International Journal of Innovative Computing, Information and Control
Volume **15**, Number **1**, February **2019**
ICIC International ©2019 ISSN 1349-4198
pp. **67–77**

FAST GLOBAL RENDERING IN VIRTUAL REALITY VIA LINEAR INTEGRAL OPERATORS

Hongxue Shen[1], Zhifeng Zhang[1,*] and Zhanlei Shang[2]

[1]Software Engineering College
[2]Engineering Training Center
Zhengzhou University of Light Industry
No. 5, Dongfeng Road, Zhengzhou 450002, P. R. China
{ ss_hongxue; shangzl }@zzuli.edu.cn; *Corresponding author: zfzhang80@126.com

Received April 2018; revised August 2018

Abstract. *Simulation of global illumination is very challenging for its expensive computing requirement. In general, rendering technique of global illumination requires several hours or even days to generate a smooth and physically correct image. Though the real-time rendering has been acquired recently, it is still difficult to be applied in some fast interactive applications such as virtual reality and 3D games nowadays. In this paper, we present the fast simulation of global illumination via linear integral operators based on Fredholm theorems. By deriving the parameterization of complex rendering equation, we can acquire a more approachable form of global illumination. Using the linear integral operators based on Fredholm theorems, we obtain a discretization solution of rendering equation of global illumination in the linear operator form, which is easy to be implemented on GPU (Graphics Processing Unit). The results show that we can acquire realistic effects of global illumination while implementing the high rendering speed.*
Keywords: Global illumination, Virtual reality, Linear integral operators, Realistic effects

1. **Introduction.** Global Illumination (GI) is a family of algorithms used in computer graphics that simulate how light interacts and transfers between objects in a scene. With its roots in the light transport theory, GI takes account of both the light that comes directly from a light source (direct lighting/illumination), as well as how this light is reflected by and onto other surfaces (indirect lighting/illumination). In general, effects such as color bleeding, reflection, caustics and soft shadow, should be included and simulated in the global rendering. So all combinations of diffuse, reflections and transmissions also have to be accounted for global illumination.

Global rendering algorithms can produce many amazing realistic effects, but generally the main or principal issue with global illumination is that no matter what you do, it is generally very expensive to simulate and difficult to simulate efficiently. In the past few decades, a variety of classic solutions such as path tracing, ray tracing and photon mapping to simulate the GI problem have been proposed and implemented. Path tracing and ray tracing [1, 2] are computer graphics Monte Carlo methods of rendering images of 3D scenes such that the global illumination is faithful to reality. However, a smooth and correct image generated by path tracing or ray tracing will cost very long time. Then, Jensen [3, 4] presents a practical guide to ray tracing and photon mapping. He makes it possible to efficiently compute global illumination including caustics, diffuse color bleeding, and participating media. However, their algorithms of global illumination have still

DOI: 10.24507/ijicic.15.01.67

been very time consuming and are only suitable for off-line computations. Recent researches in real-time ray tracing have improved global illumination performance to allow for illumination updates at faster rates. Based on the NVIDIA hardware, real-time ray tracing [5, 6] is presented to realize fast rendering of scenes illuminated with global light. However, the rendering speed is relatively low when rendering large scene in 3D games application. To accelerate the computation of photon mapping, the parallel implementation algorithms for photon mapping [7, 8] are presented to demonstrate that current graphics hardware is capable of fully simulating global illumination. Similarly, Zhang et al. [9] present a framework for pre-computed photon mapping that achieves real-time rendering of global illumination effects for volume data sets such as multiple scattering, and volumetric shadows. Later, by constructing multiple reference octrees on GPU in application, Frolov et al. [10] introduce an efficient and a simple algorithm for photon mapping and irradiance caching techniques [11]. With the rapid development of graphics hardware, GI has become increasingly attractive even for real-time applications nowadays. However, the computation of physically-correct global illumination still cannot achieve high FPS (Frames Per Second), or even interactive performance. To improve above photon relaxation methods, optimization of photon map denoising algorithm [12] is presented to facilitate the discovery of novel parameter spaces which can be used to dissociate complex caustic illumination. However, these implementations are somewhat complicated and they do not always achieve easy implementation on GPU. More recently, based on reduction of the overhead associated with re-computation of photon maps, Jonsson and Ynnerman [13] present a method for interactive global illumination of both static and time-varying volumetric data. The utility of the method is demonstrated in several examples which show that visual quality can be retained when the fraction of retraced photons is as low as 40%-50%. Silvennoinen and Lehtinen [14] factorize the direct-to-indirect transport operator into global and local parts by pre-computed local reconstruction from sparse radiance probes. They obtain indirect illumination on glossy surfaces, and approximate global illumination in scenes with large-scale dynamic geometry. To improve the overall performance to meet the high requirement of rendering speed, the line space theory [15] is used as an acceleration data structure in different variations, resulting in better empty space skipping. Using the line space, path tracing can be performed mostly independent of the complexity of the scene geometry with over 70 frames per second. Nowadays, the simulation of global illumination for real-time interactive Virtual Reality (VR) applications is a challenging process. The main reason for this is the high computational complexity of the Monte Carlo integration which makes sufficient frame rates hard to achieve [16]. It is even more challenging to adopt this process for large, high-resolution displays, because the resolution of an image to be generated becomes huge compared to a single display. So some faster rendering approaches to realize interactions and behaviors in VR [17, 18] have been presented to enable high-quality realistic image generation.

Physically-based rendering and high frames speed requirement are increasingly used in a wide field of interactive applications. By that, modern GPU graphic engines can provide a realistic output using physical correct values instead of an analytical approximation. Linear operator theory [19, 20] has also arisen in a range of applications of mathematical physics and can be exploited recently to approximate complex analytical integral equation which is hardly evaluated by computer [15].

With the increasing requirements of high frames speed in interactive applications such as VR and 3D real-time games, the main challenge of rendering is not only to approximate the complex light interaction to give a global rendering such as caustics, specular reflections and soft shadow, but also develop sufficiently efficient method to allow for high FPS and easy to be implemented for practical applications. So in this paper, we present fast global rendering via linear integral operators based on Fredholm theorems. Rendering equation of global illumination via BSSRDF (Bidirectional Scattering Surface Reflectance Distributed Function) is cumbersome as a result of complex integral rendering equation. Using linear integral operators based on Fredholm theorems, we define an operator that takes a radiance function and integrates its product with the BRDF (Bidirectional Reflectance Distributed Function). With the notational simplification, it becomes straightforward to solve the rendering equation, which is very practical. The results show that realistic effects of global illumination can be achieved while keeping high rendering frames speed.

The rest of this paper is organized as follows. The rendering equation of global illumination is given in Section 2. Solution of rendering equation by linear operator theory is deduced in Section 3. At last, experimental results are discussed in Section 4, and finally conclusions are given in Section 5.

2. **Rendering Equation of Global Illumination.** Global illumination is the simulation of physically based light transport. It is a means to simulate how light is transferred between surfaces in your 3D scene and it will greatly increase the realism of your scene. Not only that, it is also a means to convey mood and with clever use can be used to improve the rendering applications. GI algorithms take account of not only the light which comes directly from a light source (the direct illumination), but also subsequent cases in which this light is reflected by surfaces in the scene using different materials (indirect illumination).

Figure 1 shows two examples of illumination types that we distinguish for points that are visible from the camera. The solid line path denotes the direct illumination, since it leads directly from the light source, bouncing off the surface and then to the camera. The dotted line path adds one more light bounce. The surface is then lit indirectly. We call global illumination algorithms combining both direct and indirect illumination.

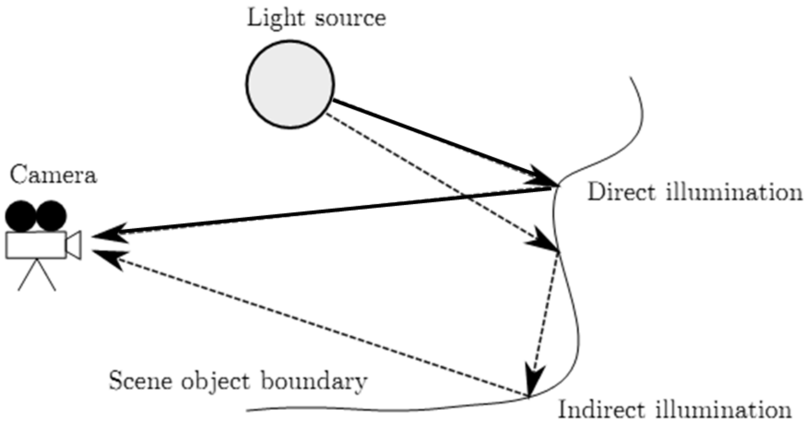

FIGURE 1. Direct illumination (solid line) and indirect illumination (dotted line)

Rendering equation of global illumination is used for obtaining amount of radiation, leaving a certain point in every direction. Outgoing radiation L_o is a sum of emitted radiation (special properties of the material) L_e and the reflected radiation L_r as follows.

$$L_o(x, \overrightarrow{\omega}) = L_e(x, \overrightarrow{\omega}) + L_r(x, \overrightarrow{\omega}) \tag{1}$$

According to real process of light reflection, the reflected radiation L_r can be calculated as:

$$L_r(x, \overrightarrow{\omega}') = \int_\Omega f_r(x, \overrightarrow{\omega}', \overrightarrow{\omega}) dE_i(x, \overrightarrow{\omega}') = \int_\Omega f_r(x, \overrightarrow{\omega}', \overrightarrow{\omega}) L_i(x, \overrightarrow{\omega}')(\overrightarrow{\omega}' \cdot \overrightarrow{n}) d\overrightarrow{\omega}' \tag{2}$$

where f_r is BRDF function which defines a connection between reflected radiation and irradiation. L_i is the incoming radiance in incoming direction $\overrightarrow{\omega}'$ at the point x.

Considering Equation (2), we can rewrite Equation (1) into the following fundamental equation called BSSRDF equation:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_\Omega L_i(x, \omega_i) f(x, \omega_i, \omega_o) \cos\theta_i d\omega_i \tag{3}$$

where ω_o is the exiting direction. f is the BRDF coefficients at point x. L_i is the incoming radiance in incoming direction ω_i. θ_i is the angle between ω_i and normal direction at point x.

In a practical setting, integrating over the visible hemisphere is sometimes cumbersome or insufficient. It is much more convenient to integrate over all visible surfaces in the scene. To make this change of variables, we must first derive the equivalent form of $d\omega_i$ as shown in Figure 2, where dA' is a micro-area around point x. θ_o is the angle between exiting direction and normal direction.

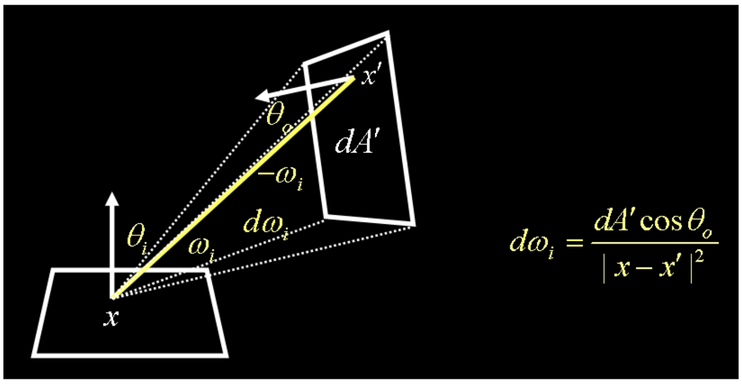

FIGURE 2. Deriving the surface parameterization of the rendering equation

As a result, we could rewrite the rendering Equation (3) in a more approachable form:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_{VP} L_i(x, \omega_i) f(x, \omega_i, \omega_o) \frac{\cos\theta_i \cos\theta_o}{\|x - x'\|^2} dA' \tag{4}$$

where VP is the set containing all visible point x'. dA' is the micro-area surrounding the exiting point.

As the domain of the integral has become somewhat awkward, the rendering equation is still cumbersome. It seems impenetrable as we have currently written it. For simplification and implementation, we can make it an integral over all surfaces in the scene if we simply introduce a binary visibility function V:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_{AP} L_i(x, \omega_i) f(x, \omega_i, \omega_o) G(x, x') V(x, x') dA' \tag{5}$$

where AP is the set of all points. The geometric factor G and the visibility function V can be defined as follows:

$$G(x, x') = G(x', x) = \frac{\cos\theta_i \cos\theta_o}{\|x - x'\|^2} \tag{6}$$

$$V(x, x') = \begin{cases} 1, & \text{if } x \text{ and } x' \text{ are mutually visible;} \\ 0, & \text{in other case.} \end{cases} \tag{7}$$

In Equation (5), we now have, modulo minor notational differences, the canonical form of the rendering equation. Hence, the linear operator theory is applied to making things more manageable. In Equation (5) above, L_e and f are known, and G and V can be precomputed. The crux of the problem lies with the fact that the unknown quantity L_o is on the left-hand side of the equation and inside the integral. Fortunately for us, this type of problem has already been the subject of extensive study. In the mathematics community, it is known as a *Fredholm Integral Equation* of the second kind and has the following canonical form:

$$l(u) = e(u) + \int l(v)K(u,v)dv \tag{8}$$

where l is the unknown, e is known, and K is a continuous kernel function of the integral equation in Fredholm space.

3. Solution of Rendering Equation by Linear Operator Theory.

3.1. The Fredholm theory in Banach space.

Theorem 3.1. *Let A be a completely continuous operator which maps a Banach space E into itself. If the equation $y = x - Ax$ is solvable for arbitrary y, then the equation $x - Ax = 0$ has no solutions distinct from zero solution.*

Proof: Assume the contrary: there exists an element $x \neq 0$ such that $x_1 - Ax_1 = Kx_1 = 0$. Those elements x for which $Kx = 0$ form a linear subspace E_1 of the Banach space E. We denote by E_n the subspace of E consisting of elements x for which the powers $K^n x = 0$.

It is clear that subspaces E_n form a nondecreasing subsequence

$$E_1 \subseteq E_2 \subseteq \cdots \subseteq E_n \subseteq \cdots \tag{9}$$

We shall show that the equality sign cannot hold at any point of this chain of inclusions. In fact, since $x \neq 0$ and equation $y = Kx$ is solvable for arbitrary y, we can find a sequence of elements $x_1, x_2, \ldots, x_n, \ldots$ distinct from zero such that

$$\begin{gathered} Kx_2 = x_1, \\ Kx_3 = x_2, \\ \ldots, \\ Kx_n = x_{n-1}, \\ \ldots \end{gathered} \tag{10}$$

The element x_n belongs to subspace E_n for each n but does not belong to subspace E_{n-1}. In fact,

$$K^n x_n = K^{n-1}x_{n-1} = \cdots = Kx_1 = 0 \tag{11}$$

but

$$K^{n-1}x_n = K^{n-2}x_{n-1} = \cdots = Kx_2 = x_1 \neq 0 \tag{12}$$

All the subspaces E_n are linear and closed; therefore, for arbitrary n there exists an element $y_{n+1} \in E_{n+1}$ such that

$$\| y_{n+1} \| = 1 \text{ and } \rho(y_{n+1}, E_n) \geq \frac{1}{2} \tag{13}$$

where $\rho(y_{n+1}, E_n)$ denotes the distance from y_{n+1} to the space E_n:

$$\rho(y_{n+1}, E_n) = inf\{\| y_{n+1} - x \|, x \in E_n\} \tag{14}$$

Considering the sequence $\{Ay_k\}$, we have (assuming $p > q$)

$$\|Ay_p - Ay_q\| = \|y_p - (y_q + Ky_p - Ky_q)\| \geq \frac{1}{2} \tag{15}$$

since $y_q + Ky_p - Ky_q \in E_{p-1}$. It is clear from this that the sequence Ay_k cannot contain any convergent subsequence which contradicts the complete continuity of the operator A. This contradiction proves Theorem 3.1.

Theorem 3.2. *In order to make equation $y = x - Ax$ solved easily, a necessary and sufficient condition is that $f(y) = 0$ for all f which $f - A^*f = 0$.*

Proof 1: If we assume that the equation $y = x - Ax$ is solvable, then

$$f(y) = f(x) - f(Ax) = f(y) = f(x) - A^*f(x) \tag{16}$$

i.e., $f(y) = 0$ for all f for which $f - A^*f = 0$.

Proof 2: Now assume that $f(y) = 0$ for all f for which $f - A^*f = 0$. For each of these functions we consider the set L_f of elements for which f takes on the value zero. Then our assertion is equivalent to the fact that the set $\bigcap L_f$ consists only of the elements of the form $x - Ax$. Thus, it is necessary to prove that an element y_1 which cannot be represented in the form $x - Ax$ and cannot be contained in $\bigcap L_f$. To do this we shall show that for such an element y_1 we can construct a function f_1 satisfying

$$f_1(y_1) \neq 0, \quad f_1 - A^*f_1 = 0 \tag{17}$$

These conditions are equivalent to the following:

$$f_1(y_1) \neq 0, \quad f_1(x - Ax) = 0, \quad \forall x \tag{18}$$

In fact,

$$(f_1 - A^*f_1)(x) = f_1(x) - A^*f_1(x) = f_1(x) - f_1(Ax) = f_1(x - Ax) \tag{19}$$

Let G_0 be a subspace consisting of all elements of the form $x - Ax$. We consider the subspace $\{G_0, y_1\}$ of the elements of the form $z + \alpha y_1$, $z \in G_0$ and define the linear function by setting

$$f_1(z + \alpha y_1) = \alpha \tag{20}$$

Extending the linear function to the whole space E (which is possible by virtue of the Hahn-Banach theorem) we do obtain a linear function satisfying the required conditions. This completes the proof of Theorem 3.2.

Theorem 3.3. *A necessary and sufficient condition that the equation $f - A^*f = h$ is solvable is that $h(x) = 0$ for all x for which $x - Ax = 0$.*

Proof 1: If we assume that the equation $f - A^*f = h$ is solvable, then

$$h(x) = f(x) - A^*f(x) = f(x - Ax) \tag{21}$$

i.e., $h(x) = 0$ if $x - Ax = 0$.

Proof 2: Now assuming that $h(x) = 0$ for all x such that $x - Ax = 0$, we shall show that the equation $f - A^*f = h$ is solvable. Considering the set F of all elements of the form $y = x - Ax$, we construct a function f on F by setting

$$f(Kx) = h(x) \tag{22}$$

This equation defines in fact a linear function. Its value is defined uniquely for each y because if $Kx_1 = Kx_2$ then $h(x_1) = h(x_2)$. It is easy to verify the linearity of the function and extend it to the whole space E. We obtain

$$f(Kx) = K^*f(x) = h(x) \tag{23}$$

i.e., the function is a solution of equation $f - A^*f = h$. This completes the proof of Theorem 3.3.

Theorem 3.4. *If the equation $x - Ax = 0$ has $x = 0$ for its only solution, then the equation $x - Ax = y$ has a solution for all y.*

Proof: If the equation $x - Ax = 0$ has only one solution, then by virtue of the corollary to the preceding theorem, the equation $f - A^*f = h$ is solvable for all h. Then, by Theorem 3.1, the equation $f - A^*f = 0$ has $f = 0$ for its only solution. Hence, corollary to Theorem 3.2 implies that the equation $x - Ax = y$ has a solution for all y.

All the results obtained for the equation $x - Ax = y$ remain valid for the equation $x - \lambda Ax = y$ (the adjoint equation is $f - \bar{\lambda}A^*f = h$). It follows that either the operator $I - \lambda A$ has an inverse or the number $1/\lambda$ is a characteristic value for the operator A. In other words, in the case of a completely continuous operator, an arbitrary number is either a regular point or a characteristic value. Thus, a completely continuous operator has only a point spectrum.

3.2. **Linear integral operators based on Fredholm theorems.** A linear operator acts on functions the way a matrix acts on vectors. In fact, one can think of a real-valued function as an infinite-dimensional vector where each "element" gives the value of the function when it is evaluated at a particular point.

$$h(u) = (T \circ f)(u) \tag{24}$$

where T is a linear operator by discretizing the continuous kernel function K. h and f are functions of the variable u. $\circ$ is the linear operation like matrices act on vectors, and one example of linear operations is the integration process of linear operator and one function in Equation (25).

Many common operations on functions can be expressed as linear operators, including differentiation and integration:

$$(T \circ f)(u) = \int T(u, v) f(v) dv \tag{25}$$

We also define the identity operator I, analogous to the identity matrix in the discrete case, to be the operator that takes every function to itself. In other words, $I \circ f = f$ for all functions f.

3.3. **Discretization solution in linear operator form.** In general, there is no analytical solution for *Fredholm Equation* of the second kind. However, we can write down a formal solution. In order to do that, we must rewrite the equation using an *integral operator*. Let us define a *reflection operator* T as an operator that takes a radiance function and integrates its product with the BRDF everywhere in the scene:

$$(T \circ L_o)(x, \omega_o) = \int_V L_i(x, \omega_i) f(x, \omega_i, \omega_o) GV dA' \tag{26}$$

This is a linear operator, whose physical interpretation is to reflect the light in the scene once off all the surfaces. Using operator notation, the rendering equation can be rewritten as:

$$L = L_e + TL \tag{27}$$

Here, T is the operator representing integration against the kernel (which in our case is the BRDF f modulated by a cosine factor). When reflected light L_o is not the function evaluated at a particular location and direction, L represents L_o (reflected light) in Equation (26). So L is the unknown reflected radiance at each measured surface point and outgoing angle. If discretization, L_e becomes the vector of known light sources. T is typically called the *Light Transport Matrix*, which characterizes the reflectance of light around the scene.

With this notational simplification in place, it becomes straightforward to solve the rendering equation:

$$IL - TL = L_e \Rightarrow (I - T)L = L_e \Rightarrow L = (I - T)^{-1} L_e \tag{28}$$

Finally, the rendering equation has a straightforward solution which can be easily implemented on GPU:

$$\begin{aligned} L &= (I - T)^{-1} L_e = (I + T + T^2 + T^3 + \cdots) L_e \\ &= L_e + TL_e + T^2 L_e + T^3 L_e + \cdots = \sum_{n=0}^{\infty} T^n L_e \end{aligned} \tag{29}$$

This result, in addition to its simplicity, realizes a parallel hierarchical rendering and has an amazingly intuitive physical explanation. The first term, L_e, gives the emission directly from the light sources. The next term, TL_e, describes the direct illumination on surfaces. The following term, $T^2 L_e$, represents one-bounce indirect lighting. This sequence continues indefinitely, with the $T^i L_e$ $(i = 3, \ldots, n)$ term representing i-bounce indirect lighting. If we sum up infinitely many terms in this sequence, the sum converges to the exact solution of the rendering equation.

4. **Experiment and Results.** Based on GPU platform and CUDA acceleration, we have implemented interactive global rendering via linear integral operators based on Fredholm theorems. Our final rendering is tested on a computer equipped with experiment on Intel i7-6700K@4Ghz, 16GB DDR3@1800Mhz and Nvidia GeForce GTX 780 Ti 3GB with 2880 CUDA Cores. We have realized our method using CUDA acceleration and C++ programming in Visual Studio 2016. Rendering results of global illumination are achieved at a high FPS to meet the requirement of VR and 3D games, which clearly shows the effective implementation of our technique.

Global illumination is simply the process by which light interacts with physical surfaces such as surfaces absorbtion, reflect or transmit light. This here is a famous scene of the Cornell box containing some glass spheres, which is a standard environment for testing out global illumination algorithms. The only light source in this image is a spotlight in the ceiling of the box. Global illumination adds soft shadows and color between the cubes and the walls. There is also natural darkening along the edges of the walls, floor and ceiling. Two scenes as shown in Figure 3 are rendered by our method. The red and

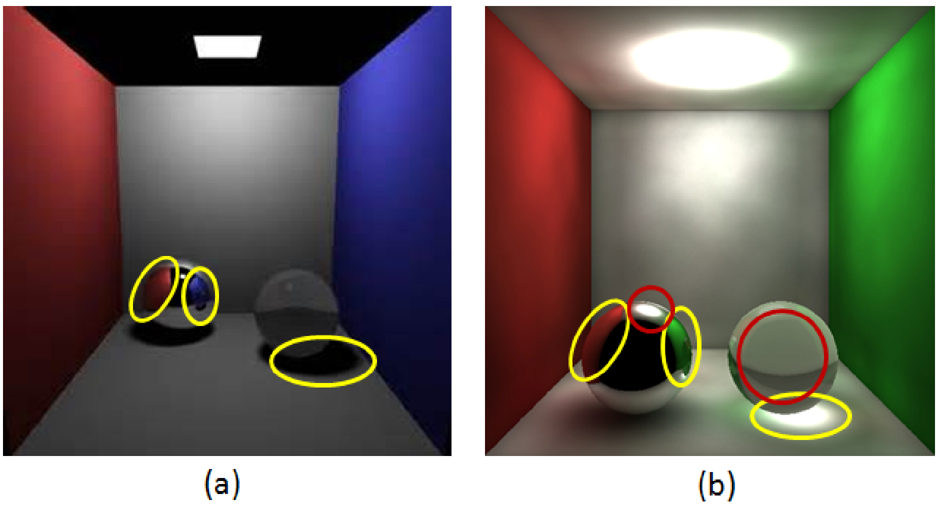

FIGURE 3. Glass spheres in Cornell box under global illumination

yellow circles are used to highlight effects of global illumination such as reflection, caustics and soft shadow.

Then we render two dragon models with different materials such as glass and marble in another two Cornell boxes as shown in Figure 4 and Figure 5, and focus on rendering effects of global illumination with yellow circles.

From Figure 3 to Figure 5, we can observe that effects of global illumination such as inter-reflection, refraction, caustics, luster and soft shadow, can be acquired efficiently, which is in accord with physical phenomena. Hence, good quality of rendering can be enhanced and sense of reality can be acquired via our method.

Then, using the occlusion method for GI [21] and our method, we respectively render two faces of virtual character in the same scene as shown in Figure 6 and Figure 7, and give a comparison of rendering appearances of global illumination. The global illumination in the scene is represented by Uffizi Gallery environment map (Pauldebevec LPIG) [22], and environment lighting is projected to the appearance of human face.

From Figure 6, it shows the virtual character is separated from the virtual environment and cannot be naturally integrated into the surroundings for lack of good simulation of global illumination. Figure 7 emphasizes the realistic rendering appearance using our

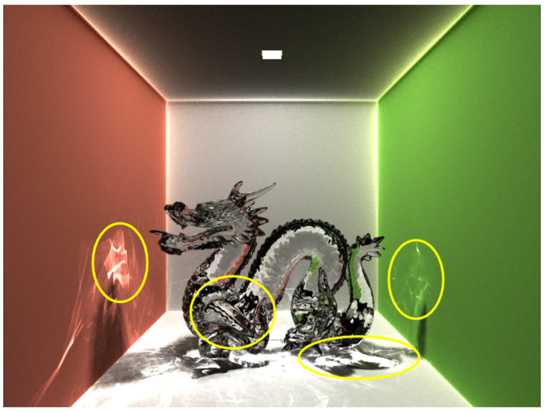

FIGURE 4. Global illumination for glass dragon model

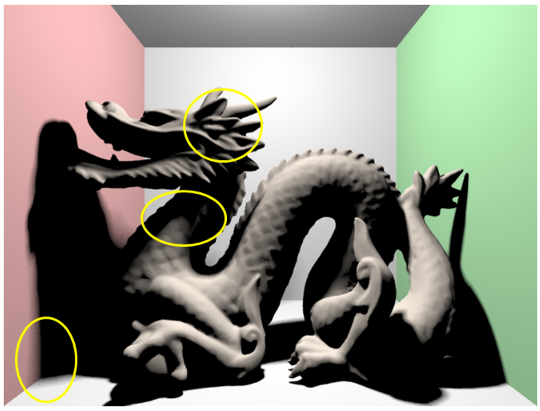

FIGURE 5. Global illumination for marble dragon model

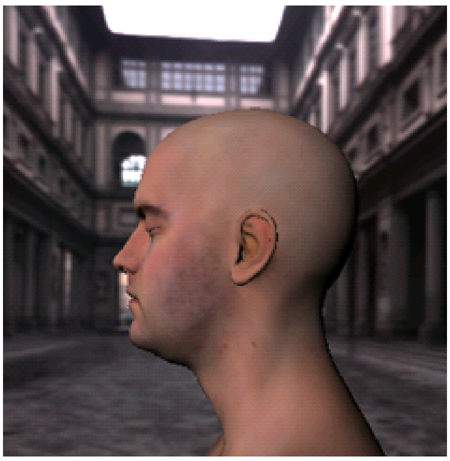

FIGURE 6. Effects of Global rendering for virtual character using the literature method [21]

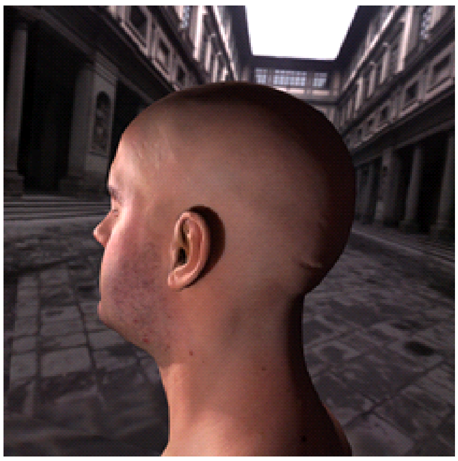

FIGURE 7. Effects of Global rendering for virtual character using our method

method, and achieves much more realistic rendering effects of global illumination such as color bleeding, soft shadow and reflection. Since our methods can make the character integrated into the virtual scene naturally via different effects of global illumination, which is valuable practically in the creation of virtual character in the virtual reality, 3D games and film-making development.

Further, to give an objective evaluation of simulation of the virtual character under global illumination, we use FPS as performance parameter which is evaluated based on different mesh quantity of rendering scene. The comparison between the referential method [21] and our method is shown in Figure 8.

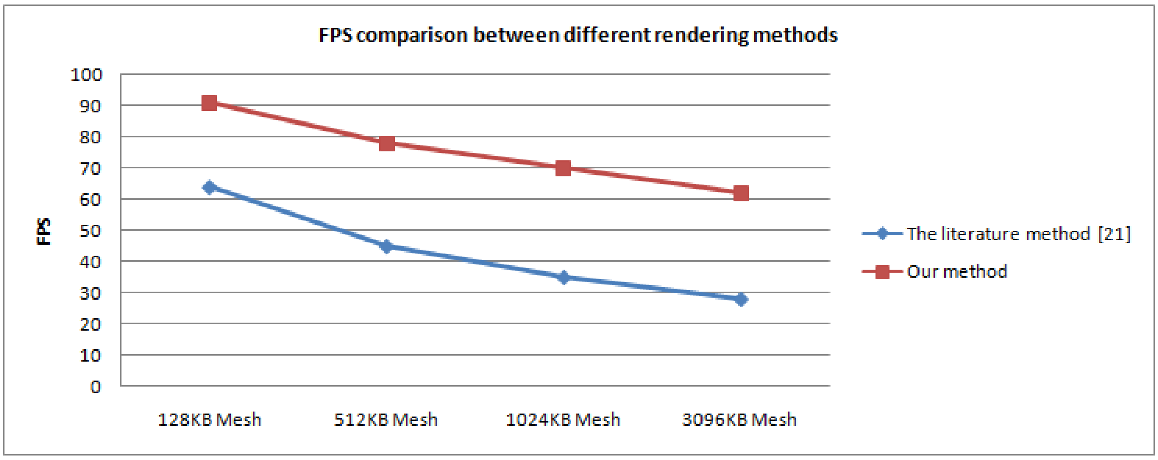

FIGURE 8. FPS comparison between two different rendering methods

From Figure 8, we can obtain a relatively higher FPS while producing highly plausible images. Especially, when increasing mesh quantity, rendering speed of our method is more obvious over the referential method [21]. This is because the computational efficiency of our method is higher when dealing with large scene. When dealing with different mesh quantity 128KB, 512KB, 1024KB and 3096KB, we can obtain rendering speedup ratio 1.42, 1.73, 2, and 2.21 respectively. So the speedup performance of our method will get better and better when mesh quantity of rendering scene is larger.

5. **Conclusions.** We have implemented fast global rendering via linear integral operators based on Fredholm theorems. Using linear integral operators based on Fredholm theorems, we define an operator as transformation formulation and acquire the straightforward solution of the complex rendering equation. The results show that realistic effects of global illumination such as inter-reflection, refraction, caustics, luster, color bleeding and soft shadow, can be acquired efficiently, which is in accord with physical phenomena. Also we can obtain a better acceleration rate of rendering when increasing triangles size of virtual character in the large scene. Solving the rendering equation analytically proves intractably difficult. Instead, approximation solution via linear integral operators is presented here. Besides its simplicity, we realize a parallel hierarchical rendering and have an amazingly intuitive physical explanation. Our method has been used in practice for rendering character. We hope that our method may help game developers to improve existing databases at low cost, which is a good starting point if one wishes to extend the proposed framework to commercial applications such as virtual reality and augmented reality in game development.

Some special applications need interactive global illumination for fully dynamic scenes, so how to compute and visualize global illumination effects in dynamic scenes at interactive rate is a direction which we are working on.

Acknowledgment. This work was supported by the National Natural Science Foundation of China under Grant No. 61180173, and Science and Technology Foundation of Henan Province Education Department of China (No. 140065).

REFERENCES

[1] R. Tamstorf and H. W. Jensen, Adaptive sampling and bias estimation in path tracing, *Proc. of the Eurographics Workshop on Rendering Techniques*, pp.285-296, 1997.

[2] I. Wald, P. Slusallek, C. Benthin et al., Interactive rendering with coherent ray tracing, *Computer Graphics Forum*, vol.20, no.3, pp.153-165, 2001.

[3] H. W. Jensen, *Realistic Image Synthesis Using Photon Mapping*, A. K. Peters, Ltd., Natick, MA, USA, 2001.

[4] H. W. Jensen, A practical guide to global illumination using ray tracing and photon mapping, *ACM SIGGRAPH 2004 Course Notes*, no.5, pp.21-30, 2004.

[5] V. Govindaraju, P. Djeu, K. Sankaralingam, M. Vernon and W. R. Mark, Toward a multicore architecture for real-time ray-tracing, *The 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pp.176-187, 2008.

[6] S. Laine, T. Aila, U. Assarsson et al., Soft shadow volumes for ray tracing, *ACM Trans. Graphics*, vol.24, no.3, pp.1156-1165, 2010.

[7] T. Hachisuka and H. W. Jensen, Parallel progressive photon mapping on GPUs, *ACM SIGGRAPH ASIA Sketches*, 2010.

[8] A. S. Kaplanyan and C. Dachsbacher, Adaptive progressive photon mapping, *ACM Trans. Graphics*, vol.32, no.2, pp.1-13, 2013.

[9] Y. Zhang, Z. Dong and K.-L. Ma, Real-time volume rendering in dynamic lighting environments using precomputed photon mapping, *IEEE Trans. Visualization and Computer Graphics*, vol.19, no.8, pp.1317-1330, 2013.

[10] V. A. Frolov, A. A. Kharlamov, V. A. Galaktionov et al., Multiple reference octrees for a GPU photon mapping and irradiance caching, *Programming and Computer Software*, vol.40, no.4, pp.208-214, 2014.

[11] V. Frolov, K. Vostryakov, A. Kharlamov et al., Implementing irradiance cache in a GPU realistic renderer, *Trans. Computational Science XIX*, pp.17-32, 2013.

[12] B. Spencer, M. W. Jones and I. S. Lim, A visualization tool used to develop new photon mapping techniques, *Computer Graphics Forum*, vol.34, no.1, pp.127-140, 2015.

[13] D. Jonsson and A. Ynnerman, Correlated photon mapping for interactive global illumination of time-varying volumetric data, *IEEE Trans. Visualization & Computer Graphics*, vol.23, no.1, pp.901-910, 2016.

[14] A. Silvennoinen and J. Lehtinen, Real-time global illumination by precomputed local reconstruction from sparse radiance probes, *ACM Trans. Graphics*, vol.36, no.6, pp.1-13, 2017.

[15] F.-L. Schröder, *Global Illumination Using GPU Path Tracing and the Line Space Data Structure*, Bachelor Thesis, Universität Koblenz-Landau, 2017.

[16] A. Sigitov, T. Roth and A. Hinkenjann, Enabling global illumination rendering on large, high-resolution displays, *Software Engineering and Architectures for Realtime Interactive Systems*, pp.18-24, 2017.

[17] D. Bischoff, T. Schwandt and W. Broll, A real-time global illumination approach for high resolution reflective shadow maps in open world scenes, *International Conference on Computer Graphics Theory and Applications*, pp.116-126, 2017.

[18] D. Bonatto, S. Rogge, A. Schenkel et al., Explorations for real-time point cloud rendering of natural scenes in virtual reality, *IEEE International Conference on 3D Imaging*, pp.1-7, 2017.

[19] C. Lederman, R. Martin and J. L. Cambier, Time-parallel solutions to differential equations via functional optimization, *Computational & Applied Mathematics*, pp.1-25, 2016.

[20] M. Hajarian, Least squares solution of the linear operator equation, *Journal of Optimization Theory & Applications*, vol.170, no.1, pp.205-219, 2016.

[21] L. I. Hua and H. M. Yang, An occlusion method for approximate global illumination, *The International Conference on Computer Science and Technology*, pp.1250-1256, 2017.

[22] P. Debevec, *Light Probe Image Gallery*, http://www.pauldebevec.com/Probes/.