Calculation of Noise Components for Bidirectional Path Tracing with Photon Maps*

Sergey Ershov[0000-0002-5493-1076], Elisey Birukov[0000-0003-4297-6813], Alexey Voloboy[0000-0003-1252-8294] and Vladimir Galaktionov[0000-0001-6460-7539]

Keldysh Institute of Applied Mathematics RAS, Miusskaya sq. 4, 125047 Moscow, Russia
{ersh,voloboy,birukov,vlgal}@gin.keldysh.ru

Abstract. The classic Monte-Carlo ray tracing is a powerful method which allows to simulate virtually all effects in ray optics, but it may be inadmissibly slow for many cases, such as calculation of images seen by a lens or pin-hole camera. In this cases another stochastic method is more efficient such as the bi-directional ray Monte-Carlo tracing with photon maps (BDPM). The level of noise i.e. the r.m.s. ("root mean square") of pixel luminance calculated in one iteration of the method, depends on various parameters of the method, such as the number of light and camera paths, radius of integration sphere etc. so it is desirable to be able to predict this dependence to choose optimal parameters of the method. It was shown that this r.m.s is a sum of 3 functions scaled by reverse number of camera and light rays. These functions themselves are independent of the number of rays, so knowing them one can predict the noise for any number of rays and thus find the optimal one. These functions are a sort of correlations and their calculation from ray tracing is not a trivial problem. In this paper we describe a practical method of calculation and demonstrate the usage of its results for the choice of ray number.

Keywords: Realistic Rendering, Bi-directional Monte-Carlo Ray Tracing, Photon Maps.

1 Introduction

Currently simulation of light propagation is widely used not only in optical engineering but also in design of new materials. It is intensively applied in architectural, automotive and aircraft design tasks. If wave effects can be neglected, a kind of stochastic ray tracing is a good choice. This realm mainly includes Metropolis Light Transport and Monte-Carlo ray tracing (MCRT) method [1]. When an image is to be calculated the classic forward ray tracing from light source is inefficient and so is replaced with various bi-directional modifications [2], [3], [4]. Of them we consider the so-called bi-directional Monte-Carlo ray tracing with photon maps (BDPM) [3], [5]. The weak side of all stochastic methods is that their results are noisy, and so is this one, thus much work had been done to decrease it [6], [7], [8].

* Publication is supported by RFBR grant 20-01-00547

The level of noise in this method mainly depends on the method of random scattering of the forward and backward rays, on choosing the vertex to merge (or, in other words, in which vertex of the camera path to use photon maps for estimation of the illumination) and, at last, on the number of forward and backward rays traced during one iteration. Most publications are devoted to the first two means, for example, [9], [7], [10] or [8], while less attention had been paid to the number of rays. Meanwhile it is an important factor and frequently it happens that e.g. the number of forward rays is already superfluous, so increasing it further does not decrease noise but only increases the run time. In other cases it happens that the number of forward rays is really critical while we traced excess number of camera rays. It is usually difficult to predict which proportion of them is optimal while its good choice may accelerate calculations several times.

In this paper we address the latter problem. In [6] the general law which determines the noise in BDPM had been derived. It states that the variance of pixel luminance is a sum of three components scaled by inverse number of rays. These 3 components are independent from the number of rays and thus once calculated, at least approximately, they can be later used to predict how the noise level depends on the number of rays traced in one iteration and thus choose the better number of rays. In other words, had we knew these 3 components (which usually depend on pixel) we would be able to predict the noise level for any number of rays.

Although their mathematical definition is trivial, these three values are not that easy to calculate numerically by ray tracing. In this paper we describe a method of their efficient calculation and demonstrate their usage to choose the optimal number of rays.

2 Noise in BDPM

Usually BDPM proceeds iteration by iteration. In each iteration we trace N_F light ("forward") paths and, for each pixel p, $N_B(p)$ camera ("backward") paths. Then each light path is checked against each camera path and if there are vertices they run close, they are "merged" into a join path (which connects camera and light source). We then calculate *contribution* of this join path to the luminance of pixel and increment pixel luminance by this value. The accumulated sum, divided by $N_F N_B(p) N_I$ where N_I is the number of iterations, converges to the expectation. This average is independent from N_F and $N_B(p)$, but the noise (variance) is. Say, if there is already too many light paths while few camera paths, tracing these excess light rays only takes time without gain, so it is advantageous to reduce their number.

The optimal number of rays (total per scene for light paths and individually for each pixel for camera paths) is that which results in the least noise after the fixed calculation time. To solve this minimization problem we need to know how the variance of the value produced in one iteration depends on that number of rays. This dependence is described by the 3-term law [6], but its terms are not that easy to calculate numerically.

Here and below we shall always speak about the values for one given pixel p and so drop this notation, writing N_B instead of $N_B(p)$ etc.

As shown in [6], the variance of pixel luminance calculated *in one iteration* is

$$V = \frac{1}{N_F N_B}\left(\langle\langle C^2\rangle\rangle - \langle\langle C\rangle\rangle^2\right) + \frac{1-N_F^{-1}}{N_B}\left(\langle\langle C\rangle_F^2\rangle_B - \langle\langle C\rangle\rangle^2\right)$$
$$+\frac{1-N_B^{-1}}{N_F}\left(\langle\langle C\rangle_B^2\rangle_F - \langle\langle C\rangle\rangle^2\right) \quad (1)$$

where $C(X^{(F)}, X^{(B)})$ is the contribution to pixel luminance from "merging" the light path $X^{(F)}$ and camera path $X^{(B)}$, its average $\langle\langle C\rangle\rangle$ is obviously the limiting pixel luminance L, and $\langle\cdot\rangle_F$, $\langle\cdot\rangle_B$ denote the averages over the ensemble of light and camera paths, respectively.

Notice this law holds regardless of whether we use Multiple Importance Sampling [9], [11], [4] or not which only affects the values of $\langle\langle C^2\rangle\rangle$, $\langle\langle C\rangle_F^2\rangle_B$ and $\langle\langle C\rangle_B^2\rangle_F$.

Calculation of $\langle\langle C^2\rangle\rangle$ is easy: whenever we add C to the pixel luminance, we also add C^2 to the "error image". Then at each iteration we calculate

$$\mathcal{C} \equiv \frac{1}{N_B N_F}\sum_{j=1}^{N_B}\sum_{i=1}^{N_F} C^2(X_i^{(F)}, X_j^{(B)}) \quad (2)$$

and averaging it *over iterations* $\overline{\mathcal{C}}$ just gives what we need: $\overline{\mathcal{C}} \to \langle\langle C^2\rangle\rangle$. But with $\langle\langle C\rangle_F^2\rangle_B$ and $\langle\langle C\rangle_B^2\rangle_F$ we face difficulty.

For example, let us suppose there is *very many* light paths per iteration. Then for *each* camera ray the sum over them gives a good approximation to $\langle C\rangle_F$. The average over camera paths is bad because normally we have a few (10, max. 100) camera paths through one pixel per iteration. But if we additionally average over iterations, we get a good approximation to $\langle\langle C\rangle_F^2\rangle_B$. This, however, does not work if there are not that many light paths (to be precise, not that many *intersections*). In this case the average over light paths from one iteration deviates seriously from $\langle C\rangle_F$. Meanwhile averaging the squared value over iterations may give averaged square, not square of average because averaging over iterations is $\langle\langle\cdot\rangle\rangle$. What to do then?

Solution is very simple. Let the number of light and camera paths traced in one iteration be the same (for all iterations). To calculate $\langle\langle C\rangle_F^2\rangle_B$, let us subdivide the set of light paths traced in one iteration into two halves containing N_{F_1} and N_{F_2} rays. Then, let at each iteration

$$\mathcal{B} \equiv \frac{1}{N_B}\sum_{j=1}^{N_B}\left(\left(\frac{1}{N_{F_1}}\sum_{i\in\text{half }_1} C(X_i^{(F)}, X_j^{(B)})\right)\left(\frac{1}{N_{F_2}}\sum_{i\in\text{half }_2} C(X_i^{(F)}, X_j^{(B)})\right)\right) \quad (3)$$

Its average over iterations is just $\langle\langle C\rangle_F^2\rangle_B$. Indeed, the average over iterations is $\langle\langle\cdot\rangle_F\rangle_B = \langle\langle\cdot\rangle_B\rangle_F$, because of linearity the order of averaging has no effect. For any camera path $X_j^{(B)}$ the two inner sums are *independent* because the two halves of light paths are statistically independent. The average of each of them is naturally $\langle C\rangle_F(X_j^{(B)})$. Therefore averaging it *over iterations*

$$\overline{\mathcal{B}} = \langle\langle\mathcal{B}\rangle_F\rangle_B = \langle\langle C\rangle_F\langle C\rangle_F\rangle_B = \langle\langle C\rangle_F^2\rangle_B \quad (4)$$

gives just what we need.

Similarly, to calculate $\langle\langle C\rangle_B^2\rangle_F$ we subdivide the set of camera paths (through *the* pixel) into two halves, N_{B_1} and N_{B_2}, calculate at each iteration

$$\mathcal{F} \equiv \frac{1}{N_F}\sum_{i=1}^{N_F}\left(\left(\frac{1}{N_{B_1}}\sum_{j\in half_1} C(X_i^{(F)}, X_j^{(B)})\right)\left(\frac{1}{N_{B_2}}\sum_{j\in half_2} C(X_i^{(F)}, X_j^{(B)})\right)\right) \tag{5}$$

and average it over iterations.

Notice the two halves have not to be equal (i.e. contain a different number of rays). Usually it is convenient to relate odd rays to the first half and even rays to the second half.

Knowing these quadratic averages and the limiting image luminance L one can calculate the variance of pixel luminance as

$$V = \frac{1}{N_F N_B}(\overline{\mathcal{C}} - L^2) + \frac{1 - N_F^{-1}}{N_B}(\overline{\mathcal{B}} - L^2) + \frac{1 - N_B^{-1}}{N_F}(\overline{\mathcal{F}} - L^2) \tag{6}$$

Notice $\overline{\mathcal{C}}, \overline{\mathcal{B}}, \overline{\mathcal{F}}$ as well as L depend on pixel but *are independent of the number of rays*. This latter property is very important because allows to predict noise level for any number of rays. One can perform calculation for some inoptimal number of rays and find $\overline{\mathcal{C}}, \overline{\mathcal{B}}, \overline{\mathcal{F}}, L$ and then find the number of rays (with N_B generally depending on pixel) which minimizes the noise.

3 Results

We traced the famous Cornell Box with an isotropic point light source slightly below the center of the floor. All scene surfaces are gray Lambert with albedo 50%. Both direct and indirect illumination was taken from photon maps. Integration sphere radius was 1/120 of the scene size. "Diffuse depth" i.e. the maximal allowed number of diffuse events for camera ray [9] was BDD=1. Scene image is shown in Figure 1.

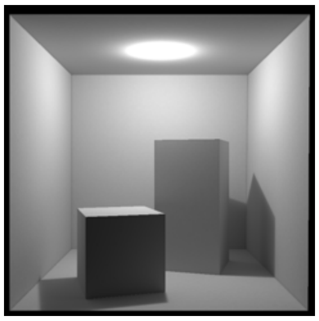

Fig. 1. Scene image for the variant of Cornell Box.

Pixel luminance calculated *in one iteration* is a random value which is *independent* from another iteration. Therefore the reference value of noise can be calculated *directly* from the *sample variance* of the series of iterations. Also, the above 3 noise terms had been calculated as described above and the total noise had been calculated from them according to (6).

The amplitude of noise calculated by these two ways are shown in Figure 2. One can see that these latter nearly coincide.

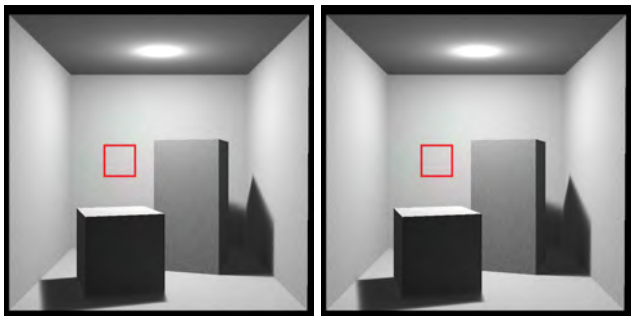

Fig. 2. Total noise as the r.m.s. of pixel value calculated in one iteration for $N_B = 100$, $N_F = 300000$. **Left panel**: the reference distribution, calculated from the sample variance. **Right panel**: $\sqrt{V}$ from (6). The average r.m.s. value over the red square is 0.637951 in the left image and 0.638509 in the right one.

The contribution of the three noise components is shown in Figure 3.

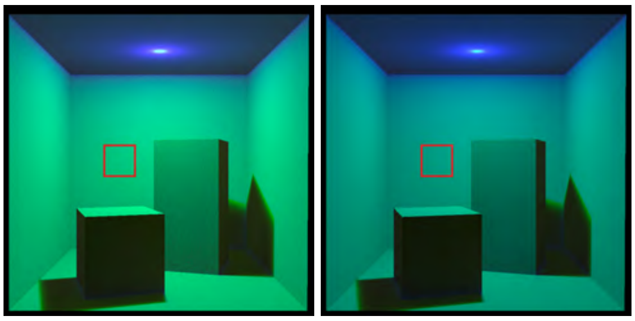

Fig. 3. Noise components $\sqrt{\frac{1}{N_F N_B}(\overline{\mathcal{C}} - L^2)}$, $\sqrt{\frac{1-N_F^{-1}}{N_B}(\overline{\mathcal{B}} - L^2)}$, $\sqrt{\frac{1-N_B^{-1}}{N_F}(\overline{\mathcal{F}} - L^2)}$ put into the R, G and B color channel of the image. **Left panel**: $N_B = 100$, **right panel**: $N_B = 300$. In both cases $N_F = 300000$. The average values of the three components over the red rectangle is $(0.0293, 0.595, 0.229)$ in the left image and $(0.0169, 0.344, 0.230)$ in the right one, so the first two components decreased just $\sqrt{3}$ times. Average r.m.s. over the red rectangle is 0.639 in the left image and 0.414 in the right one.

In Figure 3, $\sqrt{\frac{1}{N_F N_B}(\overline{\mathcal{C}} - L^2)}$, $\sqrt{\frac{1-N_F^{-1}}{N_B}(\overline{\mathcal{B}} - L^2)}$ and $\sqrt{\frac{1-N_B^{-1}}{N_F}(\overline{\mathcal{F}} - L^2)}$ are put into the R, G and B color channels. One can see that the image is mainly green, i.e. the second component dominates. Since $\overline{\mathcal{C}}, \overline{\mathcal{B}}, \overline{\mathcal{F}}$ and L are independent of the number of rays, to reduce noise one needs to increase the number of camera rays per pixel N_B while increase of the number of light rays is nearly useless. Results for N_B increased to 300 are shown in the right panel of Figure 3.

4 Conclusion

We proposed a method for calculation the components $\overline{\mathcal{C}}, \overline{\mathcal{B}}, \overline{\mathcal{F}}$ of the noise in BDPM and demonstrated how it can be used to choose the optimal number of rays traced in one iteration of the BDPM method.

The noise itself can be calculated directly from the sample variance of the series of values accumulated in successive iterations, but this does not let one decide which number of rays (camera or light) is critical and must be increased. Meanwhile knowing the three noise components $\overline{\mathcal{C}}, \overline{\mathcal{B}}, \overline{\mathcal{F}}$ one can see which of them dominates and thus which number of rays (camera or light) must be increased. If time spent on tracing camera and light rays is known together with time spent on their merging, one can also calculate the optimal number of rays i.e. the one which gives the smallest noise during the same calculation time.

References

1. Pharr, M., Humphreys, G.: Physically Based Rendering, Second Edition: From Theory To Implementation. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edn. (2010).
2. Dodik, N.: Implementing probabilistic connections for bidirectional path tracing in the Mitsuba Renderer (Sep 2017), https://www.cg.tuwien.ac.at/research/publications/2017/dodik-2017-pcbpt. Last accessed 01 Sep 2020.
3. Jensen, H.W., Christensen, P.: High quality rendering using ray tracing and photon mapping. In: ACM SIGGRAPH 2007 Courses. ACM, New York, NY, USA (2007).
4. Veach, E.: A dissertation: Robust Monte-Carlo methods for light transport simulation (1997), http://graphics.stanford.edu/papers/veach_thesis/thesis.pdf. Last accessed 01 Sep 2020.
5. Vorba, J.: Bidirectional photon mapping. In: Proceedings of CESCG 2011: The 15th Central European Seminar on Computer Graphics. pp. 25–32. Charles University, Prague (2011).
6. Ershov, S.V., Zhdanov, D.D., Voloboy, A.G.: Estimation of noise in calculation of scattering medium luminance by MCRT. Mathematica Montisnigri **45**, 60–73 (2019).
7. Georgiev, I., Křivánek, J., Davidovič, T., Slusallek, P.: Light transport simulation with vertex connection and merging. ACM Trans. Graph. **31**(6), 192:1–192:10 (Nov 2012).
8. Hachisuka, T., Pantaleoni, J., Jensen, H.W.: A path space extension for robust light transport simulation. ACM Trans. Graph. **31(6)**, 191:1–191:10 (2012).
9. Ershov, S.V., Zhdanov, D.D., Voloboy, A.G.: Calculation of MIS weights for bidirectional path tracing with photon maps. In: Proceeding of conference on Computing for Physics and Technology. pp. 20:1–20:5. Moscow (2020).
10. Sbert, M., Havran, V., Szirmay-Kalos, L.: Multiple importance sampling revisited: breaking the bounds. EURASIP Journal on Advances in Signal Processing **15**, 1–15 (2018).
11. Hachisuka, T., Jensen, H.W.: Stochastic progressive photon mapping. In: ACM SIGGRAPH Asia 2009 Papers. pp. 141:1–141:8. ACM, New York, NY, USA (2009).