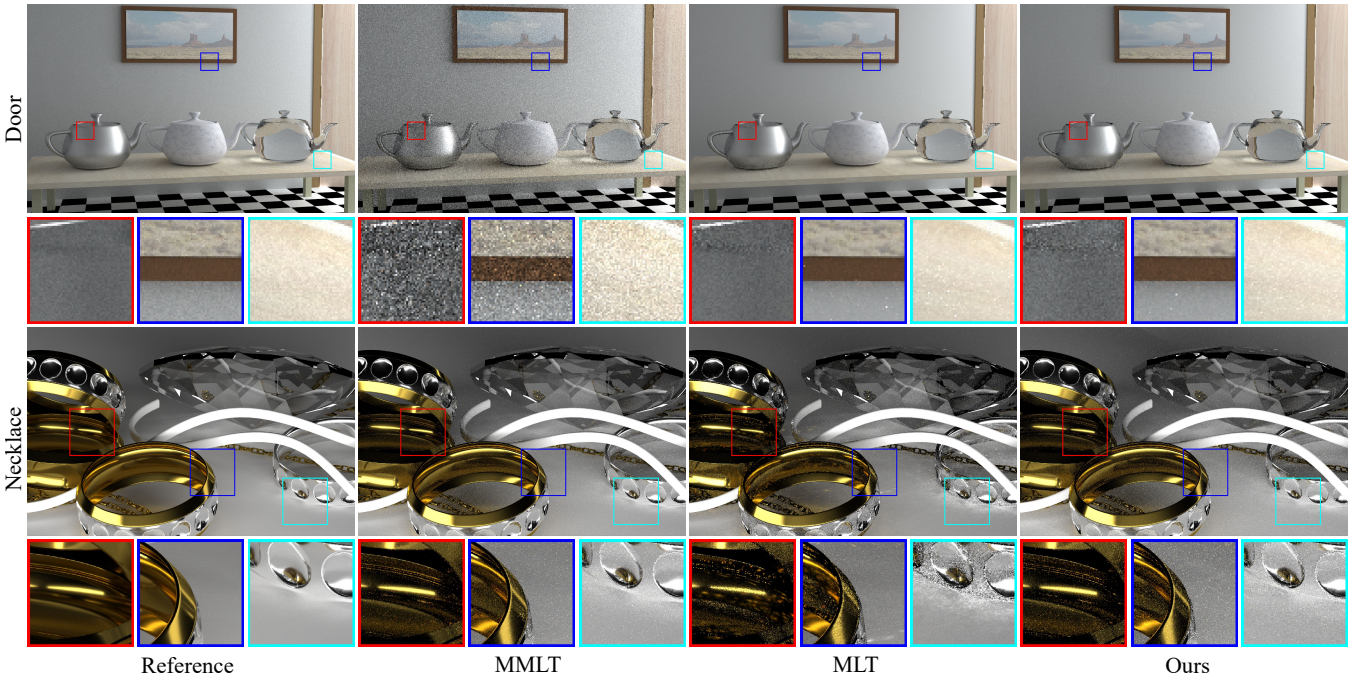

Fig. 1. Equal-time rendering (20 minutes) of two scenes. The door scene contains glossy and specular materials and is illuminated by indirect lights with difficult visibility. This scene can be effectively rendered with MLT [Veach and Guibas 1997], but exhibits suboptimal performance with multiplexed MLT (MMLT) [Hachisuka et al. 2014]. On the other hand, the necklace scene, which is characterized by glossy interreflections, shows the opposite behavior. Our framework makes it possible to combine mutation strategies using the two different state spaces of MLT and MMLT, enabling the combination of specialized mutation strategies, and resulting in a general algorithm that works robustly in many cases.

Rendering algorithms using Markov chain Monte Carlo (MCMC) currently build upon two different state spaces. One of them is the path space, where the algorithms operate on the vertices of actual transport paths. The other state space is the primary sample space, where the algorithms operate on sequences of numbers used for generating transport paths. While the two state spaces are related by the sampling procedure of transport paths, all existing MCMC rendering algorithms are designed to work within only one of the state spaces. We propose a first framework which provides a comprehensive connection between the path space and the primary sample space. Using this framework, we can use mutation strategies designed for one space with mutation strategies in the respective other space. As a practical example, we take a combination of manifold exploration and multiplexed Metropolis light transport using our framework. Our results show that the simultaneous use of the two state spaces improves the robustness of MCMC rendering. By combining efficient local exploration in the path space with global jumps in primary sample space, our method achieves more uniform convergence as compared to using only one space.

CCS Concepts: • **Computing methodologies** → **Ray tracing**;

Additional Key Words and Phrases: global illumination, Markov chain Monte Carlo

ACM Reference format:
Hisanari Otsu, Anton S. Kaplanyan, Johannes Hanika, Carsten Dachsbacher, and Toshiya Hachisuka. 2017. Fusing State Spaces for Markov Chain Monte Carlo Rendering. *ACM Trans. Graph.* 36, 4, Article 74 (July 2017), 10 pages.
DOI: 0.1145/3072959.3073691

1 INTRODUCTION

Physically-based rendering with light transport simulation is widely used nowadays. One class of simulation algorithms is based on

Markov chain Monte Carlo (MCMC) [Veach and Guibas 1997]. MCMC rendering generates a Markov chain of light transport paths to follow an arbitrary user-defined target distribution. This target distribution is defined according to the contributions of light transport paths to the image, which allows MCMC algorithms to efficiently focus computation on contributing paths. In rendering, a technique for proposing the next state of a Markov chain is typically called a mutation strategy. Recent work has proposed a variety of sophisticated mutation strategies [Hachisuka et al. 2014; Jakob and Marschner 2012; Kaplanyan et al. 2014; Li et al. 2015].

The MCMC rendering algorithms use either the path space [Veach and Guibas 1997] or the primary sample space [Kelemen et al. 2002] as the state space of a Markov chain. The path space defines a path by a sequence of vertices, and the algorithms mutate a path by *directly* modifying its vertices. The primary sample space instead defines a path by a sequence of numbers used to generate the path, and *indirectly* mutates a path by modifying the corresponding sequence of numbers. This sequence corresponds to numbers generated by a pseudo-random number generator in regular Monte Carlo path samplers, and can be mapped to a path by path tracing [Kajiya 1986] or bidirectional path tracing [Lafortune and Willems 1993; Veach and Guibas 1994] for example. Due to this fundamental difference between the two state spaces, a mutation strategy for one space is not applicable to the other space. For example, it is impossible to use manifold exploration [Jakob and Marschner 2012] in the primary sample space as it is designed to work on path vertices in the path space. This situation prohibits us to take full advantage of all the advanced mutation strategies within a single rendering algorithm.

We propose a framework to *fuse* the different state spaces in MCMC rendering for the first time. The main idea is the use of an *inverse path sampler* which acts as the inverse operation of a regular path sampler. The inverse path sampler takes a complete path as input and maps it to a corresponding sequence of numbers in a unit hypercube. Since this mapping uses the inverse of the inverse cumulative distribution functions (CDFs) used in importance sampling, it requires just the CDFs themselves which are often readily available. Our formulation does not impose any modification to mutation strategies themselves, making it easy to use it with existing implementations. We tested our formulation for the combination of manifold exploration [Jakob and Marschner 2012] and multiplexed Metropolis light transport [Hachisuka et al. 2014]. The results demonstrate that this combination robustly handles scenes with different characteristics where using only one of the algorithms fails. To summarize, our contributions are:

- A novel framework which fuses the two state spaces currently used for Markov chain Monte Carlo rendering,
- Introduction of the concept of an inverse path sampler,
- Demonstration of MCMC rendering with a combination of mutation strategies mixing both state spaces.

2 RELATED WORK

Light Transport Simulation. Rendering algorithms based on Monte Carlo integration are primarily characterized by how they generate paths connecting a light source to a sensor. Path tracing [Kajiya 1986] generates a path starting from the sensor, light tracing [Arvo 1986] traces from a light source, and bidirectional path tracing [Lafortune and Willems 1993; Veach and Guibas 1994] from both sides with deterministic connections of subpaths.

Another family of approaches is based on *photon density estimation* [Shirley et al. 1995], such as (progressive) photon mapping [Hachisuka et al. 2008; Jensen 1996], which estimates illumination using the density of light subpath vertices. Recent work [Georgiev et al. 2012; Hachisuka et al. 2012] combines Monte Carlo integration and photon density estimation into a single rendering algorithm.

We employ existing Monte Carlo integration approaches as *path samplers*, which can generate a light transport path from a sequence of numbers. A path sampler in our formulation can be any of the existing approaches as long as they are based on the path integral formulation [Veach 1998]. We introduce the inverse of such a path sampler and show how it can be used in Markov chain Monte Carlo (MCMC) rendering with fused state spaces.

MCMC in Path Space. Veach and Guibas [1997] introduced Markov chain Monte Carlo methods to rendering. The resulting algorithm, Metropolis Light Transport (MLT), perturbs the vertices of a path and generates a history of paths based on the Metropolis-Hastings algorithm. Since MLT directly manipulates vertices, it works within the path space of the path integral formulation.

Manifold exploration [Jakob and Marschner 2012] is also built on the original MLT framework and extends its mutation strategy to efficiently handle a chain of specular and highly glossy events. Half vector space light transport [Hanika et al. 2015; Kaplanyan et al. 2014] represents paths by their endpoints and half vectors at the interactions between them. This representation has been shown to flatten the target sampling distribution, which makes it easier to sample by Markov chain Monte Carlo algorithms.

Path space algorithms are often efficient at rendering certain effects since we can explicitly consider characteristics of such effects in mutations (e.g., caustics in the caustic mutation [Veach and Guibas 1997]). We show how to incorporate such efficient mutations into the other class of MCMC rendering algorithms.

MCMC in Primary Sample Space. Kelemen et al. [2002] introduced an alternative formulation of MLT based on the primary sample space. The algorithm, primary sample space MLT (PSSMLT), indirectly mutates paths by perturbing a vector of (random) numbers that is used to generate paths. They showed how this formulation significantly simplifies the MCMC process and flattens the target distribution by utilizing the information of the probability density function of a given path sampler. Li et al. [2015] showed how to achieve locally adaptive anisotropic mutations in the primary sample space based on the approximation of Hessian-Hamiltonian dynamics. The resulting algorithm demonstrates robust sampling even in the presence of complex light transport paths.

If PSSMLT is used with bidirectional path tracing as a path sampler, it generates a family of bidirectional paths, rather than a single path such as MLT. This difference makes the connection between the primary sample space and the path space ambiguous since one sample in the primary sample space corresponds to a family of paths. Hachisuka et al. [2014] showed that this ambiguity can be resolved simply by extending the primary sample space by another dimension, encoding the type of the used bidirectional technique.

They showed how this multiplexed primary sample space leads to a method that can distribute samples based on the weighted primary sample spaces according to multiple importance sampling [Veach and Guibas 1994]. We show how to combine the path space MLT techniques and the primary sample space MLT techniques for the first time.

Bridging Sampling Spaces. The half vector space [Kaplanyan et al. 2014] can be considered as yet another space and one possibility to bridge the half vector space and the path space. We instead consider bridging the primary sample space and the path space. Concurrent works [Bitterli et al. 2017; Pantaleoni 2017] also use an inverse mapping from the primary sample space to the path space. Although their methods are different from ours, the underlying concept is equivalent.

3 STATE SPACES OF MCMC RENDERING

This section briefly introduces the path space and primary sample space variants of MLT. We also recapitulate the multiplexed primary space, as it has beneficial properties that lead to our framework.

3.1 Path Space

Light transport simulation computes the solution of the *path integral* [Veach 1998]. It determines the intensity of the j-th pixel in the image as the integral over the measurement contribution function $f_j(\bar{x})$ with respect to the product area measure μ:

$$I_j = \int_{\mathcal{P}} f_j(\bar{x}) d\mu(\bar{x}), \tag{1}$$

where $\mathcal{P}$ is the *path space* which comprises all paths of all possible lengths ($\mathcal{P} \equiv \cup_{k=2}^{\infty} \mathcal{P}_k$) and $\mathcal{P}_k$ denotes a sub-space containing paths with $k = [2, \ldots, \infty)$ vertices. An individual *path* $\bar{x} \equiv (\mathbf{x}_1, \ldots, \mathbf{x}_k) \in \mathcal{P}_k$ is defined as a sequence of points on the scene's surfaces. Note that we do not consider participating media in this work.

MLT [Veach and Guibas 1997] uses the Metropolis-Hastings algorithm [Hastings 1970] to sample paths: the path space $\mathcal{P}$ serves as the state space and MLT generates a sequence of samples that follow f^*/b as the target distribution. Here f^* is a scalar contribution function (typically luminance) proportional to the measurement contribution f, and $b \equiv \int_{\mathcal{P}} f^*(\bar{x}) d\mu(\bar{x})$ is the normalization constant, which is estimated with regular Monte Carlo techniques. Using this sequence of samples, the estimate $\hat{I}_j$ becomes

$$I_j \approx \hat{I}_j \equiv \frac{b}{N} \sum_{i=1}^{N} \frac{f(\bar{x}_i)}{f^*(\bar{x}_i)}. \tag{2}$$

A tentative sample $\bar{y}$ is generated based on the current path $\bar{x}_i \in \mathcal{P}$ and according to the transition kernel $T_{\mathcal{P}}$, i.e., $\bar{y} \sim T_{\mathcal{P}}(\bar{x}_i \to \cdot)$. For a Metropolis-Hastings update, the path $\bar{y}$ is accepted as the next state $\bar{x}_{i+1}$ with an *acceptance probability* of $\min(1, a(\bar{x}_i \to \bar{y}))$ where

$$a(\bar{x}_i \to \bar{y}) = \frac{f^*(\bar{y}) T_{\mathcal{P}}(\bar{y} \to \bar{x}_i)}{f^*(\bar{x}_i) T_{\mathcal{P}}(\bar{x}_i \to \bar{y})}. \tag{3}$$

Otherwise $\bar{x}_i$ is kept as the current state ($\bar{x}_{i+1} = \bar{x}_i$).

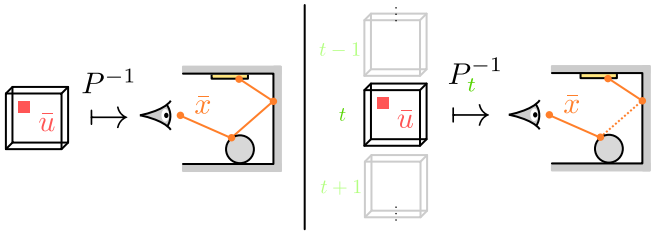

Fig. 2. Relationship between the path space and the primary sample space (left), and the multiplexed primary sample space (right). For primary sample space, the random number $\bar{u}$ is mapped to the path $\bar{x}$ using the inverse cumulative distribution function P^{-1}. Multiplexed primary sample space extends the primary sample space by the index of the sampling strategy t corresponding to the strategy of the bidirectional path samplers. In this example, a path with length three is mapped from $\bar{u}$ with path connection between the second and third vertices ($t = 2$).

3.2 Primary Sample Space

PSSMLT [Kelemen et al. 2002] simplified the original MLT algorithm by using the space of uniform random numbers as the state space, based on the observation that paths are sampled by a sequence of random numbers (Fig. 2, left). This state space is denoted as the *primary sample space*, described as the unit hypercube $\mathcal{U} = [0,1]^{O(k)}$ ($O(k)$ random numbers are normally required to define a path of length k).

The relationship between a sample in the state space $\bar{u} \in \mathcal{U}$ and a path $\bar{x} \in \mathcal{P}$ can be written as the inverse of the cumulative distribution function $P^{-1} : \mathcal{U} \to \mathcal{P}$. The actual mapping of $\bar{u}$ to a path $P^{-1}(\bar{u})$ is obtained by using the underlying path sampler and using the random number sequence $\bar{u}$. With P^{-1} Eq. 1 can be rewritten as

$$I = \int_{\mathcal{U}} \tilde{f}(\bar{u}) \left| \frac{d\mu(\bar{x})}{d\bar{u}} \right| d\bar{u} = \int_{\mathcal{U}} \tilde{f}(\bar{u}) \left| \frac{dP^{-1}(\bar{u})}{d\bar{u}} \right| d\bar{u} = \int_{\mathcal{U}} \tilde{C}(\bar{u}) d\bar{u}, \tag{4}$$

where $\tilde{C}$ is the path throughput in primary sample space. We use the tilde to explicitly express the dependence on random numbers $\bar{u}$ instead of the path space vertices $\bar{x}$, i.e., $\tilde{C}(\bar{u}) = C(P^{-1}(\bar{u})) = C(\bar{x})$, where $C(\bar{x}) = f(\bar{x})/p(\bar{x})$ is the path contribution in path space.

Now using an appropriate scalar target function $\tilde{C}^*(\bar{u}_i)$ (the luminance of $\tilde{C}(\bar{u})$), MCMC can generate a sequence of samples distributed according to $\tilde{C}(\bar{u})$. That is, using a Markov chain $\bar{u}_i \in \mathcal{U}$ with N samples, we can compute an estimate $\langle I \rangle$ of I (Eq. 4) as

$$\langle I \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{\tilde{C}(\bar{u}_i)}{\tilde{C}^*(\bar{u}_i)/b} = \frac{b}{N} \sum_{i=1}^{N} \frac{\tilde{C}(\bar{u}_i)}{\tilde{C}^*(\bar{u}_i)}, \tag{5}$$

where $b = \int_{\mathcal{U}} \tilde{C}^*(\bar{u}) d\bar{u}$ is the normalization constant, which again is estimated with regular Monte Carlo techniques.

PSSMLT also uses the Metropolis-Hasting algorithm to generate the next sample. Given a current state $\bar{u}_i$, we first generate the next tentative state $\bar{v}$ from the proposal distribution q, i.e., $\bar{v} \sim T_{\mathcal{U}}(\bar{u}_i \to \bar{v})$, which is accepted as $\bar{u}_{i+1}$ with the probability:

$$a(\bar{u}_i \to \bar{v}) = \frac{\tilde{C}^*(\bar{v}) T_{\mathcal{U}}(\bar{v} \to \bar{u}_i)}{\tilde{C}^*(\bar{u}_i) T_{\mathcal{U}}(\bar{u}_i \to \bar{v})}, \tag{6}$$

and $\bar{u}_{i+1} = \bar{u}_i$ otherwise. A symmetric proposal distribution avoids the computation of the transition probabilities and Eq. 6 becomes

$$a(\bar{u}_i \to \bar{v}) = \frac{\tilde{C}^*(\bar{v})}{\tilde{C}^*(\bar{u}_i)}. \tag{7}$$

In general, there is no one-to-one mapping between primary space and a path since one primary space sample can be mapped to a set of paths (e.g., with bidirectional path tracing).

Multiplexed Primary Sample Space. Hachisuka et al. [2014] extended the formulation of PSSMLT to facilitate multiple importance sampling (MIS) [Veach and Guibas 1994] within this framework (Fig. 2, right). MIS combines multiple sampling strategies $p_t(\bar{x})_{t=1,\ldots,M}$ into an estimate I:

$$I = \int_{\mathcal{P}} \sum_{t=1}^{M} w_t(\bar{x}) f(\bar{x}) d\mu(\bar{x}) = \sum_{t=1}^{M} \int_{\mathcal{P}} w_t(\bar{x}) f(\bar{x}) d\mu(\bar{x}) \tag{8}$$

$$\approx \sum_{t=1}^{M} \frac{1}{N_t} \sum_{i=1}^{N_t} w_t(\bar{x}_{t,i}) C_t(\bar{x}_{t,i}), \tag{9}$$

where M is the number of techniques and $w_t(\bar{x})$ are the MIS weights satisfying $\sum_{t=1}^{M} w_t(\bar{x}) = 1$, and $C_t(\bar{x}_{t,i}) = f(\bar{x}_i)/p_t(\bar{x})$ is the throughput of the path $\bar{x}_i$ generated with a technique t.

Given the sampling technique t, we can write the relationship between $\bar{u}$ and $\bar{x}$ with the mapping P_t^{-1} associated with the technique t. Similar to PSSMLT, Eq. 8 can be written using this mapping as:

$$\begin{aligned} I &= \sum_{t=1}^{M} \int_{\mathcal{U}} \tilde{w}_t(\bar{u}) \tilde{f}(\bar{u}) \left| \frac{dP_t^{-1}(\bar{u})}{d\bar{u}} \right| d\mu(\bar{u}) \\ &= \sum_{t=1}^{M} \int_{\mathcal{U}} \tilde{w}_t(\bar{u}) \tilde{C}_t(\bar{u}) d\mu(\bar{u}) = \int_{\mathcal{U}} \sum_{t=1}^{M} \tilde{w}_t(\bar{u}) \tilde{C}_t(\bar{u}) d\mu(\bar{u}). \end{aligned} \tag{10}$$

Instead of using $\mathcal{U}$ for each t as a state space, MMLT utilizes an extended space named *multiplexed primary sample space* $\mathcal{U} \times \mathcal{T}$ where $\mathcal{T} = \{1, \ldots, M\}$, facilitating the idea of *serial tempering* [Marinari and Parisi 1992]. This method explores the state spaces $\mathcal{U}$ parameterized by the parameter t, and also facilitates Markov chain updates between the two different parameters t and $\bar{u}$. Using this method, we can sample the states according to $\sum_{t=1}^{M} \tilde{w}_t(\bar{u}) \tilde{C}_t(\bar{u})$ and estimate the last expression in Eq. 10 using a single Markov chain.

The Metropolis-Hasting update for MMLT considers two types of mutations: (1) a mutation within the same technique, and (2) a mutation among the different techniques. Both types of mutations can be considered in a *single* update with the acceptance ratio $\min(1, a([\bar{u}, t] \to [\bar{v}, t']))$ using

$$a([\bar{u}, t] \to [\bar{v}, t']) = \frac{\tilde{w}_{t'}(\bar{v}) \tilde{C}_{t'}(\bar{v}) T_{\mathcal{U}}([\bar{v}, t'] \to [\bar{u}, t])}{\tilde{w}_t(\bar{u}) \tilde{C}_t(\bar{u}) T_{\mathcal{U}}([\bar{u}, t] \to [\bar{v}, t'])}. \tag{11}$$

3.3 Discussion

MLT based on the path space is good for local explorations, as it can selectively re-create parts of a path. PSSMLT variants usually need to re-trace all path segments after updating the random numbers.

On the other hand, path space MLT has problems with global discovery of important "islands" in path space. This is because the only mutation strategy designed to fulfill ergodicity, the bidirectional mutation, needs to fix the number of eye and light subpath vertices upfront. This is required to be able to evaluate the transition kernel $T(\bar{x}_i \to .)$, and often leads to very low acceptance rates.

This a-priori decision for one particular technique is similar in spirit to MMLT, but MMLT also performs a local exploration of the state space. The MMLT formulation has one more property which is important to us: we are able to give a mapping from random numbers to path vertices.

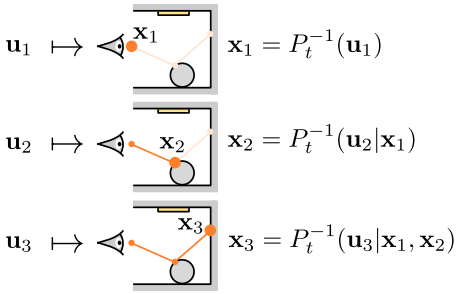

Fig. 3. Illustration of path sampling. A sequence of numbers $\bar{u} = (\mathbf{u}_1, \mathbf{u}_2, \ldots)$ is successively transformed by inverse CDFs P_t^{-1}. This process creates the path vertices $\mathbf{x}_1, \mathbf{x}_2, \ldots$ one after the other, depending on the chosen technique t and the path vertices that have already been sampled so far.

4 FUSING THE STATE SPACES

Our goal is to enable the use of mutation strategies from different state spaces, in particular from the (multiplexed) primary sample space and the path space as in MLT, in a single framework. We explain how to incorporate path space mutations into MMLT by introducing the concept of an *inverse path sampler*.

4.1 Paths to Numbers

An *inverse path sampler* returns a sequence of numbers $\bar{u}$ from a given path $\bar{x}$. While such a mapping cannot be uniquely determined in PSSMLT with the BDPT sampler, the multiplexed primary sample space leads to a straightforward derivation as follows.

Revisiting Path Samplers. In order to find such an inverse mapping, we need to revisit the precise meaning of a path sampler $P_t^{-1}(\bar{u}) = \bar{x}$. In general, one can factorize a multivariate CDF $P_t(\bar{x}) = P_t(\mathbf{x}_1, \cdots, \mathbf{x}_k)$ into a product of conditional CDFs:

$$\begin{aligned} &P_t(\mathbf{x}_1, \cdots, \mathbf{x}_k) = \\ &P_t(\mathbf{x}_1) P_t(\mathbf{x}_2|\mathbf{x}_1) \cdots P_t(\mathbf{x}_k|\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}), \end{aligned} \tag{12}$$

where $P_t(\mathbf{x}_k|\mathbf{x}_1, \cdots, \mathbf{x}_{k-1})$ is a CDF of $\mathbf{x}_k$ given $\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}$. Since all the conditional CDFs are different CDFs, $P_t(\mathbf{x}_2|\mathbf{x}_1)$ strictly should be written as $P_{t,\mathbf{x}_1}(\mathbf{x}_2|\mathbf{x}_1)$ for example. We use the notation $P_t(\mathbf{x}_2|\mathbf{x}_1)$ throughout this paper for brevity.

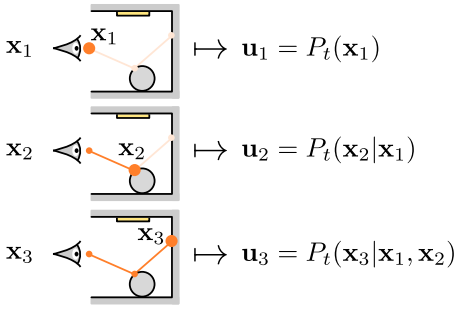

Fig. 4. Illustration of inverse path sampling. Analogous to creating a path from random numbers (Fig. 3), we can invert the process and successively compute random numbers which would have created the given path via path sampling. This is possible if the CDFs to create the vertices are bijections.

Using this factorization, the multivariate variant of inverse transform sampling generates $(\mathbf{x}_1, \cdots, \mathbf{x}_k) \sim p_t(\mathbf{x}_1, \cdots, \mathbf{x}_k)$ by a sequence of inverse transform sampling (see Fig. 3):

$$\begin{aligned} \mathbf{x}_1 &= P_t^{-1}(\mathbf{u}_1) \\ \mathbf{x}_2 &= P_t^{-1}(\mathbf{u}_2|\mathbf{x}_1) \\ &\cdots \\ \mathbf{x}_k &= P_t^{-1}(\mathbf{u}_k|\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}). \end{aligned} \tag{13}$$

This factorization approach, in general, is not very practical since an analytical form of the inverse of a conditional CDF is not often available. In typical Monte Carlo rendering systems, however, we can rely on this exact approach by generating each vertex $\mathbf{x}_i$ based on the previous vertices $\mathbf{x}_{i-1}, \cdots, \mathbf{x}_1$ (usually only on $\mathbf{x}_{i-1}$ and $\mathbf{x}_{i-2}$) according to a PDF proportional to the BRDF at $\mathbf{x}_{i-1}$.

This approach is often denoted as $P_t^{-1}(\bar{u}) = \bar{x}$, but it is not precise. The exact meaning of $P_t^{-1}(\bar{u}) = \bar{x}$ is the above sequential approach. We folded common cases where we need to use multiple numbers to sample a vertex in each CDF for the sake of brevity, but one can generally also factorize them into conditional univariate CDFs similar to above.

Inverse Path Samplers. This precise definition leads to the following inverse sampler from $\bar{x}$ to $\bar{u} = (\mathbf{u}_1, \cdots, \mathbf{u}_k)$ (see Fig. 4):

$$\begin{aligned} P_t(\mathbf{x}_1) &= \mathbf{u}_1 \\ P_t(\mathbf{x}_2|\mathbf{x}_1) &= \mathbf{u}_2 \\ &\cdots \\ P_t(\mathbf{x}_k|\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}) &= \mathbf{u}_k. \end{aligned} \tag{14}$$

We assume that each CDF is already available and can be evaluated easily. This assumption holds in practice since we usually define the CDF first in order to derive its inverse in path samplers. Note that the technique index t is unaffected due to the use of the multiplexed primary sample space.

While the definition of inverse path samplers might look trivial after the fact, such CDFs have not been used for practical purposes in rendering so far. Our work shows that they can be used for fusing the state spaces in MCMC rendering. We provide some examples of CDFs and how they can be trivially defined.

4.2 Examples of Inverse Path Samplers

Example 1: Cosine distribution. The cosine distribution is used to sample a direction according to the cosine of the angle from the surface normal. Since there are multiple algorithms to sample such a direction, we provide an example based on Malley's method [Malley 1988]. The method uses the polar coordinates (r, θ) of the projected direction onto the tangent plane around the normal. The PDF of the cosine distribution in this case is $p(r, \theta) = r/\pi$, so the marginal and conditional densities for r and θ are

$$p(r) = \int_0^{2\pi} p(r, \theta) d\theta = 2r \qquad p(\theta|r) = \frac{p(r, \theta)}{p(r)} = \frac{1}{2\pi}. \tag{15}$$

The mapping from (r, θ) to (u_1, u_2) is

$$\begin{aligned} u_1 = P(r) &= \int_0^r p(r) dr = r^2, \\ u_2 = P(\theta|r) &= \int_0^\theta p(\theta|r) d\theta = \frac{\theta}{2\pi}, \end{aligned} \tag{16}$$

Example 2: Uniform sampling on a triangle. Uniform sampling on a triangle is often used to generate an initial vertex of a light path. Similar to the previous example, we sample a point in different coordinates and convert to the surface point. Typically, we transform to the barycentric coordinates of the isosceles right triangle for this purpose. Given a point with barycentric coordinates (b_1, b_2), the mapping from (b_1, b_2) to (u_1, u_2) is

$$\begin{aligned} u_1 = P(b_1) &= \int_0^{b_1} p(b_1) db_1 = 2b_1 - b_1^2, \\ u_2 = P(b_2|b_1) &= \int_0^{b_2} p(b_2|b_1) db_2 = \frac{b_2}{1 - b_1}. \end{aligned} \tag{17}$$

Example 3: Tabulated PDF. In some cases, such as sampling a direction according to an environment map, we generate samples according to a tabulated PDF. Such a tabulated PDF $p(x)$ can be defined as a piecewise-constant function with N bins over $[x_0, x_N]$:

$$p(x) = \begin{cases} p_j & x \in [x_0, x_N] \\ 0 & \text{otherwise} \end{cases}, \tag{18}$$

where $j = \lfloor N \frac{x - x_0}{x_N - x_0} \rfloor + 1$. We consider a one dimensional CDF for simplicity. The CDF in this case becomes a piecewise-linear function, and the mapping from x to u is

$$u = P(x) = \begin{cases} P_j + p_j N (x - \frac{j-1}{N}) & x \in [x_0, x_N] \\ 0 & x \in x < x_0 \\ 1 & x \in x > x_N \end{cases}, \tag{19}$$

where $P_j = \sum_{k=1}^{j-1} \frac{p_k}{N}$. The summation for P_j can be accelerated by precomputing P_j and finding a bin index using binary search.

Example 4: GGX distribution. There are various microfacet normal distributions and the shape of the distributions can be complex. The distributions, however, are well designed to make it possible to derive CDFs for importance sampling. The GGX distribution [Walter et al. 2007] controls the width of the distribution with a parameter α. The PDF in spherical coordinates (θ, ϕ) is

$$p(\theta, \phi) = \frac{\alpha^2 \sin\theta}{\pi \cos^3\theta (\alpha^2 + \tan^2\theta)^2}. \tag{20}$$

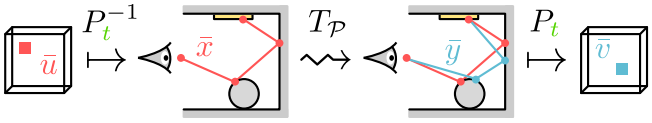

Fig. 5. We perform path space mutations by first converting the Markov chain state from our base space (multiplexed primary sample space) to path space. This is done using the standard Monte Carlo path sampler, applying P_t^{-1} to the random number sequence. We then mutate the state in path space via $T_{\mathcal{P}}$ as regular MLT does, without any modifications. Lastly, we convert the resulting path back to the base state space by inverting the path sampling process and applying P_t to the path to compute the random number sequence.

Following similar steps as previous examples, we obtain the marginal and conditional CDFs for θ and ϕ, and the mapping from (θ,ϕ) to $\mathbf{u} = (u_1,u_2)$ is

$$\begin{aligned} u_1 &= P(\theta) = \int_0^{\theta} p(\theta,\phi)d\theta = \frac{\tan^2\theta}{\alpha^2+\tan^2\theta}, \\ u_2 &= P(\phi|\theta) = \int_0^{\phi} p(\phi|\theta)d\phi = \frac{\phi}{2\pi}. \end{aligned} \tag{21}$$

4.3 Fusing State Spaces with Inverse Path Samplers

With the definition of inverse path samplers, we now have a mapping between the path space $\mathcal{P}$ and the primary sample space $\mathcal{U}$ for simple unidirectional samplers. Since we keep the technique t fixed up front, we can now also formulate a mapping between the two state spaces. In the following we consider paths with equal lengths since the change of path lengths would complicate the inverse mapping. Similar to the implementation of MMLT by Hachisuka et al. [2014], we use a separate Markov chain for each path length. In the following discussion, we opt to use the multiplexed primary sample space as the base space since we found that it naturally incorporates our formulation.

Fig. 5 illustrates this process: in order to perform a mutation in the path space, we first map the current state $(\bar{u},t)$ to a path $\bar{x} = P_t^{-1}(\bar{u})$ using the current sampling strategy t and the path sampler. We then apply a mutation and obtain the new path $\bar{y} \in \mathcal{P}$ according to $T_{\mathcal{P}}(\bar{x} \to \bar{y})$. Using the inverse path sampler, we convert it back to $\mathcal{U}$ with $\bar{v} = P_t(\bar{y})$. The transition kernel in the path space $T_{\mathcal{P}}(\bar{x} \to \bar{y})$ is transformed to

$$T_{\mathcal{P}}(\bar{x} \to \bar{y}) = \tilde{T}_{\mathcal{P},t}(\bar{u} \to \bar{v}) \left| \frac{dP_t^{-1}(\bar{v})}{d\bar{v}} \right|, \tag{22}$$

where $\tilde{T}_{\mathcal{P},t}(\bar{u} \to \bar{v}) = T_{\mathcal{P}}(P_t^{-1}(\bar{u}) \to P_t^{-1}(\bar{v}))$. The acceptance ratio for the mapped paths then becomes

$$a([\bar{u},t] \to [\bar{v},t]) = \frac{\tilde{w}_t(\bar{v})\tilde{C}_t(\bar{v})\tilde{T}_{\mathcal{P},t}(\bar{v} \to \bar{u}) \left| \frac{dP_t^{-1}(\bar{u})}{d\bar{u}} \right|}{\tilde{w}_t(\bar{u})\tilde{C}_t(\bar{u})\tilde{T}_{\mathcal{P},t}(\bar{u} \to \bar{v}) \left| \frac{dP_t^{-1}(\bar{v})}{d\bar{v}} \right|}, \tag{23}$$

where $\left|dP^{-1}(\bar{u})/d\bar{u}\right| = 1/\tilde{p}(\bar{u})$ and $\left|dP^{-1}(\bar{v})/d\bar{v}\right| = 1/\tilde{p}(\bar{v})$ by the definition of CDFs. Eq. 23 is in fact equal to

$$a([\bar{u},t] \to [\bar{v},t]) = \frac{\tilde{w}_t(\bar{v})\tilde{f}(\bar{v})\tilde{T}_{\mathcal{P},t}(\hat{v} \to \hat{u})}{\tilde{w}_t(\bar{u})\tilde{f}(\bar{u})\tilde{T}_{\mathcal{P},t}(\hat{u} \to \hat{v})} \tag{24}$$

since we have

$$\tilde{C}_t(\bar{v}) \left| \frac{dP^{-1}(\bar{u})}{d\bar{u}} \right| = \frac{\tilde{C}_t(\bar{v})}{\tilde{p}(\bar{u})} = \frac{\tilde{f}(\bar{v})}{\tilde{p}(\bar{u})\tilde{p}(\bar{v})} \tag{25}$$

$$\tilde{C}_t(\bar{u}) \left| \frac{dP^{-1}(\bar{v})}{d\bar{v}} \right| = \frac{\tilde{C}_t(\bar{u})}{\tilde{p}(\bar{v})} = \frac{\tilde{f}(\bar{u})}{\tilde{p}(\bar{u})\tilde{p}(\bar{v})}. \tag{26}$$

We thus do not need to modify existing implementations of mutation strategies in order to use them in conjunction with another space.

4.4 Handling Deterministic Cases

Deterministic distributions such as perfect reflectors become a PDF with a delta function. We cannot uniquely determine the inverse mapping in this case, since no component of $\bar{u}$ was actually used to generate a sample from such distributions. To solve this issue, we use the lower dimensional subspace $\mathcal{U}_* \subset \mathcal{U}$ excluding all the components related to the delta distributions, which corresponds to a specular manifold in the path space [Jakob and Marschner 2012]. Using this subspace, the function $P(\bar{u}_*)$ defined on $\mathcal{U}_*$ becomes invertible. We fill the rest of the components with uniform random numbers, which is a special case of reversible jump MCMC [Green 1995]. This approach is also equivalent to lazy mutation in PSSMLT, which also accounts for the case where there is a difference in the number of dimensions between MCMC states.

The transition kernel in this case becomes

$$\begin{aligned} T_{\mathcal{P}}(\bar{x} \to \bar{y}) &= \tilde{T}_{\mathcal{P},t}(\bar{u} \to \bar{v}) \left| \frac{dP_t^{-1}(\bar{v}_*)}{d\bar{v}_*} \right| \left| \frac{d(\bar{v}_*,\bar{r})}{d\bar{v}} \right| \\ &= \tilde{T}_{\mathcal{P},t}(\bar{u} \to \bar{v}) \left| \frac{dP_t^{-1}(\bar{v}_*)}{d\bar{v}_*} \right|. \end{aligned} \tag{27}$$

The conversion from $(\bar{v}_*,\bar{r})$ to $\bar{v}$ is just a permutation so we have $\left|\frac{d(\bar{v}_*,\bar{r})}{d\bar{v}}\right| = 1$. This eventually yields the same equation as Eq. 24.

It is also possible to handle layered materials in a similar way as described above. Instead of reflection or transmission for specular surfaces, we can assign a selected lobe, e.g., diffuse or glossy, as a material type. When we want to convert a path to random numbers, we can then choose a uniformly distributed random number in the appropriate range that maps to this lobe. This treatment is similar in spirit to different mathematical formulations of concurrent works [Bitterli et al. 2017; Pantaleoni 2017].

5 IMPLEMENTATION

We implemented our proposed method in our renderer, including primary sample space mutation techniques as well as the path space mutation techniques in MLT [Veach and Guibas 1997] and manifold exploration [Jakob and Marschner 2012]. As for manifold exploration, we only implemented the variant for specular surfaces and omitted handling of glossy surfaces. The path space mutation techniques do not contain any code specific to our technique, so we can reuse them for MLT without modifications. Similar to MMLT,

Algorithm 1 The mutation step with the combined proposal distribution. The mutation is limited to maintain the path length. Given $[\bar{u}, t] \in \mathcal{U} \times \mathcal{T}$ as the current state, this algorithm computes the next state $[\bar{u}_*, t_*]$.

if primary space space mutation is selected **then**
  $[\bar{v}, t'] \sim T_{\mathcal{U}}([\bar{u}, t] \to \cdot)$
  $a \leftarrow \min\left(1, \frac{\tilde{w}_{t'}(\bar{v})\tilde{C}_{t'}(\bar{v})T_{\mathcal{U}}([\bar{v},t']\to[\bar{u},t])}{\tilde{w}_t(\bar{u})\tilde{C}_t(\bar{u})T_{\mathcal{U}}([\bar{u},t]\to[\bar{v},t'])}\right)$ ▷ Eq. 11
  if Random() $< a$ **then**
    $[\bar{u}_*, t_*] \leftarrow [\bar{v}, t']$
  else
    $[\bar{u}_*, t_*] \leftarrow [\bar{u}, t]$
  end if
else
  $\bar{x} \leftarrow P_t^{-1}(\bar{u})$
  $\bar{y} \sim T_{\mathcal{P}}(\bar{x} \to \cdot)$
  $a \leftarrow \min\left(1, \frac{\tilde{w}_t(\bar{v})\tilde{f}(\bar{v})\tilde{T}_{\mathcal{P},t}(\hat{v}\to\hat{u})}{\tilde{w}_t(\bar{u})\tilde{f}(\bar{u})\tilde{T}_{\mathcal{P},t}(\hat{u}\to\hat{v})}\right)$ ▷ Eq. 24
  if Random() $< a$ **then**
    $[\bar{u}_*, t_*] \leftarrow [P_t(\bar{y}), t]$
  else
    $[\bar{u}_*, t_*] \leftarrow [\bar{u}, t]$
  end if
end if

our method separates the integral into path lengths and generates one Markov chain for each path length. Therefore the bidirectional mutation which involves a change in path length cannot be used in combination with our technique. Instead we implemented a fixed length variant of the bidirectional mutation. Similar to MLT, our method combines various mutation techniques and thus the selection of the mutation strategy has a great influence on the efficiency. Following the suggestion in Section 5.3.4 in the MLT paper [Veach and Guibas 1997], we randomly select a *valid* mutation strategy given the current state to avoid meaningless selections which would always be rejected. We parallelize the algorithm by computing several Markov chains assigned to different threads.#45

The core of the technique is almost the same as MMLT. We maintain the state in the multiplexed primary sample space and initialize it with random numbers. In our method, on the other hand, the mutation process in MMLT is replaced by Algorithm 1. The entire flow of the algorithm is a mixture of two Metropolis-Hastings updates for each state space. One difference is that we need to multiply MIS weights in addition to the measurement contribution function in Eq. 24. Obviously, the mapping functions introduce additional computational overhead for the path space mutations. To alleviate this issue we maintain the mapped state $P^{-1}(\bar{u})$ as a cache as well as the state $\bar{u}$ in the process of the mutations.

When going from path space to primary sample space, we need to decide on a technique t. The choice does not affect the correctness of the algorithm. It would, for instance, be possible to importance sample t based on the MIS weight of a given path, i.e., the acceptance probability of the tentative sample in the Markov chain. In our current implementation, we always initialize the Markov chain with a multiplexed primary sample space state, such that it comes with a valid t. If we choose a path space mutation, we will leave the technique t and the path length untouched, such that t is still valid when going back to multiplexed primary sample space.

6 RESULTS

Setup. We implemented MMLT [Hachisuka et al. 2014], MLT [Veach and Guibas 1997], and the proposed algorithm in the same rendering system. For mutation techniques, we implemented small and large step mutations in the primary sample space, and the bidirectional mutations, lens/caustic/multi-chain perturbations, and manifold exploration [Jakob and Marschner 2012] as the path space mutations. The reference images are rendered using BDPT or vertex connection and merging [Georgiev et al. 2012]/unified path sampling [Hachisuka et al. 2012] with more than six hours of computation for each scene. We conducted all the experiments on a machine with an Intel Xeon E5-2698 v3 at 2.3 GHz using 32 threads.

The comparisons are all equal-time (20 minutes) with the maximum path length between 9–20 depending on the scenes. We manually tuned parameters for mutation techniques to achieve the best results for MMLT and MLT. We then used the same set of best parameters for our methods. The code for our proposed method will be available on our website.

Experiments. Fig. 1 compares rendered images by MMLT, MLT, and our combination of both. We rendered two different scenes with different characteristics to highlight the need for fusing the state spaces. The *door* scene contains diffuse, specular, and glossy surfaces indirectly illuminated by a light leaking through the opening door. MLT handles this scene well because it can partially regenerate a path depending on the selection of the mutation strategies. The performance of MMLT is worse for this scene because MMLT needs to re-trace the complete path for every mutation. The *necklace* scene consists mainly of glossy surfaces illuminated by light sources of varying sizes causing complex interreflection between glossy surfaces. MMLT is effective for this scene because the target distribution includes the MIS weights. On the other hand, MLT tends to get stuck in local subspaces and generates non-uniform artifacts in the rendered image. In both cases, the result shows that our method can alleviate this robustness issue by combining mutation techniques from the two different state spaces.

We show two more scenes in Figs. 7 and 8. The *Salle de bain* scene contains diffuse and glossy surfaces illuminated with an area light source. Specifically the scene contains mirror models with low glossiness. This scene successfully captures the trade-off between MMLT and MLT in a single image. MMLT can render the part of the image visible from the mirror model more efficiently than MLT. On the other hand, MLT can render the part containing diffuse surfaces better than MMLT. The *Grey and White Room* scene consists of diffuse and glossy surfaces illuminated with several area light sources. This scene shows similar characteristic as the *door* scene and MLT outperforms MMLT. Again in this scene, our method can render the image with similar performance as MLT, the better one.

Error Analysis. Fig. 6 shows the pixel-wise error distributions for the images in Fig. 1 compared to the references. We also show the error images for the scenes in Figs. 7 and 8. We used relative root mean square error (rRMSE) as an error metric. Our method exhibits

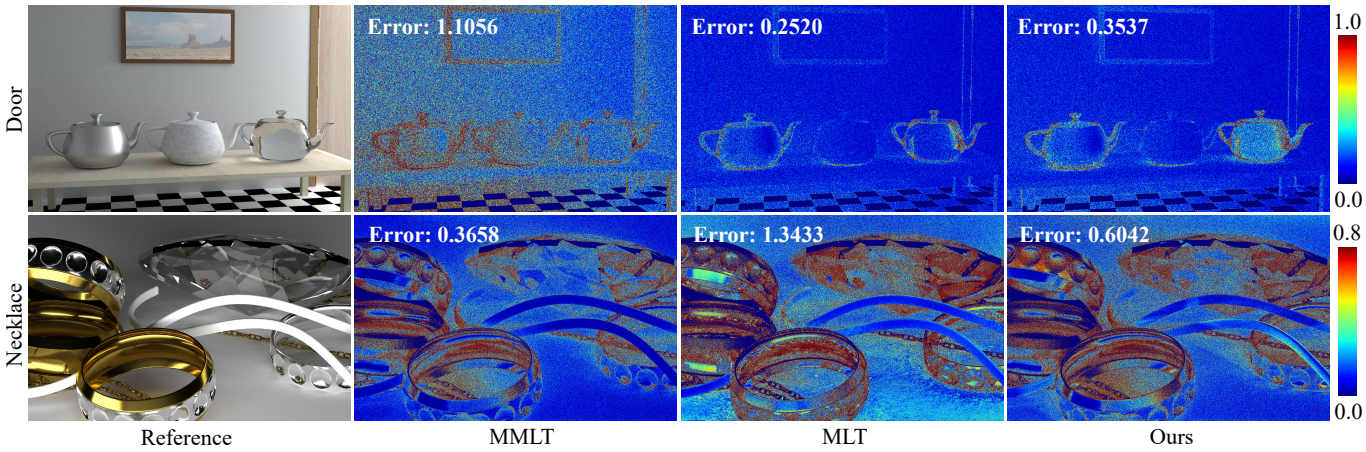

Fig. 6. Comparisons of the errors for the scenes in Fig. 1. We show the relative RMSE for each images, and also show the pixel-wise relative error. We can observe that the combined mutation techniques with our method exhibit the error distribution similar to the better one of MMLT or MLT.#45 Although the errors are suboptimal compared to the better one for the respective situation, this figure indicates our proposed method can moderately alleviate trade-offs between the two techniques, also in terms of the errors.

similar error distributions as the better one in both scenes. In both cases, however, rRMSE values are larger than the respective better method, and some regions of the images display higher error than the better one.

We want to stress that achieving the best result among all is not our claim. The practical benefit is the generality to provide good performance for a wide spectrum of scenes, namely improved *robustness*, which is possible only by combining various mutation techniques. This context is different from multiple importance sampling which aims at an optimal combination of different sampling techniques. While it would be ideal if we could select an optimal mutation strategy (or sets thereof) and the optimal space for a given scene, a combination of mutation strategies in different spaces has been just *impossible* to begin with, prior to our work. The optimal combination of different mutation strategies also remains as an interesting direction of the future work.

7 DISCUSSION AND LIMITATIONS

7.1 Discussion

Computational cost. When switching from one space to another, our algorithm requires extra computational cost to convert a sequence of numbers in the primary sample space to a path, and then convert a path back to a sequence of numbers. This computational cost is, however, negligible compared to the entire render time. For instance, the conversion takes only 0.03% of the rendering time for the *necklace* scene in Fig. 1. The expected number of required transitions was 0.131 per single mutation for this scene. In addition, as was discussed in Section 5, it is possible to cache some of these computations.

Combination with other state spaces. We focused on combining the primary sample space and the path space as they are major state spaces used in MCMC rendering. The half-vector space [Kaplanyan et al. 2014] is another state space that has its own advantages over the path space. While we have not implemented their techniques in the rendering system we used for this paper, it should be straightforward to combine them by a composition of the transformations between domains (e.g., half vector space to path space to primary sample space, and vice versa).

Non-bijective sampling procedures. Our solution of handling deterministic cases generalizes to more cases where the primary sample space formulation consumes more numbers than the resulting dimensions in the path space. A concurrent work [Pantaleoni 2017] provides a mathematical formulation of a similar process. There is also an interesting connection to using expected values in the measurement contribution when evaluating the acceptance probability [Kronander et al. 2015], which might lead to a more efficient approach for handling trans-dimensional moves in general.

Numerical precision. When working in the primary sample space, extra care should be taken to avoid problems with numerical precision. For example, when mapping an outgoing direction to random numbers for BRDF sampling, numerical errors can accumulate and introduce bias. It is essential to work in double precision and add numerical guards against error accumulation and drifting. We implemented two techniques to assess numerical precision.

Firstly, during initial path sampling, we use only the valid samples based on the *round-trip values*. Given the candidate of the initial state u, we compute $\bar{x} = P_t^{-1}(\bar{u})$ and the round-tripped initial state $P_t(\bar{x})$. We decided not to use $\bar{u}$ as an initial state if $\tilde{C}(P_t(\bar{x}))$ has zero contribution. This test in the initialization phase can introduce MCMC start-up bias, which will vanish while converging as the Markov chain runs for longer. Without this test, the chain might get stuck in the same state, depending on the combination of the mutation strategies. Second, in the case of path space mutation, we also check the round-trip values for the mutated path $\bar{x}$. We check if $C(P_t^{-1}(P_t(\bar{x})))$ is non-zero, and the proposal is rejected immediately if the condition is not met.

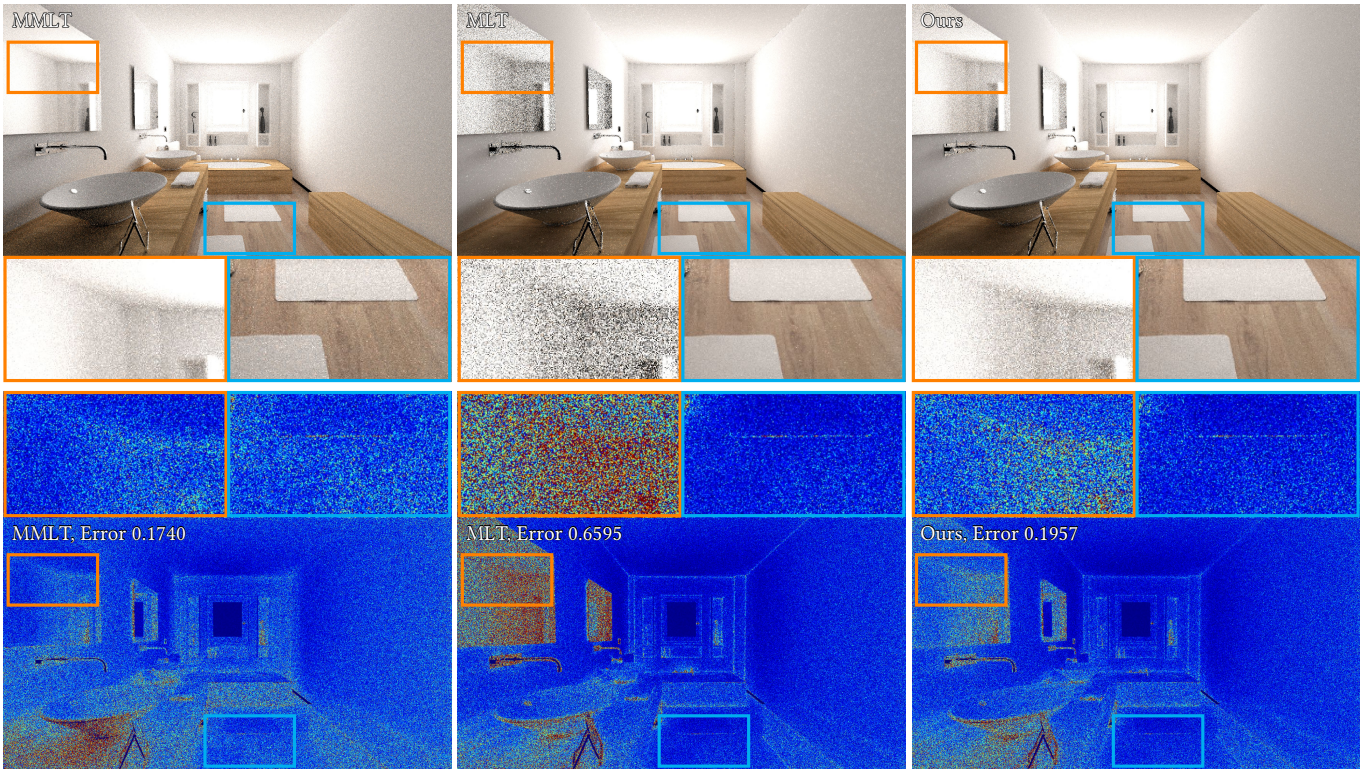

Fig. 7. The *Salle de bain* scene (equal time comparison 20 min). A set of mutation strategies employed in MLT has problems with the mirror (orange inset) while MMLT is more noisy on the floor. Our combination alleviates both issues. Also note the reduced clumping of samples in the edges of the room as compared to MLT and the reduced noise in front of the first washing basin as compared to MMLT.

This verification process can be optimized because $P_t^{-1}(P_t(\bar{x}))$ can be reused if the next mutation is a path space mutation. Similarly, this early rejection in the MH update can introduce bias.

7.2 Limitation

One limitation of our formulation is the requirement of an analytical mapping between a path to a sequence of numbers. This requirement is particularly violated for rejection sampling such as Woodcock tracking. In this case, we do not have an analytical form of (inverse) cumulative distribution functions since rejection sampling does not rely on the existence of such analytical solutions. It is still unclear how to handle cases where an analytical form of such a mapping is not easily accessible.

8 CONCLUSION

We proposed a framework to fuse the path space and the primary sample space of MCMC rendering for the first time. We explained how to formulate the connection between these two state spaces by introducing a novel mapping from a path to a sequence of numbers. This mapping is the inverse of existing path samplers and we thus named them as inverse path samplers. We show how such an inverse mapping can be formulated with cumulative distribution functions of samples which are oftentimes readily available as we already use inverse cumulative distribution functions for path samplers. Inverse path samplers allow us to use mutations designed only for one state space in another state space without modifying the mutation algorithms themselves. The results demonstrate that the fusion of two state spaces brings a practical benefit of robustness to different scene configurations, even when the use of one state space alone fails. Our framework should be immediately useful for existing MCMC rendering systems since it essentially introduces new mutation strategies in each state space without modifying each implementation. We believe that our framework leads to a new family of MCMC rendering methods since it provides a well-defined connection between two state spaces.

ACKNOWLEDGEMENTS

We would like to thank the reviewers for their insightful comments. The *door* scene is originally made by Miika Aittala, Samuli Laine, and Jaakko Lehtinen, and the *necklace* scene is made by Alex Telford. We thank the blendswap.com artist Wig42 for the *Grey and White Room* scene and nacimus for the *Salle de bain* scene. Both scenes are converted and distributed by Benedikt Bitterli, and converted to Wavefront OBJ format by Nicholas Hull. This work was supported by JSPS KAKENHI Grant Number 15H05308.

REFERENCES

James Arvo. 1986. Backward ray tracing. In *Developments in Ray Tracing, ACM SIGGRAPH Course Notes*. 259–263.

Benedikt Bitterli, Wenzel Jakob, Jan Novák, and Wojciech Jarosz. 2017. *Reversible Jump Metropolis Light Transport using Inverse Mappings.* Technical Report. arXiv preprint

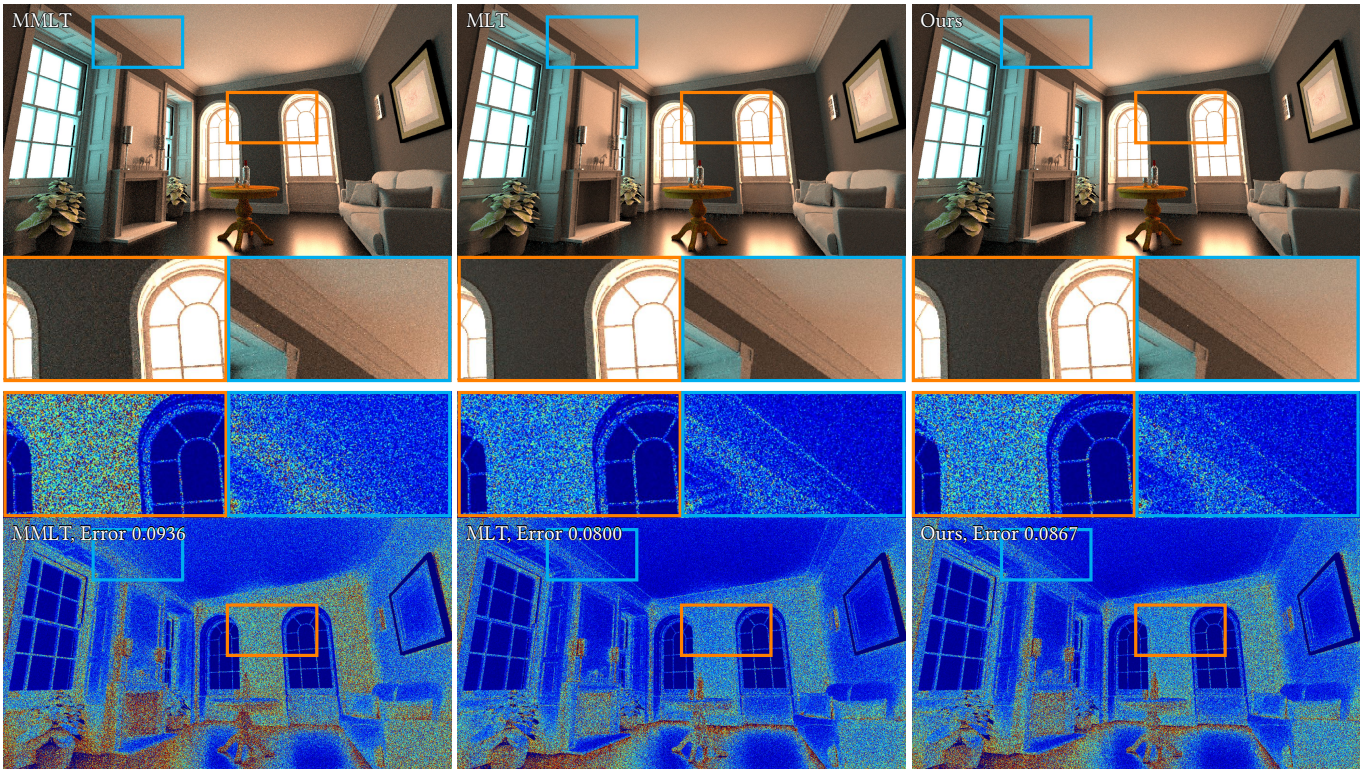

Fig. 8. The *Grey and White Room* scene (equal time comparison 20 min). The scene has similar noise characteristics as the *door* scene and MLT generates better result than MMLT. Similar to the *door* scene, our method inherits the noise characteristics from the better method.

arXiv:1704.06835v1.
Iliyan Georgiev, Jaroslav Křivánek, Tomáš Davidovič, and Philipp Slusallek. 2012. Light Transport Simulation with Vertex Connection and Merging. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 31, 6 (2012), 192:1–192:10.
Peter J. Green. 1995. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* 82 (1995), 711–732.
Toshiya Hachisuka, Anton S. Kaplanyan, and Carsten Dachsbacher. 2014. Multiplexed Metropolis Light Transport. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 33, 4, Article 100 (2014).
Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. 2008. Progressive photon mapping. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 27, 5 (2008), 130.
Toshiya Hachisuka, Jacopo Pantaleoni, and Henrik Wann Jensen. 2012. A Path Space Extension for Robust Light Transport Simulation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 31, 6 (2012), 191:1–191:10.
Johannes Hanika, Anton S. Kaplanyan, and Carsten Dachsbacher. 2015. Improved Half Vector Space Light Transport. *Computer Graphics Forum (Proc. Eurographics Symposium on Rendering)* 34, 4 (2015), 65–74.
W Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, 1 (1970), 97–109.
Wenzel Jakob and Stephen Marschner. 2012. Manifold exploration: a Markov chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 31, 4, Article 58 (2012).
Henrik Wann Jensen. 1996. Global illumination using photon maps. In *Proc. Eurographics Workshop on Rendering*. 21–30.
James T. Kajiya. 1986. The rendering equation. *Computer Graphics (Proceedings of SIGGRAPH '86)* 20, 4 (1986), 143–150.
Anton Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. 2014. The Natural-Constraint Representation of the Path Space for Efficient Light Transport Simulation. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 33, 4 (2014), 1–13.
Csaba Kelemen, László Szirmay-Kalos, György Antal, and Ferenc Csonka. 2002. A simple and robust mutation strategy for the Metropolis light transport algorithm. *Computer Graphics Forum* 21, 3 (2002), 531–540.
Joel Kronander, Thomas B. Schön, and Jonas Unger. 2015. Pseudo-marginal Metropolis Light Transport. In *SIGGRAPH Asia 2015 Technical Briefs*. 13:1–13:4.
Eric P. Lafortune and Yves D. Willems. 1993. Bi-Directional Path Tracing. In *Compugraphics '93*. 145–153.
Tzu-Mao Li, Jaakko Lehtinen, Ravi Ramamoorthi, Wenzel Jakob, and Frédo Durand. 2015. Anisotropic Gaussian Mutations for Metropolis Light Transport Through Hessian-Hamiltonian Dynamics. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 34, 6 (2015), 209:1–209:13.
Thomas Malley. 1988. *A Shading Method for Computer Generated Images*. Master's thesis. University of Utah.
Enzo Marinari and Giorgio Parisi. 1992. Simulated tempering: a new Monte Carlo scheme. *EPL (Europhysics Letters)* 19, 6 (1992), 451.
Jacopo Pantaleoni. 2017. Charted Metropolis Light Transport *(SIGGRAPH '17)*. ACM.
Peter Shirley, Bretton Wade, Philip M. Hubbard, David Zareski, Bruce Walter, and Donald P. Greenberg. 1995. Global Illumination via Density-Estimation. In *Rendering Techniques 1995 (Proceedings of the Eurographics Workshop on Rendering)*. 219–230.
Eric Veach. 1998. *Robust Monte Carlo methods for light transport simulation*. Ph.D. Dissertation. Stanford University, USA. Advisor(s) Guibas, Leonidas J. AAI9837162.
Eric Veach and Leonidas Guibas. 1994. Bidirectional Estimators for Light Transport. In *Proc. Eurographics Workshop on Rendering*. 147–162.
Eric Veach and Leonidas Guibas. 1997. Metropolis Light Transport. In *SIGGRAPH '97*. 65–76.
Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. 2007. Microfacet Models for Refraction Through Rough Surfaces. *Proc. Eurographics Symposium on Rendering* (2007), 195–206.