EUROGRAPHICS '0x / Jan Kautz and Sumanta Pattanaik
(Editors)

Volume 0 (1981), Number 0

Photorealistic Image Rendering with Population Monte Carlo Energy Redistribution

Y.-C. Lai[1] and S.H. Fan[1] and S. Chenney[2] and C. Dyer[1]

[1]University of Wisconsin at Madison, U.S.A.
[2]Emergent Technology, U.S.A.

Abstract

This work presents a novel global illumination algorithm which concentrates computation on important light transport paths and automatically adjusts energy distributed area for each light transport path. We adapt statistical framework of Population Monte Carlo into global illumination to improve rendering efficiency. Information collected in previous iterations is used to guide subsequent iterations by adapting the kernel function to approximate the target distribution without introducing bias into the final result. Based on this framework, our algorithm automatically adapts the amount of energy redistribution at different pixels and the area over which energy is redistributed. Our results show that the efficiency can be improved by exploring the correlated information among light transport paths.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Raytracing

1. Introduction

To generate a physically correct image involves the estimation of a large number of integrals of path contributions falling on the image plane. It is well known that the integrals have highly correlated integrands. However, a standard Monte Carlo rendering algorithm evaluates the integrals independently. As a result, even a small but important region in the domain is located during the process. This information is lost to other samples because of the independent sampling. Sample reuse is an important technique to reduce the variance by exploiting the correlation between integrals. Markov Chain Monte Carlo algorithms for global illumination, such as Metropolis Light Transport [Vea97] and Energy Redistribution Path Tracing [CTE05], enable sample reuse by mutating existing samples into new ones, but the choice of good mutation strategies is non-trivial and has a major impact on image quality. Population Monte Carlo (PMC) algorithms provides us a tool to reuse the information collected in previous iterations. PMC energy redistribution, adapts the framework of PMC to energy redistirbution algorithm, exploits information from important samples through reuse with a mutation process whose mutation strategy is adapted on-the-fly. It is self-tuning to a large extent.

The PMC energy redistribution algorithm iterates on a population of light transport paths passing through the image plane. The population paths are created by tracing the view rays passing through stratified pixel positions on the image plane by a general Monte Carlo ray tracing algorithm such as path tracing and bidirectional path tracing. In our implementation, we use a general path tracing algorithm. Any information available in the previous iterations can be used to adapt the *kernel function* of each population path that produce a new population based on the current population. The resampling process eliminates part of the population paths and regenerate new paths to achieve ergocity. We carefully design the resampling process to eliminate the well explored or low-contribution paths from the current population and to generate new paths according to the need of exploring the image plane evenly for achieving unbiasedness. As a result, new samples are designed to explore the image plane in an even manner. The procedure is then iterated: sample, iterate, resample, adapt, iterate, resample .... The result is a self-tuning unbiased algorithm which can explore the important paths locally.

Our contribution is a new rendering algorithm, **PMC Energy Redistribution**(PMC-ER), based on the PMC framework. The algorithm adapts the amount of energy redistribution at different pixels and the area over which energy is re-

distributed. For example, pixels near a sharp shadow boundary will not attempt to widely distribute energy, while those in a smooth diffuse image region will distribute energy over a wide area.

The remainder of this paper is organized as follows: section 2 reviews a number of works related to this algorithm. Section 3 presents the generic PMC frame work. Section 4 presents the PMC-ER in detail. Section 5 shows the results generated by this algorithm. Section 6 discusses the limitation and relation to the existing algorithm. Finally, section 7 gives the conclusion of our algorithm.

2. Related Work

Currently, most global illumination algorithms are based on ray tracing and Monte Carlo integration. There exist two catefories: unbiased methods such as [Kaj86, VG94, LW93]; and biased methods such as [WRC88, Hec90, Jen01]. Interested readers can refer to Pharr and Humphreys [PH04] for an overview of Monte Carlo rendering algorithms. Here we only focus on sample reuse which is directly related to this work.

Sample reuse via Markov Chain Monte Carlo (MCMC) algorithms is a powerful means of exploiting hard-to-find light transport paths in global illumination. Metropolis Light Transport (MLT) [Vea97] was the first algorithm to use this approach. MLT replaces the Monte Carlo integrator used in path tracing with a Metropolis sampler. The main advantage of the Metropolis algorithm over Monte Carlo intergration is the ability to preserve the sampling context. This is done by using path mutation to explore path space in a localized way. Thus, when high conbribution paths are found, nearby paths will likely be explored as well.

A number of extensions have been introduced since the original 1997 paper. [PKK00] extended MLT to handle participating media such as smoke and fog. [KSKAC02] made the MLT algorithm more robust by mutating in an abstract space of random numbers rather that on the expected quality. The start-up bias of the MLT was analyzed by [SKDP99] and the analysis of the algorithm was presented in [APSS04]. However, the disadvantage of MLT was and continues to be that very large numbers of samples are required, and stratification is difficult.

Energy redistribution path tracing (ERPT) [CTE05] attempted to address this problem by starting with a well-stratified set of initial samples and locally redistributing energy using MCMC. The noise-reduction techniques they proposed introduce bias. In addition, the extent of redistribution was manually set. [FCL05] used the MLT sampler to take visual importance into account with complete paths from light to eye when distributing photons according to paths' contribution on the final image. Their method solved the difficult path problem such as light passes through a small hole on the wall. However, the bias inherited from photon mapping methods prevents the usage of advanced convergence test mechanism. Our PMC-ER algorithm automatically adapts parameters in an ERPT-like algorithm and uses the adaptation of the kernel functions to locally explore important light transport paths. In addition, the algorithm is unbiased.

Ghosh, Doucet and Heidrich [GDH06] applied the framework of Sequential Monte Carlo algorithm to the problem of sampling environment maps in animated sequences. Their work re-uses samples from previous iteration and is a complementary to our method. However, their work is limited to the environment map. Our algortihm can be applied to more general types of light transport paths.

3. D-Kernel Population Monte Carlo

The Population Monte Carlo algorithm [CGMR04] provides us an iterative importance sampling framework. The distinguishing feature of PMC is that the kernel functions are modified after each step based on information gathered from prior iterations. The kernels adapt to approximate the ideal importance function based on the samples seen so far. While this dependent sampling may appear to introduce bias, it can be proven that the result is either unbiased or consistent, depending on whether certain normalizing constants are known (in our case they are known). The generic D-Kernel PMC sampling algorithm [DGMR05a, DGMR05b] which is an evolution of PMC is stated in Figure 1.

1 generate the initial population, $t = 0$
2 **for** $t = 1, \cdots, T$
3 adapt $K_i^{(t)}(x^{(t)}|x^{(t-1)})$
4 **for** $i = 1, \cdots, N$
5 generate $X_i^{(t)} \sim K_i^{(t)}(x|X_i^{(t-1)})$
6 $w_i^{(t)} = \pi(X_i^{(t)})/K_i^{(t)}(X_i^{(t)}|X_i^{(t-1)})$
7 resampling process: elimination and regeneration

Figure 1: *The generic D-Kernel Population Monte Carlo algorihtm.*

The algorithm works on a population of samples denoted by $\left\{X_1^{(t)}, \ldots, X_N^{(t)}\right\}$, where t is the iteration number and N is the population size, to evaluate $\int_{\mathcal{D}} f(x)dx$, where is $f(x) = \pi(x)h(x)$ by sampling according to the target distribution $\pi(x)$.

The algorithm first creates a set of initial population by using any unbiased sampling method. A *kernel function*, $K_i^{(t)}(x_i^{(t)}|x_i^{(t-1)})$, for each member in the population is adapted in the outer loop. The responsibility of the member kernel fuction is to take the existing member sample, $X_i^{(t-1)}$, as input and produces a candidate new sample, $X_i^{(t)}$, as output (line 5). The resampling step in line 7 is designed to cull candidate samples with low weights and promote high-weight samples. The resampling process consists of two

steps: elimination and regeneration. It is designed to eliminate the samples with low contribution to the final result and to explore new unexplored regions. The weight computed for each sample, $w_i^{(t)}$, is essentially its importance weight. At any given iteration, an estimator of the integral of interest can be computed and is unbiased for $\pi(h)$:

$$\begin{aligned}
\tilde{f}(x) &= \tilde{\pi}(h) = \frac{1}{N}\sum_{i=1}^{N} w_i^{(t)} h(X_i^{(t)}) \\
\mathcal{E}[\frac{1}{N}\sum_{i=1}^{N} w_i^{(t)} h(X_i^{(t)})] &= \frac{1}{N}\sum_{i=1}^{N} E[w_i^{(t)} h(X_i^{(t)})] \\
&= \frac{1}{N}\sum_{i=1}^{N} \int_{\mathcal{D}} \frac{\pi(x)h(x)}{K_i^{(t)}(x|x_i^{(t-1)})} \\
&\quad K_i^{(t)}(x|x_i^{(t-1)})dx \\
&= \frac{1}{N}\sum_{i=1}^{N} \int_{\mathcal{D}} \pi(x)h(x)dx \\
&= \int_{\mathcal{D}} f(x)dx
\end{aligned}$$

It concludes that $\tilde{\pi}(h)$ is an unbiased estimator of $\pi(h)$.

Before apply PMC to rendering problems, we must first answer the following questions:

- What is the sampling domain and how big is population size?
- What is the member function and what is the adaption criteria?
- What techniques are used for sampling from the kernel functions and resampling step?

The following sections describe an application of this framework by mutating the energy redistribution algorithm through answering each question properly. Then, we conclude with a general discussion on PMC for rendering problems.

4. PMC Energy Redistribution (PMC-ER)

PMC Energy Redistribution (PMC-ER) is an algorithm motivated by energy redistribution path tracing (ERPT) [CTE05] that adaptively selects pixels for redistribution, and can also adapt algorithm parameters. ERPT as originally proposed traces a path into the scene from each pixel, using path tracing to form complete light transport paths from the eye to the light. For each pixel, the path is used as the initial state for a Markov Chain Monte Carlo (MCMC) sample chain that redistributes the path's energy to nearby pixels and finds additional light paths. The intuition is that different pixels will find different initial paths, and the information can then be conveyed to neighboring pixels through the Markov Chain. Due to space limitations, we cannot discuss ERPT in detail; readers are referred to the original paper.

ERPT uses the estimation of the energy of the entire image from the path contribution to determine how many constant length chains are needed for every pixel, regardless of how much it differs from its neighbors. In addition, the redistribution region is also fixed and manually set. This is sub-optimal — some pixels that have high variance should take more samples and more time to redistribute its energy, while others are in a neighborhood where most light transport paths are similar and redistribution achieves nothing. To address the former problem, Cline et al. [CTE05] designed filters that introduce bias into the calculation, making the image darker than it should be.

Our PMC-ER algorithm uses the same basic premise as ERPT: high-energy paths should be mutated to distribute the information they carry to neighboring pixels. The sample population is a set of light transport paths through the scene. The kernel functions mutate these paths to create new paths. The resampling step removes low energy or well-distributed paths, keeps high-energy paths and generates new paths to evenly explore regions and adapts the kernel function for each population path. The work is focused on the important transport paths and correlated sampling of the integration domain. In this section, we first present an overview of our two energy redistribution algorithms. The remaining of the section is to explore the implementation detail needed for these two algorithms.

4.1. PMC-ER Equal Deposition Algorithm

Figure 2 shows the PMC-ER equal deposition algorithm. In the preprocess phase, the algorithm first generates a pool of stratified pixel positions used to explore the image plane evenly. This pool of pixel positions is used to generate initial population paths and to generate new stratified replacement paths during the resampling process in each iteration in order to guarantee even exploration of the image plane. Then, the algorithm estimates the average energy contained in the image, $\tilde{E}$, and the deposition energy, e_d, for each mutation which are discussed in [CTE05]. An initial population of paths are created by using the path tracing alogirhtm, the rays of which shoot from the camera and pass through the pixel position, (x,y), selected from the stratified pool. In this work, a path, $\tilde{\mathbf{Y}}$, is referred to as a light transport path starting from a light, **L**, scattering diffusely, **D**, or specularly, **S**, inside the scene several times, and ending at the camera, **E**. The path is denoted as $L(S|D)^*E$. Interested readers can refer to [Hec90, Vea97] for detail. Figure 3 and 4 shows two examples of such paths.

In each inner loop, we do N_{Equal} mutations at each path in the population according to the path's kernel function, $K_i^{(s)}(\tilde{\mathbf{y}}^{(t)}|\tilde{\mathbf{Y}}_i^{(t-1)})$, which is a mixture distribution with adaptable mixture weights, $\alpha_i^{(s)}$ for mixture components. The detail of the kernel function is given in section 4.2. After mutation, the acceptability probability, $A(\tilde{\mathbf{Y}}_i'^{(t)}|\tilde{\mathbf{Y}}_i^{(t-1)})$, is used

to determine whether the path in the population switches to the new generated path, $\tilde{\mathbf{Y}}_i'^{(t)}$, or stays as the original path, $\tilde{\mathbf{Y}}_i^{(t-1)}$, before mutation. Then, e_d energy is deposited on the image plane at the pixel position of the new population path, $\tilde{\mathbf{Y}}_i^{(t)}$.

In the outside loop, the resampling process which is discussed in section 4.3 is to eliminate well-distributed and low-contribution paths, regenerate paths considering the stratification, and adapt the weights, $\alpha_i^{(s)}$, for perturbations with different radiuses.

```
1   generate a pool of stratified pixel position
2   estimate the Ẽ, e_d
3   generate initial population of paths in t = 0
4   for s = 1,···,T
5     determine α_i^(s) for each perturbation
6     for i = 1,···,n
7       if E_i,remain + U(0,1) > Ẽ
8         for t = 1,···,N_mutations
9           generate Ỹ'_i^(t) ~ K_i^(s)(ỹ^(t) | Ỹ_i^(t-1))
10            Ỹ_i^(t) = (U(0,1) < A(Ỹ'_i^(t) | Ỹ_i^(t))) ? Ỹ'_i^(t) : Ỹ_i^(t-1)
11          deposit e_d energy on Ỹ_i^(t)
12          E_i,remain −= e_d
13      w_i^(t) = E_i,remain
14    resample the population: elimination and regeneration
```

Figure 2: *The PMC-ER equal deposition iteration loop. $U(0,1)$ generates a random number uniformly distributed between 0 and 1, and $E_{i,remain}$ is the energy left in the population path, i, after the innter energy redistribution loops.*

4.2. The Kernel Function for Each Path

The kernel function for each population path is a conditional kernel, $K_i^{(s)}(\tilde{\mathbf{y}}^{(t)}|\tilde{\mathbf{Y}}_i^{(t-1)})$, that generates a sample path i in iteration t, $\tilde{\mathbf{Y}}_i'^{(t)}$, given sample i in iteration $t-1$, $\tilde{\mathbf{Y}}_i^{(t-1)}$ (see Figure 2 and **??**). we use a mixture distribution:

$$K_i^{(s)}(\tilde{\mathbf{y}}_i^{(t)}|\tilde{\mathbf{Y}}_i^{(t-1)}) = \sum_{d_j} \alpha_{i,d_j}^{(s)} T(\tilde{\mathbf{y}}^{(t)}|\tilde{\mathbf{Y}}^{(t-1)} : d_j) \qquad (1)$$

Each component, $T(\tilde{\mathbf{y}}|\tilde{\mathbf{Y}} : d)$, mutates an existing path to generate a new one for exploration of the path space according to the perturbing radius, d. Since the ergocity of the algorithm is achieved by tracing paths at stratified pixel positions, the mutation is only used for local exploration. Therefore, $T(\tilde{\mathbf{y}}|\tilde{\mathbf{Y}} : d)$ is only designed to perform a *perturbation* on the member path based on the perturbation radius, d. Lens and caustic perturbation are two good candidates for this job. The following is simple description of these two mechanisms:

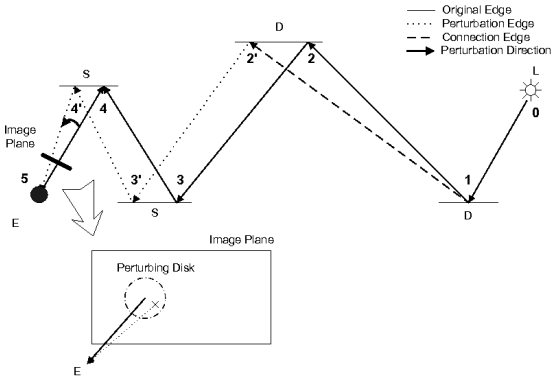

Figure 3: *The top is a path with the form of LDDSSE and used to demostrate the lens perturbation. We would like to replace the lens subpath $\mathbf{y}_5\mathbf{y}_4\mathbf{y}_3\mathbf{y}_2\mathbf{y}_1$ of the form of ESSD. We first perturb the pixel position of the original path at $\mathbf{y}_5$ by uniformly choosing a point from the perturbing disk and then cast a view ray to pass through the new pixel position as showed in the bottom to get $\mathbf{y}_4'$. We extend the subpath through the same specular bounces at $\mathbf{y}_4'$ and $\mathbf{y}_3'$ as the corresponding $\mathbf{y}_4$ and $\mathbf{y}_3$ to get $\mathbf{y}_2'$. Then, $\mathbf{y}_2'$ and $\mathbf{y}_1$ are linked to form a new lens-perturbed path with the same form of LDDSSE as the original one.*

- **Lens perturbation**:

  Figure 3 shows an example of lens perturbation. The lens perturbation is to replace a subpath $\mathbf{y}_{n-1}\cdots\mathbf{y}_k$ of the form $EDS^*(L|D)$. The perburbation takes the existing path and moves the image point which it passes. In our case, the new pixel location is uniformly sampled within a disk of radius, d, a parameter of the kernel component. The remainder of the path is reconstructed to pass through the new image point and extend the subpath through additional specular bounces to be the same length as the original path. The transition probability for lens pertubation can be computed as

$$T_{d,lens}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) = \frac{G(\mathbf{y}'_{n-1},\mathbf{y}'_{n-2})}{A_d} \prod_{j=n-2}^{n-k-2} \frac{G(\mathbf{y}'_j,\mathbf{y}'_{j+1})}{|\cos\theta_{j',in}|}$$

  where $G(\mathbf{y}'_j,\mathbf{y}'_{j+1})$ is the geometric term between $\mathbf{y}'_j$ and $\mathbf{y}'_{j+1}$, A_d is the area of the perturbation, and $\theta_{j',in}$ is the angle between the normal of the surface and the direction of the incoming light ray at $\mathbf{y}'_j$.

- **Caustic perturbation** Figure 4 shows an example of caustic perturbation. The caustic perturbation is to replace a caustic subpath with a suffix $\mathbf{y}_m\cdots\mathbf{y}_k$ of the form $(D|L)S^*D^+E$. To do this, we generate a new subpath starting from the vertex $\mathbf{y}_m$, the head vertex of the caus-

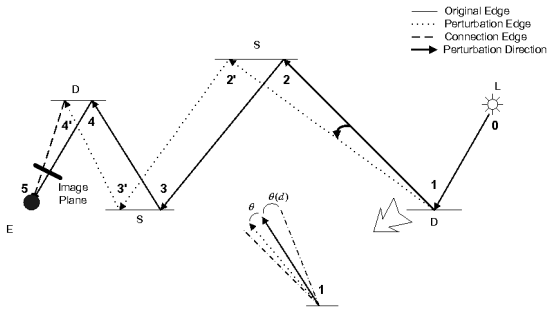

Figure 4: *The top is a path with the form of LDSSDE and used to demostrate the caustic perturbation. We would like to replace the caustic subpath* $\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3\mathbf{y}_4\mathbf{y}_5$ *of the form DSSDE. At the head vertex of the caustic subpath,* $\mathbf{y}_1$*, we perturbed the outgoing light ray direction by an angle,* θ*, uniformly sampled from* $[0,\theta_{max}]$ *to get* $\mathbf{y}'_2$ *as showed in the bottom. We extend the subpath through the same specular bounces at* $\mathbf{y}'_2$ *and* $\mathbf{y}'_3$ *as the corresponding* $\mathbf{y}_2$*and*$\mathbf{y}_3$ *to get* $\mathbf{y}'_4$*. Then,* $\mathbf{y}'_4$ *and* $\mathbf{y}_1$ *are linked to form a new complete caustics-perturbed path with the same form of LDDSSE as the original one.*

tic subpath. The direction of the segment $\mathbf{y}_m \rightarrow \mathbf{y}_{m+1}$ is perturbed by a random amount (θ,ϕ) uniformly sampled from $[0,\theta_{max}]$ and $[0,2\pi]$ where the central axis, $\theta = 0$, corresponds to the direction of the original ray and extend the subpath through additional specular bounces to be the same length as the original one, and θ_{max} is the range of sampling angle computed from correpsonding perturbation radius, d, by the following equation from [Vea97]:

$$\theta_{max} = \theta(d)\frac{|\mathbf{y}_{n-1}-\mathbf{y}_{n-2}|}{\sum_{k=m}^{n-1}|\mathbf{y}_k-\mathbf{y}_{k-1}|} \tag{2}$$

where $\theta(d)$ is the angle through which the ray $\mathbf{y}_n \rightarrow \mathbf{y}_{n-1}$ needs to be perturbed to change the image location by a distance of d pixels. The transition probability for caustic perturbation can be computed as

$$T_{d,caustics}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) = \frac{G(\mathbf{y}_m,\mathbf{y}_{m-1})}{2\pi\theta_{max}\cos\theta_{m,out}}\prod_{j=m-1}^{m-k-2}\frac{G(\mathbf{y}'_j,\mathbf{y}'_{j+1})}{|\cos\theta_{j',out}|}$$

where $\theta_{j',out}$ is the angle between the normal of the surface and the direction of the leaving light ray at $\mathbf{y}'_j$.

In original ERPT work, the size of the pertubation was a parameter to be fixed at startup. In PMC-ER, we can choose a reasonable set of different sized perturbations in the mixture which is three in our case. The large pertubation is effective at redistributing information over a wide area, while the smallest is benefit for image regions where illumination is changing quickly.

When using the kernel function to perturb a path, we first choose d according to the weights, $\alpha_i^{(s)}$, where d is the radius of the lens perturbation and $\sum_{d_{j'}}\alpha_{i,d_{j'}}^{(s)} = 1$. And then either lens or caustic perturbation is chosen according to $\gamma_{lens} = 0.1$ and $\gamma_{caustic} = 0.9$ in our case which is set to prefer caustic perturbation when it is possible. We can then perturb the current path to generate a new perturbed path. The acceptability is to determine whether a path switches to the newly generated path and calculated accordingly as follow:

$$A(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) = \min(1.0,\frac{f(\tilde{\mathbf{Y}})K_i^{(s)}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}})}{f(\tilde{\mathbf{Y}}')K_i^{(s)}(\tilde{\mathbf{Y}}|\tilde{\mathbf{Y}}')}) \tag{3}$$

where $f(\tilde{\mathbf{Y}})$ is the path contribution defined in [VG97]. When evaluating the acceptability probability, all possible proposals that might generate $\tilde{\mathbf{Y}}'$ from $\tilde{\mathbf{Y}}$ should be considered which is:

$$K_i^{(s)}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) = \sum_{d_{j'}}\alpha_{i,d_{j'}}^{(s)}\quad(\ \gamma_{lens}T_{d_{j'},lens}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) + \gamma_{caustic}T_{d_{j'},caustic}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}))\tag{4}$$

However, it is also acceptable to consider only the function derived from the proposal strategy chosen to generate $\tilde{\mathbf{Y}}'$ [Tie98]:

$$K_i^{(s)}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) = T_{d_j,\,op-type}(\tilde{\mathbf{Y}}'|\tilde{\mathbf{Y}}) \tag{5}$$

In this work, we use Equation 5 to avoid the computation of other possible transition functions to improve the efficiency of mutation.

4.3. Resampling

The resampling step in this algorithm achieves three purposes: it carries forward to next round samples that have high energy remaining without flowing out, it provides an opportunity to add some completely new paths into the population for evenly exploring the image space, and the information about which perturbations are chosen inside the inner loop guides the adaption of the kernel functions.

The following decribes these three steps in detail:

- **Elimination**:
  This step is to eliminate well-explored and low-contribution samples from the popluation. When we generate a new population path, the energy of the path, $E(\tilde{\mathbf{Y}})$, is computed and set it to E_{remain}. After each perturbation, we reduce the energy remaining in the path by e_d. The probability of the paths surviving in the elimination process is proportional to the energy

remaining in the path, E_{remain}.

- **Regeneration**:
  Regeneration is to maintain the constant number of paths in the population. It also gives us the chance to decide where we would like to explore in the next iterations. For achieving unbiasedness, we need to evenly explore the image plane. Thus, the regeneration of new paths is according to the criteria of stratification. In the preprocess phase, we compute the total stratified number of pixel positions needed for the entire process. Then a pool of stratified pixel positions is generated according to that number. During the regeneration process, we keep asking the pool to give us the next unused stratified pixel position. A new path is generated by tracing through the new pixel position with the path tracing algorithm and the energy of the path, $E(\tilde{\mathbf{Y}})$, is computed and set it to E_{remain}.

- **Adapt α's Values**
  The purpose of α's valuse is to choose a proper perturbation radius for deciding the area of exploration according to the successes of the perturbations. Thus, when a new path is generated, the $\alpha_{i,k}^{(s)}$ is set to be a constant probability for each component, which allows us to uniformly choose all perturbtations. After initialization, each perturbation acceptability was tagged with the kernel mixture component that generated it and the index of the path in the population. At the adaptation step, we computate the accumulation of the acceptability probabilities tagged with k-th component for each member path and uses it to adjust the mixture probabilities. We can then set:

$$\alpha'_{i,k} = \sum A_{d_j}^{(t)}(\tilde{\mathbf{Y}}_i'^{(t)}|\tilde{\mathbf{Y}}_i^{(t-1)})\delta_{j,k}$$
$$\alpha_{i,k}^{(s)} = \varepsilon + \frac{(1-\varepsilon)\alpha'_{i,k}}{\sum_{k'=1}^{n}\alpha'_{i,k'}}$$

  where $\delta_{j,k} = 1$ if d_k is chosen as the radius of perturbation in step j, i.e. $j = k$

5. Results

We have implemented balance energy transfer algorithm [CTE05] with with the same PMC framework. The results show that although we can improve the bright spots caused by the energy remaining in the original path by keeping the energy that fails to be distributed in the path itself for further exploration at the next iteration, we realize that when finding a high-energy path, the energy being distributed out at the very first step is large comparing to the energy being distributed out in the following iterations. This causes high variance, which is showed as a bright spot, in the final result. This motivate us to develop the PMC-ER equal deposition algorithm. Thus, the results demostrated in this section are generated from the PMC-ER equal deposition algorithm.

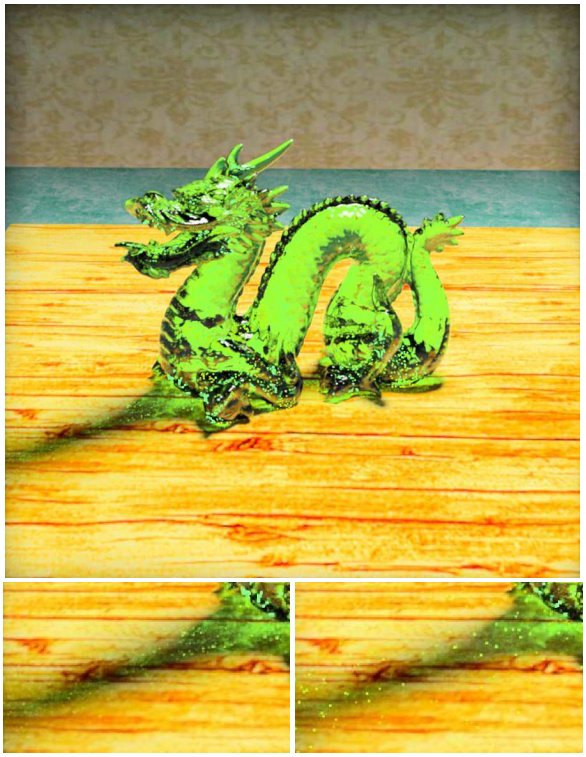

Figure 5: *A dragon scene computed using our PMC-ER equal deposition at the top. The bottom left is the zoom-in of the caustic part computed by PMC-ER equal deposition and the bottom right is the same part computed by ERPT. PMC-ER has fewer artifacts overall. By sharing more information among paths and by better reusing the high contribution paths, PMC-ER is an improvement over ERPT.*

We observe that the deposition energy, e_d, and perturbation radiuses are two important factors for ERPT algorithm. If the e_d is too small, the algorithm becomes too slow and inefficient but it converges to smooth results. However, if e_d is too large, the algorithm generates bright spots because a path must have high energy to pass the distribution criteria to run a MC muation chain, which distributes its energy. However, most paths fail to reach the criteria. In addition, the perturbation radius affects the area where the energy can be diffused to and the success rate of the diffuse operation. In the smooth lighting area, we hope that this radius is large, in order to get a smooth image as soon as possible. However, in complex lighting areas such shadow, caustic regions, we hope that it is small or the rate of success declines largely. Our algorithm automatically adjusts these two aspects through the process of resampling and adapting α's values.

We compared our PMC-ER equal deposition algorithm with the energy redistribution path tracing (ERPT) algorithm on the Cornell Box scene, a dragon scene, and a complex

room scene using the criteria of starting with a similar number of initial PT paths. In all three cases we used a population size of 5000. There are three pertubation radiuses: 5, 10, and 50 pixels, respectively. The caustic perturbation is computed with Eqn. 2. In each step inside the inner loop, each member generates 16 mutations, and 40% of the population is eliminated based on its remaining energy and regenerated using the stratification mechanism. We also use 4 spps for estimating the energy contained in an image for both PMC-ER and ERPT algorithms.

The Cornell Box image (Figure 6) is rendered using our PMC-ER equal deposition algorithm with 1000 iterations which roughly has the same total number of initial PT paths as the image rendered using the ERPT with 8 spps. We can see that our algorithm removes the bright spot artifacts from ERPT algorithm. When we compare our result with an image rendered with ERPT with 16 spps, our image get fewer artifacts. Observing the strategy image whose brightness shows the perturbation count, we see that the probability of paths staying in the population for next iteration are is proportion to its energy remaining. In other words, regions such as the caustic area contained more high energy paths get more number of mutations. In addition, the radius of mutation near physical borders and lighting borders such as, the shadow and caustic area and the light edge, automatically adjusts to increase the success rate of flowing energy out. However, generally, the average time for paths staying in the population is short. Thus, our algorithm cannot have enough time to adjust to the shortest radius at this area. We can only observe a yellow color around the edge instead of a red color for the edge. PMC-ER achieves a visually more converged image compared to the corresponding image generated by the ERPT algorithm with the same number of initial PT paths.

The dragon scene (Figure 5) was rendered at 900×900 with 12800 iterations and 20 mutations for each member in the population inside the loop in comparison with image rendered using ERPT with 32 spps and 20 mutations to each initial PT path. We can see that image rendered using PMC-ER has fewer artifacts than the image rendered using ERPT.

The room scene (Figure 7) was rendered at 720×405 with 19200 iterations and 20 mutations for each member in the population inside the loop in comparison with image rendered using ERPT with 128 spps and 20 mutations to each initial PT path. We can see that image rendered using PMC-ER has fewer artifacts than the image rendered using ERPT. Note that for all PMC-ER equal deposition and ERPT implementations, we did not use the filter proposed in the original ERPT paper to smooth the final image.

The statistics for three rendered images is presented in Table 1. We use the mean squared efficiency (Eff) metric for comparing algorithms, computed as:

Image	Method	Time (s)	Err	Eff
Box1	ERPT(8)	4401.8	0.85	2.7e-4
	ERPT(16)	8935.7	0.526	2.1e-4
	PMC-ER	5281.2	0.37	5.4e-4
Dragon	ERPT(32)	88596.1	1.13	1.0e-5
	PMC-ER	97455.7	0.46	2.3e-5
Room	ERPT(128)	82656.5	0.052	2.3e-4
	PMC-ER	96575.1	0.010	1e-3

Table 1: *Measurements comparing energy redistribution path tracing (ERPT) with PMC-ER, for a roughly equal number of sample rays.*

$$Err = \frac{\sum_{pixels} e^2}{N_{pixels}}, \quad Eff = \frac{1}{T \times Err}$$

where e is the difference in intensity between a pixel, the ground truth value, T is the running time of the algorithm on that image and N_{pixels} is the overal pixel count. Eff is a measure of how much longer (or less) you would need to run one algorithm to reach the quality of another [PH04]. We can see that our algorithm gets better efficiency than ERPT algorithm does.

6. Discussion

The most important variable parameter in our algorithms is the resample rate. A small resample rate reduces the number of samples kept in the population, which results in a faster exploration of the sample domain but at the cost of a large amount of iteration information being lost during the regeneration process. On the other hand, a larger resample rate means that more iteration information related to paths is kept during the iteration. However, the rate to explore the entire sample domain is slow.

Many PMC kernels in the literature are mixture models. Mixtures are typically formed by combining several components that are each expected to be useful in some cases but not others. The adaption step then determines which component are useful for a given input. Mixtures allow otherwise unrelated functions to be combined, such as the perturbation with different sized radiuses. We would prefer the kernel function having many components. However, when the kernel function contains many adaptable parameters, each iteration would requires high adaptive sample counts for gathering proper information to adapt the kernel function. This prevents us from using a larger number of different perturbing radiuses. Such a strategy would be appealing for efficiently rendering a scene with geometries having very different sizes appearing on the image plane, but the adaptive sample count required to adequately determine the mixture component weights would be too large. Instead we use three perturbation radiuses for all images rendered.

The ERPT and PMC-ER algorithms depends on the path tracing algorithm to find proper initial population paths and replacement paths. If the scene consists of paths that are hard to be found but have high energy such as the caustic paths in the dragon scene, although the energy redistribution mechanism can reduce the variance around these high-energy paths, the large difference in initial energy between these high-energy paths and surrounded paths still exists and causes. obvious artifact. In order to remove these artifacts, we must use a larger number of iterations to increase the chance of finding more high-energy paths or use a larger number of mutations per iteration to distribute those energy around to average out.

7. Conclusion

A new global illumination algorithm, PMC-ER, is presented by applying PMC framework to energy redistribution algorithms. PMC-ER learns to become an effective sampler based on the information collected from early iterations. The algorithm automatically explores the important light paths found in the previous iteration, adjusts the area of exploration according to results of previous muations, and also uses resampling to achieve ergocity. There are several future research directions. The PMC-ER should be able to use the perceptual variance as regeneration criteria to focus on the high perceptual variance area. However, the energy brought by a variance path generated this way should also be adjusted accordingly. Also, how to identify a variance coming from an artifact of rendering not from physical and lighting discontinuity is another question. In addition, all paths initialized the α's values to a constant value. However, we can record the alpha used previously in an image because spacial correlation will give us similar α's values in most places in the image plane. We can reuse the α information to reduce the process of probing to estimate a proper set of α's values. Based on the framework of Population Monte Carlo, PMC-ER can improve the rendering efficiency. PMC should be able to provide further research opportunities for global illumination community.

Acknowledgments

PBRT NSF grant CCR-0204372, and equipment donations from Intel.

References

[APSS04] ASHIKHMIN M., PREMOZE S., SHIRLEY P., SMITS B.: A variance analysis of the metropolis light transport algorithm. *Computer and Graphics 25*, 2 (2004), 287–294.

[CGMR04] CAPPÉ O., GUILLIN A., MARIN J.-M., ROBERT C.: Population Monte Carlo. *Journal of Computational and Graphical Statistics 13*, 4 (2004), 907–929.

[CTE05] CLINE D., TALBOT J., EGBERT P.: Energy redistribution path tracing. In *SIGGRAPH '05* (2005), pp. 1186–1195.

[DGMR05a] DOUC R., GUILLIN A., MARIN J. M., ROBERT C. P.: *Convergence of adaptive sampling schemes*. Technical Report 2005-6, University Paris Dauphine, 2005.

[DGMR05b] DOUC R., GUILLIN A., MARIN J. M., ROBERT C. P.: *Minimum variance importance sampling via population Monte Carlo*. Technical report, University Paris Dauphine, 2005.

[FCL05] FAN S., CHENNEY S., LAI Y.: Metropolis photon sampling with optional user guidance. In *Proc. of the 16th Eurographics Symposium on Rendering* (2005), Eurographics Association, pp. 127–138.

[GDH06] GHOSH A., DOUCET A., HEIDRICH W.: Sequential sampling for dynamic environment map illumination. In *Proc. Eurographics Symposium on Rendering* (2006), pp. 115–126.

[Hec90] HECKBERT P. S.: Adaptive radiosity textures for bidirectional ray tracing. In *SIGGRAPH '90* (1990), pp. 145–154.

[Jen01] JENSEN H. W.: Realistic image synthesis using photon mapping. AK Peters.

[Kaj86] KAJIYA J. T.: The rendering equation. In *SIGGRAPH '86* (1986), pp. 143–150.

[KSKAC02] KELEMEN C., SZIRMAY-KALOS L., ANTAL G., CSONKA F.: A simple and robust mutation strategy for the metropolis light transport algorithm. vol. 21, pp. 531–540.

[LW93] LAFORTUNE E. P., WILLEMS Y. D.: Bi-directional path tracing. In *Proceedings of Compugraphics* (1993), pp. 145–153.

[PH04] PHARR M., HUMPHREYS G.: *Physically Based Rendering from Theory to Implementation*. Morgan Kaufmann, 2004.

[PKK00] PAULY M., KOLLIG T., KELLER A.: Metropolis light transport for participating media. In *Proc. of the 11th Eurographics Symposium on Rendering* (2000), Eurographics Association, pp. 11–22.

[SKDP99] SZIRMAY-KALOS L., DORNBACH P., PURGATHOFER W.: *On the Start-up Bias Problem of Metropolis Sampling.* Technical report, Univ.of Plzen, 1999.

[Tie98] TIERNEY L.: A note on Metropolis-Hastings kernels for general state spaces. *The Annals of Applied Probability 8*, 1 (1998), 1–9.

[Vea97] VEACH E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.

[VG94] VEACH E., GUIBAS L. J.: Bidirectional estimators for light transport. In *Proc. of the 5th Eurographics Workshop on Rendering* (1994), Eurographics Association, pp. 147–162.

[VG97] VEACH E., GUIBAS L. J.: Metropolis light transport. In *SIGGRAPH '97* (1997), pp. 65–76.

[WRC88] WARD G. J., RUBINSTEIN F. M., CLEAR R. D.: A ray tracing solution for diffuse interreflection. In *SIGGRAPH '88* (1988), pp. 85–92.

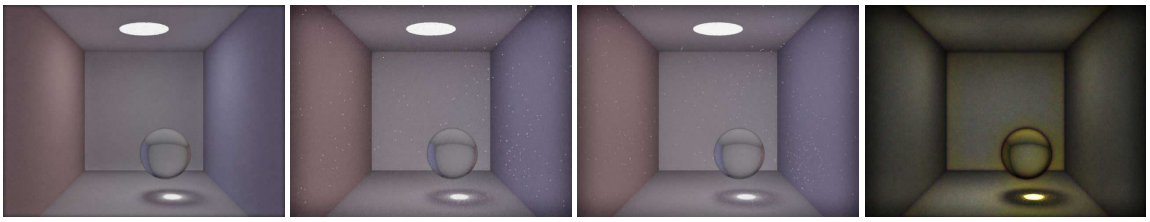

Figure 6: *The first image on the left is a Cornell Box image computed using PMC-ER equal deposition algorithm; the second image is computed using ERPT with 9 spps; the third image is computed using ERPT with 16 spps; and the fourth image is the mutation strategy used during the process. The strategy image shows that the mutation near the physical border and lighting border will automatically adjust to increase the success rate of transfering image.*

Figure 7: *A room scene computed using our PMC-ER equal deposition at the left and ERPT at the right. PMC-ER has fewer artifacts overall. By sharing more information among paths and by better reusing the high contribution paths, PMC-ER is an improvement over ERPT.*