A Remedy For Proposal Failures in Metropolis Light Transport

HARRY POTTER, HERMIONE GRANGER, and RON WEASLEY, Hogwarts School of Witchcraft and Wizardry

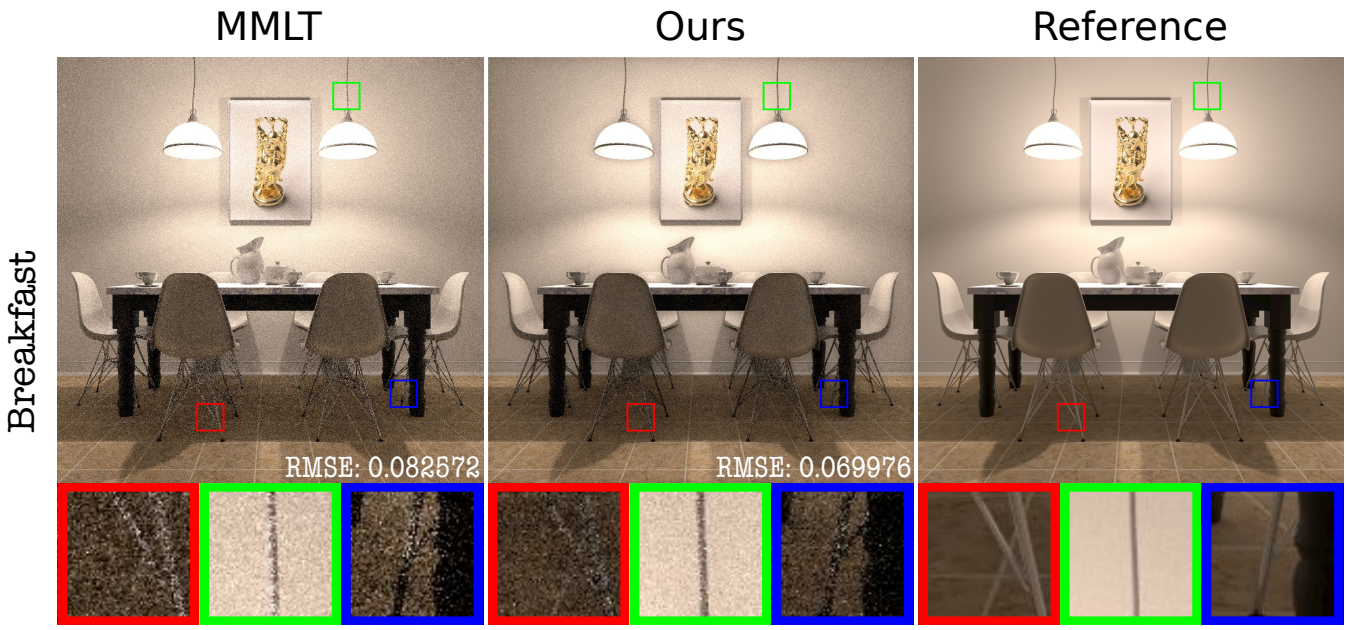


Fig. 1. *The Breakfast scene.* Equal-time comparison. Left: Image rendered with MMLT shows suboptimal performance, because more than 37% of the paths that were used to reconstruct the image carried no radiance. Middle: Giving special treatment to those zero-contribution paths, our method reduces the percentage to less than 2% and exhibits much better results with smaller RMSE. If our method is used to produce the same RMSE/quality result as MMLT does, in general, about a third of the computation will be saved.

The high percentage of the paths used to reconstruct the final image that carries zero radiance is a big problem for all path-based rendering algorithms, The situation for MLT-related algorithms building on both path space and primary sample space, is improved, but still serious. These paths with no contribution will present as noises in the final image, which also slows down the convergence speed of the Metropolis-Hastings algorithm and decreases the efficiency of the rendering algorithms.

The root cause of this problem in MLT-type rendering algorithms piggybacking on bidirectional path tracing is the failure of obtaining a complete mutation light path. We call the failure as *proposal failure*, which may be caused by the failure of tracing required two sub-paths or the failure of connecting the two sub-paths.

To address this issue, we propose a novel algorithm which provides special remedy for those proposal failures to decrease the percentage of the paths used to reconstruct the final image that carries zero radiance. Firstly, our algorithm eliminates the impact of the current proposal failure on both the contribution of the current pixel and subsequent proposal paths, and then traces several extra proposal paths trying to find a nonzero contribution path. Though some overhead must be paid for the extra mutation paths, our method substantially decreases the percentage of zero contribution paths that are used to calculate the final picture. Generally, MLT-type rendering algorithms extended with our method will be much more efficient than the original algorithms. Furthermore, the extensions are easy to implement.

CCS Concepts: • **Computing methodologies** → **Computer graphics**;

Additional Key Words and Phrases: rendering

ACM Reference format:
Harry Potter, Hermione Granger, and Ron Weasley. 2017. A Remedy For Proposal Failures in Metropolis Light Transport. *ACM Trans. Graph.* 36, 4, Article 1 (July 2017), 14 pages.
https://doi.org/

1 INTRODUCTION

Physically-based rendering is extensively used now, but solving the rendering equation is still a laborious task, especially for scenes with challenging lighting, geometry or material.

[Veach 1997] introduced Metropolis-Hasting Sampling ([Hastings 1970; Metropolis et al. 1953]) into computer graphics and presented the original Metropolis Light Transport (MLT). The main advantage of MLT is that the path space is explored locally by making small mutations to the current path to generate proposal path, which makes MLT a substantially efficient rendering algorithm for particularly challenging scenes. However, MLT's mutation rules are complex

and none of the rules is symmetric, which makes its implementation a significant undertaking.

[Kelemen et al. 2002] presented Primary Sample Space MLT (PSSMLT), whose mutation strategy in the unit cube of pseudo-random numbers is symmetric and easy to implement. [Hachisuka et al. 2014] presented an extension to PSSMLT named Multiplexed Metropolis Light Transport (MMLT) which combines MCMC methods with Multiple Importance Sampling [Veach 1997].

Some Adaptive mutation strategies working in path space or primary sample space have been developed, such as MEMLT [Jakob and Marschner 2012] for specular interactions, HSLT [Hanika et al. 2015] for rough materials, and GeoMLT [Otsu et al. 2018], which incorporates visibility into mutation strategies of MCMC rendering, and H2MC [Li et al. 2015], which is based on MMLT's approach and utilizes Hamiltonian Monte Carlo to adaptively control the shape of the transition kernels.

Also, several methods have been developed to bridge path space and primary sample space by using invertible mappings to transform samples between them, such as [Bitterli et al. 2018; Otsu et al. 2017; Pantaleoni 2017].

Unfortunately, the percentage of the paths that are used to reconstruct image carried no radiance in all of those algorithms is still high for challenging scenes, which brings serious variance to the final image. This paper provide a simple but efficient way to reduce the percentage substantially without any upgrades of mutation strategies.

We consider the proposal paths of zero-contribution as proposal failures and distinguish them from normal proposal paths. When proposal failure occurs, instead of rejecting it and then accumulating contribution on the basis of the acceptance rate, which is zero for proposal failures, our method discards the proposal failure directly and completely by eliminating the impact of the current proposal failure on both the contribution of the current pixel and subsequent proposal paths, and then traces several extra proposal paths trying to find a nonzero contribution path. The number of extra proposal paths can be set by users.

Though some overhead must be paid for the extra mutation paths, our method substantially decreases the percentage of zero contribution paths that are used to calculate the final picture. As a result, our method increases the overall efficiency a lot. As exemplified in Figure 1, compared to MMLT, our method reduces the percentage of zero-radiance paths that were used to reconstruct the final image from 37% to less than 2%. If our method is used to produce the same RMSE/quality result as MMLT does, about a third of the computation will be saved. What's more, our method can be easily extended to all MLT-type algorithms.

In summary, the contributions of this paper are:

(1) Introduction of the concept of proposal failure.
(2) A novel extension of Metropolis sampling algorithm which simulates proposal failures and provides a remedy for them.
(3) Combination of the novel extension of Metropolis sampling with MMLT.

2 RELATED WORK

Light Transport Simulation. Light transport equation was first described by [Kajiya 1986]. [Veach 1997] presented that light transport simulation can be expressed as the integral:

$$I_j = \int_\Omega h_j(\bar{x}) f(\bar{x}) d\mu(\bar{x}) \tag{1}$$

, where I_j is the intensity of the j-th pixel of image, $\Omega = \bigcup_{k=1}^{\infty} \Omega_k$ is the space of light paths of all finite lengths k, $\bar{x}$ is a light path, $h_j()$ is the reconstruction filter that selects out all the light paths that contribute to the j-th pixel, f is the contribution of $\bar{x}$ and μ is the area measure.

Monte Carlo Integration. The integral in Equation (1) is infinite-dimensional because of the length of $\bar{x}$ can be infinite, which cannot be solved analytically. Monte Carlo numerical integration methods provide one solution to this kind of problem. The Monte Carlo integration estimator of Equation (1) is:

$$I_j \approx \hat{I}_j = \frac{1}{N} \sum_{i=1}^{N} \frac{h_j(\bar{x}^{(i)}) f(\bar{x}^{(i)})}{p(\bar{x}^{(i)})} \tag{2}$$

, where $\bar{x}^{(i)}$ is the i-th independently sampled light path, $p(\bar{x}^{(i)})$ is the probability density of the sample and N is the total number of samples.

In order to decrease the variance of the estimator, $p(\bar{x}^{(i)})$ should be approximately proportional to $f(\bar{x}^{(i)})$. Unfortunately, $f(\bar{x}^{(i)})$ is a spectrally valued function, and thus there is no unambiguous notion of what it means to generate samples proportional to $f(\bar{x}^{(i)})$. To address this issue, a scalar contribution function f^* is defined as the luminance of the path contribution f. Intuitively, $p(\bar{x}^{(i)})$ can be $f^*(\bar{x}^{(i)})/b$ (where b is the normalization constant $\int_\Omega f(\bar{x}) d\mu(\bar{x})$). So, Equation (2) can be presented as:

$$I_j \approx \hat{I}_j = \frac{b}{N} \sum_{i=1}^{N} \frac{h_j(\bar{x}^{(i)}) f(\bar{x}^{(i)})}{f^*(\bar{x}^{(i)})} \tag{3}$$

Now, we face two problems: calculating b and geting samples from $p(\bar{x}^{(i)})$ that is $f^*(\bar{x}^{(i)})/b$. The value b is typically estimated with another independent rendering algorithm such as bidirectional path tracing ([Lafortune and Willems 1993, 1996; Veach and Guibas 1994]). Then, the only problem is how to do the sampling.

Path-Space MLT.. The remaining problem in Equation (3) is how to get sample $\bar{x}^{(i)}$ in path space Ω with probability density $p(\bar{x}^{(i)})$ proportional to it's function value $f(\bar{x}^{(i)})$, which is so-called *importance sampling*. Veach[Veach and Guibas 1997] originally introduced Metropolis-Hastings (MH) sampling ([Hastings 1970; Metropolis et al. 1953]) into computer graphics and proposed a novel rendering algorithm, Metropolis Light Transport (MLT).

The mathematical foundation of MH sampling is building a Markov Chain whose stationary distribution is exactly the target distribution. After reaching its stationary distribution, every subsequent mutation state of the Markov Chain is a valid sample of the target distribution. However, it's never easy to know when the stationary is reached. MH sampling provides a practically way to solve this problem. It generates a set of samples from a non-negative function f that is distributed proportionally to $f's$ value without waiting for

the stationary and whithout requiring the evaluation of b in two steps:

- *Proposal.* A new proposal path $\bar{y}$ is obtained from current path $\bar{x}$ by a mutation kernel $Q(\bar{y}|\bar{x})$.
- *Acceptance-Rejection.* Acceptance rate α is calculated:

$$\alpha(\bar{y}|\bar{x}) = min\{1, \frac{f^*(\bar{y})Q(\bar{x}|\bar{y})}{f^*(\bar{x})Q(\bar{y}|\bar{x})}\} \tag{4}$$

  . Then accept the proposal path $\bar{y}$ with the probability α, otherwise reject it.

Mutation kernel Q is the key factor increasing acceptance rate and computation efficiency. In rendering algorithms, mutation kernel Q actually is the strategy to get proposal paths. The original MLT designs three mutation strategies targeting specific families of light paths, such as caustics or paths containing sequences of specular-diffuse-specular interactions. Some other mutation strategies have been developed in variants of path-space MLT, such as MEMLT [Jakob and Marschner 2012] for specular interactions, HSLT [Hanika et al. 2015] for rough materials, and GeoMLT [Otsu et al. 2018], which incorporates visibility into mutation strategies of MCMC rendering.

Primary Sample Space MLT.. Mutating in path space is not symmetric, which means that $Q(\bar{x}|\bar{y})$!= $Q(\bar{y}|\bar{x})$. This makes implementing these rendering algorithms a significant undertaking and makes debugging a notorious task. [Kelemen et al. 2002] presented a new MLT-type algorithm, Primary Sample Space MLT (PSSMLT), whose mutation strategy works in the unit cube of pseudo-random numbers, which is symmetric and easy to implement. The unit cube is called Primary Sample Space U.

PSSMLT converts random numbers $\bar{u}$ into light path $\bar{x}$ by path sampling. $\bar{x}$ can be thought of as being mapped from $\bar{u}$ by the inverse cumulative distribution function $P^{-1}(\bar{u})$, where $P(\bar{u}) = \int_0^{\bar{u}} p(\bar{u}')d\bar{u}'$. That is: $\bar{x} = P^{-1}(\bar{u})$. For the brevity of notation, we define:

$$\begin{aligned} C(\bar{x}) &= \frac{f(\bar{x})}{p(\bar{x})} \\ \hat{C}(\bar{u}) &= C(P^{-1}(\bar{u})) = C(\bar{x}) \\ \hat{p}(\bar{u}) &= p(P^{-1}(\bar{u})) = p(\bar{x}) \\ \hat{h}_j(\bar{u}) &= h_j(P^{-1}(\bar{u})) = h_j(\bar{x}) \end{aligned}$$

Then, in primary sample space, Equation (1) can be expressed as:

$$I_j = \int_U \hat{h}_j(\bar{u})\hat{C}(\bar{u})d\bar{u} \tag{5}$$

Like Equation (3), the Monte Carlo integration estimator of Equation (5) can be presented as:

$$I_j \approx \hat{I}_j = \frac{b}{N}\sum_{i=1}^{N}\frac{\hat{h}_j(\bar{u}^{(i)})\hat{C}(\bar{u}^{(i)})}{\hat{C}^*(\bar{u}^{(i)})} \tag{6}$$

, where $\hat{C}^*$ is the luminance of the importance function $\hat{C}$. Mutating in primary sample space is symmetric. That is $Q(\bar{u}|\bar{v}) = Q(\bar{v}|\bar{u})$, which makes implementing PSSMLT much easier, and which also avoids the computation of the transition probabilities altogether. So, the calculation of acceptance rate can be simplified as:

$$\alpha(\bar{v}|\bar{u}) = min\{1, \frac{\hat{C}^*(\bar{v})}{\hat{C}^*(\bar{u})}\} \tag{7}$$

, where $\bar{v}$ is the proposal state of $\bar{u}$ in primary sample space.

PSSMLT makes the integrand much flatter, and increases the average acceptance rate and therefore reduces the variance. The generality of PSSMLT has led to a lot of applications, such as [Gruson et al. 2017; Hachisuka and Jensen 2011; Hoberock and Hart 2010; Kitaoka et al. 2009; Šik et al. 2016]. As mentioned before, the method of conversion from random numbers $\bar{u}$ into light path $\bar{x}$ is path sampling. Actually, PSSMLT adopts bidirectional path tracing to do the path sampling. Bidirectional path tracing produces many paths from a single primary sample by using different connection strategies to connect camera sub-path and light sub-path. And then, the connected path of maximum luminance is selected as the final proposal light path. Unfortunately, the process of finding the path of maximum luminance is not efficient.

Multiplexed MLT.. In order to solve the efficient problem of PSSMLT, [Hachisuka et al. 2014] presented an extension to PSSMLT called Multiplexed Metropolis Light Transport (MMLT) which combines MCMC algorithm with Multiple Importance Sampling [Veach 1997]. Instead of always implementing all BDPT connection strategies and finding the path of maximum luminance, the algorithm chooses a single strategy according to an extra random number and returns its contribution scaled by the inverse discrete probability of the choice. The additional random number used for strategy selection can be mutated in the same way as the other components of $\bar{u}$ in primary sample space. The Monte Carlo Integration estimator is generalized as:

$$I_j \approx \hat{I}_j = \sum_{i=1}^{M}\frac{b}{N_t}\sum_{i=1}^{N_t}\frac{\hat{h}_j(\bar{u}^{(i,t)})\hat{w}_t(\bar{u}^{(i,t)})\hat{C}(\bar{u}^{(i,t)})}{\hat{C}^*(\bar{u}^{(i,t)})} \tag{8}$$

, where, t is determined by the extra state dimension and is used to choose sample technique, and $\hat{w}_t$ is a weighting function for any given path. The extension of the extra state dimension for choosing sample technique results that every mutation may change both random numbers from $\bar{u}$ to $\bar{v}$ and sample technique from t to t'. So, based on Equation (7), the calculation of acceptance rate is generalized:

$$\alpha((\bar{v}, t')|(\bar{u}, t)) = min\{1, \frac{\hat{w}_{t'}(\bar{v})\hat{C}^*_{t'}(\bar{v})}{\hat{w}_t(\bar{u})\hat{C}^*_t(\bar{u})}\} \tag{9}$$

This generalization produces the practical effect that the Metropolis sampler mainly focus on more effective strategies that leads to greater MIS-weighted contributions to the final image. Furthermore, the individual iterations of every Markov Chain are much more efficient since they only execute a single connection strategy. Based on MMLT's approach, some developments have been made, such as H2MC[Li et al. 2015], which utilizes Hamiltonian Monte Carlo to adaptively control the shape of the transition kernels.

Bridging Path-Space and Primary-Sample-Space. In order to combine the flexibility of mutation strategies of path space MLT with the simplicity and efficiency of primary sample space MLT, several

methods have been developed to bridge the two state spaces by using invertible mappings to transform samples between them, such as [Bitterli et al. 2018; Otsu et al. 2017; Pantaleoni 2017].

Analysis and Problem Statement. Generally, the improvements in MLT-type rendering algorithms can be classified into three categories. First, developments of mutation strategies in path space, such as MEMLT [Jakob and Marschner 2012] for specular interactions, HSLT [Hanika et al. 2015] for rough materials, and GeoMLT [Otsu et al. 2018], which incorporates visibility into mutation strategies. Second, developments of mutation strategies in primary sample space, such as MMLT and H2MC [Li et al. 2015]. Third, bridging the two spaces, so that the developments of mutation strategies in both spaces can be used in the same algorithm framework, such as [Bitterli et al. 2018; Otsu et al. 2017; Pantaleoni 2017].

All of these efforts are put on variant mutation strategies to increase the acceptance rate and to reduce the variance of their different aim scene scenarios. However, none of them can generally avoid the serious impact caused by the high ratio of proposal failures that are paths carrying no radiance. For example, when MMLT is used to render *Villa Scene* in Figure 3, the ratio of proposal failures is up to 62.82%, which causes serious noise in the final image.

To our knowledge, all MLT-type algorithms don't distinguish these proposal failures from normal proposal light paths. Considering that the acceptance rate is zero, the contribution of the current path is accumulated again. However, in the proposal failure case, neither the contribution of the current path nor the contribution of the proposal failure path is proportional to the probability with which the proposal failure path was sampled, so the variance of Equation (2) is surely increased. Since proposal failures are unavoidable even for the rendering algorithms with state-of-the-art mutation strategies, a natural idea comes to mind: can we find a way to minimize the impact of proposal failures? Yes, we can. Our method present a simple but robust remedy for proposal failures without any change in mutation strategies. Instead of trying to decrease proposal failures in mutation strategy end, we take three steps to minimize the impact of proposal failures: (1), discard the proposal failure directly without accumulating any contribution; (2), eliminate the influence of the proposal failure on future proposals; (3), try a new proposal path.

3 OVERVIEW

The main idea of our method is distinguishing proposal failures from normal proposal paths and providing a simple remedy for proposal failures, so that minimizing the overall impact of proposal failures.

First (Section 4), we verify our method in math and present a novel MCMC, Proposal Failure MCMC (PFMCMC), which simulates proposal failures by setting function value $f(X_t)$ to 0 with failure probability p_f and provides a special remedy for proposal failures. In the remedy, *MaxRep* indicates the maximum number of repetition to avoid a proposal failure and "$MaxRep = 0$" means no special treatment for proposal failures. In the example based on normal distribution function, we can assign different value to p_f to simulate scenes with different ratio of proposal failure and to watch the results sampled with or without the remedy available.

Second (Section 5), we combine PFMCMC with MMLT and present a novel light transport algorithm, Proposal Failure Metropolis Light Transport (PFMLT). In order to exhibit the advantages of PFMLT over MMLT, we test several different scenes with complex materials, geometry and lighting. In general, PFMLT produces much better results than MMLT does, as shown in Figure 1 and Figure 3.

4 METHOD

An adaptive MCMC algorithm would automatically tune the parameters of its proposal distribution, [Brooks et al. 2011; Shaby and Wells 2010]. Some adaptive MCMC algorithms have been developed for various problems in different fields, such as [Giordani and Kohn 2006; Haario et al. 2001; Roberts and Rosenthal 2009; Vihola 2012]. Unfortunately, all of these adaptive algorithms are either too complicate or too inefficient for rendering. Furthermore, all of them consider that drawing proposal value from current state of Markov chain never fails, which is not true for MLT-type rendering algorithms. For example, Multiplexed Metropolis Light Transport (MMLT) [Hachisuka et al. 2014] has trouble searching light-carrying paths in Breakfast scene (As exemplified in Figure 1): about 37 percent of the generated paths don't carry any radiance. These zero-radiance paths correspond to the failures of proposal. Without mechanism of failure handling, the state-of-the-art MLT-type algorithms still just reject the proposal failures. Rejection is equivalent to generating a sample in the same place as current state, so this kind of rejections result extra noises in the final image.

Here, we come up with a simple but efficient adaptive MCMC algorithm specially for light transport model. We call it Proposal Failure MCMC (PFMCMC), which simulates proposal failures by setting function value $f(X_t)$ to 0 with failure probability p_f and provides a special remedy for proposal failures.

4.1 Math

Our algorithm, PFMCMC, restricts focus to the case of proposal failure. PFMCMC is an extension of Metropolis Hastings (MH) [Hastings 1970; Metropolis et al. 1953], one of the most popular, classical MCMC algorithms. PFMCMC proceeds according to Algorithm 1. We aggregate some important terms of our notation in Table 1 for reference.

From Algorithm 1, we can see that the basic structure of PFMCMC is similar to MH's. The major modification is that PFMCMC introduces probability p_f to simulate proposal failure, which is corresponding to the red part (line 5 to 8) in the pseudocode of Algorithm 1, and provides a remedy for the failure situation in the blue part (line 9 to 14).

Proposal Failure Simulation. $f(X_t)$ is the function of sample X_t (we think of samples with function values being zero as proposal failures). p_f indicates the overall probability of proposal's random failure. Generally, the percentage of zero-contribution paths in MMLT for our test scenes, like Breakfast scene in Figure 1, is between 35% and 65%. So, usually, reasonably, users can set p_f at a value in $[0.35, 0.65]$ to verify the effectiveness of PFMCMC.

Proposal Failure Remedy. The basic idea of the remedy for proposal failure is discarding the current failure proposal and proposing again expecting a success within the maximum times of repeating

Table 1. Table of notation

Symbol	Explanation
X	Sample result set
X_0	Initial sample
X^*	Proposal sample
X_t	Current sample
X_{t+1}	Next sample
$f(X^*)$	Function value of proposal sample
$f(X_t)$	Function value of current sample
$sTotalNum$	Set number of total proposal
$rTotalNum$	Real number of total proposal
$TotalRep$	Total number of repetition for proposal failure
t	Proposal counter
p_f	Probability of proposal failure
$MaxRep$	Maximum number of repetition
CNT	Repetition counter
α	Acceptance rate

Require: Set initial values X_0, $sTotalNum$, failure possibility p_f and the maximum number of repeating $MaxRep$.
Ensure: output sample set X

$CNT \leftarrow 0$
for $t = 0$ to $sTotalNum$ **do**
  Draw a proposal value X^* from X_t
  Calculate $f(X^*)$
  Draw $v \sim U(0,1)$.
  if $v < p_f$ **then**
    $f(X^*) \leftarrow 0$
  end if
  if $(f(X^*) == 0)$ and $(CNT < MaxRep)$ **then**
    $t--$
    $CNT++$
    continue
  end if
  $CNT \leftarrow 0$
  Calculate acceptance rate α. $\alpha = \min\{1, \frac{f(X^*)}{f(X_t)}\}$
  Draw $u \sim U(0,1)$.
  if $u < \alpha$ **then**
    $X_{t+1} \leftarrow X^*$
  else
    $X_{t+1} \leftarrow X_t$
  end if
end for
return X

Algorithm 1. *Proposal Failure MCMC (PFMCMC).* This is an extension of MH sampling, which simulates proposal failures by setting function value $f(X_t)$ to 0 with failure probability p_f (as shown in the red part) and gives a special remedy to proposal failures (as shown in blue part).

$MaxRep$. If the current proposal X^* fails, we discard it and start a new proposal all over again. If we have tried $MaxRep$ times before getting a success, we don't try any more and move forward. In the process, CNT acts as the counter of failure and repetition. This mechanism of proposal repetition which increases the total number of proposal seems to be inefficient, but it turns out to be more efficient because of the fact that it substantially decreases the ratio of zero-function-value samples. So, this remedy ensures that the overall distribution of samples is much more approximately proportional to the contribution that the samples make to the target probability distribution function.

Note that discarding proposal failures includes two major operations. First, do not accumulate the contribution of the current path. Second, shift t backward (as $t--$ in Algorithm 1), so that clear modifications caused by the current proposal failures. The second operation is very important when t is used in the process of drawing a proposal value X^* from X_t, because this operation actually prevents next proposal value X^* from the impact of the current proposal failure.

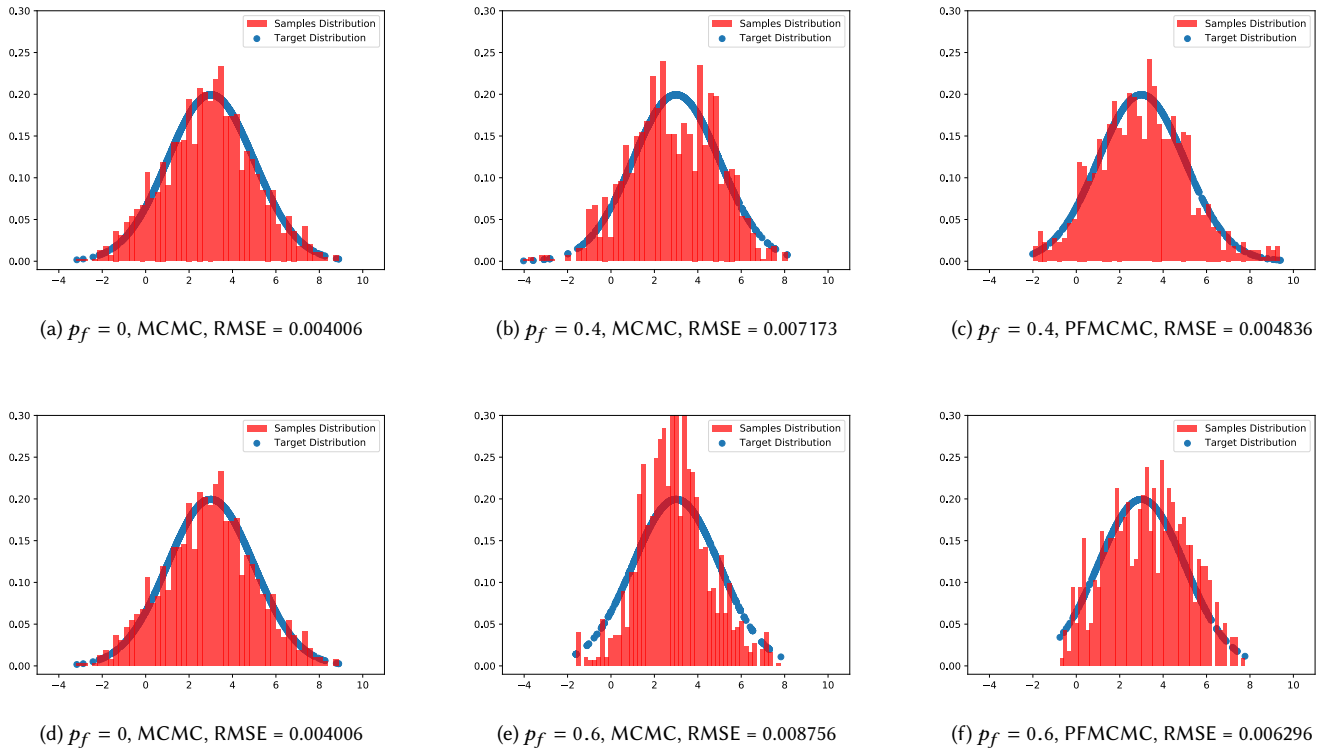
4.2 Example

In order to illustrate the effect of PFMCMC, we'll show how it can be used to sample a 1D model based on a normal distribution function with a probability p_f of proposal failure. The effect comparison of PFMCMC and MCMC is based on equal sampling time. Considering different systems, such Mac, Windows and Linux, have different performance, we use the total number of proposal $rTotalNum$ as an indicator of sampling overhead, which is more reliable because of the elimination of influences of system performance. For MCMC, $rTotalNum$ equals to $sTotalNum$. For PFMCMC, $rTotalNum$ is the additive result of $sTotalNum$ and $TotalRep$ which is related to p_f and $MaxRep$.

Parameter Setting. We are assuming that, for the normal distribution function, the mean is 3 and the standard deviation is 2. Considering that the percentage of proposal failure for most of scenes rendered with MMLT locates in the interval $[0.4, 0.6]$, we test two cases with p_f set at 0.4 and 0.6 respectively. $MaxRep$ is fixed at 4. For MCMC, $sTotalNum$ is set at 1600. For PFMCMC, in order to make $rTotalNum$ close at 1600, $sTotalNum$ setting is more complicate, which is affected by p_f: when p_f is 0.4, we set $sTotalNum$ at 960; when p_f is 0.6, we set $sTotalNum$ at 690.

Results Analysis. Figure 2 demonstrates the sample results of MCMC and PFMCMC. (a) and (d) show reference sample results sampled with MCMC in the condition of zero-p_f, which is the case of normal MH sampling; (b) and (e) show the results sampled with the extension of MCMC, which includes failure simulation and no remedy; (c) and (f) show the results sampled with the extension of MCMC, which is our PFMCMC including both failure simulation and remedy. From (b) and (c), we can see that PFMCMC gets better results than MCMC does when p_f being set to 0.4. From (e) and (f), we can see that PFMCMC also gets better results than MCMC does when p_f being set to 0.6.

Because of the use of correlated samples, a single run of an MCMC integrator may not be representative. We test every case for 100 times and average the corresponding results, which are



(a) $p_f = 0$, MCMC, RMSE = 0.004006 (b) $p_f = 0.4$, MCMC, RMSE = 0.007173 (c) $p_f = 0.4$, PFMCMC, RMSE = 0.004836

(d) $p_f = 0$, MCMC, RMSE = 0.004006 (e) $p_f = 0.6$, MCMC, RMSE = 0.008756 (f) $p_f = 0.6$, PFMCMC, RMSE = 0.006296

Fig. 2. *PFMCMC is used to sample 1D normal distribution model with proposal failure probability p_f.* The total number of proposal is used as an indicator of sampling overhead. All of these results are produced with equal number of total proposal, 1600. (a) and (d) show reference sample results sampled with MCMC in the condition of zero-p_f, which is the case of normal MH sampling; (b) and (e) show the results sampled with the extension of MCMC, which includes failure simulation and no remedy; (c) and (f) show the results sampled with the extension of MCMC, which is our PFMCMC including both failure simulation and remedy. From (b) and (c), we can see that PFMCMC gets better results than MCMC does when p_f being set to 0.4. From (e) and (f), we can see that PFMCMC also gets better results than MCMC does when p_f being set to 0.6.

Table 2. Table of average result of 100 times

Reference (p_f==0, MCMC)	$rTotalNum$: 1600 RMSE: 0.004983	
	p_f==0.4	**p_f==0.6**
MCMC	$rTotalNum$: 1600 RMSE: 0.006834	$rTotalNum$: 1600 RMSE: 0.008760
PFMCMC, (similar $rTotalNum$)	$rTotalNum$: 1584 RMSE: 0.006370	$rTotalNum$: 1589 RMSE: 0.007728
PFMCMC, (similar RMSE)	$rTotalNum$: 1383 RMSE: 0.006787	$rTotalNum$: 1262 RMSE: 0.008686

recorded in Table 2. Row "PFMCMC, (similar $rTotalNum$)" of the table shows that PFMCMC produces higher quality results than MCMC with similar overhead ($sTotalNum$), which supports the results shown in Figure 2 well. Less RMSE means higher quality. But, RMSE numbers may not be intuitive enough. In order to fully exhibit the superiority of PFMCMC, we tune its parameters so that it produces similar quality results to MCMC, which is shown in Row "PFMCMC, (similar RMSE)" of the table. We can see that PFMCMC pays much less overheads to achieve results similar to MCMC's. For example, in the case "p_f==0.4", PFMCMC only pays 1383/1600($\approx$ 0.8643) of MCMC's overheads, and in the case "p_f==0.6", the ratio is 1262/1600($\approx$ 0.7888).

5 IMPLEMENTATION

Although we can integrate PFMCMC framework with any current MLT-type algorithms with mutation in primary sample space or path space, in this section, we will describe our rendering algorithm, Proposal Failure Metropolis Light Transport (PFMLT), by recasting MMLT ([Hachisuka et al. 2014]) in the framework of PFMCMC. MMLT improves rendering efficiency by selecting seed path with probability proportion to its contribution to the final image. Fusing MMLT with the idea of proposal failure remedy of PFMCMC makes PFMLT even much more efficient than MMLT. For example, rendering Breakfast scene (As exemplified in Figure 1) with PFMLT saves about 30 percent of time to produce the same quality image as MMLT does. Algorithm 2 shows the pseudocode of PFMLT.

Require: The maximum number of repeating *MaxRep*.
Ensure: Accumulation of Path Contributions
Sample a seed path $\bar{s}$ from Initial Path Set
Calculate the depth of $\bar{s}$, k and path contribution $f(\bar{x}_s)$
$\bar{s}$ is used as current path $\bar{x}$: $\bar{x} \leftarrow \bar{s}$, $f(\bar{x}) \leftarrow f(\bar{x}_s)$
$CNT \leftarrow 0$
$IsFailure \leftarrow 0$
if *MaxRep* is not given **then**
  $MaxRep \leftarrow (k+2)$
else
  $MaxRep \leftarrow min\{MaxRep, k+2\}$
end if
for $t = 0$ to $sTotalNum$ **do**
  Draw a proposal vector of random numbers $\bar{v}$ from $\bar{u}$.
  $t \leftarrow \text{int}((k+2)v_s)$
  $s \leftarrow (k+1) - t$
  if !($\bar{y}_{camera} \leftarrow$ SampleCameraSubpath($\bar{v}_{camera}, t$)) **then**
    $IsFailure \leftarrow 1$
  end if
  if !($IsFailure$) **then**
    if !($\bar{y}_{light} \leftarrow$ SampleLightSubpath($\bar{v}_{light}, s$)) **then**
      $IsFailure \leftarrow 1$
    end if
  end if
  if !($IsFailure$) **then**
    if !($f(\bar{y}) \leftarrow$ Connect($\bar{y}_{camera}, \bar{y}_{light}, \bar{v}_{connect}$)) **then**
      $IsFailure \leftarrow 1$
    end if
  end if
  if ($IsFailure$) and ($CNT < MaxRep$) **then**
    $t--$
    REJECT()
    $CNT++$
    $IsFailure \leftarrow 0$
    Continue
  end if
  $CNT \leftarrow 0$
  $IsFailure \leftarrow 0$
  Calculate acceptance rate α. $\alpha = \min\{1, \frac{L(f(\bar{y}))}{L(f(\bar{x}))}\}$
  Draw $u \sim U(0,1)$.
  if $u < \alpha$ **then**
    $\bar{x} \leftarrow \bar{y}$
    ACCEPT()
  else
    REJECT()
  end if
  AccumulatePathContribution($f(\bar{x}), f(\bar{y})$)
end for

Algorithm 2. *PFMLT (Proposal Failure MLT).* The blue part is the simple remedy for proposal failures. When proposal failure is detected, we'll try at most $MaxRep$ times to avoid it, so that to minimize its impact on the final image.

Initialization. First, the same as MMLT, we sample a seed path $\bar{s}$ from Initial Path Set and calculate it's depth k and contribution $f(\bar{x}_s)$. The depth k should be emphasized here, like MMLT, all of subsequent proposal paths are of the same depth as this seed path. Seed path $\bar{s}$ should be assigned to current path $\bar{x}$. $sTotalNum$ can be obtained by dividing total proposal number by total number of Markov chains. Then, some special variables for implementing the core idea of PFMCMC should be initiated. Both proposal failure flag $IsFailure$ and repeating counter CNT are initiated as 0. As to the maximum number of repeating $MaxRep$, if its value is not given by users, we set it at $k+2$, which is an experiment value that makes PFMLT perform well enough. If users provide a value for $MaxRep$, the smaller value in the user-value and $k+2$ will be used as the final $MaxRep$. This mechanism enables users to effectively adjust $MaxRep$ for some extremely challenging scene to produce better results.

Random Numbers Proposal. For MMLT, the same as PSSMLT, two main steps are needed to draw a proposal $\bar{y}$ from $\bar{x}$ in path space. Firstly, draw a proposal vector of random numbers $\bar{v}$ from $\bar{u}$ which is the primary sample space counterpart of $\bar{x}$. Secondly, obtain $\bar{y}$ by path sampling in path space using the proposal vector of random numbers $\bar{v}$. As to the first step, note that we apply normally distributed perturbations to each component of the vector of random numbers. The advantage of sampling with a normal distribution like this is that it naturally tries a variety of mutation sizes. It preferentially makes small mutations that remain close to the current state, which help locally explore the path space in small areas of high contribution.

Path Sampling and Potential Failures. As mentioned in "Random Numbers Proposal" part, the second step of path proposal is path sampling. It is this step where proposal failures occur. The proposal vector of random numbers $\bar{v}$ has four main uses in path sampling. First, one random number, v_s, of $\bar{v}$ is used to choose sample technique, so that t and s are determined. Second, a sub-vector, $\bar{v}_{camera}$, of $\bar{v}$, along with t, is used to sample a camera sub-path $\bar{y}_{camera}$ with depth exactly being t. Third, a sub-vector, $\bar{v}_{light}$, of $\bar{v}$, along with s, is used to sample a light sub-path $\bar{y}_{light}$ with depth exactly being s. Fourth, a sub-vector, $\bar{v}_{connect}$, of $\bar{v}$ is used to connect $\bar{y}_{camera}$ and $\bar{y}_{light}$ to make a complete proposal path $\bar{y}$ and to calculate proposal path contribution $f(\bar{y})$.

Except for the first stage, the other three are of potential failures. In the sampling sub-path stages, the depth of camera sub-path $\bar{y}_{camera}$ may not be t, or the depth of light sub-path $\bar{y}_{light}$ may not be s, so failures happen. In the connecting stage, the connection between the two sub-paths, $\bar{y}_{camera}$ and $\bar{y}_{light}$, may be blocked, which is a very high probability event. The existence of these failures is the very reason why PFMLT is much more efficient than MMLT.

Failure Remedy. $IsFailure$ is used to indicate whether any failure has happened. If any failure is detected, the flag $IsFailure$ is set to 1. Note that if a failure has been detected earlier, the latter path sampling programs are not necessary to run.

MMLT, which does not give any special care to proposal path failures, just rejects the failure proposal path and accumulates the

contribution of the current path. However, neither the contribution of the current path nor the contribution of the proposal failure path is proportional to the probability with which the proposal failure path was sampled, which surely introduces a lot of extra noises.

Our method, PFMLT, will reduce this kind of noise substantially. If failure happens in the process of obtaining proposal path $\bar{y}$, we don't even get into the step of calculating acceptance, not to mention repeatedly accumulating the contribution of the current path. We just discard the whole current proposal, including the trace of the proposal vector of random numbers $\bar{v}$, and try a new proposal base on the current path, which is shown in Line 28 to Line 34 of Algorithm 2. The maximum number of this kind repeating is $MaxRep$. If we have try $MaxRep$ path proposals of some current path, we don't try any more and just move forward like MMLT does, because we find that too many repetitions will decrease overall efficiency. As mentioned in Initialization part, the default value of $MaxRep$ is set at $k+2$, and users can tune the value to reach an even higher efficiency for some extremely challenging scene. When we move forward, both proposal failure flag $IsFailure$ and repeating counter CNT should be reset.

Again, note that Line 29 in Algorithm 2, "$t--$", is very important. Because t is used to draw proposal vector of random numbers $\bar{v}$ from $\bar{u}$ in primary sample space. "$t--$" is the very operation that prevents the new proposal "$\bar{v}$" from the impact of the current proposal failure. We show the related test images in Appendix A.

Accept Probability. Considering that e used the same mutation function for getting the vector of random numbers $\bar{v}$ from $\bar{u}$ as MMLT. Since these mutations are all symmetric, transition probability density functions are not needed to evaluate. The acceptance probability α is simply the ratio of the luminosities of proposal path contribution $f(\bar{y})$ and current path contribution $f(\bar{x})$.

Contribution Accumulation. The same as MMLT, the scaling by the reciprocal of the discrete probability density of the selected path length as noted before, we also need to scale each contribution by the number of techniques $k+2$. This scaling corresponds to the fact that the chain explores $k+2$ different sub-spaces.

6 RESULTS

This section includes two subsections, "Main Results" and "Extra Results". In the "Main Results", we implement our method by extending the system of PBRT, and we compare against MMLT implemented in the same system. In the "Extra Result", we show that our method can be easily used to extend other systems and extend both RJMLT and MMLT in Tungsten [Bitterli 2017], a renderer that is a much simpler and faster than PBRT, and we compare against MMLT and RJMLT implemented in Tungsten.

6.1 Main Results

We implement our method by extending the system of PBRT, and we compare against MMLT implemented in the same system. Five scenes — Breakfast (1024 × 1024), Villa (1200 × 580), Bathroom (1280 × 720), Bidir (768 × 576), Living Room (1280 × 720) — with different geometry, lighting and material configurations are rendered using Mac pro with Intel Core i5 at 2.7GHz. Villa scene reference is rendered using MMLT in PBRT and the references of the other four scenes are rendered with BDPT in PBRT. Rendering each of those references costs several days of computation. We set the maximum path length at 9 for Living Room scene and at 5 for the other four scenes. The value of $MaxRep$ is set at $k+2$ by default.

Equal-time Comparison. We show the image comparison results in Figure 1 and Figure 3. To make equal-time comparisons between MMLT and our method, we rendered all the five scenes. Because of extra overheads for failure remedy in our method, the parameter of mutations per pixel should be set at a smaller value than in MMLT, as show Column "Mutations Per Pixel" of Table 3.

In order to obtain the statistic data of these comparisons, we define some counters in Algorithm 2: $TotalPaths$ is the total number of paths traced; $RemedyPaths$ is the number of paths traced for the remedy of proposal failures, which is the extra overhead of our method; $FinalFailurePaths$ is the number of final failure paths after remedy; $AcceptancePaths$ is the number of paths accepted. All of these counters should be initialized to 0 before the beginning of rendering (Line 11 of Algorithm 2). In the Algorithm 2, we add some other code for statistics: "$TotalPaths++$;" at Line 12; "$RemedyPaths++$;" at Line 29; "$if(\alpha < 0.000001)FinalFailurePaths++$;" at Line 37; "$AcceptancePaths++$;" at Line 40. We use the following equations to calculate $FailureRate$ and $AcceptanceRate$.

$$FailureRate = \frac{FinalFailurePaths}{(TotalPaths - RemedyPaths)}$$

$$AcceptanceRate = \frac{AcceptancePaths}{(TotalPaths - RemedyPaths)}$$

These statistics are also recorded in Table 3.

Scenes	Algorithms	Mutations Per Pixel	Failure Rate	Acceptance Rate	RMSE
Breakfast	MMLT	150	37.37%	49.38%	0.082572
	Ours	100	1.54%	77.84%	0.069976
Villa	MMLT	200	62.82%	32.39%	0.083772
	Ours	80	10.47%	75.26%	0.070404
Bathroom	MMLT	280	40.00%	57.00%	0.120891
	Ours	150	5.88%	89.01%	0.103564
Bidir	MMLT	75	37.40%	57.82%	0.067776
	Ours	45	1.86%	90.18%	0.056806
Living Room	MMLT	670	51.12%	45.07%	0.129880
	Ours	310	14.25%	78.35%	0.102854

Table 3. Table of statistics of equal-time comparison (2 hours for Breakfast, 1.5 hours for Villa, 1 hour for Bathroom, 25 minutes for Bidir, and 110 minutes for Living Room). Because of extra overheads for failure treatment in our method, the parameter of mutations per pixel should be set to a smaller value than in MMLT. We also show the comparisons of failure rate, acceptance rate and RMSE for these scenes. From the table, we can see that our method always produce better results (smaller RMSE) than MMLT does.

Breakfast Scene. Figure 1 shows an equal-time (2 hours) comparison on the Breakfast Scene which is illuminated by two lamps. Part of the scene, like the objects on the top of the desk, get direct illumination, however, the lighting for objects under the desk or over the lamps is much more complicated. From Figure 1 and Table 3, we can see that MMLT shows suboptimal performance because about 37.37% of the paths that were used to reconstruct image carried no radiance. Giving special treatment to zero-contribution paths, our

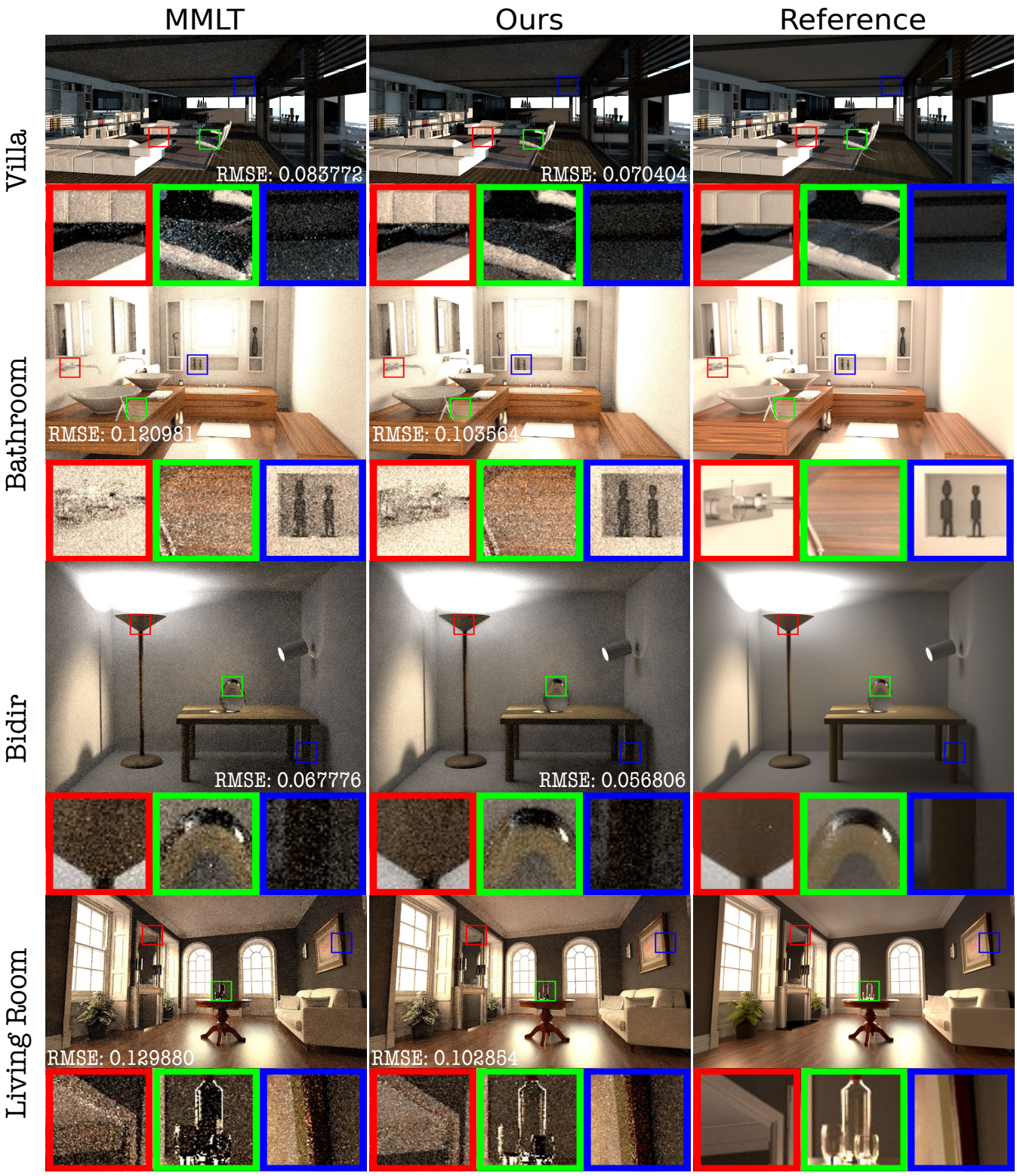


Fig. 3. *Equal-time comparisons.* These challenging scenes, Villa, Bathrooom, Bidir and Living Room, are of various complex material, lighting and geometry. RMSE is used as the indicator of image quality. Our method always produce better results, with smaller RMSE, than MMLT does.

method reduces the percentage to around 1.54% and exhibits much better results with smaller RMSE. Our method distinguishes itself in those difficult settings where a large fraction of all of the possible proposal paths fail to carry any radiance in MMLT rendering, as shown in the three insets of Figure 1.

Villa Scene. The first part of Figure 3 shows an equal-time (1.5 hours) comparison on the Villa scene with complex materials and a difficult geometry configuration lit by outside environment daylight. This is a challenging scene because of the hard-to-find specular-diffuse-specular (SDS) light paths between the villa interior and the near-specular glass windows. The reference rendered by MMLT with 10000 mutations per pixel is still slightly noisy after several days of computation. Also, from Figure 3 and Table 3, we can see that MMLT produces a lot of noises because more than 62.81% of the paths that were used to reconstruct image were zero-contribution. Our method considers zero-contribution proposal as failure and gives special treatment to it. As a result, the percentage was reduced to about 14.47 and higher quality image was obtained. If our method is used to render the image with the same RMSE of 0.083772 like MMLT does, more than 37% of the time will be saved.

Bathroom Scene. The second part of Figure 3 shows an equal-time (1 hour) comparison on the Bathroom scene which contains several different materials including diffuse, specular, and glossy. The scene is illuminated with a large area light source directly visible from the camera. Unlike the Villa scene, the major part of the scene is directly illuminated by the light source. Even in such a simple lighting situation, our method can still produce a better image than MMLT does in equal time. If our method is used to render the image with the same RMSE of 0.120981 like MMLT does, more than 45% of the time will be saved.

Bidir Scene. The third part of Figure 3 shows an equal-time (25 minutes) comparison on the Bidir scene which is very classical. The illumination resembles the Breakfast scene whose part of objects get direct lighting and other parts not. If our method is used to render the image with the same RMSE of 0.067776 like MMLT does, about 28% of the time will be saved.

Living Room Scene. The last part of Figure 3 shows an equal-time (110 minutes) comparison on the Living Room scene. Light coming from outside environment enter the room through glass widows like the Villa scene does. Again, this kind of lighting makes rendering the scene a challenging task. What's more, the materials of the Living Room scene is more complex. The floor, the table and the paneling are made of substrate material, a layered model that varies between glossy specular and diffuse reflection depending on the viewing angle. The cups and the bottle on the table are glass. Other objects like the big mirror and the brushed stainless steel lampshades also contribute to the complication of materials of the scene. All of this result that more than 51.12% of all paths that were used to reconstruct image in MMLT don't carry any radiance. Again, our method is even more efficient in rendering scenes with complex material and lighting. If our method is used to render the image with the same RMSE of 0.129880 like MMLT does, about 47% of the time will be saved.

Convergence Comparison. To make our method more convincing, we did a sequence of comparisons with different computations and compared the convergence of MMLT and our method. We used ABCDEF and OPQRST to number the resulting images rendered by MMLT and our method respectively. The mutations per pixel of the images rendered with similar time were set as Table 4. The resulting comparison images were shown in Figure 4-(a).

MMLT	A	B	C	D	E	F
	20	40	85	170	340	670
Ours	O	P	Q	R	S	T
	10	20	35	75	150	310

Table 4. Table of parameter setting of mutations per pixel for convergence comparison.

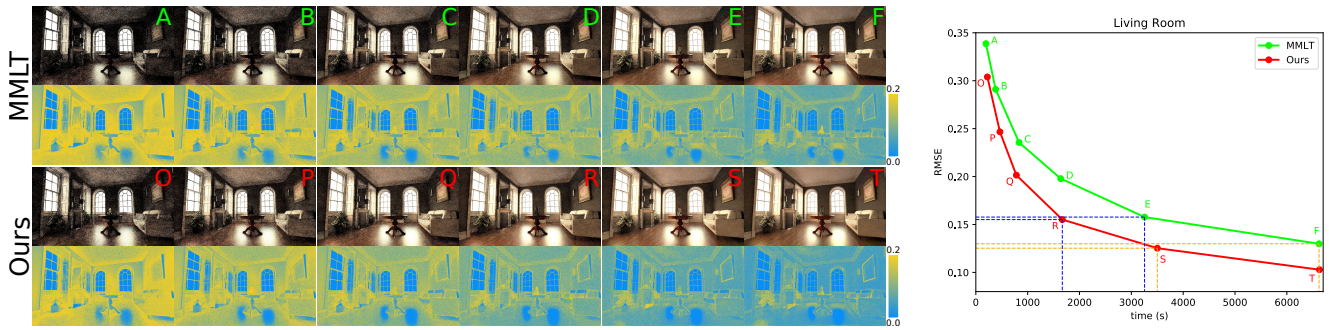
Based on the comparison images and their error images in Figure 4-(a), we can see that our method converges much faster than MMLT does. To make this statement clearer, we calculated the RMSEs of each of these images and exhibited them in line chart as shown in Figure 4-(b).

As the blue dash lines mark in Figure 4-(b), the RMSE of image R rendered with our method is slightly smaller than the RMSE of image E rendered with MMLT, but the rendering time of image R is just about half of image E. Also, as the orange dash lines show the comparison the RMSE of image S and image F, our method saves even bigger ratio of time. So, we can deduce that: *the higher same-quality of images rendered by MMLT and our method, the bigger ratio of time will be saved by our method.*

6.2 Extra Results

In order to further verify the effectiveness of our method and the fact that other MLT-type algorithms can be easily extended with our method, we now do some comparison tests with RJMLT [Bitterli et al. 2018] with our method. We extend both RJMLT and MMLT in Bitterli's original source code, Tungsten [Bitterli 2017], a renderer that is a much simpler and faster than PBRT, and get two new algorithms, RJMLT+PF and MMLT+PF ("PF" means Proposal Failure). Considering that the random seeds used in RJMLT are not fixed, the result of any single run of RJMLT may not be representative, so average behavior of many times of rendering with exactly same settings is used to do comparison test with other algorithms, which is the very reason that we don't include RJMLT in all of tests of the prior scenes as shown in Figure 3.

Here, we'll compare equal-time images of the Glass-Of-Water scene rendered with the four algorithms in Tungsten, MMLT, RJMLT, MMLT+PF, RJMLT+PF. We set the maximum path length at 15 and *MaxRep* at 5, which is an empirical value. Expected results should be: MMLT+PF is better than MMLT, and RJMLT+PF is better than RJMLT. First, we produce the two five-minute images of MMLT and MMLT+PF. By the way, five-minute images are of relatively high quality because of the fact that Tungsten is much simpler and faster than PBRT. Second, we find the parameters of RJMLT and RJMLT+PF to produce five-minute images. Third, we run RJMLT and RJMLT+PF with the parameters 50 times to produce 50 images respectively. Fourth, calculate the average RMSEs of RJMLT and RJMLT+PF with their own 50 images respectively. The Results are shown in Figure 5. For RJMLT and RJMLT+PF, while the two images



(a) Images comparisons

(b) Convergence plot

Fig. 4. *Convergence comparisons for the Living Room scene.* Based on the comparison images and their error images in (a), we can see that our method converges much faster than MMLT does. The RMSEs of each of these images are exhibited in line chart as (b). As the blue dash lines show, the RMSE of image R rendered with our method is slightly smaller than the RMSE of image E rendered with MMLT, but the rendering time of image R is just about half of image E. Also, as the orange dash lines show the comparison the RMSE of image S and image F, our method saves even bigger ratio of time.

are specially chosen from their own 50 images, which may not be representative, the RMSEs are the average of 50 results, which is reliable. In this scene, based on the RMSEs, we can see that: both RJMLT and MMLT+PF are better than MMLT; RJMLT+PF is the best (of course better than RJMLT); in particular, we get a bonus: *MMLT+PF is better than RJMLT*, which means that PF makes MMLT not just better than MMLT, but also better than RJMLT.

7 CONCLUSION

We first came up with the concept of *Proposal Failure*, and then presented a novel extension of MCMC, Proposal Failure MCMC (PFMCMC). A general example shown the effectiveness of PFMCMC. Furthermore, we applied PFMCMC to light transport simulation, getting a novel rendering algorithm, PFMLT. PFMLT is able to substantially decrease the rate of zero-contribution paths involved in reconstructing image and increases the overall efficiency a lot, which was verified by 5 challenging scenes. In order to further demonstrate that our algorithm is easy to implement and can be used to extend other MLT-type rendering algorithms, we extended MMLT and RJMLT in Tungsten renderer. With the test on the Glass-Of-Water scene, we exhibited that our method made RJMLT more efficient than the original RJMLT, and even made MMLT more efficient than the original RJMLT. All of these confirmed the general effectiveness of our method, the remedy for proposal failures.

7.1 Limitations and Future Work

During the extensions of our method to other algorithms, the corresponding parameters, like *MaxRep*, may need to be tuned correspondingly to reach optimal results. As a MLT-type rendering algorithm, generally, our method is also only good at rendering challenging scenes which contain complex materials and lighting. Like other adaptive MLT-type algorithms, such as *MEMLT* [Jakob and Marschner 2012], H^2MC [Li et al. 2015], *HSLT* [Hanika et al. 2015] and *GeoMLT* [Otsu et al. 2018], extra computations are needed to address challenging parts of scenes. So, the number of mutations per pixel that can be finished in equal time as non-adaptive algorithms decreases, which may slightly affect the rendering of simple parts of the same scene. Our method cannot change/improve the original nature of the MLT-type algorithm that is extended with our method. For example, RJMLT+PF inherited the nature that the random seeds used are not fixed from RJMLT, which means that the result of any single run of RJMLT+PF may also not be representative, so average behavior of many times of rendering with exactly same settings must be used to show its effectiveness.

Providing remedies to minimize the impact of proposal failure without modifying mutation strategies is a new direction to improve the efficiency of MLT-type algorithms. We just presented a simple remedy in this paper, and we believe that many more efficient and sophisticated remedies can be developed to further decrease the serious impact caused by proposal failures in the future research. Considering that MCMC is widely used in various fields, if necessary, the idea of remedy for proposal failures can be effectively introduced into these fields.

REFERENCES

Benedikt Bitterli. 2017. Tungsten Source Code. (2017). https://github.com/tunabrain/tungsten.

Benedikt Bitterli, Wenzel Jakob, Jan Novák, and Wojciech Jarosz. 2018. Reversible Jump Metropolis Light Transport using Inverse Mappings. *ACM Transactions on Graphics (Presented at SIGGRAPH)* 37, 1 (jan 2018). https://doi.org/10.1145/3132704

S. Brooks, A. Gelman, G. L. Jones, and X. L. Meng. 2011. Handbook of Markov Chain Monte Carlo. *Chapman and Hall/CRC* (2011), 93–111.

P. Giordani and R. Kohn. 2006. Adaptive independent Metropolis-Hastings by fast estimation of mixtures of normals. *Preprint* (2006).

Adrien Gruson, Mickaël Ribardière, Martin Šik, Jiří Vorba, Rémi Cozot, Kadi Bouatouch, and Jaroslav Křivánek. 2017. A Spatial Target Function for Metropolis Photon Tracing. *ACM Trans. Graph.* 36, 1 (2017), 13. https://doi.org/10.1145/2963097

H. Haario, E. Saksman, and J. Tamminen. 2001. An adaptive metropolis algorithm. *Bernoulli* 7 (2001), 223–242.

Toshiya Hachisuka and Henrik Wann Jensen. 2011. Robust Adaptive Photon Tracing using Photon Path Visibility. *ACM Trans. Graph* 30, 5 (2011). https://doi.org/10.1145/2019627.2019633

Toshiya Hachisuka, Anton S. Kaplanyan, and Carsten Dachsbacher. 2014. Multiplexed Metropolis Light Transport. *ACM Trans. Graph. (Proceedings of SIGGRAPH)* 33, 4 (July 2014), 10. https://doi.org/10.1145/2601097.2601138

Johannes Hanika, Anton Kaplanyan, and Carsten Dachsbacher. 2015. Improved half vector space light transport. *Computer Graphics Forum (Proc. EGSR)* 34, 4 (2015), 65–74.

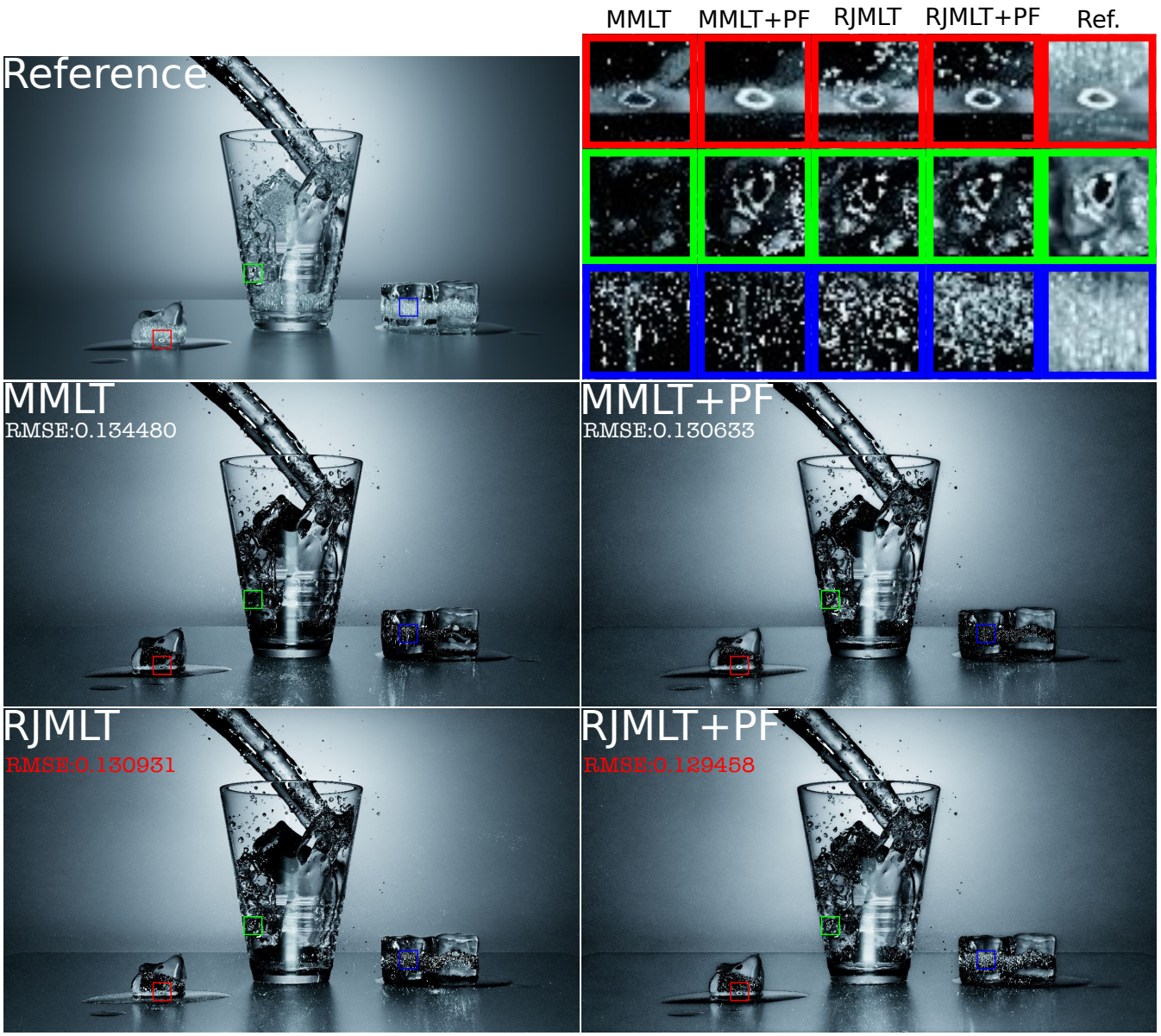


Fig. 5. *Equal-time comparisons with RJMLT.* "MMLT+PF" and "RJMLT+PF" are Proposal Failure extensions of MMLT and RJMLT respectively. Considering that the random seeds used in RJMLT are not fixed, the result of any single run of RJMLT may not be representative, so average behavior of many times of rendering with exactly same settings is used to do comparison test with other algorithms. For RJMLT and RJMLT+PF, while the two images are specially chosen from their own 50 images, which may not be representative, the RMSEs are the average of 50 results, which is reliable. In this scene, based on the RMSEs, we can see that: both RJMLT and MMLT+PF are better than MMLT; RJMLT+PF is the best (of course better than RJMLT); in particular, we get a bonus: *MMLT+PF is better than RJMLT*, which means that PF makes MMLT not just better than MMLT, but also better than RJMLT.

W. K. Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, 1 (1970), 97–109. https://doi.org/10.1093/biomet/57.1.97

Jared Hoberock and John C. Hart. 2010. Arbitrary Importance Functions for Metropolis Light Transport. *Computer Graphics Forum* (2010), 1993–2003. https://doi.org/10.1111/j.1467-8659.2010.01713.x

Wenzel Jakob and Stephen Marschner. 2012. Manifold exploration: a Markov chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Trans. Graph. (Proc. SIGGRAPH)* 31, 4 (2012).

James T. Kajiya. 1986. The Rendering Equation. *In Computer Graphics (Proceedings of SIGGRAPH)* 20 (1986), 143–150. https://doi.org/10.1145/15922.15902

C. Kelemen, L. Szirmay-Kalos, G. ANTAL, and F. CSONKA. 2002. A simple and robust mutation strategy for the Metropolis light transport algorithm. *Computer Graphics Forum* 21, 3 (July 2002), 531–540.

Shinya Kitaoka, Yoshifumi Kitamura, and Fumio Kishino. 2009. Replica Exchange Light Transport. *Computer Graphics Forum* 28, 8 (2009), 2330–2342. https://doi.org/10.1111/j.1467-8659.2009.01540.x

Eric P. Lafortune and Yves D. Willems. 1993. Bi-Directional Path Tracing. *Proceedings of Compugraphics '93* (1993), 145–153.

Eric P. Lafortune and Yves D. Willems. 1996. Rendering Participating Media with Bidirectional Path Tracing. *Proceedings of the 7th Eurographics Workshop on Rendering* (1996), 91–100.

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 Trans. Graph. (Proc. SIGGRAPH Asia)* 34, 6 (2015), 209:1–209:13.

N. Metropolis, N. Metropolis, M. N. Rosenbluth, A. H. Teller, and E. Teller. 1953. Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.* 21, 6 (1953), 1087–1092. https://doi.org/10.1063/1.1699114

Hisanari Otsu, Johannes Hanika, Toshiya Hachisuka, and Carsten Dachsbacher. 2018. Geometry-Aware Metropolis Light Transport. *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia)* 37, 6, Article 278 (2018), 278:1–278:11 pages.

Hisanari Otsu, Anton S. Kaplanyan, Johannes Hanika, Carsten Dachsbacher, and Toshiya Hachisuka. 2017. Fusing State Spaces for Markov Chain Monte Carlo Rendering. *ACM Transactions on Graphics (Proc. of SIGGRAPH)* 36, 4, Article 74 (2017), 74:1–74:10 pages.

Jacopo Pantaleoni. 2017. Charted Metropolis Light Transport. *ACM Transactions on Graphics* 36 (2017).

Gareth O. Roberts and Jeffrey S. Rosenthal. 2009. Examples of Adaptive MCMC. *Journal of Computational and Graphical Statistics* 18, 2 (2009), 349–367. https://doi.org/10.1198/jcgs.2009.06134 arXiv:https://doi.org/10.1198/jcgs.2009.06134

Benjamin Shaby and Martin T. Wells. 2010. Exploring an Adaptive Metropolis Algorithm. *Preprint* (2010).

E. Veach. 1997. Robust Monte Carlo Methods for Light Transport Simulation. *PhD thesis, Stanford University* (1997).

Eric Veach and Leonidas J. Guibas. 1994. Metropolis Light Transport. *In Photorealistic Rendering Techniques (Proceedings of the Eurographics Workshop on Rendering)* (1994), 147–162. https://doi.org/10.1007/978-3-642-87825-1_11

Eric Veach and Leonidas J. Guibas. 1997. Metropolis Light Transport. *In Annual Conference Series on Computer Graphics (Proceedings of SIGGRAPH)* (1997), 65–76. https://doi.org/10.1145/258734.258775

Matti Vihola. 2012. Robust Adaptive Metropolis Algorithm with Coerced Acceptance Rate. *Statistics and Computing* 22, 5 (2012), 997–1008.

Martin Šik, Hisanari Otsu, Toshiya Hachisuka, and Jaroslav Křivánek. 2016. Robust Light Transport Simulation via Metropolised Bidirectional Estimators. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 35, 6 (Nov. 2016), 12. https://doi.org/10.1145/2980179.2982411

A THE IMPORTANCE OF "$T--$"

Line 29 in Algorithm 2, "$t--$", is very important, because t is used to draw proposal vector of random numbers $\bar{v}$ from $\bar{u}$ in primary sample space. "$t--$" is the very operation that prevents the new proposal "$\bar{v}$" from the impact of the current proposal failure. As shown in Figure 4, Image D and Image R are equal-time rendering results of MMLT and PFMLT(with "$t--$") respectively, and as shown in Table 4, the mutations per pixel for Image D and Image R are 170 and 75 respectively. In order to show the importance of "$t--$", we need to produce another rendering result D-R with PFMLT(without "$t--$") with the mutations per pixel as 170.

As shown in Figure 6, the fact that the quality of Image D-R is much lower than Image R shows the importance of "$t--$" operation. Why is Image D-R worse than Image D? Considering that MMLT uses the contribution of the current path for proposal path, while PFMLT(without "$t--$") use zero-contribution for proposal failure, which contribution is more proximately proportional to the probability with which the proposal failure was sampled is of uncertainty. So, whether PFMLT(without "$t--$") produces a better or worse result than MMLT does is normal.

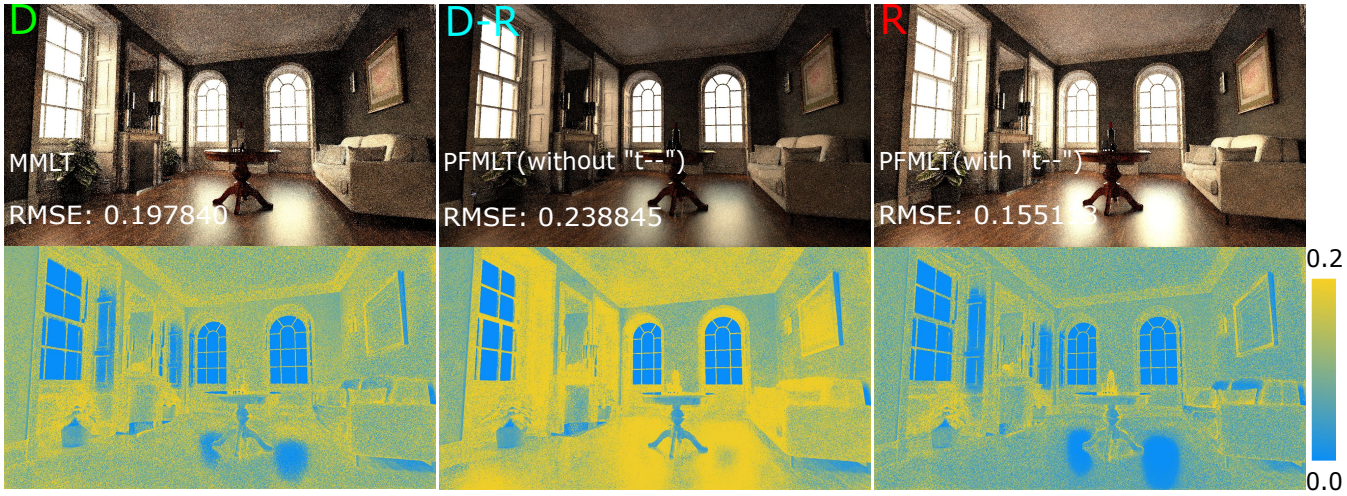


Fig. 6. *Equal-time "$t--$" comparisons.* Image D, Image D-R, Image R are rendered with MMLT, PFMLT(without "$t--$") and PFMLT(with "$t--$") respectively. The fact that the quality of Image D-R is much lower than Image R shows the importance of "$t--$" operation. Why is Image D-R worse than Image D? Considering that MMLT uses the contribution of the current path for proposal path, while PFMLT(without "$t--$") use zero-contribution for proposal failure, which contribution is more proximately proportional to the probability with which the proposal failure was sampled is of uncertainty. So, whether PFMLT(without "$t--$") produces a better or worse result than MMLT does is normal.