# Bi-directional Rays in Global Illumination [1]

**Csaba Kelemen, Balázs Benedek, László Szirmay-Kalos, László Szécsi**

Department of Control Engineering and Information Technology, TU Budapest,
Budapest, Magyar Tudósok krt. 2., H-1117, HUNGARY
Email: szirmay@iit.bme.hu

## ABSTRACT

Global illumination algorithms use rays to transfer the light in the scene. A ray connects two points that are mutually visible from each other. Although it seems intuitive to transfer the light into both directions along the ray, only a few algorithms have taken advantage of bi-directional rays so far, which can increase the samples for minimal additional cost. In this paper we propose a non-diffuse global illumination algorithm that exploits this promising alternative. The algorithm is easy to implement but robust and is able to render complex scenes within a few tens of second.

**Keywords:** Global illumination, finite-element techniques, Monte-Carlo methods.

## 1 Introduction

In order to simulate the bounces of light, global illumination algorithms repeat the same elementary operation which transfers the radiance of a point to that point or points which are visible from here. The visibility between two points is mutual, thus it seems to be worth transporting the radiance into both directions. However, with a few exceptions [4, 1] most of the global illumination algorithms apply uni-directional rays and ignore the transfer from the other direction, which would be available almost free of charge. In the following section, a ray-based stochastic iteration algorithm is presented that can efficiently exploit the additional samples of bi-directional rays. Global illumination algorithms evaluate the light transport operator:

$$L^r(\vec{x},\omega) = \int_S L(\vec{y},\omega') \cdot f_r(\omega',\vec{x},\omega) \cdot G(\vec{x},\vec{y})\ d\vec{y}.$$

In this equation $L$ is the radiance of point, which is the sum of the emission $L^e$ and reflection $L^r$, $\omega'$ points from $\vec{y}$ to $\vec{x}$, $f_r$ is the BRDF, and $G$ is the geometric factor:

$$G(\vec{x},\vec{y}) = v(\vec{x},\vec{y}) \cdot \frac{\cos\theta_{\vec{x}} \cdot \cos\theta_{\vec{y}}}{|\vec{x}-\vec{y}|^2}$$

where $v$ is the visibility function which is 1 if the two points are mutually visible and zero otherwise, and $\theta_{\vec{x}}, \theta_{\vec{y}}$ are the angles between the surface normals and the direction connecting $\vec{x}$ and $\vec{y}$.

Iteration algorithms rely on the fact that the fixed point of iteration $L^{(n)} = L^e + \mathcal{T}_{f_r} L^{(n-1)}$ is the solution of the rendering equation. Thus if this formula is iterated with a contractive light transport operator, then the process will converge to the solution from an arbitrary initial function. Iteration requires the radiance function to be stored, which is made possible by finite-element techniques. Suppose that the surfaces are decomposed to patches of area $A_1, A_2, \ldots, A_K$ and after each iteration step the average radiance is stored on each patch. Thus the stored directional function on patch $i$ is:

$$L_i^r(\omega) = \frac{1}{A_i} \cdot \int_{A_i} L^r(\vec{x},\omega)\ d\vec{x}.$$

Substituting this into the iteration formula, we obtain for $L_i^{r,(n)}(\omega)$:

$$\frac{1}{A_i} \cdot \int_{A_i} \int_{S} L^{(n-1)}(\vec{y}, \omega') \cdot f_r(\omega', \vec{x}, \omega) \cdot G(\vec{x}, \vec{y}) \; d\vec{y} d\vec{x}. \tag{1}$$

Iteration uses the complete radiance function, thus the radiance of all patches should be transported, which can take a long time. In practical cases the neighboring patches are usually very similar, thus it would be enough to select one of them and transfer the total power of the neighborhood at once from here. This idea is exploited by Monte-Carlo radiosity [2, 3]. Monte-Carlo radiosity selects a patch just with certain probability, but the transferred radiance is divided with this probability.

We use a random transport operator $\mathcal{T}_{f_r}^*$ that behaves as the real operator in the expected case. The randomization of the light transport operator will select two points $\vec{y}$ and $\vec{x}$ in order to transfer the radiance of point $\vec{y}$ to $\vec{x}$ with probability density $p(\vec{y} \to \vec{x})$. Thus the random reflected radiance $L^{rr}$ generated by the random transport operator is

$$L^{rr} = \mathcal{T}_{f_r}^* L = \frac{1/A_i \cdot L(\vec{y}, \omega') \cdot f_r(\omega', \vec{x}, \omega) \cdot G(\vec{x}, \vec{y})}{p(\vec{y} \to \vec{x})}.$$

Replacing the light transport operator by a random operator, on the other hand, results in an iteration scheme that does not converge, but the iterated radiance functions will fluctuate around the real solution. However, if we compute an image from the radiance estimates of different iteration steps, the average of the images will really converge to the real image [5].

A good random transport operator introduces minimum variance, which is met if the probability density of selecting an $\vec{x}, \vec{y}$ pair mimics the integrand

$$l(\vec{y} \to \vec{x}) = \frac{1}{A_i} \cdot L(\vec{y}, \omega') \cdot f_r(\omega', \vec{x}, \omega) \cdot G(\vec{x}, \vec{y}) =$$

$$L(\vec{y}, \omega') \cos\theta_{\vec{y}} \cdot \frac{v(\vec{x}, \vec{y}) \cos\theta_{\vec{x}}}{|\vec{x} - \vec{y}|^2} \cdot \frac{f_r(\omega', \vec{x}, \omega)}{A_i}.$$

Unfortunately, we can sample with a probablity density that mimics not the whole integrand, only a part of it. We can select, for example, source point $\vec{y}$ and source direction $\omega'$ with probability density $p(\vec{y}, \omega')$ that is proportional to $L(\vec{y}) \cos\theta_{\vec{y}}$ and shoot a ray to indentify the receiver point $\vec{x}$. Note that such sampling would find $\vec{x}$ with probability density $L(\vec{y}, \omega') \cdot G(\vec{x}, \vec{y})$ which is the first two major factors of the integrand. Such sampling will also produce a density for backward transfers from $\vec{x}$ to $\vec{y}$, thus with each ray we have two random samples. Note that we have two estimators, where the backward estimator is poorer since the probability density mimics the forward transfers from $\vec{y}$ to $\vec{x}$, thus the backward estimates will have higher variance. Taking into account the two techniques with equal weights, we also inherit the disadvantages of backward estimates. Fortunately, more sophisticated combination of the two techniques is also possible as suggested by multiple importance sampling [7, 1]. Balanced heuristic of multiple importance sampling [7] divides the integrand by the average probability density of the methods to be combined no matter which method has generated the given sample.

## 2 A bi-directional global illumination algorithm

Let us first sample $\vec{y}$ and $\omega'$ with $p(\vec{y}, \omega')$ that is proportional to $L(\vec{y}, \omega') \cos\theta_{\vec{y}}$ and then $\vec{x}$ given $\vec{y}$ by the ray shooting process. This can be realized by first finding the patch $i$ of $\vec{y}$ proportionally to its power, i.e. the selection probability is the ratio of the power of patch $j$

$$\Phi_j = A_j \cdot \int_{\Omega} L_j(\omega') \cos\theta_{\vec{y}} \; d\omega',$$

and the total power of the scene $\Phi = \sum_k \Phi_k$. If a ray transfers the radiance on several wavelengths simultaneously, then radiance $L$ and total power $\Phi$ are vectors, thus this formula should be slightly modified to obtain a scalar probability density $p$. We can, for example, use the luminance $\mathcal{L}(L)$ of radiance $L$, which computes a weighted average of the radiances of different wavelengths. Replacing the radiance and the power by their luminances, we obtain:

$$p(\vec{y} \to \vec{x}) = \frac{\mathcal{L}(L(\vec{y}, \omega')) \cos\theta_{\vec{y}}}{\mathcal{L}(\Phi)} \cdot \frac{v(\vec{x}, \vec{y}) \cos\theta_{\vec{x}}}{|\vec{x} - \vec{y}|^2}.$$

Note that $p(\vec{y} \to \vec{x})$ is the probability density of sampling two points, thus this can be used not only when the radiance is transferred from $\vec{y}$ to $\vec{x}$, but also when — taking advantage of the bi-directional ray concept — the radiance is transferred from $\vec{x}$ to $\vec{y}$. Thus we have two Monte-Carlo estimates for the transfer, a classical one, called *forward estimator*, and a reverse one, called *backward estimator*.

### 2.1 Representing the patch radiance

Since the patches may be non-diffuse, their radiance is a function of the outgoing direction. Our goal is to avoid the complete representation of this function, because that would pose prohibitive memory requirements. If a patch is hit by a ray in iteration $n$, its irradiance $I(n) = l/p \cdot \cos\theta_n$ (i.e. the incoming radiance estimate multiplied by the cosine of the incoming angle) and the direction of the ray $\omega_n$ are stored on the patch. For those patches that are not hit by the ray, the irradiance of this iteration step is zero. From the irradiance and incoming direction, the radiance of the patch in an arbitrary direction $\omega'$ can be obtained as $I(n) \cdot f_r(\omega_n, \omega')$. Examining this sequence, we can note that it has a high fluctuation, it is mostly zero but when the patch is lucky enough to be hit by a ray, then it gets a larger contribution. The variance of the whole method can be reduced if the fluctuation of this sequence is decreased. The general idea is to replace $I(n)$ sequence by another sequence which smoother but still results in the correct reflected radiance when averaging takes place. Two techniques are presented, the first is based on the main part separation [6] and the second applies random acceptance and rejection similar to Metropolis Sampling [8].

The first method separates the constant main part of the reflected radiance, which is replaced by its average. Let us store the directional average of the reflected radiance in variable $L^{d,(n)}$ in each patch computed as

$$L^{d,(n)} = \frac{1}{n} \cdot \sum_{m=1}^{n} I(m) \cdot \frac{a(\omega_m)}{\pi},$$

where $a(\omega')$ is the albedo of the material. Note that this main part is computed not only from the last transfer but from the average of all transfers happened so far. We could take advantage of the fact that the main part is independent of the outgoing direction, thus an average could be computed that is valid for all directions. This technique reduces the general fluctuation but the variation of the transfers represented by the difference BRDF ($\Delta f_r = f_r - a/\pi$) still remains high in the sequence. The second variance reduction technique solves this problem without requiring to store more than a single incoming direction and irradiance per patch.

This method reduces the fluctuation of a sequence replacing by another sequence of ,,similar samples", thus zero samples are ignored, large samples of the original sequence will be scaled down and small samples will be scaled up. In order not to distort the average of the sequence, a scaled down larger value will appear more times in the new sequence sequence. An appropriate scaling of $I(n)$ is

$$\frac{I(n)}{n \cdot \mathcal{L}(I(n)a)} \cdot C, \quad \text{where } C = \frac{1}{n} \cdot \sum_{k=1}^{n} \mathcal{L}(I(k)a)$$

since it makes the elements of the series as equal as the luminance can reflect the importance of a spectrum. In this formula $a$ is the albedo of the surface if this method is used alone, or the albedo of the absolute value of the difference BRDF $\Delta a$ if this method is applied together with the separation of the main part. The average will be correct if we can guarantee that $I(n)$ is expected to appear $\mathcal{L}(I(n)a)/C$ times. A sampling scheme that can produce samples proportionally with $\mathcal{L}(I(m)a)$ is based on random acceptance and rejection. At each iteration step the new irradiance $I(n)$ is compared with the stored irradiance $I(m)$. If $\mathcal{L}(I(n)a)$ is greater or equal than $\mathcal{L}(I(m)a)$, then the new irradiance will replace $I(m)$ in the random representation of the radiance. However, when $\mathcal{L}(I(n)a)$ is smaller than $\mathcal{L}(I(m)a)$, the new irradiance is accepted with probability $\mathcal{L}(I(n)a)/\mathcal{L}(I(m)a)$.

### 2.2 Implementation details

Note that in the formulae of the reflected radiance, the main part and the irradiance appear to be divided by the actual iteration number. This means that the radiances of all patches change in an iteration, thus the computational complexity of a single iteration loop is linear with the number of patches. This is clearly undesirable in complex scenes. Fortunately, the update of all patches can be avoided if we store variables that are multiplied with the iteration number and carry out the division on the fly. The iteration cycle also contains a hidden loop to consider every single patch, which is responsible for the random patch selection. This includes the calculation of the selection probabilities and then the random selection itself. The selection probabilities are proportional to the luminance of the power of the patches, which is the sum of the luminance of the emission and the luminance of the reflected radiance (diffuse and specular reflections). To speed up the update of the selection probabilities and the random selection according to these probabilities, we use a binary tree and a binary search process. The leaves of this tree correspond to the patches and contain

two values, the luminance of the emitted power and the luminance of the reflected power multiplied by the current iteration number. The nodes of the tree also contain two values, which are the sums of the corresponding values of the two children of the node. Finally the root of the tree has the luminance of the total emitted power and the luminance of the total reflected power multiplied by the iteration number. Note that each node contains two variables, an emission and a reflection times iteration number, from which the luminance of the power of the node will be computed on the fly. The on-the-fly computation can allow us not to visit all nodes of the tree at each iteration to update the iteration number. The update of this data structure is also logarithmic. Suppose that the reflected radiance at a patch have changed. The luminance of its increment added to the reflected radiance variable of the leaf and this value is propagated upwards adding the increment also to the parents, grandparents, etc. of this leaf until the root is reached.

Figure 1: An architectural scene: 5 million iterations, 80 sec rendering time, 78443 patches

## 3 Conclusions

In this paper we proposed a stochastic iteration algorithm that uses bi-directional rays and applies multiple importance sampling to optimally combine the forward and backward methods. The application of bi-directional rays itself provides a significant increase of samples in the integral quadratures, especially for the computation of higher order bounces, for very small additional cost. We also proposed a low variance random approximation for the non-uniform directional radiance. This approximation speeds up the convergence, and can also be regarded as a good initial guess, when the camera moves, thus we can expect fast convergence during animated walkthroughs. Finally we have shown, how the random selection can be supported by a binary tree, which results in an overall sub-linear algorithm. Future improvements include the combination of different bi-directional strategies in the framework of multiple importance sampling.

## 4 Acknowledgements

This work has been supported by the National Scientific Research Fund (OTKA ref. No.: T029135), the Bolyai Scholarship, by Intel and by the Slovene-Hungarian Action Fund (SLO7/01). The scenes have been modeled by ArchiCAD (Graphisoft).

## REFERENCES

[1] Ph. Bekaert, M. Sbert, and Y. D. Willems. The computation of higher-order radiosity approximations with a stochastic jacobi iterative method. In *Spring Conference of Computer Graphics '00*, 2000.

[2] L. Neumann. Monte Carlo radiosity. *Computing*, 55:23–42, 1995.

[3] S. N. Pattanaik and S. P. Mudur. Adjoint equations and random walks for illumination computation. *ACM Transactions on Graphics*, 14(1):77–102, 1995.

[4] M. Sbert. *The Use of Global Directions to Compute Radiosity.* PhD thesis, Catalan Technical University, Barcelona, 1996.

[5] L. Szirmay-Kalos. Stochastic iteration for non-diffuse global illumination. *Computer Graphics Forum (Eurographics'99)*, 18(3):233–244, 1999.

[6] L. Szirmay-Kalos, F. Csonka, and Gy. Antal. Global illumination as a combination of continuous random walk and finite-element based iteration. *Computer Graphics Forum (Eurographics'2001)*, 20(3):288–298, 2001.

[7] E. Veach and L. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Rendering Techniques '94*, pages 147–162, 1994.

[8] E. Veach and L. Guibas. Metropolis light transport. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 65–76, 1997.