PROGRESSIVE LIGHT PATH BUILDING

László Szirmay-Kalos, György Antal and Mateu Sbert

Department of Control Engineering and Information Technology, TU of Budapest
Budapest, Műegyetem rkp. 11, H-1111, HUNGARY
szirmay@iit.bme.hu

ABSTRACT

The paper proposes a global illumination method that builds up the light paths progressively taking into account all relevant previous information. The basis of the method is a self-correcting stochastic iteration scheme, which works with a population of photon hits. In each iteration step a ray is generated randomly either from a light source or by reflecting an earlier hit, then the ray is traced to obtain a new hit. In order to limit the size of the hit population, hits are decimated randomly after certain iteration steps. Comparing the new approach to random walk techniques, this method can reuse the illumination and visibility information gathered with previous rays. By defining the decimation strategy properly, the view-importance can be built into the algorithm.

Keywords: Global illumination, stochastic iteration, light-tracing

1 Introduction

Global illumination algorithms are expected to find the possible light paths connecting the light sources to the camera [16]. Paths can be build starting at the light sources (shooting) [14, 4], starting at the camera (gathering) [7], or even simultaneously starting two paths, one from the eye and one from the camera and connecting them (bi-directional methods) [9, 20]. Random walk algorithms usually build up the light paths independently, that is, having a path established and its carried power computed, this path is thrown away, and the new path is started from scratch. This means that random walk methods do not reuse the visibility and illumination information gathered by previous walks, which can be very inefficient if the lighting situation is difficult, that is when just a small fraction of the paths may contribute to the image. Having the random process eventually found a light path of non-zero contribution, this knowledge is used just once, then the hopeless search is started again. It is not surprising that methods aiming at faster solutions try to reuse the previous information, for example, by combining the random walk with finite-element techniques [3] or using special data structures and heuristics [23, 22]. The method called Metropolis light transport [21] has proposed the perturbation of the current path to obtain a new light path. Although it remembers only the last path, even this limited memory can significantly improve the speed if the lighting is difficult. However, when the lighting is not difficult the start-up bias problem of this algorithm may invalidate the benefits of the approach [19]. Adaptation can be built into random walk approaches also by building a data structure during walks, which can store "discretized importance" information about regions and sets of directions [10]. The Rayvolution algorithm [11] demonstrated that the concepts of genetic algorithms are also worth considering to find adaptive global illumination methods.

Two-phase approaches also offer the reuse of the information gathered in the first-phase. An efficient way of storing the result of the first phase is the photon map [5]. These methods can be viewed as special bi-directional methods where a gathering path is connected to all shooting paths simultaneously [8, 6].

In this paper we propose a shooting like one-phase approach. The adaptation strategy used in the single phase is different from what is used in Metropolis sampling and in discretized importance based techniques. The new strategy is able to use a much greater portion of the information about previous paths than the Metropolis algorithm to guide the generation of new light paths, and this information is more precise than that is stored in discretized importance maps. In theory, it would be possible to use the gathered knowledge of all previous paths, but it would require prohibitive memory. In order to reduce the storage demand, the data structure remembering the previous path is decimated regularly in a way which can extract relevant information.

Mathematically, the method is based on an improved stochastic iteration solution of the rendering equation of the following form

$$L(\vec{x},\omega) = L^e(\vec{x},\omega) + (\mathcal{T}L)(\vec{x},\omega) \qquad (1)$$

where $L(\vec{x},\omega)$ is the radiance at point $\vec{x}$ in direction ω, L^e is the emission, and $\mathcal{T}$ is the light-transport operator.

We first review how stochastic iteration can be adapted when the radiance is transferred by rays carrying power on all wavelengths simultaneously and when the local albedo can only be approximated. Then the concept of self-correcting iteration is explained and used with transferring the radiance by single rays. We also discuss the random hit population control and finally present implementation details and running time statistics.

2 Stochastic iteration with a single ray

The concept of stochastic iteration has been proposed in [18] as a tool to attack the non-diffuse global illumination problem. Here we briefly summarize the idea for a single ray and discuss the calculation of the image contribution. In stochastic iteration a random transport operator is used instead of the light transport operator in the iteration sequence, which gives back the original light transport operator in the average case:

$$L_m = L^e + \mathcal{T}^* L_{m-1}, \qquad E[\mathcal{T}^* L] = \mathcal{T}L. \quad (2)$$

Then the pixel colors are computed as an average of the estimates of all iteration steps

$$P_m = \frac{1}{m} \cdot \sum_{i=1}^{m} \mathcal{M}L_i = \tau_m \cdot \mathcal{M}L_m + (1-\tau_m) \cdot P_{m-1} \quad (3)$$

where $\tau_m = 1/m$ and $\mathcal{M}$ is the measuring operator that computes the pixel color from the radiance of the surfaces visible in it.

Let the random transport operator use a single ray having random origin $\vec{y}_i$ and direction ω_i generated with a probability density $p(\vec{y},\omega)$ that is preferably proportional to the cosine weighted radiance of this point at the given direction. This ray transports the power

$$\Phi(\vec{y},\omega') = \frac{L(\vec{y},\omega') \cos\theta_{\vec{y}}}{p(\vec{y},\omega')}$$

to that point $\vec{x}$ which is hit by the ray, where it is reflected, modifying the radiance function. On a single wavelength, the probability of reflection is the BRDF times the cosine of the outgoing angle, i.e.

$$f_r(\omega_i,\vec{x},\omega) \cdot \cos\theta_{\vec{x}},$$

but the cosine angle is compensated when the power is converted to radiance. Formally, the random transport operator is

$$(\mathcal{T}^* L)(\vec{x},\omega) = \frac{L(\vec{y},\omega') \cos\theta_{\vec{y}}}{p(\vec{y},\omega')} \cdot \delta(\vec{x} - h(\vec{y}_i,\omega_i)) \cdot f_r(\omega_i,\vec{x},\omega), \quad (4)$$

where $h(\vec{y}_i, \omega_i)$ is the visibility function which selects that point which is visible from $\vec{y}_i$ in direction ω_i. Using the definition of the solid angle, $d\omega_{\vec{y}} = d\vec{x} \cdot \cos\theta'_{\vec{x}} / |\vec{y} - \vec{x}|^2$, a symmetry relation can be established

$$d\vec{y} \cdot d\omega_{\vec{y}} \cdot \cos\theta_{\vec{y}} = d\vec{y} \cdot \frac{d\vec{x} \cdot \cos\theta'_{\vec{x}}}{|\vec{y} - \vec{x}|^2} \cdot \cos\theta_{\vec{y}} =$$

$$d\vec{x} \cdot \frac{d\vec{y} \cdot \cos\theta_{\vec{y}}}{|\vec{y} - \vec{x}|^2} \cdot \cos\theta'_{\vec{x}} = d\vec{x} \cdot d\omega'_{\vec{x}} \cdot \cos\theta'_{\vec{x}},$$

which allows us to easily prove that the requirement of equation (2) holds, that is, the expectation of the random transport operator defined in equation (4) really gives back the original light transport operator.

Let us discuss how this algorithm works. Suppose that the first random operator $\mathcal{T}_1^*$ is applied to L^e thus the light sources should be sampled with probability density p_e resulting in a point $\vec{y}_1$, direction ω_1, and ray power

$$\Phi_1(\vec{y}_1, \omega_1) = \frac{L^e(\vec{y}_1, \omega_1) \cos\theta_{\vec{y}_1}}{p_e(\vec{y}_1, \omega_1)}.$$

This power is sent to a single point $\vec{x}_1 = h(\vec{y}_1, \omega_1)$ that is hit by the ray. Before continuing with the second step of the iteration, the radiance should be measured, that is, an image estimate should be computed from $L^e + \mathcal{T}_1^* L^e$. We can separately calculate the effect of the light sources on the image and then add the effect of $\mathcal{T}_1^* L^e$. Note that $\mathcal{T}_1^* L^e$ is concentrated in a single point, thus its contribution can be computed by tracing a ray from the eye to this point, and if this point is not occluded, then adding

$$\Phi_1 \cdot f_r(\omega_1, \vec{x}, \omega_{\text{eye}}) \cdot \cos\theta_{\vec{x}} \cdot g(\vec{x})$$

to that pixel in which $\vec{x}$ is visible. Function g is the weight associated with the pixel in which $\vec{x}$ is visible [4]:

$$g(\vec{x}) \approx \frac{f^2}{|\vec{x} - \vec{\text{eye}}|^2 \cdot S_p \cdot \cos^3\theta_{\text{pix}}}$$

where f is the focal distance of the camera, i.e. the distance between the eye and the plane of the window, $\vec{\text{eye}}$ is the eye position, S_p is the area of the pixel, and θ_{pix} is the angle between the pixel normal and the viewing direction.

The second operator $\mathcal{T}_2^*$ should be applied to $L_1 = L^e + \mathcal{T}_1^* L^e$, thus the domain of non-zero radiance is modified, which requires a new probability density to be constructed. Suppose that first it is decided randomly whether the selected point is the new point $\vec{x}$ or the light source is sampled again. Let the probability of selecting the new point be $s_{\vec{x},\omega_1}$ which may depend on both point $\vec{x}$ and previous direction ω_1. If the hit point is selected, then the direction is sampled with a probability density $p_{\vec{x},\omega_1}(\omega_2)$. If the light source is selected, then we can use again the probability density p_e that was applied in the previous step. The new combined probability density is

$$p_{\vec{x},\omega_1}(\vec{y}_2, \omega_2) =$$

$$p_e(\vec{y}_2, \omega_2) \cdot (1 - s_{\vec{x},\omega_1}) + p(\omega_2) \cdot \delta(\vec{y}_2 - \vec{x}) \cdot s_{\vec{x},\omega_1}.$$

Having defined the ray, it is traced and the contribution of the hit onto the camera is computed.

The algorithm keeps doing this in each iteration. First it is decided randomly whether the new ray will start at the hit point or at the light source. Then either the light source or the directions around the hit point are sampled, and the resulting ray is traced, which defines a hit point of the following iteration step. Before repeating the random selection, the effect of the hit point and the light sources on the camera is computed and averaged in an image. This average will converge to the final solution. Interestingly this iteration is similar to a sequence of variable length random walks, since at each step the point that is last hit by the ray is selected with a given probability as the starting point of the next ray. If not the hit is selected, then the iteration is continued by sampling the light source, which can be considered as terminating the previous walk. The termination probability is $s_{\vec{x},\omega_i}$ at each step.

So far, we have had complete freedom to choose probabilities $p_e(\vec{y}, \omega)$, $p_{\vec{x},\omega_i}(\omega_{i+1})$ and $s_{\vec{x},\omega_i}$. According to the concept of importance sampling, these should be set to force the random variable representing the transported power to have low variance, that is close to constant. The variance of the

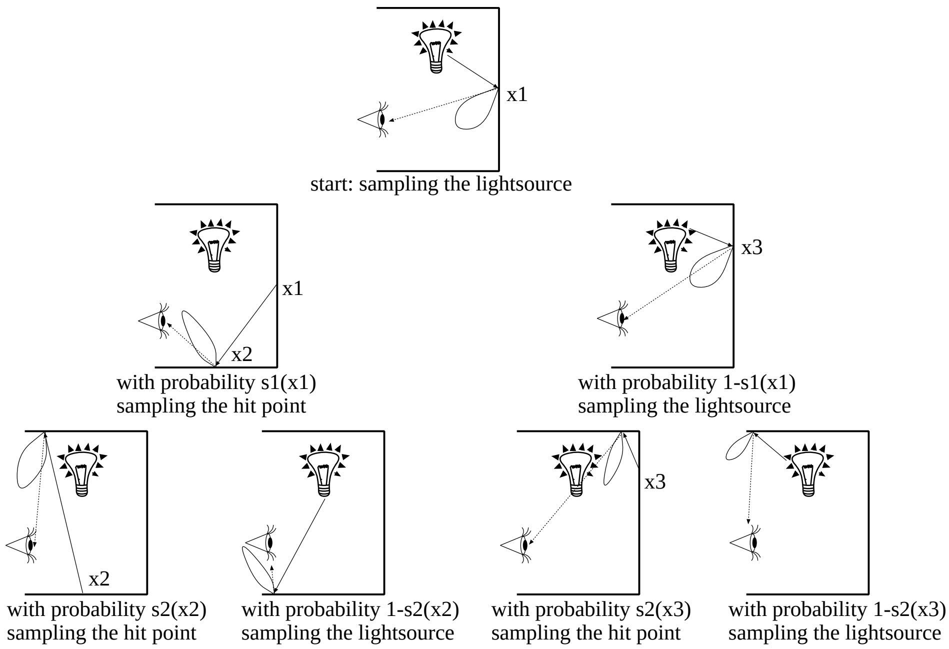

Figure 1: Four possible evolutions of two iteration steps

transported ray is reduced if p_e is proportional to $L^e(\vec{y}, \omega') \cos\theta_{\vec{y}}$ or its luminance if the calculation takes place on different wavelengths. Similarly $p_{\vec{x},\omega_i}(\omega_{i+1})$ should be proportional to the cosine weighted BRDF, i.e. to $f_r(\omega_i, \vec{x}, \omega_{i+1}) \cos\theta_{\vec{x}}(\omega_{i+1})$ or its corresponding luminance. Finally probability s should mimic the ratio of the power (or luminance) reflected at the last visited point and the total power (total luminance). It can be shown that for closed scenes s will converge to the average albedo. For open scenes, s will be smaller and equal to the average contraction of the light-transport operator.

3 Self-correcting iteration

Note that stochastic iteration uses the radiance estimate L_m only for the calculation of the final pixel colors (equation (3)). A better iteration scheme can be obtained if it is also used to continue the iteration in the following way:

$$\begin{aligned} L'_m &= L^e + \mathcal{T}^*_m L_{m-1}, & (5)\\ L_m &= \tau_m \cdot L'_m + (1-\tau_m) \cdot L_{m-1}, & (6)\\ P_m &= \mathcal{M} L_m. & (7) \end{aligned}$$

This scheme is called as the self-correcting iteration since it is similar to the radiosity iteration of [12]. Comparing self-correcting iteration to the discussed stochastic iteration, we can conclude that while in stochastic iteration the last hit point competes with the light source surface to be the origin of the next ray, in self-correcting iteration all previous hit points have some chance to spawn the next ray. Since it uses more information about the past, it converges faster.

Hit point i is characterized by its incoming direction hits$[i].\omega^{in}$, incoming power hits$[i].\Phi$ and location hits$[i].\vec{x}$. Suppose that at step m hit point i is selected with probability s_i, and the light source is sampled with probability $1 - s_1 - \ldots - s_n$ if n is the number of hit points. If hit point i is selected, then the direction of the ray is sampled from density $p_i(\omega_m)$ and the ray carries

$$\Phi_i \cdot \frac{f_r(\omega_i, \vec{x}_i, \omega_m) \cos\theta_{\vec{x}}}{p_i(\omega_m)}$$

power to a new hit point. This step corresponds to equation (5).

Then, according to equation (6), this new hit point is merged with the current hit population. This is done by multiplying the incoming power of the newly born hit point by τ_m

and the incoming power of all previous hit points by $1 - \tau_m$.

The following program summarizes a self-correcting iteration step, where "hits[]" is the array of photon hits, n_{hit} is the number of hits, and m is the current index of the iteration.

Iterate(hits[], n_{hit}, m)
  Select i with prob. s_i or the light source with $1 - \sum s_i$
  if light source is selected **then**
    Sample light source with $p_e(\vec{x}, \omega)$ to get $\vec{x}$ and ω
    $\Phi = L^e(\vec{x}, \omega) \cdot \cos\theta / p_e(\vec{x}, \omega)/(1 - \sum s_i)$
  else
    $\vec{x} = \text{hits}[i].\vec{x}$
    Sample the directional sphere with
      $p_{\text{hits}[i].\vec{x},\text{hits}[i].\omega^{in}}(\omega)$ resulting in ω
    $\Phi = \text{hits}[i].\Phi \cdot f_r(\text{ hits}[i].\omega^{in}, \text{hits}[i].\vec{x}, \omega) \cdot \cos\theta_{\vec{x}} /$
      $p_{\text{hits}[i].\vec{x},\text{hits}[i].\omega^{in}}(\omega)/s_i$
  endif
  for $i = 1$ **to** n_{hit} **do** hits[i].Φ *= $(1 - 1/m)$
  $\vec{y} = \text{Trace Ray}(\vec{x}, \omega)$
  if ray hits object **then**
    hits[n_{hit} + 1] = $(\vec{y}, \omega, \Phi/m)$
    $n_{hit} = n_{hit}$++
  endif
end

Running the iteration for a number of steps, the light source and the hit population are measured by operator $\mathcal{M}$ in the sense of equation (7).

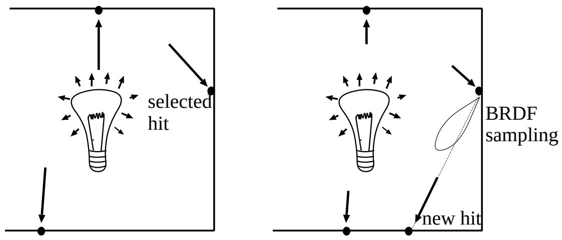

Figure 2: A self-correction iteration step

As in stochastic iteration, probabilities $s_{\vec{x},\omega^{in}}$ and $p_{\vec{x},\omega^{in}}$ should be defined to decrease the variance of the carried power, which corresponds to importance sampling. Thus it is recommended to set the directional probability $p_{\vec{x},\omega^{in}}$ to be proportional to the cosine weighted BRDF. Similarly, if we aim at the reduction of the variance of the output radiance, then hit selection probability s_i should be approximately proportional to the product of the incoming power of hit i and the local albedo at the hit point. However, considering the fact that just a fraction of the paths may eventually reach the camera, and our ultimate objective is to reduce the variance of the image, other hit selection probabilities are also worth considering.

3.1 Definition of hit selection probabilities

From the point of view of the impact on the image, the hit selection strategy should emphasize those hits whose children are likely to contribute the image and the contribution power should be made approximately constant. The image impact is proportional to the outgoing power and to the visibility indicator and is inversely proportional to the square distance. Thus we should prefer those hits that can generate large outgoing power and their possible children are not hidden from the camera and are close to it.

To achieve this goal, heuristics can be used, which build some form of view-importance into the algorithm. It is reasonable to believe that if the children of a hit have had large contribution to the image so far, then the neighborhood of this hit is not hidden and this hit is worth keeping as a mother of new hit points. From this point of view, the importance of a hit point can be characterized by the luminance of the average image contribution of the hit's children. Note that the impact of the child hit is the luminance of

$$\Phi_i \cdot f_r(\omega_i, \vec{x}_i, \omega_{\text{eye}}) \cos\theta_{\vec{x}} \cdot g(\vec{x}) \qquad (8)$$

if $\vec{x}$ is visible through some pixel and zero otherwise.

On the other hand, important hits may exist which have no children or their children have had no impact on the image yet. To take into account also these hits, we can also define the importance of a hit as the weighted sum of its output luminance (i.e. the luminance of the product of the incoming power and the local albedo). In order to convert this importance to a form similar to equation (8), we have to estimate how much contribution a hit could have if it were visible and its orientation allowed to reflect into the direction of the camera. This consideration leads to the following estimated *potential impact* of a parent:

$$\Phi_i \cdot \frac{a(\omega_i)}{|\vec{x} - \vec{\mathrm{eye}}|^2 \cdot 4\pi} \cdot \frac{f^2}{S_p} = \Phi_i \cdot a(\omega_i) \cdot g^*(\vec{x}), \quad (9)$$

where $a(\omega_i)$ is the albedo. Note that here the reflection probability density $f_r \cdot \cos\theta$ has been replaced by its average $a/4\pi$.

The importance of a hit can then be defined as the weighted average of its potential impact and the real impact of its children. The weighting factor is denoted by λ.

The algorithm of computing the hit selection probabilities is as follows:

```
ComputeImportance( hits[], n_hit )
    I_total = 0
    for i = 1 to n_hit do
        hits[i].s = 0        // hit selection probability
        hits[i].nchild = 0   // number of children
    endfor
    for i = 1 to n_hit do
        hits[i].s += Lum( potential impact of hits[i]
        pi = hits[i].parent   // id of the parent hit
        hits[pi].s += λ · Lum(impact of hits[i]
        hits[pi].nchild ++
    endfor
    for i = 1 to n_hit do
        hits[i].s /= (hits[i].nchild+1)
        I_total += hits[i].s
    endfor
    for i = 1 to n_hit do hits[i].s /= I_total
end
```

4 Random purge of the hit population

The self-correction iteration in the version proposed so far is not appropriate for practical implementations since each iteration step may increase the hit population by one, which results in memory overflow sooner or later. To avoid the overpopulation of hits, the algorithm is broken into phases. Each phase is like the presented self-correcting iteration. The image estimates are computed at the end of each phase and the final image is obtained as the average of images of the phases. The phases can either start from scratch or continue the previous phase with a strongly decimated hit population. The killing of hits happens in a way that the expectation of the power does not change. Each hit point is considered for killing. Suppose that the survival of hit i happens with probability k_i. If it is killed then its power will be zero. However, when it is given clemency, then its power is divided by k_i, which guarantees the correct expectation:

$$E[\Phi_i^{new}] = k_i \cdot \frac{\Phi_i}{k_i} + (1 - k_i) \cdot 0 = \Phi_i.$$

During this, those hits should stay alive which are likely to spawn ancestors that can have significant contribution on the image. Note that this is the same requirement as selecting the hit point to spawn the next hit, thus setting k_i proportional to s_i seems to be a good strategy.

5 Implementation details and simulation results

In the current implementation a hit-point is represented by the following parameters: incoming power on all wavelengths, incoming direction, location, pointer to the surface in order to get the BRDF, the surface normal, luminance of the product of the incoming power and the local albedo, image contribution, number of children, importance and the index of its parent. The hit structure is similar to the photon-map [5, 6]. The algorithm is broken into phases of 400 steps and at the end of each phase the image computation and the hit selection probability calculation happen simultaneously. Then the population is decimated keeping just a few hits and the iteration is continued.

Figure 3 shows the used test scene where the lighting is made difficult by placing the camera and the light source into two separate rooms connected by a small gap. The image on the right side has been computed by the proposed method. The walls are diffuse, but the objects have both specular and diffuse reflections. Figure 4 compares the

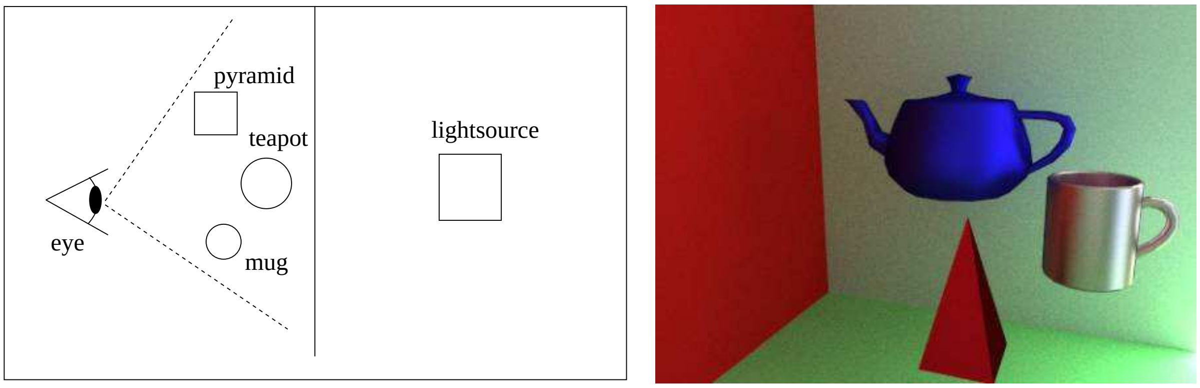

Figure 3: Geometry of the test scene (top view) and the image obtained with self-correcting iteration

new self-correcting iteration with the classical light tracing, where both images have been computed with the same computational effort (30 million rays). The variance of the image obtained with the new method is 14 times smaller than that of the light tracing. However, this does not mean that the algorithm is 14 times faster since the manipulation of the hit population has some overhead. For the current scene this overhead finally results in 7 times improvement. For more difficult scene, the overhead amortizes, making the relative speed even higher.

6 Conclusions and future developments

This paper presented a global illumination algorithm that works with a population of photon hits. At each iteration a randomly selected hit gives birth to a new hit point, and at the end of each phase an image estimate of the hit population is obtained and the hits are decimated randomly to start the next phase with a low number of hits. The hit selection can prefer those hits that have possibly large contribution on the image. This is similar to importance sampling, but when comparing to classical random walks we have to note two important differences. On the one hand, not only the outgoing luminance of the hit can be taken into account but also how far and how hidden its neighborhood is from the camera. On the other hand, a hit is not forgotten right after its selection, but may be selected again if it is important enough which allows to reuse previous information. This reuse is particularly important if the lighting situation is difficult and only a few paths can reach the light source.

Acknowledgements

This work has been supported by the National Scientific Research Fund (OTKA ref. No.: T029135) and the Spanish-Hungarian Fund, ref. No.: E9. The authors are greatful to Philippe Bekaert for providing the RenderPark program, which is used as a framework for the current implementation.

REFERENCES

[1] Ph. Bekaert. *Hierarchical and stochastic algorithms for radiosity.* PhD thesis, University of Leuven, 1999.

[2] M. R. Bolin and G. W. Meyer. An error metric for Monte Carlo ray tracing. In *Rendering Techniques '97*, pages 57–68, 1997.

[3] K. Bouatouch, S. N. Pattanaik, and E. Zeghers. Computation of higher order illumination with a non-deterministic approach. *Computer Graphics Forum*, 15(3):327–337, 1996.

[4] Ph. Dutre, E. Lafortune, and Y. D. Willems. Monte Carlo light tracing with direct computation of pixel intensities. In *Compugraphics '93*, pages 128–137, Alvor, 1993.

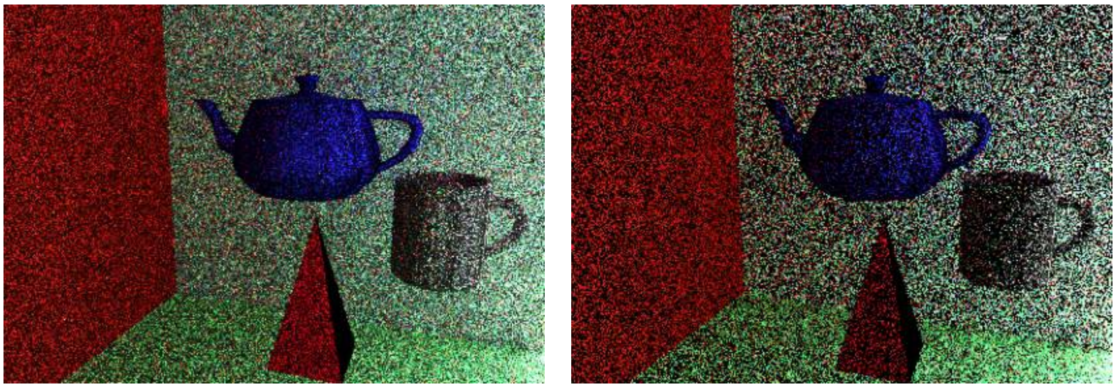

Figure 4: Comparison of self-correcting iteration (left) and classical light tracing (right). Both images have been computed with tracing 30 million rays.

[5] H. W. Jensen. Global illumination using photon maps. In *Rendering Techniques '96*, pages 21–30, 1996.

[6] H. W. Jensen and P. H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. *Computers and Graphics (SIGGRAPH '98 Proceedings)*, pages 311–320, 1998.

[7] J. T. Kajiya. The rendering equation. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, pages 143–150, 1986.

[8] A. Keller. Instant radiosity. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 49–55, 1997.

[9] E. Lafortune and Y. D. Willems. Bi-directional path-tracing. In *Compugraphics '93*, pages 145–153, Alvor, 1993.

[10] E. Lafortune and Y. D. Willems. A 5D tree to reduce the variance of Monte Carlo ray tracing. In *Rendering Techniques '96*, pages 11–19, 1996.

[11] B. Lange and B. Beyer. Rayvolution: An evolutionary ray tracing algorithm. In *Photorealistic Rendering Techniques*, pages 136–144, 1994.

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

[13] J. Nimeroff, J. Dorsey, and H. Rushmeier. Implementation and analysis of a global illumination framework for animated environments. *IEEE Transactions on Visualization and Computer Graphics*, 2(4), 1996.

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

[15] I. Peter and G. Pietrek. Importance driven contruction of photon maps. In *Rendering Techniques '98*, pages 269–280, 1998.

[16] P. Shirley. Time complexity of Monte-Carlo radiosity. In *Eurographics '91*, pages 459–466. Elsevier Science Publishers, 1991.

[17] P. Shirley, C. Wang, and K. Zimmerman. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, 1996.

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

[19] L. Szirmay-Kalos, P. Dornbach, and W. Purgathofer. On the start-up bias problem of metropolis sampling. In *Winter School of Computer Graphics '99*, pages 273–280, Plzen, Czech Republic, 1999.

[20] E. Veach and L. Guibas. Bidirectional estimators for light transport. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 419–428, 1995.

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

[22] G. J. Ward. The RADIANCE lighting simulation and rendering system. *Computer Graphics*, 28(4):459–472, 1994.

[23] J. Zaninetti, P. Boy, and B. Peroche. An adaptive method for area light sources and daylight in ray tracing. *Computer Graphics Forum*, 18(3):139–150, 1999.