

Estimation of Optimal Number of Rays in the Bidirectional Photon Mapping Method

Sergey Ershov¹ and Alexey Voloboy¹

¹ Keldysh Institute of Applied Mathematics RAS, Miusskaya sq. 4, Moscow 125047, Russia

Abstract

The classic Monte-Carlo ray tracing is a powerful technique for simulating almost all effects in ray optics, but it may be prohibitively slow for, for example, calculation of images seen by a lens camera. Therefore, in practice there are often used its various modifications, in particular, bi-directional stochastic ray tracing with photon maps. The well-known flaw of all stochastic methods is their noise. The noise level, that is, the root mean square of pixel brightness calculated during given time, depends, besides all, on the number of rays traced from the light source and from the camera. The choice of the optimal parameters must provide the lowest noise level in a fixed time. This article is devoted to the choice of the optimal number of rays that minimizes the noise. It is proved that this minimal noise is in the same time homogeneous over the image. We produce the formulae to calculate the optimal number of rays from several coefficients which can be obtained from a bi-directional ray tracing of several auxiliary variants. It happens that this optimum is rather wide i.e. the noise level changes with the number of rays slowly, which allows to choose it including other factors e.g. limit this number to save memory.

Keywords

realistic rendering, bi-directional Monte-Carlo ray tracing, photon maps, denoising

1. Introduction

Nowadays light simulation is widely used in realistic computer graphics and for design of new materials and optical systems [1]. If wave effects may be neglected in simulation, then stochastic ray tracing methods are preferable. This group of methods includes the simulation of light transport using the Metropolis method [2] and stochastic ray tracing [3]. The classical forward Monte Carlo ray tracing (which starts tracing from the light source) is inefficient for image generation, and therefore it is replaced by bidirectional modifications of it [4–6]. One of the most popular methods among them is the bidirectional stochastic ray tracing with photon maps (BDPM – Bidirectional Photon Mapping) [5, 7]. A well-known flaw of all stochastic methods is that they produce noisy results. Therefore, the noise reduction problem always has primary importance. It is considered in many works, e.g., see [8–10].

The level of noise in BDPM mainly depends on the random scattering of the forward and backward rays, on the choice of the vertex for their merging (or, in other words, on the vertex of the camera ray trajectory at which luminance is estimated from photon maps), and also on the number of forward and backward rays traced in one iteration step. The majority of studies is devoted to the first two issues (e.g., [9–12]), and the number of rays got less attention. However, this is an important factor, and it often happens that the number of forward rays is already redundant and its further increase only increases the computation time but does not reduce the noise. In other cases, it may happen that the number of forward rays is indeed critical, while the number of traced backward rays is redundant. It is usually difficult to predict which fraction of the forward and backward rays is optimal: however, a good choice can speed up the computations by several times.

GraphiCon 2021: 31st International Conference on Computer Graphics and Vision, September 27-30, 2021, Nizhny Novgorod, Russia

EMAIL: ersh@gin.keldysh.ru (S. Ershov); voloboy@gin.keldysh.ru (A. Voloboy)

ORCID: 0000-0002-5493-1076 (S. Ershov); 0000-0003-1252-8294 (A. Voloboy)



© 2021 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

In this paper, we propose a method which allows to estimate the optimal number of rays. A general rule determining noise in the BDPM was derived in [8].

One must realize that the rate of convergence of stochastic methods like BDPM is not actually determined the variance of the image calculated in one iteration (or another number of them), but the variance after the fixed time of calculations. Indeed, if the variance of one iteration is decreases twice while the time to calculate it increases tenfold, the situation obviously worsens.

Therefore, calculation of the optimum i.e. those number of camera and light paths that provide “the best image” (or the fastest convergence) we must account for both the variance from one iteration and *the time spent on one iteration*.

In this paper we produce the simple approximating law for that time and, combining it with the law of dependence of the variance from one iteration on the number of rays, we derive the formulae for the optimal number of rays. It happens that those distribution of camera rays per pixel that makes the noise *homogeneous* is just the one which makes it *minimal*. This is especially convenient because eliminates the problem of which spatial aggregate to treat as the “noise amplitude”.

2. Expression for the noise level

Generally the number of camera rays traced through a pixel may depend on this pixel, i.e. vary across the image. The variance of the contribution of one iteration of BDPM can be calculated individually for each pixel and is [11, 13]

$$\begin{aligned} V(p) &= \frac{C(p)}{N_F N_B(p)} + \frac{1 - N_F^{-1}}{N_B(p)} B(p) + \frac{(1 - N_B^{-1}(p))}{N_F} F(p) C(p) \equiv \langle\langle C^2 \rangle\rangle(p) - \langle\langle C \rangle\rangle^2(p) B(p) \\ &\equiv \langle\langle C \rangle_F^2 \rangle_B(p) - \langle\langle C \rangle\rangle^2(p) F(p) \equiv \langle\langle C \rangle_B^2 \rangle_F(p) - \langle\langle C \rangle\rangle^2(p) \end{aligned}$$

where N_F is the number of light paths per iteration, $N_B(p)$ is the number of camera paths through the pixel p per iteration, $C(X^{(F)}, X^{(B)})$ is the contribution to the pixel’s luminance from merging of the light path $X^{(F)}$ with camera path $X^{(B)}$, its average $\langle\langle C \rangle\rangle$ obviously equals the limiting (exact) pixel’s luminance $L(p)$, and $\langle\cdot\rangle_F$ and $\langle\cdot\rangle_B$ denote the averaging over (the ensemble of) light and camera paths, respectively.

The variance of the image obtained during time T is obviously

$$V_T(p) \equiv V(p) \frac{\tau}{T}$$

where τ is the (average) time spent on one iteration. Therefore the value that determines convergence rate and which, therefore, must be minimized to obtain the best image, is

$$V(p) \equiv V(p) \tau \tag{1}$$

Let’s estimate the average time per iteration. The iteration in BDPM consists of three successive parts:

- BMCRT (from camera);
- FMCRT (from light source);
- “Merging” of the above sub-paths, when *each* light path is (attempted to) join with *each* camera path.

The total iteration time is the sum of these partial timings denoted as $\tau_B, \tau_F, \tau_{merge}$ respectively. The time spent on the first parts is obviously linear in the number of rays. For the third part, it is linear in the number of their *pairs*. The general functional form is then

$$\tau_B = \sum_p \beta(p) N_B(p), \quad \tau_F = \alpha N_F, \quad \tau_{merge} = N_F \sum_p \gamma(p) N_B(p) \tag{2}$$

where $\beta(p)$ has the sense of the *average* time spent on tracing of *one* camera ray through the pixel p , α has the sense of the *average* time spent on tracing of *one* light ray and $\gamma(p)$ is the *average* time spent on “merging” one pair of light ray and one camera ray (through pixel p). This $\gamma(p)$ can depend on the pixel because for different pixels the camera rays go different parts of the scene where the density of light rays (and thus the probability of the merging) is different.

Summing, we have

$$\tau = \tau_B + \tau_F + \tau_{merge} = \alpha N_F + \sum_p \beta(p) N_B(p) + N_F \sum_p \gamma(p) N_B(p) \quad (3)$$

Substituting (3) in (1) we see that the value to be minimized to obtain the best image is

$$V(p) = \left(\frac{C(p)}{N_F N_B(p)} + \frac{1 - N_F^{-1}}{N_B(p)} B(p) + \frac{1 - N_B^{-1}(p)}{N_F} F(p) \right) \left(\alpha N_F + \sum_p \beta(p) N_B(p) + N_F \sum_p \gamma(p) N_B(p) \right)$$

Usually the number of rays is much greater than 1, so we can neglect the terms $N_B^{-1}(p)$ и N_F^{-1} :

$$V(p) = \left(\frac{1}{N_B(p)} \left(\frac{C(p)}{N_F} + B(p) \right) + \frac{F(p)}{N_F} \right) \left(\alpha N_F + \sum_p \beta(p) N_B(p) + N_F \sum_p \gamma(p) N_B(p) \right)$$

3. Minimization of noise

Usually the noise is not homogeneous over the image, i.e. it depends on the pixel. It is not very convenient because it is difficult to decide, which is better to have less noise on the floor yet stronger in the ceiling or vice versa? Therefore usually we use a single measure of noise, e.g. the maximum over the image, which better corresponds to the visual perception of “noisy” or “clear” images.

Therefore, what we need to minimize is $V(p)$.

It happens that (it will be so should $N_B(p)$ be continuous instead of integers) in this case the noise is homogeneous i.e.

$$V(p) = V(p) \equiv \underline{V}$$

Indeed, let us suppose the opposite: that in some pixel p_* the noise is below the maximum i.e. $V(p_*) < V(p)$. Now let us decrease (infinitesimally as now we treated it as a continuous parameter!) $N_B(p_*)$. Obviously, in all the other pixels but p_* the noise will *decrease* after it because of decrease of the time spent on one iteration τ :

$$\delta V(p) = V(p) \frac{\delta \tau}{\tau}$$

while $V(p)$ for these pixels won't change. In the same time in pixel p_* the change of noise level will be

$$\delta V(p_*) = -\frac{1}{N_B^2(p_*)} \left(\frac{C(p_*)}{N_F} + B(p_*) \right) \tau \delta N_B(p_*) + V(p_*) \frac{\delta \tau}{\tau}$$

In view of our supposition $V(p_*) < V(p)$ the maximum is achieved outside p_* where for each pixel $\delta V(p) = V(p) \frac{\delta \tau}{\tau}$. Therefore, $\delta V(p) = V(p)$ and

$$\delta V(p) - \delta V(p_*) = (V(p) - V(p_*)) \frac{\delta \tau}{\tau} + \frac{1}{N_B^2(p_*)} \left(\frac{C(p_*)}{N_F} + B(p_*) \right) \tau \delta N_B(p_*)$$

By supposition, $V(p) - V(p_*) > 0$ and since $\delta N_B(p_*) < 0$, $\delta \tau < 0$, then

$$\delta V(p) - \delta V(p_*) < 0$$

In other words, after our change $V(p) - V(p_*)$ decreases, i.e. the distribution of noise becomes more homogeneous. Simultaneously the maximum of noise decreases i.e. we approach the optimum.

Therefore, if the distribution of noise is *not* homogeneous, then varying the number of camera rays (through those pixels where the noise is below the maximum) we can make it *more optimal* (smaller). In other words, an inhomogeneous distribution *cannot* be the optimal (since it admits improvement). Therefore, ***the optimal noise distribution is homogeneous.***

In fact the limitation that $N_B(p)$ must be integer distort this simple solution, but in case $N_B(p)$ is large enough, the effect must be weak. One may not though hope to reach the homogeneous distribution with small number of camera rays, so it is better to have it large enough so that it be at least 3 in the “worse” pixels (recall that the roundoff error is below half unit!).

So we should estimate the number of camera rays that yields a homogeneous distribution of noise. For the sake of simplicity let us consider the case when $\frac{1}{N_F} F(p)$ is negligible (as compared to the other noise components) which for the case of BDD>0 is if not always then very frequently. Then

$$V(p) = \frac{1}{N_B(p)} \left(\frac{C(p)}{N_F} + B(p) \right) \left(\alpha N_F + \sum_p \beta(p) N_B(p) + N_F \sum_p \gamma(p) N_B(p) \right) \quad (4)$$

and one can see that the homogeneous distribution of noise is achieved when

$$N_B(p) = \text{const} \times \left(\frac{C(p)}{N_F} + B(p) \right) = \left(\frac{C(p)}{N_F} + B(p) \right) \frac{\underline{N}_B}{\underline{B} + \frac{\underline{C}}{N_F}} \quad (5)$$

where n_x, n_y denote the image size in pixels and

$$\underline{B} \equiv \frac{1}{n_x n_y} \sum_p B(p), \quad \underline{C} \equiv \frac{1}{n_x n_y} \sum_p C(p)$$

Now we have just two free parameters: N_F and the scale factor \underline{N}_B for the number of camera rays (notice the actual *average* number of camera rays per pixel will deviate from it because of rounding) which must be varies to minimize the (now homogeneous!) noise.

4. The optimal N_F and \underline{N}_B

Substituting (5) in (4) and writing V instead of $V(p)$ because now the noise is homogeneous, one has

$$V = \alpha \hat{C} \frac{1}{\underline{N_B}} + \alpha \hat{B} \frac{N_F}{\underline{N_B}} + \frac{1}{N_F} \hat{\beta} \hat{C} + N_F \hat{\gamma} \hat{B} + \hat{\beta} \hat{B} + \hat{\gamma} \hat{C}$$

where the overhat denotes the sum:

$$\hat{f} \equiv \sum_p f(p)$$

One easily finds that for the given $\underline{N_B}$, minimum in N_F is achieved for

$$N_F = \sqrt{\frac{\hat{\beta} \hat{C}}{\frac{\alpha \hat{B}}{\underline{N_B}} + \hat{\gamma} \hat{B}}} \quad (6)$$

Denoting

$$\eta \equiv \sqrt{\frac{\alpha \hat{B}}{\underline{N_B}} + \hat{\gamma} \hat{B}}$$

we can write this minimum as

$$V = (\eta^2 - \hat{\gamma} \hat{B}) \left(\frac{\hat{C}}{\underline{B}} + \frac{\sqrt{\hat{\beta} \hat{C}}}{\eta} \right) + \eta \sqrt{\hat{\beta} \hat{C}} + \frac{\hat{\gamma} \hat{B} \sqrt{\hat{\beta} \hat{C}}}{\eta} + \hat{\beta} \hat{B} + \hat{\gamma} \hat{C} = \frac{\hat{C}}{\underline{B}} \eta^2 + 2\eta \sqrt{\hat{\beta} \hat{C}} - \frac{\hat{C} \hat{\gamma} \hat{B}}{\underline{B}} + \hat{\beta} \hat{B} + \hat{\gamma} \hat{C}$$

Obviously, this is a monotone increasing function of η and thus a monotone decreasing function of $\underline{N_B}$. Therefor there is no extremum in $\underline{N_B}$ and formally the greater this value, the better. However too many rays per pixels is bad as concerning memory etc. Therefore it is reasonable to take such $\underline{N_B}$ that the noise is only $(1 + \epsilon)$, where ϵ is a small number, times the limiting noise value for $\underline{N_B} \rightarrow \infty$ i.e.

$$\frac{\hat{C}}{\underline{B}} \eta^2 + 2\eta \sqrt{\hat{\beta} \hat{C}} - \frac{\hat{C}}{\underline{B}} \hat{\gamma} \hat{B} + \hat{\beta} \hat{B} + \hat{\gamma} \hat{C} = (1 + \epsilon) \left(2\sqrt{\hat{\gamma} \hat{B}} \sqrt{\hat{\beta} \hat{C}} + \hat{\beta} \hat{B} + \hat{\gamma} \hat{C} \right)$$

or

$$\frac{\hat{C}}{\underline{B}} \eta^2 + 2\eta \sqrt{\hat{\beta} \hat{C}} - \left(\frac{\hat{C}}{\underline{B}} \hat{\gamma} \hat{B} + \epsilon(\hat{\beta} \hat{B} + \hat{\gamma} \hat{C}) + 2(1 + \epsilon) \sqrt{\hat{\gamma} \hat{B}} \sqrt{\hat{\beta} \hat{C}} \right) = 0$$

which is achieved for

$$\begin{aligned}
\eta &= \frac{-\sqrt{\widehat{\beta C}} + \sqrt{\widehat{\beta C} + \frac{C}{\underline{B}} \left(\frac{C}{\underline{B}} \widehat{\gamma B} + \epsilon(\widehat{\beta B} + \widehat{\gamma C}) + 2(1 + \epsilon)\sqrt{\widehat{\gamma B}}\sqrt{\widehat{\beta C}} \right)}}{\frac{C}{\underline{B}}} \\
&= \frac{-\sqrt{\widehat{\beta C}} + \left(\sqrt{\widehat{\beta C}} + \frac{C}{\underline{B}} \sqrt{\widehat{\gamma B}} \right) \sqrt{1 + \epsilon \frac{C}{\underline{B}} \frac{\widehat{\beta B} + \widehat{\gamma C} + 2\sqrt{\widehat{\gamma B}}\sqrt{\widehat{\beta C}}}{\left(\sqrt{\widehat{\beta C}} + \frac{C}{\underline{B}} \sqrt{\widehat{\gamma B}} \right)^2}}}{\frac{C}{\underline{B}}} \\
&\approx \sqrt{\widehat{\gamma B}} + \frac{\epsilon}{2} \frac{\widehat{\beta B} + \widehat{\gamma C} + 2\sqrt{\widehat{\gamma B}}\sqrt{\widehat{\beta C}}}{\sqrt{\widehat{\beta C}} + \frac{C}{\underline{B}} \sqrt{\widehat{\gamma B}}}
\end{aligned}$$

(We naturally discarded the negative root) or

$$\begin{aligned}
\frac{N_B}{\underline{B}} &= \frac{\alpha \widehat{B}}{\eta^2 - \widehat{\gamma B}} = \frac{\alpha \underline{B}}{\left(\frac{-\sqrt{\widehat{\beta C}} + \sqrt{\widehat{\beta C} + \frac{C}{\underline{B}} \left(\frac{C}{\underline{B}} \widehat{\gamma B} + \epsilon(\widehat{\beta B} + \widehat{\gamma C}) + 2(1 + \epsilon)\sqrt{\widehat{\gamma B}}\sqrt{\widehat{\beta C}} \right)}}{\frac{C}{\underline{B}}} \right)^2 - \widehat{\gamma B}} \\
&\approx \frac{1}{\epsilon} \frac{\alpha \underline{B} \left(\sqrt{\widehat{\beta C}} + \frac{C}{\underline{B}} \sqrt{\widehat{\gamma B}} \right)}{\sqrt{\widehat{\gamma B}} \left(\widehat{\beta B} + \widehat{\gamma C} + 2\sqrt{\widehat{\gamma B}}\sqrt{\widehat{\beta C}} \right)}
\end{aligned} \tag{7}$$

Substituting it in (6) we obtain the optimal number of the light (forward) rays:

$$N_F = \frac{\sqrt{\widehat{\beta C}}}{\eta} \approx \sqrt{\frac{\widehat{\beta C}}{\widehat{\gamma B}}} \tag{8}$$

5. How to estimate the timings

The value of α , i.e. time spent on average on one light (forward) ray is rather easy to measure. We trace some large number of FMCRT rays N_F , measure the time spent and divide it by N_F .

But $\beta(p)$ and $\gamma(p)$ are more difficult to obtain. In principle, we can perform a series of BMCRT simulations for *single* pixel each, tracing a large number of rays through this pixel and then dividing the time spent by their number. This is expensive, though, because for an accurate measurement of time requires that it be at least milliseconds, while there usually is millions of pixels.

The value of $\gamma(p)$ also can be measured straightforwardly, but this is even more difficult.

Happily, we do not need all this because the expressions (7) and (8) include $\beta(p)$ and $\gamma(p)$ only in the following combinations

$$\sum_p \beta(p)B(p), \quad \sum_p \gamma(p)B(p), \quad \sum_p \beta(p)C(p), \quad \sum_p \gamma(p)C(p)$$

By definition, the two former are BMCRT time and merging time when there are $B(p)$ camera rays per pixel and one FMCRT ray. The two latter are for $C(p)$ camera rays per pixel.

These values can be measured easily just by doing a few iterations of bi-directional ray tracing for some specially chosen $N_B(p)$ and some (more or less arbitrary) medium N_F about, say, 10000.

Namely, let us take $N_B(p) = n_B \times B(p)$ where the constant n_B is chosen so that the average number of rays per pixels be about, say, 10. Then we measure the time spent on the backward ray tracing τ_B , on the forward ray tracing τ_F and time spent on merging the paths τ_C . In view of (2),

$$\alpha = \frac{\tau_F}{N_F}, \quad \sum_p \beta(p)B(p) = \frac{\tau_B}{n_B}, \quad \sum_p \gamma(p)B(p) = \frac{\tau_{merge}}{n_B N_F}$$

Similarly, choosing $N_B(p) = n_C \times C(p)$ we can measure the two remaining sums.

6. Results

We use the famous Cornell Box with an isotropic point light source slightly below the center of the ceiling. All scene surfaces are gray Lambert with albedo $k_d=50\%$. Both direct and indirect illumination was taken from photon maps. “Diffuse depth” i.e. the maximal allowed number of diffuse events for camera ray [11] was BDD=1. Scene image is shown in Figure 1.

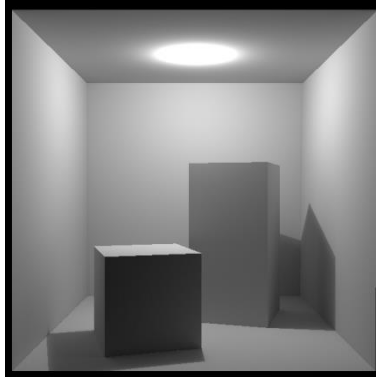


Figure 1: Camera image of the test scene.

There are two variants which differ in the radius of integration sphere R : “large” one equals 0.0083 of the scene size, and the “small” is 0.0015 of the scene size. Naturally, one can expect that smaller radius requires more FMCRT rays.

Calculation of \hat{B} and \hat{C} was done by the usual bi-MCRT, using radius 0.0083 of the scene size, 10000 iteration had been done with $N_F = 10000$ light paths and $N_B(p) = 25$ camera rays per pixel. The matrices $B(p)$ and $C(p)$ were calculated as described in [13]. The sums are presented in Table 1.

Table 1

Averages of the BMCRT term and cross-term for the two test variants

	$R = 0.0083$ of scene size	$R = 0.0015$ of scene size
\underline{B}	16.47	16.47
\underline{C}	2599	80556

Calculation of time-related terms was done like described in Section 5 for 10000 light paths and average of 50 camera paths per pixel. Obviously the pure BMCRT and FMCRT terms are independent of radius. Since measurement of time is not very accurate (because of loading by background processes, adaptive changing of CPU clock etc), the figures are rounded (Table 2).

Table 2

Time-related sums for the two test variants

	$R = 0.0083$ of scene size	$R = 0.0015$ of scene size
α	$6.35 \cdot 10^{-7}$	$6.35 \cdot 10^{-7}$
$\widehat{\beta B}$	1.75	1.75
$\widehat{\beta C}$	247	9000
$\widehat{\gamma B}$	$3.3 \cdot 10^{-5}$	$1.3 \cdot 10^{-6}$
$\widehat{\gamma C}$	$5.7 \cdot 10^{-3}$	$6.5 \cdot 10^{-3}$

Approximation of the iteration time (3) for the number of camera rays given by (5) becomes

$$\tau = \alpha N_F + \frac{N_B}{C + N_F B} (\widehat{\beta C} + N_F (\widehat{\beta B} + \widehat{\gamma C}) + N_F^2 \widehat{\gamma B}) \quad (9)$$

Tables 3 and 4 presents comparison of this approximation of the real iteration time with the coefficients taken from Table 2.

Table 3

Iteration time for radius = 0.0083 of scene size. The first number is real value, the second one is calculated according to (9).

	$N_B = 5$	$N_B = 50$
$N_F = 1000$	0.71 vs 0.53	5.3 vs 5.3
$N_F = 3000$	0.79 vs 0.56	5.5 vs 5.6
$N_F = 10000$	0.86 vs 0.63	6.3 vs 6.3

Table 4

Iteration time for radius – 0.0015 of scene size. The first number is real value, the second is from (9).

	$N_B = 5$	$N_B = 50$
$N_F = 10000$	0.78 vs 0.54	5.3 vs 5.4
$N_F = 60000$	0.82 vs 0.56	5.85 vs 5.6
$N_F = 200000$	0.98 vs 0.61	6.3 vs 6.1

One can see that approximation is quite good for large N_B but not for small N_B . Mainly this is because (9) did not include rounding of the number of camera rays which increase $N_B(p)$ and thus the real run time is larger. Finally the optimal numbers of rays are in Table 5.

REMARK. Rather large times are because we did not optimized the calculations and did not use SIMD and even multithreading.

Table 5

The number of rays for the two test variants

	$R = 0.0083$ of scene size	$R = 0.0015$ of scene size
N_B for $\epsilon = 0.1$	0.115	3.26
N_B for $\epsilon = 0.01$	1.6	43.7
N_F for $N_B = 5$	2653	51153
N_F for $N_B = 50$	2727	77224

Obviously, the small estimates for \underline{N}_B is rather absurd since it yields less than one ray per pixel everywhere. Small $N_B(p)$ results in strong rounding and the error of rounding which can reach ± 0.5 is a serious distortion when $N_B(p)$ is about 3. So we used $\underline{N}_B = 5$ as the low bound. As explained above, there is no formal optimum i.e. the greater the better, but too large values result in unjustified memory consumption. So for we adopted $\underline{N}_B = 50$ for the high bound. The distribution of $N_B(p)$ for low and high bounds is presented in Figure 2. The effect of this distortion is maximal where $N_B(p)$ is small, i.e. in the blue image areas, mainly the ceiling and the small box. Later we shall see that it is these areas where the noise deviates from the mean value.

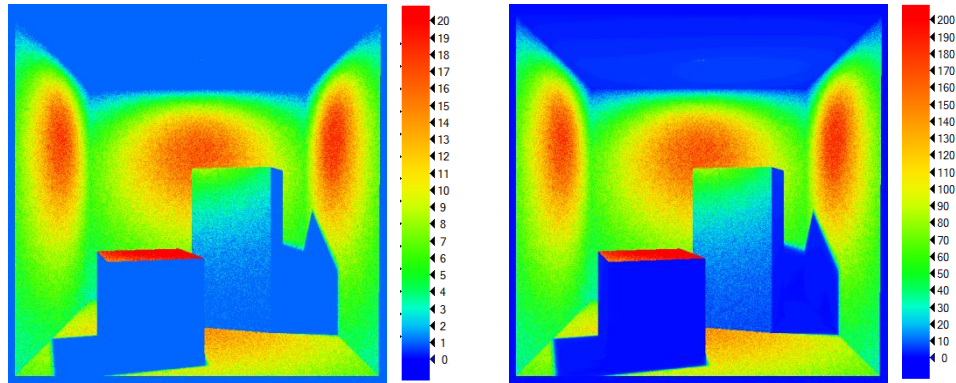


Figure 2: The distribution of $N_B(p)$ for larger radius, $N_F = 3000$ and $\underline{N}_B = 5$ (left) and for $\underline{N}_B = 50$ (right). The actual average is 5.6 and 54, respectively, because rounding to integers increases the values as compared to (5).

Table 6

Noise invariant \sqrt{V} for the test variant with larger radius of integration sphere = 0.0083 of scene size

	$\underline{N}_B = 5$	$\underline{N}_B = 50$
$N_F = 1000$		
$N_F = 3000$		
$N_F = 10000$		

As explained in Section 2, what determines the speed of convergence is the product (1) of the variance *in one iteration* and τ is time per iteration. We produce its root \sqrt{V} (notice it relates to the *absolute* (not relative!) stochastic error) and visualize it in Table 6 (for larger radius of integration sphere) and Table 7 (for smaller radius). According to toolbar the red color corresponds to value 2, yellow color to 1 and blue color to 0. The middle row is for N_F close to optimal.

One can see that the noise image roughly has three distinct areas: walls of the main box, ceiling and the small box, the noise being nearly homogeneous throughout each area. The corresponding values are printed below each image for quantitative comparison.

The areas where the noise considerably deviates from the mean level coincide with areas where $N_B(p)$ is below 3 or 5, see Figure 2. Meanwhile prediction of noise level (4) itself is quite good as it can be seen from comparison of images in Figure 3, so the inhomogeneity is due to the deviation of the integer $N_B(p)$ from the *continuous* values (5):

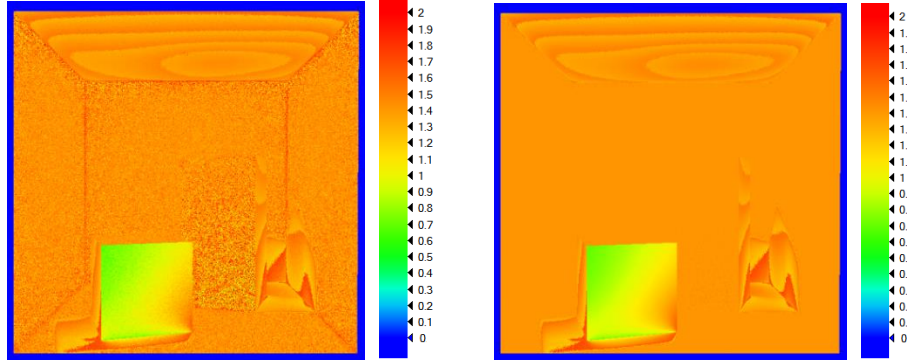
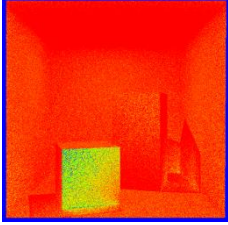
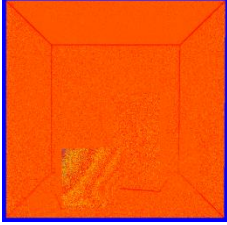
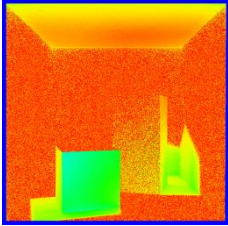
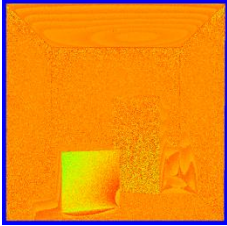
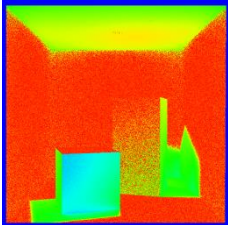
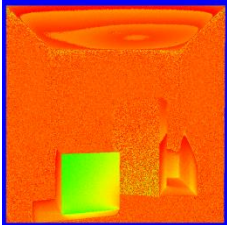


Figure 3: The distribution of $\sqrt{V}(p)$ for $N_B = 50$ and $N_F = 3000$: actual (left) and prediction (4) (right).

Table 7

Noise invariant \sqrt{V} for the test variant with larger radius of integration sphere = 0.0015 of scene size

	$N_B = 5$	$N_B = 50$
$N_F = 10000$	 walls: 1.85; ceiling: 3.0; small box: 1.0	 walls: 1.7; ceiling: 1.7; small box: 1.6
$N_F = 60000$	 walls: 1.7; ceiling: 1.3; small box: 0.45	 walls: 1.44; ceiling: 1.44; small box: 1.2
$N_F = 200000$	 walls: 1.8; ceiling: 0.9; small box: 0.3	 walls: 1.47; ceiling: 1.46; small box: 0.75

The situation is qualitatively similar to the case of larger radius (Table 6). Again one sees the same three distinct areas: walls of the main box, ceiling and the small box, the noise being nearly homogeneous throughout each area. The corresponding values are printed below each image for quantitative comparison. Again the areas where the noise level deviates from mean are those where $N_B(p)$ is below 2 or 3 (cf. Figure 2) and so rounding to integers deviated them from (5).

7. Conclusion

From comparison of the calculated variants one can see that

1. Noise is the more homogeneous the larger the \underline{N}_B (natural: because of the rounding $N_B(p)$ becomes less essential) and the larger the ratio \underline{N}_B/N_F (which looks strange).
2. The noise for the most of the area is not sensitive to the \underline{N}_B and N_F . In other words, the optimum is “very wide” and deviating from it even threefold does not change the mean noise level much. This is good in many aspects because in real simulation of complex scenes it may be difficult to find the exact values of the number of rays (as they use coefficients that must be ray-traced themselves). Also time per iteration is difficult to measure as it is a random value and additionally it may have regular trends because of repeating caching etc.
3. The areas where the noise considerably deviates from the mean level coincide with areas where $N_B(p)$ is below 2 or 3, see Figure 2. Therefore it is reasonable to choose \underline{N}_B (which all the same does not have a precise optimum value) large enough so that $N_B(p)$ exceeds, say, 3 in the most of the image.

This paper operates the *absolute* noise, while sometimes it is better to work with the relative noise (divided by the luminance in this pixel). Our approach can be easily adapted to this case as well.

8. Acknowledgements

We should like to thank Elissey Birukov for his help with computations.

9. References

- [1] D. Zhdanov, V. Galaktionov, A. Voloboy, A. Zhdanov, A. Garbul, I. Potemin, V. Sokolov, Photorealistic rendering of images formed by augmented reality optical systems, *Programming and Computer Software* 44 (2018) 213–224. doi:10.1134/S0361768818040126.
- [2] M. Sik, J. Krivanek, Survey of Markov Chain Monte Carlo methods in light transport simulation, *IEEE Transactions on Visualization and Computer Graphics* 26(4) (2018) 1821–1840.
- [3] M. Pharr, G. Humphreys, *Physically Based Rendering. Second Edition: From Theory To Implementation*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2010.
- [4] N. Dodik, Implementing probabilistic connections for bidirectional path tracing in the Mitsuba Renderer 2017 URL: <https://www.cg.tuwien.ac.at/research/publications/2017/dodik-2017-pcbpt>
- [5] H.W. Jensen, P. Christensen, High quality rendering using ray tracing and photon mapping, in: *ACM SIGGRAPH 2007 Courses*, ser. SIGGRAPH '07. ACM, New York, NY, USA, 2007. doi:10.1145/1281500.1281593
- [6] E. Veach, A dissertation: Robust Monte-Carlo methods for light transport simulation, 1997. URL: http://graphics.stanford.edu/papers/veach_thesis/thesis.pdf
- [7] J. Vorba, Bidirectional photon mapping, in: *Proceedings of CESC 2011: The 15th Central European Seminar on Computer Graphics*. Prague: Charles University, 2011, pp. 25–32. URL: <https://cgg.mff.cuni.cz/~jaroslav/papers/2011-bdpm/vorba2011-bdpm.pdf>
- [8] S.V. Ershov, D.D. Zhdanov, A.G. Voloboy, Estimation of noise in calculation of scattering medium luminance by MCRT, *Mathematica Montisnigri* 45 (2019) 60–73. doi:10.20948/mathmontis-2019-45-5

- [9] I. Georgiev, J. Krivánek, T. Davidovic, P. Slusallek, Light transport simulation with vertex connection and merging, *ACM Trans. Graph.*, 31(6) (2012) 192:1–192:10. doi:10.1145/2366145.2366211
- [10] T. Hachisuka, J. Pantaleoni, H.W. Jensen, A path space extension for robust light transport simulation, *ACM Trans. Graph.*, 31(6) (2012) 191:1–191:10.
- [11] S.V. Ershov, A.G. Voloboy, Calculation of MIS weights for bidirectional path tracing with photon maps in presence of direct illumination, *Mathematica Montisnigri* 48 (2020) 86–102. doi:10.20948/mathmontis-2020-48-8
- [12] M. Sbert, V. Havran, and L. Szirmay-Kalos, Multiple importance sampling revisited: breaking the bounds, *EURASIP Journal on Advances in Signal Processing*, 15 (2018) 1–15.
- [13] S.V. Ershov, E.D. Birukov, A.G. Voloboy, V.A. Galaktionov, Noise Dependence on the Number of Rays in Bidirectional Stochastic Ray Tracing with Photon Maps, *Programming and Computer Software* 47(3) (2021) 194-200. doi:10.1134/S036176882103004X