Multiple Photon Sampling Technique Based on Stochastic Progressive Photon Mapping

Liyan Yang¹,² and Chunmeng Kang¹,²,*

¹School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China
²Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan 250014, China

*Corresponding author’s e-mail: kcm89kimi@163.com

Abstract. This paper introduces a multiple photon sampling technique based on stochastic progressive photon mapping. We use the image space concept to divide the scene into continuous sub-blocks and then we calculate our proposed distance function and photon number function in each of the sub-blocks. The distance function is used to calculate the distance error of the hit point and to determine whether each sub-block is located at a boundary between different objects. The photon number function is used to calculate the photon number error and to determine whether the photon distribution in each sub-block is uniform. Based on the values of the distance error and the photon number error, the multiple photon sampling technique is used to acquire multiple samples of the hit point in each sub-block. Instead of using a single radius for the radiance estimate, we use three different radii and compute the final radiance estimate as a weighted average of the three values. When compared with the existing stochastic progressive photon mapping method, our method provides a better solution to the photon distribution problem and can also reduce bias and noise, especially in the scene with drastic changes in light and dark.

1. Introduction

The purpose of graphical reality in computer graphics is to generate a single or continuous frame of images with the aim of representing virtual scenes while making them look as similar as possible to the real scenes. In the ray tracing approach, light is mainly emitted from the viewpoint towards the scene and the hit point on the object is recorded. However, ray tracing involves large numbers of calculations and high memory requirements. In 1996, Jensen introduced photon mapping, which is an extension of the ray tracing method. Photon mapping (PM) can simulate many of the phenomena in global illumination, including caustics and glow [1]. However, because of the randomness of the photon emission function, PM leads to uneven distributions of the photons and the low-frequency noise that need to be resolved by combining the final gather algorithms. Fortunately, Hachisuka et al. proposed the progressive photon mapping method [2]. Progressive photon mapping (PPM) reverses the order of the ray tracing and photon tracing processes, stores the hit point detected during the ray tracing phase, and then emits the photons in one round in a process that could "send light" to the hit point. A new radiance calculation method was proposed for PPM that can converge to the correct radiance value if sufficient photons are provided. Subsequently, Hachisuka improved the PPM algorithm slightly and proposed the stochastic progressive photon mapping (SPPM) algorithm [3]. The difference between
SPPM and PPM is that SPPM uses a distributed ray tracing algorithm [4] to generate hit points in the region randomly after each photon emission, which means that the photon data stored in the k-dimensional tree (k-D-tree) is different each time. SPPM is used to calculate the average radiance value of a region and it is more efficient for complex rendering of scenes in terms of aspects such as depth of field and motion blur.

In this paper, we have combined our previous works to propose the multiple photon sampling (MPS) technique. MPS can improve photon coverage, make the photon distribution more uniform, and reduce the noise. Especially in the scene with drastic changes in light and shade, where the illumination is low, there are fewer photons and more errors. In general, our contributions in this work are summarised as follows:

(i) The concept of image space is used to divide images into uniform sub-blocks and extract the image features sequentially such that they show consistency and continuity.
(ii) The distance function is proposed and is then used to determine the rendering points at different objects and find the boundaries between the different objects.
(iii) The photon number function is proposed and is used to analyse the photon distribution in the sub-block.
(iv) The multiple photon sampling technique, which is based on the value of the distance function and the photon number function, is used to change the sampling radius of the hit point in each sub-block several times to reduce the elimination noise.

2. related work

When drawing complex scenes, use of global illumination can greatly improve the authenticity of the rendering results [5]. Ray tracing simulates the propagation of light in the air and the collisions of the light with objects in space to render different colours and the effects of light and darkness. PM simulates the flux in a scene to produce realistic images. Kang et al. introduced the research status and the limitations of PM algorithm in detail [6]. To make the expected values of the results consistent with the real solution to the rendering equation, Qin et al. proposed a novel photon gathering method [7]. Jönsson and Ynnerman presented a method for interactive global illumination that further improved the performance of the PM algorithm [8].

However, the memory sizes of the computers used to perform the required processing are limited, and the rendering results of some algorithms based on the PM method cannot be arbitrarily accurate. The PPM provides a good solution to this problem. Similar to the original PPM algorithm, APPM technology establishes the connection between PPM and the field of recursive estimation in statistics, further derives a new adaptive bandwidth selection method [9]. Liu and Zhen proposed an adaptive method to redirect the photon reflection direction via adaptive photon shooting [10]. At present, many researchers use hardware to improve the efficiency of the PPM algorithm. Chiu et al. used a progressive radiance estimation engine to increase the processing speed and the image quality of PPM [11]. SPPM is slightly less efficient than PPM but is generally more robust and flexible. SPPM is more robust than PPM over a wide range of scenes. Weiss and Grosch extended the SPPM algorithm to dynamic scenes to simulate animated objects and materials effectively [12]. In 2015, Gunther and Grosch proposed the first consistent out-of-core SPPM algorithm for complex scenarios [13]. Generally, SPPM becomes inefficient when the photon distribution is uneven. Chen et al. combined the SPPM algorithm with Metropolis sampling and proposed a new scalar contribution function [14]. By Metropolis sampling, the visible light paths that affects the results can be locally explored, and more photons are tracked in low-density areas. According to their experimental results, the percentage of visible light paths is significantly increased, thereby achieving the purpose of optimizing the photon distribution. Based on the same starting point, we use the photon resampling method to optimize the photon distribution and further reduce image noise. In addition, we propose the error function to improve the area with more noise and obtain more obvious results. MPS technology is designed to solve some particularly complex scenes, optimize some small details, and further improve the quality of the picture. Experimental results
show that the proposed method provides a better solution to the photon distribution problem and reduces noise.

3. Overview
Most of rendering algorithms are used to solve rendering equations. SPPM uses shared statistics to calculate the correct average radiance value over a specified region. In the ray tracing stage, light is emitted from the required viewpoint. The light intersects with the objects in the scene and the non-mirror-type hit points of the light and the scene are stored. Photons are then emitted from the light source several times to update the hit point statistics. The radiance is calculated based on the density estimate for each hit point using the updated data, and the irradiance received by each pixel is calculated based on the relationship between the intersections and the corresponding pixels.

$$L_O(p, \omega_o) \approx \frac{1}{N_p \pi r^2} \sum_j N_p \beta_j f(p, \omega_o, \omega_j)$$

(1)

Here, $L_O(p, \omega_o)$ is the outgoing radiance at position $p$, $N_p$ denotes the total number of photons emitted by the light source in all iterations, and $\pi r^2$ is the surface area of the shared search radius. $\sum_j N_p \beta_j f(p, \omega_o, \omega_j)$ is the total light flux stored at a specific hit point and is equivalent to the $\tau_i$ parameter proposed by Hachisuka and Jensen in their SPPM paper.

When $N_p \rightarrow \infty$, $r \rightarrow 0$, and the progressive radiance estimate converges towards the correct radiance value.

In the multiple sampling processes, each pixel in the sub-block is projected onto the surface of the scene to form a hit point and each sub-block can then be regarded as a collection of hit points. The value of the sampling radius is determined by analysis of the error values for the distance and the number of photons. We will introduce the distance and photon number functions specifically in Section 4 and the MPS technique in Section 5. Figure 1 shows the algorithm flow for MPS.

4. Distance function and Photon number function
At the ray tracing stage, light is emitted from the viewpoint and the information about the hit points of the light and the object is then stored in a GatherPoint structure. Simultaneously, the distance function and the photon number function are stored in the GatherPoint sub-blocks. We save the resulting photon information in the GatherPoint to calculate the values of the distance error and the photon number error for each of the hit points in the sub-block. During the shading stage, the hit points are traversed, and we acquire multiple samples for each hit point based on the values of the distance error and the photon number error. In the following sections, we will describe the formulas for these functions and the judgment conditions in detail.

4.1. Distance function
First, we save the distance function to the GatherPoint sub-block and then calculate the distance error value for the hit point within the sub-block. The distance error checks if different geometry objects are
in the block which then indicates that a small radius should be used to avoid that the search radius is overlapping with geometric edges. If the sub-block is located at a different object, this suggests that a boundary exists within the sub-block. Because the boundary is a detailed problem in the image rendering process, multiple sampling of the hit point is performed using a small radius. The distance function consists of the average values and the maximum values of the distances to the hit points in the sub-block.

\[ d_{\text{error}} = \frac{d_{\text{ave}}}{d_{\text{max}}} \]  

(2)

In Equation (2), \( d_{\text{error}} \) is the distance error value within the sub-block. \( d_{\text{ave}} \) is the average value of the distances to all the hit points in the sub-block. \( d_{\text{max}} \) is the maximum value of the distances to all the hit points in the sub-block. The distance error value indicates whether the sub-block is located at a boundary between different objects. When the sub-blocks are located on different objects, \( d_{\text{error}} \) is larger; when \( d_{\text{error}} > d_{\text{ave}} \), the sampling radius for the photons near the hit point becomes narrower.

4.2. Photon number function

When photons are emitted, the photon distribution is uneven because of the randomness of the photon emission function, making the change between light and dark more obvious. We propose the photon number function, which is saved to the GatherPoint sub-block and then used to calculate the photon number error. The photon error is used to calculate photon density near hit points. When the photon density is low, we select a larger radius for photon sampling. Photon number function is composed of the average and maximum values of the number of photons in the sub-block.

\[ m_{\text{error}} = \frac{m_{\text{ave}}}{m_{\text{max}}} \]  

(3)

In Equation (3), \( m_{\text{error}} \) is the error value for the number of photons in the sub-block, \( m_{\text{ave}} \) represents the average number of photons in the sub-block, as shown in Equation (8), and \( m_{\text{max}} \) is the maximum number of photons in the sub-block.

\[ m_{\text{ave}} = \frac{\sum \text{\#} \text{of} \text{ photons} \text{ i } \text{th} \text{ point}}{n} \]  

(4)

In Equation (4), \( N_i \) is the number of photons within the search radius used for the \( i \)th hit point in the sub-block, while \( n \) is the number of hit points.

\[ m_{\text{ave}}_{\text{ave}} = \frac{\sum \text{\#} \text{of} \text{ photons} \text{ j } \text{th} \text{ sub-block}}{\text{\#} \text{of} \text{ sub-blocks}} \]  

(5)

In Equation (5), \( m_{\text{ave}}_{\text{ave}} \) is the average value of the average values for all the sub-blocks, \( m_{\text{ave}} \) is the average number of photons in the \( j \)th sub-block, and \( \text{block} \) represents the number of sub-blocks.

At the beginning of the second iteration, several photons are randomly emitted from the light source, and these photons collide with the scene through reflection, refraction, and diffuse reflection processes. Using the distance function and the photon number function, we acquire multiple samples of the photons near the hit points in the sub-blocks based on Equations (6), which are proposed below. Figure 2 depicts the conditions used to determine the MPS.

**Figure 2.** Conditions for assessment of MPS using the Sponza scene as an example. (a) Light and dark block; (b) uniform illumination block; (c) dark block; and (d) different object boundaries block.
The value of $m_{error}$ increases when the sub-block is located at the junction between shadow and a bright position. As shown in Figure 2(a), multiple sampling is performed with a small radius when $m_{error} > m_{ave}$. When the sub-blocks are in an area in which the illumination is consistent, if the scene is bright, as shown in Fig.2(b), the average number of photons in the sub-block increases. In contrast, when $m_{ave} > m_{ave ave}$, the multiple sampling operations are performed with a small radius. As the scene becomes darker, as shown in Fig.2(c), then the values of $d_{error}$ and $m_{error}$ will be smaller. Furthermore, when $m_{ave} < m_{ave ave}$, multiple sampling is performed using a relatively large radius; when the sub-blocks in a scene are located at different objects, as shown in Fig.2(d), $d_{error}$ increases. In addition, when $d_{error} > d_{ave}$, a small radius is required to enable multiple sampling.

5. Photon multiple sampling

In Section 4 above, we introduced the conditions for MPS. When the conditions above are satisfied, we first determine the sub-blocks corresponding to the hit points in the shading stage and then acquire multiple samples at the hit points in the sub-block using the MPS technique. When choosing the proper radius, we used the following equation:

$$r_1 = a_1 r \quad r_2 = a_2 r \quad r_3 = a_3 r \quad (6)$$

In Equation (6), $r_1$, $r_2$ and $r_3$ are the sampling radii of the hit point used in MPS, and $a_1$, $a_2$ and $a_3$ are the radius variation coefficients. We change the sampling radius by changing the values of $a_1$, $a_2$ and $a_3$.

$$a_1 = a'_1 2 \times m_{error} \times d_{error} \quad a_2 = a'_2 2 \times m_{error} \times d_{error} \quad a_3 = a'_3 2 \times m_{error} \times d_{error} \quad (7)$$

In Equation (7), $a_1$, $a_2$ and $a_3$ are the power functions, where the power is given by $2 \times m_{error} \times d_{error}$. $a'_1$, $a'_2$ and $a'_3$ are the initial radius coefficients. Below, we will introduce the derivation process of the formula.

In MPS, a change in the sampling radius leads to a change in the number of photons $M$ to be searched for within the sampling radius and a change in the light flux $\Phi$. When the value of the radius is $r_1$, the surface area of the search radius can be given as follows:

$$S = \pi (a_1 r)^2 = \pi a_1^2 r^2 \quad (8)$$

The surface area is enlarged by $a_1^2$ times in this case. We believe that there is a linear relationship between the light flux and the area. When the radius is $r_1$, the light flux can be described as:

$$\Phi = \Phi_1 a_1^2 \quad (9)$$

After the photons are sampled many times, the light flux is accumulated. When the radii $r_1$, $r_2$ and $r_3$ are used for sampling, the $\Phi$ is given by:

$$\Phi' = \Phi_1 x a_1^2 + \Phi_2 x a_2^2 + \Phi_3 x a_3^2 \quad (10)$$

When we only change the sampling radius, to remain in line with the original Equation (1), the $\Phi$ is calculated as follows after several sampling cycles:

$$\Phi = \frac{\Phi'}{\beta} \quad (11)$$

Here, $\Phi'$ is the cumulative flux with different radii, $\beta$ is:

$$\beta = a_1^2 + a_2^2 + a_3^2 \quad (12)$$

In the ideal case, for calculation of $M$, the photon is within a small range and the illumination intensity is the same, so the photon is a relatively uniform sample, and we thus believe that the number of photons that is collected within the search radius is proportional to the surface area. To maintain consistency with the original Equation (1) through the analysis of the flux, the calculation of $M$ is performed as follows by sampling several times:

$$M = \frac{M'}{\beta} \quad (13)$$

In Equation (13), $M'$ is the number of photons in the search area with different radii. The values of $\Phi$ and $M$ are modified to maintain consistency with Equation (1) via the use of Equations (11) – (13).
The experimental results presented in Section 5 show that our method increases the rendering efficiency, reduces the image noise while improving the algorithm performance. Figure 3 shows the example diagram of the MPS procedure. As shown in Figure 3(a), we set the initial sampling radii: \( r_1 = b_1 r \), \( r_2 = b_2 r \) and \( r_3 = b_3 r \). As shown in Figure 3(b), when the conditions for a small radius illustrated in Figure 2(a), Figure 2(b) or Figure 2(d) are satisfied, \( r_1 = c_1 r \), \( r_2 = c_2 r \) and \( r_3 = c_3 r \) (where \( c_1 < b_1 \), \( c_2 < b_2 \) and \( c_3 < b_3 \) ). As shown in Figure 3(c), similarly, when the conditions for a large radius illustrated in Figure 2(c) are satisfied, we use Equations (7) to modify the radius values. Simultaneously, the hit points in the eligible sub-blocks are sampled in the second iteration through the MPS process.

![Figure 3. Pictures illustrating an example diagram for MPS.](image)

### 6. Results and discussion

In this section, we introduce the results from the MPS process. Both our proposed algorithm and the previous methods were implemented on the Mitsuba renderer. All rendering results were obtained from processing performed on a computer with 16 GB of RAM and a 3.1 GHz Intel® Core (TM) i7-6700 CPU. 100,000 photons per pass were emitted in the Cbox scene, while the other scenes emitted 250,000 photons per pass, with \( \alpha = 0.7 \) and the resolution ratio of each sub-block was 8×8. In the multiple sampling process, we set \( r_1 = 0.8 r \), \( r_2 = 1.5 r \), and \( r_3 = 2 r \) as the initial values of the sampling radii. And then we take \( r_1 = 0.8 r \), \( r_2 = r \), and \( r_3 = 1.2 r \) as the small radius in the sampling process. Table 1 summarizes the statistics of the different scenes in the experiment as follows. We used the iterative 2000 SPPM images as the ground truth (GT). Our method often requires fewer passes than SPPM because of the adoption of the different radii. However, our method is better than the SPPM method in terms of similarity with the GT image.

| Scene           | Rendering time[min] | SPPM passes | Our method passes | SPPM RMSE  | Our method RMSE |
|-----------------|---------------------|-------------|-------------------|------------|-----------------|
| Cbox (Figure 4) | 10                  | 405         | 318               | 7.13E-3    | 5.94E-3         |
| Sponza (Figure 5)| 50                  | 767         | 661               | 6.78E-3    | 5.87E-3         |
| Torus (Figure 6)| 120                 | 973         | 815               | 5.12E-3    | 4.62E-3         |
| Classroom (Figure 8) | 150              | 794         | 743               | 3.27E-2    | 2.97E-2         |
Figure 4. Comparison diagram of the Cbox scene.

Figure 5. Comparison diagram of the Sponza scene.

Figure 6. Comparison diagram of the Torus scene.
Figure 7. Comparison diagrams of the RMSE of the Cbox scene shown in Figure 7(a) and the Torus scene shown in Figure 7(b).

We performed several comparison experiments using the SPPM method and our proposed method at 5 min, 10 min and 15 min in the Cbox scene, as illustrated in Figure 4. At 5 min, the numbers of iterations for our method and for the SPPM method are 170 passes and 200 passes, respectively. At 10 min, the numbers of iterations for our method and for the SPPM method are 318 passes and 405 passes, respectively. At 15 min, the numbers of iterations for our method and for the SPPM method are 520 passes and 654 passes, respectively. The resolution of the image is 512×512. The figures show that our method has lower noise than the SPPM method and that the convergence speed is faster over the same rendering time. We also performed comparison experiments using the SPPM method and our method at 15 min, 30 min and 50 min in the Sponza scene, as illustrated in Figure 5. The image resolution for all pictures in Figure 5 is 768×576. The figures show that our method has lower noise and that the speed of the convergence is faster than that of the SPPM method over the same rendering time. We performed another comparison experiment between the SPPM method and our method for the Torus scene at 25 min, 50 min and 120 min, as illustrated in Figure 6. The resolution of the image was 1024×768. The figures above show that our method can reduce the noise within a short time. In Figure 7, we plot the computed image root mean square error (RMSE) results for our method and the SPPM method when compared with the GT image. For the Cbox scene, we analyzed the image RMSE of SPPM and our method at 5 min, 10 min, 15 min, and 20 min. The GT uses the iterative 2000 SPPM images. In the figure above, the RMSE of our method is obviously smaller than that of the SPPM over the same rendering time. For the Torus scene, we again analyzed the image RMSE of the SPPM and that of our method at 25 min, 50 min, 75 min and 100 min. The GT uses an image from the SPPM method with 2000 iterations. The figure shows that the RMSE for our method is lower than the SPPM method error over the same rendering time. Furthermore, we compared the partial magnification images of the Sponza scene, the Torus scene, and the Classroom scene in Figure 8. We selected two areas to compare from each image: the first is a junction between different objects in the image; the other is a location that is lit consistently. Our method produces less noise and provides faster convergence than the SPPM method over the same rendering time.
Our method

SPPM 25min RMSE 1.10E-2
SPPM 30min RMSE 9.47E-3
SPPM 40min RMSE 5.07E-2

Figure 8. Partial magnification images of the Sponza scene, the Torus scene, and the Classroom scene.

7. Conclusions and future work

In our paper, we have presented a multiple photon sampling (MPS) technique that divides a scene into successive sub-blocks. Simultaneously, we store the distance function and the photon number function in the sub-block. The distance function is used to calculate the distance error value for the hit point in the sub-block, which is used to judge whether or not the sub-block is located at a junction between different objects. The photon number function is used to compute the photon number error value in the sub-block, which is then used to determine whether the photon distribution in the sub-block is uniform. The multiple photon sampling technique is then applied to the hit points in the sub-block based on the values of the distance error and the photon number error. The sampling radius is varied continuously and the photons near the hit point are sampled multiple times. The experimental results demonstrate that our technique reduces both bias and noise effectively. It should be noted here that our method can also be applied to the PPM algorithm.

The multiple photon sampling technique (MPS) enables optimization of most rendering situations. Some sub-blocks with complex conditions may not work well because of the complexity of their lighting conditions. Our implementation may also require more calculations to be performed in the program. However, at the present stage, the acceleration of the effective optimization has been greater than the
calculation time required. In next stage, our program will further optimize the code to achieve more rapid iterations.

Acknowledgements
This research was supported by the National Natural Science Foundation of China (61702311).

References
[1] Jensen H W 2001 Realistic Image Synthesis Using Photon Mapping A K Peters Ltd
[2] Hachisuka T, Ogaki S, Jensen H W 2008 Progressive photon mapping Acm Transactions on Graphics vol.27 no.5 pp 1-8
[3] Hachisuka T and Jensen H W 2009 Stochastic progressive photon mapping Acm Transactions on Graphics vol.28 no.5 pp 1-8
[4] Cook R L, Porter T, Carpenter L 1984 Distributed ray tracing ACM SIGGRAPH computer graphics vol.18 no.3 pp 137-145
[5] Kajiya J T 1986 The rendering equation ACM SIGGRAPH computer graphics vol.20 no.4 pp 143-150
[6] Kang C M, Wang L, Xu Y N, Meng X X 2016 A survey of photon mapping state-of-the-art research and future challenges Frontiers of Information Technology & Electronic Engineering, vol.17 no.3 pp 185-199
[7] Qin H, Sun X, Hou Q, Guo B, Zhou K 2015 Unbiased photon gathering for light transport simulation ACM Trans-actions on Graphics vol.34 no.6 pp 208
[8] Jönsson D and Ynnerman A 2016 Correlated photon mapping for interactive global illumination of time-varying volumetric data IEEE transactions on visualization and computer graphics vol.23 no.1 pp 901-910
[9] Kaplanyan A S and Dachsbacher C 2013 Adaptive progressive photon mapping ACM Transactions on Graphics vol.32 no.2 pp 1-13
[10] Liu X D and Zheng C W 2014 Adaptive importance photon shooting technique Computers & Graphics vol.38 pp 158-166
[11] Chiu C C, Van L D, Lin Y S 2018 Efficient Progressive Radiance Estimation Engine Architecture and Implementation for Progressive Photon Mapping IEEE Transactions on Circuits and Systems I: Regular Papers vol.65 no.8 pp 2491-2502
[12] Weiss M and Grosch T 2012 Stochastic Progressive Photon Mapping for Dynamic Scenes vol.31 pp.719-726 John Wiley and Sons Ltd
[13] Günther T and Grosch T 2015 Distributed out-of-core stochastic progressive photon mapping vol.33 no.6 pp 154-166
[14] Chen J, Wang B, Yong J H 2011 Improved stochastic progressive photon mapping with metropolis sampling Computer Graphics Forum vol.30 no.4 pp 1205-1213